BIM Applications of Rule-based Checking in Construction Site Layout Planning Tasks

K. Schwabe[a], M. König[a] and J. Teizer[b]

[a] Department of Civil and Environmental Engineering, Ruhr-Universität Bochum, Germany
[b] RAPIDS Laboratory, Ettlingen, Germany
E-mail: kevin.schwabe@rub.de, koenig@inf.bi.rub.de, jochen@teizer.com

Abstract

Site layout and logistics planning generally plays an important role for the successful execution of construction activities. The allocation of the right amount and size and the timely use of resources play critical roles. Compared to other industrial sectors the construction industry shows a lack of technological progress in site logistics and fabrication planning. The automotive or ship building industries, for example, have stringent production planning methods in place to the point that almost every step in the planning and manufacturing processes is supported by digital simulation and optimization. On the other hand, construction planning appears to remain a manual process slowly taking advantage of building information modeling (BIM) processes and techniques. Our work investigates automated rule-based checking in construction site layout planning tasks to simplify the existing manual processes. A gap analysis of the traditional site layout planning process is performed to identify key areas that are of high concern to practitioners. A rule-based checking algorithm for site layout planning embedded in a commercially-available BIM-platform was created and tested on specific cases in the site layout planning process of a realistic building. Promising results and a discussion to the existing limitations of rule-based checking approaches for site layout planning are presented. A short outlook towards future research gives a potential path forward in advanced construction site layout planning.

Keywords –

building information modeling; construction schedule; logistics; rule-based checking; site layout planning

1 Introduction

Before construction starts, site layout planning must be performed in order to provide the necessary equipment and temporary facilities for the construction process. This includes the allocation and dimensioning of elements like tower cranes, containers or storage areas. Decisions during this planning phase have direct impact on cost development and occupational safety on site during construction. Meeting all requirements, such as safety standards and minimizing on-site overheads, is a complex process that follows specific rules or best-practices. Experience is needed for the time-consuming evaluation and checking of all relevant rules.

One of the most frequent problems during traditional construction and planning phases of a building is loss of data. Therefore, the most ambitious goal of implementing BIM (Building Information Modeling) - processes is to provide secure and consistent data management. This is achieved by adding relevant information into a parametric 3D model. Once stored in the 3D model, the information can be retrieved at any time in the future. This similarly applies to the phase of site layout planning. The 3D environment allows an exact representation of the actual equipment geometry. This can be used to easily derive site layout plans as 2D drawings from the 3D model, which are most commonly used in the field. The exact geometry and additional semantic data can also be used to retrieve required information for the dimensioning of site layout elements, where strict code compliance is mandatory. With the help of rule-based algorithms the process of checking the site layout model for code compliance can be automated. This could minimize human error, which increases economic benefit and occupational safety.

In this research we propose a concept for BIM-based site layout planning tasks. The concept includes predefined parametric site equipment models and rule-based evaluation of site layouts. Rule-checking algorithms are prototypically implemented to evaluate our findings in a case study.

2 Related Work

The steadily progressing digitalization of construction processes covers several construction

phases, but cannot yet be described as exhaustive in every discipline. For example, for the phase of site layout planning no specialized BIM-based software is available. However, the idea of digitally assisted site layout planning is not new. Researchers around the world try to optimize the allocation of site equipment, e.g. tower cranes, in order to improve the workflow and logistics on construction sites. First approaches use optimization algorithms to find the optimal position of site equipment in a 2D site layout plan [1-3]. The next level of optimization and simulation approaches to site layout planning was to consider 3D and 4D environments [4-7]. The optimization of equipment positioning is highly attractive for BIM-based site layout planning, but basic requirements have to be met first. For example, the optimization of the crane's position requires characteristic parameters. But the basic process of site layout planning is the choice and dimensioning of equipment in order to define these parameters. Therefore, BIM-assisted dimensioning of site equipment will be the next step towards digital site layout planning. Choice and dimensioning of equipment is more than a geometric problem, as it is highly time-dependent [6]. For example, the size of storage areas is not calculated for all required material, but for a peak consumption. This peak consumption per day can be derived from the construction schedule linked with the 3D-model by dividing the volume of a construction element by the scheduled completion time. Another important parameter that can be derived at this point is the required workforce. Repeating this procedure for every construction phase takes the dynamic nature of workflows on construction sites into account. Furthermore, databases or object catalogs for parametric site equipment are proposed to simplify element selection and perform dynamic site layout planning [8].

