AUTOMATED DETAILING OF 4D SCHEDULES

A dissertation

in partial fulfillment of the requirements

for the degree of

Doktor-Ingenieur (Dr.-Ing.)

submitted to the Faculty of Civil Engineering

of

Bauhaus-Universität Weimar

by

M.Eng. Dang, Thi Trang

Examiner:

1. Prof. Dr.-Ing. Hans-Joachim Bargstädt
2. Prof. Dr.-Ing. Kay Smarsly
3. Prof. Dr. Geoffrey Shen

Weimar, April 2014

ACKNOWLEDGEMENTS

Over the past four years I have received support and encouragement from a great number of individuals. I would like to express my deep appreciation to my advisor, Prof. Hans-Joachim Bargstädt, for the patient guidance and mentorship he provided to me. I will never forget his attempt at helping me to approach the science and practice of construction industry in Germany. I would like to thank Prof. Rolf Steinmetzger for his kind support throughout my four years of studying.

Amir Elmahdi, Jürgen Melzner, Ha Le, and Sebastian Hollermann have been my colleagues and friends, who have made my research life full of fun. I will never forget our "Imbiss" at the room 201. During writing, Amir Elmahdi and Maureen Anderson spent countless hours proofreading and listening to me about my research. I have learnt much through our conversations.

I would also like to thank MOET (Vietnamese Ministry of Education and Training) and DAAD (German Academic Exchange Service) for providing me a chance of studying in Germany. This is really a great time in my life.

Finally, I would like to give my special thanks to my parents and my sister for always being beside me.

Trang Dang

Weimar 03.10.2014

TABLE OF CONTENTS

1 INTRODUCTION

"A schedule is robust if its performance is rather insensitive to the data uncertainties."

--Billaut, Moukrim, and Sanlaville (2008)--

1.1 Problem statement

The lack of detailed planning is always a challenge for managers on site. Through an empirical study, Mallasi and Dawood (2001) indicated that the lack of detailed planning is a reason for 30 % non-productive time. It was also stated as one of the main reasons for trade stacking in the research of Hanna, Russell, and Emerson (2008). Generating a well detailed schedule, therefore, is necessary for a finishing period of execution, when several trades work side by side in a limited space.

The strategies that have been applied for generating a detailed schedule can be categorized into two groups: bottom-up and top-down. In order to determine the strategy that can be applied for this research, I have thoroughly weighed the advantages and disadvantages of each strategy.

The bottom-up strategy uses an aggregation mechanism to develop a schedule at the desired level of detail from a micro plan. It demands, at first, generating activities that are associated with primitive elements. These activities are called element activities. Afterwards, constraints will be established between them to represent either their physical constraints or the limitation of the associated resources. Manipulating the element activities to meet the assigned constraints would commence at the element-detail level according to a given rule, such as a spatial hierarchy. Subsequently, the element activities will be aggregated into summary activities at a desired level of detail. Examples for the bottom-up plan are the GPM (Geometry-based Process Model) of Akbaş (2004), the application of topology for generating construction planning from 3D CAD model of de Vries and Harink (2007) and the model of a simulation application in scheduling by König and Baling (2010).

The top-down strategies, on the other hand, use expansion mechanisms, or rather task-decomposition, to generate a detailed plan from a macro plan. Rough activities in a macro schedule must first be decomposed into sub-activities at a desired level of detail. Each sub-activity in this case normally involves a group of components. After this, physical dependencies such as technical requirements and other constraints such as spatial constraints between newborn sub-activities will be assigned based on the relationships of their direct summary activities. Manipulating tasks to meet constraints will be implemented in this desired level of detail. The GPM of Akbaş (2004), with the decomposition of schedules using region hierarchy, is also considered an example of top-down planning.

