INSITER INTUITIVE SELF-INSPECTION TECHNIQUES

# BIM-based Self-Instruction models for mobile devices

Deliverable report D4.4

**Deliverable Report D4.4, final version, issue date on 12 December 2017**

**INSITER - Intuitive Self-Inspection Techniques using Augmented Reality for construction, refurbishment and maintenance of energy-efficient buildings made of prefabricated components.**

**T**his research project has received funding from the European Union's H2020 Framework Programme for research and innovation under Grant Agreement no 636063.

INSITER INTUITIVE SELF-INSPECTION TECHNIQUES

# BIM-based Self-Instruction models for mobile devices

## Deliverable report D4.4

| | | |
|---|---|---|
| Issue Date | 12 December 2017 | |
| Produced by | RDF | |
| Main author | Iveta Bonsma | (RDF) |
| Co-authors | Peter Bonsma | (RDF) |
| | Tzvetelina Zayakova | (RDF) |
| | Jan-Derrick Braun | (HVC) |
| | Richard Deighton | (DMO) |
| Version: | Final | |
| Reviewed by | Arjan Broers | (ISSO) |
| | Richard van Dommelen | (ISSO) |
| | Jacques Cuenca | (SISW) |
| Approved by | Ton Damen – Project Coordinator | (DMO) |
| | Rizal Sebastian – Technical Coordinator | (DMO) |
| Dissemination | Public | |

### Colophon



This research project has received funding from the European Union's H2020 Framework Programme for research and innovation under Grant agreement no 636063.

# Publishable executive summary

This report is accompanying the software concerning (offline) 3D self-instruction content on mobile devices. This report is divided into two major parts:

- state-of-the-art research
- instructions for creating self-instruction content

The State-of-the-Art research gives an overview of what is currently available to the public concerning online self-instruction content from the commercial companies. In this part the documentation from large companies with expected high budgets for creating such content has been analysed. Typical companies, clearly focused on such a content as a base for their business case are companies like IKEA, toy manufacturers like LEGO and CUBORO. Since they base their solutions on the customer taking place in the process of the product installation, they clearly put substantial efforts in research on how to present the instructions to the end user. For them is a major task to enable very clear and unambiguous solutions for self-instruction manuals. That is why their approach and experience was considered as important start point in the creating of self-instruction content. Next to the named larger companies and their solutions also input from smaller companies focussing on professionals has been reviewed for completeness.

Functionality found in the different solutions has been embedded as much as possible in the 3D self-instruction solution from INSITER. Clearly the 3D self-instruction solution will not replace the PDF documents, especially when we are looking at professionals. However the solution from INSITER gives a more clear, interactive view on the task to be completed and seems to be unique till this point of time. The leap in knowledge consists in bringing existing 2D self-instruction concept to a 3D or 4D interactive environment available on virtually any modern mobile device.

The issue on visualization of the self-instruction content is solved and the end result is of a higher TRL level than originally targeted. It can be used offline, on virtually any mobile device or computer and is fast and flexible. The complex part however is how to create the content. To enable this an existing solution has been adapted to ensure support for modelling processes, the eventual results can be converted by a dedicated converter and published online.

The real deliverable 4.4 is however the software, the software for creation on self-instruction content as well as the software to publish it online. The software as well as several examples can be found on the INSITER website.

# Fulfilment of the Description of Action (DoA) in D4.4

| Summarized objectives as stated in DoA | Results in this deliverable addressing the DoA |
|---|---|
| WP 4 scope and objectives:<br>- To establish and validate comprehensive BIM (containing components, buildings and sites), and to generate BIM-based self-instruction models for construction and refurbishment, accessible on mobile devices of construction workers and other stakeholders.<br>- To develop a harmonized and standard way to communicate the different entities involved in new construction, refurbishment or maintenance phases of a building. | Addressed:<br>- [100%] BIM based self-instruction models have been created for the Delft use-case. The individual components, their relations and meta data are available and can be edited with RDF Concept Builder. A converter is available to convert the content to the open standard CMO with Extensions.<br>- [100%] tooling is created to convert any content according to the open standard CMO with Extensions towards a HTML5 / WebGL based solution that can be accessed and visualized in 3D on any modern mobile device as well as on normal computers (also in offline modes)<br>- [100%] use of open standard for exchange of BIM related data<br><br>Not addressed in this deliverable:<br>- The harmonized and standard way to communicate the different entities involved as this is mainly part of task 4.1 and within task 4.2 covered in D43 |
| Task 4.2 scope and objectives:<br>- Development of solutions to the existing data interoperability issues<br>- Resolving one of the identified gaps in the assessment methodologies, which is the lack of interoperability among the data sources and evaluation tools with the challenge of big data upcoming. Thus, the main progress in this project is to bridge the gap for feeding and running automatic assessment tools with data inputs from multiple data sources (as for instance databases) in a harmonized way, applying with the “BIG DATA” concept, and supporting evaluation and decision tools.<br>- Dealing with multiple data sources and tools to be inter-connected through data collectors / communication protocols / gateways in a harmonised way for feeding the different assessment tools, allowing interoperability among all the entities involved<br>- Direct / automatic data transfer from the mobile devices via wireless connections, cable or data medium (especially for mass data automatically recorded in certain intervals). The data in these forms are then sent to a server and can be automatically processed. An advantage of this approach is that all instruments can be used simultaneously and valuable metadata can be stored. | Not addressed in this deliverable:<br>- Development of solutions to the existing data interoperability and all sub items are not covered by this deliverable as they are completed within D43. The task 4.2 is divided in two independent deliverables. |

| | |
|---|---|
| Task 4.2 (D4.4) scope and objectives:<br>- Development of self-instruction models based on BIM, in conjunction with the development of training modules in WP6/T6.1, comprising the following activities:<br>- Development of self-instruction content based on research findings; updating the self-inspection protocols.<br>- Development of self-instruction content based on findings in practice and by suppliers; the suppliers will be provided with formats to enable them to create self-instruction for their products<br>- Development of the self-instruction app for mobile devices.<br>- Adding a community feedback tool based on social ratings in the self-instruction app.<br>- Adding self-inspection to the qualification schemes for involved occupations.<br>- Integration of the self-instruction modules into the BIM models.<br>- Generating and deploying BIM-based Augmented Reality (AR) for self-instruction and self-inspection;<br>- Embedding BIM and VR in Augmented Reality (AR), and extracting BIM / VR process information into 'self-instructions' (e.g. installation manuals and planning schedule) for construction workers on their mobile devices (e.g. iPad) | Addressed:<br><br>- [100%] The model created for the Delft demonstration case shows a clear self-instruction model based on BIM / IFC data<br>- [100%] The state-of-the-art research were the base research findings for development of this self-instruction solution<br>- [100%] based on the practical examples from the state-of-the-art research several examples have been developed and available; suppliers are offered the open standard CMO with Extension format as well as 3 different modelling tools to create content<br>- [100%] the self-instruction app can work on the active server as delivered also, however this was not the main research goal.<br>- [100%] as the solution works as a static webpage, the result can be added next to existing PDF's, qualification schema etc.<br>- [100%] as can be found in paragraph 3.3 of this document the self-instruction results can be accessed by a single URL. In cooperation with RE Suite as found in D3.2 these URL's can be added as meta data to individual objects within the IFC file.<br>- [100%] The same way URL's are added as meta data towards individual components within a BIM model, the self-instruction solution can be linked to AR and VR solutions. The real AR / VR applications can be found in chapter 2, this deliverable only shows enabling the solution in a HTML5 Canvas that is a standard object also able to be integrated in other solutions like AR / VR<br>- [100%] process steps can be shown on any mobile device (iPad's, iPhones, Android smart phones etc.), this can also be done offline. After loading the page all process steps will be available for viewing offline; at that time the user can interactively browse through the process steps in both directions in 3D without any need for a connection. |

# List of acronyms and abbreviations

- **AR**: Augmented Reality
- **BIM:** Building Information Model
- **IFC:** Industry Foundation Classes
- **CMO:** Concept Modelling Ontology, an open standard based on Semantic Web for product modelling
- **CMOwE:** CMO with Extensions, CMO extended with technology to support parametric and geometry
- **RDF:** Resource Description Framework (W3C)
- **RDFS**: Resource Description Framework Schema (W3C)
- **OWL:** Web Ontology Language (W3C)
- **W3C:** the organization developing and organizing open standards for the web
- **WebGL:** reduced version of OpenGL API available within HTML5 based web browsers
- **VR / AR / MR:** Virtual Reality / Augmented Reality / Mixed Reality

# Definitions

***Augmented Reality / Mixed Reality***

Augmented Reality is a technology that superimposes the computer generated virtual content over a live view of the world which not only contains the input from the 3D model but may also have sound, video, graphics or GPS data inputs enhancing our perception of real world situations. It has the potential to change how site managers and construction workers can interact and access to virtual information in real-time.

***Building Information Model***

BIM is a method for optimizing the design, execution and operation of building structures. The basis of BIM is formed by a 3D computer model which can be enhanced by adding further information, such as time, costs, utilization. It is not a software package but a method of working, collaborating, designing, managing, constructing and operating.

***Industry Foundation Classes***

Independent standard for describing the building models of various CAD systems. As well as the geometrical data, further properties of the building structures are depicted. The IFC are defined by the Industrial Alliance for Interoperability (IAI). The IFC permit the exchange across different software systems of construction and facility management data.

# Table of Contents

# 1. Introduction and scope

## 1.1 Requirements and scope

This report is accompanying the software as developed / extended / configured for task 4.2. The software itself is the real deliverable 4.4; references to the software are of course included within this document. As no software goes without explanation and the context behind the development is essential to understand why the software is as it is, this report is accompanying the software and plays an important role to explain its functionality.

The scope of deliverable 4.4 is to have software enabling BIM-based Self-Instruction models for mobile devices. The focus is here that as much as possible mobile devices should be able to work with this solution, preferably also working in an offsite situation as well as on normal PC's / Laptops. An important focus for the processed (BIM) content is on using open standards for storage, creation and deployment of the self-instruction models.

The software is created to result in a very generic component allowing an interactive 3D interface supporting different process steps. The resulting component is available via a single URL and works from technical point of view not different than a simple PDF document allowing even the most basic web servers to support it, huge amount of devices supporting it, allowing offsite access and integration in many other solutions including RE Suite, BIM models, AR / VR applications etc.

To achieve this result a detailed state-of-the-art research was applied. This deliverable exists therefore of three parts:

- State-of-art research on what is currently available online in the context of self-instruction content
- References towards the software and content
- Instructions how to use the main modelling package directly showing its capabilities.

### *Context*

To give context for the rest of the document, the end-result of the demonstration case from Delft is shown.

### *State-of-the-art*

To understand the current state of self-instruction content a state-of-the-art research is done. Within the second chapter of this report the result of this state-of-the-art research are discussed as well as what example are found and looked at. The results of this research were used as input for the solution as created.

### *Prototype Software*

In the third chapter the different relevant software packages needed are described. The role of the different software solutions is discussed and the information where the software can be found is included.