Both optimization and dimensioning tasks are based on constraints. These constraints consist of specific rules and standards. In previous research the constraints are assumed, but their source is not defined any further. The process of defining and systematically evaluating rules is called *rule checking* or *code compliance checking* [9]. The automated process of rule-based model checking is still in early stages of research. The biggest obstacle is the transcription of existing mostly textual rules, standards and best-practices into an algorithmically readable form. The definition of such a form is called *rule language*. A consistent rule language that enables users to intuitively define and check rules in their 3D model does not yet exist. Simplified approaches to rule checking tasks are available in commercial software, e.g. Solibri Model Checker [10]. The software provides certain alterable rule sets, but the user is not able to define custom rules. This makes a comprehensive adoption of site layout planning rules impossible.

The application of rule checking in practice does not cover site layout planning tasks, but the similar topic of occupational safety can be found in recent research [11-12]. Safety-related conflicts are detected in a 3D model and proposals for possible solutions are provided. However, the calculation of possible solutions is very case-specific, but it gives an impression of what rule-checking can look like. Knowledge-based solution proposals belong to the field of artificial intelligence.

2.1 Rule checking

The process of rule checking in building models, as summarized by Eastman et al. [9], can be structured into four steps:

1. Rule interpretation
2. Building model preparation
3. Rule execution
4. Reporting of checking results

In the first step the naturally written rule texts need to be transcribed into an interpretable computer language. The form or grammar of an adequate rule language is still a matter of research, so that applications of rule languages remain to be customized. Eastman et al. [9] suggest parametric tables for rule interpretation.

The second step deals with objects and their parameters in a 3D model. The parameters in the parametric table must be equal to the parameters of the 3D objects, so that adequate filtering is guaranteed. The most common parameter type is *string*.

Step three checks the rules for compliance. Both positive (*true*) and negative (*false*) return values are added into the data structure. A false return value detects a rule conflict. The form of a conflict depends on the associated rule. Conflict information needs to be transferred to the next step.

The fourth and last step receives the conflict information from the rule execution und displays the results in two different ways: the textual report, e.g. IDs of intersecting elements, and the visual report, e.g. element selection or isolation in the model viewer.

This approach to rule checking is a subsequent procedure. This means, that the rule check takes place after design decisions are made. The afore mentioned commercial software Solibri Model Checker uses this principle to check 3D models for design quality. Predefined rules include element specific clash detection or element parameter comparison. The editing of these predefined rule sets is possible, but very limited. New rules cannot be defined.

3 Methods

Our general concept for BIM-supported site layout planning is depicted in Figure 1. Two of the four requirements are the building model and the construction schedule. The other two requirements have yet to be developed. They contain a database or so-called object catalog of parametric models of all existing site layout equipment and a comprehensive database of rules concerning site layout planning. These rules must be in form of an algorithmically readable rule language that enables the transcription of all existing rules, standards and best-practices. Company-specific expert knowledge or best-practices that have not yet been written down formally, can and have to be included as well. With this information the automated rule checking can support the dimensioning and placement of site layout elements. The 2D site layout plans that are needed in the field can be derived, costs including site equipment can be calculated and logistics can be planned.