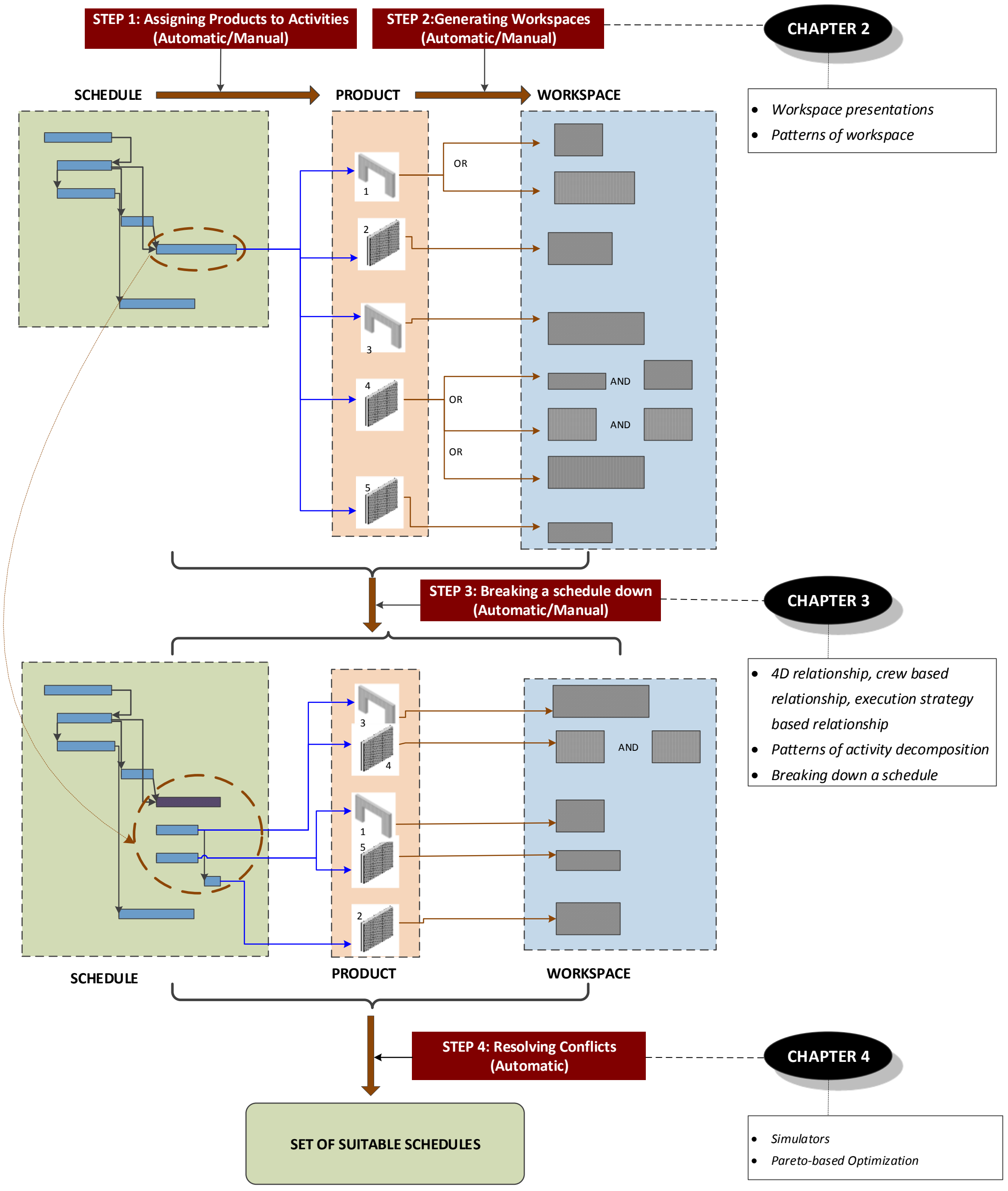
The most attractive feature of bottom-up planning is the ability to handle a complicated system with ease. Schedules with differently desired levels of detail for a complex building can be obtained by dealing with simple subsystems, each of which is only associated with a primitive component. Taking into account the advantages of Building Information Models (BIMs), the bottom-up approach is likely promising for an automation of scheduling. However, various drawbacks set limitations in implementing this method in practice. The first drawback is that solving conflicts at the component level might cause infeasible interruptions of trades and discontinuities of products, which lead to the inefficient relocation of equipment and temporary storage on site. The second drawback is that schedules established by summarizing fine-detailed activities lose their immanent flexibility. This decreases the adaptability of a schedule in facing uncertainties, which, by nature, exist in construction projects.

