Automated Generation of Construction Sequences using Building Information Models

E. Tauscher[1], K. Smarsly[1], M. König[2] and K. Beucke[3]

[1]Bauhaus University Weimar, Computing in Civil Engineering, Coudraystraße 7, 99423 Weimar, Germany; PH: +49 3643 584215; FAX: +49 3643 584216; e-mail: {eike.tauscher | kay.smarsly}@uni-weimar.de
[2]Ruhr University Bochum, Computing in Engineering, Building IC, 44781 Bochum, Germany; PH +49 234 3223047; FAX +49 234 3214292; e-mail: koenig@inf.bi.rub.de
[3]Bauhaus University Weimar, Geschwister-Scholl-Straße 8, 99423 Weimar, Germany; PH: +49 3643 581111; FAX: +49 3643 581120; e-mail: karl.beucke@uni-weimar.de

ABSTRACT

Construction scheduling is usually limited to the documentation of one final construction schedule, and construction alternatives that might have been considered during the planning process are usually not included in the final schedule. Moreover, a formal control of construction schedules in terms of completeness and correctness is very limited, because existing methods, such as 4D visualization, are insufficiently integrated into the construction planning process. This paper addresses the development of a software framework that has been designed to support the process of construction scheduling. As will be shown in this paper, the process of construction scheduling, which is usually carried out manually, is conducted automatically to a large extent using building information models (BIM) and 4D visualization. Due to the logical interconnections of BIM objects and construction tasks, flexible schedules and visual representations of construction processes are automatically generated without permanent human interaction.

INTRODUCTION

In most construction projects, construction schedules are generated manually using state-of-the-art software tools. Unfortunately, the generated schedules are typically subject to the personal experience and the subjective knowledge of the project manager responsible for scheduling. Furthermore, construction scheduling is usually restricted to the documentation of the final result, i.e. only one final construction schedule is available, neglecting potential changes that might occur during construction. Construction alternatives that might have been considered during the construction planning process are typically not included in the final schedule. A revision of these schedules in case of changes occurring during construction is usually not done, because revisions are extremely time-consuming and expensive. Also, a formal control of construction schedules in terms of completeness is very limited since available methods, such as 4D visualizations, are insufficiently

integrated in the schedule planning process. As a consequence, the reuse of existing construction schedules is almost impossible.

This paper presents an integrated software framework that actively supports the process of construction scheduling and facilitates the automated generation of construction schedules. The work shown in this paper is part of a collaborative research project that has been presented by König *et al.* (2006), Tauscher *et al.* (2009), Mikulakova *et al.* (2010), and Hartmann *et al.* (2012). In essence, the software framework provides a model for handling construction schedules both in the planning and in the execution process. Due to the automated generation of construction schedules and the integration of 4D visualizations, a high degree of reusability is achieved, entailing precise and flexible construction schedules even if changes in construction planning occur. The approach proposed herein uses building information created by CAD planning tools based on Industry Foundation Classes (IFC). The software framework is based on the IFC Tools project (Tauscher and Theiler, 2014).

The paper is organized as follows. First, the basic concept of automatically generating construction schedules is briefly explained. Then, the utilization and the operation of the software framework are shown by the example of a research and office building located in Weimar, Germany. The paper concludes with a concise summary and a brief outlook on potential future research directions.

GENERATING CONSTRUCTION SEQUENCES

The first step when modeling a construction sequence is to define the granularity for the underlying model. In this study, it is assumed that one construction task results in the construction of exactly one building element. If there are several construction tasks required to construct a building element, the corresponding construction sequences can be substituted by a single construction task, as shown in Figure 1.

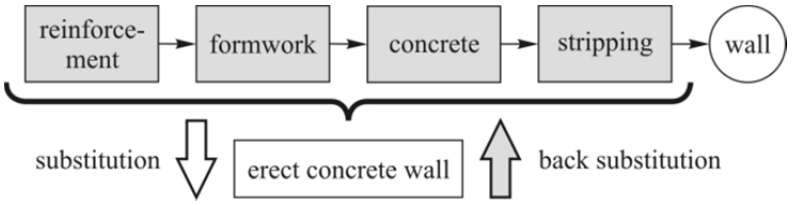

Figure 1. Example substitution of construction sequences.

A resulting building element is computationally represented by its object characteristics. Primary object characteristics, which are relevant to defining a task for constructing a building element, are type (e.g. wall), material (e.g. concrete), and dimension. Based on the above assumptions, two aspects are deduced that are crucial for the proposed generation concept, (i) the transferability of construction tasks and (ii) the sequence of construction tasks.

