Master Thesis

im Rahmen des
Universitätslehrganges „Geographical Information Science & Systems“
(UNIGIS MSc) am Interfakultären Fachbereich für GeoInformatik (Z_GIS)
der Paris Lodron-Universität Salzburg

zum Thema

„BIM and construction process data in mechanized tunnel construction“

Milestone control for tunnel construction sites using automatically created process data in comparison with 4D BIM

vorgelegt von

B.Eng. Robert Lensing
103255, UNIGIS MSc Jahrgang 2013

Zur Erlangung des Grades

„Master of Science (Geographical Information Science & Systems) – MSc(GIS)”

Heidelberg, 20.12.2016

Foreword

This work was created during my work as a project engineer for TBM navigation systems and construction information systems working for VMT GmbH. I thank my employer for giving me some time in the time of thesis writing, even in busy working times.

I'd like to thank Dr.-Ing. Tobias Rahm for developing the idea of thesis topic with me and Dr.-Ing. Felix Hegemann for providing me with the essentials to be able to create the software, developed within this work, especially the source code of the IFC Tools Project and his IFC file of a tunnel project, originating from the SFB 837 research initiative of the Ruhr University Bochum.

Thank you very much for the researchers active in the topic of BIM in tunnel construction, which publications were inspiring and helped a lot to create this work.

Thank you also to my girlfriend Jitske Westra for all the encouragement and having so much sympathy and understanding for me having so little time lately.

Abstract

This work is a cross-section of topics. It involves Building Information Modeling (BIM) and its use underground in mechanized tunnel construction as well as sensor data and data sources of the machines used to create concrete lined tunnel buildings. After a general introduction, a literature review focusses on the current use of BIM in tunnel construction.

BIM goes along with sharing data across users and platforms using open file standards. As the de facto-standard for open source file standards the Industry Foundation Classes (IFC) are introduced. Yet, IFC is not yet fully adapted for use in underground construction. Hence, research initiatives, which focused on the adaption of IFC for use in tunnel construction were investigated and presented: IfcTunnel and IfcShieldTunnel, individually driven by researchers in Japan and Germany.

Construction surveillance nowadays goes hand in hand with the usage of information systems, which present near real-time information online. Examples of these systems are presented.

The dynamic use of available data sources of various kind, resulting from the construction process, in combination with planned model data from BIM needs the constant dilution of BI models (design data) and process data (as-built data and more).

This work was continued examining a practical use case: the creation of a software tool to match tunneling process data with a BI Model, using an example tunnel model on one hand and construction meta data on the other hand, from the TBM navigation system, the tunnel ring installation survey and a segment management software.

Future use cases for such interaction of construction process data and BI models are discussed, but also the problems faced during this work: the non-available specialization and thus the need for designated IFC extensions specifically for tunnel construction, with an outlook to IFC5, which is going to be focused on infrastructure.

Table of content

Abbreviations and symbols

AP .....Application Protocol

API .....Application Programming Interface

BEP.....BIM Execution Plan

BCCM .....Building Construction Core Model

BCF.....BIM Collaboration Format

BIM.....Building Information Modeling

BMVI Bundesministerium für Verkehr und digitale Infrastruktur (Federal Ministry for Traffic and digital Infrastructure)

GIS.....Geographical Information System

IDM .....Information Delivery Manual

IFC .....Industry Foundation Classes

IFD .....International Framework for Dictionaries

IRIS .....Integrated Risk and Information System

ISO.....International Standardization Organization

HOAI Honorarordnung für Architekten und Ingenieure (Fee Structure for Architects and Engineers)

KPI .....Key Performance Indicator

LOD.....Level Of Detail

LOI .....Level Of Information

REST .....REpresentational State Transfer

STEP.....Standard for the Exchange of Product Model Data

TBM .....Tunnel Boring Maschine

UML .....Universal Markup Language

VDI.....Verband Deutscher Ingenieure (Union of German Engineers)

List of figures

Figure **Page Nr**

List of tables

Table **Page Nr.**

1 Introduction

The typical procedure for the construction of a civil engineering constructive work is divided into a diverse number of execution phases. Taking the German Fee Structure for Architects and Engineers (HOAI) as an example, 9 phases are distinguished. Simplistic those can be abstracted as Design, Procure, Construction and Operation. The construction industry is a fragmented market, in which lots of jobs during planning and execution are managed by specialized Engineers and companies. Lots of participants are just part of the team during one or some phases, very few of them in all phases of the construction. Cooperation of all parties must be ensured by a rigid project controlling and site supervision.

That is why during the work with planning information and project knowledge (which may have to be transferred into new specialized software products again and again) severe information breaks can occur. That may happen especially after reaching a mile stone and with the start of a new project phase with new participants, as depicted in Figure 1. Thus, the risk for mistakes during planning or construction work is high.

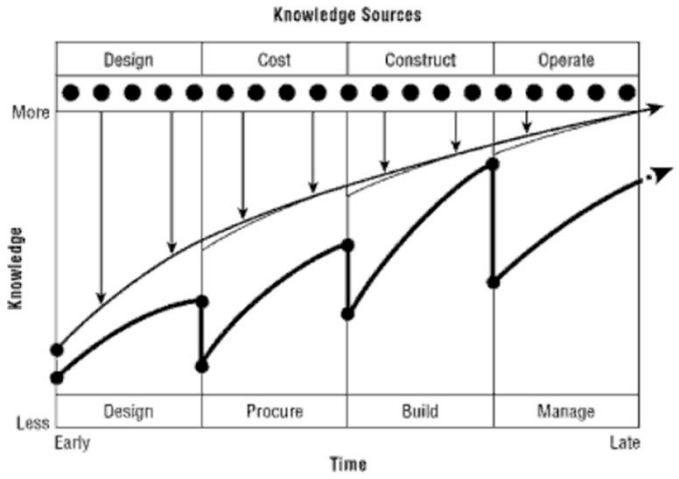


Figure 1: *Information transfer between parties* (Source: (Eastman, 2011))

One approach for the fully digital planning without media discontinuity for the dissemination of information is Building Information Modeling (BIM): The involved partners plan and work on a shared digital model, sometimes using a client-server structure, centralized parts libraries and more. BIM has been in discussion for years and is slowly adapted to practice in surface building works.

Yet, buildings are oftentimes not built as planned. The knowledge about the results of the construction works are necessary during construction for supervision and for the following operation phase. That is why in a BIM point-of-view it is necessary to merge the model of design with construction information to generate an as-built model.

For civil engineering underground in general and – in context of this thesis – for tunnel construction using tunnel boring machines there is no standardization on BIM rules nor regulations up to date. After a short introduction into key figures of the thesis (tunnel construction, BIM and tunnel construction information systems) and the state of affairs for BIM in tunnel construction, IFC add-on proposals, published by universities and designed for the use in tunnel construction are presented. After that the work focuses on the development of a

case-study for the semi-automatic generation of an as-build tunnel model. This is done by merging a design model with construction information. The merging process is realized by an independent software created for this work, which reads a tunnel model as IFC file format, makes a web connection to a third party tunnel construction information system and matches relevant tunnel construction data onto the entities of the segments in the model.

2 Background

The following chapter gives a short introduction into the relevant background for this thesis and is based on a literature review for all of the topics involved. The concept of building information modeling (BIM) is explained, with more information and examples about the use of BIM for tunneling projects. The use of BIM data goes along with necessary strong semantic data modeling, which is explained in the second part, also with reference to tunneling. After that basic concepts of machine based tunnel construction are explained, followed by an introduction to the automated collection of meta data for tunneling projects in tunnel information systems.

This thesis is a cross-cut of the topics building information modeling (BIM), data acquisition and aggregation in the specific context of tunnel construction. It focusses on an approach to update a building model with as-built information. Updated near real-time building information models for instance can be judged necessary in a more general approach of the "digital jobsite", next to e.g. methods of computational engineering, augmented reality, the use of RFID technology and live construction monitoring (König et al., 2015).

2.1 Building Information Modeling

Construction projects can be characterized as being divided into small sections of specialized work, either viewed by project phase or working profession: projects go through the phases of design, procure, execution and operation. All of which may be performed by different actors (e.g. planning engineers, construction companies, the owner) and different participating professions per phase, like support structure engineers, civil engineers, static engineers, survey engineers, building services and more. After completed construction the owner / manager of a building later on will use available documentation, created in earlier phases, in operation to open the bidding process for reconstructions and renovations. Figure 2 shows this concept.

In a classical course of action, the project communication is mainly focused on static CAD-files and drawings. Different parties will then often use this data to manually enter those into specialized software products, to be able to do their work. This is ineffective and prone to errors[1].

Hence there is a growing movement for the implementation of a fully digitized planning and execution procedure, where all parties work on the same data base. Such a model is called "Building Information Model" (BIM). The work with such a model is supported by a growing list of software vendors Wikipedia (2016).

A building information model can inherit all information necessary for the building. Those are geometrical data describing each part. Moreover, the describing product properties of the

[1] A study of the US Department of Commerce (NIST) conducted a study in 2004 already, where one conclusion was that the cost of non-existent interoperability between used software systems during the phases of planning, construction and operation for US companies were summed up to $15,8B for the questioned year of 2002 already. That result was published in (Gallagher et al., 2004)

single installation parts (pipes, doors, …)[2], their spatial position and their relation to one another can be modeled, together with construction time, cost and more.

The work with BIM comes along with a change in the working operations. This is why BIM is much more referred to as a "Method" rather than just a piece of software and consequently creates new job profiles like "BIM Manager" and "BIM Coordinator" (Borrmann et al., 2015, p. 195 - 250).

The use of a model-based approach in working needs detailed planning of responsibilities in the project pre-phase. Several use cases must be designed, answering the question what do the parties want to achieve by using this method, depending on the project phase. Only by answering these questions the necessary level of detail and content for a number of models can be defined. All this is put together in the project-specific "BIM Execution Plan" (BEP). Detailed specifications for information exchange can then be put into "Information Delivery Manuals" (IDM). König et al. (2016b) give an overview about a BIM Workflow for jobsites, also with reference to tunneling sites.

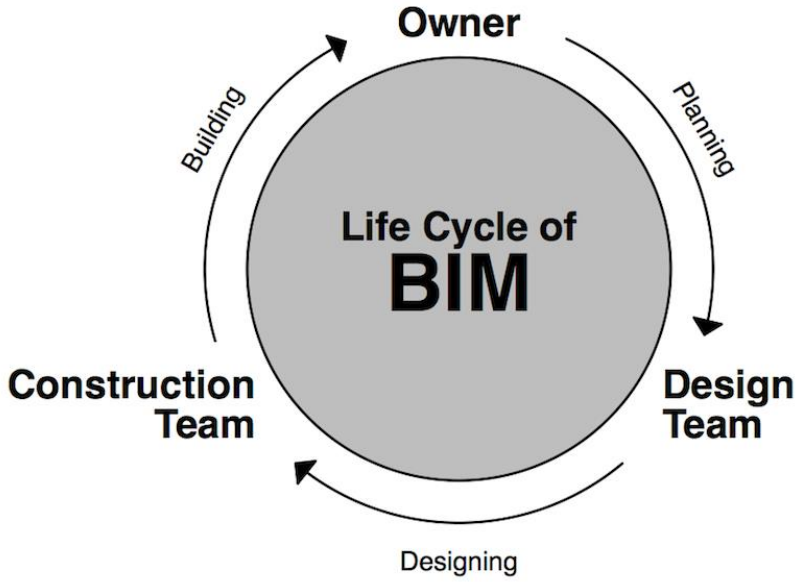


Figure 2: *BIM life cycle*

2.1.1 n-dimensional BIM

BIM as an instrument for planning can basically be multi-dimensional. This approach exceeds the three visual dimensions. This will be explained in the following.

As the planning process is happening in 3D directly, 2D drafts can be created out of those as cuts through the model. The model is enriched with topological neighboring information[3], which

[2] Current international standardization efforts e.g. focus on the creation of standardized parts libraries. Examples for that are the buildingSMART data dictionary "bsdd" (see: http://bsdd.buildingsmart.org/) or the work on ISO 16757 "Product data for building services system models".

[3] This so called "parametric modelling" approach evolved over the years with the maturity of CAD engines (Autodesk, 2005). So 3D for BIM is more than just visualization for spot announcements: 3D also has to fulfill the further requirements for BIM, as described in (Liebich et al., 2011, p. 48), p. 48 ff. (in German). For example: Door X "needs to know" that it belongs to wall Y. When the wall is moved, the door will move with it. „[…] software built on parametric building modeling technology is a powerful tool for thinking about buildings. […] a change made anywhere in a parametric building or BIM model is immediately reflected throughout the model, as described in: (PCI, 2009, p. 2), p. 2

is yet different to those in GIS models, as the ability for topological information in the model is more advanced within GIS.

The fourth dimension for BIM (for construction project management and the bidding procedures) is time: All objects of the model are linked to the schedule of construction. Using time-based animation, the model can be viewed in its development[4]. Figure 3 refers to that. Additionally, in the later phase of actual construction this information may help planning the site, the processes, logistics and more. The fact that this time based information is “normal” for BIM will be used as a matter of discussion later, when enriching a tunnel BIM model.

As a fifth dimension of BIM construction costs can be mentioned. Costs will be added to each group of the model. When running the 4D animation, the cost development of the project is visible, as also shown in Figure 3.

Viewing it more general, a model can be equipped with add-on information, acting as another dimension to the model. This view was also in use for the later creation of an own model for the case study.

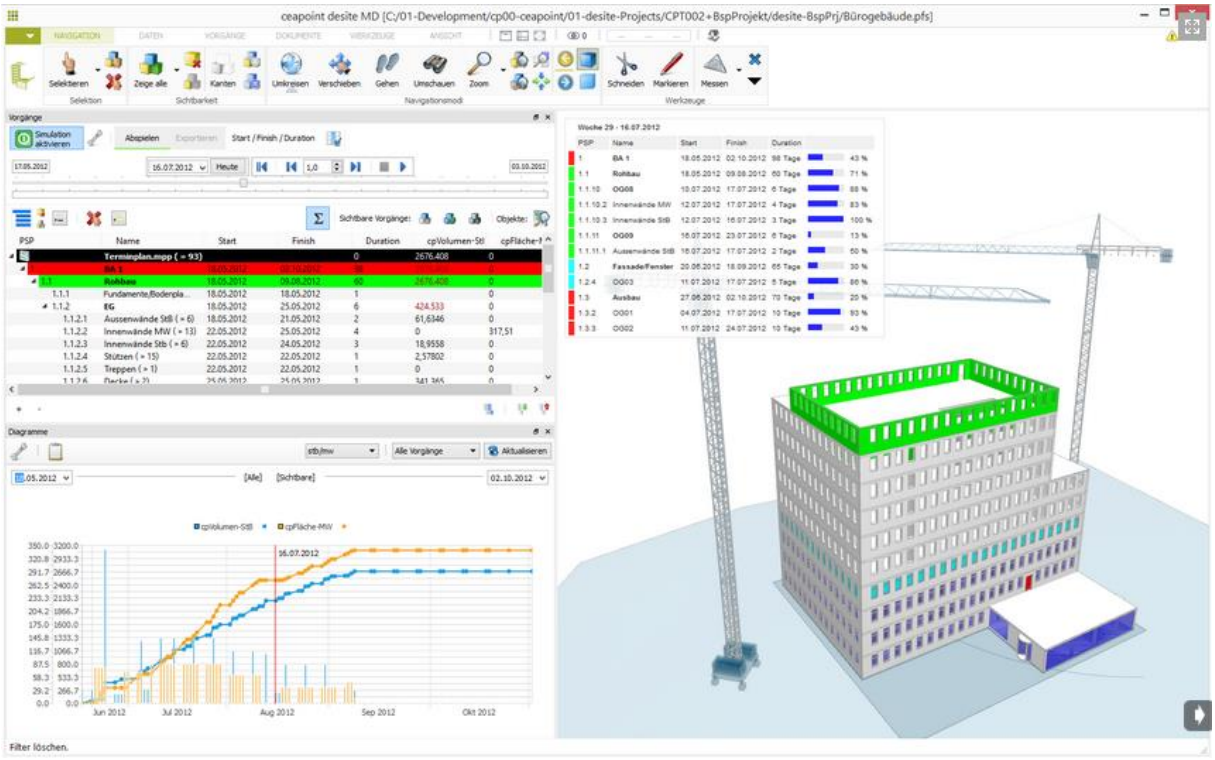


Figure 3: *Exemplary view onto a building information model, enriched with time-based information.Parallel running construction processes and their degree of completion are shown as bars. Cumulative costs are shown in a graph.* Source: ceapoint aec technologies GmbH, (2015), 4D Example [ONLINE]. Available at: http://www.ceapoint.com/index.php?option=com_content&view=article&id=59&Itemid=64 [Accessed 16 December 2016]

[4] This 4D use has multiple positive benefits with it: the generally better understanding of 3D models and visualization for human beings can help numerous stakeholders of the project to better understand it: “Diverse project stakeholders can view a planned construction sequence as a 4D animation, and stakeholders such as users and neighbors can understand it even though they do not understand 2D drawings or Gantt charts, and construction professional consistently find that the 4D animations enable their finding time – space interferences more effectively than they can using traditional drawings and Gantt charts” (Kunz and Fischer, 2009, p. 23), p. 23

2.1.2 BIM state of implementation

Introducing BIM as a work methodology highly depends on regulatory works in countries. Some countries are more progressive than others. Stuhlmacher (2014) gives an overview of the difference in advancement for selected countries.

The German view:

Due to the mentioned fragmentation of the construction industry, a market like this needs a general push to adapt new technologies fully. This could be done quickest by big players in the market demanding adaptation, for bidders to be considered in the tendering phase. For the construction the government is the biggest supplier, mostly in terms of infrastructure projects. While bigger consulting firms and engineering offices have already realized the big potential of BIM as a method (and are consequently pushing it within committees and organizations) first projects are managed with BIM. The government is still evaluating the worth and potentials. Yet, the lack of regulation is understood by both parties, which is why regudlatory push comes from both sides: private and governmental.

Some examples for initiatives coming from private actors: The Union of German Engineers (Verband Deutscher Ingenieure – VDI) set up the "BIM Koordinierungskreis" (BIM Coordination Circle) in 2013 to consult the creation of guidelines, rules and normative frameworks (VDI, 2013). The German Institute for Standardization (Deutsches Institut für Normung – DIN) published the specification DIN SPEC 91400 to link open BIM files with the official standardized descriptions of construction work (STLB-Bau) (DIN, 2015). The DIN furthermore initiated the founding of standards committee "Building Information Modeling", which takes care about the reflection of international BIM standardization into Germany[5]. It is described in König et al. (2016a), p. 54 ff.. The VDI's coordination circle is also working on the regulation VDI 2552 for BIM works in Germany, which will first be published in later 2016 (VDI, 2016).

Some examples for administrative initiatives shall be given in the following: The ministry for traffic and digital infrastructure (Bundesministerium für Verkehr und digitale Infrastruktur – BMVI) published a BIM guideline (Egger et al., 2013), set up a BIM advisory board in 2010[6], and created the "Major Projects Reform Commission", which promoted the extensive use of BIM within use-case studies for major construction projects. The railway tunnel under the city of Rastatt is one of those projects and will be explained in chapter 2.1.3. In late 2015 the graduated plan was announced for all major federal projects to be run by BIM level 1 in 5D from 2020 onwards (BMVI, 2015).

An international view:

BIM as a method is further implemented in other European countries and abroad. Some countries demand the use of BIM within federal construction projects exceeding a certain budget or in general, e.g. Denmark, the Netherlands, Norway and Singapore (Stuhlmacher, 2014). Norway at last demands the use of BIM for all construction projects. This full adaption leads an industry to implement the concept of "BIG BIM", the full building-lifecycle use of Building Information Modelling between all evolved parties. If this process takes place using

[5] The committee NA-005-01-39 AA takes care about the transfer of data models, processes and more into the local market.

[6] Which evaluated, among other things, the changes for the scope of services, compensation regulation for architects and engineers and the necessary form of contracts (BuildingSMART (2010)).

BIM file exchange formats, which specification is documented free and open, it is considered to be “BIG open BIM” with “maturity level 3”[7].

2.1.3 BIM in the tunneling business

In Germany the railway project of the Tunnel Rastatt, a project by the German railway company “Deutsche Bahn” is the first big tunneling project of its kind in Germany, being executed with BIM fully and therefore funded by the German government’s “Major Projects Reform Commission” (also refer to chapter 2.1.2).

For the tunnel construction industry BIM was tried out by some construction companies in other countries already and first experiences have been gained. Given are some examples: The Seattle State Route 99 tunnel project is using BIM (Trimble, 2011). The Swedish Transport Authority “TRAFIKVERKET” used BIM during the Hallandsås tunnel project in Sweden[8] (Smith, 2014). The Crossrail Project Consortium, responsible for the construction of Europe’s biggest infrastructure project, the Crossrail tunnel in London, used BIM for the project coordination[9] (Taylor, 2013). The generated digital model of the tunnel shall be used in operation phase and is targeted to save a higher percentage of maintenance cost (Zeiss, 2016). Also future tunnel projects will focus on BIM more: The High Speed Rail Project 2 in the UK is still in development, but is already one of the biggest infrastructure project to-come in the world. The use of BIM to a certain degree is considered a minimum requirement for all tendering companies.

Milestones of tunnel construction (and other goals to be reached) are added to the design model. “As tunneling progressed, by continuously feeding the ‘as-built’ data from the tunneling production into the 3D model, integrating it with the latest design and engineering information, the team can effectively manage the changing conditions. The shared information empowers the design engineers to analyze the effect of, and react to, changes immediately, avoiding the risk of costly surprises during the construction as a result.” (Smith, 2014) Yet, currently the comparison of those construction results with the as-built (as exemplarily displayed for a tunnel project in Figure 4) still needs manual input. With this thesis, an approach shall be presented to semi-automate this step.

[7] The “BIM Maturity Ramp” defines the steps of adaption to the overall BIM working progress in 4 Levels: Level 0 (working sparsely in 2D without cooperation between parties) to level 3 (full cooperation between parties, working on a shared model repository, using open, standardized file formats, like IFC)

[8] The use was rated profitable: About 3000 geometrical collisions of different trades planning were detected beforehand and could be changed accordingly, saving about £4.5M

[9] About £8.0M were saved with 4D BIM at one of the construction sites “Farringdon Station”

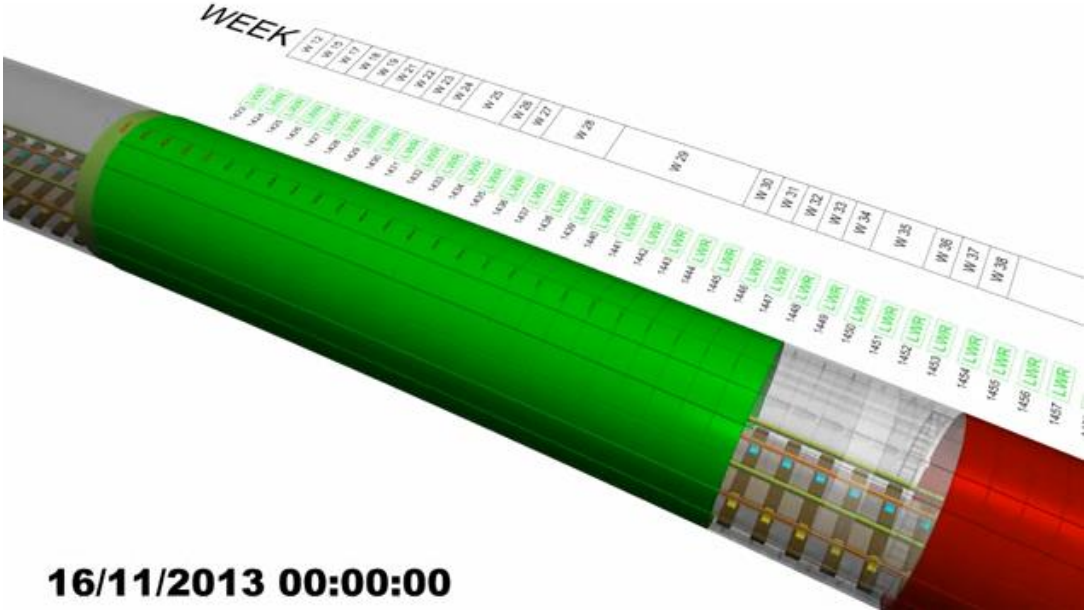


Figure 4: *Visualization of a tunnel construction schedule for demonstration*. Source: Paul Brown, (2013), 4D BIM Tunneling Sequence Schedule [ONLINE]. Available at: http://www.coroflot.com/PaulBrown/4D-BIM-Tunneling-Sequence-Schedule [Accessed 16 December 2016]

2.2 Product modeling for the construction industry

BIM as a method was already described. For the construction industry, BIM also being a design-methodology, it derives from the procedure of so called “product modeling”, the digital description and abstraction of desired products. Completed buildings are the final product of the construction industry. All information of buildings may be structured in a digital manner. The goal is the usage of all information over the whole product lifecycle[10], saved into appropriate file formats. Eastman (1999) gives a broad introduction into the theory of building product models. He furthermore thoroughly explains the STEP file notation, which is the fundamental basis for the IFC file format schema. IFC is used to store building product models in an openly documented file format.

Buildings can last several decades up to centuries. For a usable documentation of the building not being dependent on the existence of specific software, which it was created with and their vendors, the need for a software vendor-neutral open file format is tremendous. Hence, certain organizations push the creation of such files formats. One of those main actors is the “buildingSMART” organization, which file format “Industry Foundation Classes” (IFC) is the industry agreed-upon quasi-standard.

Both of these topics, Product Modeling and IFC will be of necessity for a desired tunnel product model and will hence be explained in the following.

[10] A more generalized developement of the term „product model“ and „multi-model“ for the construction industry, as well as a description of the need for centralized digital planning is described in chapter 1 of (Scherer and Schapke, 2014). For this work a broader description will not be given.

2.2.1 Product modelling standardization: ISO STEP norm

Till the late 80th of the 20th Century standard file formats for data exchange were limited to the description of geometrical information (one example is the open file format IGES). Such files are lacking the description of the semantic context. The International Standardization Organization ISO declared "that none of the existing formats could on their own be extended to serve the needs of an open computer modeling standard for multiple industrial and manufacturing industries" (Laakso and Kiviniemi, 2012). This reason and the desire for standardization and non-proprietary file formats led to standardization efforts with the ISO STEP project (Standard for the Exchange of Product Model Data). STEP was published as ISO 10303 standard[11].

The standardization efforts in itself were judged as highly necessary, but within the main bodies of ISO declared as too slow for the big demand of action. The standardization efforts for a neutral, semantic-rich file format hence was extracted from ISO, the IAI (International Alliance for Interoperability) was founded to develop the IFC format (Industry Foundation Classes) thereafter. Laakso and Kiviniemi (2012) give in-depth information about the development of the IFC file format.

2.2.2 BuildingSMART / Industry Foundation Classes (IFC)

The International Alliance for Interoperability was later re-branded into "buildingSMART". The organization is set-up to the development of open data standards and procedures for the building industry. BIM was already described as "more of a method, rather than a standard". Next to a neutral data exchange standard and file format for built structures (Industry Foundation Classes – IFC), their application-relevant subsets (Model View Definition – MVD) and project model update files (Building Collaboration Format – BCF) buildingSMART furthermore tries to standardize the terminology of the international building sector (International Framework for Dictionaries – IFD) and processes of cooperation in information exchange (Information Delivery Manuals – IDM)[12]. This thesis will only deal with the data model part of it, the IFC model.

IFC was created independently by buildingSMART, yet its current version IFC 4 was later on standardized by ISO as their norm ISO 16739:2013[13]. Even though also proprietary file formats exist, IFC is recognized as the only vendor-neutral format and is being pushed as the mandatory file format for big projects[14]. Its predecessor was the version IFC 2x3. For this work an example file of a BIM tunnel model was studied, which was only available in IFC 2x3 format, which is why all reflections are always to be seen in relation to that version. Chapter 2.2.4 later on picks up from there showing the downsides of this in context to the use of it for this thesis, but also of the current version. And chapter 6.3 gives further information for the use in infrastructure and underground construction in the future.

---

[11] Eastman (1999) gives a thorough introduction into this topic: Several operative ranges of numerous industries categorized "their data and models" into "application protocols". Data of the construction industry are part of the AP 225 (Building elements using explicit shape representation). The central geometrical model of the AP is the "Building Construction Core Model" (BCCM), which was developed until 1994. Previous system engineering efforts tried to break down the structure of industries output (mainly buildings) using graphical notations like IDEF1 and EXPRESS-G (developed within the STEP process). A model will be made computer-readable by transformation into the EXPRESS schema and serialized as a "STEP Physical file" (as described in ISO 10303-21 "Clear Text Encoding") – also refer to chapter 2.3.3.5.

[12] More information is given on their website: http://www.buildingsmart.de and http://www.buildingsmart-tech.org.

[13] The acceptance of the ISO norm within Europe for IFC4 to also be credited as a European norm EN ISO 16739 was decided upon in October 2016 – refer to: http://www.cafm-news.de/ifc-jetzt-auch-europaeische-norm/ (Website in German, visited 06.11.2016) – a German national acceptance as DIN EN ISO 16379 is about to come.

[14] The BIM guideline 2020 of the German Government (BMVI – mentioned in chapter 2.2.2) e.g. demands the use of IFC

2.2.3 Basic concepts in the IFC hierarchy

The basic set-up of the IFC file format follows the classical paradigms of object oriented programming. All created objects, relationships and dependencies are also reflected in the human-readable, ASCII-encoded STEP Phyiscal File (SPF) format, a building model can be handed over with. To make the work of chapter 4 and 5 understandable, this section is intended to give a brief introduction into the basic concepts of the IFC file format. More thorough documentation can be found in the online corrigendum of BuildingSMART (2007) and in the IFC Model Implementation Guide (Liebich, 2009).

2.2.3.1 Class hierarchy, schema extensions and layer structure

Same as e.g. in Java programming, where all objects inherit basic functionality of the root class “object”, most objects in IFC are derived from its base class “IfcRoot”. IfcRoot itself, same as for several layers of its child classes, are all abstract and thus cannot be instantiated.

The overall hierarchy is built in schemas and extension packages. The basic classes, which are necessary for every project, are part of the “IfcKernel” schema. The schemas above that already differentiate between what is to be described: A “product” (in terms of the described “product modeling”), a process (defined as an act in time, e.g. the construction of a part of the building) or a “control” (external influences, like Actors, or rules and law). They all make their own extension with more specialized classes. That is the core layer of IFC. Figure 5 shows that concept.

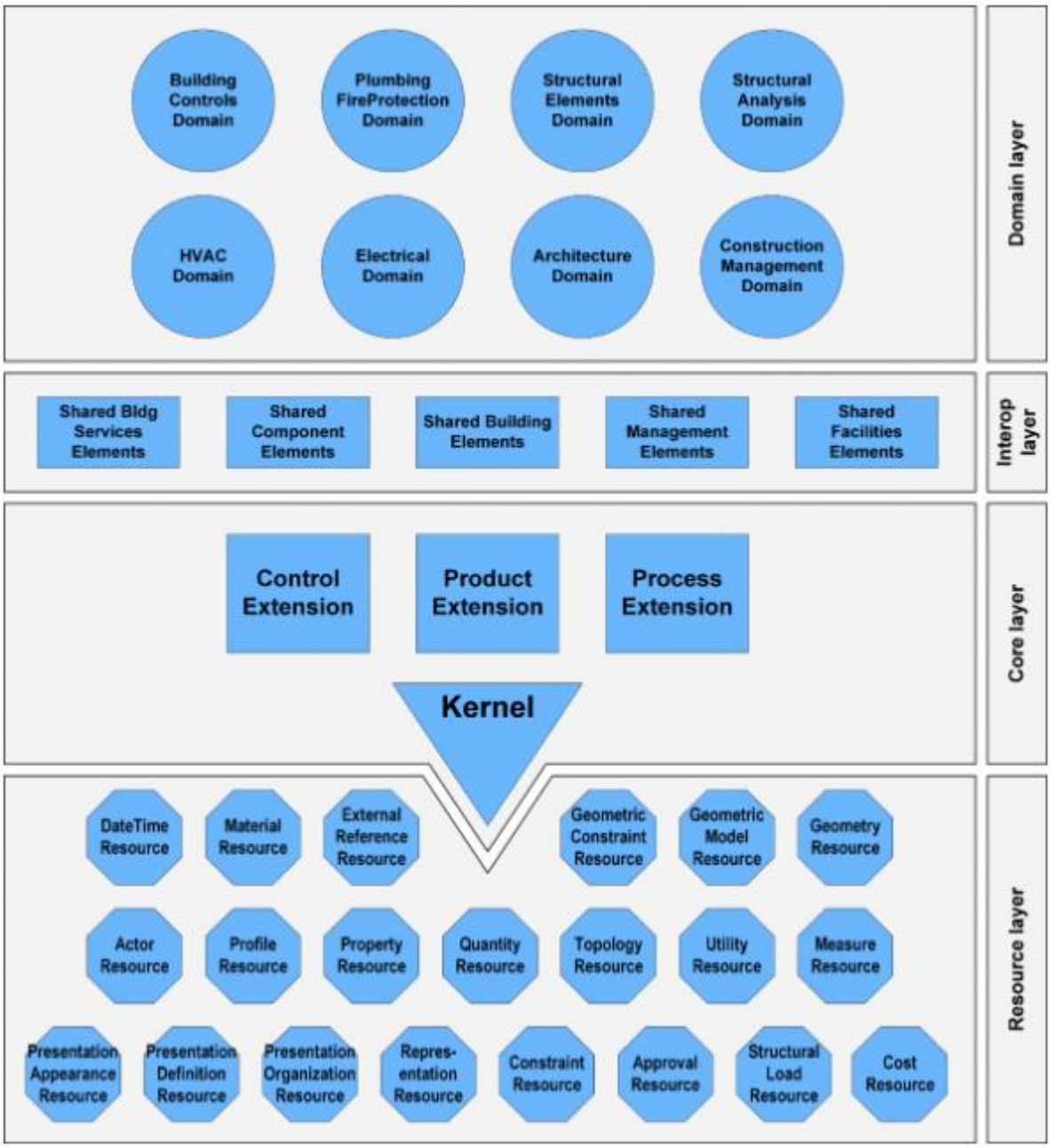


Figure 5: *The IFC Architecture and layer structure* Source: Borrmann et al. (2015), p. 81

The “interop layer” or “shared layer” hosts classes which are used by several end-user domains. For a building, the schema “shared building elements” e.g. provides basic classes every building needs, like “IfcWall” or “IfcWindow”. By definition a full building model was to be constructed mainly using schemas of the domain layers, which provide further specialization and apply to certain domains in the construction business.

2.2.3.2 Spatial structure vs. Objects

Also non-physical concepts of a jobsite, the structuration of available space, is reflected and objectified in the IFC schema, e.g. with classes for “IfcSite”, “IfcBuilding” or “IfcSpace”. A “building” in that sense, same as the terminologically more generic “space”, is only a notional construct. A physical building is made of several smaller pieces, which is why the building object itself is abstract, consisting of its physical constituent parts.