The chapter consists of four paragraphs:

- Modelling software, the applications that can be used to create the content
- Deployment software, the application that is able to convert the modelled content into a self-instruction solution
- Client side software, the software that is the self-instruction solution itself
- Content, the examples created available in different open standard formats.

### *Modeller Instructions*

Although the end result, i.e. the self-instruction content, is easy to use, understandable and light weight, the development of this content is complex. The dedicated tooling available for creating the content has a 3D interface however, within an EU project it is impossible to create something even near to a commercial CAD package. This means that modelling is complex and cumbersome. Correct modelling is essential; in case of small mistakes somewhere in the conversion process something could go wrong resulting in empty results.

This chapter tries to explain the ideas behind the modeller and how it can be used to create self-instruction content.

### *Assumption*

As tooling is complete and a state-of-the-art research is done there are clear conclusions to make based on the effectivity and attractively of the created software solution. The bottleneck is here less the eventual resulting self-instruction solution and more the current not commercially usable modeller solutions.

.

# 2. The Delft Demonstration Case

The IFC file of the Delft demonstration case was opened with Archicad software (http://www.archicad.co.nz/). The model is a 3D representation of a building floor.

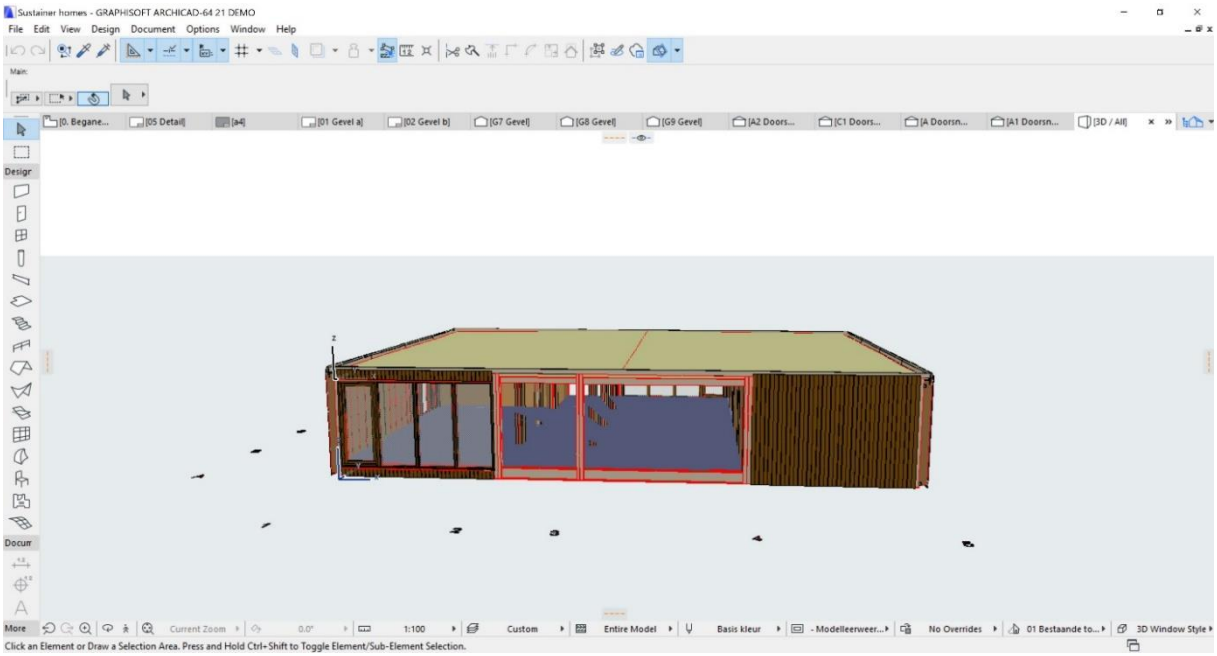


*Figure 1: the Delft demonstration case model visualized with Archicad software*

Since this is an IFC file, all the construction elements can be selected separately which made easy to get the dimensions and the measurements of each element – floor, ceiling, windows and etc. Once all the measurements were taken the next step is to analyse if there is existing knowledge that can be used in that model – model of a colour, material etc. The files that contain the knowledge how to move objects, translate and rotate, (Matrix.ttl) and the file that describes the colours (Color.ttl) are copied to the location of the new project. Translation and rotation we need because, once an element is created, it should be moved from the beginning of the coordinate system to its location in the model.
The second step is to analyse the model and to plan how to create each part. For the purpose of structuring, reusing and easy updates, all the elements are separated in dedicated files. The next step is to take as precise as possible the dimensions of the details. In this case that was very easy as soon as there is already an export in IFC file. When all the measurements are available than every detail is created in a separate file. In that file there are two different types of windows. That's why there are two concepts for windows each in a dedicated file.


*Figure 2, two types of windows available; although they are quite similar, each of them has to be in a separate file, inheriting a common basic structure*

The other unique elements of the model that had been created were the floor, the ceiling, wall without opening, wall with a window. The quality of the model depends on the number of details on it that are modelled. The idea is to model as much as possible details in the construction to make the model more realistic. When the details are complicated and can be separated into a few sub details, than this entire sub details are modelled separately as concepts and later on this sub details are connected to create the complete construction. This is also done when the details have the same parts in their structure.

In this model there are few parts that are reused in the complete model. They are created in separate files, they are assembled in additional file. The result detail is combination of two or three sub details. In this specific case the wall with opening inherits the wall without opening. The opening is created with the size of already created window and then the window itself is placed at the place of the opening. That is a new detail. When this wall with a window is found in the model, the modeller does not have to create the piece from scratch but can just reuse the already created piece of wall with window inside. Usually the benefits are visible in more complicated models, but the Delft demonstration case can present also very well the sufficiency of this approach due to the existence of structures and components that are similar but not the same.

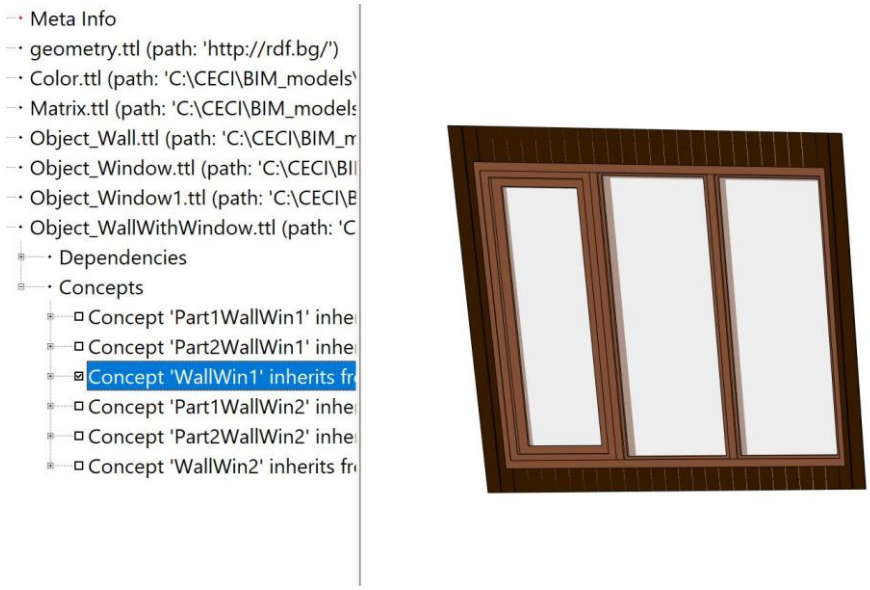


***Figure 3, the window is created from subcomponents, but also together with the wall it creates a new detail which can be reused***

Once all defined details of the Delft demonstration case were modelled, the next phase was to connect them into the model itself. This was done step by step. Each connection is structured in a dedicated file which inherits the previous step and the new item that should be added to the already built structure.


***Figure 4, step 15***

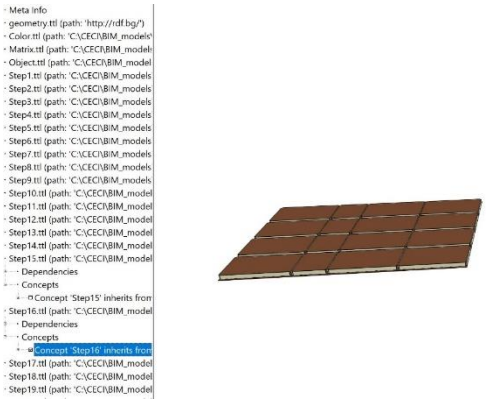


***Figure 5, step 16***

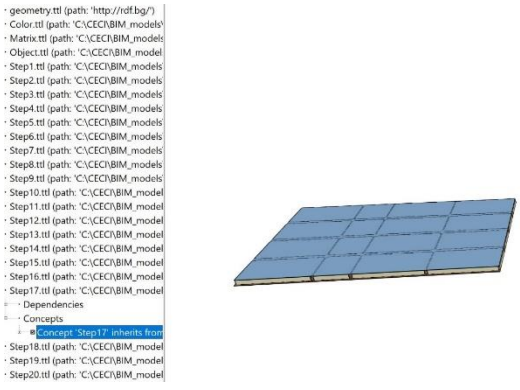


***Figure 6, step 17***

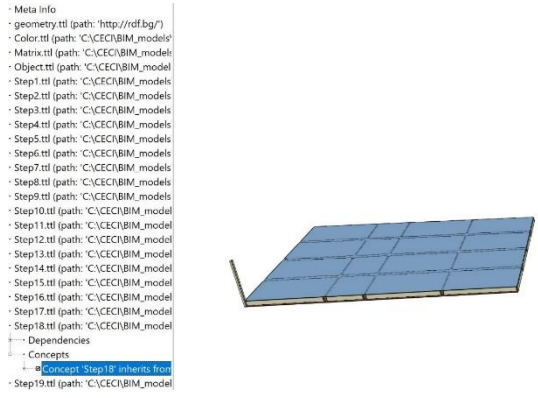


***Figure 7, step 18***

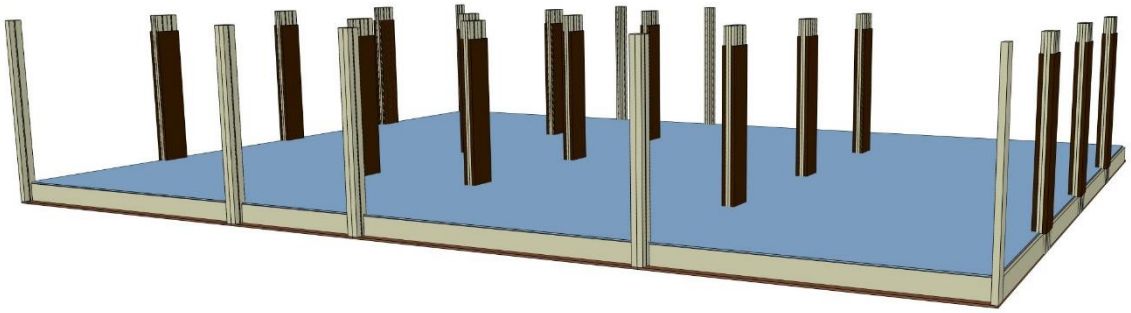

*Figure 8, the Delft demonstration case, floor and columns*

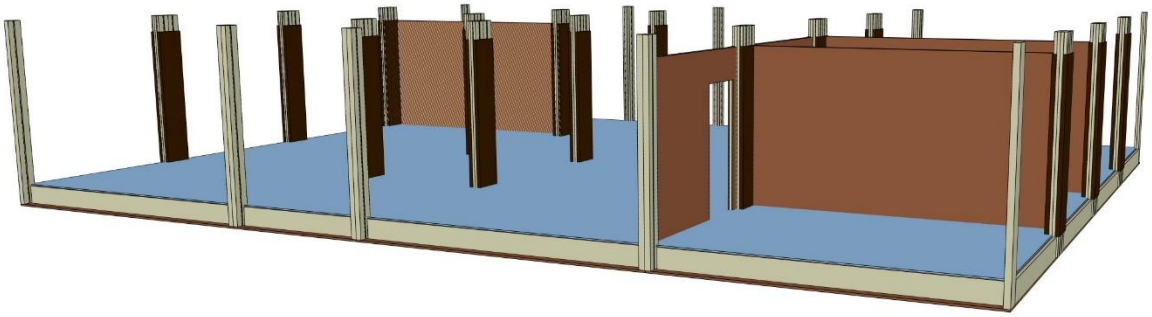

*Figure 9, the Delft demonstration case, assembled floor, columns, inside walls, inside walls with opening. Most of the elements share knowledge in common*