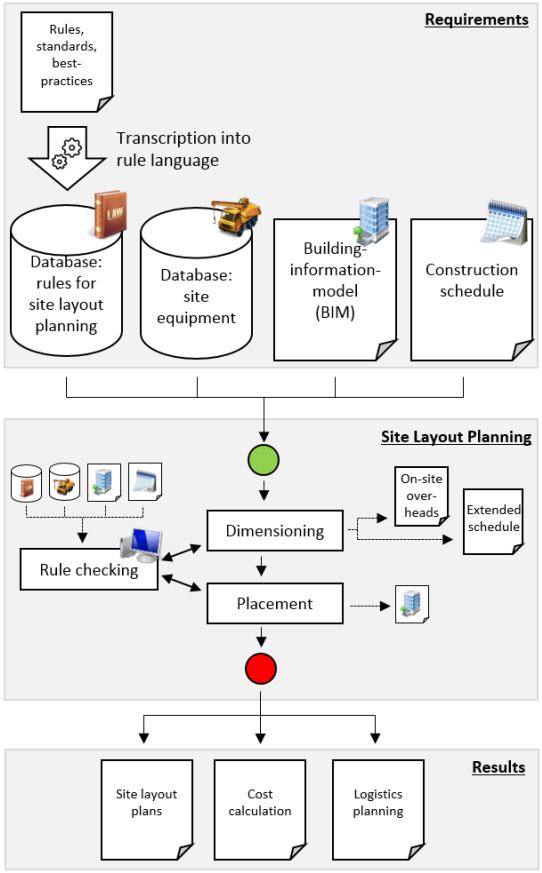

Figure 1. Concept of rule checking application

3.1 Interactive rule checking

An extended use of rule checking that is still in early stages of research describes an interactive process, where design decisions are supported by preprocessed rules. This means, that the supporting software visualizes rule-based constraints in advance. Figure 2 shows a conceptual example of interactive preprocessed rule checking. The light green area around the building indicates where the tower crane can be placed. Rules that would prohibit the placement are considered, for instance the installation of a site fence and its safe distances towards the surrounding terrain or inclined surfaces.

Although we consider the application of interactive preprocessed rule checking to be very attractive for design supporting software, the first step towards a rule-based evaluation of site layout plans is the implementation of subsequent rule checking. The problem with currently available rule checking software is the limited ability to edit the predefined rules. The solution is the development of a rule language, which can be operated intuitively by the user to create custom rules. This avoids the exclusion of the user from the rule algorithm and leads to a better understanding of the consequences of the self-made rule. Furthermore, the user is able to reedit the custom rule, if the result does not satisfy previous expectations.

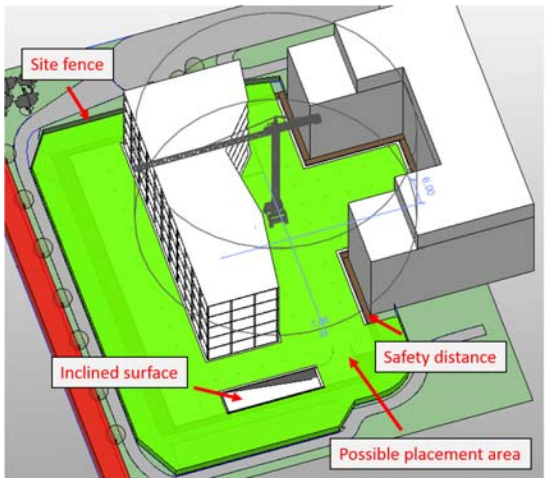

Figure 2. Interactive rule checking: possible placement area of a tower crane

3.2 Parametric site equipment

A required step of rule checking is building model preparation. This includes both the building model and the parametric 3D objects of site equipment. A database of predefined site equipment elements needs to be

developed, serving as an object catalog during site layout planning to simplify model creation significantly. Parameters added to the 3D objects are required for adequate filtering and rule execution, e.g. crane radius or maximum load capacity. The digital site equipment catalog should include the following element categories:

- Large equipment (construction machinery, cranes, etc.);
- Social services and office equipment (containers, sanitary facilities, etc.);
- Traffic areas and transportation routes (construction road, storage areas, etc.);
- Supply and disposal (electricity, garbage containers, etc.);
- Construction site safety (site fence, scaffolding, etc.);
- Temporary pit system (excavation support, slope, etc.).

The commercial 3D modelling software Autodesk Revit [13] provides a small portion of these elements as prototype objects. So-called Revit-families of a tower crane, concrete pump vehicles, containers and a material storage area can be seen in Figure 3.