Transferability of construction tasks

Each construction task results in exactly one building element. To define an appropriate model, the characteristics of building elements or, more precisely, their descriptions are compared with each other by the software framework. Once a construction task has been defined for a certain building element, this task can be transferred to similar building elements (Figure 2).

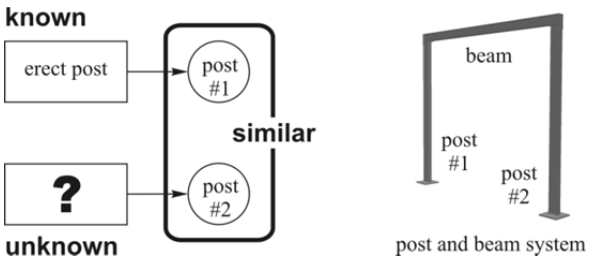

Figure 2. Example of transferability of construction tasks.

Sequence of construction tasks. A task within a construction sequence can be executed only if all relevant preceding tasks have previously been executed. In other words, the accomplishment of one or more construction tasks is a prerequisite of the succeeding task(s). As a consequence, given that a task results in a building element, one or more building elements are preconditions of the construction task to be executed. In the software framework, the sequence of construction tasks is achieved as a result of a concatenation of preconditions with identical resulting building elements, as shown in Figure 3.

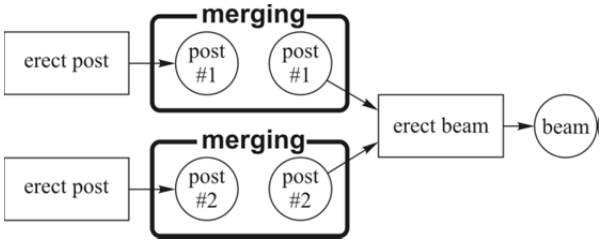

Figure 3. Example of sequencing construction tasks.

The construction tasks, modeled in accordance with the aforementioned assumptions, are stored in a database. Based on the pool of construction tasks stored in the database, an automated assignment of construction tasks to building elements, which are part of the underlying building information model, as well as an accurate determination of the task order, is achieved.

UTILIZATION AND OPERATION OF THE SOFTWARE FRAMEWORK

The utilization and operation of the software framework is elucidated by means of the "CIB.Weimar" building, a 4-story research and office building located in Weimar (Germany), which serves as a reference building in this study (Figure 4). For the sake of clarity, only the third floor of the 4-story building is used in the following subsections to illustrate the efficient utilization and the operation of the framework.

Figure 4. "CIB.Weimar" reference building (CAT, 2013).

In this study, the process of generating construction schedules is divided into three main phases, as illustrated in Figure 5. These phases are the *structuring phase*, the *generation phase*, and the *linking phase*, all of which being supported by the software framework.