On the contrary, top-down planning is able to provide a big picture. The relative positions between components and organization strategies can be considered during the generation of sub-activities. Besides this, dealing with conflicts at the desired level of detail ensures the workflow is not too cluttered. However, it is not easy to apply the top-down approach for the automation of scheduling. The complexity and diversification of product groups up to now makes this method difficult to be systematized despite the demand for automation. This is the crucial disadvantage of an application in implementing a top-down approach to automate the generation of detailed schedules.

Taking into account the discussion above, I will follow the top-down strategy for generating a detailed schedule. I will develop a framework to make a rough schedule into more detailed schedules. This strategy ensures the flexibility of the schedule and enables it to adapt to uncertainties. In order to resolve the aforementioned disadvantages of the top-down approach in terms of automation, I will analyze the properties of trades, characteristics of their product composition, and the interaction between them to establish a knowledge-based system. This knowledge-based system can systemize and simplify the complexity of building construction and, thus, can make the top-down approach more compatible to automation.

1.2 Research objectives

This approach aims to build a framework for the automated detailing of a schedule. The input data for the model of detailing a schedule contains a rough schedule and a 3D model of its products. Figure 1.1 illustrates four steps that must be sequentially done to detail a schedule from such an input data: 1) Assigning products to activities; 2) Generating workspace; 3) Breaking down a schedule and 4) Resolving conflicts of activities.



STEP 1: Assigning Products to Activities (Automatic/Manual)
STEP 2:Generating Workspaces (Automatic/Manual)
CHAPTER 2
SCHEDULE
PRODUCT
WORKSPACE
Workspace presentations
Patterns of workspace
OR
AND
STEP 3: Breaking a schedule down (Automatic/Manual)
CHAPTER 3
4D relationship, crew based relationship, execution strategy based relationship
Patterns of activity decomposition
Breaking down a schedule
SCHEDULE
PRODUCT
WORKSPACE
STEP 4: Resolving Conflicts (Automatic)
CHAPTER 4
Simulators
Pareto-based Optimization
SET OF SUITABLE SCHEDULES

Figure 1.1: Framework of the automated detailing of a schedule

With the idea of supporting project planners with a transparent tool to control the processes associated with detailing, the model provides two options, automatic and manual methods, for every step. Planners can, at first, use the automatic options to efficiently generate or connect data to each other. After that they can manually check, investigate and refine the results of the automation process, as well as add other information if needed before moving to the next step.

It should be noted that step one, assigning products to tasks, has already been considered thoroughly in various previous research such as the models of Tulke (2010) and Tauscher (2011). Therefore, the connection of 3D products and activities is not the consideration of this research. This research just uses a simple XML file as a data template of information interacting between a schedule and a 3D product model. Then an activity can be assigned automatically with the objects that have correct categories as defined in the template and are located on the right locations as required. Steps 2 to 4 are the main parts of this dissertation.

In order to facilitate the automated detailing of a schedule, I have developed the following specific goals:

- to propose patterns for workspace generation by analyzing the characteristics of trades and the positions of their products.
- to identify and develop new sets of advanced relationships between activities, which are needed to enable the breakdown of a schedule to become an automation process.
- to establish patterns for decomposing activities by analyzing the characteristics of trades, execution strategies and positions of their products.
- to develop patterns for breaking down an activity by combining advanced relationships and patterns of activity decomposition.
- to build simulators that manipulate activities once spatial conflicts are detected.
- to suggest an algorithm that can generate suitable schedules based on the results of simulation processes.
- to carry out a prototype implementation, which is used as a tool for evaluating the proposed framework. This prototype is embedded in BIM-based software by using C#. The implementation is presented in each chapter to illustrate the theory and to evaluate its results.

1.3 Research scope

Building projects are the primary domain for the detailing of a schedule in this research. Due to the limited availability of resources such as material, workspace, manpower, and equipment on construction sites, detailed planning mostly involves resource-oriented scheduling. Among these resources, this research focuses on the constraints associated with the limitation of workspace and trade crews. Moreover, conflicts in terms of workspace often occur at the finishing phase of execution, when several trade crews work concurrently within a limited space. This research, therefore, prioritizes establishing strategies for the organization of detailed activities to resolve these conflicts in the finishing phase of execution.