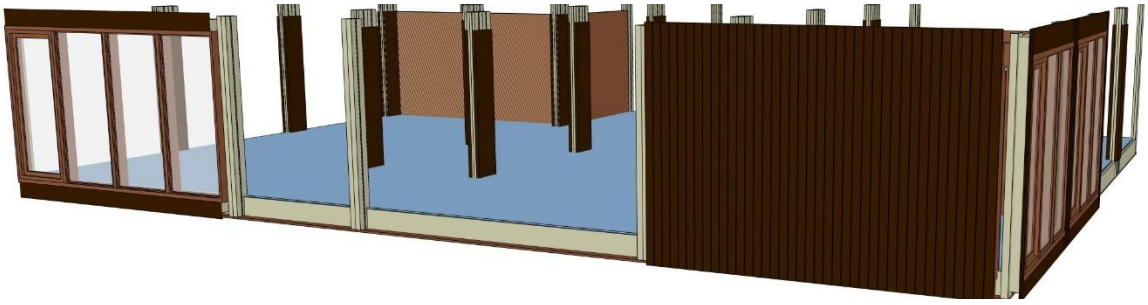

*Figure 10, the Delft demonstration case, most of the outside walls and windows are attached to the construction*

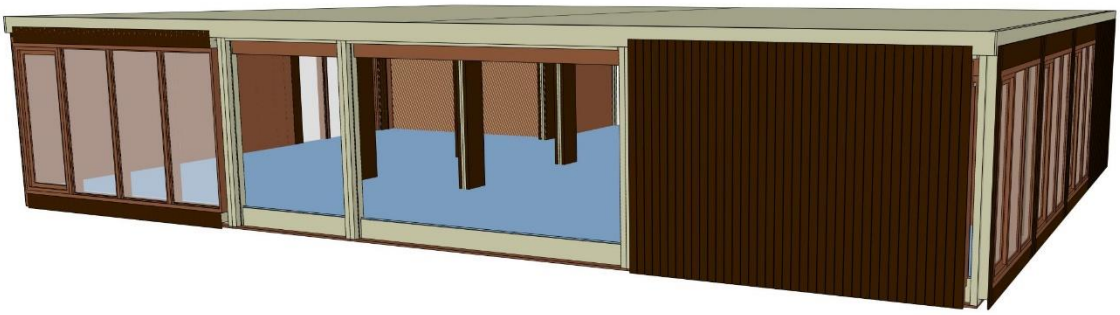

*Figure 11, including all properties and elements like the model from the IFC file*

# 3. State-of-the-Art

This chapter concerns the research made on the existing types of user manuals that are available. Most of the reviewed documents were available online for free download. The goal was to review as much as possible documents focused on different topics and audience and to extract the knowledge how to structure guidelines that is clear, complete and non-ambiguous for the user. The outcome of this first step was to select a number of examples and type of manuals that will be the main focus of the research. During the process it was observed that most of the manuals that were checked do not require specific prior knowledge, although the selected samples have several different purposes and introduce different type of products. For example, in the study were checked manuals for installing professional equipment and appliance, manuals that are focused at the end user, manuals to install / assemble furniture and equipment for kitchens and etc. Very interesting for our goals were the instructions to build models with pre - build blocks like Lego, Cuboro and IKEA. A lot of interesting knowledge was found also in the car operational guides.

During the process, the research focused on the manuals that do not require specific prior knowledge by the user. The reason this kind of manuals were selected is that they are widely used, very well thought of to be able to give the right steps to a non - professional to do the work that normally would require professional knowledge. They are normally very detailed, very clear and non-ambiguous. Therefore the assumption was that they can give the right building blocks and experience to create the correct way of an automated self-instruction manual. For completeness of the results, were contacted international engineering companies and asked to provide examples of manuals focused on professional users. Unfortunately the information in this guides is confidential but this does not affect the results due to the main purpose was to get an idea for the manner and the structure that is used in such a document, for the completeness of the information presented there but not for the contents itself. The research was facilitated due to the fact that almost every company that creates products for end users publish the manuals online, available for public access and download .In many cases the instructions were available in widely used file format - PPT or PDF. It is also common to show the instruction in an application that can be downloaded and installed on most of the end user devices – laptops, tablets, smartphones. An interesting approach is to have the instruction as a 3D model on a web site.

## 3.1 Most common features

The expert writers of user manuals advice before starting to create a user guide to analyse the audience it is meant for. The typical user of the manual should be defined. The content should be addressed to the typical user. It should explain 'how to'. It should not go into unnecessary details and should give sufficient explanation in the same time. A good advice is that is better to improve the interface and to automate some tasks in the process rather than write a long user manual. During the study it was observed that the documents are written using a similar approach. There are a number of common features that were found by analysing a number of documents with different purposes. Some of the most common features found during the research:

The manuals start with the most important chapter and go towards the lower prioritized issues. The answers of the more frequent questions come before the less often asked. The chapters that are outside of the scope of the regular user, are often mentioned at the end of the instruction. For example, how to give a basic technical support to a car is described in

one of the last chapters, assuming that most of the users will ask professionals to solve technical issues. When the manual is meant to lead the user to install an equipment, than as the most important, it starts with safety instructions to avoid situations that might be dangerous for the operator.

Most of the manuals just lead the user through the process, step by step. Explaining each level separately- (installing operating system, connecting a washing machine).

The manuals explain: How to do a certain action, rather than what it is, what that action means.

For the big part of the instructions in the beginning is provided a list of materials and elements that will be needed during the process and / or set of tools, when applicable.

What is common in all the manuals is that the information is structured. Every topic is a separate structure. It can be read as part of the total instruction but also can be seen separately. The manual can be used as a reference, just for a quick check of a certain issue. In the more complicated manuals there are also references between the steps.

The manual starts with the most important. It gives the scope and purpose of the instruction. Then follows the preparation – what kind of tools are needed, what materials and items shall be prepared to complete the process?

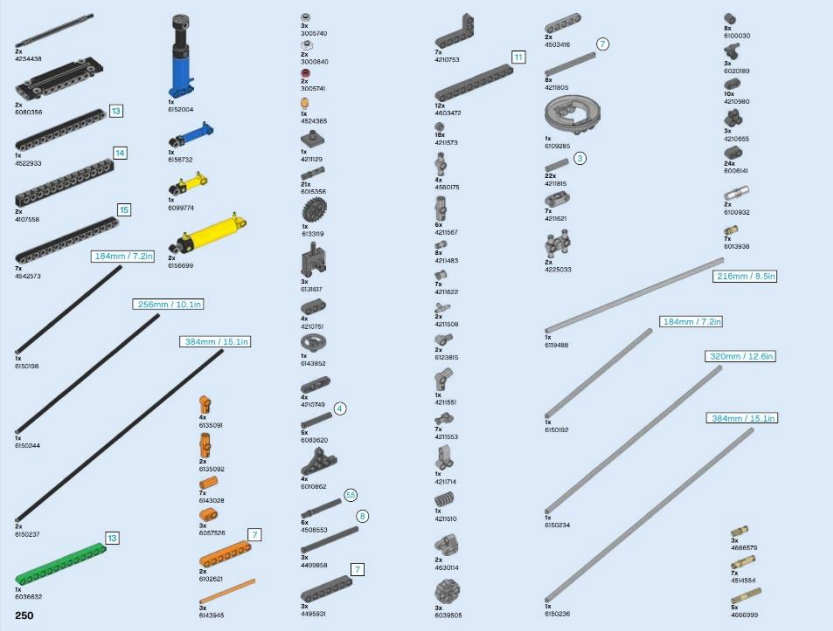


***Figure 12, an example from a manual that shows a list of parts which will be needed to assemble the model. A lot of the studied manuals follow that concept. First show what is needed and then how the elements are assembled step by step***

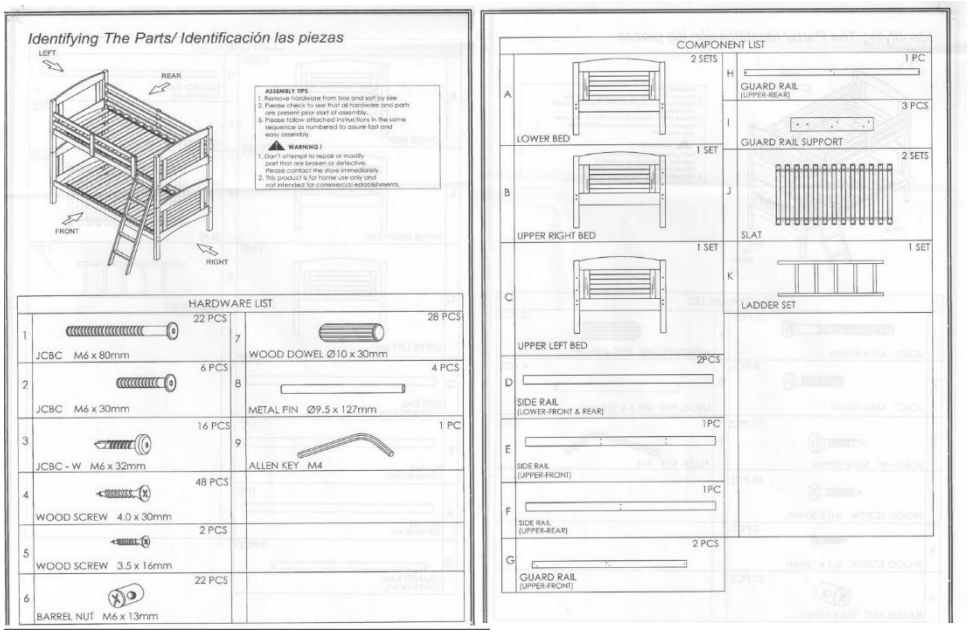


*Figure 13, manuals for products from different areas are using the same concepts*