In the same way “space” is used to group physical pieces to a package – it is widely used also for this work: Segments, as the physical pieces which form a ring in tunneling, are grouped to a space called “ring” within the data model.

It is this mixture and linking of objects representing “real physical things” and objects representing theoretical constructs, which are used to create the spatial hierarchy of the model. Figure 10 shows this concept exemplarily for a selected ring of the used tunnel model. Figure 6 furthermore shows the described concept in reference to a selected group of classes of the IFC hierarchy.

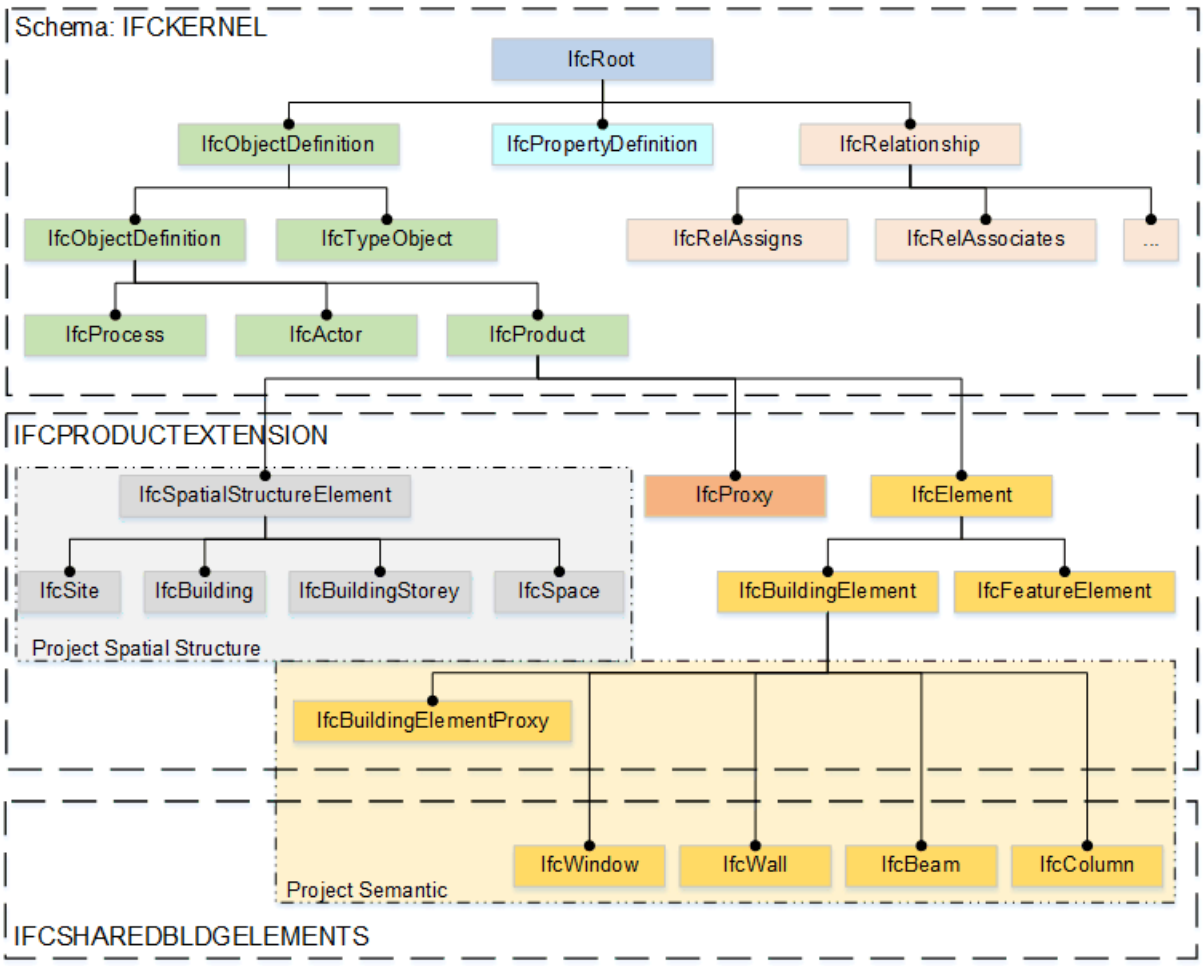


Figure 6: *Structure of the IFC schema. Classes for the creation of a spatial structure are to be differentiated from those creating a semantic structure, or to create relationships. They can also be nested in different schema extensions. Adopted from* Borrmann et al. (2015), p. 84

2.2.3.3 Objectified relationships

In object oriented programming, information that links objects together are typically saved using attributes: An object will have the direct link to the related object in a named reference: An object of class “wall” may save the identification of an object of class “void”, which is needed to represent the wall opening, as a prerequisite to house a window, as shown in Figure 7.

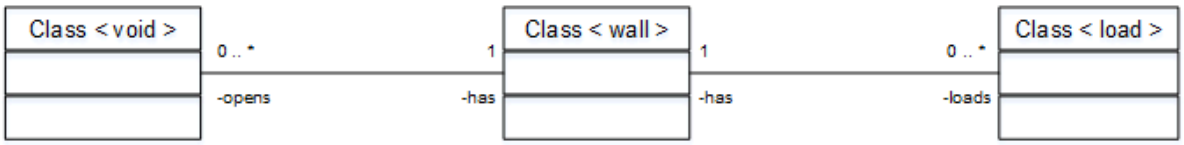


Figure 7: *Relationships in normal OOP: A wall can have one or several voids (e.g. for windows) and is exposed to one or several loads. This is handled using attributes in the objects itself.*

In the IFC schema uses the concept of explicit objectification, where also the relationship between two instances of a class is handled as an object itself, as shown in Figure 8.

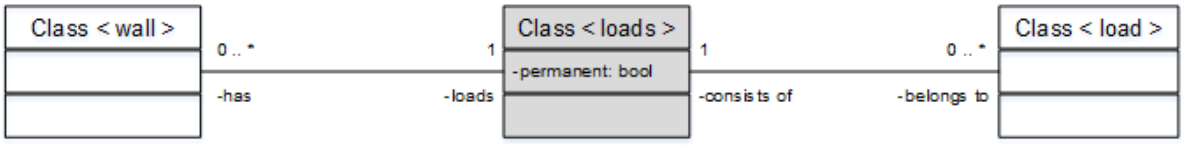


Figure 8: *Objectified object relationships, as used in the IFC schema: The very fact that the wall is exposed to a load is transferred to its own class.*

The abstract class “IfcRelationship” is parent to several types of relationship classes, which are used to structure the data model. These classes structure relationship types into those of spatial aggregation / composition for physical parts, those for linking to external sources, those for association to further properties and more. Examples for such classes are:

- IfcRelAggregates: Aggregation-Relationship between one object and one or a set of other objects, aggregating / linking them to form a unit. E.g. linking several concrete segments to form a complete tunnel ring
- IfcRelDefinesByProperties: Links a class to a single or a set of properties. E.g. linking a tunnel ring to further information about that ring, like the ring number
- IfcRelContainedInSpatialStructure: Links a class to a physical structure, like a space, to create an overall spatial hierarchy and structure within the building model

Besides the ones mentioned there are many more relationship types. The relationships follow the same logic as the rest of the data model: as the semantics of the building (using well defined classes for cases, defined in the application schemas in the domain layers) is separated from the creation of the overall spatial structure, there have to be classes which can link together the semantic horizon and other horizons. This topic is again picked up in the next section. Figure 9 shows an example for that from the official IFC documentation: Entities are linked using relationship references.

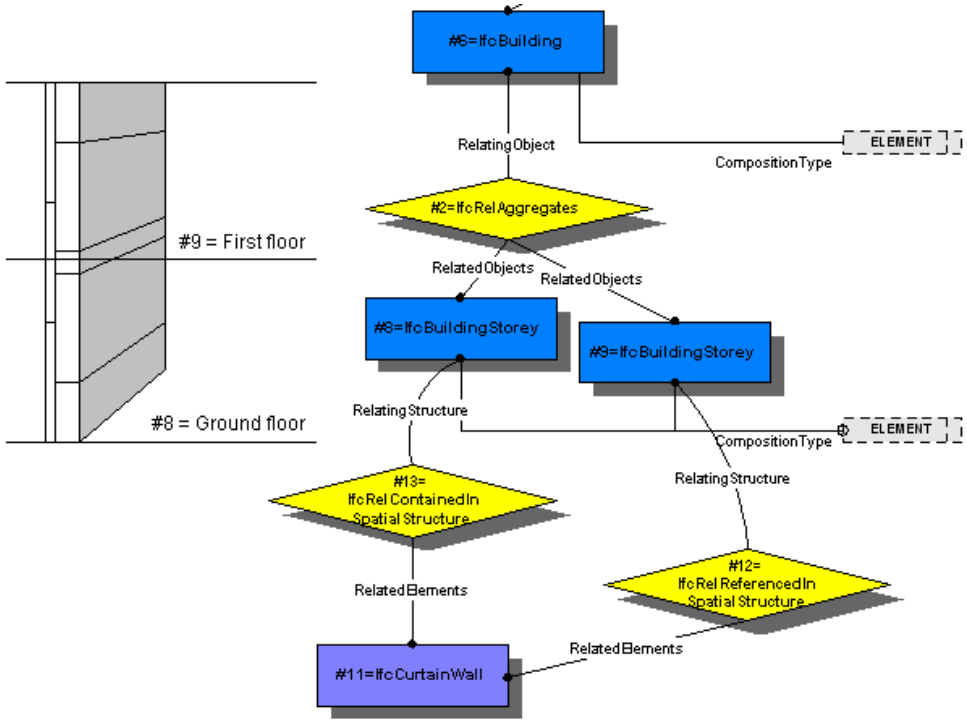


Figure 9: *Example for spatial relationship: A reference with IfcRelContainedInSpatialStructure assigns the physical item of IfcCurtainWall to the structure object IfcBuildingStorey. Two storeys are aggregated to form the overall building* Source: BuildingSMART (2007)

2.2.3.4 Semantic vs. Geometry and Properties

The semantics is furthermore also to be seen separated from any form of geometrical representation for the classes: A building model can be linked to one or more ways of geometrical encoding, which can be used to visually display a model on a computer screen when loading the file. Within CAD systems different ways of geometrical modeling are distinguished[15], which all find their representation in the IFC data model. As this work is more focused on the data model "in the background of visual representation", further introduction into the topic will not be given.

More than that, also linking the whole model or pieces of it to any other information is an additional horizon to it, handled using properties. Single properties or sets or different kinds of properties can be assigned to object at any level of the model using the appropriate relationship class.

Figure 10 gives an overview of the decomposition of the tunnel model file for an exemplarily shown ring number one, to highlight this approach.

2.2.3.5 STEP Physical File format

The file format for transferring an IFC Model is the ISO 10303-21 standardized "STEP" format (Standard for the Exchange of Product Model Data)[16]. It is based on the EXPRESS schema

[15] Forms of geometrical representation e.g. are: Solid modeling, Constructive Solid Geometry or Boundary Representation (BREP) which is also used for the tunnel model in this work. An introduction into the means of that is given in Borrmann et al. (2015)

[16] A more modern approach to file transfer, in 2001 the first version of ifcXML was released. But as even STEP creates big files for BIM models, the XML equivalent created even bigger files. "Because of the inherently different structures of the STEP-File and XML modeling language, translation of native IFC STEP-File to ifcXML result in needlessly large, lossy, and unoptimized files which compromise the strengths of XML modeling" (Laakso and Kiviniemi (2012), p. 16, p. 16) Still, the XML-Version of the IFC file format is still maintained, also for IFC4, as the latest version of it.

and serializes the IFC model into an ASCII-based file. This file hence is human-readable, but still remains almost incomprehensible to the human eye. Even for a simple model it is not unusual for a file to grow to several one hundred thousand lines. The following lines depict an example section of such a file:

```
#245924= IFCMEASUREWITHUNIT(IFCRATIOMEASURE(0.01745329251994328),#245923);
#245925= IFCDIMENSIONALEXPONENTS(0,0,0,0,0,0,0);
#245926= IFCCONVERSIONBASEDUNIT(#245925,.PLANEANGLEUNIT.,'DEGREE',#245924);
#245927= IFCUNITASSIGNMENT((#245921,#245926,#245919,#245920,#245922));
#245928= IFCPROJECT('e7l3m3q5v2x5l0',#11,'Reference Project Wehrhanhlinie
         D\S\|sseldorf - Tunnel','Subway',$,$,$,(#4),#245927);
#245929= IFCBUILDING('a3h9c3a5o1d3h9',#11,'Tunnel structure','Tunnel
         structure of the tunnel alignment',$,#3,$,$,.COMPLEX.,$,$,$);
#245930= IFCBUILDINGSTOREY('g5g8j4l8f5m5o5',#11,'Tunnel part','Part of the
         tunnel structure of the tunnel alignment',$,#3,$,$,.COMPLEX.,$);
#245931= IFCRELAGGREGATES('c3w1t4p2x0d1y0',#11,'Tunnel part to
         tunnel','Linking tunnel part to whole tunnel',#245929,(#245930));
#245932= IFCSPACE('c7x6e6w8u1b9h0',#11,'Full Tunnel Space of tunnel','Full
         Tunnel space of the tunnel
         alignment',$,#3,$,$,.COMPLEX.,.EXTERNAL.,$);
```

Every line describes a single entity of the IFC schema. In context of inheritance and relationship, the entities are linked with jump marks using a diamond- or hash-sign. In shown example the entity #245924 links to entity #245923. The attributes of each entity enclosed with dashes mark the full inheritance path of the object type to the root type “IfcRoot”.

A file of such coding can be read into a software, implementing a suited EXPRESS parser, which is able to recreate the class model and hierarchy accordingly. For this work the IFC-Java parser of the IFC Tools Project was used (refer to chapter **Fehler! Verweisquelle konnte nicht gefunden werden.**).

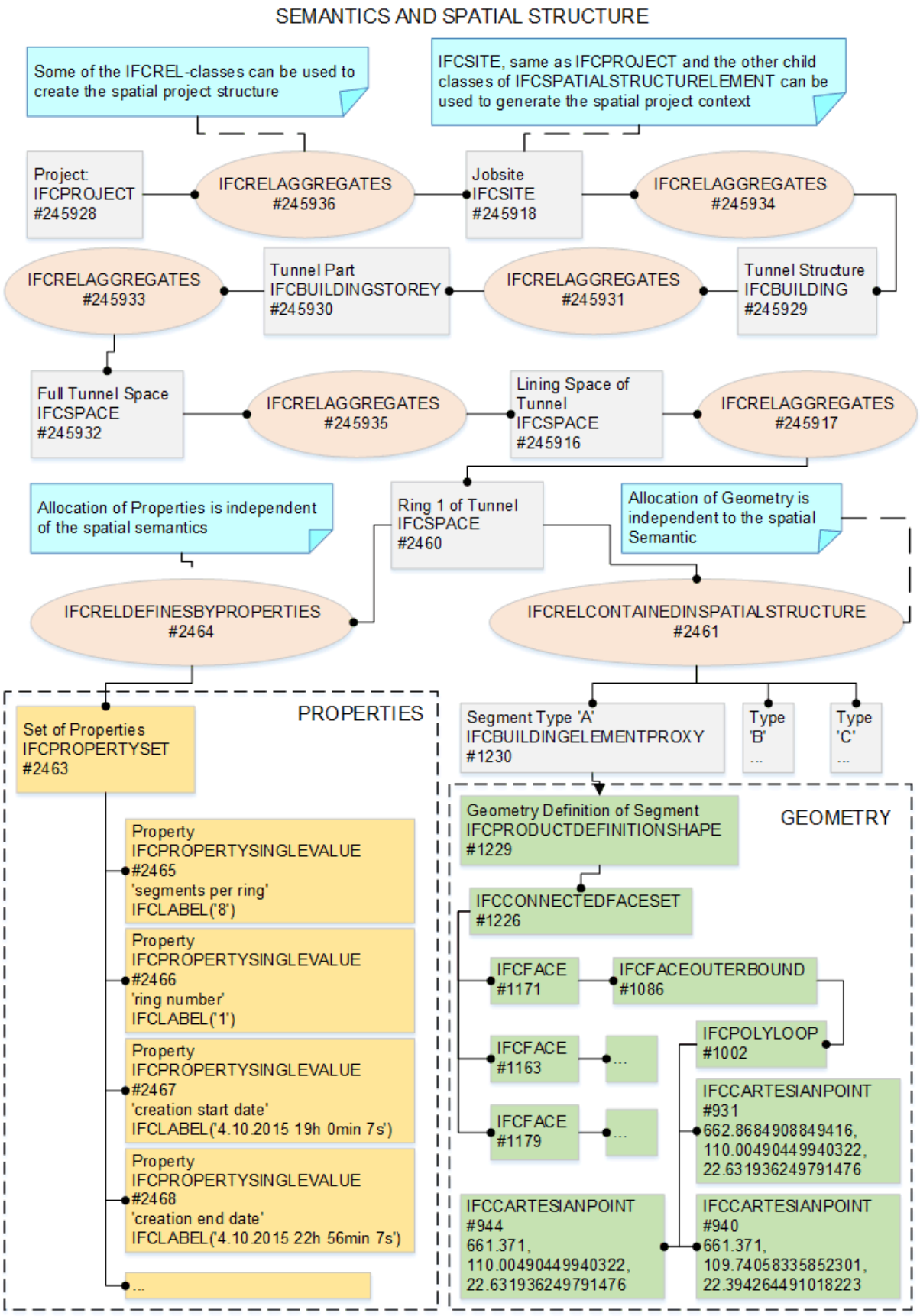


Figure 10: *Decomposition of ring #1 within the tunnel model and further linked classes for geometry and properties*

2.2.4 BIM-based Tunnel Models using IFC: IfcTunnel & IfcShieldTunnel

Up to now all versions of the IFC file schema, also the current version IFC4, are merely focused on surface construction engineering, like buildings. Yet, “the current version of IFC lacks the ability to comprehensively describe infrastructure facilities such as roads, bridges, railways or tunnels in detail” (Vilgertshofer et al., 2016, p. 1).

Though, it has to be stated, that it is not the final goal of the format to fully describe the whole built structure and environment in every detail. A model made in any specific BIM-enabled design software might always comprise details, which are not reflected in the schema: It is mainly special object types of any kind, not part of the IFC schema, which generic entity classes – so called “proxy objects” – are made for: Containers to save characteristics unrepresented in the IFC schema. Like that up to now it is necessary to use a lot of those proxy containers, when saving tunnel models into an IFC file. This is also true for the use of generic “Space objects”. This necessarily generic approach during saving goes along with information loss: A specialized simulation software e.g. could need the detailed object type information for further calculation processes.

To demonstrate this issue, the first ring of the tunnel model used for this work was decomposed into its IFC classes. Figure 10 shows the result of the decomposition. It is clearly recognizable that the use of mentioned proxy objects was used a lot to create this model. The model is aggregated over several steps, to form the spatial structure and hierarchy of a jobsite:

1. Project (Object type `IfcProject`)
2. Jobsite (Object type `IfcSite`)
3. Tunnel (Object type `IfcBuilding`)
4. Tunnel part (Object type `IfcBuildingStorey`) as the model is only showing a part of the whole tunnel
5. Tunnel full space (Object type `IfcSpace`) as it is anticipated that a model of the tunnel may not only consist of the lining (concrete segment) part of it
6. Tunnel lining space (Object type `IfcSpace`) as the part relevant to concrete segments, housing all rings
7. Each tunnel ring (Object type `IfcSpace`)
8. Ring single segment (Object type `IfcBuildingElementProxy`)

Research initiatives of several universities are tackling this specific point. Projects funded by the German Research Foundation (DFG)[17] researched the necessities and made proposals for IFC extensions in relation to a better reflected tunnel model (creation of schema adaption “IfcTunnel”). Before that another attempt has been made by researchers in Japan, leading to the creation of the differently structured schema adaption called “IfcShieldTunnel”. Both should be described in the following.

Early attempts for the creation of an IFC schema extension were made in 2007 by researches of the University of Osaka and were presented in Yabuki et al. (2007), (2009) and (2013). The team broke down the semantic structure of a tunnel building into great detail and created IFC classes for most of them. The resulting proposal is very rich in its amount of classes. Figure

[17] A DFG project within the Collaborative Research Center SFB 837, as well as the project grant FOR 1546 lead to various publications in that field, mainly focusing on the Technical Universities of Bochum and Munich.

11 shows the proposal for the extension of the spatial structure part of the IFC schema. Classes for real physical elements (child classes of IfcElement) are shown in Figure 12.

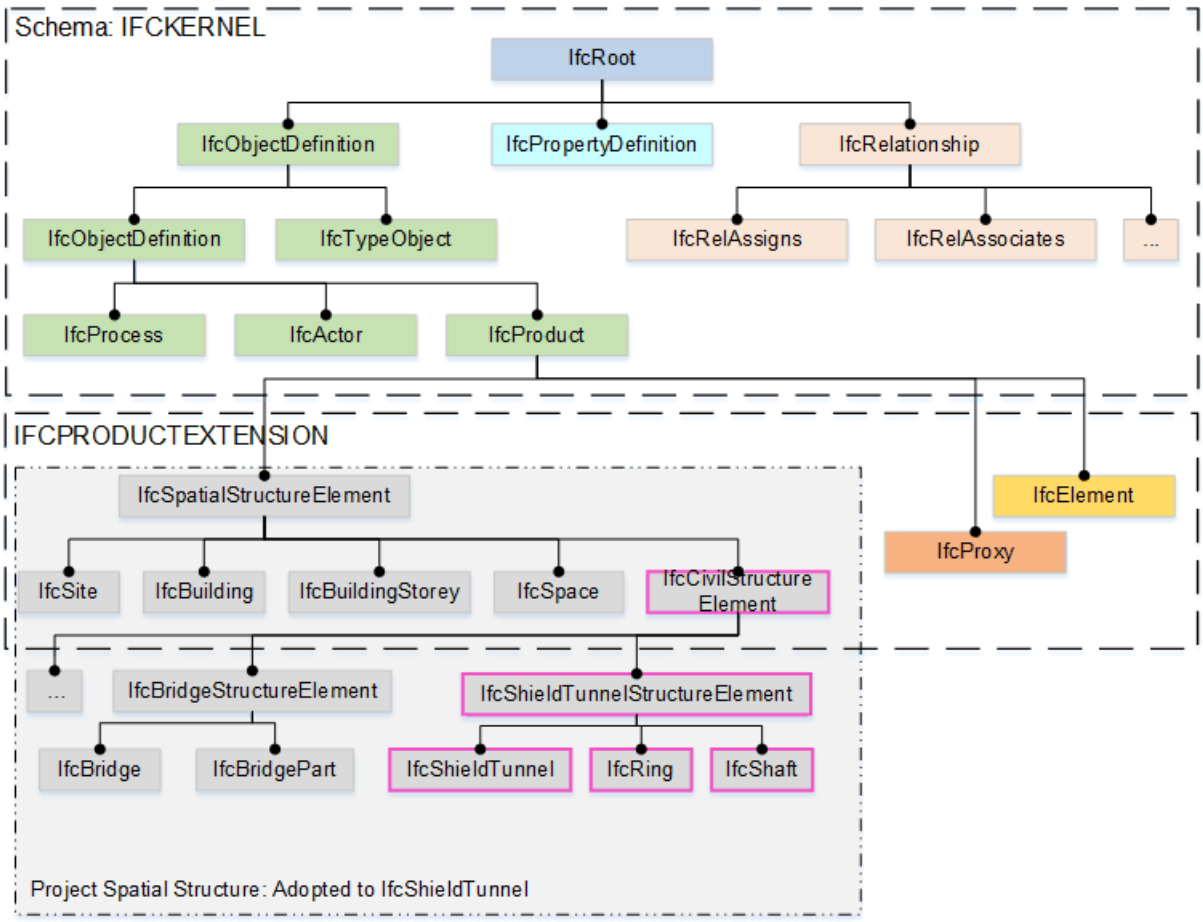


Figure 11: *Extension of the spatial structure approach of the IFC Product Extension, with respect to "any other results of civil engineering" (other than buildings in surface construction). Not only tunnels where considered. The relevant part for this thesis is highlighted in pink)*

Another attempt to the creation of an IFC structure for tunnel buildings was made by research groups based at the universities of Bochum and Munich. It references the work of Yabuki et al., but has a more generic approach to the modelling part. It was published first in Amann et al. (2013). Also for the tunnel model at hand for this work it is true, that the extensive use of the concept of generic space objects, was used for mere reasons of necessity: no other spatial concept for tunnel buildings was available. Amann et al. now proposed six new classes for modelling purposes, three of which are used for the spatial structure part of it. The specialization of certain classes should be done with enumeration classes, offering a set of options to choose from. Figure 13 shows that concept. To only use child classes of IfcSpace thus is taken over into the general design considerations.

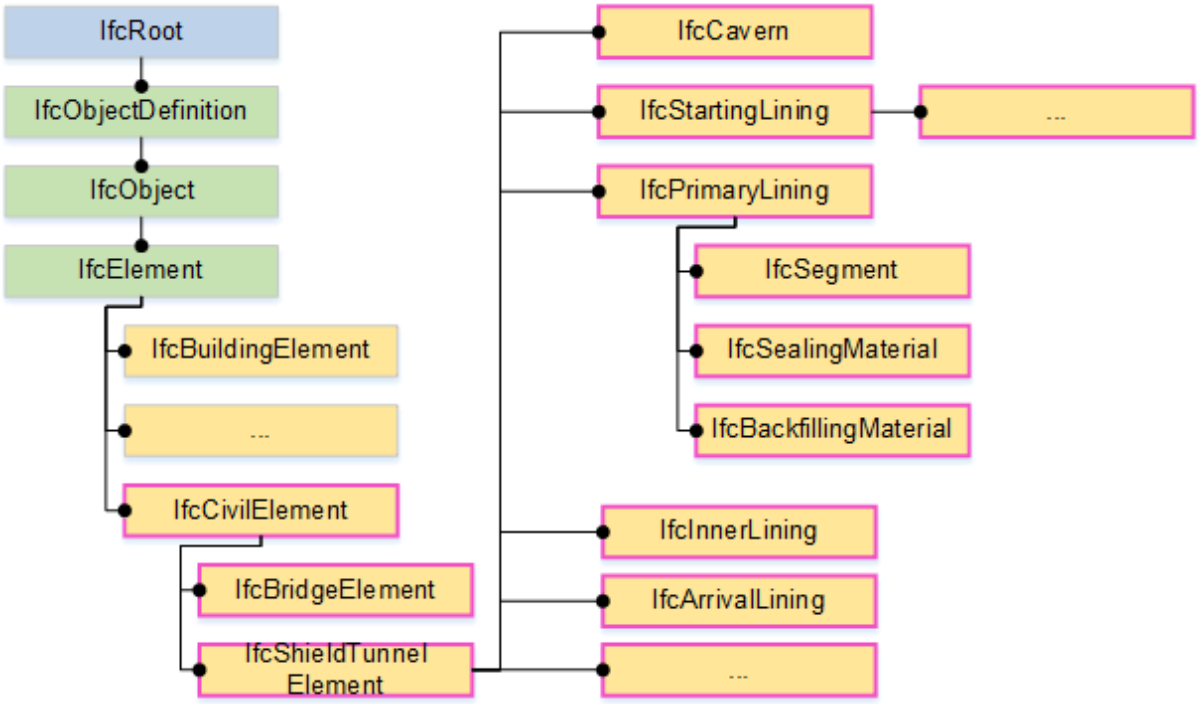


Figure 12: *Extension of the element list, with respect to tunnel buildings. Newly proposed elements highlighted in pink. Adopted from* Yabuki (2009)

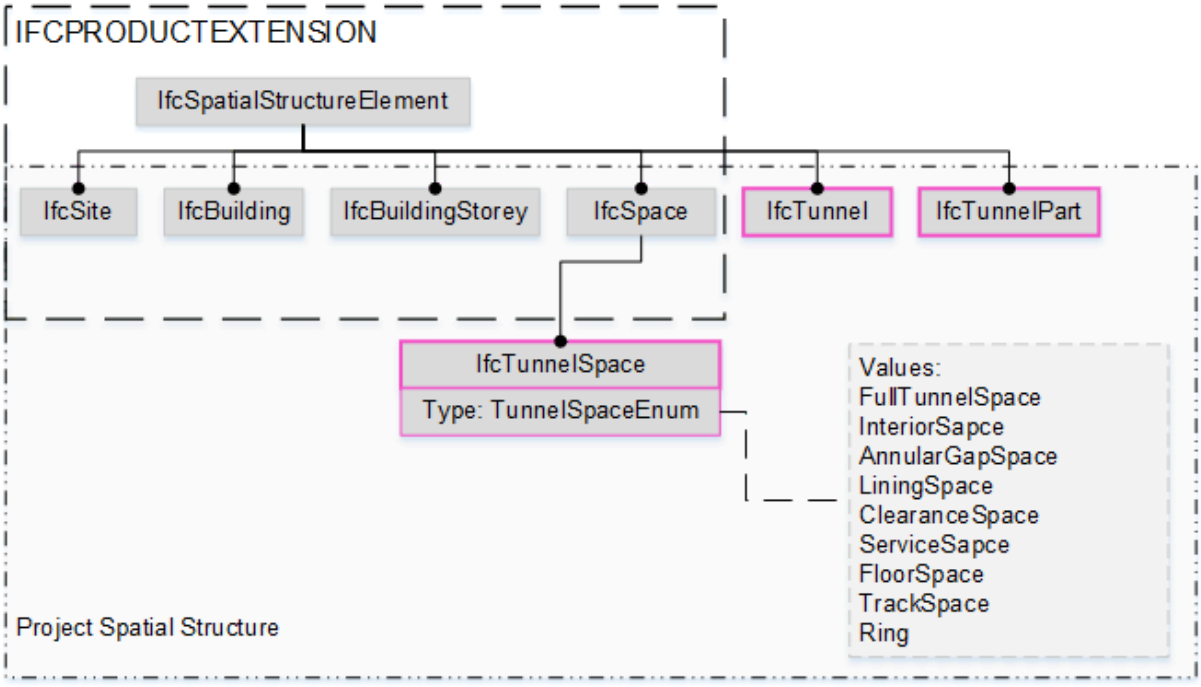


Figure 13: *Extension of the spatial structure approach of the IFC Product Extension, highlighted in pink color. Only tunnel buildings are considered within the publication of Amman et al. An enumeration list offers to choose from a set of attributes for the IfcTunnelSpace object to clarify the specifics of the object*

Also the list of classes for physical elements was only extended with three further classes and an enumeration class for specialization, as shown in Figure 14.

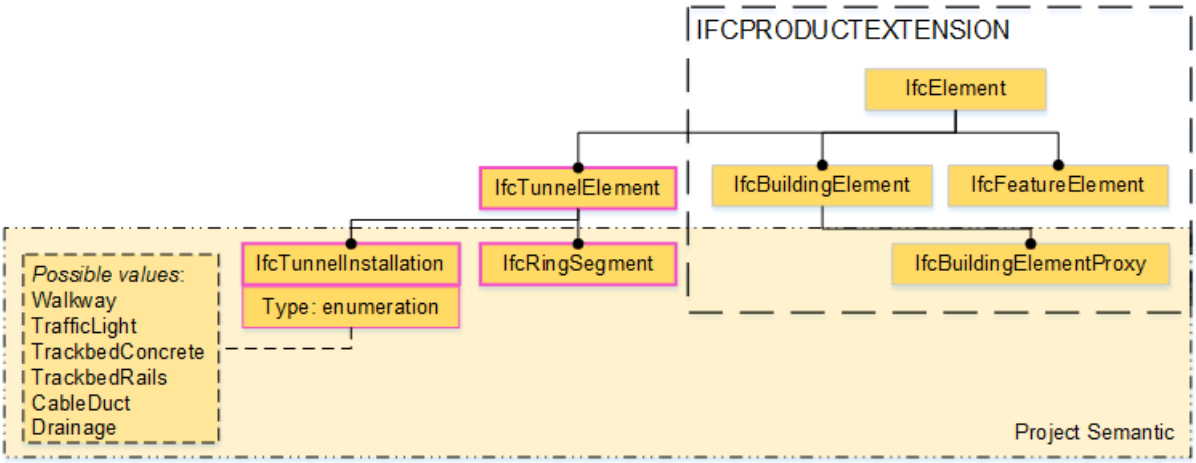


Figure 14: *Extension of the element list, using enumeration classes. Newly proposed elements highlighted in pink.*

Both of the presented extensions for tunnel construction are only proposals, none of that have yet found reflection in any commercial software products available on the market. Yet, taking the density of those publications as an indicator, surrounding the further research projects in which the German extension “IfcTunnel” was created, one could come to the impression that this version might be more likely to later be further developed, handed in and recognized as an official addendum to the IFC standard.

2.3 Tunnel construction

Generalized, a tunnel is categorized to the group of cavity structures, next to galleries, shafts, caverns and cave chambers (Maidl et al., 2013b). In the spectrum of construction and advance methods for tunnel buildings the shielded tunnel boring machine (TBM) is only one technical means, next to partial-face excavation machines, open face construction without ground water and earth pressure compensation, Roadheader machines, drill and blast tunneling and more. Girmscheid (2012), p. 67 ff. as well as Maidl et al. (2013b) distinguish the different methods of tunnel construction. This work focuses on TBM tunnel construction only.

Tunnel, constructed with a TBM, with an outer diameter of four meter and more most often use a set of precast steel-cast reinforced concrete segments or steel fiber reinforced concrete segments as their structural lining system. Those segments are set together to form a ring. The segments are produced watertight and drained, enabling to use them as a single liner. This work will only talk about concrete segments in general and not distinguish between different types of concrete. Also it will neglect other types of tunnel lining segments like steel block or cassette segments.

The complex interaction of construction machinery, the ground in which they operate and other involved components and systems is described as „system behavior”. The works on a tunnel construction can thus be divided into subsystems, which most often represent a single construction step and are thus process-oriented[18]. Every process must be controlled. Set alarming values for each system and “Key Performance Indicators” (KPI) can be observed near real-time (Maidl et al., 2014, p. 355 ff.). This requires a data acquisition system, which

[18] Tunnel construction using a TBM for instance can be subdivided into the processes advance, foam injection, grouting, segment installation, material transport and more.

collects all necessary values of sensor installations, makes KPI calculation and visualization. Chapter 2.4 will give a further introduction into the topic of those tunnel construction information systems.

Exemplary information of TBMs and segment production will later be used in this thesis in a context of a tunnel construction information system, which is why an introduction is given next.

2.3.1 Mechanized tunneling using TBMs

Tunnel boring machines with segmental lining are subdivided into different types, according to the model. Model types differentiate how a machine deals with earth- and face pressure and how it handles muck- and material transport. A model is chosen mainly in consideration of the geology the machine is used in[19]. Removal of earth material as well as the installation of segments is performed protected under a steel shield, which is why the machines are called “shielded”. Distinction is drawn between Earth-Pressure-Balance Shields (EPB), Mix- or Slurry-Shields and Gripper Shields. Those types can have the format of a single shield or double shield geometrically[20]. Figure 15 shows such an EPB machine exemplarily.