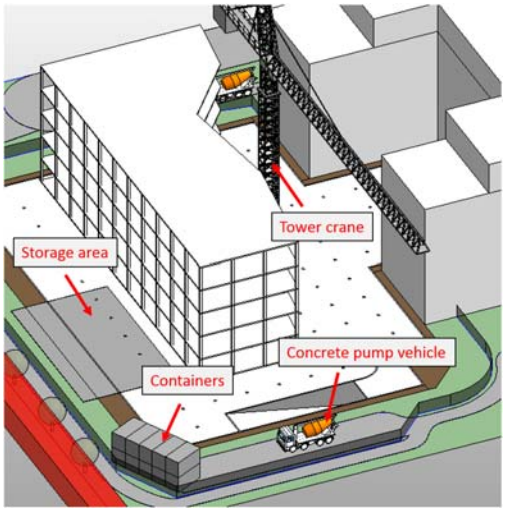

Figure 3. Examples of 3D site equipment in Autodesk Revit

4 Case Study

For the prototypical implementation of rule checking we chose the software Autodesk Revit. The 3D modelling software provides design of parametric objects and an API (application programming interface) that enables custom programming of so-called Add-Ins. Add-Ins can be designed for additional and custom interaction with the objects in the 3D model to extend the original software functionality. An evaluation or testing of similar modelling software for this particular purpose was not performed. The case study includes a realistic construction project of a five story office building with surrounding terrain and urban cityscape (see Figure 3). The implementation follows the four steps of rule checking mentioned earlier.

4.1 Rule interpretation

At first we defined a tabular structure for the transcription of the rules, as recommended by Eastman et al. [9]. The rule table consists of two different input types. The first type is the rule category. Every category invokes a specified algorithm. We defined three prototype categories for the implementation:

- *Radius rule:* Checks, whether an element cuts the pan radius of another element
- *Reach rule:* Checks, whether a set of elements can be reached by other elements
- *Lift rule:* Checks, whether the heaviest element of a selected type can be lifted by another element

The second input type are rule parameters. The amount and value of rule parameters depend on the associated rule category. For example, the category *Reach rule* requires information about which elements it affects. Both element types then serve as input parameters. Figure 4 shows the resulting rule table as a CSV-file. The first column always contains the category, so that the program can identify which algorithm to invoke. After identifying the category, the input parameters found in the subsequent columns are transferred to the invoked algorithm.

Radius rule	Tower_Crane	Container	False
Reach rule	Tower_Crane	Building	
Lift rule	Tower_Crane	Column	
Reach rule	Tower_Crane	Storage_area	

Category Parameter 1 Parameter 2 ... Parameter n

Figure 4. Prototypical rules transcribed into a rule table

The rule in line one transcribes the example rule text '*Containers* may *not* be placed within the load pan *radius* of *tower cranes*'. The keywords in italics signalize the input types for transcription, e.g. *radius* for Radius rule. The second rule represents another example rule.