1.4 Fundamentals of detailing a schedule

Detailing a schedule is a process which requires and involves various data existing in different formats of the construction project. I have raised several preliminary discussion aspects for identifying the data that should be considered in the process of detailing a schedule. This section presents briefly the three aspects that will help the reader to understand the structure of this research and its scope. The first aspect describes the definition of detailing a schedule. The second aspect indicates the parameters that must be considered in detailing a schedule. And the third aspect identifies the platform requirements for the automation of detailing a schedule.

1.4.1 Definition

Hendrickson (2008) stated that "construction planning is a fundamental and challenging activity in the management and execution of construction projects. It involves the choice of technology, the definition of work tasks, the estimation of the required resources and durations for individual tasks, and the identification of any interactions among the different work tasks". The work breakdown structure (WBS), which breaks down a construction project into finer, more manageable parts, is a very practical and efficient method to define activities. The duration of activities are afterwards generally estimated based on the availability of controlled resources, such as the amount of manpower for masonry, or the number of cranes for façade installation. According to Patrick (2004), after the definition of activities, it is best to consider physical constraints, which exist due to the physical process of construction, as the only constraint to sequence the defined activities. Other resources, i.e., materials, equipment, and manpower, may be considered at a later time when the time frame of activities of the schedule has been established and appropriate resource requirements have been identified.

Detailing of a schedule is a part of construction planning. After the tendering process for the project, the main contractors already have a rough schedule in hand. However this schedule does not contain enough detailed information for an execution phase. In order to meet the need of construction management in this phase, more detailed schedules at various levels of detail are needed to be generated.

Detailing of a schedule is defined as the process of creating a more detailed schedule by expanding a part or whole of the original schedule. It involves the decomposition of activities into finer and more manageable sub-activities at the desired levels of detail, identification of the interaction among sub-activities based on physical constraints, and sequencing the activities based on the availability of resources.

After the detailing process, the activities of the original schedule are maintained as integrated rollups or summaries of the detailed schedule activities. The relationships between the detailed activities are able to reflect the relationships of their summary activities as well as possibly depict their own dependencies more precisely. And finally, sequencing activities in the consideration of resource limitation can resolve various conflicts that the original schedule cannot.

1.4.2 Key parameters

In agreement with Echeverry (1991, p. 29), the breakdown of a project must consider two factors: delegation of responsibilities, such as subcontractors or trade crews, and workspaces. In this research, the delegation of responsibilities is assumed available in the initial rough schedule where each rough activity is just associated with a trade crew. In order to make a detailed activity more manageable, workspaces associated with a schedule activity should be organized such that the movements of crews and the reallocation of equipment are kept to a minimum. This means that the execution strategies and their suitability to individual trades are important. Furthermore, product positions and workspaces within an activity should not be too isolated from each other. In particular, it is ideal when workspace for an activity is formed as a singular region (Figure 1.2). Therefore, the key parameters that should be considered in terms of organization during breaking down activities associated with an individual trade are the execution strategy, product position and workspace location.