The helicopter example is simple way to represent most of the ideas described in the report. This is the reason why a lot of effort is put in developing this example. The end-result will not be useful for INSITER, however the steps needed to come to the end-result teach us valuable lessons learned that drive the software development

*Figure 14, a model of helicopter of Heros toys*

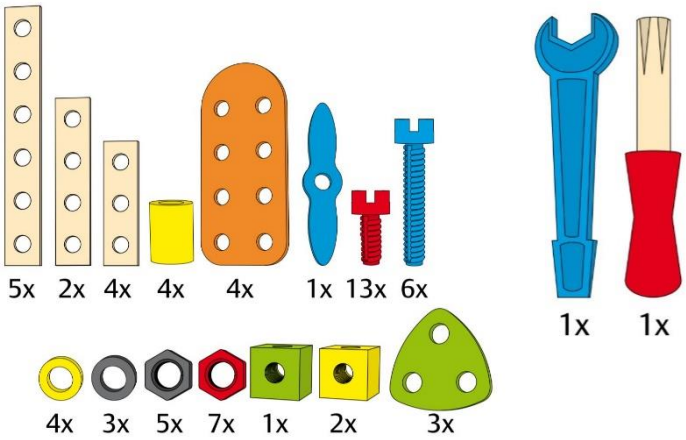


*Figure 15, showing all the elements and how many of them are needed to create the model. The set of tools that is necessary to assemble the helicopter is also shown*

Some of the most common features found during the research:

- The independent actions are expressed independently;
- The manuals are written in a way, that they give quick answers to the user questions. The description is brief but just enough for the user to complete the step;
- Each step starts with an imperative word that shows the user action – click, open, etc. This way it is clear for the user what should be done. The details that gives the user the reasoning behind the step, follows. The system response is not specified as a step. When necessary, the response is incorporated within the step that initiates the response;
- The order of the actions that should be performed by the user is always clear. The steps follow each other, the steps have numbers, the pages have numbers;
- The text is standardized. When there is more than one way to spell a word or create abbreviation, refer to an object, caption graphics, punctuate sentences, lay out a page, and organize information the choice is made that applies for the entire document. It could be said that a standard is established for the entire document. This obviously makes the presentation much more clear and understandable for the user. But also seems to make the task of the writer much clearer, because the frame is set up. No choices should be made, just predefined rules are being followed while creating the document;
- When the user would need a predefined set of tools and parts to complete the instructions in the manual at the beginning of the instructions this set is shown together with the parts. The number of the blocks of each type that will be needed are indicated. In longer instructions this independent can be seen a few times. The work is separated by modules that can be seen as independent. Before starting a module there is an overview what should be performed in the following step, then the set of tools and the needed parts;
- The pictures show precisely how the connection between the individual parts should be performed

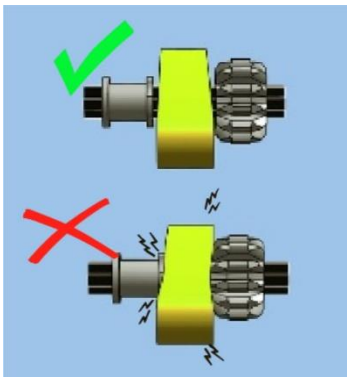

*Figure 16, the right way to connect the details is graphically presented*

- To be distinguished easier, the details are used in different colour.
- Sometimes all the details that have to be moved towards the all construction in the manual are shown in a predefined colour that difference from the colour of the rest of the objects. The direction they should be moved towards the construction or the details towards each other is shown often with arrows. Sometimes it is marked where exactly the user has to make the connection. In the more complicated cases the connection is detailed presented with a picture or separated on steps. In the more complicated models the items are shown also with its colour.
- To make clear what object should be chosen the details are shown also in scale 1:1 (with their real dimensions). The user can just put the selected object on the paper and check directly if the one that was chosen is the item that is needed. This is really helpful optimization that increases the speed and reduces the complexity of the task that the user has to perform;

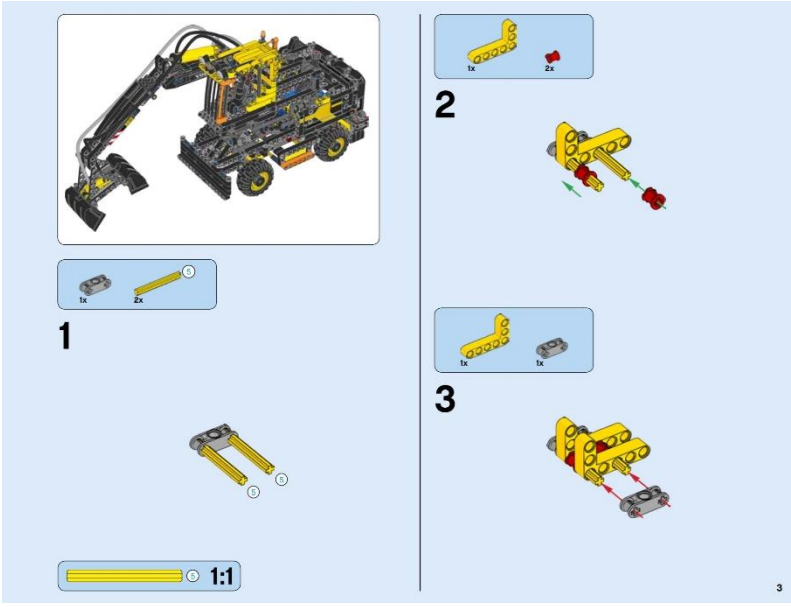


*Figure 17, how to move the details towards each other to connect them properly is shown with arrows*

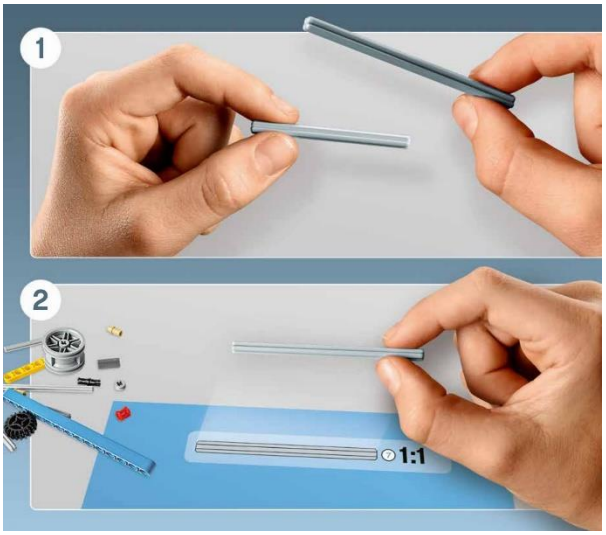


*Figure 18, the part is shown in scale 1:1, when there are few similar parts by placing it over the picture the user can chose very easy the right one*

In the professional user guides although the content is more complicated, requires a certain pre-knowledge, the main principles of writing a user manual are also observed. The content is strongly divided by chapters and the most important comes first. The process that should be completed is organized as individual steps. All knowledge that belongs to a certain group is organized in a way that all the possibilities are seen at once. For example all the possible connections for the electricity outlet are shown in a table. This way the use can decide easily which one he needs and find out the attached specification.

In professional user manuals, the content is organized in a way that the user can access easily the information when he needs to use the manual as a reference by selecting the needed chapter from the table of contents.

## 3.2 Applicability in INSITER

Within this state-of-the-art research most of the focus was on examples from other industries, even looking at many non-professional end-users. The main idea behind this was to look at the typical solutions of large companies with a strong interest in self-instruction virtually not bounded by limited funding for the self-instruction.

The clear and reusable parts are identified and reused for the INSITER solution. It is expected that INSITER can benefit this way indirectly from the knowledge from other sectors and large vendors without reinventing the wheel concerning needs from professional and non-professional users.

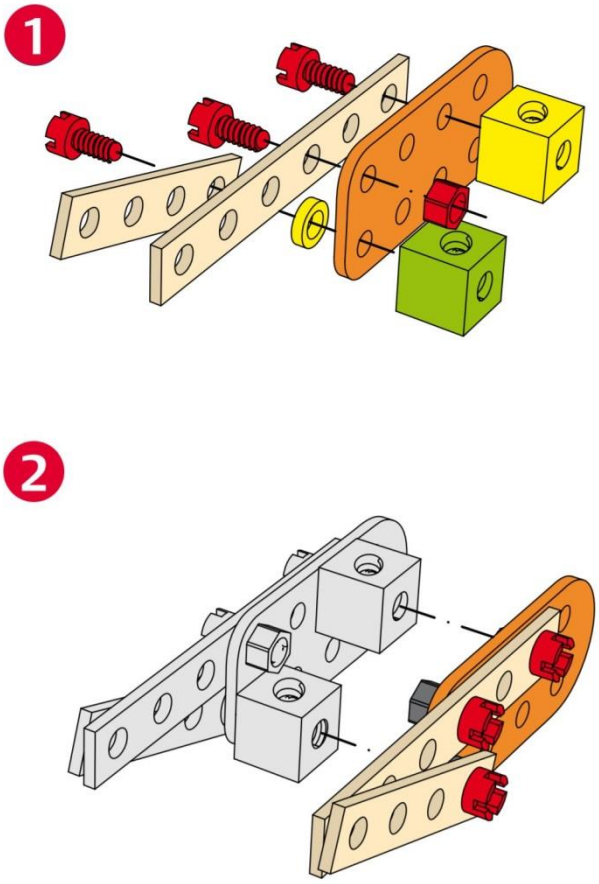


*Figure 19, the direction and the place where the items are connected is shown, the already montaged part is shown transparent. The part that should perform an action with is in colour*

# 4. Prototype Software

## 4.1 Modelling Software

The solution is following the open standard CXMO with Extensions for parametric geometrical knowledge based on Semantic Web technology. This means for creation of content both commercial and open source solutions can be used to develop content. Although this is true, it is only true in theory. Creation of geometrical representations without direct feedback on how the geometry looks like is a very difficult task if not impossible even for small projects.

This is why all our models for testing are modelled using a dedicated software the RDF Concept Builder that is an updated version from already existing prototype geometry modeller now able to model the processes as required to be visualized in the self-instruction content.

### 4.1.1 RDF Concept Builder

This is the main software to create content for the self-instruction 3D interface. The software can be downloaded from here:

https://www.insiter-project.org/Shared%20Documents/10%20WP4/D44/Modelling/RDFConceptBuilder.zip

Within the INSITER consortium, the source code of this application is available and can be downloaded from here:

https://www.insiter-project.org/Shared%20Documents/10%20WP4/D44/Modelling/RDFConceptBuilder-VisualStudio-2013.zip

The explanation how to use this software can be found in the following chapter.