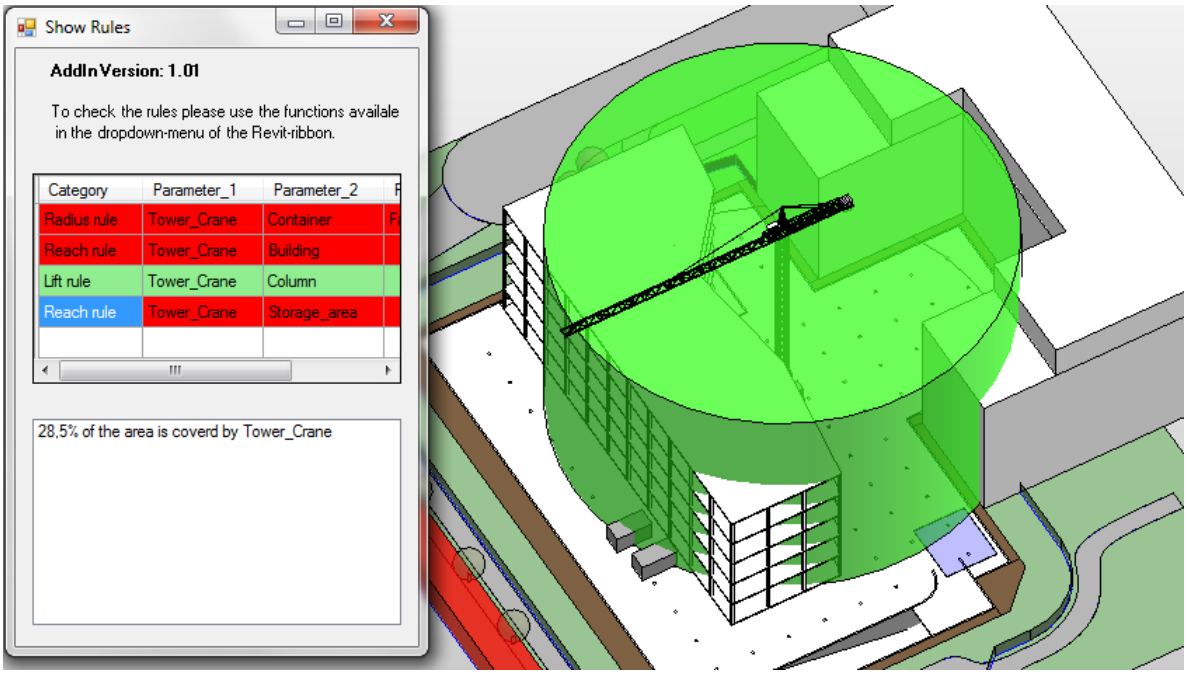

Figure 5. Rule execution: conflict appearance

'All building elements must be within reach of tower cranes'. Line three applies the example rule text 'Tower cranes must be able to lift the heaviest precast column'. In the fourth line a second reach rule is invoked with the example rule text 'All material storage areas must be within reach of tower cranes'. These example rules are derived from a German handbook for site layout planning [14]. Further rules will be implemented. This procedure can be repeated until every desired rule is categorized. If the user wants to include another rule and it can be assigned to one of the existing rule categories, another line must be added into the CSV-file. But if the user wants to define a new category new source code has to be written in order to include the new rule-based algorithm.

4.2 Model preparation

As seen in Figure 4 the rule parameters are in string format. For adequate filtering the exact string must be found among the parameters of the desired elements. For example, the keyword *Tower_crane* can be assigned to the name of the element. The keyword *Building* requires a different approach. Either the algorithm is taught which elements belong to a building, or the user assigns the string 'Building' to a custom parameter of the necessary elements manually. In order to check whether all necessary parameters have reasonable values a so-called pre-checking process must be carried out before the actual rule execution starts. However, pre-checking is not part of this research.

4.3 Rule execution & reporting of checking results

For the process of rule execution the three example rule categories mentioned above are implemented using the API of Autodesk Revit. At first the rule table must be read and saved into a data model. The data model mainly consists of two classes: *Rule* and *Conflict*, as seen in Figure 6. Other classes consider the user interface or event handling and play no role in the given data model.

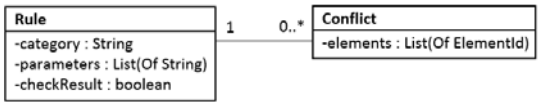

Figure 6. Classes in the data model

A rule can cause zero or an infinite amount of conflicts. But one conflict can only be associated with a specific rule. During the reading of the rule table the rule objects are created and the fields *category* and *parameter* are filled with values. During rule execution conflict objects will be created, if elements do not apply to the rule and added to the list of conflicts within the

rule object. In addition the field *checkResult* will be set to *false*, if at least one conflict occurs. If no conflict is found the algorithm will return *true*.

The results of the first rule check can be seen in Figure 5. The 3D building and site layout model are depicted on the right, the Add-In window on the left. The site layout objects include two containers (grey cuboids at the bottom), a storage area (blue rectangle on the right) and a tower crane, whose radius is represented by a semitransparent green cylinder. The rule table introduced in Figure 4 can be found within the Add-In window.