Figure 15: *Tunnel Boring Machine in type of an Earth Pressure Balance (EPB) shield for the London Crossrail Project* - Source: *EPB Machine* [ONLINE]. Available at: http://www.Herrenknecht.com[Accessed 16 December 2016].

Full-face Tunnel Boring Machines generally work with a rotating cutter wheel – thus only circular profiles can be created with such machines. The persisting tunnel, made of before mentioned concrete segments, is put into place using an erector behind the shield.

[19] The responsible working group of the German Tunnelling Committee DAUB issues their “recommendations for the selection of tunneling machines” (Maidl et al. (2013a), p. 401 ff., p. 401 ff.)

[20] Manufactures of such machines e.g. are the companies Herrenknecht AG of Germany, The Robbins Company of USA, NFM of France and Hitachi Zosen of Japan.

Excavated material is brought out of the tunnel either on conveyor belts (when EPB shields are used), as displayed on Figure 16, or pumped out in a slurry circuit (when Mix- or Hydroshield machines are used).

Figure 16: *Muck removal using a conveyor belt system* (Source: Own imagery)

As in the following this work will only issue data of TBMs generically, the further text will not distinguish any types of machines but will rather use the concept “TBM”.

2.3.2 Segment Production

Segments are built increasingly in a manner, that the resulting carcass of the tunnel later on does not need a secondary internal lining or protection. Thus, it is later those installed segments, which form the gross of the final state of the tunnel. These tunnel have an aimed life expectancy of up to 100 years. During this time, they are exposed to forces, water, chemicals and more[21]. To ensure the operating safety of the tunnel, accordingly rigid means of quality assurance must be guaranteed.

Segments for a tunnel project are fabricated in existing precast concrete factories at some companies. Otherwise they must be fabricated in special field factories, built especially for one tunnel project. Segments are produced in precise steel moulds, where the concrete is poured in. Two choices of production organization are well-established: the stationary production and the carousel production. Which one to choose depends on various parameters, like “local conditions, tunneling advance rate, project time schedule, expected operation period, time and available land plot, personnel requirement, production capacity, […]” (Riechers, 2014). An example of a field factory is shown in Figure 17. Ready produced segments can be seen in Figure 18.

[21] Demands on durability are even higher, when the tunnel is supposed to serve for transportation of water or waste water

Figure 17: *Example of a segment field factory using a stationary production scheme* (Source: own imagery)

During production reinforced steel cages are inserted into the moulds first, or steel fibers into the concrete mixture or both. This is necessary for the segments to be able to resist forces during production, transport, installation and the long lasting project lifetime. Concrete is poured into the moulds, vibration machinery is necessary soon after to release air off the concrete. Exposure to heat and a decent humidity for a right amount of time is necessary for the concrete curing process. The production and storing processes in itself make it necessary for the segments to be moved, turned and lifted. Every such working step is a potential hazard to the quality of the segment. Lastly the segments must be stored and the concrete must cure a right amount of days, for it to be allowed to be installed in the tunnel.

All the before mentioned working steps must be controlled. Therefore, specially designed software systems are used to ensure the correct production steps, long enough curing times and more[22]. Data of those systems are used in the later context of this thesis, which is why it will furthermore be described in chapter 0.

[22] Providers of such systems are e.g. VMT GmbH (Product “Segment Documentation System”), Tunnelsoft (Product “Segment Tracker”) or SSB Strauch

Figure 18: *Tunnel segments on storage yard* (Source: Company VMT GmbH)

2.4 Information Systems

Information systems assist in aggregating data, of divers sources and structure and merge it into a single framework. As for tunnel construction: a linkage of heterogeneous data pools is established between systems, which normally operate on their own, like TBMs, surface and sub-surface surveying systems, systems for the treatment of excavated material (like conveyor belt systems or slurry circuits with their separation plant) and logistical applications. They can serve as an “aggregation layer” over the jobsite information infrastructure, which is established for the process data gathering and analysis. Figure 19 displays such a concept.

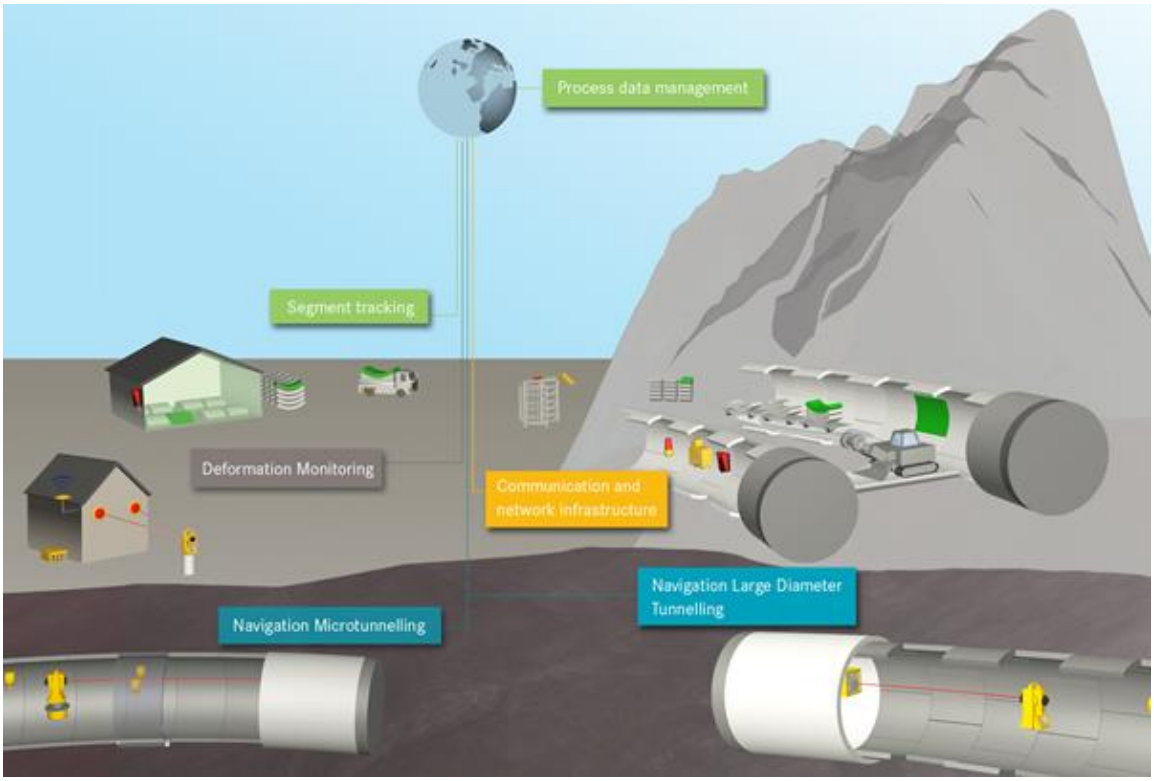


Figure 19: *Example of a Process Data Management System for data aggregation as a head layer over the jobsite system infrastructure* (Source: Company VMT GmbH)

Next to automatically generated building process data of machines and devices, a successful evaluation approach needs more input: information of the design phase of the building, alarm values and geotechnical profiles and information. Maps and layered viewers furthermore help for a visual localization of events, machines and more. Particularly expert knowledge and engineering models play a vital role in understanding of events on the TBM. Maidl and Stascheit (2014) as well as Mayer et al. (2015) give an overview and examples of such engineering data evaluation approach in tunnel construction. Figure 20 gives an impression of possible input sources.

In context of tunnel construction, the use of such tunnel construction information system prevails on big jobsites. Their added value is of no question, “the question whether such a system makes sense or not does not arise” (Mayer et al., 2015, p. 180). The use of such systems enables the construction companies in the analysis and optimization of their processes. For the client it enables them to control the construction overall in means of supervision. Different solutions are already available on the market[23].

This work was done in conjunction with the software product “IRIS” and its tunnel module IRIS.tunnel, which served as an aggregating data pool to TBM data, data of a geodetic TBM navigation system, its ring sequencing software and a logistical management software of the segment production facility. All of these data sources are explained in the following sections.

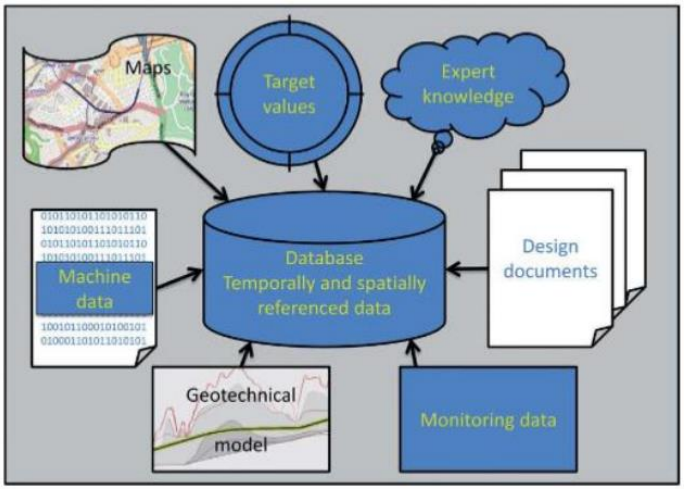


Figure 20: *Sources and input into a spatially and temporally referenced information system database* - Source: (Maidl and Stascheit, 2014)

2.4.1 IRIS.tunnel

IRIS (Integrated Risk and Information System) is a modular web based information system for use in tunnel construction and geomonitoring. The combination of single contextually meaningful modules creates the applications IRIS.tunnel and IRIS.geomonitoring. Both work on a universal, integrated data base structure, which is why data of both applications can be diluted. Following pictures Figure 21 to Figure 24 show some example visualizations of process information in the software.

IRIS embarks on sensor information of the tunnel boring machine or other machinery and devices, which are equipped with sensors and are somewhat reachable within the local IT network for the data acquisition services. Direct raw sensor information can be displayed such

[23] Manufacturer are e.g. the companies Maidl Tunnel Consultants (PROCON II), ITC Engineering (IRIS), VMT GmbH (VDMS) Geodata (Kronos), Tunnelsoft (TPC), Vinci Construction (CAP) or Acciona Infrastructure (TCC) and more

as statistically treated values (e.g. minimum or mean values within a certain time range) or on-the-fly calculated “virtual sensors”. Their value results from the use of e.g. certain civil engineering mechanical engineering formulas. Similar to the before mentioned KPIs can be calculated and displayed near-realtime. These and other sources relevant to the thesis will be explained in the next section.

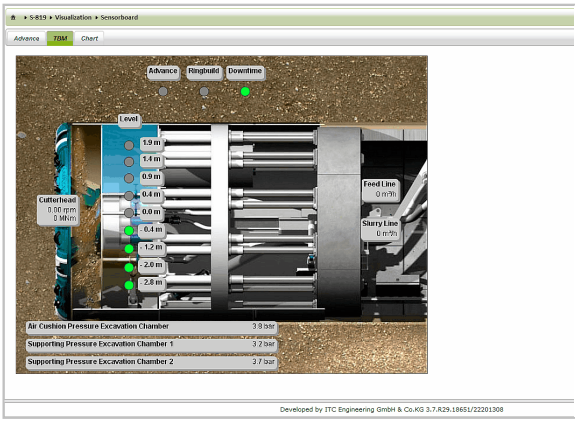


Figure 21: *Display of live machine information on a web dashboard* (Source: Company ITC Engineering)

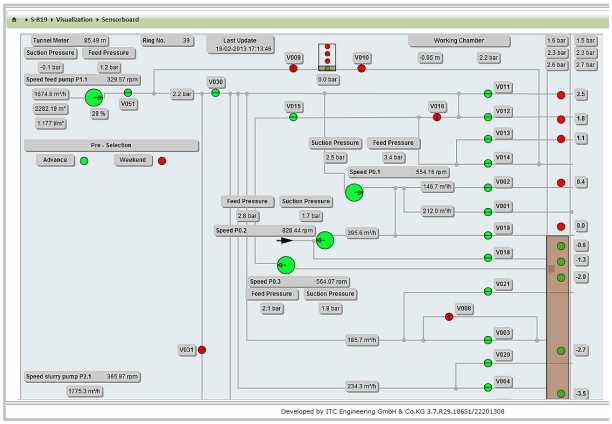


Figure 22: *Display of live information on a slurry circuit of a TBM* (Source: Company ITC Engineering)

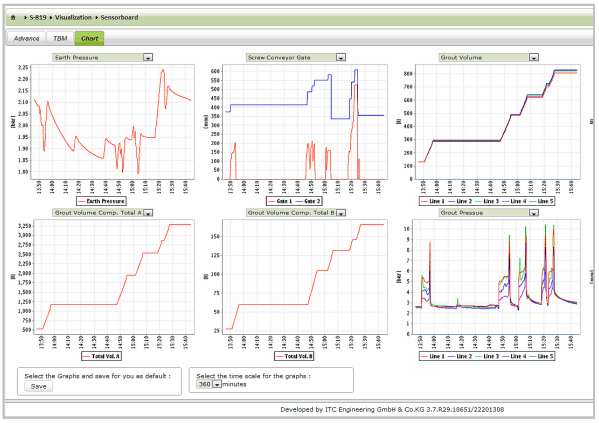


Figure 23: *Display of live evaluated Key Performance Indicators* (Source: Company ITC Engineering)

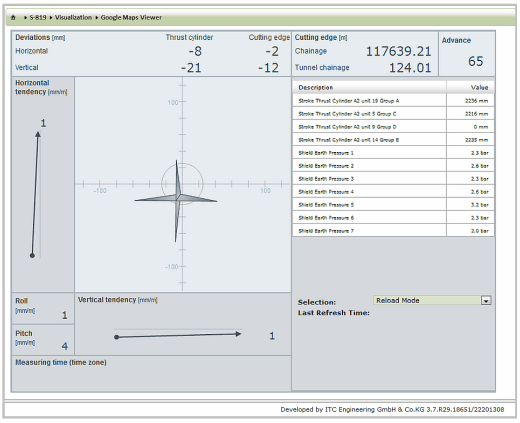


Figure 24: *Display of information of the TBM navigation system, position of the machine related to the project alignment* (Source: Company ITC Engineering)

A fork of the software “IRIS.tunnel” will be marketed as a product named “VDMS” from 2017 onwards – by an cooperation which the author is associated to, which is why it has also been used in this text, in chapter 4.2 and onwards.

2.4.2 Data sources

All relevant data generating systems need to be connected in one job site network, in a way that a central data acquisition computer can gather information from machines, sensors and systems. The information systems are developed as web based systems. A cloud server enables access to the data anytime from anywhere.

2.4.2.1 Machine data

A central and important controlling element of the TBM is its internal sensor data storage. Sensors generate values of physical sources and represent e.g. pressure or longitudinal extension. The results are transferred to and available in a Programmable Logic Controller

(PLC) and are updated in a range of a few Milliseconds. A PLC is a vital element of Control Technology for mechanical and plant engineering[24]. Moreover, the PLC offers external interfaces, which can be used to read and write data from and to the PLC.

A data acquisition computer on the TBM establishes a connection to the upstream operating software “OPC Server” of the PLC and reads current sensor information in cyclic intervals of a few seconds. The values are buffered to a local hard drive. Depending on the requirements and the type and infrastructure of the machine it easily adds up to several hundred sensors[25].

The forwarding of received data to the information system can be carried out via transfer of incremental CSV files to an FTP server. There the files must be read by an importer service, parsed to the coded sensor schema of the information system and read into the database. Also possible is the use of the REST paradigm, which was developed for machine-to-machine communication.

2.4.2.2 Navigation and ring sequencing data

A guidance or navigation system for a TBM in bigger diameter of such machines is most often based on a Laser Total Station and an active target unit in the TBM, which detects and processes an emitted laser beam of the total station and internal inclination sensors. A software system on a screen of the operator’s cabin displays the calculated live navigation results and updates them every few seconds. Figure 25 shows and example layout. Figure 26 shows the exemplary result of navigation on a screen.

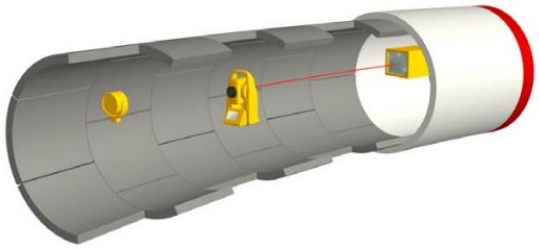


Figure 25: *Exemplary scheme of a laser based TBM navigation system* (Source: Company VMT GmbH)

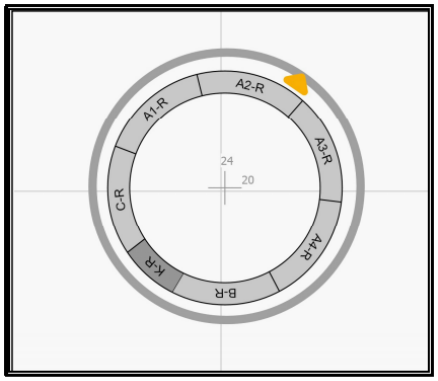
Figure 26: *Calculated navigation result on a computer screen* (Source: Company VMT GmbH)

Navigation ensures that the driver of the machine is capable of building the tunnel in the designed position. When an advance is finished, a new ring will be built in the shield of the TBM. An as-built survey of the ring is done by the navigation system automatically (VMT-GmbH, 2016). This is vital information, as it serves as installation proof in a practical context and can serve as necessary input for the tunnel model of this thesis: The ring sequencing of the navigation system is able to deliver automated values for an as-built tunnel model. These information, as well as values of the navigation system itself, are saved in the PLC of the

[24] As the name implies, the logic of the machine behavior can be programmed: Depending on input values into the system output values are set. With the operating software of the PLC the overall behavior and logic of a machine can be guided. Like that it is e.g. possible to ensure that the cutting wheel of the TBM only turns if all necessary technical prerequisites and safety precautions are complied with.

[25] With e.g. 600 Sensors and a reading frequency of 5 seconds that sums up to c.a. 10,3 Million values per day. This does not make the system a “big data” kind of system – but this simple number still underlines the potential to grow into that in the future, especially when predicting machine behavior would come into play

machine and are read by the data acquisition computer. Following figure 27 gives an examples for that.



Ring type:		**R11**
Chainage ring centre:	[m]	139.00
Tunnel distance ring centre:	[m]	39.01
Horizontal deviation ring centre:	[mm]	24
Vertical deviation ring centre:	[mm]	20
Easting ring centre	[m]	871.899
Northing ring centre	[m]	510.479
Elevation ring centre	[m]	114.554

Figure 27: *Impression of a ring protocol. As-built information of an erected ring* (Source: Company VMT GmbH)

2.4.2.3 Segmental data

The necessity of a high quality for the segments was already mentioned in chapter 2.3.2. To ensure this, it requires a strict process compliance during production. Software systems help to make that possible and safe all relevant production information for a complete production documentation. This work was done with data of the product “Segment Documentation System” (SDS) of VMT GmbH. The system works with barcodes and mobile scanner units. The scanner is connected to a central database via WIFI. The fix process chain is coded to the system. Every new step will be announced to the system via scan. The software then double checks, whether that action is allowed or not. Working steps and events are saved to the database (VMT-GmbH, 2015). This can be done for the whole production chain, starting with mixing the concrete, the production, storage, delivery up to the installation of the segments in the tunnel. Figure 28 shows the connection of various production steps to a central data base.

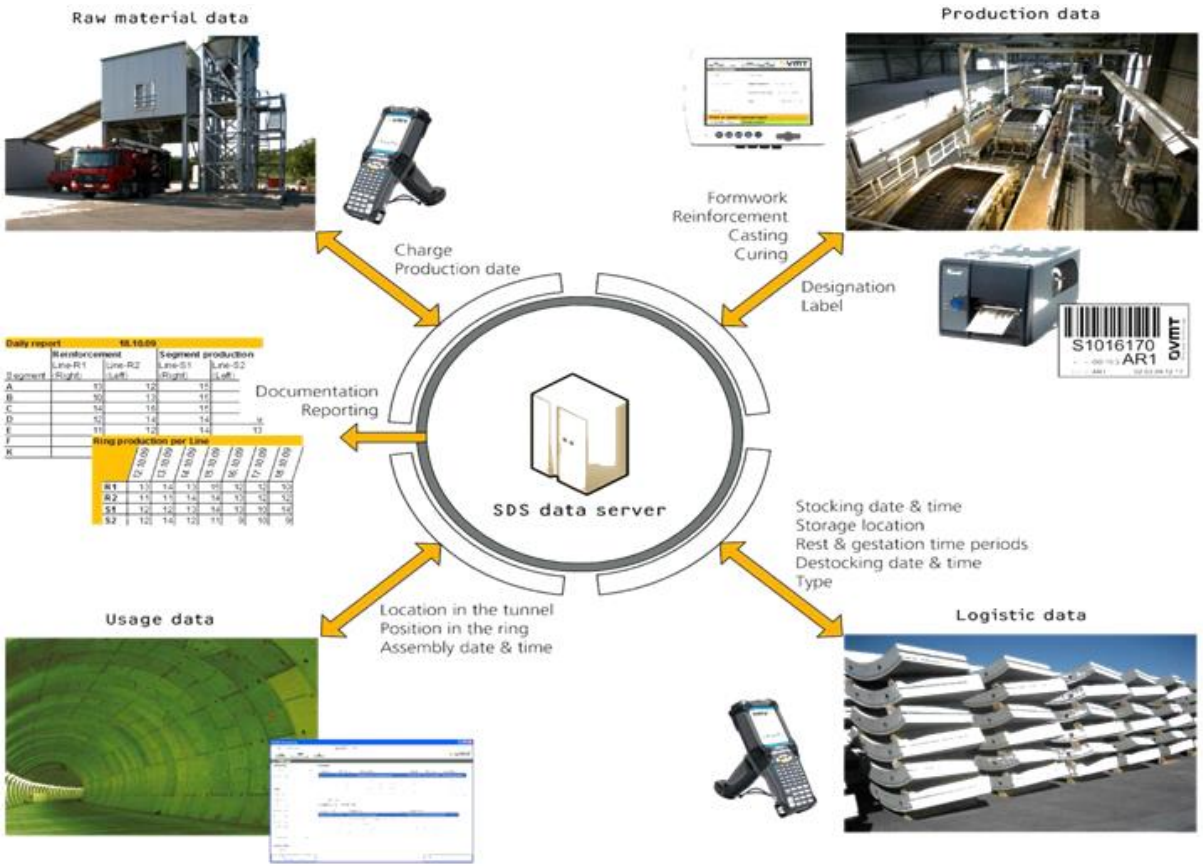


Figure 28: *Exemplary documentation circle of the SDS system* (Source: Company VMT GmbH)

The documentation “start to end” can be finished, when a further scanner unit is available on the TBM itself. The segments’ bar codes are scanned again prior to erection and the ring number to which they are assembled is assigned. By doing this a linkage of the documentation data to a ring number is given. For a potential semi-automatically generated as-built tunnel model that means, information on the quality of a segment would be available for later facility management.

3 Research Questions for the thesis

After the introduction was made to explain the context of the thesis, this chapter is meant to give an overview of the resulting research questions, which arose from a user point of view. The main research question is presented and then broken down into further steps, yet to be described in Chapter 4.

From a practical point of view, the topic as well as the implementation originates from the desire of live observation of machine data for adaption of construction processes, the so called “observation method”, which will be described in the following, as a lead-in.

3.1 The Observation Method

The German norm DIN 1054[26] of the German Institute for Standardization describes the “observation method” as a combination of the usual geotechnical analysis and calculations (prognosis) with an ongoing metrological control of the building and the foundation soil during its construction. The characterized approach means the ongoing adaption of the construction works to measurement results, especially when the prediction of the geotechnical behavior of the soil is difficult (Herten, 2012, p. 11), which often is the case especially for very elongated structures, like tunnels.

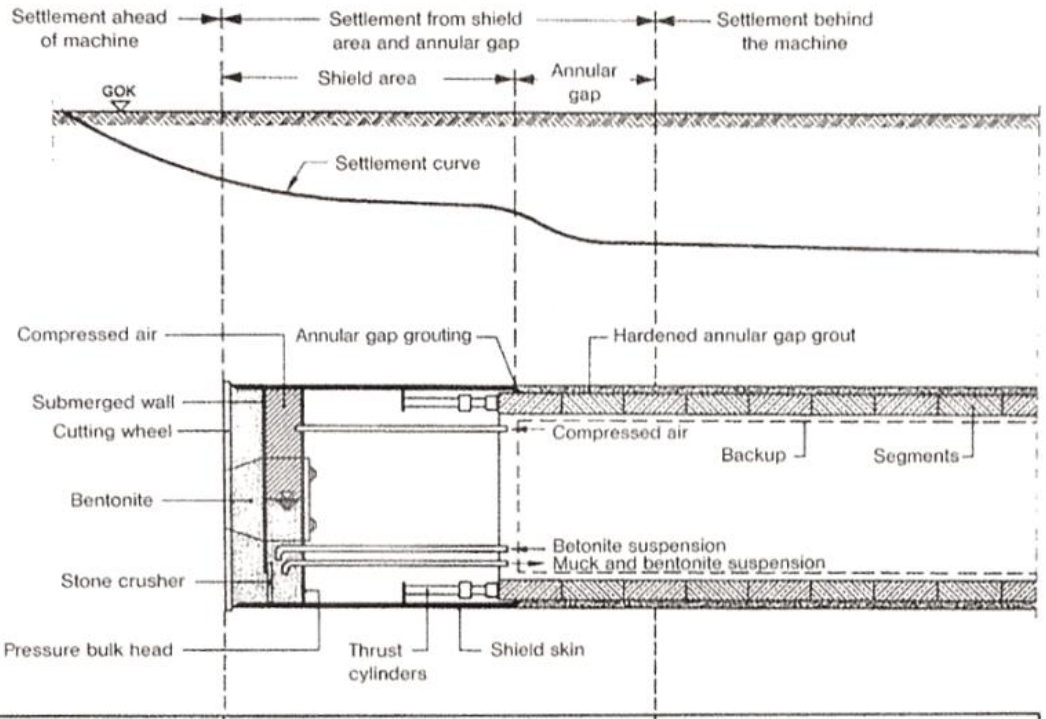
The construction of tunnel buildings requires comprehensive geological and geotechnical ground investigations during the planning phase of a project. The input parameter resulting from such investigations are used by experts during planning to simulate the behavior of the soil, when being penetrated with a TBM (Stascheit and Meschke, 2013). It is engineering methods, but also simulations like these which lead to the design and sizing of allowable machines and construction methods, as well as to the definition of threshold and alarm-values for single construction phases during excavation (advance, face support, mucking, grouting, …). Yet, those values are based on a lot of a priori assumptions.

During construction it is thus necessary to constantly check the made assumptions, to adapt the construction method to new results, based on evidence. In this connection the analysis of live gathered process data of the TBM itself, as well as from surrounding systems (geotechnical measurement systems, geodetic monitoring systems and more) comes into play. One example may be the control of parameter, which could lead to uplift and abatement of the ground level, like face pressure of the machine and grout injection pressure in the ring gap, as shown in Figure 29.

Furthermore, a lot of other parameters can and must be observed over the time of construction. Nagel (2012) as well as Maidl and Stascheit (2014) give examples for the necessary TBM process observation. This analysis is done by using systematic data evaluation, statistics or fuzzy logic approaches (Maidl et al., 2014, p. 142 ff.). This whole procedure is also recommended by the German Tunnelling Committee “DAUB” (Breidenstein et al., 2015, p. 89 ff.).

Next to the observation of sensor equipment, another important construction parameter is time: The compliance with planned construction time decides on the success of a project. Classically, the knowledge of construction time difference can only be gained with manual effort, consulting charts and documents.

[26] Full title: „DIN 1054:2010-12 Ground Verification of the safety of earthworks and foundations”



Ahead of tunnel	Shield area and annular gap	Subsequent
• Deformation and instability, overbreak at the face • Alteration of the primary stress state in longitudinal direction	**Shield area** • Overbreak and shield taper • Crushing and vibration caused by steering and driving curves • Shear stresses along the shield skin • Deformations of the shield skin • Subsidence of the machine **Annular gap** • Properties of the filing material • Deficit in the fill quantity • Deficit in the grouting pressure • Deformation of the segment tube due to loading from the thrust jacks and the backup	• Settlement due to the increased compressibility of the soil reshaped by the tunnel advance • Deformations of the segment tube due to long-term effects, e.g. creep

Figure 29: *Diagram of the interaction between the cause of settlement and the advance of the machine*, as of Maidl et al. (2014), p. 141

3.2 The Research Question

In combination of the topics tunnel construction using TBMs, process data recording and evaluation, BIM with IFC data and in reference to the described observation method, using the parameter „time“ as an example, the research question for this thesis is:

How can selected machine- and construction-process data, which are stored in Tunnel construction information systems, be utilized to be compared semi-automatically to design data from a previous construction phase? An approach for an easy tool for tunnel construction process control shall be made, with relevance to the parameter “time”. Process and design data should be compared in a simple target performance analysis. The potential of a Segmental Lining-specific extension of the IFC data model in general and the proposed “IFCTunnel” in specific shall be compared to the meta data structure, which is already gathered

by automated systems underground today. The goal is to judge the applicability of said IFC extension in a practical context.

The practical implementation to fulfill the goals and to answer the last question was based on a 3D tunnel model at hand, which was in form of an IFC file. This file was (1) imported using a BIM IFC Viewer, which parsed all IFC classes into Java classes in runtime. This was the base to (2) display the model. The software was then (3) expanded with a web interface to a tunnel construction information system, which was used to (4) read relevant building process information. This data was (5) matched to the tunnel model. (6) Using an Editor exemplarily target data was added to the model. That was the basis for (7) a graphical visualization of whether the process data adhered the design data.

The next chapter will focus on the methodology of the realized example software, which was used for a case study, described in chapter 5.

4 Methodology and implementation

The software created for the case study is based on the work of the „IFC Tools Project“, which was adapted to the need of this work. Tis software and the datasets used for the work are explained first. After that the performed software customization for this thesis is described in more detail.

4.1 Tools and data

4.1.1 The IFC Tools Project

The “IFC Tools Project” is a project fork[27] of the former, no longer maintained, “Open IFC Tools” project of a team of researchers associated with the Bauhaus University of Weimar, Germany. It is provided under the Creative Commons License “by-nc-sa 3.0[28]” and provides a “framework for accessing IFC based Building Information Models” (Tauscher and Theiler, 2013), implementable in any Java-IDE, lice Eclipse. It is written in Java 7 and complies with the early-binding paradigm, meaning that all entities of the IFC EXPRESS scheme serialized in an IFC file will be parsed to its own Java class. So it realizes the one-on-one copy of the inherently object oriented IFC class model. A programmer with knowledge of the IFC format and the ability to read visual EXPRESS schema models can then develop own applications on it. It furthermore provides a geometry package to handle different types of 3D geometry encoded to IFC files (e.g. “Solid Modeling” or “Parametric Modeling”) and a Java 3D and WebGL Viewer component.

It is provided as a JAR file from the website of the project, to be easily implementable into IDEs.

4.1.2 The BIM Tunnel Model

An exemplary IFC file containing a tunnel model was available for testing. This file was thankfully received from one of the researches of the German IFC tunnel schema adaption initiatives – thus, the influence of the “IfcTunnel Initiative” (see chapter 2.2.4) is visible in the file structure: The use of space objects is widely present. The file first had to be adopted for this work, to be able to actually load it into the IFC Tools Project software: The designed classes for IfcTunnel (e.g. classes “IfcTunnelSpace”) had to be replaced with the common “IfcSpace” classes, in order for the Java Parser to load the file. The resulting structure is exemplarily shown in Figure 10.

4.1.3 Construction Meta Data

It was the strategy to enable the software to connect to the tunnel information system software “IRIS.tunnel” (during the course of this work re-branded “VDMS”) to retrieve a meaningful set of construction meta data from an actual project. Mainly due to software bugs on the server-side VDMS software, which could not be solved during the creation of this thesis, a workaround had to be found (see chapter 4.2.2.2), which at the end ignored the live system and retrieved the desired data from a third-party service, which allows to host random data.

[27] In software engineering, a “project fork” happens when developers take a copy of source code from one software package and start independent development on it, creating a distinct and separate piece of software

[28] Creative Commons “Attribution-NonCommercial-ShareAlike 3.0”, conditions can be accessed e.g. under the URL: https://creativecommons.org/licenses/by-nc-sa/3.0/

4.1.4 Alternative approaches

The use of web-based information systems hosting near real-time information can be seen as state of the art and common practice for big tunneling projects in the construction phase. Thus, the approach to be working with those systems on the one hand and with BI-models on the other hand the author judges as “the right approach” to date. Approaches e.g. connecting to some physical sensors directly, skipping an information system (e.g. requiring the sensors to be available via an OGC Sensor Observation Service) would hold their own numerous problems (because a lot of the meta data used do not originate from physical sensors, but are either time stamps or are the result of a calculation), but are not yet available anyway.

4.2 Software Adaption

As mentioned, the software was adapted to the need of the author for this work.

Following sub-chapters first focus on the principles of software documentation and then the design and documentation of the software created for this work.

4.2.1 Software design and documentation

The final goal of the project was the creation of a middleware software, which would be in use between the two classical types of software used in the tunnel construction process: The design software and the software for construction process supervision. This is depicted in Figure 30.