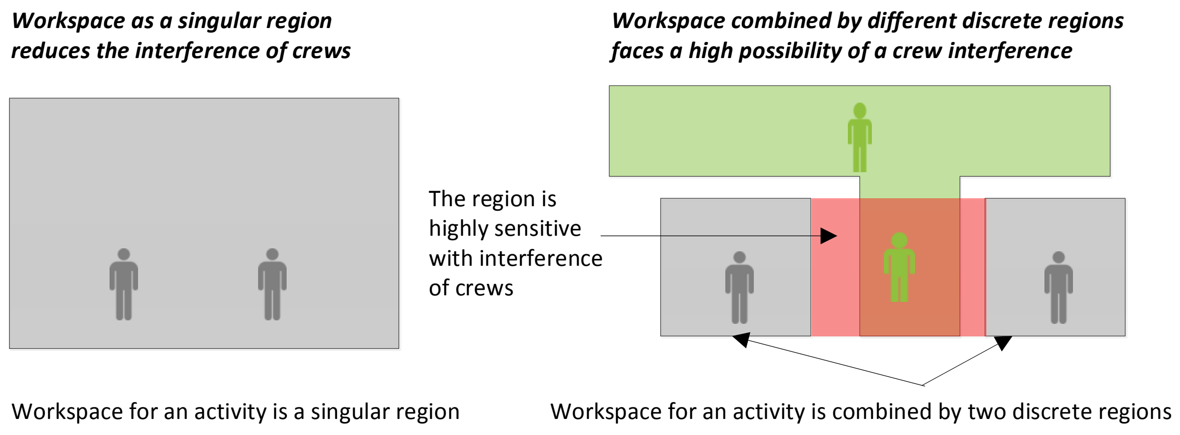

Workspace as a singular region reduces the interference of crews
Workspace combined by different discrete regions faces a high possibility of a crew interference
The region is highly sensitive with interference of crews
Workspace for an activity is a singular region
Workspace for an activity is combined by two discrete regions

Figure 1.2: The impact of workspace formulation on the possibility of crew interferences

1.4.3 Platform requirement for automation

To adapt the top-down approach to automation, this research will establish a knowledge based system, which is associated with trade types and characteristics of their products, such as their categories and positions. These properties play essential roles as parameters of the model. In order to enable the model to automatically recognize and categorize these input data, a building information model (BIM) must be integrated within a schedule as a 4D schedule. In this approach, a prototype implementation is embedded in commercial BIM-based software to provide a tool for evaluating and improving the proposed framework. C# is used as the programming language to develop this module.

1.5 Literature review and research gap

1.5.1 Schedule levels of detail

Different schedule levels of detail have been established based on available information ranging from preliminary schedules in the initial feasibility study to short-term schedules during the execution phase. Figure 1.3 shows the hierarchy of possible different schedules adopted during the life cycle of a project. The first level, the so-called executive summary, is a major milestone type of schedule. It contains major project activities, milestones and key deliverables for the whole project. It is usually used for the initial feasibility study of a project. The second level, the so-called master summary, depicts the overall project broken down into its major components by an area. It is developed as part of the client's commitment planning for the project and then maintained by the contractor. The third level is called the publication schedule. The publication schedule serves for the work performed at the macro-level and is used to focus on long term coordination, specifying terms of payment, off-site activities, equipment and manpower acquisitions for construction projects. It is usually developed by the main contractor as part of its tendering process or by the project team during the initial phases of planning. The fourth level, the so-called execution schedule, is the detailed working schedule developed by the contractor, trade contractor or project team prior to commencing work on the project execution, or work on a phase or an area of the project. Activities at this level are generally associated with major sections of the work over a week in duration. This level is used as short term plans or three week look-ahead schedules. The fifth level, the so-called detailed schedule, is developed to support day-to-day coordination. This level is also developed by trade contractors and superintendents. An activity in this level of detail is associated with a specific zone and varies from a couple of days to weeks depending on the complexity of the project. The sixth level, the so-called micro-detailed schedule, may only be used for analysing what-if scenarios in order to investigate complex problems before planning. An activity at this level may be several hours in duration and only consists of a segment of product in a simple shape. Each activity in this case is normally provided with full data information about resources such as manpower, material, workspace, and equipment. This level of detail might be too detailed to be controllable on site. However, it can be used for discovering or investigating the process of a specific part in complex situations. The first five levels of detail are referred to Project Management Institute (2007, p. 43) and the sixth level, the micro-detailed schedule, is added here to be able to illustrate the results of several research on detailed planning.