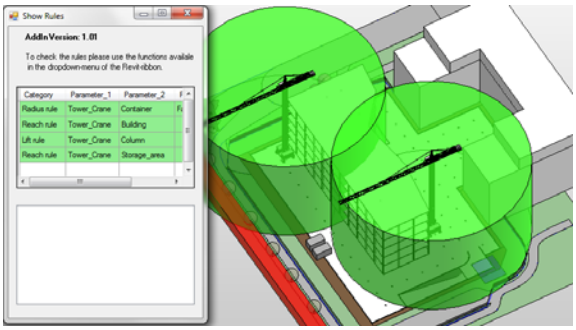

Figure 7. Rule execution: conflicts solved by layout changes

Below a text field reports potential conflicts. The first rule check marked three of the four rules as false (red rows) and one rule as true (green row). Clicking a red row adds the textual report of a conflict to the text field. The textual report of the selected Reach rule states, "28.5% of the area is covered by Tower_Crane". It returned false, because the rule dictates 100% storage area coverage. A click on the conflict text selects all objects involved in the conflict, in this case the tower crane. The Radius rule returned false, because both containers are within the load pan radius of the tower crane. The upper Reach rule returns false, because the tower crane does not cover 100% of the building. The next step is eliminating the conflicts. Therefore, a second tower crane is added to the layout model and the objects are rearranged. The conflict solution can be seen in Figure 7. Now the two tower cranes reach 100% of the building and storage area. The Lift rule still is marked as true and the containers are moved out of the load pan radius of the tower cranes.

The scenario described above illustrates our prototype implementation of subsequent rule checking. Although the displayed checking results do not report any conflict, not all imaginable rules have been considered in this case: for example, that both cranes pan at the same height or that the crane pan radius is not affected by the surrounding objects. This indicates how important a comprehensive rule data base is, so that even rules that can easily be missed will be considered. Another aspect that becomes obvious at this point is that a conflict cannot always be solved by relocating an object. In the case of the crane its load pan radius must be edited in the model and pan restrictions must be given to the crane operator.

4.4 Implementation

The Revit API provides the ability to automate features the user would otherwise carry out manually. Therefore the automation of rule checking in Revit is limited to the features provided by the API. One of those features for the analysis of object geometry is called 'SolidSolidCutUtils'. It allows intersections between two solid geometries and returns useful information, especially volume differences before and after the cut. This makes it easier to observe and evaluate object interactions. Otherwise the algorithm would have to use bounding boxes. A bounding box determines a cubic space that includes every geometrical point of an object. The problem with bounding boxes is that they remain cubic, even if the object does not. In Figure 6 this problem is visualized. Both spheres obviously do not intersect, but their bounding boxes do. Thus, the calculation of intersections via bounding boxes cannot guarantee complete accuracy. This problem can be avoided, by working with CSG (constructive solid geometry) - objects. CSG describes a method that uses boolean operators to interact with predefined mathematical geometries, e.g. cylinder, sphere etc. With this method the calculation of intersections is mathematically precise and time efficient.

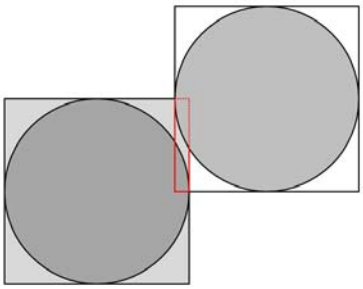

Figure 8. Intersections of objects and their bounding boxes

An obstacle of using 'SolidSolidCutUtils' is that it cannot be used for all family types. Standard family types, e.g. floors, are restricted by Revit. The solution to bypass these restrictions is to duplicate the element geometry. Each solid in the geometry needs to be read

and rebuild. However, this procedure is not practicable in every case, so that each case must be handled individually.

The pseudo code of the algorithm implemented for the radius rule can be seen in Figure 7. First all elements in the BIM-model are filtered by the two element types received from the input parameters. Both element lists then get iterated over. If an intersection between the two elements is found they are added to a new element list that functions as the input parameter for the newly created conflict object.