### 4.1.2 INSITER 2TTL converter

This conversion tool allows any model created in the RDF Concept Builder to be converted into content valid against the open standard CMO with Extensions. This tool is available from this location:

https://www.insiter-project.org/Shared%20Documents/10%20WP4/D44/Modelling/2ttl.zip

Within the INSITER consortium, the source code of this application is available and can be downloaded from here:

https://www.insiter-project.org/Shared%20Documents/10%20WP4/D44/Modelling/2ttl-VisualStudio-2013.zip

Important to note is that this is a converter in beta stage and it is not able to create directories by itself. If content needs to be converted into a directory, this directory has to be created by the user otherwise the creation of the content will be skipped.

### 4.1.3 TopBraid Composer

TopBraid Composer is a commercial software package to show, edit and create ontologies. This means also our content can be edited and created using TopBraid Composer. Although creation of complex models is almost impossible without having a direct 3D interaction like is possible with the RDF Concept Builder the TopBraid Composer software can be very relevant to make small edits or additions. Also as it supports the full CMO with Extensions capabilities the freedom in modelling is larger than what is possible with the RDF Concept Builder.

The tool can be downloaded via the official page of TopQuadrant: https://www.topquadrant.com/tools/ide-topbraid-composer-maestro-edition/

At the moment of writing this deliverable it is possible to download a free version of the TopBraid Composer as well as a 30 day trail of the full version; the free version is powerful enough to handle the models used within INSITER. Usability of the tool is clearly described on the website.

#### 4.1.4 Protégé

Protégé is an open source software package created by Stanford University. Protégé is a free, open-source ontology editor and framework for building intelligent systems. It can be downloaded from here (registration required):
https://protege.stanford.edu/
Usability of the tool is clearly described on the website.

## 4.2 Deployment Software

Once the content is available in the correct format it still is not possible to visualize it in 3D. To make this possible, a deployment software is required that reads and understand the content and can convert it in something that can be understood by HTML5 / WebGL browsers.

The software doing this is actually packed in a server. The server needs to be started to apply the conversion. This server could be used to apply this conversion 'on-the-fly'. However, in most cases it is easier to just apply the static content as is done with PDF files. The ability to use the results also without active server solves a lot of security issues and makes deployment of the results literally as easy as deployment of a static PDF file.

The server can be downloaded from here:
https://www.insiter-project.org/Shared%20Documents/10%20WP4/D44/Deployment/SelfInstructionServer.zip
The server contains a server.docx document explaining in more detail the settings of the server. The server can be unzipped on any location within a Windows machine and runme.bat can be started to run the server. By default the server will use the http://127.0.0.1:2233/webgl/ address and it uses by default the Delft demonstration case, however this is configurable using the server.settings file.

Please note that due to being prototype software it will take a few minutes to process all data the first time. Starting up the server should not take longer than 10 minutes, once the server is running everything should work almost instantly.

## 4.3 Client Side Software (Results)

The actual client side software is the HTML / JAVA Script content that is fully HTML5 / WebGL compatible and runs on any modern mobile device as well as any PC / Laptop. The software is adaptable towards the used system; it has support for mouse interaction as well as support for (multi) touch interfaces. The software is adaptable, this means that on a small window it will have a slightly different interface more useful for phones and iPads while on devices where high resolutions are possible the interface will automatically adapt the interface to better fit this resolution.

Several examples are available on the INSITER SharePoint website, the Delft demonstration case can be found here:
https://www.insiter-project.org/Shared%20Documents/10%20WP4/D44/Results/Delft/index.html
or publically available from here:

http://rdf.bg/clean/Delft/

To visualize in more detail the capabilities of the solution (based on the state-of-the-art research) the following example is developed:
https://www.insiter-project.org/Shared%20Documents/10%20WP4/D44/Results/Lego/index.html
or publically available from here:
http://rdf.bg/clean/helicopter/ / http://rdf.bg/clean/webgl/

## 4.4 Content

The real proof is in the pudding, therefore three examples are created using above named tooling. The content is available on the INSITER SharePoint and can be used to create new solutions as well as check how the software is working.
The Delft demonstration case example can be found here, both respectively the RDF Concept Builder solution as well as the converted CMO with Extensions content:
https://www.insiter-project.org/Shared%20Documents/10%20WP4/D44/Content/Delft-RDFCB.zip
https://www.insiter-project.org/Shared%20Documents/10%20WP4/D44/Content/Delft-CMOwE.zip
The Heros Toys example can be found here, both respectively the RDF Concept Builder solution as well as the converted CMO with Extensions content:
https://www.insiter-project.org/Shared%20Documents/10%20WP4/D44/Content/HerosToys-RDFCB.zip
https://www.insiter-project.org/Shared%20Documents/10%20WP4/D44/Content/HerosToys-CMOwE.zip
The Lego example can be found here, both respectively the RDF Concept Builder solution as well as the converted CMO with Extensions content:
https://www.insiter-project.org/Shared%20Documents/10%20WP4/D44/Content/Lego-RDFCB.zip
https://www.insiter-project.org/Shared%20Documents/10%20WP4/D44/Content/Lego-CMOwE.zip

# 5. Modeller Instructions

The most popular tool available on the market, like TopBraid Composer supports RDF schema, RDF data, OWL ontologies, visual editing and querying has a focus that is quite different than the task of creating a 3D model by describing its properties in RDF files. The task of creating even a simple model seems to be almost impossible.

To complete this sort of task, a program that is focused on 3D modelling is needed. Such a dedicated tool is the in-house developed RDF Concept Builder. To create a small model with RDF concept builder takes few day. The reason is that the details should be mathematically described. The visualization is based on calculations. The modeller gives all the formulas that should be used to calculate the model. The tool is visualizing the content based on that calculations. This is a bigger effort than to use a cad program but the benefit is that the model can be parametric. When the value of a parameter is changed, the model is recalculated directly based on the dependencies given by the modeller.
It is also a bit more difficult when the model is not available physically or when the dimensions are not specified in a documentation. In this case there is an extra effort to find out what are the right measures or proportions between the different sizes to be able to make the model realistic.
Another time consuming issue could be a complex movement, that has to be attached to the model or there are few dependent from each other moving parts. First the exact movement has to be identified, the task for the movement should be brought to abstract representation, the dependency should be found and at the end it should be mathematically represented into the conceptual representation of the model. There should be also sufficient test that should prove the concept is designed correctly against the requirements – the object is moving how it is expected.

A good example to present the process is to describe how a LEGO set is modelled. It has a lot of parts that are standard, parts that can be reproduced from the same basic details and at the end but not at least, it also has mechanical movements that should be attached to the model.

*Figure 20, LEGO set SET-42061*

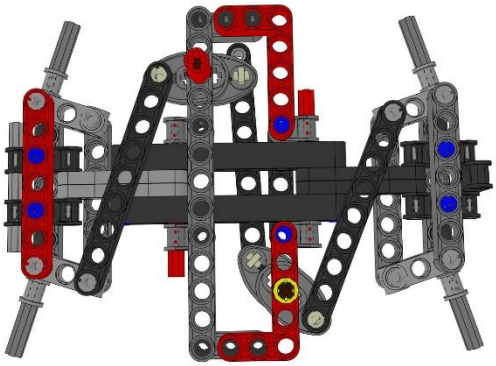

*Figure 21, when one detail is moved, the entire construction performs a complex movement. Each part has to be accompanied with the knowledge about how to move it and how this movement depends parametrically from the movement of the other objects. Only this way the complicated movement of the whole structure can be recreated*

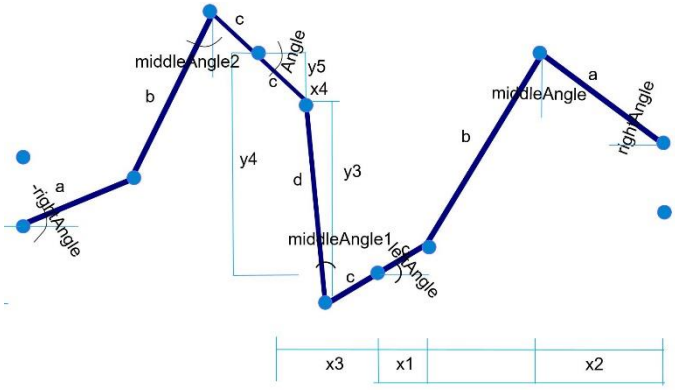


*Figure 22, here, the modeller has represented part of the task that has to be solved*

x2 = Cos ( rightAngle ) * a
y1 = Sin ( rightAngle ) * a
x1 = x - x2
b = √ (y+y1)² + x1²
leftAngle = ArcSin ( (y+y1) / b )
p= ( a + b + c ) / 2
*S = √ p * ( p – a ) * ( p – b ) * ( p – c )
h = ( S / c ) * 2
c1 = √ b² - h²
c2 = √ a² - h²
leftAngle = ArcSin ( h / b )
rightAngle = ArcSin ( h / a )

*Figure 23, formulas*

Based on the previous figure 19 these equations are defined, these are described in the RDF Concept Builder and has to be attached to the model.

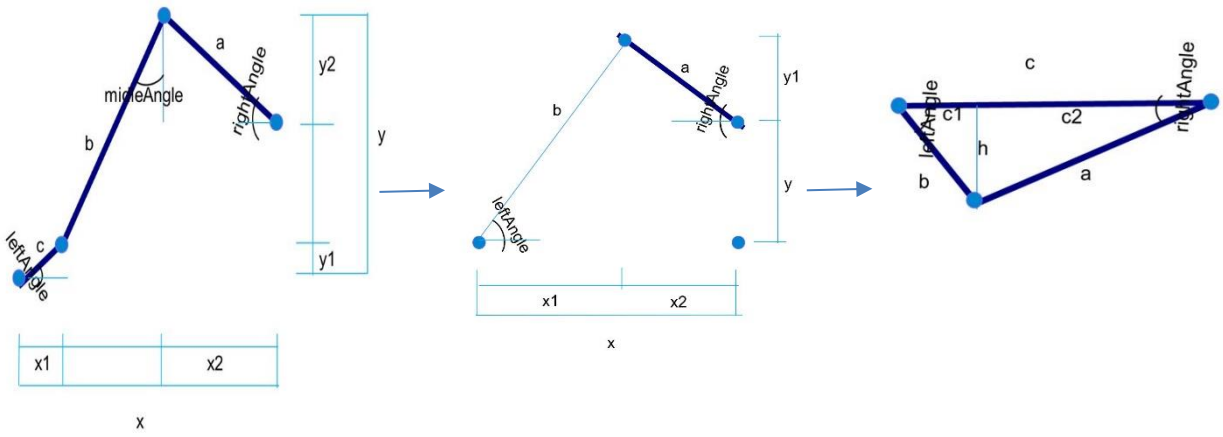


***Figure 24, the task is still not easy to be solved completely at once. That is way it is separated on few subtasks that are relatively easy solvable***

One of the most complicated tasks in this model is to recreate the movement of the basket and to model the complex sequence between the parts that ensures the drive of the excavator itself. The information how to do that it is not given in the instructions, but it is something that the modeller has to find out herself / himself.