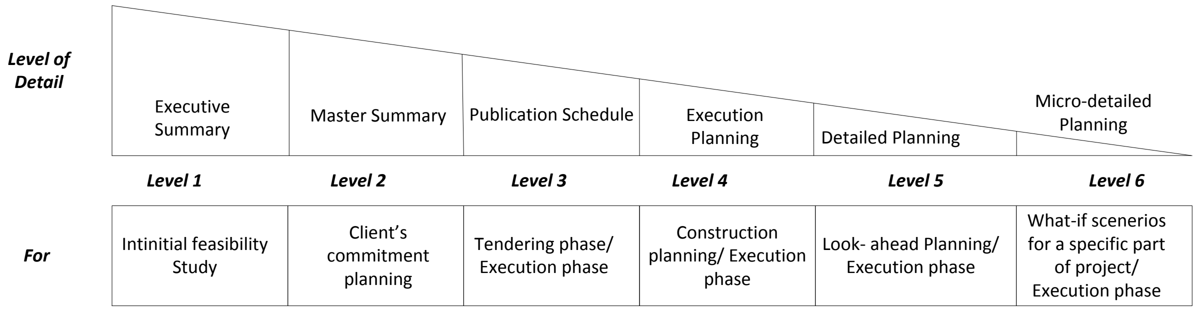

Level of Detail
Executive Summary
Master Summary
Publication Schedule
Execution Planning
Detailed Planning
Micro-detailed Planning
Level 1
Level 2
Level 3
Level 4
Level 5
Level 6
For
Intinitial feasibility Study
Client's commitment planning
Tendering phase/ Execution phase
Construction planning/ Execution phase
Look- ahead Planning/ Execution phase
What-if scenerios for a specific part of project/ Execution phase

Figure 1.3: Schedule levels of detail
(Level 1 to 5 are adopted from Project Management Institute (2007, p. 43))

1.5.2 Schedule generation based on BIM

The rapid development of Building Information Models (BIM) in architecture, engineering and construction (AEC) industries over the last decade has transferred the scheduling technique to 4D scheduling. BIM provides the insight into geometric structures, spatial relationships, quantities and properties of building components. Due to this benefit, BIM has improved communication between the stakeholders of a project, with more data sharing occurring between architects and engineers than previously. Recently, several researchers have studied the generation of 4D construction schedules based on end-product models, which can be received from architects. For example, Tulke (2010) proposed a model for the collaboration of 3D BIM with scheduling, and Tauscher (2011) and Kim et al. (2013) developed frameworks to extract construction schedules from BIM. More details are described following.

Tulke (2010) developed a linking language for data exchange between critical path method (CPM) schedules and BIM based building models. In his framework, Tulke also proposed different methods to support breaking down a project into finer zones, such as using axes and spatial algorithms, and splitting objects into smaller parts to ensure the compatibility of data granularity. This framework provides a better integration of an activity sequence and 3D components. Although Tulke advocated an efficient method for creating and managing 4D and 5D schedule, his work has been limited to the combination of a CPM schedule and a 3D model.

Tauscher (2011) presented an effort for the automated generation of construction schedules based on BIM. In his model, the schedule generation includes two processes: 1) a manual breakdown of a project in terms of geometry into finer packages of components, such as floors and zones; and 2) an automated generation of a schedule for each package based on the data extracted from its associated components. Based on the data extracted from the BIM based model, such as properties, structures and spatial relationships of building components, a database of required tasks and precedence relationships associated with components was developed to facilitate the automated generation of a schedule. A schedule for each package is generated by combining tasks and relationships with its associated components. In this schedule of a package, each task here involves a primitive design element. The schedule of a whole project is the combination of schedules of packages. Besides the schedule generation, this model also provides an advanced method to better investigate and manage 4D schedules, such as the functionality of floor shifting, which enables users to investigate all activities performing concurrently on various floors.

Tauscher advocated a framework for extracting information from a BIM based model to establish a database of tasks and precedence relationships associated with components. A schedule generated by this model is very detailed in terms of visualization, which can show exactly when a component is conducted and which task must be involved on it. However, the fact that each task of this schedule is associated with only one building element makes the schedule too cluttered and not systematic. For the sake of site managers, thus, such a detailed schedule is confusing and too expensive to be managed. Moreover, the interaction of tasks that are associated with different work packages has not yet been paid attention. In this model, schedules for individual packages are developed independently, and then they are combined and sequenced by the user-defined relationships at the level of packages. Thus the workflow of crews from one package to another was overlooked during scheduling.

Kim et al. (2013) also proposed a model for the automated generation of schedules based on BIM. This model first breaks a project down into several floors. Similarly to Tauscher's (2011) model, they also built a database of tasks and precedence relationships associated with a BIM based component by extracting the information from properties, structures and spatial relationships of that component. However, this model has a different mechanism to generate a schedule. First, they distribute element data into different activity groups. Then, the quantity data associated with an activity, such as durations and materials, is calculated at the floor level of detail. The relationships among tasks associated with a component are finally generalized to sequence the activities at the floor level of detail.