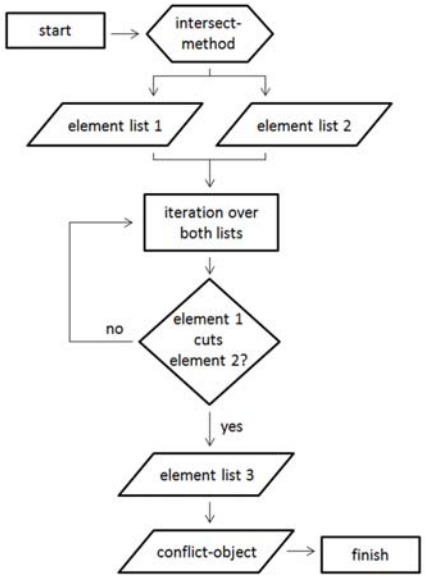

Figure 9. Schematic pseudo-code of the rule algorithm involving intersections

4.5 Dimensioning of site layout elements

The process of dimensioning site layout elements mostly includes time-dependent and logistic rules that can be derived from the construction schedule by calculating peak consumptions [6]. Figure 10 shows the schematic calculation of the size of a storage area for masonry in a simple example. The required information is derived from different sources. The first information is derived from the schedule. The completion time of the masonry works in this case is 2 weeks. The volume of masonry works is derived from the BIM-model: 80 m³. This can be calculated to a peak consumption of 40 m³/week. The best-practice for planning temporary storage of masonry material, found in the aforementioned German handbook for site layout planning [14], recommends a size of 2 m^3 per square meter of masonry work during a given week. Considering the masonry work tasks altogether and leveled by the available resources, the calculated storage area for masonry in the given project is 20 m². This information is used by the site layout planner to define a rectangular storage area with dimensions 5 x 4 m in the model. The use of peak consumptions for dimensioning of storage areas prevents the waste of useful space on the construction site. Time-dependent rules like these have not yet been implemented for this research.

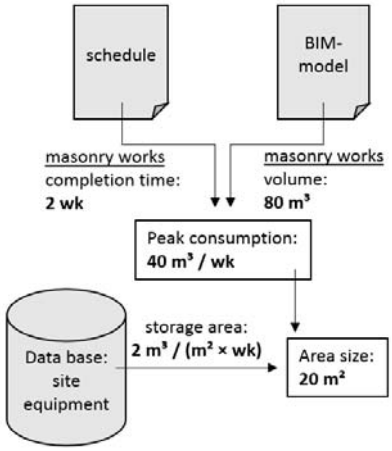

Figure 10. Schematic calculation of storage area size using peak consumption

5 Discussion

The implemented rules presented in this research demonstrate only a small number of prototype rules and raise no claim to be exhaustive. It is questionable, if more diverse and complex rules can be managed by a tabular approach to rule transcription. In addition, the exclusion of the user from the implemented rule algorithm further questions the use of rule tables. Nevertheless, it shows a realistic first application of subsequent rule checking in site layout planning tasks. The concept of interactive preprocessed rule checking introduced in this paper has not yet been tested, but should be addressed in future work. In order to calculate a proposed design solution the program requires a comprehensive database of rules. Another database including site layout equipment and machinery needs to be developed, so that choice and placement of elements is massively simplified, due to the connection of 3D and characteristic parameters.

The choice of the commercial software Autodesk Revit for implementation was arbitrary and served only for testing purposes. Although Revit provides some of the necessary requirements for the implementation, restricted or limited functions of the API leave implementing to be counterintuitive. Another important

issue is the dependency on a single commercial software, which prevents the application from contributing to open-BIM. This could be solved by developing an external software dealing solely with rule checking and design optimization in site layout planning tasks using independent data exchange formats, e.g. IFC (Industry Foundation Classes) [15]. Early examples that require some human input and expert knowledge are shown in [16, 17].