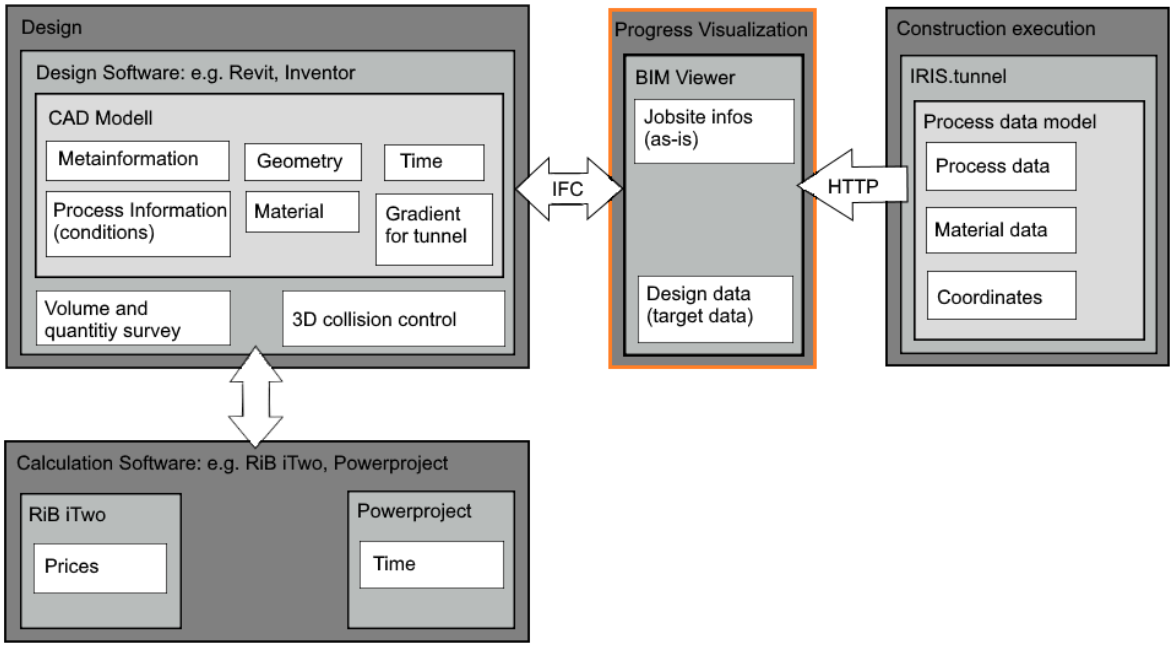


Figure 30: *Concept for the integration of data from different construction phases into a visualization context / middleware software: A design model, e.g. created with a BIM software tool like Autodesk Revit, and updated with project information using e.g. RiB iTwo, is blended with construction process information from an information system* – adapted from Hegemann et al. (2016)

To have a methodology at hand to first analyze the necessary adaption of the available software (IFC Tools Project), to plan the realization of the adaption and to be able to document it, it was tried to produce a software documentation comprising architecture and design,

according to the definition of ISO/IEC 42010 [29] ("bigger picture", the software architecture: answering "what does the software do") and IEEE 1016-2009 [30] ("smaller picture", the software design: answering "how does the software do that"). These standards structure the overall documentation into

- Architecture Description
- Architecture Viewpoints
- Concerns
- Design Views

For the description of an architecture view, several techniques are available (so called „Architecture Frameworks"). In this work the "4+1 Architectural View Model" of Kruchten (1995) was chosen [31]. As with most means of software architectural design it provides different views for the decomposition of a software, referring to Figure 31.

Kruchtens "scenario" is what is described as a "use case" in other literature: It describes the interactions of a user (an "actor") with the product and thus defines, from a practical point of view, what different users use the software for and what they expect from it.

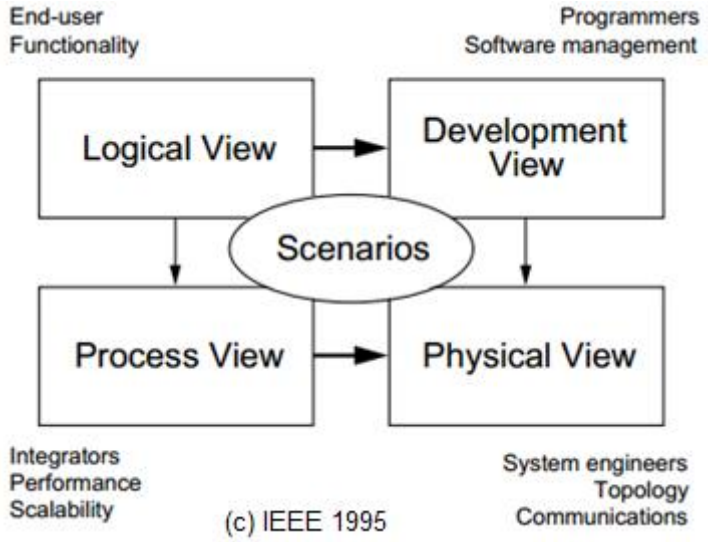


Figure 31: *The "4+1" view model*, as of Kruchten (1995)

A "view" is the decomposition of the system from a specific angle and describes the system in a more technical way. Kruchten distinguishes the four main views: Logical view, Development view, Process View and Physical view, as described in Figure 31. They are dependent on one another, as indicated by the arrows in the figure. Views are most often described using standard diagrams in a graphical notation, like the Universal Markup Language Notation (UML). Generally, the most common types of UML diagrams can be divided into "structural

[29] Full title: "ISO/IEC/IEEE 42010 – Systems and software engineering – Architecture descriptions"

[30] Full title: "IEEE Standard for Information Technology—Systems Design—Software Design Descriptions"

[31] Which is one of a vast catalog of possibilities for software architectural descriptions – and one of the older ones. E.g. May (2005) surveyed five different types (4+1, SEI, RM-ODP, Siemens and Rational ADS) of descriptions and comes to the conclusion that "The 4+1 […] model also focus on satisfying the documentation needs of the stakeholder and, as a result, also provide good coverage", yet also finds it to be "biased towards design" of software - (May, 2005, p. 9) – anticipating that the design of a software is only one of several domains.

diagrams" and "behavioral diagrams", depending on whether they show a static software structure, or dynamic software behavior. This is shown exemplarily in Figure 32.

The definition which kind of UML diagrams are useful in the project context and to which extend always varies with the project itself, but also the stakeholders and their information needs.

Structural Diagrams	Behavioral Diagrams
Class Diagram – set of classes and their relationships. Describes interface to the class (set of operations describing services)	**Use Case Diagram** – high level behaviors of the system, user goals, external entities: actors
Object Diagram – set of objects (class instances) and their relationships	**Sequence Diagram** – focus on time ordering of messages
Component Diagram – logical groupings of elements and their relationship	**Collaboration Diagram** – focus on structural organization of objects and messages
Deployment Diagram – set of computational resources (nodes) that host each component	**State Chart Diagram** – event driven state changes of system
	Activity Diagram – flow of control between activities

Figure 32: *Distinction between structural and behavioral UML diagrams*, adopted from Kraemer (2014)

Some of these diagrams where used for this work and will be demonstrated in the following, as part of the documentation of the created software.

The following subsections are used to make a brief architectural documentation as proposed e.g. by ISO 42010, to make a distinction of and briefly describe the architecture, viewpoints, concerns and views. The content of what is documented is first of all relies on the stakeholder´s demands. For this small thesis project, the only defined stakeholder of this software project is the end-user[32].

4.2.1.1 Architecture Description

An „architecture description" to some extent is the „leading document" of a project documentation, which mostly verbally describes use cases of a software and the views onto the architecture. In context of this work, this whole thesis and the derivation of the software project itself can be seen as the architecture description, which is why it will not be explicitly written down again for this subsection.

An architectural description mostly works with some generic concepts, which have evolved over time – one of which is the already mentioned 4+1 View Model of Kruchten. The relationship of basic terms (which will be used in the following subsections) is described in Rozanski and Woods (2005) and shown in Figure 33.

[32] For bigger software systems the generic category of „user" will likely be too general, as different groups of users (managers, accountants, engineers, …) want to take different results out of a software. For bigger scenarios the "users" would have to be divided into different groups.

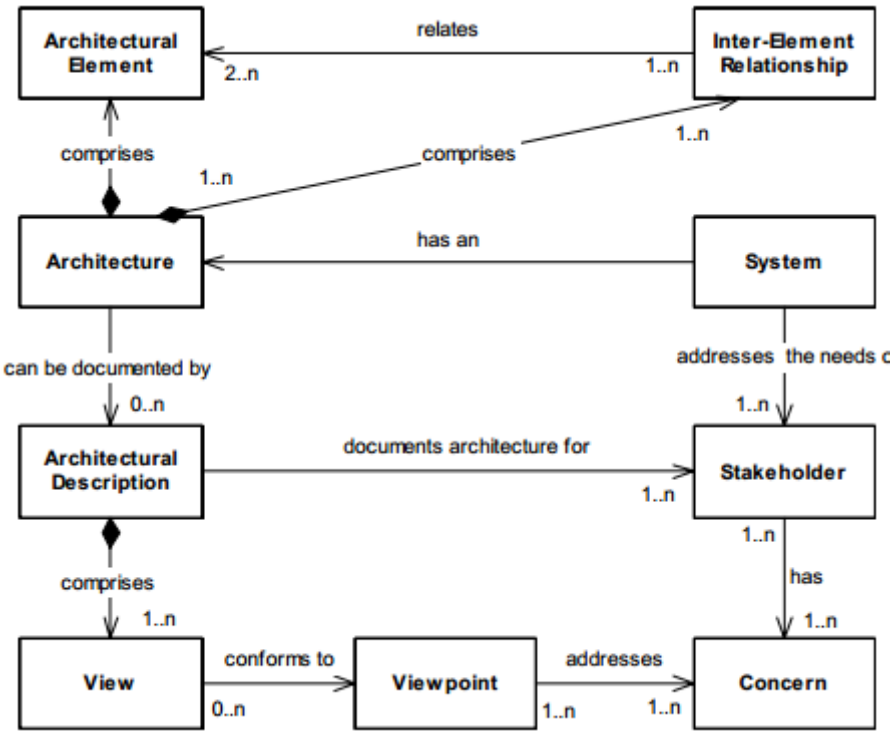


Figure 33: *Architectural concepts and relationships* - taken from Rozanski and Woods (2005), p. 6

4.2.1.2 Architecture Viewpoint

The content of each view (see subchapter 4.2.1.5 ff.), that means what system structure or behavior is described, furthermore could vary depending on the viewpoint –this is also true for the means of description: a system can be viewed e.g. from a

- "functional viewpoint": what elements does the system have, which interfaces to external software? What is the internal structure, based on data handling etc. So the "process view" of the 4+1 view model could show the procedural interaction of the software as a whitebox, specifying the software-internal behavior, e.g. using UML sequence diagrams.
- "context viewpoint": "Describes the relationships, dependencies, and interactions between the system and its environment (the people, systems, and external entities with which it interacts)" (Rozanski and Woods, 2011, p. 41). So the "process view" of the 4+1 view model could show the procedural interaction of the software as a blackbox dealing with input from external resources, e.g. using UML communication diagrams.

As a consequence, the type of description used as a means of visual communication in a view, e.g. a component model in UML notation, could or must look different depending on the viewpoint.

The IEEE 1016-2009 defines 12 viewpoints[33], other literature a smaller catalog: Rozanski and Woods (2011) e.g. only differentiate between 7 viewpoints[34]. As the software modification for this work is rather small and also because it is mainly the aim to document the functional

[33] Viewpoints: context, composition, logical, dependency, information, patterns use, interface, structure, interaction, state dynamics, algorithm and resource – taken from: IEEE (2009)

[34] Context, functional, information, concurrency, development, deployment, operational

behavior of the made changes, the documentation was made only using the functional viewpoint.

4.2.1.3 Concerns

The stakeholder of a software project has an objective or an intention, when it comes to dealing with said piece of software and its documentation. Generically that is defined to be a “concern”. Concerns with the software documentation must be mapped and taken care of in the architecture description to make sure that all relevant aspects of the software were considered in the design.

Different concerns are addressed by different viewpoints. Based on the chosen functional viewpoint typical concerns would be functional capabilities, details about external interfaces and the internal structure. These were tried to be taken into account.

4.2.1.4 Architecture View: Scenarios

What is described as „scenarios“ are the use cases of a software system, in which it is used. It also shows the explicit demarcation to other software products – in this work being the Tunnel Construction Information System, which was queried. The use cases where modeled using the “Papyrus” Plugin in the Eclipse IDE and using UML as the graphical notation. The result is shown in Figure 34:

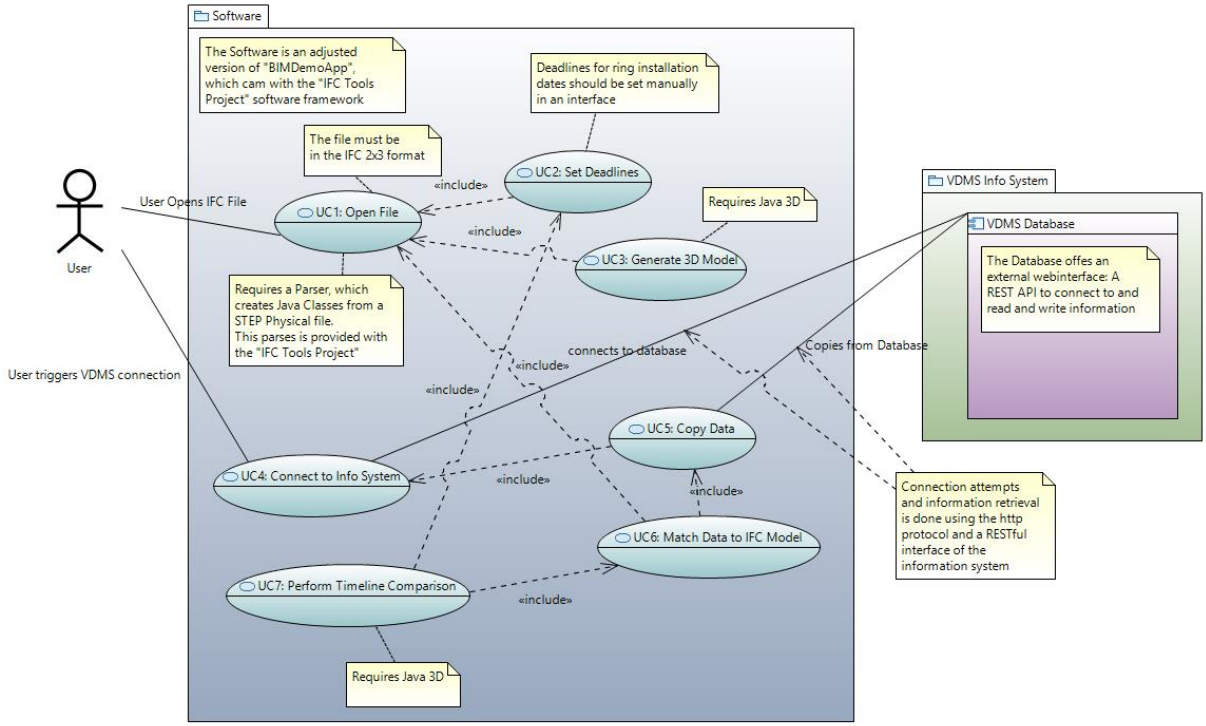


Figure 34: *Use Case Scenarios for the project*

Each use case scenario can be described more detailed with a “user story”, using a simple tabular pattern. The user story for each use case will be shown in the following Table 1 till Table 7, next to their graphical representation as UML activity diagrams.

Use Case 1 – Open File	
Use Case Name	Open File
Author	ROL
Last Revision	18.08.2016
Actors	User
Uses	-
Extends	-
Pre-Condition	• IFC Model file is a STEP file • File complies with format IFC 2x3
Main Scenario	1. User opens Software 2. User opens the file dialog and selects a file for import 3. The file is imported and put through the software-internal STEP parser, Java classes are generated according to the IFC EXPRESS schema
Other Scenarios	-
Post-Condition	File was opened, parsed and read in and ready for display. UC 2 takes over from here
Error-Condition	3: The software must identify whether or not the selected file is a valid IFC file. If not, the user is informed.

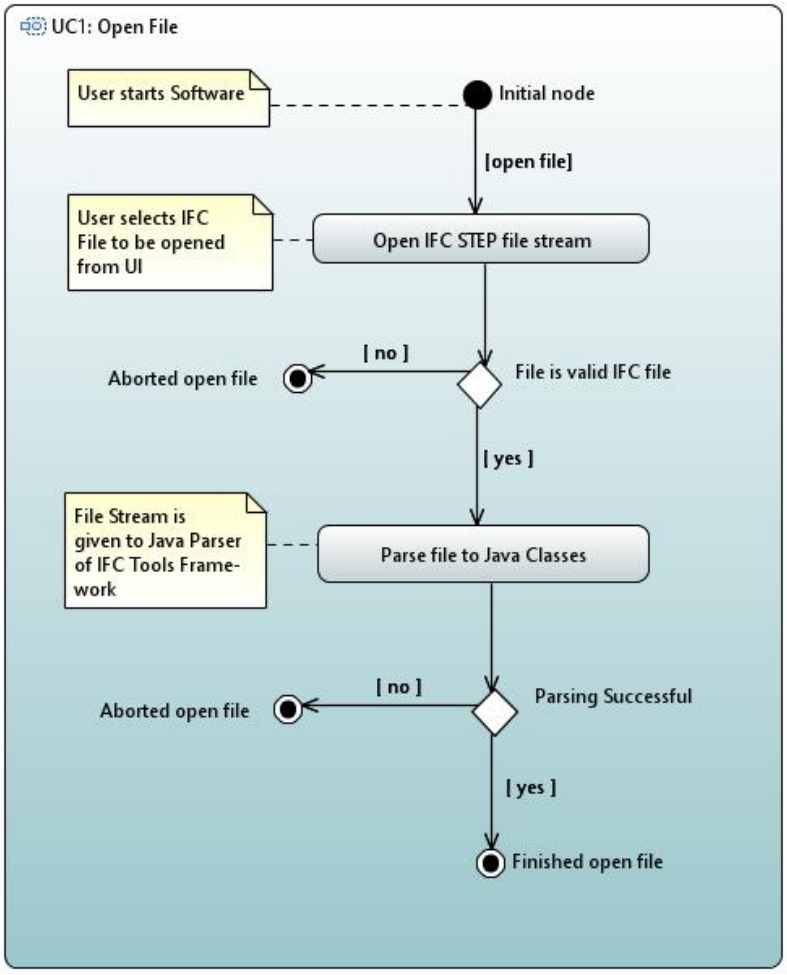


Table 1: *Use Case 1 - Open File*

Use Case 2 – Set Deadlines	
Use Case Name	Set Deadlines
Author	ROL
Last Revision	18.08.2016
Actors	User
Uses	UC 1 - Open File
Extends	-
Pre-Condition	• Reading in and parsing the opened IFC file happened correctly, the file was not corrupted
Main Scenario	1. The user opens the menu “set deadlines” 2. In a table view the user can input time-based deadlines for tunnel rings to be installed at certain dates 3. These deadlines are matched to the rings as a new attribute to the imported tunnel model
Other Scenarios	
Post-Condition	The imported model was enriched with further attributes on ring-level. A general attribute at model-level indicates that the function “UC2” was already performed on this model.
Error-Condition	Validity checks ensure that the assigned dates are ascending. Otherwise a match of the dates to the model is not performed.Validity checks ensure that the assigned dates are ascending. Otherwise a match of the dates to the model is not performed.

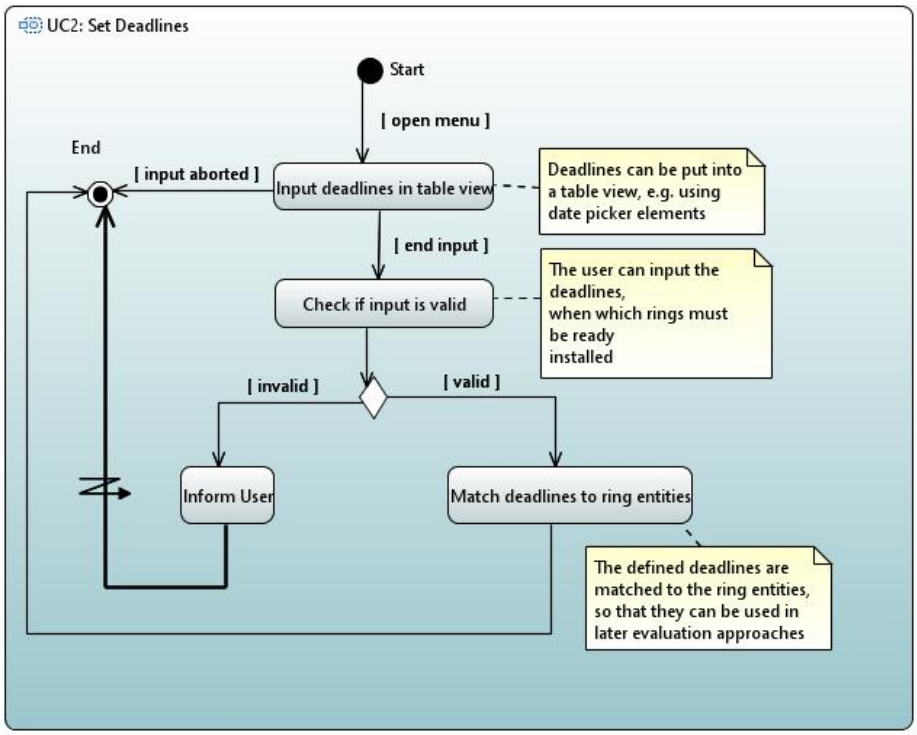


Table 2: *Use Case 2 - Set Deadlines*

Further thinkable functions to e.g. later on delete or change made settings per ring would be needed from a user perspective, but were not implemented for this work.

Use Case 3 – Generate 3D Model	
Use Case Name	Generate 3D Model
Author	ROL
Last Revision	18.08.2016
Actors	User
Uses	UC1 – Open File
Extends	-
Pre-Condition	• Reading in and parsing the opened IFC file happened correctly, the file was not corrupted • The computer has installed Java 3D libraries available
Main Scenario	1. The graphical generator has access to the generated Java classes 2. The 3D model is generated automatically by the software and displayed to the user
Other Scenarios	
Post-Condition	The 3D model was generated and displayed to the user.
Error-Condition	

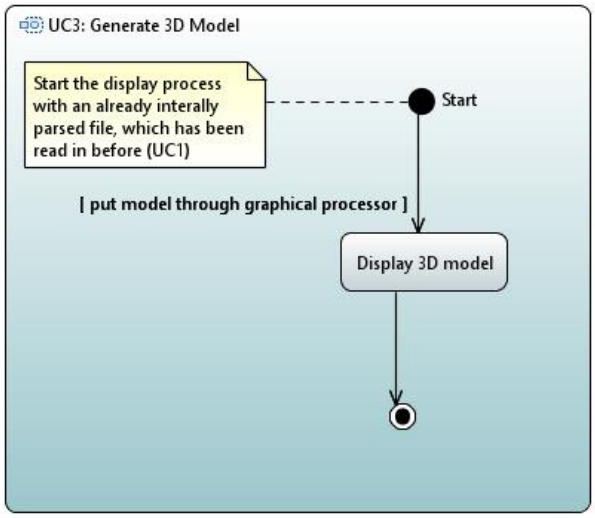


Table 3: *Use Case 3 - Generate 3D Model*

Use Case 4 – Connect to Information System	
Use Case Name	Connect to Information System
Author	ROL
Last Revision	18.08.2016
Actors	User
Uses	-
Extends	-
Pre-Condition	• Internet connection available • Necessary user credentials to access the information system are available
Main Scenario	1. The user opens the menu „VDMS Settings“ and types in the necessary user credentials, to identify him at the information system 2. The user clicks “Connect to Information System” and a connection attempt is made by the software using the HTTP protocol
Other Scenarios	
Post-Condition	The Information System delivers an response, containing an HTTP response code and JSESSIONID for use in further transactions.
Error-Condition	The Connection can fail due to • Wrong server address • Wrong user credentials The user then is warned

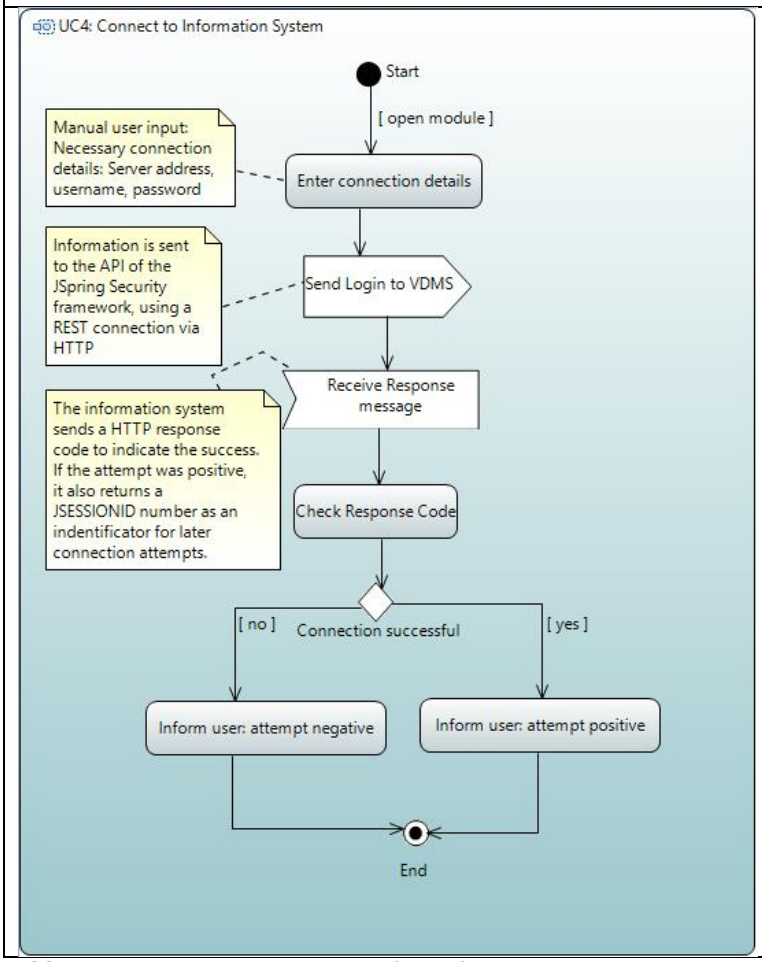


Table 4: *Use Case 4 - Connect to Information System*

Use Case 5 – Copy Data from Information System	
Use Case Name	Copy Data from Information System
Author	ROL
Last Revision	18.08.2016
Actors	User
Uses	UC 4 – Connect to Information System
Extends	-
Pre-Condition	• A connection to the information system was already successfully established
Main Scenario	1. The User clicks “Copy Data”. The list of either available ring or segment meta data is requested from the server 2. The server-side sends the list, encrypted as a JSON string 3. The received list must be processed. The software checks whether the string contains ring data or segment data 4. The string is parsed in a loop. The information is assigned to an object of a temporal data structure
Other Scenarios	
Post-Condition	• The Information system send the response status code “200” for a full transfer • The retrieved JSON string is parsed into a temporal data structure internally and ready for further usage.
Error-Condition	If the response code “200” did not arrive, the transfer was erroneous

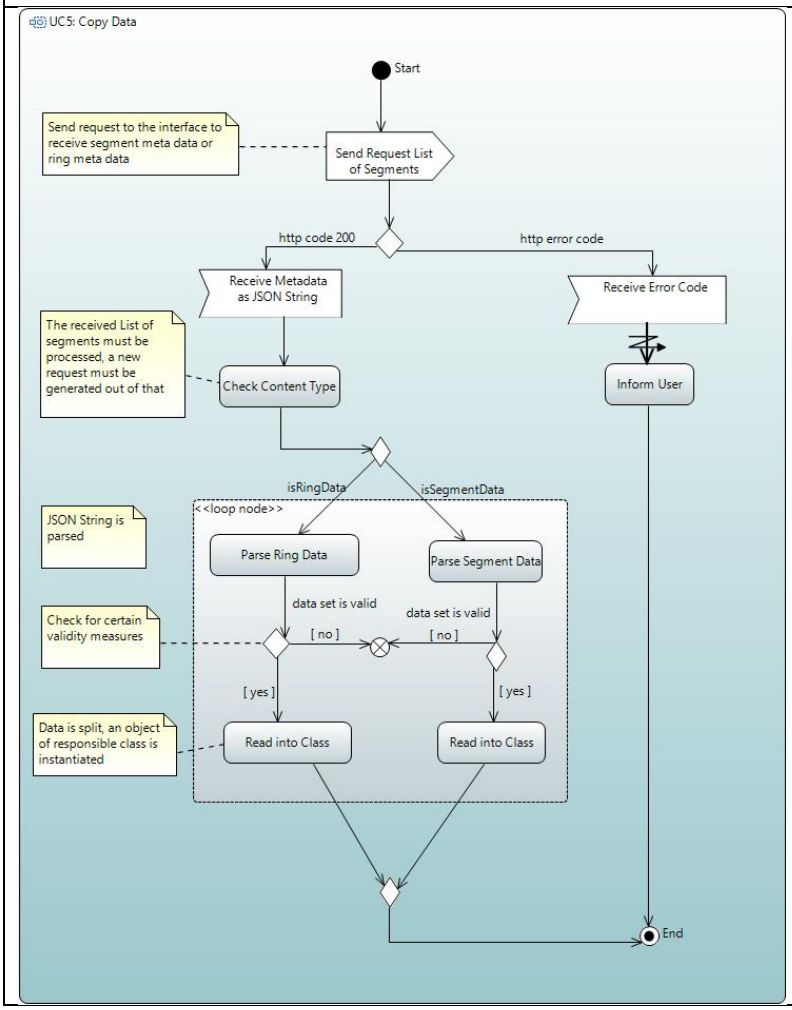


Table 5: *Use Case 5 - Copy Data from Information System*

Use Case 6 – Match Data to IFC Model	
Use Case Name	Match Data for IFC Model
Author	ROL
Last Revision	18.08.2016
Actors	User
Uses	UC 1 – Open File UC 5 – Copy Data
Extends	
Pre-Condition	• An IFC file was already successfully imported • Data from a tunnel information system, meeting the syntax requirements, was already copied and imported successfully. If that step was performed correctly
Main Scenario	1. The user opens the menu „match data“ 2. The system uses the already opened IFC tunnel model and the retrieved information from the tunnel information system and matches them onto one another: attribute information from VDMS are saved to the segments of the tunnel
Other Scenarios	
Post-Condition	The information has been matched to the model. The button relevant for Use Case 7 is enabled It is possible to load meta data to the bottom text panel of the window, for each invidiual element (Tunnel segment or tunnel ring) when highlighting it, either in the 3D widget or the table view
Error-Condition	If the matching fails the button turns red. The button for UC7 stays disabled.

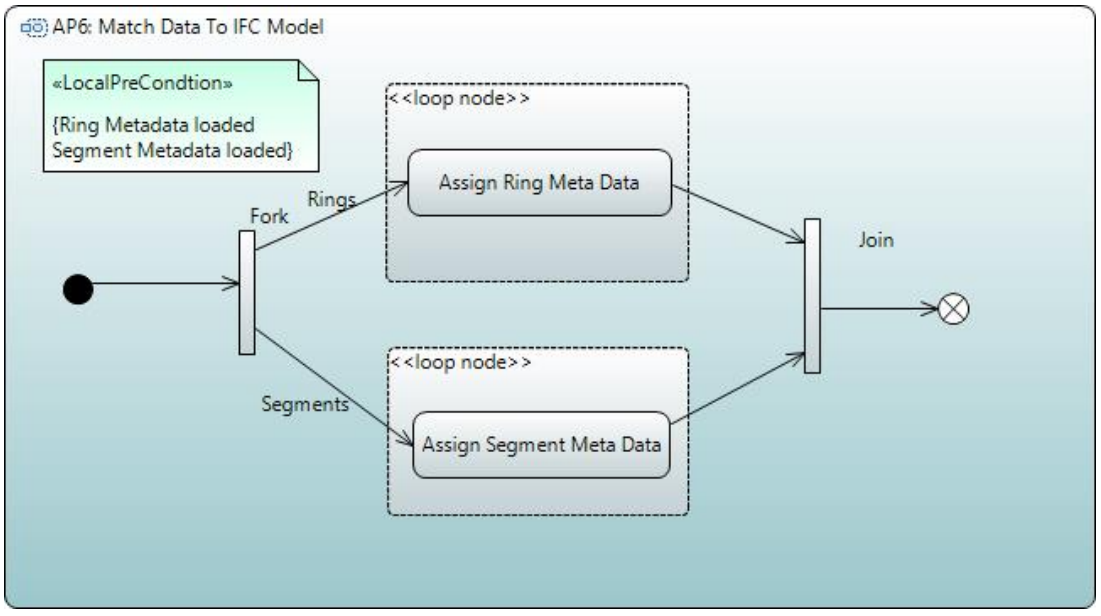


Table 6: *Use Case 6 - Match Data to IFC Model*

Use Case 7 – Perform Timeline Comarison	
Use Case Name	Perform Timeline Comparison
Author	ROL
Last Revision	03.12.2016
Actors	User
Uses	UC 2 – Set Deadlines UC 6 – Match data to the IFC Model
Extends	-
Pre-Condition	• Time-based deadlines have already been matched to the imported model at ring-level • Information system data from VDMS has already been successfully matched to a previously imported tunnel model • The computer has installed Java 3D libraries available
Main Scenario	1. The user opens the menu „Display Timeline“ 2. The set installation deadlines at the model are compared with the retrieved real installation dates. The offset in days is calculated, stating whether the ring was installed on time or not 3. A window is opened, showing the offset per ring with an additional color coding - Green: installation within deadline - Red: Deadline missed
Other Scenarios	
Post-Condition	
Error-Condition	

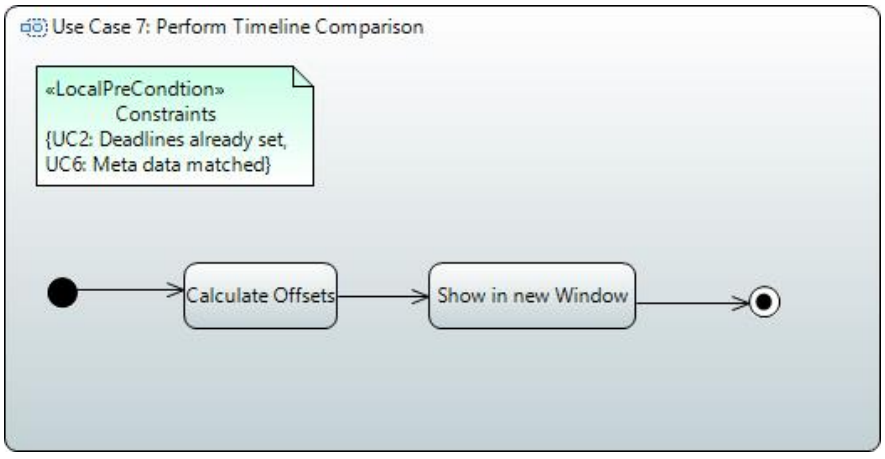


Table 7: *Use Case 7 – Perform Timeline Comparison*

4.2.1.5 Architecture View: Design View

The design view describes the logical structure of the system and could consist of several types of involved entities, like persons, programs, interfaces, input and more. For the fairly easy program created for this work, the documentation for this view focusses on the created classes. Figure 35 shows and example of a class diagram for the class “RestResponseObjectCreator” and its related classes. The graphs were reverse engineered from the created source code, using the Eclipse IDE plugin “ObjectAid” to create UML class diagrams (ObjectAid-LLC, 2016). This one and all else class diagrams are documented in the Appendix.