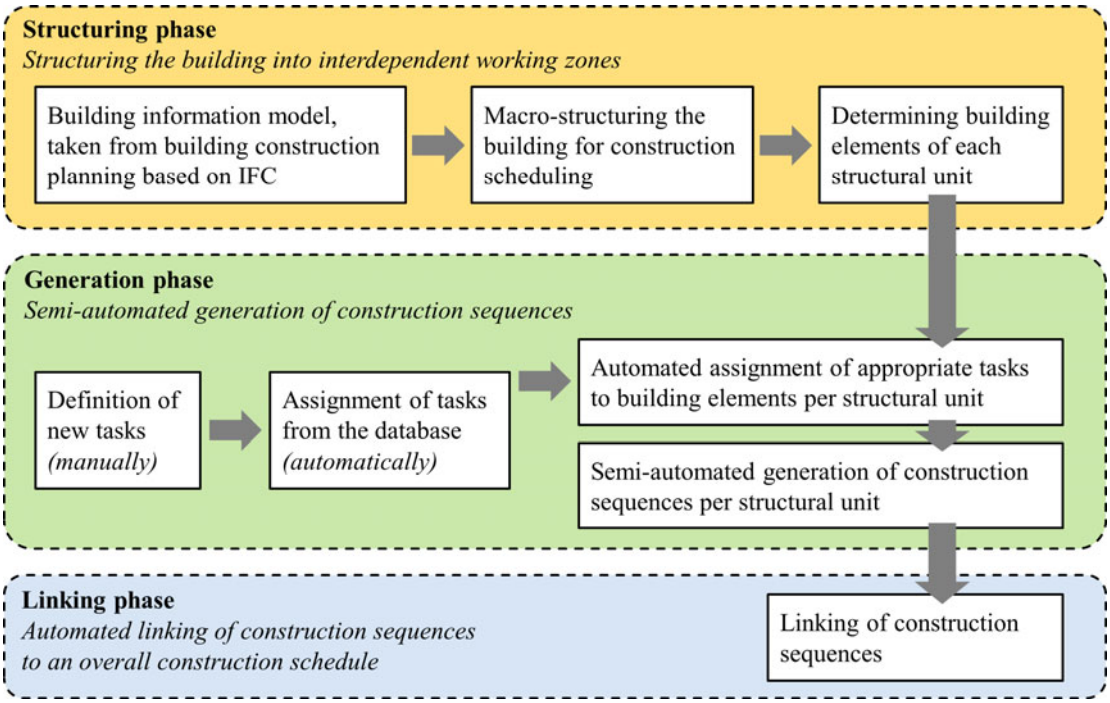

Figure 5. The process of generating construction schedules.

Structuring phase

In the structuring phase, the building information model used is structured into coherent units, referred to as "structural units". The main objective of the structuring phase is to automatically extract the building elements contained in each structural unit. For that purpose, the user defines geometric zones of the building and specifies queries on the building information model (Tulke, 2010; Tauscher, 2011). As an outcome of the structuring phase, the building to be constructed is divided into construction sections, and the building elements are precisely determined. It should be mentioned that the execution order of structural unit is defined manually because it is not possible to determine the correct order of all construction sequences automatically. Finally, the order of structural units is represented in a directed "structure graph" (Figure 6).

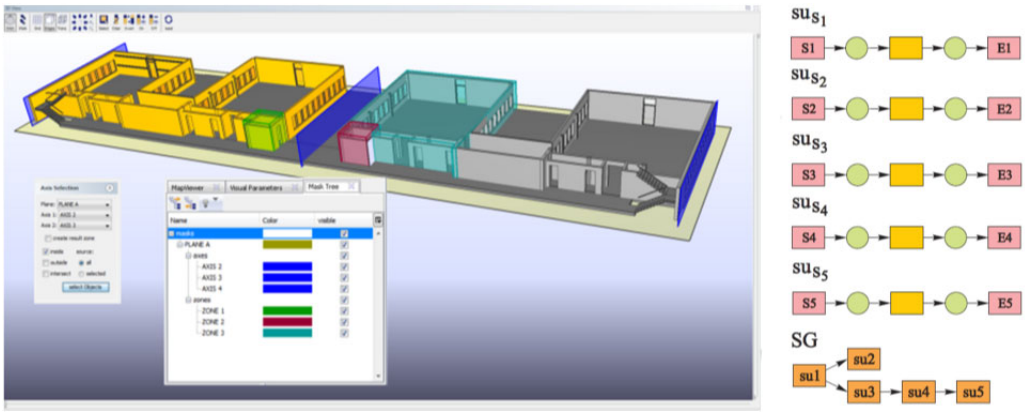

Figure 6. Visualization of the 3rd floor of the "CIB.Weimar" including geometric zones (left) and structural units, SU_i, as well as structure graph, SG (right).

Generation phase

The purpose of the second phase, the generation phase, is to automatically generate construction tasks for the building elements of the structural units as identified in the structuring phase. In addition to the automated generation provided by the software framework, manual user interaction is supported to achieve optimum results. The generation phase is subdivided into three further phases, (i) the mapping phase, (ii) the concatenation phase, and (iii) the conflict resolution phase, where the mapping phase includes two steps.

Mapping phase – step 1. Step 1 is conducted to find existing tasks stored in the database, i.e. in the building information model, that can be used for creating the building elements contained in the structural units. Technically, each building element of a structural unit is compared to building element descriptions stored in the database. Once a building element description fits to that of a building element of the considered structural unit, the corresponding tasks are mapped to the building element of the building information model.

Mapping phase – step 2. Step 2 compares the prerequisites of tasks mapped in step 1. If prerequisites are similar, the geometric adjacencies to these tasks' building elements are evaluated. If the minimal distance of the similar building elements is within a pre-defined limit, the determined building element is mapped as a prerequisite to the considered task. Exemplarily showing a rounded wall, which requires two columns as prerequisites. Figure 7 illustrates the principle of geometric adjacency. Based on the geometric distance to the wall, column "S1" and column "S2" are assigned prerequisites for the wall.