6 Conclusion

The concepts presented in this research demonstrate that BIM- and rule based site layout planning is highly attractive for the engineers involved in this process. The most significant advantages are the intuitive 3D environment of the building model, where 2D drawings can easily be derived from. The permanent availability of additional relevant information enables integrated site layout scheduling (4D) and cost calculation (5D). Joining the relevant information leads to a better data management and prevents loss of data before and during construction. In addition, the development of automated rule checking could result in minimized human error during planning phase, which increases monetary benefit and occupational safety.

Future work should consider time-dependent rules, which are necessary for the dimensioning of site layout elements. Furthermore, the development of a designated rule language, that is able to comprehensively cover the diversity of existing rules and best-practices, should be addressed. If this is achieved, a rich rule data base and the corresponding object catalog of site equipment can be generated. In the process of implementing a rule checking application the idea of open-BIM using independent data exchange formats should be favored.

References

[1] Ning, X., Lam, K., and Lam, M., A decision-making system for construction site layout planning, *Automation in Construction*, 20, 459-473, 2011.

[2] Pradhananga, N. and Teizer, J., Congestion Analysis for Construction Site Layout Planning using Real-time Data and Cell-based Simulation Model, *Computing in Civil and Building Engineering*, 681-688, 2014.

[3] Yahya, M. and Saka, M., Construction site layout planning using multi-objective artificial bee colony algorithm with Levy flights, *Automation in Construction*, 38, 14-29, 2014.

[4] Andayesh, M. and Sadeghpour, F., The time dimension in site layout planning, *Automation in Construction*, 44, 129-139, 2014.

[5] Astour, H. and Franz, V., BIM-and Simulation-based Site Layout Planning, *Computing in Civil and Building Engineering*, 291-298, 2014.

[6] Cheng, J. and Kumar, S., A BIM Based Construction Site Layout Planning Framework Considering Actual Travel Paths, *The 31st International Symposium on Automation and Robotics in Construction and Mining*, 2014.

[7] Olearczyk, J., Lei, Z., Ofrim, B., Han, S., and Al-Hussein M., Intelligent Crane Management Algorithm for Construction Operation (iCrane), *The 32nd International Symposium on Automation and Robotics in Construction and Mining*, 2015.

[8] Hollermann, S. and Bargstädt, H., 4D Site Installation Planning in Virtual Reality for Multi-user, *Computing in Civil and Building Engineering*, 777-784, 2014.

[9] Eastman, C., Lee, J., Jeong, Y., and Lee, J., Automatic rule-based checking of building designs, *Automation in Construction*, 18, 1011-1033, 2009.

[10] Solibri Model Checker, On-line: http://www.solibri.com/products/solibri-model-checker/, Accessed: 11/08/2015.

[11] Kim, K. and Teizer, J., Automatic design and planning of scaffolding systems using building information modeling, *Advanced Engineering Informatics*, 28, 66-80, 2014.

[12] Zhang, S., Sulankivi, K., Kiviniemi, M., Romo, I., Eastman, C., and Teizer. J., BIM-based fall hazard identification and prevention in construction safety planning, *Safety Science*, 72, 31-45, 2015.

[13] Autodesk Revit, On-line: http://www.autodesk.com/products/revit-family/overview, Accessed: 12/06/2015.

[14] Schach, R., Otto, J., *Baustelleneinrichtung*, Vieweg+Teubner Verlag, Wiesbaden, Germany, 2011

[15] buildingSMART, On-line: http://www.buildingsmart-tech.org/ifc/IFC4/final/html/, Accessed: 15/07/2015.

[16] Schwabe, K., Liedtke, S., König, M., and Teizer, J., BIM-based Construction Site Layout Planning and Scheduling, *International Conference on Computing in Civil and Building Engineering (ICCCBE)*, Osaka, Japan, 2016.

[17] Golovina, O., Teizer, J., Pradhananga, N. Heat map generation for predictive safety planning: Preventing struck-by and near miss interactions between workers-on-foot and construction equipment, *Automation in Construction*, http://dx.doi.org/10.1016/j.autcon.2016.03.008.