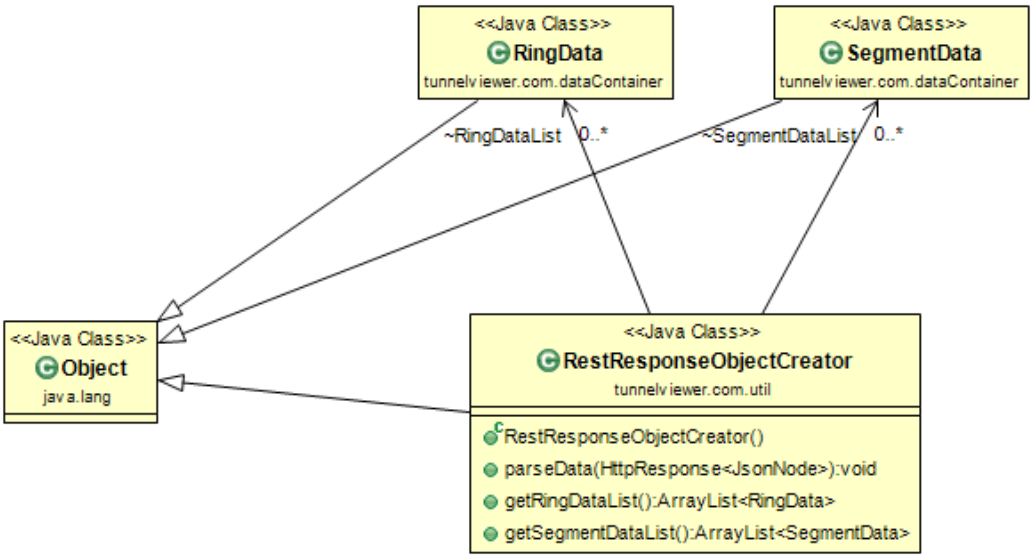


Figure 35: *Class "RestResponseObjectCreator" as the one responsible to parse incoming JSON-Strings from the information system and to create entities of objects of type "RingData" or "SegmentData", depending on the type of information which has been queried online*

Using the documentation available in Appendix it is also apparent that for the sake of this demonstrational work the names of the fetched attributes are hard coded into the software structure – a behavior which for a practical use case must be more flexible and should thus be changed.

Next to class diagrams, also UML component diagrams are a helpful tool to document the logical behavior of the software, which is documented in Figure 36 and Figure 37.

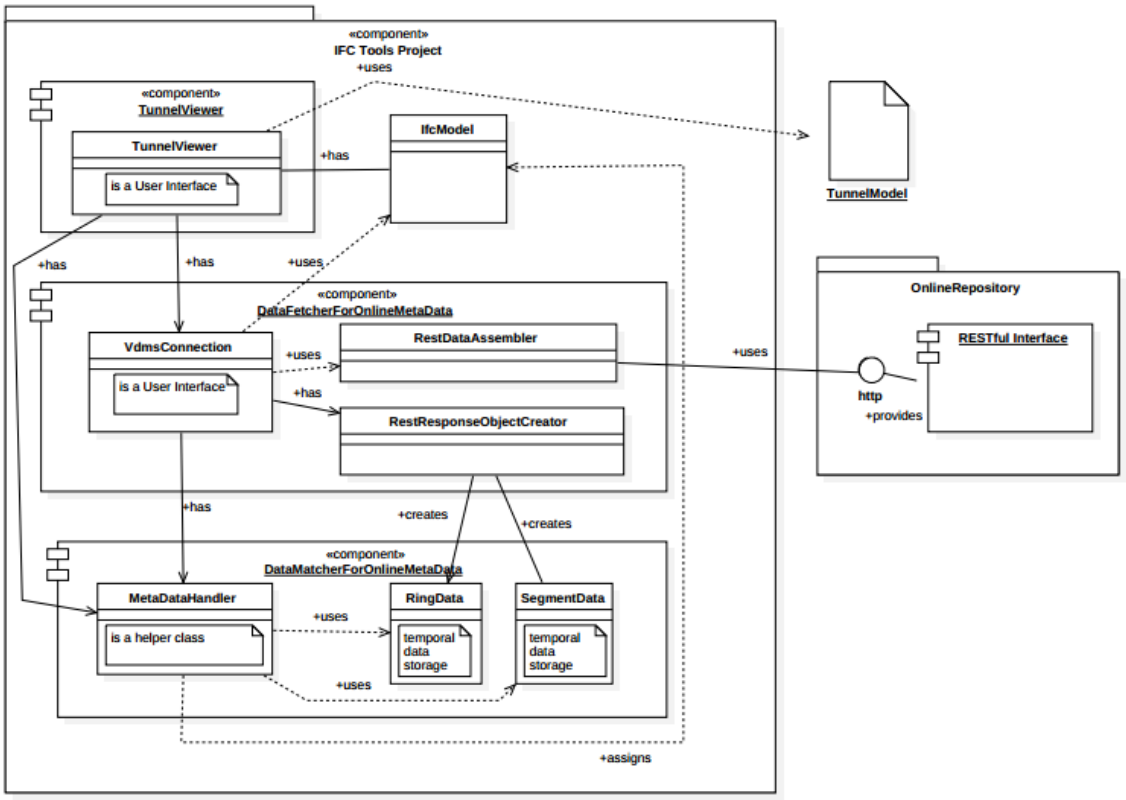


Figure 36: *Fetch and match online Meta Data: UML component diagram*

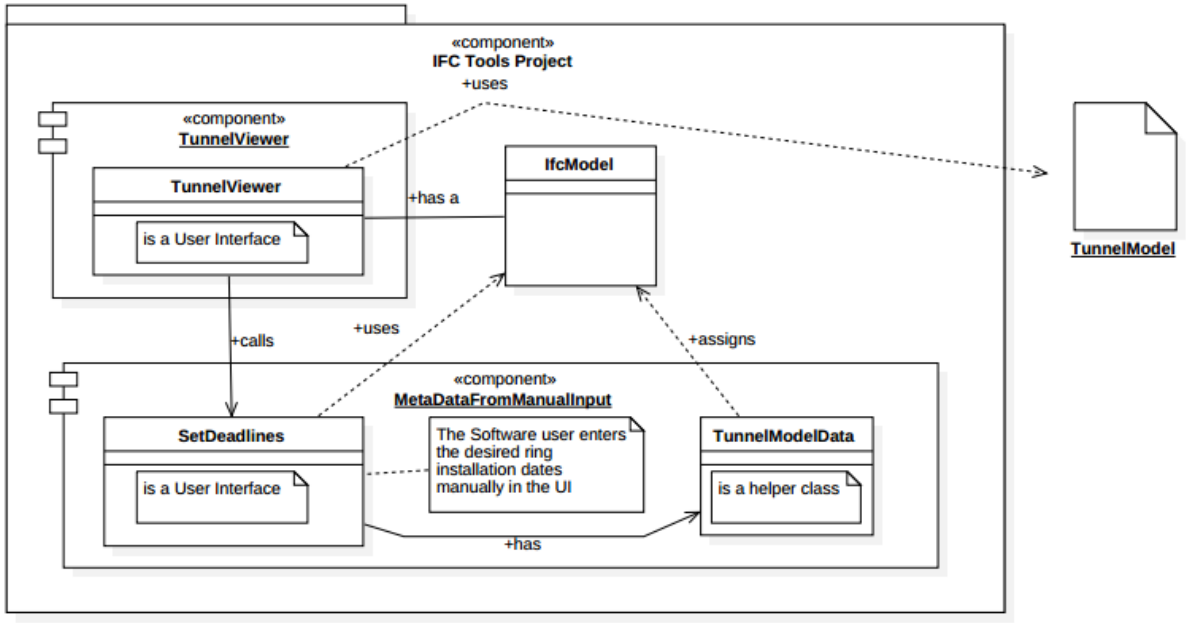


Figure 37: *Get and match ring installation dates from user input*

The component diagrams were created using the tool “StarUML”.

4.2.1.6 Architecture View: Development View

The development view could add up onto the developed component diagrams, to furthermore describe the component-internal behavior, following a whitebox paradigm. That e.g. would make sense for applications in a layered architecture, like server-side applications (Database, Application layer, services layer and graphical user interface). As the created software for this work is not that complex, it was decided that the documentation for this view will not be created, as it would not add to the understanding.

4.2.1.7 Architecture View: Process View

The process view documents the system, like it behaves during runtime. Either for single threaded or multi-threaded applications, explanations using e.g. UML sequence diagrams are useful. As the software created for this work is single threaded and not runtime sensible, it was decided that sequence diagrams will not add to the understanding substantially, which is why it was decided to leave those out.

4.2.1.8 Architecture View: Physical View

The physical view describes the used components in terms of either hardware or resources necessary to run a software. For the software created for this work, the main hardware and software components are shown in Figure 38.

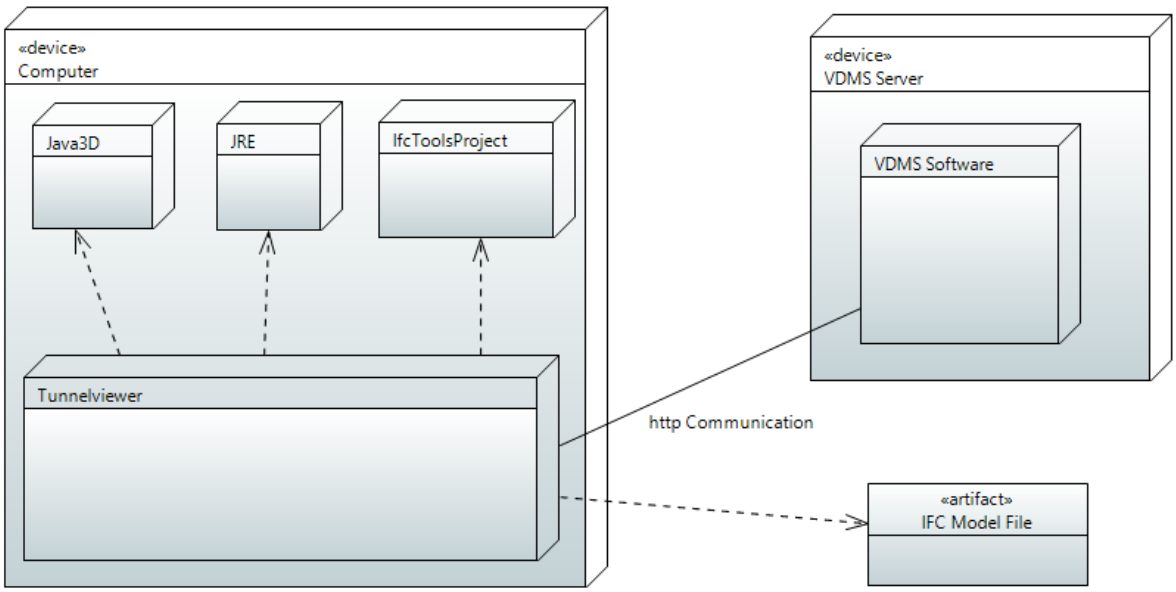


Figure 38: *Deployment view, showing all involved components of the created software. The tunnel viewer application builds up on the IFC Tools Project and uses Java as a programming language, which is why also the Java Runtime Environment (JRE) is required. The 3D widget of the application makes use of Java3D. The application communicates with another server-side software and takes a single IFC model file as an input.*

4.2.2 Methodology and Implementation

To reach the before mentioned software adaption, a set of tools, frameworks and techniques where used.

- Programming was done in Java, Version 7 using the “Eclipse” IDE
- The graphical user interface was created using the Eclipse “Window Builder”
- UML diagrams for the documentation were created using the “Papyrus” plugin for Eclipse and the additional tool “StarUML”
- IFC files were read, parsed to Java and displayed using the “IFC Tool Project”, as described in chapter 4.1.1.
- Data from the Tunnel Information System were queried using JSON strings via a RESTful interface. The necessary Java REST library was “UniRest”.

Parts of this toolset will either not be explained in more detailed (Eclipse, Window Builder and UML tools), or was already referred to (IFC Tools Project – see chapter 4.1.1.) Just the latter, RESTful connections using UniRest, shall furthermore be explained in the following.

4.2.2.1 RESTful connections for data queries using JSON encoding

Figure 30 already describes the idea (and it was also mentioned before) of the software to create to fetch data from an information system. Said information system is web-based and thus hosted on a web server. The internet mainly works on the HTTP protocol, to send and retrieve information. Requests to web servers use the HTTP protocol as a base layer, so to say, to send the explicit request. The server-based software provides collaborating software with Application Programming Interfaces (APIs), for software developers to use as documented, secured and standardized connection points to said server-based software. Clients can then “consume” these interfaces or “resources”, to request and deliver information. In a modern, most common approach these resources shall be “RESTful”. REST is an acronym

for “REpresentational State Transfer”, which prescribes that resources shall have, amongst others:

- A unique address (URI) to connect to
- A uniform interface, which is documented, which offers methods to use it (e.g. the HTTP-methods “GET”, “POST”, “DELETE” and more) and standardized representations of requested data, like offering returned data e.g. XML-Encoded or JSON-Encoded. The latter format was used for this work[35].
- Stateless: “A stateless protocol is a communications protocol that treats each request as an independent transaction that is unrelated to any previous request so that the communication consists of independent pairs of request and response” (Wikipedia, The Free Encyclopedia (2016)) Like that, each sent request is self-contained.

The requested resource, the tunnel construction information system, holds a “data services layer”, which offers availability of data to other consumers of other layers, within the same software, or to external consumers, as shown in Figure 39.

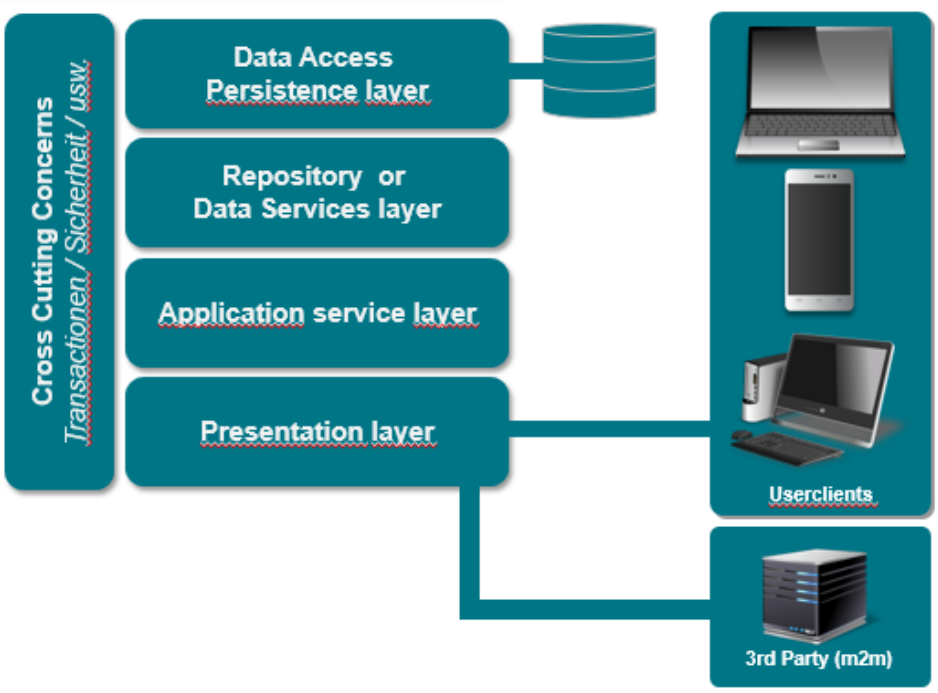


Figure 39: *Layer structure of the tunnel construction information system: The Data Services Layer offers a standardized access to resources* (ITC Engineering, Internal material)

APIs can allow every consumer to connect and query information, or restrict it to allowed users. In the case of this work, all resources of said information system are secured with user name and password. When logging into a service, these are cross-checked on the server-side against the user repository. If the user is identified, the server sends a valid “Session ID[36]”, which is saved in the client browser as a cookie. Every further query to the server must be

[35] JSON stands for “JavaScript Object Notation” and is a human-readable data format, which is widely used over the internet to transmit and receive data. Attributes and their values are delivered in attribute-value pairs, which makes it easy to read and parse.

[36] On the server, each user logging into the software opens their own session. When using a stateless communication protocol like HTTP, the server has to verify each and every request for its legitimacy. That is why after logging in, each user gets assigned a unique session ID, which can be used for further requests. The server side can also restrict the validity of the ID on a timely basis, e.g. used in online banking application: After the validity is elapsed, the user is automatically logged out.

send with the Session ID in the header of the request, to again identify the user and thus the validity of the request – the reason for that is before explained statelessness: Each request is self-contained and could thus come from random sources, as shown in Figure 40.

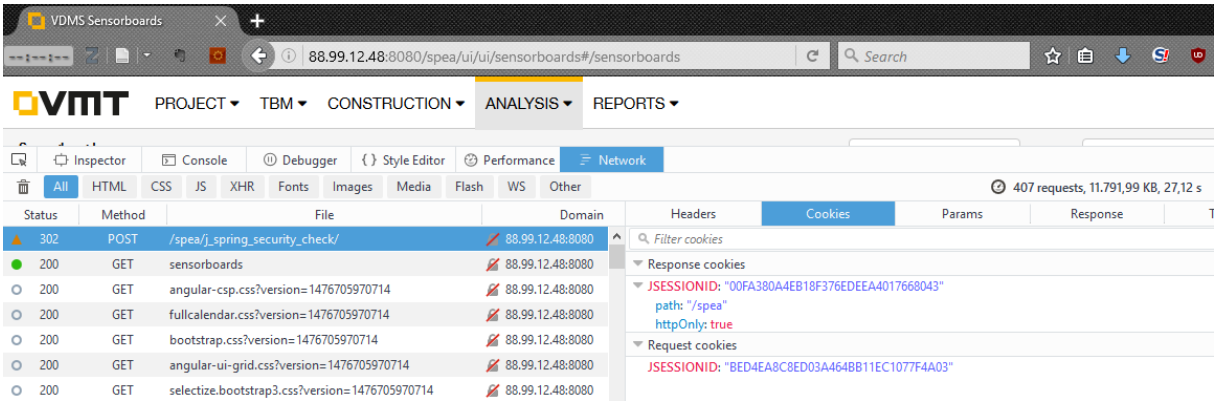


Figure 40: *JSESSIONID object retrieved from an HTTP GET-Request to the tunnel information system server, traced in the Object Inspector of the Firefox Web Browser*

Connecting to the software, thus an easy connection can be made using simple user name and password, as shown in Figure 41.

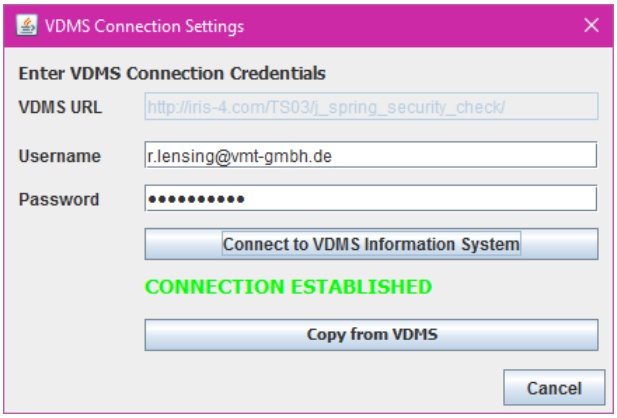


Figure 41: *Showcase login screen for BIM tunnel viewer software*

Yet, some problems occurred during the realization of this approach – basically because of software bugs on the server-side software, which is why a workaround had to be found. This was done merely for demonstrational purposes, for this work. This attempt is explained in the next section.

4.2.2.2 RESTful workaround

Two problems were identified trying to realize the desired approach, which lead to the decision that a different solution is required:

- The information system had software bugs, so that sent valid requests where not answered with the desired data
- The information system was architecturally furthermore missing a single resource, which could have delivered all required information in one reply. Even though also the reply of more than one request could have been parsed together into the desired outcome, it was alternatively used as an opportunity to design such single reply – the

behavior of the software in that step thus is different than desired and explained in Table 4.

The data was hosted on an alternative free resource online (www.myjson.com). Two files were designed and hosted on different URL. These shall be explained in the following. The first file contained information, which was needed per tunnel ring entity. An example is given for ring number 1 in Table 8:


```
{
    "Ring No.": 1,
    "Date and Time (Before Installation)": "12.10.2015 14:41",
    "Date and Time (After Installation)": "20.10.2015 08:31",
    "Tunnelmeter Ring [m]": "-11,32",
    "Chainage Ring [km]": "72+985,32",
    "Chainage Cutting Edge [km]": "72+975,78",
    "Ring Type": "TN/N3",
    "Offset HZ Ringbuild End [m]": "36,1",
    "Offset VT Ringbuild End [m]": "-8,66",
    "Ring Orientation": 13
  }
```


Table 8: *Example for a JSON String delivered as a response to requesting ring metadata*

This set of metadata gives information on the ring installation (from the navigation system of the TBM – refer to chapter 2.4.2.2) in terms of ring type, time, position and offset to the designed position and alignment.

The second file contained information, which were needed per concrete segment entity, eight entries per tunnel ring, one per segment. An example is given for ring number 1 in Table 9:


```
  {
    "Status": "Installed",
    "Ring-Nr.": 1,
    "Segment Id": 10002,
    "Segment": "A2",
    "Reinforcement": "TN/N3",
    "Installation Date":
       "2015-10-23 21:02:00",
    "Approval Contractor": "X, Johann",
    "Approval Client": "X, Siegfried",
    "Ring Type": "TN/N3"
  },
```

```
  {
    "Status": "Installed",
    "Ring-Nr.": 1,
    "Segment Id": 10013,
    "Segment": "B",
    "Reinforcement": "TN/N3",
    "Installation Date":
       "2015-10-23 21:02:00",
    "Approval Contractor": "X, Johann",
    "Approval Client": "X, Siegfried",
    "Ring Type": "TN/N3"
  },
```

```
  {
    "Status": "Installed",
    "Ring-Nr.": 1,
    "Segment Id": 10446,
    "Segment": "A4",
    "Reinforcement": "TN/N3",
    "Installation Date":
       "2015-10-23 21:02:00",
    "Approval Contractor": "X, Johann",
    "Approval Client": "X, Siegfried",
    "Ring Type": "TN/N3"
  },
```

```
  {
    "Status": "Installed",
    "Ring-Nr.": 1,
    "Segment Id": 10472,
    "Segment": "C",
    "Reinforcement": "TN/N3",
    "Installation Date":
       "2015-10-23 21:02:00",
    "Approval Contractor": "X, Johann",
    "Approval Client": "X, Siegfried",
    "Ring Type": "TN/N3"
  },
```

```
  {
    "Status": "Installed",
    "Ring-Nr.": 1,
    "Segment Id": 10539,
    "Segment": "A1",
    "Reinforcement": "TN/N3",
    "Installation Date":
       "2015-10-23 21:02:00",
```

```
  {
    "Status": "Installed",
    "Ring-Nr.": 1,
    "Segment Id": 10541,
    "Segment": "A3",
    "Reinforcement": "TN/N3",
    "Installation Date":
       "2015-10-23 21:03:00",
```

<table>
<tr>
<td><pre>
    "Approval Contractor": "X, Johann",
    "Approval Client": "X, Siegfried",
    "Ring Type": "TN/N3"
  },
</pre></td>
<td><pre>
    "Approval Contractor": "X, Johann",
    "Approval Client": "X, Siegfried",
    "Ring Type": "TN/N3"
  },
</pre></td>
</tr>
<tr>
<td><pre>
  {
    "Status": "Installed",
    "Ring-Nr.": 1,
    "Segment Id": 10798,
    "Segment": "K",
    "Reinforcement": "TN/N3",
    "Installation Date":
       "2015-10-23 21:03:00",
    "Approval Contractor": "X, Johann",
    "Approval Client": "X, Siegfried",
    "Ring Type": "TN/N3"
  },
</pre></td>
<td><pre>
  {
    "Status": "Installed",
    "Ring-Nr.": 1,
    "Segment Id": 12730,
    "Segment": "S",
    "Reinforcement": "S/C35/45",
    "Installation Date":
       "2015-10-29 02:34:00",
    "Approval Contractor": "X, Johann",
    "Approval Client": "X, Siegfried",
    "Ring Type": "S/C35/45"
  }
</pre></td>
</tr>
</table>

Table 9: *Example for a JSON String delivered as a response to requesting segment metadata*

This set of metadata gives information about each segment, originating from a segment management system (refer to chapter 0).

The content for both files were taken from real projects, but anonymized. The usability of the specific choice of attributes selected for these files can of cause be discussed. Depending on the project context and on the target software system to work with that information, the metadata set could vary.

Mentioned files remain online and can be viewed in a normal web browser:

- Ring information: https://api.myjson.com/bins/2gd34
- Segment information: https://api.myjson.com/bins/4ydts

An approach like this, on the other hand, delivers an already full data set which might not reflect a standard behavior of such information system, where e.g. a user would dynamically filter the rings of interest, as depicted in Figure 42. Yet, to have a simplified solution just for this work, the pre-defined filter results (one reply for all rings / all segments) where hosted online pre-configured.

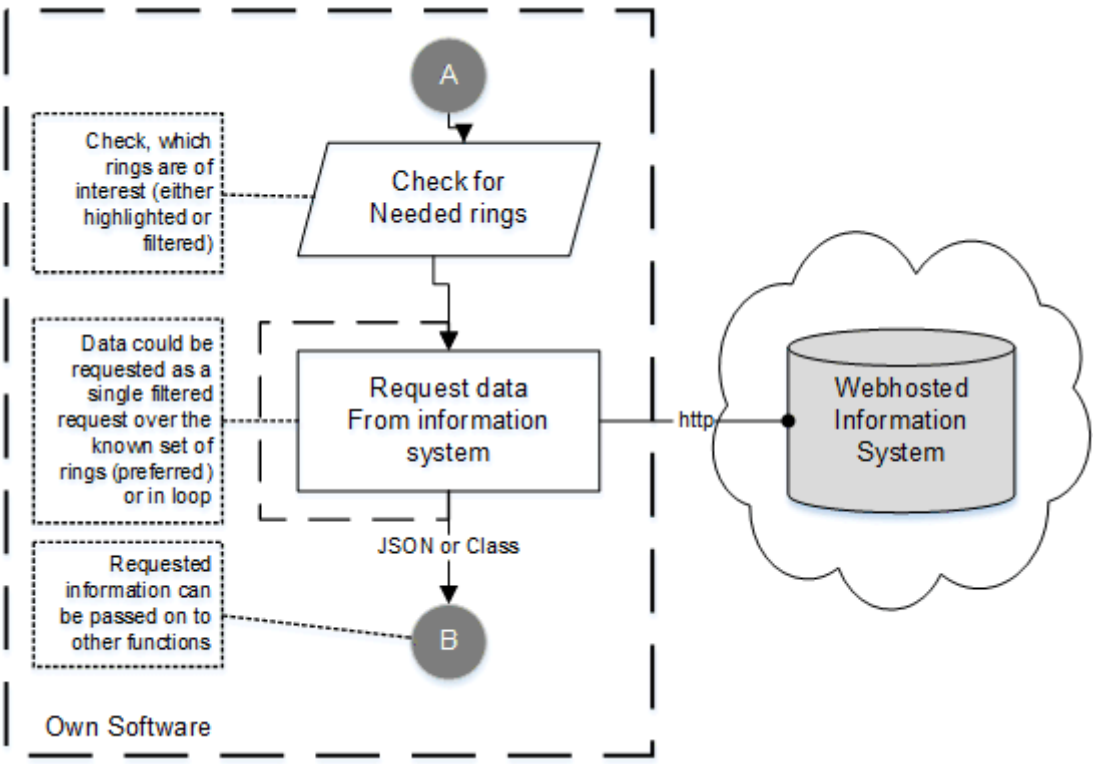


Figure 42: *Previously envisaged solution, which could not be realized due to software bugs on the server-side*

The same approach applies to loading data for each segment on a ring.

The Java library used for the REST-calls was “Unirest”, a lightweight HTTP request library[37]. The Java version was implemented in the Eclipse IDE and used for handling all HTTP requests and responses. Like that simple HTTP requests could be made without dealing with too many of specific details of the HTTP communication protocol, as exemplarily shown in Figure 43.

```
// Methods of the Class
// ====================================

public JsonNode getDataOnline() throws UnirestException{

    // Requesting information from online repository

    HttpResponse<JsonNode> response = Unirest.get(this.URL).
            header("accept", "application/json").
            header("Connection", "close").          // Close Connection after attempt
            asJson();

    // check response for successful connection
    setConnectionSuccessful(response);
```

Figure 43: *Example of some source code using the Unirest library: Performing an http GET-Request on a given resource* (own code)

[37] The library is available for 7 programming languages, including Java, and hosted using the MIT Software license. It is available for download and implementation into any IDE as a standard JAR archive. Refer to http://www.unirest.io – last accessed 08.11.2016

The fetched data was then matched to the previously loaded tunnel model. That procedure is described in the next section.

4.2.2.3 Matching data to the model

Chapter 4.2.1 showed the structure of the created Software from an architectural point of view. The Layout of the classes used to save the ring- and segment data was also described. Using objectified relationships, described in chapter 2.2.3.3 the fetched meta data were matched to the model. Figure 45 shows the procedure matching the ring metadata from the TBM navigation system to each ring entity of type IfcSpace. Figure 46 shows the equivalent approach used to save segment metadata to each segment entity. The latter furthermore shows a mechanism usable for future use, e.g. to also match material information to each segment.

It turned out to be problematic to match the data to the model one on one, because the available exemplarily tunnel model did not have diverging or unique segment types for each ring. Figure 44 shows that each ring had 5 same segments of type “A”. Hence parsed data for these five segments had to be matched randomly.

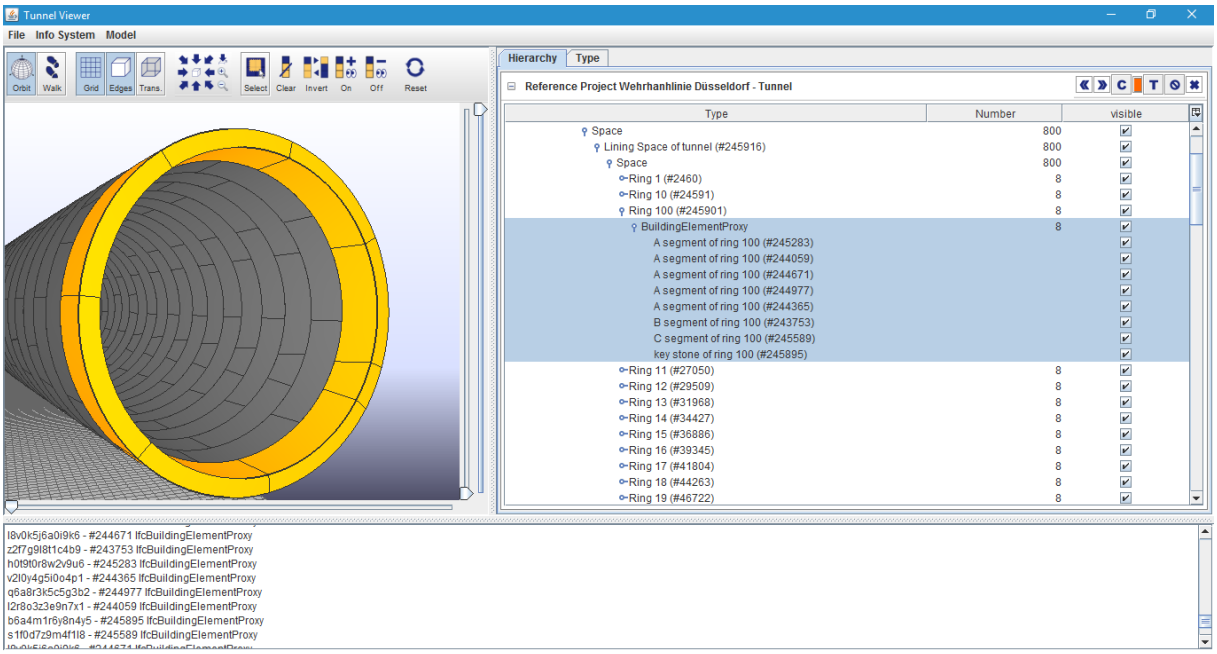


Figure 44: *Recurrent segment type "A" on each tunnel ring, as shown in the table view*

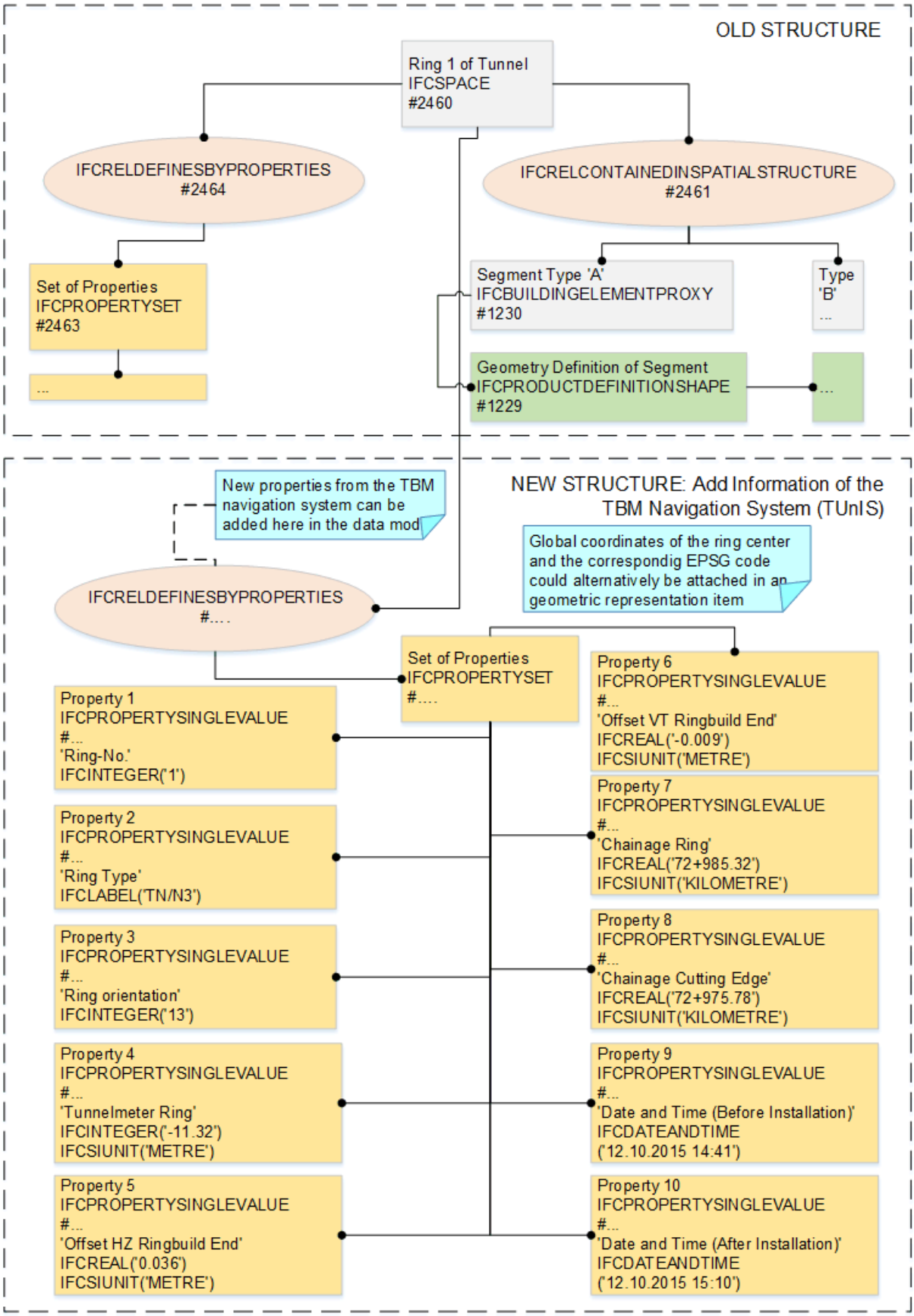


Figure 45: *Matching data of the TBM Navigation system to each ring*

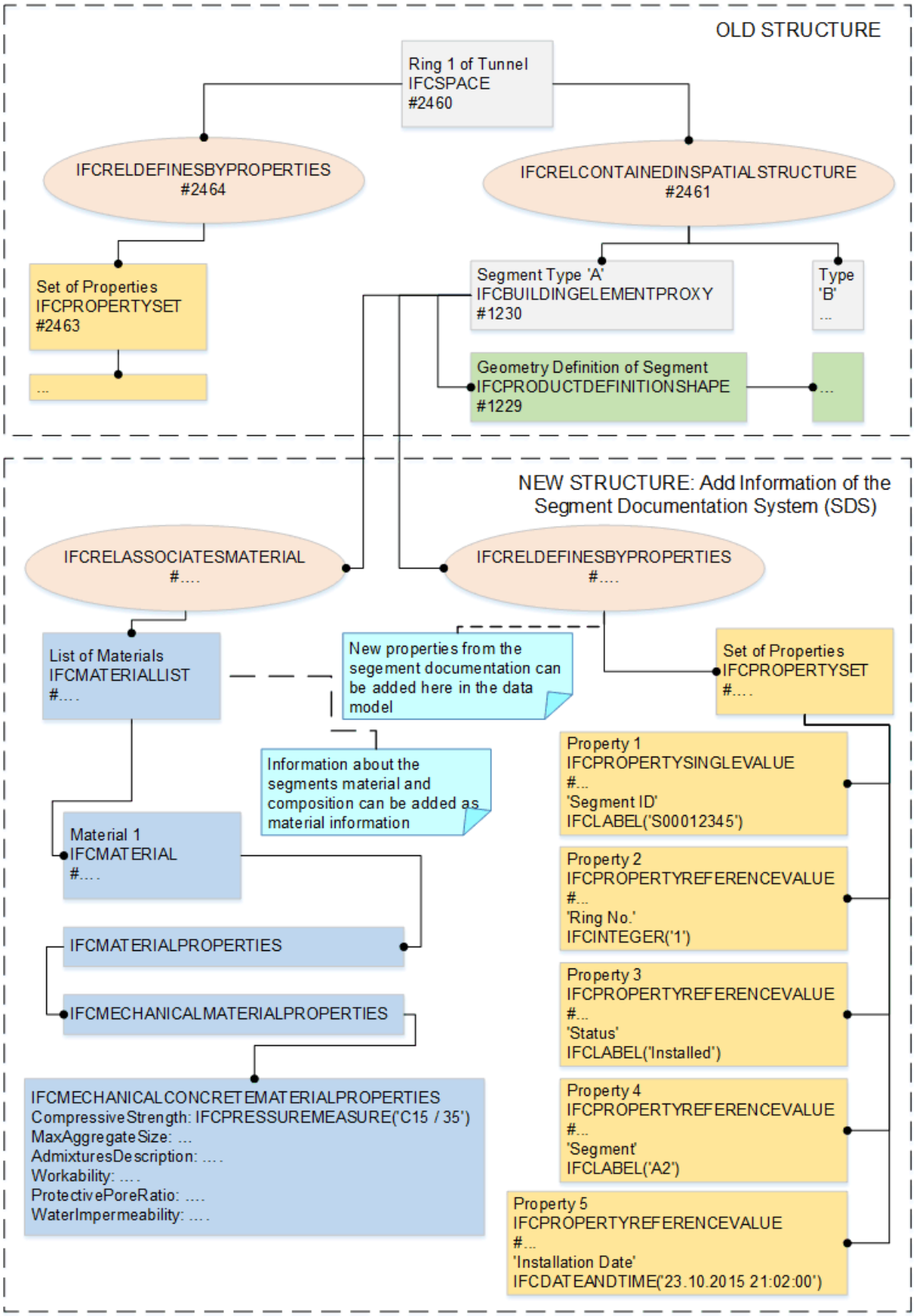


Figure 46: *Matching Data of the Segment Information Management System to each segment entity*

5 Case Study

Chapter 2 lead into the theoretical background of this work. Chapter 3 derived the overall task for this work, to create a piece of software as a showcase. Chapter 4 explained the approach to that, the involved technology and gave an overview of the software architecture. It also gave an overview of problems faced during the realization. This chapter shall show example screen captures of the created software and the workflow.