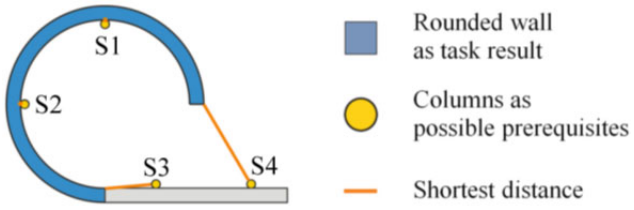

Figure 7. Principle of geometric adjacency employed for assigning prerequisites.

Concatenation phase. While in the previous phase appropriate prerequisites and resulting building elements have been mapped to each task, in the concatenation phase workflow sequences for the structural units are created. Identical prerequisites and resulting building elements of the tasks are concatenated, and start as well as end nodes are added, as shown in the workflow graph in the right hand side of Figure 8.

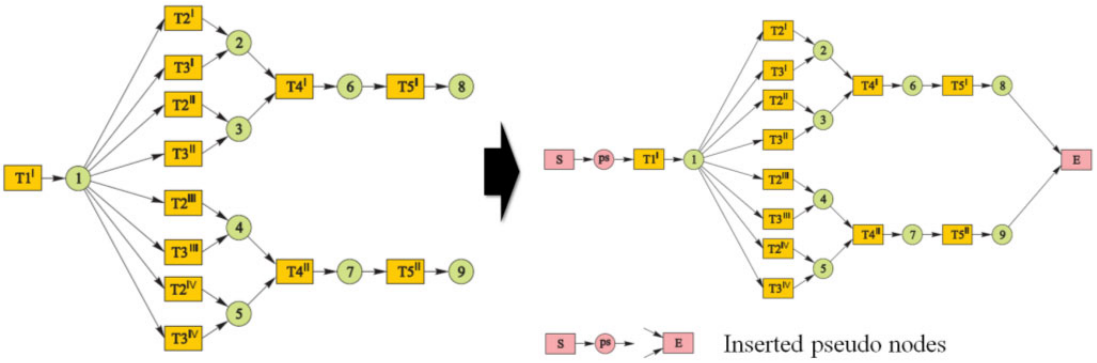

Figure 8. Workflow sequences after concatenation (left) and after inserting start and end nodes (right).

Conflict resolution phase. The workflow graphs resulting from the previous phase may potentially be inconsistent, because the algorithms for the graph generation may cause incorrect synchronization of XOR-splits (e.g. node 1 in Figure 8) and AND-joins (e.g. node T4' in Figure 8), which is a well-known problem (Tauscher, 2011; Pahl and Damrath, 2001). For this class of failure, an automated conflict-solving algorithm is integrated into the software framework that, in this example, revises the XOR-split by the insertion of pseudo nodes resulting in a correct connection of variants and parallel construction processes (Figure 9).

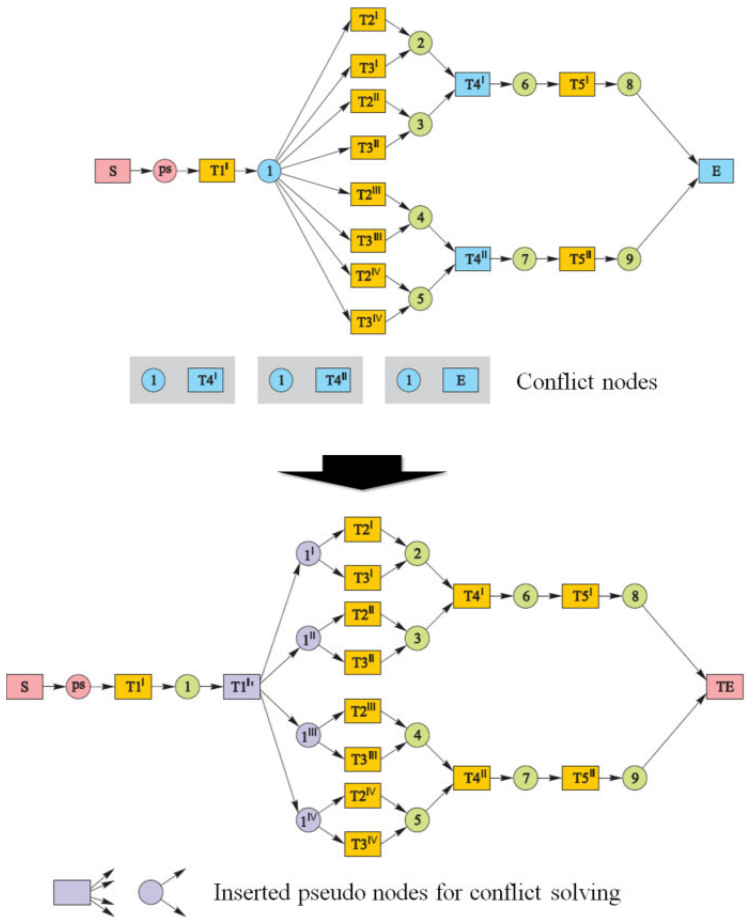

Figure 9. Automated conflict solving.