The first stage of a modelling project is planning. The structure that should be modelled is brought into parts. Each part is created separately. The process is divided into individual steps. An analyses what can be reused from already created models is performed. There is also a check if the quality can be improved. If the files can be used in the original state they are then copied into the location of the new model. Normally these are files that are often repeated, like in every model a typical files that can be reused and are reused often are files that define colour, matrix and etc. These are the basic properties and operations that can be done over the model. Of course during the modelling process it appears sometimes that this files could be updated, improved or new knowledge should be added. These basic files are all plain text files and can be open with a common text editor like for example notepad. Unfortunately it is very difficult for a reader/modeller to understand and debug the files when they are open just as a plain text. A more advanced editor is needed to preview the model and to be able to change its parameters to add new knowledge and develop new structures. The Top Braid composer is more advanced but still his purpose is more common. For the purpose of creating, previewing and updating 3D models an editor is needed that is developed towards the specific requirements of that task. That is why an indoor developed application is used in that task.

The screen below shows a file for colour definitions. In this case it is easy to identify what parameters to change but it is not always easy to understand if the result is appropriate and if there are errors in the content.

```
save version: 0.3
<import uri="http://rdf.bg/geometry.ttl" prefix="geometry">
<concept id="BrownComponent" description="none">
  <parent id="geometry.ColorComponent">
  <propertyRef id="geometry.W" minCard="1" maxCard="-1">
  </propertyRef>
  <propertyRef id="geometry.R" minCard="1" maxCard="-1">
  </propertyRef>
  <propertyRef id="geometry.G" minCard="1" maxCard="-1">
  </propertyRef>
  <propertyRef id="geometry.B" minCard="1" maxCard="-1">
  </propertyRef>
  <ruleRef content="geometry.W = 0."/>
  <ruleRef content="geometry.R = 0.218"/>
  <ruleRef content="geometry.G = 0.218"/>
  <ruleRef content="geometry.B = 0.182"/>
</concept>
<concept id="BrownColor" description="none">
  <parent id="geometry.Color">
  <relationRef id="geometry.specular" minCard="1" maxCard="-1">
    <selectedConcept id="BrownComponent">
      <propertyRef id="geometry.W" minCard="1" maxCard="-1">
      </propertyRef>
      <propertyRef id="geometry.R" minCard="1" maxCard="-1">
      </propertyRef>
      <propertyRef id="geometry.G" minCard="1" maxCard="-1">
      </propertyRef>
      <propertyRef id="geometry.B" minCard="1" maxCard="-1">
      </propertyRef>
    </selectedConcept>
  </relationRef>
  <relationRef id="geometry.emissive" minCard="1" maxCard="-1">
    <selectedConcept id="BrownComponent">
      <propertyRef id="geometry.W" minCard="1" maxCard="-1">
      </propertyRef>
      <propertyRef id="geometry.R" minCard="1" maxCard="-1">
      </propertyRef>
      <propertyRef id="geometry.G" minCard="1" maxCard="-1">
      </propertyRef>
      <propertyRef id="geometry.B" minCard="1" maxCard="-1">
      </propertyRef>
    </selectedConcept>
  </relationRef>
  <relationRef id="geometry.diffuse" minCard="1" maxCard="-1">
```

*Figure 25, example of a file for definition of colour*

When the file is opened in application as RDF Concept Builder, the geometry is visualized; the knowledge and the dependencies are shown in a way that it is possible to the modeller to change the content, to debug and to add new knowledge. The structured preview of the content allows the user to identify the different parameters, to access the information, which needs to be changed easily and to preview the result of the action performed in 3D directly after each change.

The use of RDF Concept Builder requires a bit of explanation to be able to understand the flexibility and the potential of a tool that gives the possibility of preview, ease of access to the parameters but still the freedom the modeller has in the approach.

A file that contains predefined definition of colours opened in RDF Concept Builder looks like shown on the picture below:

Three steps must be performed to create a new colour before it can be used in the definition of an object. The first step is to make a colour component. This concept contains the definitions of the tree basic values in RGB (red, green, blue).

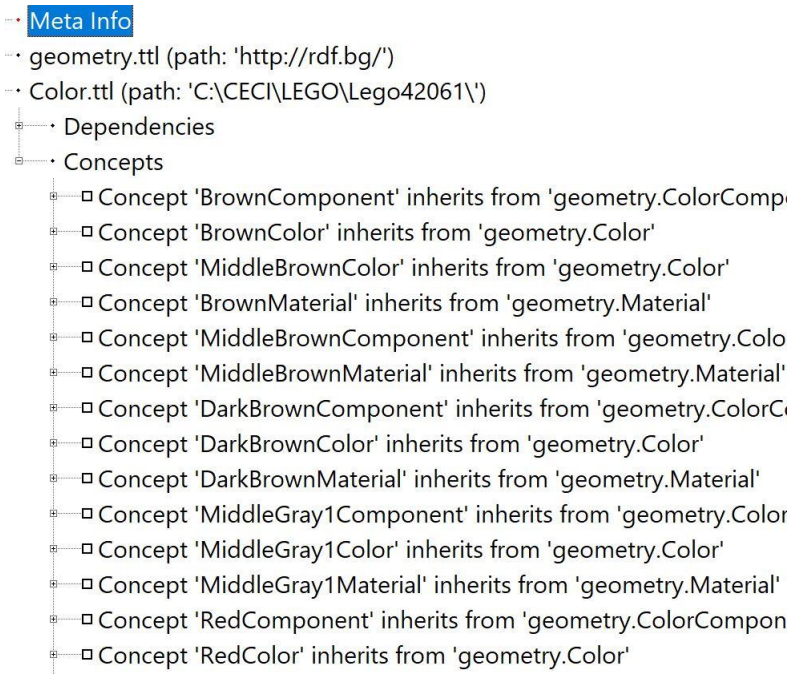


*Figure 26, a file opened with RDF Concept Builder*

Transparency is also defined there by giving of value of the component 'white'.

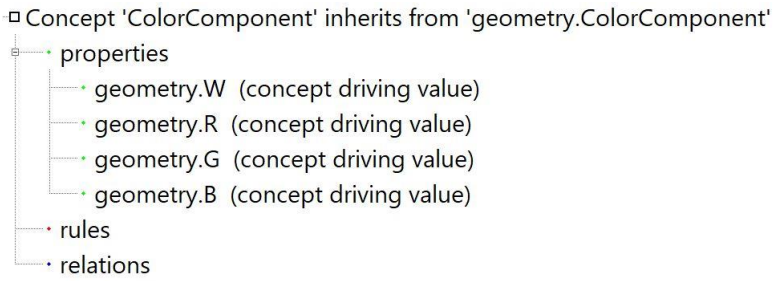


*Figure 27, concept ColorComponent in RDF Concept Builder*

Once the values of the colour components are set up in rules, than this is a fixed. The second step in the process of creating a colour is to create a colour concept. By this the values of specular, emissive, diffuse and ambient are also fixed. The colour concept inherits the colour component. By this the colour can have different 3D properties.

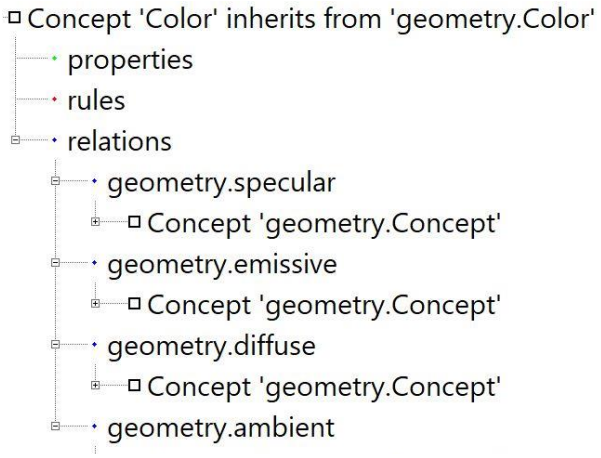


*Figure 28, a concept of a colour*

The last step is to define the material. Colour and Material are inheriting from the Appearance concept. This material can be used into different objects.

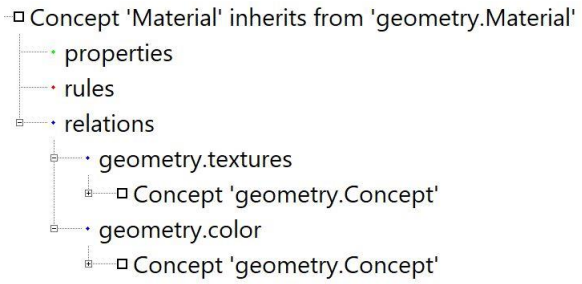


*Figure 29, a concept of a material*

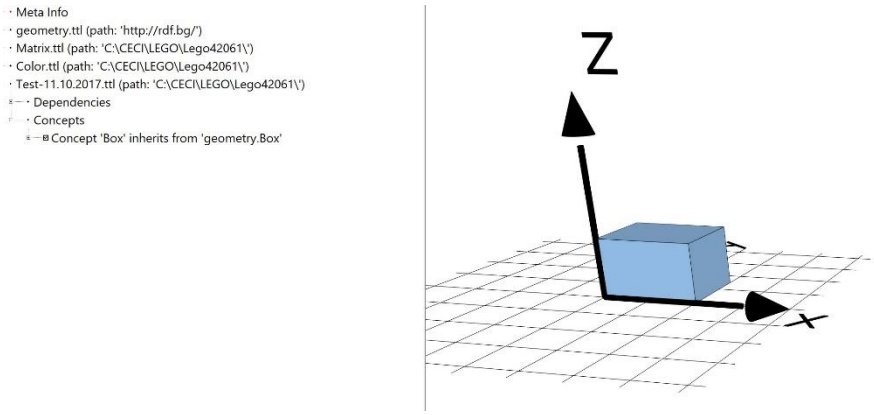


*Figure 30, a box with blue colour that is positioned in the beginning of the coordinate system*

The set of matrices created to support the movement of the objects – rotation and translation, is also reused.

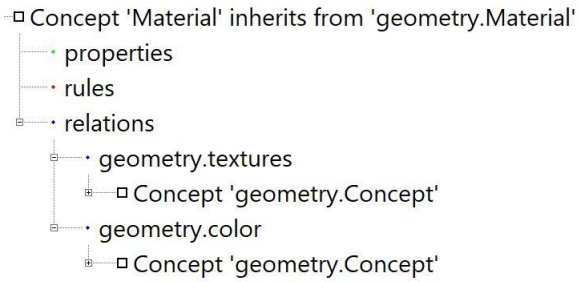


*Figure 31, a definition of a matrix for movement*

The Concept Translate ensures the movement by XY and Z. RotationXY means that the item is rotated by XY direction with the required degree of rotation. The following examples show the change of the object when the parameters are set to new values. The box is moved by X, Y and Z.