Said workflow is shown in Figure 47. Screen captures of single working steps are shown in the pictures following.

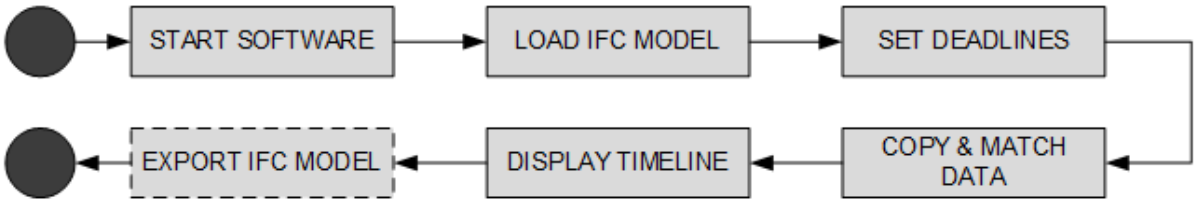


Figure 47: *Workflow of the created software. Exporting the enriched model again to an IFC file would be the next step – which yet was not realized in this work*

The following Figure 48 till Figure 50 show the working steps involved to work on the before described use cases from Table 1 till Table 6: Load a model, set deadlines and attach downloaded meta data to the model.

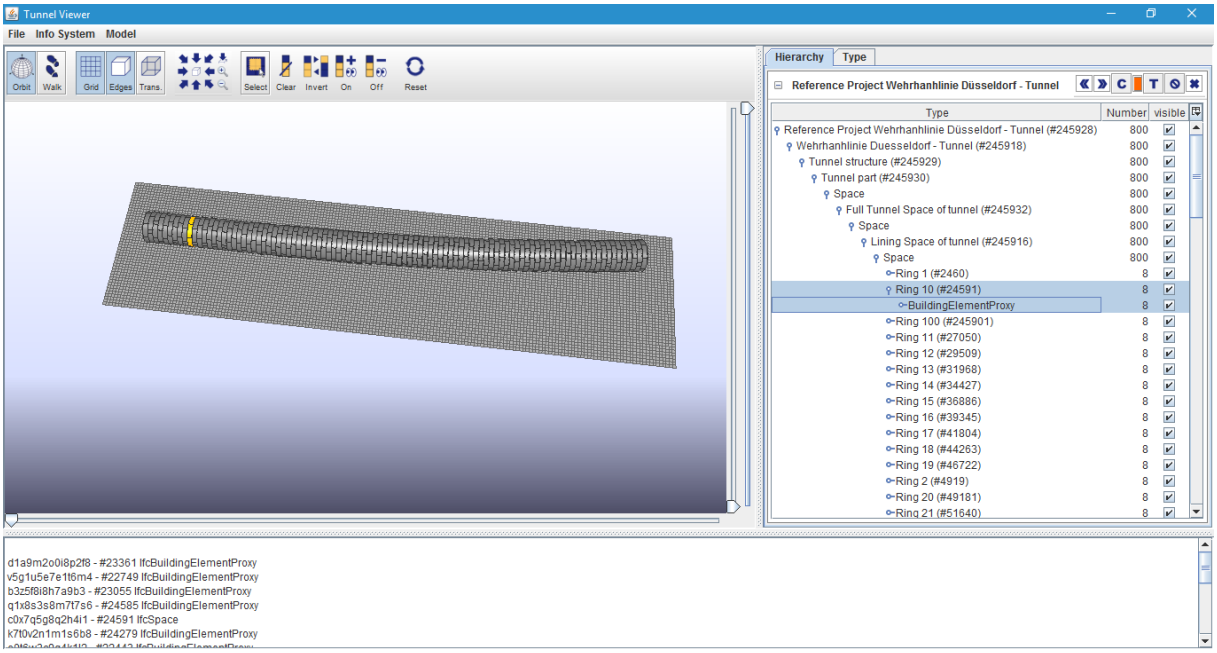


Figure 48: *Step 1: Loaded tunnel model. Next to the visual model, the semantic decomposition is shown in the table view on the right.*

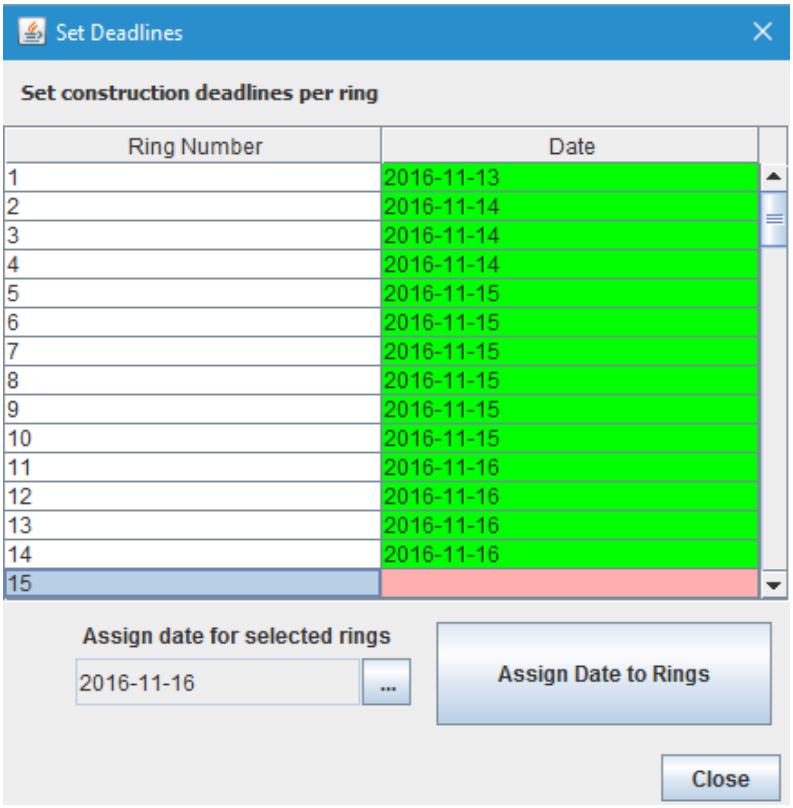


Figure 49: *Step 2: Set deadlines / installation dates to each tunnel ring entitiy individually*

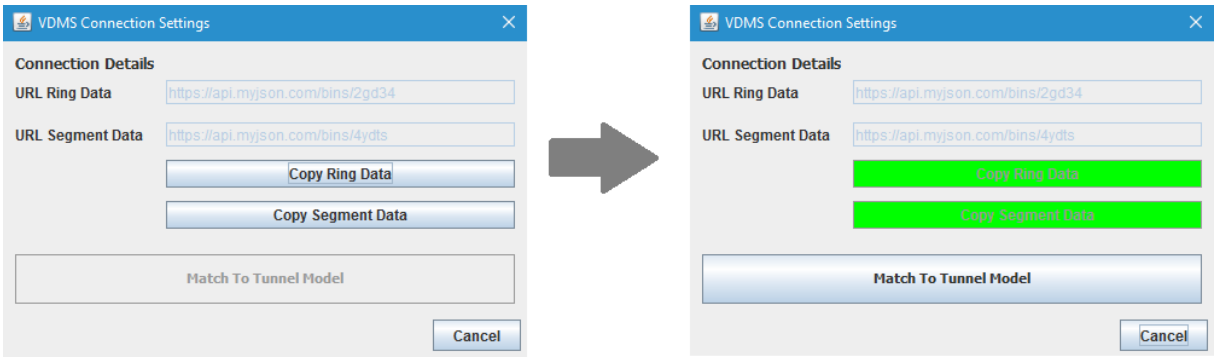


Figure 50: *Step 3: Retrieve element meta data online and match it to the tunnel model entities*

Matched data can then later on be viewed in the model: Highlighting it in either the graphic or the tabular view on the side loads the available tunnel ring meta data to that ring, as shown in Figure 51. Figure 52 shows loaded segment meta data loaded after highlighting it in the model.

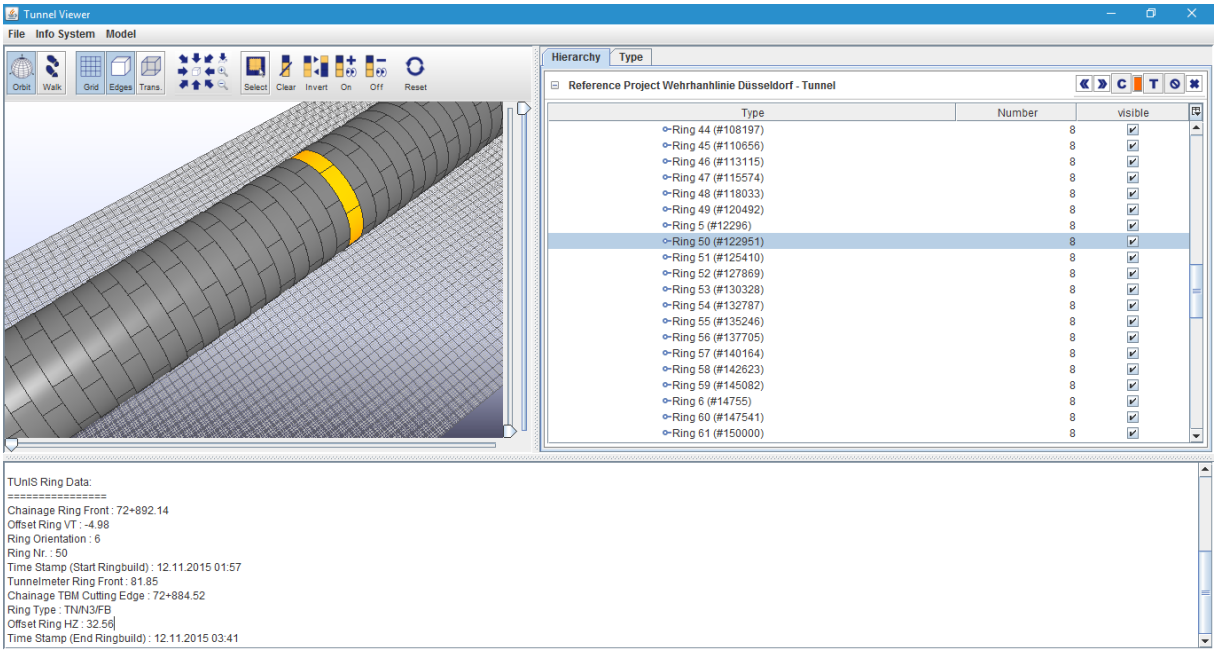


Figure 51: *Display ring meta data in the bottom text pane of the window after highlighting an element.*

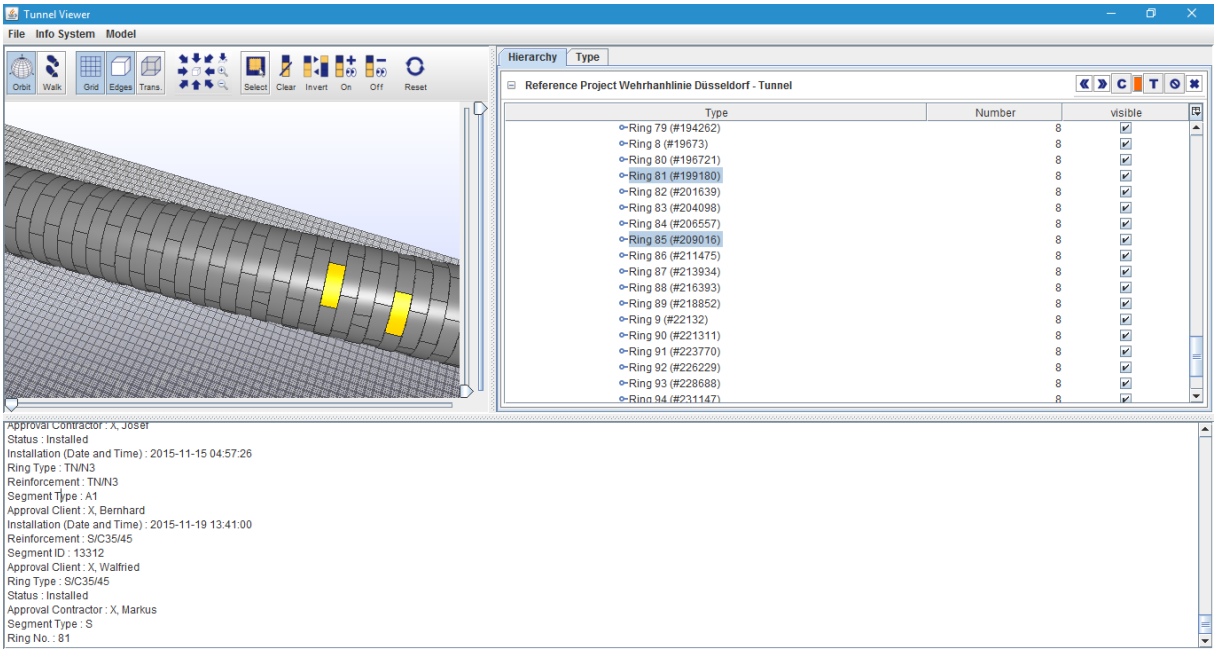


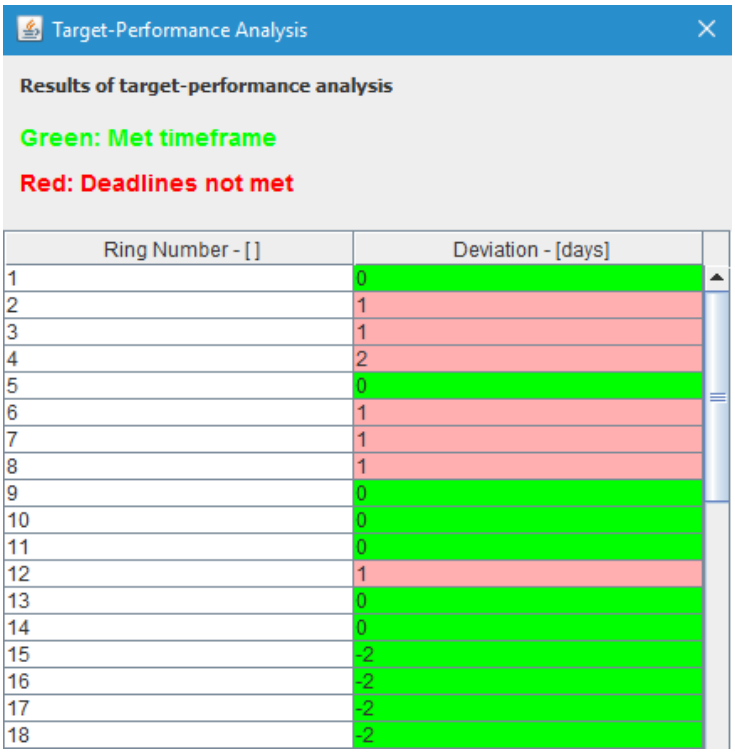
Figure 52: *Display segment meta data in the bottom text pane of the window after highlighting the element*

It turned out to be not too easy to reliably load the relevant metadata for display in the bottom text pane, depending on the element that was highlighted in either the 3D widget or the table view:

- Loading metadata for single tunnel ring segments (as shown in Table 9) worked out fine, as there is no question which element to load the metadata for.
- Loading metadata for complete tunnel rings however (as shown in Table 8) created problems. When highlighting it in the table view, a “ring” (as a semantically abstract construct, only consisting of several ring segments) is always loaded as a randomly

sorted collection of Java objects together with the child objects, the ring segments. Only one watcher-function is available, to check whether elements were highlighted in the user interface. This function can then further act on that, in this case by loading meta data from the model. Said function, on the other hand, does hence not know what is of real interest in each interaction: The parent or the child classes? That leads to the fact, that in the current realization of the software only metadata of the last element in the collection loop is listed – that might be a ring segment, or the ring itself, as the collection of Java objects is saved in an unsorted manner.

After loading both the model and the fetched online meta data and after the desired ring installation dates have been assigned to the model, the fetched online meta data will then also be used to perform a target performance analysis: The software checks, whether the desired installation dates could be met. A simple color coding of red (installation overdue) or green (installation within deadline) was used for visualization, as already used in the online example shown in Figure 4. Like that a rough inspection of timelines can be made possible also when not being on and with data from the jobsite. Yet, as the necessary source code for the relevant classes doing the actual coloring in the 3D widget of the software was not at hand, this idea could not be realized for this project. Figure 54 hence only shows an interpretation, of how a result like that could have looked like. What was possible though, was the creation of a simple colored table view shows the comparison result, as depicted in Figure 53.



Ring Number - [ ]	Deviation - [days]
1	0
2	1
3	1
4	2
5	0
6	1
7	1
8	1
9	0
10	0
11	0
12	1
13	0
14	0
15	-2
16	-2
17	-2
18	-2

Figure 53: *Table showing the comparison result of installation dates. Pink cells with values bigger than 0 indicate a delay. Green cells with values zero or below zero indicate that for a certain ring the installation deadlines where met or outperformed.*

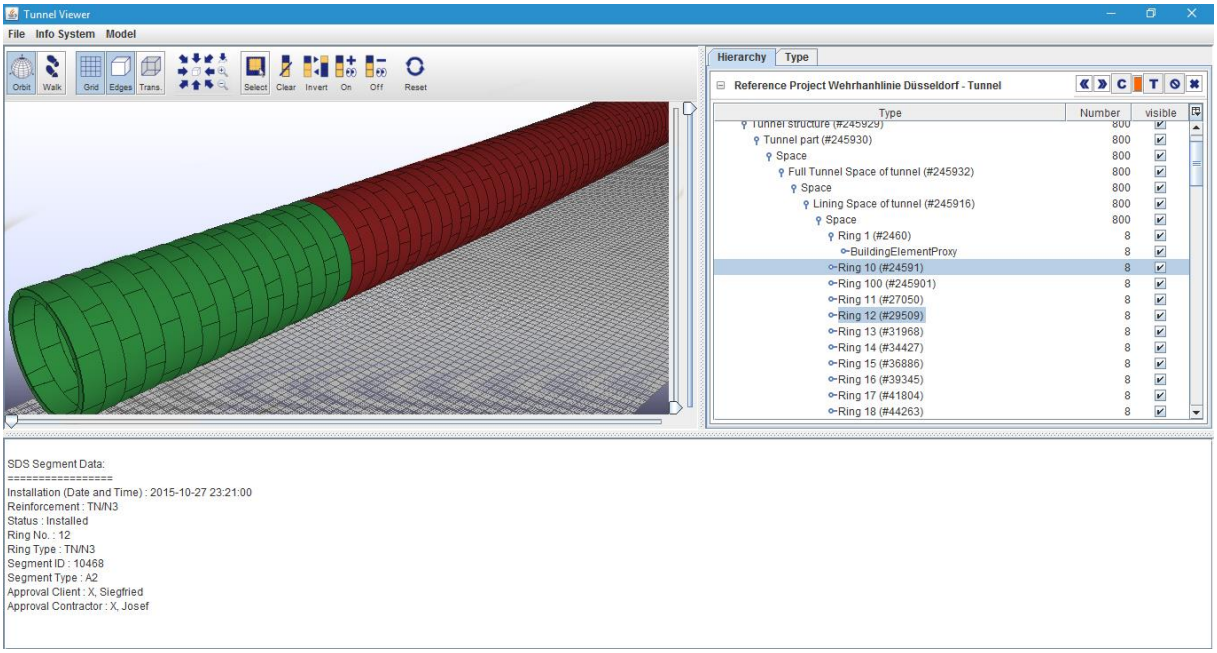


Figure 54: *Interpretation of construction success, after timeline comparison – created using Photoshop*

The readily created software, compiled as an executable .exe file and the required IFC tunnel model file is hosted on the authors Dropbox folder and is available using this link:

https://www.dropbox.com/sh/g5mf96kgk26xlfp/AAD4k8ue6fMoL2d3MVWFzLtTa?dl=0

6 Results & Discussion

This chapter shall sum up the findings of this work and the problems identified e.g. during the creation of the demo software. This part can be seen in section 6.1. In despite of current problems, the general approach to match sensor and machine information to a BI-Model can be of great usage in the future. This is briefly shown in section 6.2. Moreover, the major ones of the found problems are already being tackled to date with the creation of a further expansion of the IFC file standard – this is explained in section 6.3.

6.1 Identified problems

Problems during the creational steps of this work can be grouped into limitations of the source code for the software, which has an influence on how to create the software.

Apart of that, working with standardized file formats in general also brings further usage problems with it, what limits the time frame and the question on when to use the software.

6.1.1 How to create the software

The logic of the software, created for this work, but also for all else software working with data encoded in IFC2x3 or IFC4 in civil construction and in tunnel construction is highly bound to the use of generic proxy classes and string comparison, due to missing specialized classes, like in this Java example:

```
If (element InstanceOf IfcSpace &&
element.getName().toString().contains(„Tunnel Ring“) {
       // do something
}
```

Some examples from the source code of the created software shall be shown as an example, in the following.

This function creates a list of available tunnel rings within the overall IFC model:

```
[...]
public ArrayList<IfcSpace> getRingListFromModel (IfcModel model) {

       try {
              // Create a collection of all those entities of type "IfcSpace" in
              // the model
              Collection<?> listOfIfcSpaces = new ArrayList<IfcSpace>();
              listOfIfcSpaces = model.getCollection(IfcSpace.class);
              ArrayList<IfcSpace> listOfRings = new ArrayList<IfcSpace>();

              // Extract only the relevant entities (-> the Tunnel Rings) by
              // searching for elements with IfcLabel containing a text including
              // "ring"
              for (Object space : listOfIfcSpaces) {
                     if (((IfcSpace)
                     space).getName().toString().contains("Ring") == true)
                            listOfRings.add((IfcSpace) space);
              }

              return listOfRings;
       [...]
}
```

Same is with loading specific attributes or meta data. The following snippet of source code shows the loading process of an IfcPropertySet-element for each tunnel ring segment:

```
[...]
// and then check the label, whether it contains the SDS information
boolean check = def.getName().toString().contains("SDS Segment Data");

if (check == true) {
       // then the right relationship object is found and can be used further on
       // to check for an assigned installation date
       IfcPropertySet properties = (IfcPropertySet) def;

       // get all the properties
       SET<IfcProperty> segmentPropertySet = properties.getHasProperties();

       return segmentPropertySet;


} // end of if-clause
[...]
```

Software created that way is bound to be either highly inflexible, or merely complicated during the creation, as all possible comparison examples must have been thought through beforehand – and would then even only work with a specified dataset.

Further specialization of the IFC standard also towards tunnel construction is thus needed also observed from this angle, to be able to easily create meaningful software to work with data in a flexible matter.

6.1.2 When to use the software

Creating a model in a BIM Suite and then exporting it using the IFC format can often go along with certain loss of information in the model, due to necessary generalization of specialized custom classes of the specialized software, which are not all reflected one-on-one in the IFC schema, the so called “round-tripping”[38]. As that shall not happen at random intermediate steps, where specialized data (e.g. calculation results, parametric modeling information and more) would still be needed, parties exchanging data in IFC file format should thus only do it:

- when being sure that no crucial data is lost in the conversion process, for instance when handing over a completed partial model
- after the full completion of a project stage (e.g. refer to stages of the German HOAI)
- or at other defined points in the project, defined in the BIM Execution Plan (BEP) in the project pre-phase

For a piece of software, created in this work or comparable, that means that using an IFC-based model to enrich it periodically with as-built information (as depicted in Figure 30) should either be used merely for informational purpose or at the end of the construction phase, before the overall model is handed over to the owner and their Facility Management software.

[38] Using specialized software, e.g. for simulation processes, generates data which only said piece of software can work with. It was and is not the goal of the IFC file format, to reflect every specialized use case of model data. Hence, when saving it into IFC format, certain specialized classes have to be mapped into generic proxy classes, to not get fully lost (same mechanism used to create the tunnel model file used for this thesis, in order to be able to load it into the IFC Tools Project viewer). The information is thus not lost, but might not be completely reusable in the creators means either. This unpreventable loss of information, same as the loss of parametric CAD history of the geometry in the stages of handing over data to other participants is called “round-tripping”.

6.2 Future applications and use in BIM Workflows

For a middleware application, like the one created for this work, but also for explained tunnel construction information systems, further meaningful future use cases must be thought through. They all shall serve the overall purpose of the use of automated data in tunnel construction, using BIM as a method. Further manual entry of data into software can be averted, so that the use of model data for engineering simulations and analysis can directly be supported. A few thoughts on that are listed here:

For a long-lasting documentation of the building, next to linking building elements (Tunnel ring or segment) to metadata of the TBM navigation system and segment information system, a further meaningful set of metadata from the TBM's machine data must be collected. All of those machine meta data, which could be of interest in any later evaluation context, should also be linked. That could be, but is not limited to, pressure of cylinders pushing onto the segments, pressure of mortar injection, pressure of the tunnel face support or TBM advance speed values.

In practical use of BI Models, the use of different model types differing by their depth of content won recognition. Five steps are distinguished by their "Level of Detail" (LoD). A single modelled element can herein be linked to a growing set of meta data. This is called "Level of Information" (LoI). During the planning and the execution of construction works those different models could roughly be called the deliverable of specific project phases, when it comes to digital documentation[39]. Figure 55 shows that concept of LoDs. At the end of the construction phase, in prearrangement of the operation phase, updating the models with data of external systems can also be utilized: A piece of software to update these models with as-built data from the construction like that can contribute to update the model in LoD5 with LoI5: Update of model elements referred to the built as-is condition, e.g. real and precise dimensions, forms, position and geographical reference (König et al., 2016b , p. 207). It is considerations like these, which are also part of the use-case study of BIM in tunnel construction, evaluated at the Rastatt tunnel project (refer to chapter 2.1.3).

During the construction phase for a building project, having machine data can be used for event-based actions also in the BIM-context, e.g. the update of the tunnel building model. Sending a feedback signal to the model, each time a tunnel ring was installed underground. This can further on be used for:

- Model-based progress reporting for the jobsite. Next to progress reporting on site, done by personnel using mobile devices, an automatic interface can help in this stage. Like that, for instance the TBM navigation system can help achieving a model-based schedule-review
- Model-based invoicing to the client, per tunnel ring, when systems are also connected to the companies' ERP systems and helping with model-based cost-control
- Communication purposes: Giving a live update on the state of the tunnel, also for visualization, e.g. in context of public relations
- Giving live feedback into linked interaction models, for simulations, e.g. for settlement analysis, as shown in Figure 56. These simulations can e.g. take place for the ongoing bettering of the construction process, in context to the "observation method" described

[39] The assignment of different model types with their LOD and LOI to specific construction phases and BIM use cases are currently assigned using a simple matrix, written out in the BIM Execution Plan. That is also necessary, cause the normal project structure for building projects nowadays (e.g. in Germany in reference to HOAI stages) does not reflect the creation of digital models. For future project the BIM Workflows must also find reflection in the contracts.

in chapter 3.1. Simulation of processes on the site and their influences will play a growing role in construction projects in the future.

These actions can be performed on different models. The German “BIM Code of Practice 1.0” as an expert-proposal to the industry for the construction phase e.g. differentiate between the “execution model” and the “construction execution model” with different goals (Dohmen et al., 2016).

The ISO 19650 “Information management using BIM” and the extension of the ISO 16739 “Information sharing in the construction and FM industries (IFC)” for infrastructure construction are currently in development. It has to be seen which role model updates with real sensor information can play in that context.