Kim et al. advocated an automated framework for generating a rough schedule based on BIM. However, one limitation of this framework that should be improved is the relationships between rough activities. In this model, the relationships between activities are created by the generalization of relationships between tasks associated with primitive components. For example, two tasks of plastering and painting must be done to complete a wall. They have a relationship of Finish-to-Start (FS), or rather the task of plastering must be finished before painting can start. This relationship afterwards will be generalized to be the relationships between the corresponding rough activities: plastering and painting for a whole floor. In the case of a large building, for example a floor area of 1000 m^2, this FS relationship cannot exactly illustrate the practical execution strategies. In practice, these activities can be conducted concurrently on the same floor in different zones as long as the zone in which painting is taking place has already been completed plastered.

1.5.3 Spatial scheduling

Detailed planning is associated with several resources such as workspace, material, crew, financial, machinery. However, among them only workspace is considered a key parameter for the process of detailing a schedule in this research, which is identified in Section 1.2.2. Therefore, previous research on spatial scheduling is discussed in this section.

For the past twenty years, various researchers have paid attention to detailed planning associated with workspace. Some of them have applied the theories of execution patterns and Line of Balance (LOB) to eliminate or minimize potential spatial conflicts (Riley and Sanvido 1997, Akbaş 2004, Jongeling 2006b). Some others have considered workspaces as constraints. They used time-based simulation to overcome congestion by reducing production rates, interrupting or delaying activities whenever spatial disputes between them are detected (Thabet and Beliveau 1997) or applying discrete-event simulation to resolve workspace overlapping (Elmahdi and Bargstädt 2011, Dong et al. 2013, Elmahdi 2013). Some researchers have proposed a model to manipulate the schedule with as many alternatives as possible (Winch and North 2006) or to integrate genetic algorithms with changes of execution patterns and production rates (Dawood and Mallasi 2006) in order to find the best solution with the minimum of spatial interferences. Others do not put effort into creating a schedule, but advocate their works for supporting space planning by building a method for the automated generation of workspace requirements (Akinci, Fischer, and Kunz 2002, Dang, Elmahdi, and Bargstädt 2012) and theories of execution patterns (Thabet and Beliveau 1994, Riley and Sanvido 1995). The following is the description and analysis in more detail of the remarkable previous research on detailed planning associated with workspace.

Thabet and Beliveau (1997) proposed a conceptual model for scheduling repetitive works in multiple floors. According to this model, activities must be decomposed into multiple work areas with different levels of detail such as floors, zones and blocks. Afterwards, horizontal and vertical construction logic (HVL) for activities, as well as workspace must be created and assigned into activities. HVL and workspace are then used as the constraints for sequencing works. While considering workspace constraints, if any spatial conflict for storage occurs, then the activity associated must be delayed or split. If certain spatial conflicts of manpower and equipment occur, then they would first try to reduce the productivity. If this attempt is not satisfying then the delay or split of activities would be considered. As a result, the schedule might contain several workspace conflicts. However their impact on crew productivity has already been taken into account during scheduling.

Thabet and Beliveau were the pioneers of making rules for decomposing a schedule. They paid attention to the classification of activities based on their spatial demands and also discussed automatic workspace generation. However, their model does not involve much in the automation process. They did not regard to establish the rules or patterns to create zones, blocks, and workspace or present how this kind of geometric information could be formulized. They also did not pay attention to creating rules for generating relationships between new activities. In addition, because of the technique limitation at that time, input data in their model was done manually and with specific information.

Riley and Sanvido (1995, 1997) suggested a method for space planning for interior works. This method manually decomposes activities into various floors, rooms and building faces, and then assigns them with corresponding workspaces. This method also uses workspace as constraints to manipulate activities for detailed planning. The spatial conflicts can be resolved by changing the execution direction and adjusting workspace layout. If these adjustments cannot solve the conflicts then it is suggested that they should be decomposed into more detail. This approach is purely a conceptual model and not relevant to automation.