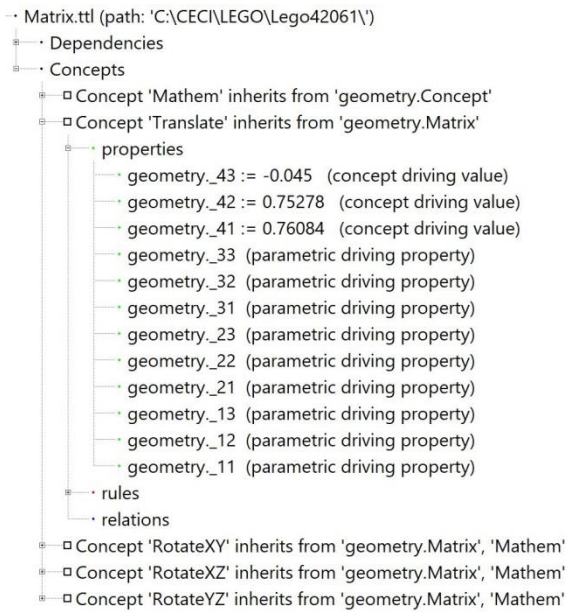


*Figure 32, translation matrix _41 is the translation by X, _42 by Y and _43 by Z*

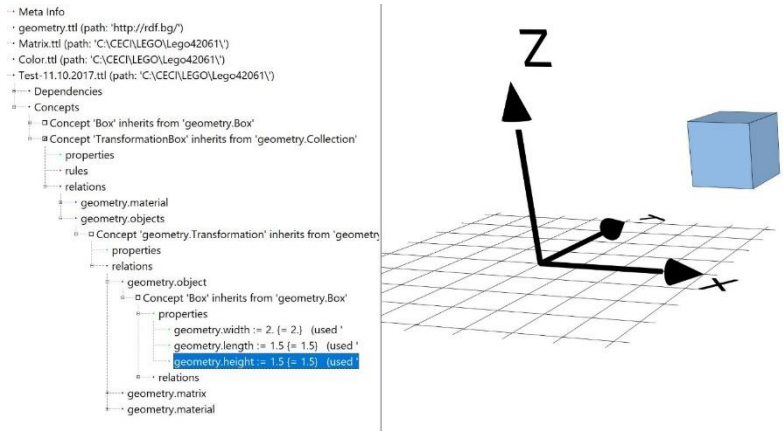


*Figure 33, the result of the applied translation*

The following example shows a rotation of an object. The given value is the angle of the rotation in degrees. It will be internally transformed to radians.

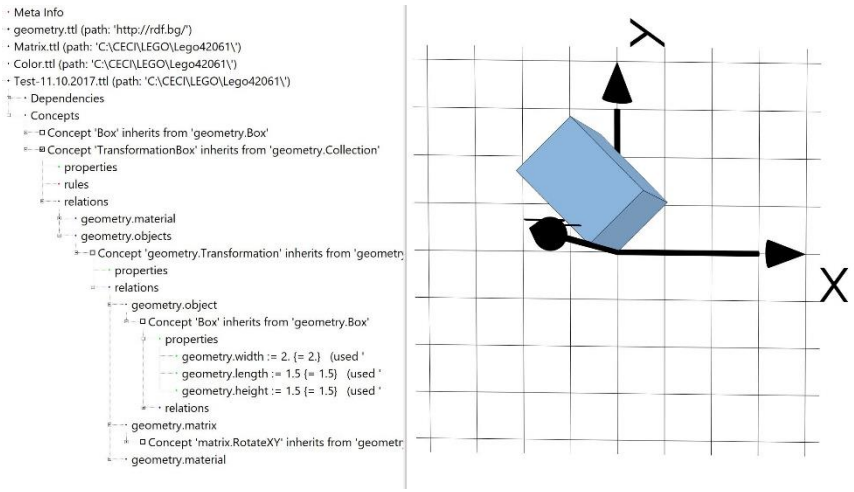


*Figure 34, an example of rotation in XY plane by 45 degrees*

RotationXZ and RotationYZ is the same like RotationXY only the direction is different.

The next step is to analyze what kind of basic objects and operation over them are needed to create the model. When the items are standard for the type of model, often most of the parts are already modeled and can be used from the existing library. When they are not already created, than often they can be modeled by using knowledge from existing models with a number of add-ons. This is possible because every detail is kept in a separate file. It is easy to redraw it, add small changes with generic editor like notepad and that open it in RDF Concept builder where the connections between the inherited and the derived concepts can be explored and changed when necessary. In the real case scenario, in the rotation will be used combination of rotation.

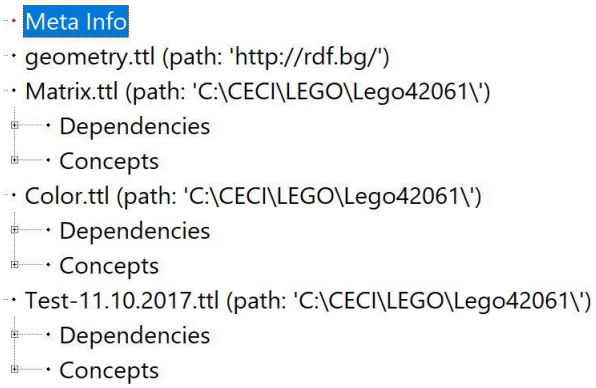


*Figure 35, a file that inherits from other files*

First, there should be a list of all the details/components that are needed to assemble the model. In some cases this set is already available with all needed properties; there are also projects where a choice how exactly to separate the object on components should be made.

Once, the details are clear, in the planning phase the plan which details can be used directly and which details should be derived from previously modelled components is made. Very often if the detail is not modelled, the differences with existing item are very small. Sometimes the details just have different colour from already modelled items.

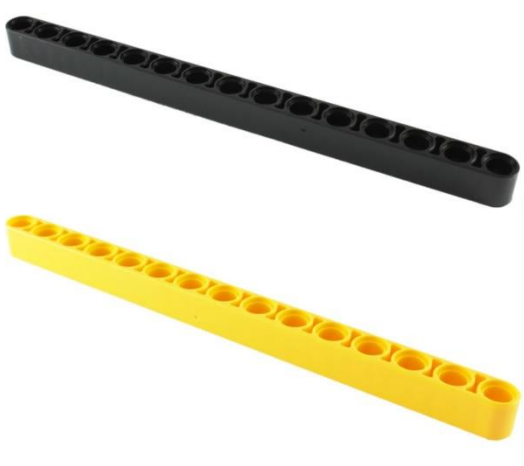

*Figure 36, An example that differ only by one property. For the rest they are exactly the same*

The samples used for the modelling process are not physically available always. The most difficult case is when not all the sides are available to observe. Sometimes the detail is available only at 2D from one side. It is not clear what is behind the visible surfaces. This is the most difficult case to model because the mistake in defining the dimensions and the shape could vary and originate from several mistakes at different places in the model.

*Figure 37, the picture is an example of details that only differ in the number of the subcomponents they are created from*

The perfect case scenario is when schematic drawing with all necessary dimensions of the details are available. Once all the details from the list of items are available, they have to be connected to each other properly. When all of details in the list are drawn, start connecting them as shown in the menu. Each step of connecting the parts is shown in the set menu. Some of the steps are shown with more connection details.

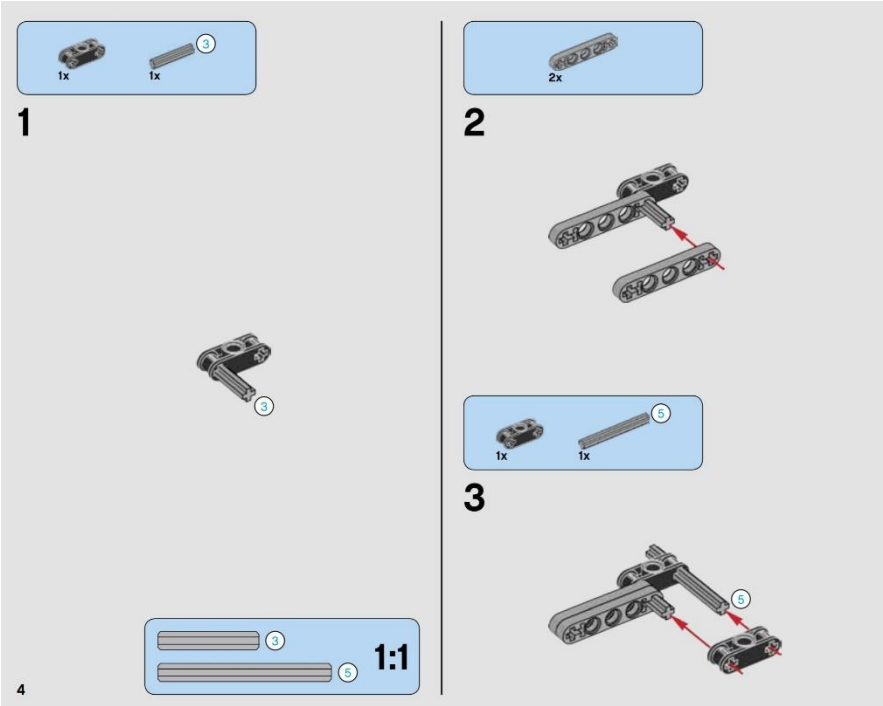


*Figure 38, the picture shows a page of the user manual which shows how the first details of the model should be connected*

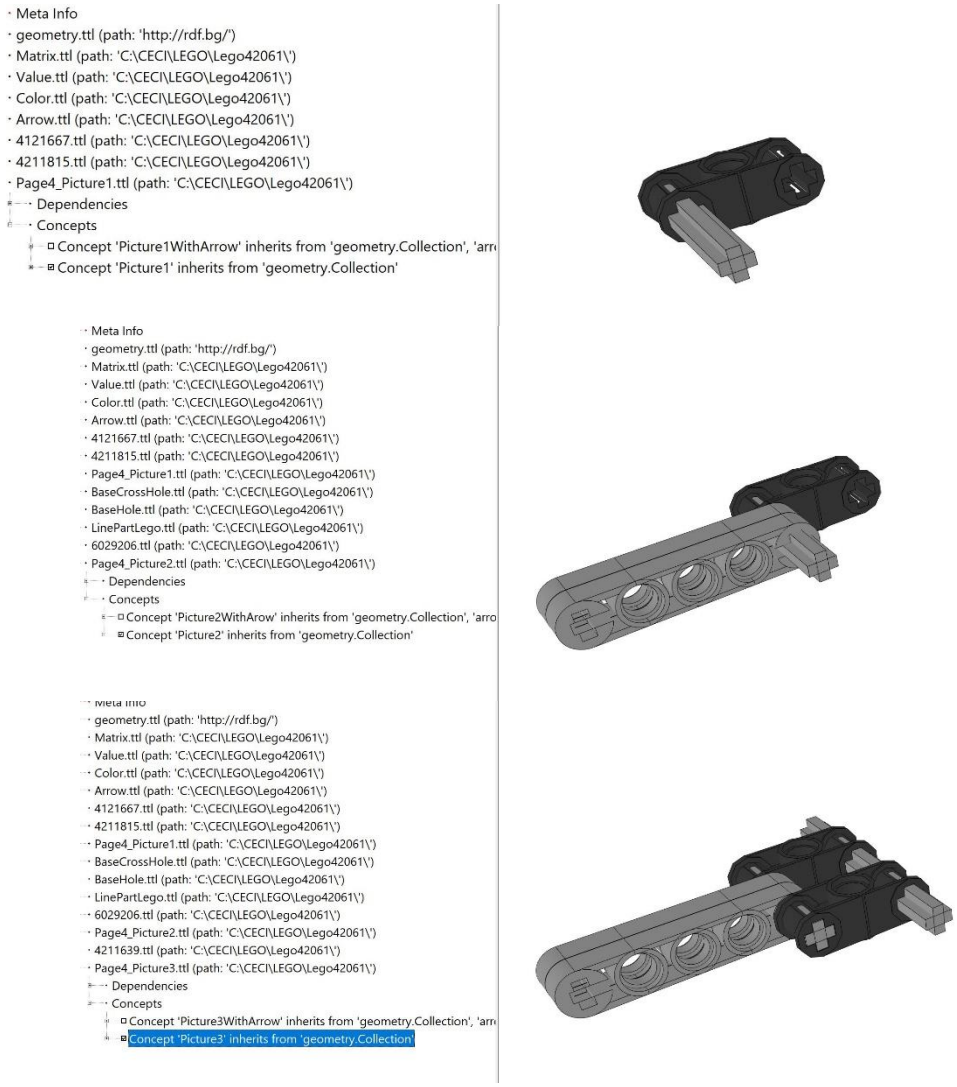


***Figure 39, how the first connected details would look in RDF ConceptBuilder. Each step is a separate file***

Sometimes the parts are able to move, this movement should be also modelled. There could be quite difficult dependencies in the synchronized movement of the separate parts. In that case the first task is to define this dependency, to isolate that part of the object, make and abstract drawing and to bring the problem to mathematical task.