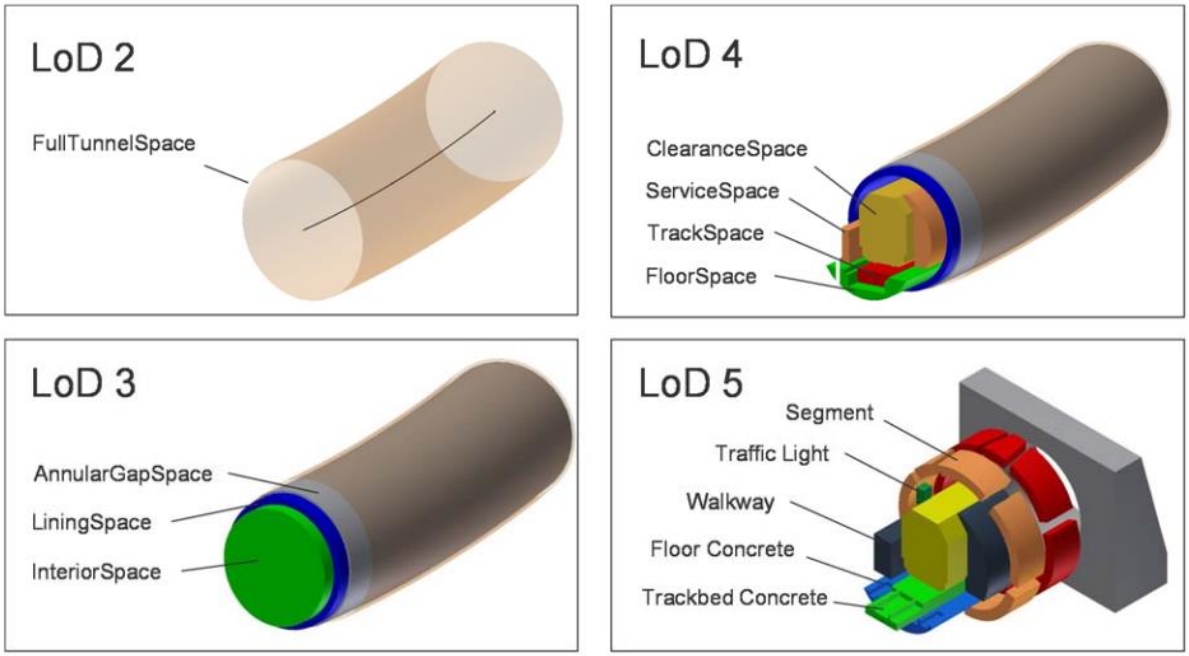


Figure 55: *Level Of Detail descriptions for a tunnel model* (Borrmann et al., 2014). *A further proposal of which LoD-Model shall be used in which project phase and their detailed content is given in* König et al. (2016b), p. 32

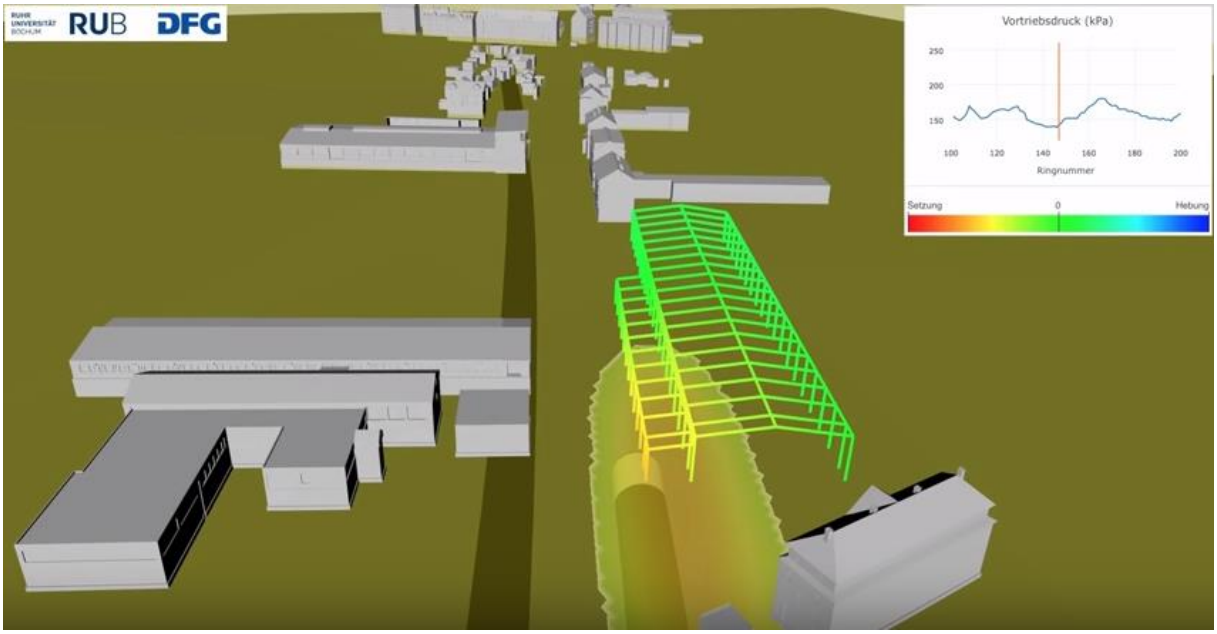


Figure 56: *Simulation of TBM advance and effective ground settlements,* Source: Bahnprojekt Karlsruhe-Basel, 2016 [online], URL: https://www.youtube.com/watch?v=-L1cJqX7IpE (02.12.2016)

6.3 IFC 5 for Infrastructure

The BIM method was created in classical engineering construction. The benefits are uncontroversial internationally – exemplarily cost benefits and other major improvements in media dissemination and project communication were named in chapter 2.1. The possible benefits of the method also for big infrastructure projects goes without saying. National and international committees clearly state their preference for open data standards when it comes to sharing information, thus also for IFC (König et al., 2016a) which does not consider infrastructure works up until now. Chapter 2.1.2 and 2.1.3 already described the current efforts for the use of BIM internationally and in Germany, same as for the tunneling business in particular. Yet, also chapter 2.2.4 identified the lack of the IFC schema to properly represent tunnel structures in the IFC file format. Not only tunnel buildings lack the representation, it is all kinds of infrastructure projects as a whole. Bridges, roads, dams and more are still in need to be modelled correctly in IFC – an effort, which is currently tackled by the buildingSMART "InfraRoom" working group and the Model Support Group. The next revision of the file format, IFC5 (no publication date available yet) is in development and will merely be focused on infrastructure. As described in chapter 2.2.2, the 2013 revision of the IFC format (IFC4) also came along with the recognition as an ISO standard. IFC 4 brought several improvements, as described in Liebich (2013), some of which were also the necessary prerequisites for the integration of larger infrastructure objects into the IFC format, for instance making the use of geospatial coordinate reference systems available.

In context of IFC, currently running research projects focus on the foundation of every very elongated infrastructure: The alignment. Figure 57 shows this concept. IfcAlignment is considered to be the first addendum to IFC4 (Amann and Borrmann, 2015). Research projects for IfcRoad and IfcBridge are already running (König et al., 2016a), yet a project for IfcTunnel is still missing funding (Borrman et al., 2016).

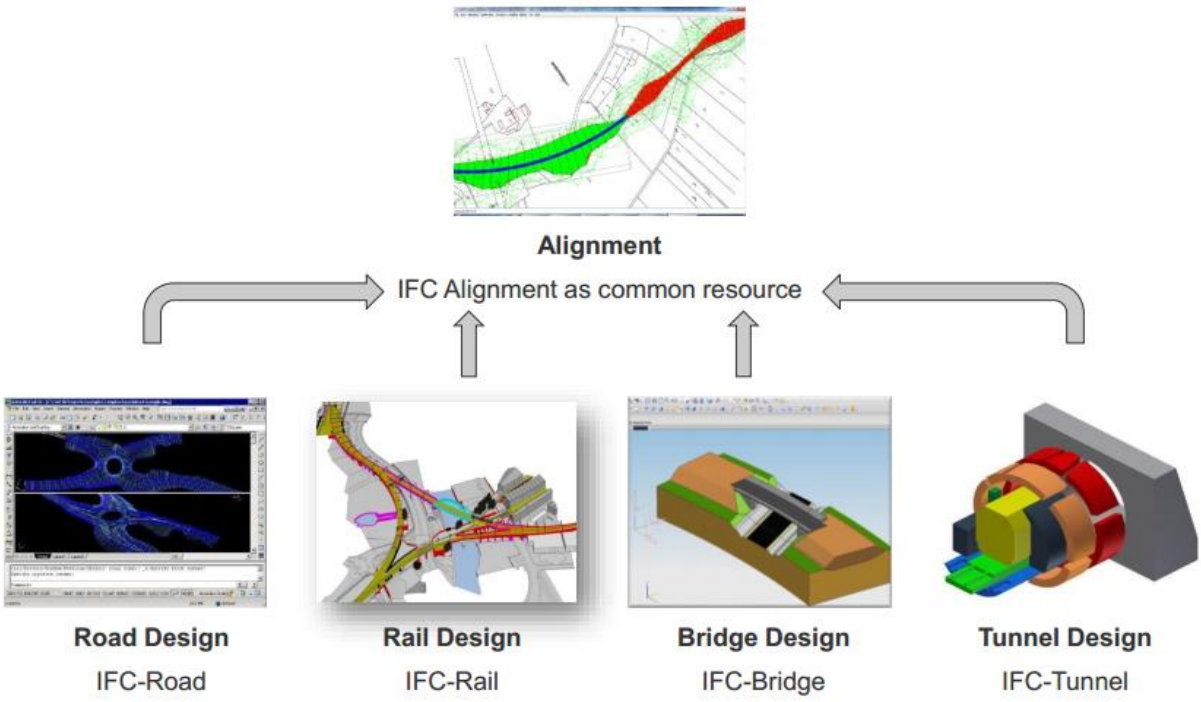


Figure 57: *IfcAlignment as the basic data structure for elongated infrastructure projects* - Source: Borrman et al. (2016)

IFC5 shall furthermore take care about a meaningful integration with GIS data structures. Infrastructure projects more than “normal building projects” have a higher demand for Geodata from GI systems. To make a software integration of both sources possible, current research also focusses on the creation of a GML-based data structure for medium scale projects. This effort is called “InfraGML” and is led by the OGC with consultation of buildingSMART. Reuvers (2014) explains the idea. The proposal was published in Scarponcini (2014). Using that approach, the industry shall also overcome the currently used quasi-standard LandXML for road data, as it is not compatible with neither GML nor IFC (Amann, 2016).

Only like that it can furthermore be made sure, that infrastructure BI Models can easily be integrated into already existing Spatial Data Infrastructures (SDI), e.g. city models created using CityGML. Also linking several models together is then quicker and easier. Using the construction of the subway line “Wehrhahnlinie” in Düsseldorf Germany as an example that was done in a publicly funded research project of the Ruhr-University Bochum and published in Schindler et al. (2014). An example use of these linked models, the usage within simulation contexts, was already shown in Figure 56.

7 Conclusion

This chapter shall shortly summarize the work and give a conclusion.

Information Systems for use in tunnel construction were introduced, together with an example choice of automated systems, which do generate data input for such information systems – the list may well be extended by several other machineries and facilities running on normal jobsites.

The current extent of available IFC extensions, applicable to TBM driven tunnel construction were assessed. None of the two versions found were yet put into any commercial software. Further research projects for further developments are yet to come.

While researching the IFC standard as a whole, it was found that the initially asked question which of the two proposals for IFC extension were “more” applicable to be enriched with further information (like from mentioned information system) is not really a “valid” question, when it comes to IFC: By using means of objectified relationships and property sets, properties of any kind can be “attached” to every object within the IFC schema – which made the work with the available IFC tunnel model easier. The model originates from the SFB837 research initiative mentioned in chapter 2.2.4 and was thus structured using non-official IFC classes for use in tunnel construction, like proposed in the work of Amann et al. (2013). It turned out, those could not be read by the official IFC Tools Project version – which is why an adopted version had to be used, as described in chapter 2.2.3.5. While working with that, the need for IFC tunnel extension were thus very apparent, as described before.

The 4D BIM approach used in this work could have also been performed a step further, to write the changed BI model into a new IFC file and to visualize the construction in a 4D-enabled BIM suite, like e.g. the software “desiteMD” (from which Figure 3 is taken). The potential for the practical use of the investigated method (to update BI models with construction process data) is clear, a few use cases were presented in chapter 6.2 – those originated from the BIM research project for the tunnel project Rastatt. Of cause, also other application and use cases are foreseeable and could be researched.

8 Literature Overview

AMANN, J. LandXML, LandInfraGML. 2. Treffen der Expertengruppe IFC-Road, 21.04.2016 2016 München.

AMANN, J. & BORRMANN, A. 2015. Creating a 3D-BIM-compliant road design based on IFC alignment originating from an OKSTRA-accordant 2D road design using the TUM Open Infra Platform and the OKSTRA class library.

AMANN, J., BORRMANN, A., HEGEMANN, F., JUBIERRE, J., FLURL, M., KOCH, C. & KÖNIG, M. 2013. A Refined Product Model for Shield Tunnels based on a generalized approach for alignment representation.

AUTODESK 2005. Parametric Building Modeling: BIM's Foundation.

BMVI 2015. Stufenplan Digitales Planen und Bauen. *In:* INFRASTRUKTUR, B. F. V. U. D. (ed.).

BORRMAN, A., KÖNIG, M. & LIEBICH, T. 2. Treffen der Expertengruppe IFC-Road. 2. Treffen der Expertengruppe IFC-Road, 21.04.2016 2016 München.

BORRMANN, A., KOLBE, T., DONAUBAUER, A., STEUER, H., JUBIERRE, J. & FLURL, M. 2014. Multi-Scale Geometric-Semantic Modeling of Shield Tunnels for GIS and BIM Applications. *Computer-Aided Civil and Infrastructure Engineering*.

BORRMANN, A., KÖNIG, M., KOCH, C. & BEETZ, J. 2015. *Building Information Modeling: Technologische Grundlagen und industrielle Praxis*, Springer Fachmedien Wiesbaden.

BREIDENSTEIN, M., HANDKE, D. & FRITZSCHE, W. 2015. Diskussionspapier zur Erarbeitung konfliktarmer Bauverträge im Tunnelbau. *In:* DEUTSCHER AUSSCHUSS FÜR UNTERIRDISCHES BAUEN E, V. & GERMAN TUNNELLING, C. (eds.).

BUILDINGSMART 2007. IFC2x Edition 3 Technical Corrigendum 1.

BUILDINGSMART. 2010. *Der BIM Beirat* [Online]. Available: http://www.buildingsmart.de/kos/WNetz?art=Compilation.show&id=32 [Accessed 24.07.2016 2016].

DIN 2015. Building Information Modeling (BIM): From model to specification of works. *In:* NORMUNG, D. I. F. (ed.).

DOHMEN, P., LIEBSCH, T., SAUTTER, H., BREDEHORN, J. & HEINZ, M. 2016. *BIM Praxisleitfaden 1.0* [Online]. Available: http://www.BIM-blog.de [Accessed 02.12.2016.

EASTMAN, C. M. 1999. *Building product models: computer environments, supporting design and construction*, CRC press.

EASTMAN, C. M. 2011. *BIM handbook : a guide to building information modeling for owners, managers, designers, engineers, and contractors,* Hoboken, NJ, Wiley.

EGGER, M., HAUSKNECHT, K., LIEBICH, T. & PRZYBYLO, J. 2013. BIM-Leitfaden für Deutschland. *In:* STADTENTWICKLUNG, B. F. V. B. U. (ed.).

GALLAGHER, M., O'CONNOR, A., DETTBAR, J. & GILDAY, L. 2004. Cost analysis of inadequate interoperability in the US capital facilities industry. National Institute of Standards and Technology (NIST).

GIRMSCHEID, G. 2012. *Baubetrieb und Bauverfahren im Tunnelbau*, Wiley.

HEGEMANN, F., FRODL, S. & MAYER, P.-M. 2016. Integration von vortriebsbegleitenden Prozessdaten in BIM-basierte Modelle zur Kontrolle des Baufortschritts. *5. Münchener Tunnelbau Symposium.* München.

HERTEN, M. Nutzen der Beobachtungsmethode nach EC 7. 14. Fachgespräch für den Tiefbau, 2012 Minden.

IEEE 2009. IEEE Standard for Information Technology - Systems Design - Software Design Descriptions. IEEE Computer Society.

KÖNIG, M., AMANN, J., BORRMAN, A., BRAUN, M., ELIXMANN, R., ESCHENBRUCH, K., GOETZ, A., HAUSKNECHT, K., HOCHMUTH, M., LIEBICH, T., NEJATBAKHSH, N., SCHEFFER, M. & SINGER, D. 2016a. Wissenschaftliche Begleitung der BMVI Pilotprojekte zur Anwendung von BIM im Infrastrukturbau.

KÖNIG, M., RANK, E., BLETZINGER, K.-U., BORRMANN, A., SMARSLY, K. & HUHNT, W. 2015. Aktuelle Entwicklungen und Herausforderungen der Bauinformatik. *Bauingenieur,* 90**,** 320 - 329.

KÖNIG, M., TEIZER, J., MARX, A., SCHLEY, F. & KESSOUDIS, K. 2016b. Building Information Modeling (BIM) im maschinellen Tunnelbau. *In:* E.V., D. G. F. G. (ed.) *Taschenbuch für den Tunnelbau 2017.* Wiley.

KRAEMER, E. 2014. *UML Sequence Diagrams* [Online]. Available: http://www.cse.msu.edu/~kraemere/cse335/Lectures/02SyntheticConcepts/uml_intro_6pp.pdf.

KRUCHTEN, P. 1995. Architectural Blueprints—The "4+ 1" View Model of Software Architecture. *Tutorial Proceedings of Tri-Ada,* 95**,** 540-555.

KUNZ, J. & FISCHER, M. 2009. Virtual design and construction: themes, case studies and implementation suggestions. *Center for Integrated Facility Engineering (CIFE), Stanford University*.

LAAKSO, M. & KIVINIEMI, A. 2012. The IFC standard: A review of History, development, and standardization, Information Technology. *ITcon,* 17.

LIEBICH, T. 2009. IFC 2x Edition 3 model implementation guide. *International Alliance for Interoperability*.

LIEBICH, T. 2013. IFC4 - the new buildingSMART Standard.

LIEBICH, T., SCHWEER, C.-S. & WERNIK, S. 2011. Die Auswirkungen von Building Information Modeling (BIM) auf die Leistungsbilder und Vergütungsstruktur für Architekten und Ingenieure sowie auf die Vertragsgestaltung.

MAIDL, B., HERRENKNECHT, M., MAIDL, U. & WEHRMEYER, G. 2013a. *Mechanised shield tunnelling*, John Wiley & Sons.

MAIDL, B., THEWES, M. & MAIDL, U. 2013b. *Handbook of tunnel engineering : Structures and methods,* Berlin, Ernst & Sohn.

MAIDL, B., THEWES, M. & MAIDL, U. 2014. *Handbook of tunnel engineering : Basics and additional services for design and construction,* Berlin, Ernst & Sohn.

MAIDL, U. & STASCHEIT, J. 2014. Real time process controlling for EPB shields / Echtzeit-Prozesscontrolling bei Erddruckschilden. *Geomechanics and Tunnelling,* 7**,** 64-71.

MAY, N. A survey of software architecture viewpoint models. Proceedings of the Sixth Australasian Workshop on Software and System Architectures, 2005. Citeseer, 13-24.

MAYER, P.-M., STASCHEIT, J. & MAIDL, U. 2015. Einsatz von Informationssystemen im maschinellen Tunnelbau. *In:* GEOTECHNIK, D. G. F. (ed.) *Taschenbuch für den Tunnelbau 2015.* Wiley-VCH Verlag GmbH & Co. KGaA.

NAGEL, F. 2012. Konsequente, computergestuetzte Anwendung der Beobachtungsmethode nach DIN 1054 zur Prozessadaption im Tunnelbau/Effective computer-aided application of the observation method accoding to DIN 1054 for process adaptation in tunnel construction. *MINING+ GEO,* 2.

OBJECTAID-LLC 2016. ObjectAid Plugin for Eclipse IDE.

PCI 2009. Building Information Modeling (BIM) Technology.

REUVERS, M. 2014. InfraBIM - Linking Pin Between Geo an BIM.

RIECHERS, J. 2014. Customized state-of-the-art segment production. *TAI Journal (A Half Yearly Technical Journal of Indian Chapter of TAI),* 3**,** 15-22.

ROZANSKI, N. & WOODS, E. 2005. Applying viewpoints and views to software architecture. *Whitepaper.* http://www.viewpoints-and-perspectives.info: Nick Rozanski and Eoin Woods.

ROZANSKI, N. & WOODS, E. 2011. *Software systems architecture: working with stakeholders using viewpoints and perspectives*, Addison-Wesley Professional.

SCARPONCINI, P. G., HANS-CHRISTOPH 2014. OGC Draft LandInfra Conceptual Model.

SCHERER, R. J. & SCHAPKE, S.-E. 2014. Multimodellbasierte Zusammenarbeit in Bauprojekten. *Informationssysteme im Bauwesen. 1, 1.*

SCHINDLER, S., HEGEMANN, F., ALSAHLY, A., BARCIAGA, T., GALLI, M., LEHNER, K. & KOCH, C. 2014. An interaction platform for mechanized tunnelling. Application on the Wehrhahn-Line in Düsseldorf (Germany)/Eine Interaktionsplattform für maschinelle Tunnelvortriebe.

Anwendung am Beispiel der Wehrhahn-Linie in Düsseldorf. *Geomechanics and Tunnelling,* 7**,** 72-86.

SMITH, C. 2014. *BIM at Sweden's Hallandsås Tunnel: Planning pioneer* [Online]. New Civil Engineer. Available: http://www.newcivilengineer.com/features/geotechnical/bim-at-swedens-hallandss-tunnel-planning-pioneer/8663146.article [Accessed 24.07.2016 2016].

STASCHEIT, J. & MESCHKE, G. 2013. Process Oriented Numerical Simulation of mechanized Tunneling using an IFC-based Tunnel Product Model.

STUHLMACHER, K. 2014. BIM-Aktivitäten außerhalb und innerhalb Deutschlands.

TAUSCHER, E. & THEILER, M. 2013. IFC Tools Project.

TAYLOR, M. 2013. Crossrail: A Case Study in BIM. *In:* 2013, L. C. D.-C. (ed.).

TRIMBLE. 2011. *Seattle's massive tunnel makes travel safer – with Tekla BIMsight* [Online]. Available: https://www.teklabimsight.com/references/seattles-massive-tunnel-makes-travel-safer [Accessed 24.07.2016 2016].

VDI. 2013. *VDI - Koordinierungskreis BIM* [Online]. Available: https://www.vdi.de/technik/fachthemen/bauen-und-gebaeudetechnik/fachbereiche/bautechnik/artikel/vdi-koordinierungskreis-bim/ [Accessed 24.07.2016 2016].

VDI. 2016. *VDI Coordination Group Building Information Modeling* [Online]. Available: http://www.vdi.eu/engineering/technical-divisions/civil-engineering-and-building-services/building-information-modeling/ [Accessed 04.04.2016 2016].

VILGERTSHOFER, S., JUBIERRE, J. & BORRMANN, A. 2016. IfcTunnel-A proposal for a multi-scale extension of the IFC data model for shield tunnels under consideration of downward compatibility aspects.

VMT-GMBH 2015. Segment Documentation System (SDS) - User Manual. internal material.

VMT-GMBH 2016. Tunnel Underground and Information Structure (TUnIS) - User Manual.

WIKIPEDIA. 2016. *Building Information Modelling* [Online]. Available: https://en.wikipedia.org/wiki/Building_information_modeling#BIM_software [Accessed 23.07.2016 2016].

YABUKI, N. Representation of caves in a shield tunnel product model. Proc. of the 7th European Conference on Product and Process Modelling, Sophia Antipolis, France, 2009. 545-550.

YABUKI, N., ARUGA, T. & FURUYA, H. Development and application of a product model for shield tunnels. Proc. of the 30th International Symposium on Automation and Robotics in Construction, Montréal, Canada, 2013.

YABUKI, N., AZUMAYA, Y., AKIYAMA, M., KAWANAI, Y. & MIYA, T. 2007. Fundamental study on development of a shield tunnel product model. *Journal of Civil Engineering Information Application Technology,* 16**,** 261-268.

ZEISS, G. 2016. *Crossrail's huge BIM+geospatial model targeted on operations and maintenance* [Online]. Available: http://geospatial.blogs.com/geospatial/2016/04/crossrails-huge-full-lifecycle-bim-model-intended-for-operations-and-maintenance-.html [Accessed 24.07.2016 2016].

Appendices

This chapter shall give more information about the software documentation, in detail:

- about the UML class diagrams, for each created class of the tunnel viewer software. The software moreover makes use of a lot of classes of the IFC file structure and the unirest HTTP library. Their documentation can viewed online – refer to the literature list.
- A short software user documentation for the created tool is provided

Appendix I: UML Documentation: Class Diagrams

This chapter gives an overview about the created classes, which dealt with the customization of the available IFC Tools Project.

Most of the newly created classes are a direct child of the Java root class “Object”. For future projects, the class structure should be reworked to be more hierarchical.

The class “TunnelViewer” describes the main user interface, opens and loads IFC files and displays them. It is shown in Figure 58.

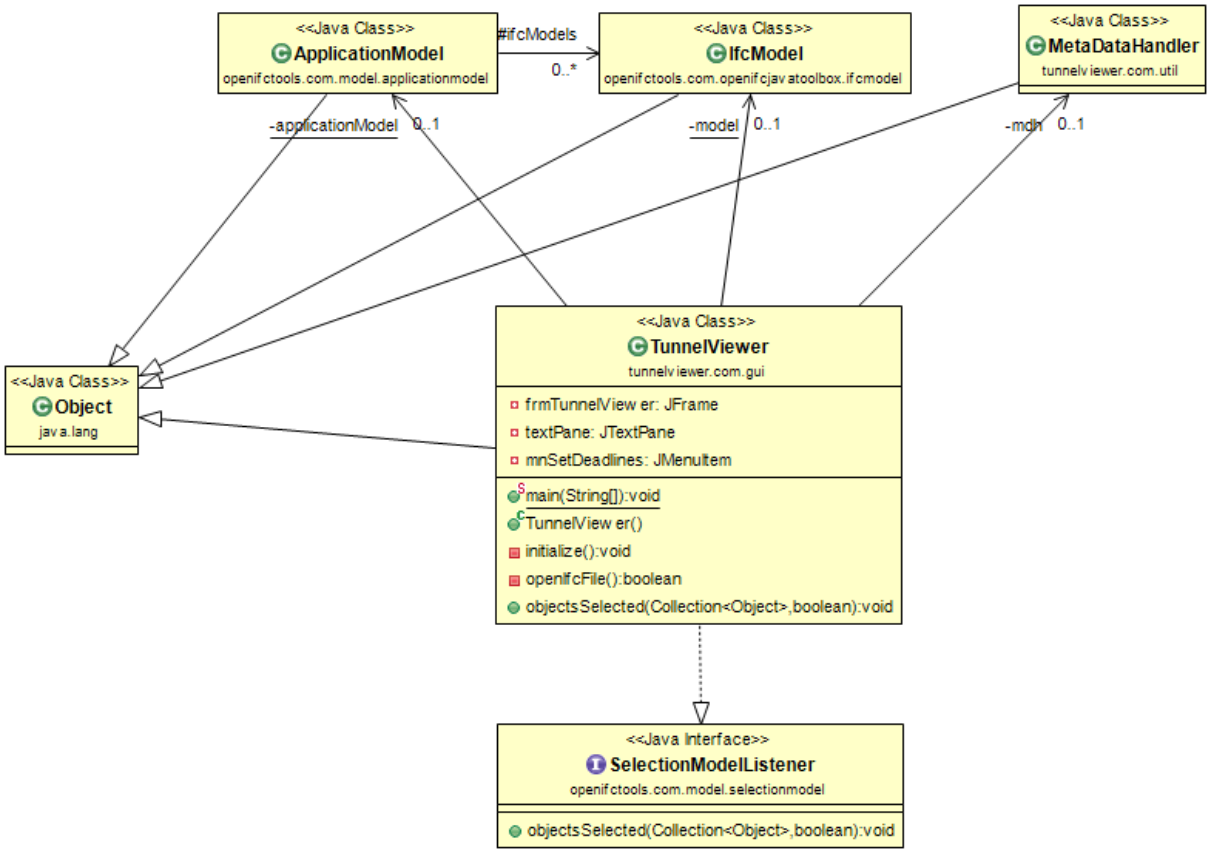


Figure 58: *Class "TunnelViewer" as the main user interface window*

The class “SetDeadlines” describes the user interface necessary, to assign planned installation dates to the loaded ring entities (also see Figure 49) and is shown in Figure 59. The class uses a designated helper class of type “TunnelModelData” to deal with the interaction with the loaded IFC model, as that one was not supposed to happen on UI-Level. This helper class

then for instance is responsible to assign dates to the model, but also to read tunnel ring objects from said model. It is described in Figure 60.

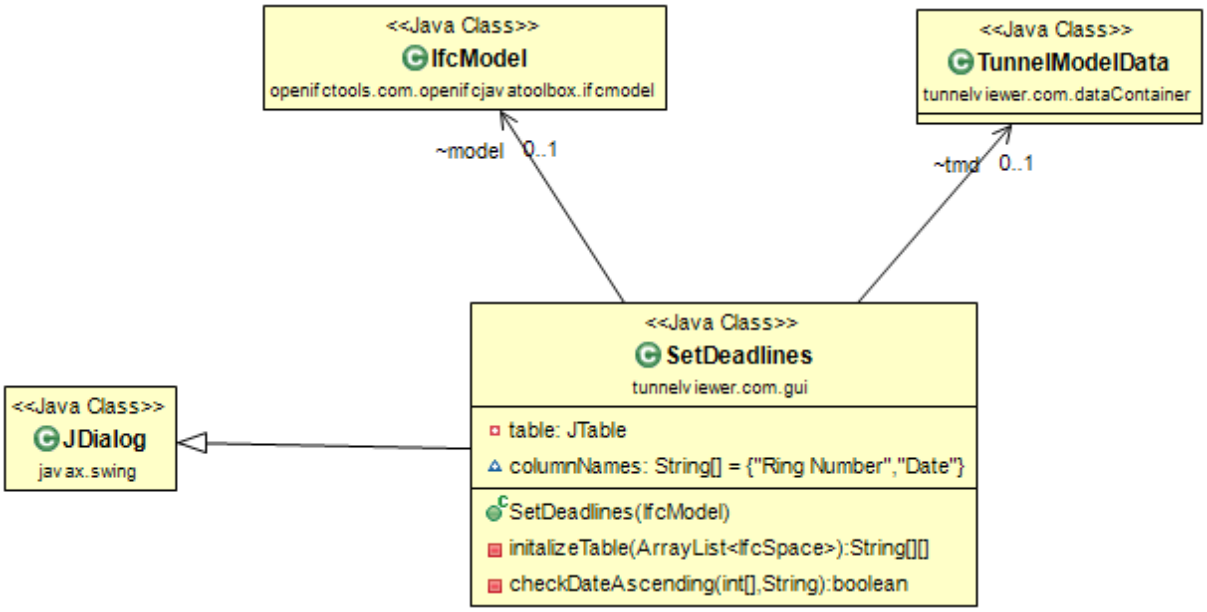


Figure 59: *Class SetDeadlines as the user interface, when wanting to assign desired installation dates to tunnel ring entities*

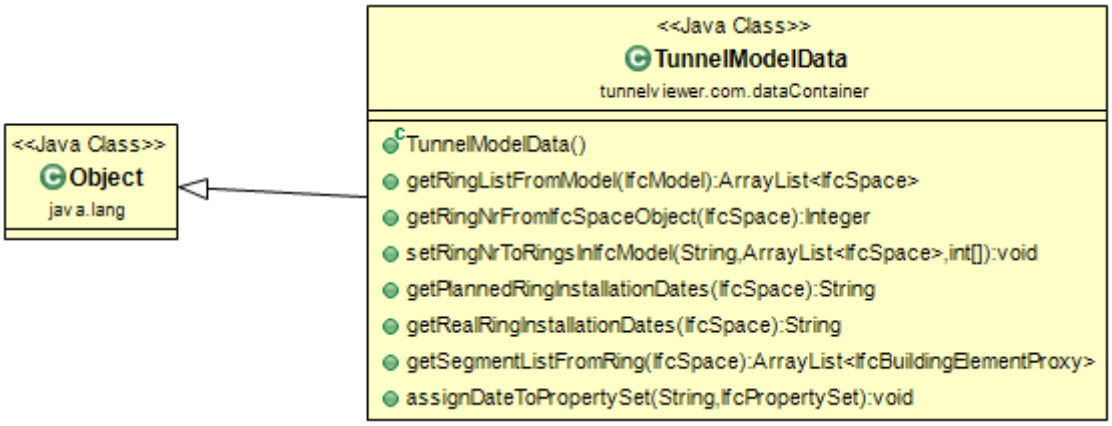


Figure 60: *Class "TunnelModelData" as a helper class to get ant set data directly from the loaded IFC tunnel model*

After assigning the installation dates, meta data can be read from the online repository and matched to the model. The interaction starts on the user interface, described in class "VdmsConnection" in Figure 61. The user interface first calls a helper class "RestDataAssembler" (see Figure 62) to connect to the given URL and fetch the meta data. To then hand it over to another helper class "RestResponseObjectCreator" (see Figure 63), responsible to parse the fetched JSON-String and break it down into objects of either class "RingData" or "SegmentData", depending on the type of information received. The latter two are show in Figure 64 and Figure 65.