Linking phase. Upon executing all previous phases, in the linking phase valid workflow graphs are generated for each structural unit, as defined in the structuring phase. As shown in Figure 10, the workflow graphs for the structural units are linked according to the previously defined by the structure graph SG in order to produce an overall workflow graph as the final result.

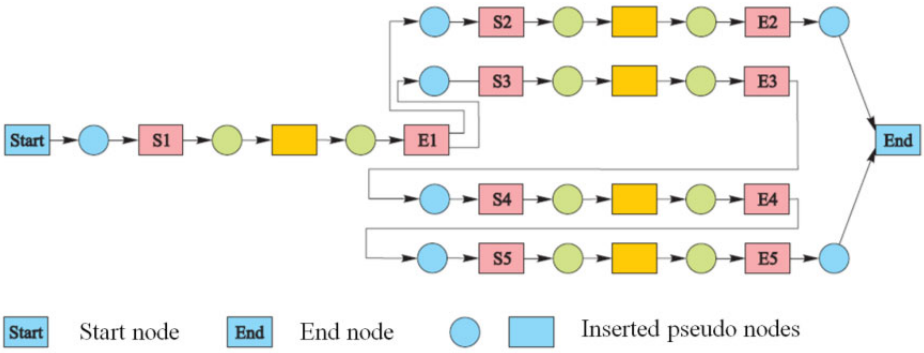

Figure 10. Final workflow graph.

SUMMARY AND CONCLUSIONS

This study has demonstrated that construction scheduling processes can reliably be automated to a large degree. Using the software framework presented in this paper, a significant reduction of time and cost has been achieved, that would otherwise be caused by manually generating and tracking construction schedules. Specifically, flexible construction schedules are stored in a database and provided to the project manager in charge. The schedules, once being stored in the database, are available for the automated generation of construction schedules in future projects. Furthermore, the integration of 4D visualizations into the planning process offers a new quality of construction scheduling as compared with state-of-the-art software tools commonly deployed because missing or inaccurate tasks are identified and eliminated early. In future research, the presented geometric adjacency in the mapping phase may be refined by modeling step 2 more detailed; this would allow further reducing mapping failures that still occur to a certain extent in the current version of the software framework. In addition, further classifications of construction tasks that are not yet supported, e.g. the creation or demolition of building elements, are envisaged in future research efforts.

REFERENCES

Chamber of Architects of Thuringia (2013). CIB.Weimar – Centrum für Intelligentes Bauen. [online] Available at: <http://www.architekten-thueringen.de> [Accessed 20/12/2013].

Hartmann, V., Beucke, K., Shapir, K., König, M. (2012) “Model-based Scheduling for Construction Planning”. In: Proceedings of the 14th International Conference ICCCBE-XIV, Moscow, Russia: MSU

König, M., Beucke, K. and Tauscher, E. (2006). “Mangement and Evaluation of Alternative Construction Tasks”. In: Proceedings of the Eleventh International Conference ICCCBE-XI, Montreal, Canada: Université du Québec.

Pahl, P. J. and Damrath, R. (2001). “Mathematical Foundations of Computational Engineering”, Heidelberg, Germany: Springer.

Mikulakova, E., König, M., Tauscher, E. and Beucke, K. (2010). “Knowledge-based schedule generation and evaluation”. Advanced Engineering Informatics 24(2010) pp. 389-403.

Tauscher, E. and Theiler, M. (2014). IFC TOOLS PROJECT - IFC Java Framework, http://www.ifctoolsproject.html (Feb. 18, 2014).

Tauscher, E., Mikulakova E., König, M. and Beucke, K. (2009). “Automated Generation of Construction Schedules based on the IFC Object Model“. In: Proceedings of the 2009 ASCE IWCCE, Austin, TX, USA.

Tauscher, E. (2011). “Vom Bauwerksinformationsmodell zur Terminplanung -Ein Modell zur Generierung von Bauablaufplänen”. PhD Dissertation. Weimar, Germany: Bauhaus University Weimar.

Tulke, J. (2010). “Kollaborative Terminplanung auf Basis von Bauwerks-Informationsmodellen”. PhD Dissertation. Weimar, Germany: Bauhaus University Weimar.