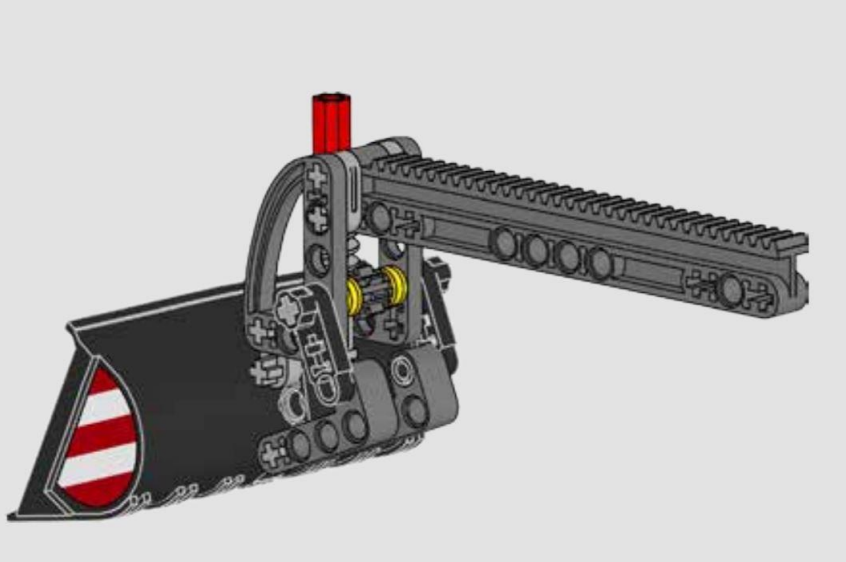

*Figure 40, a complicated movement that can't be programed directly in the model is the turning of the basket of an excavator*

Most of the sets have movements that move more than one piece. In these cases, the problem has to be reduced to a simpler problem in order to find its solution. Typically the movement of these parts is parametric, with the help of defining the appropriate parameters and the required formulas, the problem is solvable.
When we already have a solution to the abstract problem, we do the problem in the RDF ConceptBuilder to see if it works correctly. The first step is to make a separate file with all parameters and formulas. When the file is created in the notepad, we open it in the RDF ConceptBuilder and we start entering the parameters one by one, then in the rules the formulas are introduced.

```
leftAngle =  act.leftAngle  -  at.leftAngle
midleAngle = ( 90 - act.leftAngle ) - at.rightAngle
b = at.a
c = at.b
y1 = Sin ( leftAngle ) * c
x1 = Cos ( leftAngle ) * c
x2 = act.x2
y2 = act.y1
a = act.a
x = act.x
y = act.y
at.c=act.b
```

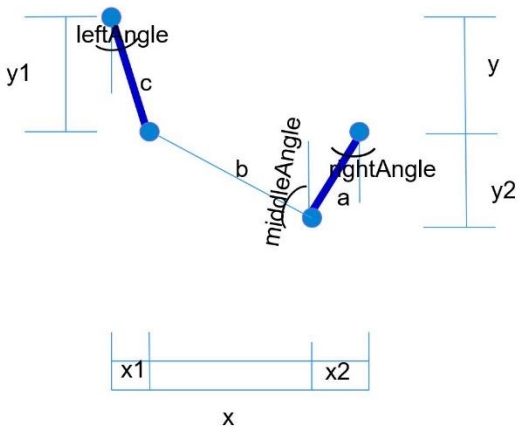


*Figure 41, The abstract representation of the task of calculating the movement of the basket of the excavator*

- AbstractComplexTriangle.ttl (path: 'C:\CECI\LEGO\Lego42061\')
  - Dependencies
  - Concepts
    - Concept 'AbstractComplexTriangle' inherits from 'geometry.Concept
      - properties
        - a (parametric driving property)
        - x2 (parametric driving property)
        - y1 (parametric driving property)
        - x (parametric driving property)
        - y (parametric driving property)
        - x1 (parametric driving property)
        - b (parametric driving property)
        - leftAngle (parametric driving property)
        - rightAngle (parametric driving property)
      - rules
        - x2 = Cos(((rightAngle * 3.14159) / 180.)) * a
        - y1 = Sin(((rightAngle * 3.14159) / 180.)) * a
        - x1 = x - x2
        - b = Sqrt((Sqr((y + y1)) + Sqr(x1)))
        - leftAngle = ArcSin(((y + y1) / b)) * (180. / 3.14159)
      - relations

*Figure 42, the file where this movement is represented mathematically with parameters*

After all the necessary files have been created using sphere and cylinders we reproduce the abstract problem, where the spheres represent the positions of the places where they are moving.

- Meta Info
- geometry.ttl (path: 'http://rdf.bg/')
- Matrix.ttl (path: 'C:\CECI\LEGO\Lego42061\lego-abstract-42061-step68\')
- Color.ttl (path: 'C:\CECI\LEGO\Lego42061\lego-abstract-42061-step68\')
- AbstractComplexTriangle.ttl (path: 'C:\CECI\LEGO\Lego42061\lego-abstract-
- AbstractTriangle.ttl (path: 'C:\CECI\LEGO\Lego42061\lego-abstract-42061-st
- AbstractShape.ttl (path: 'C:\CECI\LEGO\Lego42061\lego-abstract-42061-ste
- AbstractShape2Visualization.ttl (path: 'C:\CECI\LEGO\Lego42061\lego-abstr
  - Dependencies
  - Concepts
    - Concept 'AbstractShapeVisualization' inherits from 'geometry.Collect

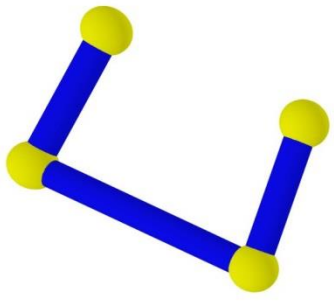

*Figure 43, the abstract representation of the task checked in RDF ConceptBuilder*

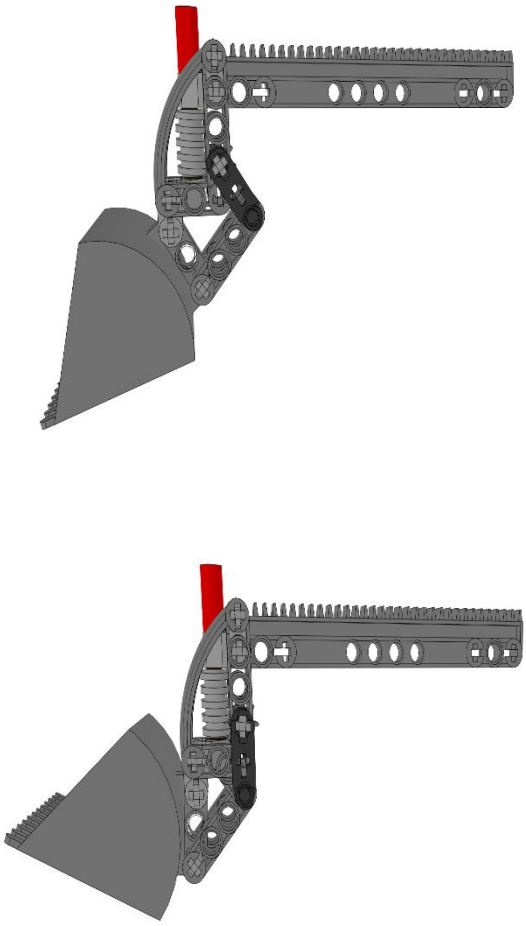

*Figure 44, when we make sure everything works correctly; we add the knowledge file to the real model and drive the necessary parts*

When every issue is solved and the concept is tested, than the final model can be assembled and all the functionality can be attached.

Any instance, concept or rule defined can be reused and / or inherited by newly created concepts. This means that although modelling is cumbersome and time consuming in the beginning, after a while reuse of generic reoccurring parts make modelling easier and faster. A small toy model like the Helicopter containing of +/- 10 steps could be created in less than 3 days from scratch when using such libraries. The time required to create PDF files as self-instruction content is expected to be considerably larger.

# 6. Conclusion

## 6.1 Usability

The software as presented in this deliverable is resulting in useful results. The results can be visualized on any computer, i.e. Windows, Linux and MacOS as well as on any modern mobile device including tablets and smartphones running Android, Windows Mobile, as well as iPhone and iPad devices. The result is clean, fast and useable offline once loaded to the device. Also hosting of the results is easy as it can be hosted on any simple server with only the most basic capabilities. This means that once running there is virtually no difference to hosting a PDF with self-instruction content, however the INSITER solution has the benefit of providing information in 3D and people can themselves walk through the process steps in both directions.

The main bottleneck of the solution is however the creation of content. Although there is tooling available as well as examples and instruction how to use these tools within this document, the real modelling of content takes a specialist. Not only creation of the content is time-consuming also the tools to generate the eventual result has its issues and works only when the models are perfectly following the rules. If there is one small mistake in the modelling it could cause the converter not to work anymore. Therefore, although the end result is of a quality level being commercially sellable the supporting tools to come to this solution are real working but prototype solutions.

In general we expect such self-instruction solutions are only relevant if the self-instruction model can be reused (as is with creation of explaining PDF documents for self-instruction). Using it for one building is simply to time consuming and cost intensive even taking into account potential improvements in the overall process of coming to an interactive 3D / 4D self-instruction model.

## 6.2 Commercial Solution

The solution raises interest from third parties, also based on the state-of-the-art research it gives a clear new innovative dimension to what large companies are providing at the moment. Based on this we are convinced further development of this tooling is relevant to support (offline usable) self-instruction manuals not only in PDF as done at the moment, but also using an interactive 3D interface with process steps.