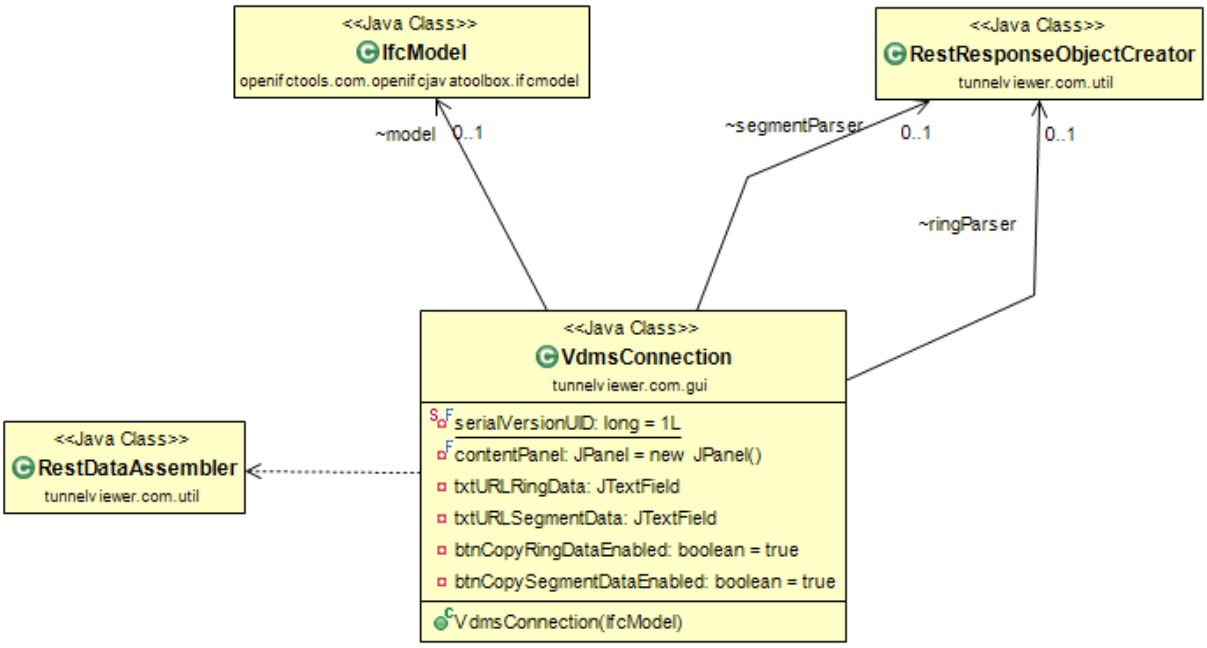


Figure 61: *Class "VdmsConnection" as a user interface to trigger a connection to the online repository and fetch meta data for the tunnel model*

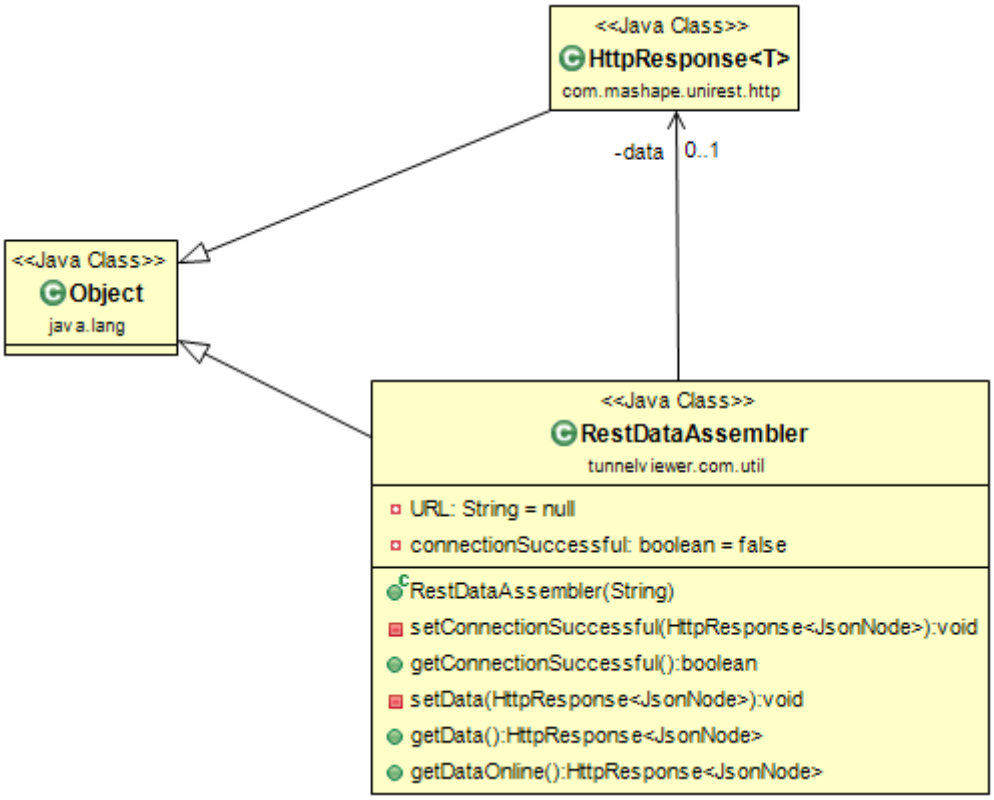


Figure 62: *Class "RestDataAssembler" to connect to the online repository and fetch meta data. The class references the Unirest HTTP-Library*

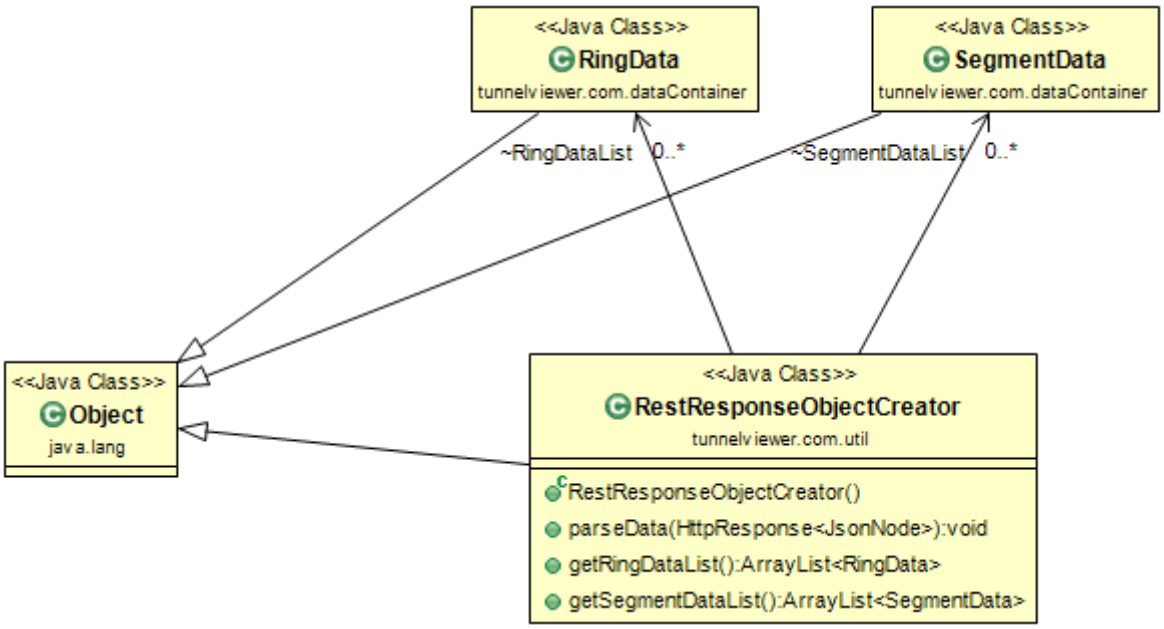


Figure 63: *Class "RestResponseObjectCreator" as a helper class to parse the received JSON-String*

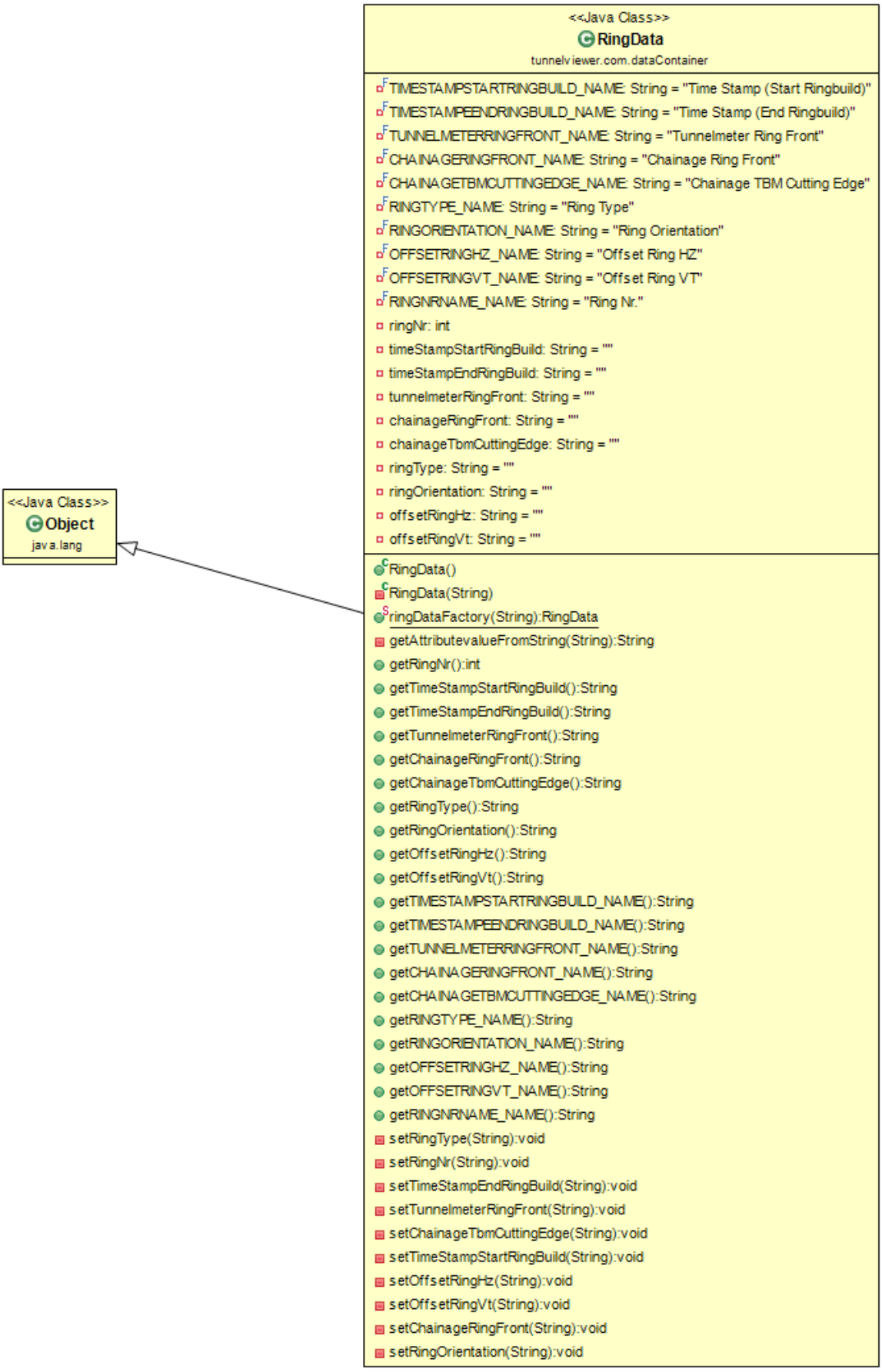


Figure 64: *Class "RingData" as a temporary data storage for tunnel ring meta data, before they are matched to the IFC tunnel models ring entities. Here it is also visible that in this realization the meta data attribute names are hard coded as static variables in capital letters.*

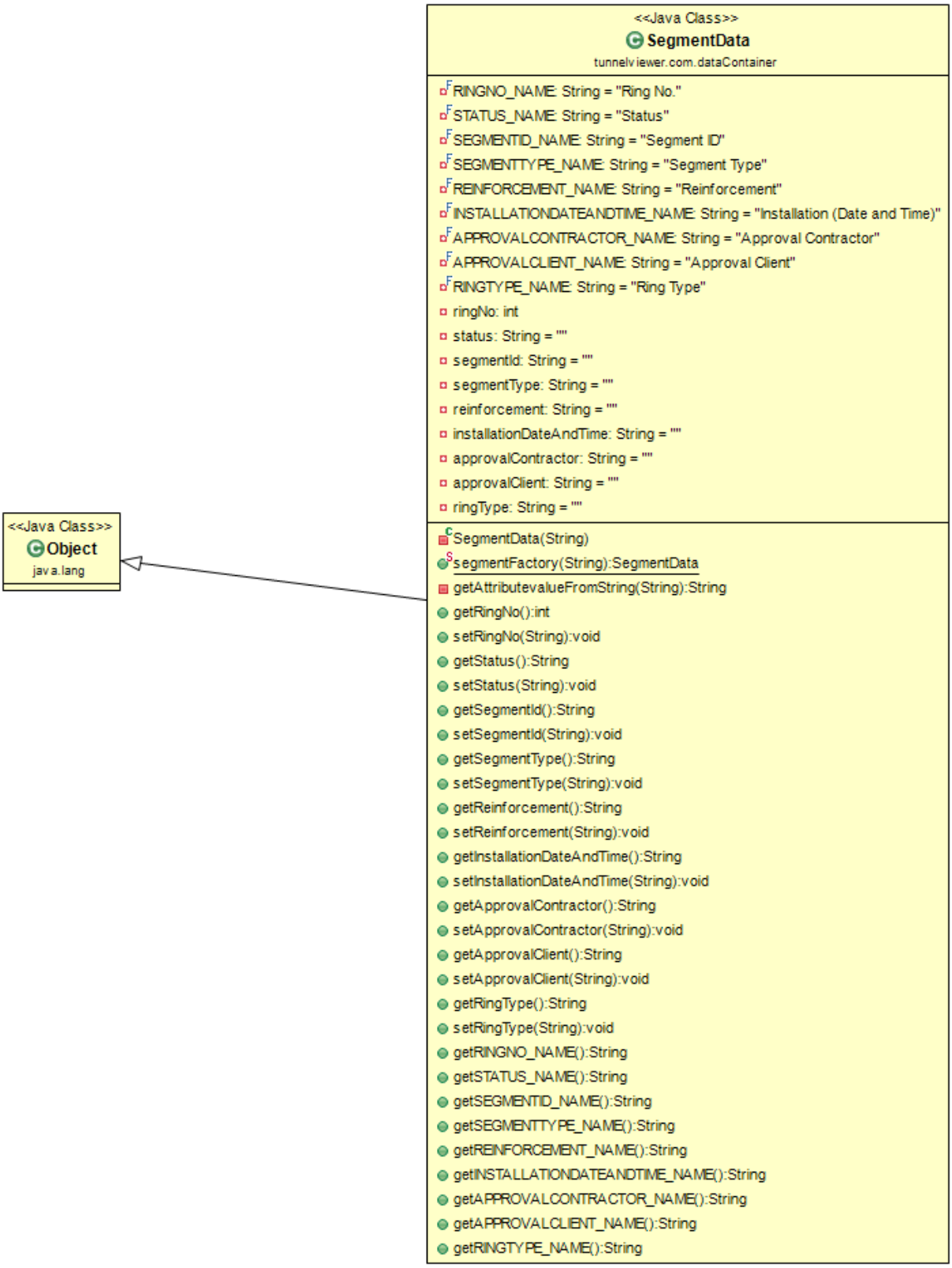


Figure 65: *Class "SegmentData" as a temporary data storage for tunnel ring meta data, before they are matched to the IFC tunnel models ring entities. Here it is also visible that in this realization the meta data attribute names are hard coded as static variables in*

After all meta data have been fetched, parsed and saved into a temporal data structure, they are then assigned to the IFC tunnel model using the functions of the helper class “MetaDataHandler”, as shown in Figure 66.

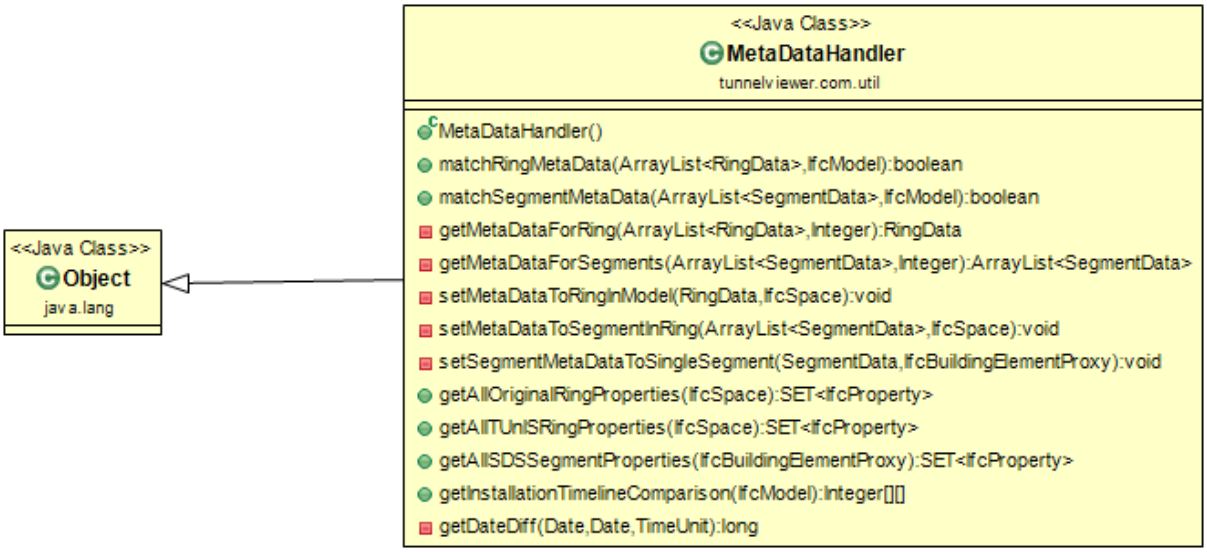


Figure 66: *Class "MetaDataHandler" to match the fetched meta data to the IFC tunnel model permanently*

The user of the software then can perform the timeline comparison, navigating the user interface window of class “TimelineComparison”, as shown in Figure 67. That one again references the helper class “TunnelModelData” (refer to Figure 60) to perform the calculation.

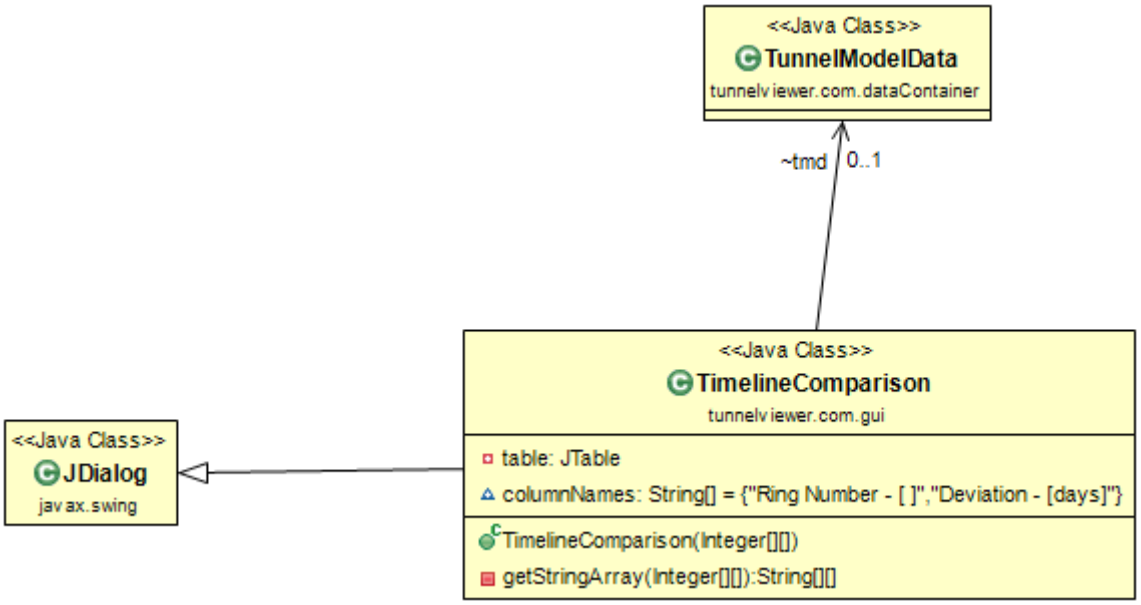
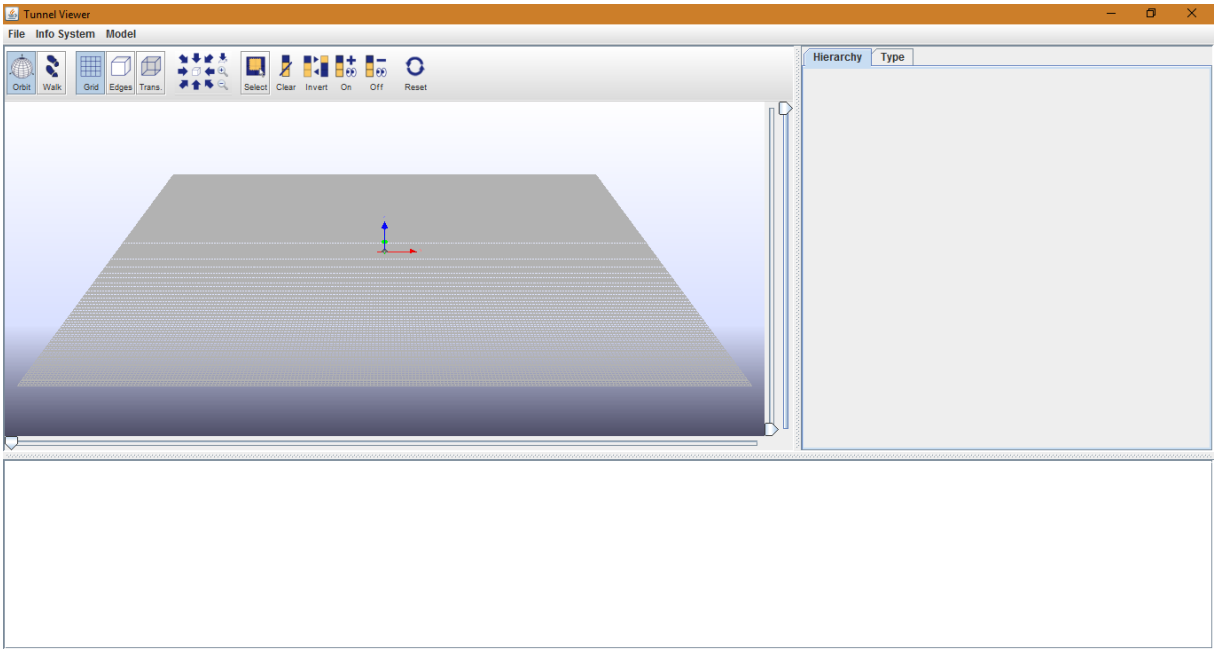


Figure 67: *Class "TimlineComparison" to do the calculation, whether the tunnel rings have been installed in due time, consulting both the manually set installtaion dates and the fetched meta data, both matched to the ITC tunnel model*

Appendix II: Software Manual

This short introduction is written to give an overview of the use of the created piece of software. The working steps are described sequentially.

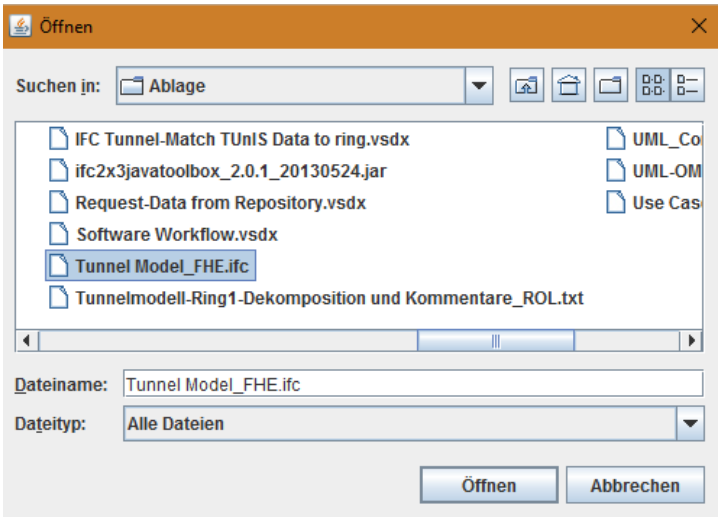
1. Open the Software



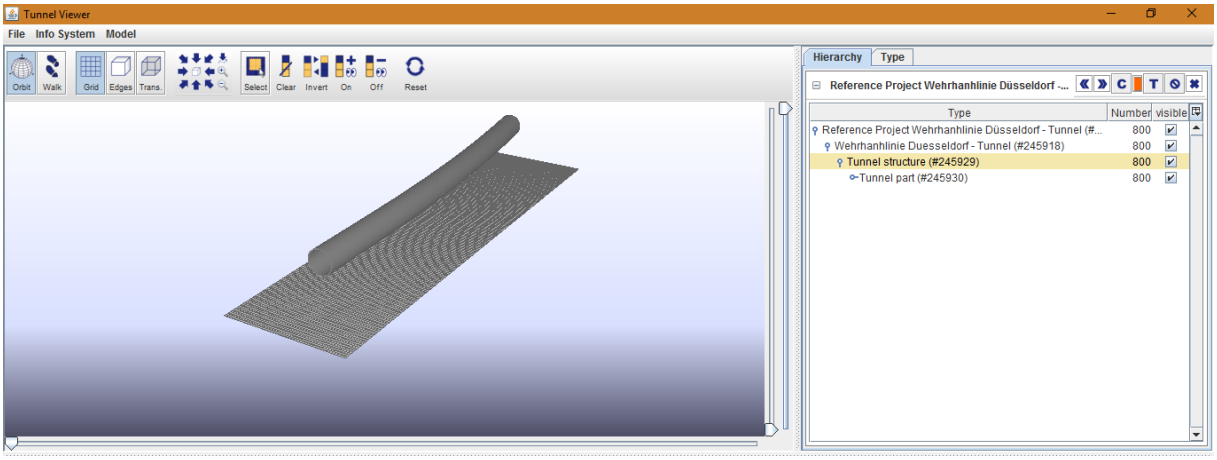
The Software will be empty and present itself with no model loaded. The side and bottom panel are adoptable by dragging the borders with the mouse.

2. Open the provided IFC file and work with the model

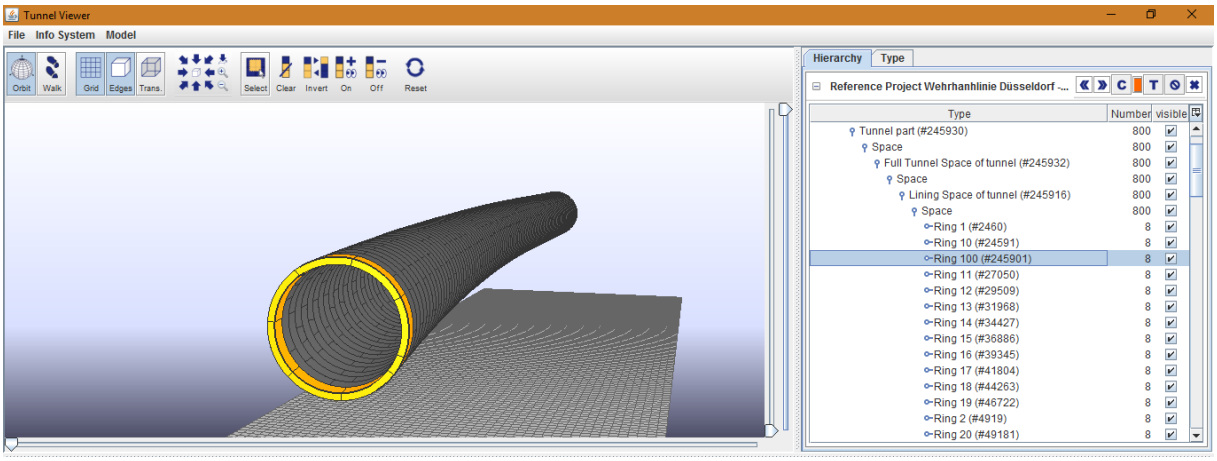
A tunnel model is available for loading. Click “File / Open IFC File” to load it from your hard disk.



The model will be loaded and displayed in the 3D widget on the left. The semantic structure of the IFC file will be broken down and displayed in a collapsed tree structure in the tab spaces on the right.



The view of the model can be altered using the mouse wheel for zoom and dragging functionalities. Entities of the model can be highlighted yellow either clicking onto it in the model or by selection in the tree structure on the right.



Using the top navigation panel, the view can be manipulated, e.g. to display the model in a transparent view (Button “Trans.”)

Selections can also be reset, cleared and reverted.

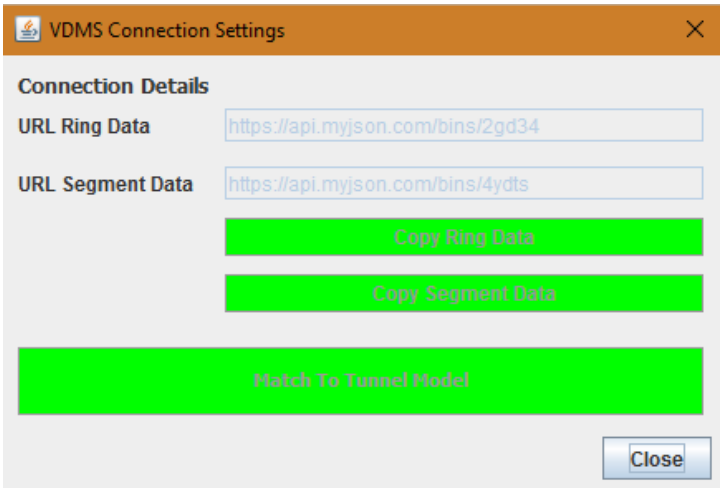
3. Using additional functions created for this thesis

The content of this theses focused around the creation of an interface to a web-based information system, to enrich the model with potentially live meta data. The following two steps can be performed in a random order.

3.1 Copy meta data from the web information system

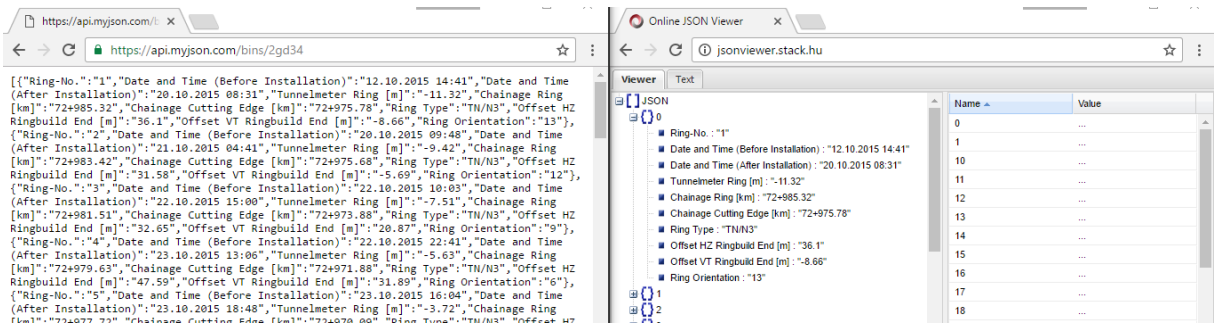
Click on “Info System / VDMS Setting” to open the dialog to communicate with the information system. As described in chapter 4.2.2.2 a workaround solution to a bug within the server-side software had to be found. This is why the URL of the web-hosted metadata is hardcoded and cannot be altered by the user.

Clicking “Copy Ring Data” and “Copy Segment Data” will get the meta data from the online resources. Clicking “Match To Tunnel Model” will do exactly that, as described in 4.2.2.3.

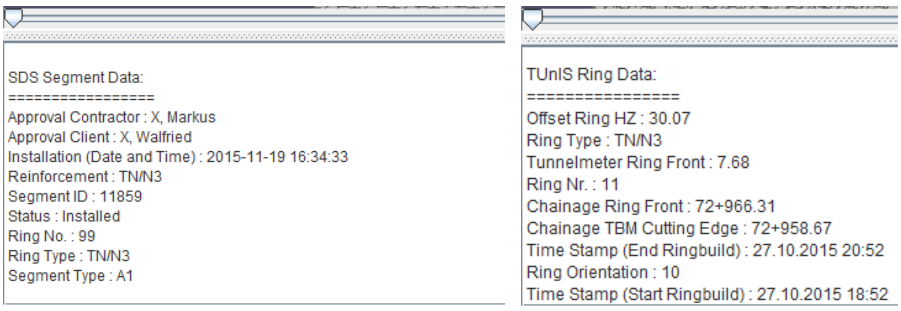


Hint 1: Known flaw of the software currently is, that it is not prevented to perform this step twice. Closing the “VDMS Connection Settings” Window and opening it again would enable the user to re-do this step. That still needs to be changed.

Hint 2: In comparison to the provided demo data set of tunnel installation dates the user should know that the data set is from 2015. For comparison reasons, these files can be downloaded from the provided links in chapter 4.2.2.2 and viewed more structured in an online viewer like jsonviewer.stack.hu



The assigned meta data can then be viewed in the model, by clicking on the entities in the 3D widget, or by selecting them in the tree structure. All meta data will be shown in the bottom text panel.

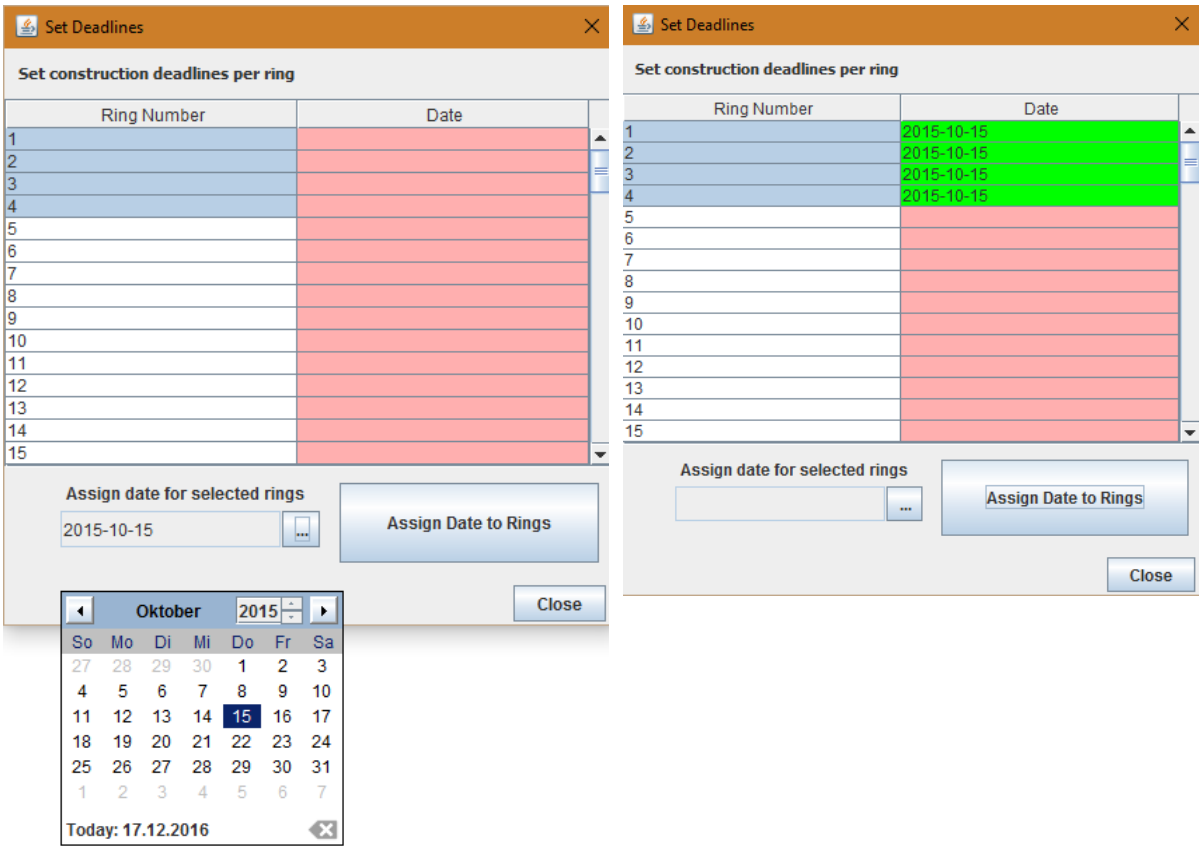


Yet, the display is currently bound to restrictions, as already described in chapter 5.

3.2 Set installation deadlines to the tunnel rings

Click “Model / Set Deadlines” to assign the installation deadlines to the model. If a tunnel model was at hand, which already incorporated those dates from the planning phase of a project, that step would not be necessary anymore.

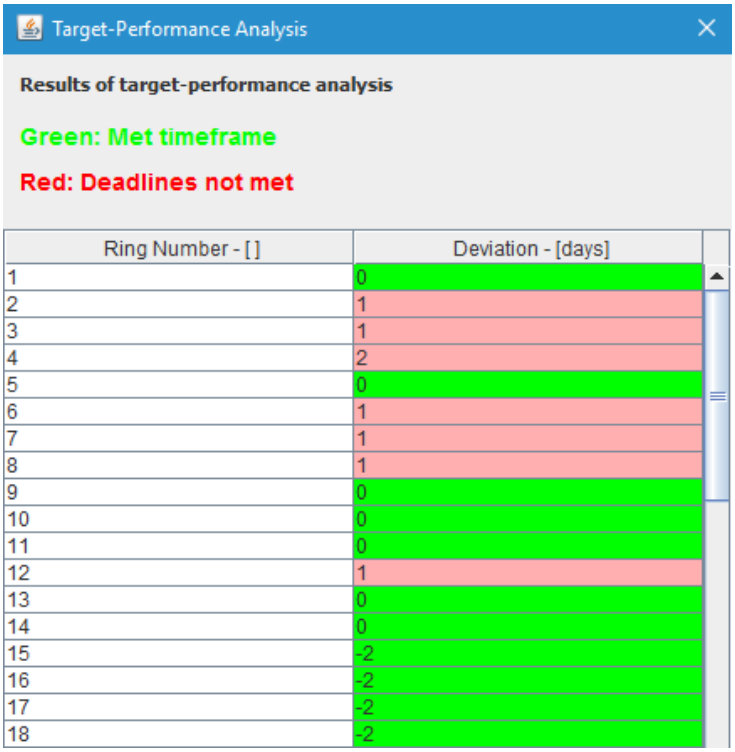
Highlight the ring numbers, to which dates should be assigned. Then open the date picker element and select the desired date and click “Assign Date to Rings”.



A date must be assigned to all rings, otherwise an installation timeline comparison will not be possible

4. Perform an installation timeline comparison

Click “Model / Display Timeline” for the software to perform a comparison and show the comparison result. The calculation result, the deviation, will be shown in days.



Ring Number - [ ]	Deviation - [days]
1	0
2	1
3	1
4	2
5	0
6	1
7	1
8	1
9	0
10	0
11	0
12	1
13	0
14	0
15	-2
16	-2
17	-2
18	-2