In this approach, Riley and Sanvido identified twelve space types and six work area patterns. This general information provides informed data so that successive research could generate knowledge-based systems for generating workspace and relationships between activities associated with a certain similar trade. In terms of conceptual scheduling, it still did not pay attention to the organization for multiple crews involved in a similar trade, nor the relationship between activities associated with different trades.

Dawood and Mallasi (2006) introduced a tool to assign workspace and identify spatial conflicts. The input data are activities with their 3D product objects. The tool considers product positions and surrounding spaces as their workspaces. This approach offers a time-based simulator for investigating spatial conflicts, a function to quantify conflict volumes, and a module to minimize workspace conflicts by changing execution directions and production rates. Execution sequences of product objects over time are the outcome of this tool, which is able to minimize the schedule's spatial conflicts.

This approach offers a good tool to investigate the alternatives for execution directions and work rates at the original planning phase. However, it is not an efficient tool to be used for controlling in the execution phase since the object sequence is not easily accessible. Any change of either this sequence or production rate in reality compared to as-planned would confuse the managers, because they could not know which effect this change could result in. In addition, the output of the tool performed in only one sequence could not provide managers a big enough picture for potential alternatives on site.

Winch and North (2006) proposed a construction management tool for analysis, optimization and visualization of the spatial loads attached with schedule tasks. It begins with manually importing schedule activities, their product objects, and creating their corresponding workspace by marking up drawing areas. It supports users manually adjusting start and end dates of activities in order to acknowledge its effect in terms of spatial congestion. It also provides “brute force” algorithms to optimize the loading of execution spaces.

In this approach, the output has the same level of detail as the input. Hence, it requires much effort to manually generate input data when working with detailed schedules. Furthermore, with only one solution as the outcome, the tool limits its functionality to providing managers an informed view of diversified possibilities of scheduling on site.

Jongeling (2006) adopted the Line-of-Balance (LOB) scheduling technique in combination with 4D CAD for space planning. This model begins with manually dividing a group of 3D components in consideration into multiple sections. These sections are afterwards prioritized in a location-hierarchy and manually linked to their processes. By using the LOB technique, a LOB diagram with non-spatial conflicts and an optimized flow of resources is given as a result. In addition, the 4D CAD embedded in the LOB diagram allows users to quickly and clearly gain insight into the spatial configuration of scheduled activities.

This approach advocates a method to manipulate activities in order to eliminate potential spatial conflicts and to optimize the flow of resources in detailed planning. However, it does not pay attention to the automation of breaking down a schedule. The scheduled activities must be manually decomposed at the desired level of detail before the model can be applied.

Elmahdi (2013) proposed a model of a grid-based discrete-event-simulation (DES) for space planning. This model requires very detailed activities with their technical constraints and workspace requirements as input data. These activities can be analyzed through a DES process. Whenever a spatial conflict is detected, then an adjustment of workspace requirements can be applied to eliminate it. If this attempt does not work, then the associated activity can be moved to later. A schedule free of spatial conflicts would be given as the result.

Although this approach suggests a knowledge system for generating micro workspace for individual interior finishing trades, it does not provide an automatic option for workspace generation. In addition, this approach, like the above mentioned approaches, does not consider breaking down a schedule. In this model, the levels of detail of input and output schedules are kept the same. This model, however, is recommended for establishing the knowledge-based system of workspace generation.

Dong et. al (2013) proposed a method to automatically generate a look-ahead schedule for finishing trades. In this approach, room is considered as the key to decomposing physical zones into smaller sections to be used as workspace for an operation. They also suggested operation-specific spatial (OSS) constraints, i.e. blocking and zone constraints, to represent relations among rooms. Input of this model contains a list of rooms, lists of trades performed in each room, constraints between trades, relations of rooms associated with individual operations through OSS constraints, and other information for resources and work durations. This information must be manually extracted from 2D drawings. These input data after that goes through a simulator to manipulate activities so that every constraint is satisfied. The result of a simulation process is a schedule which is free of spatial conflicts. This model can also run multiple random simulations to select a close-to-optimum look-ahead schedule.

Like other approaches mentioned above, the level of detail for input and output schedules are the same. In this model, the input activities are generated somehow automatically by combining the information of associated rooms and trades. However, the input data of this model, such as a list of OSS constraints, and a list of trades associated with each room, are difficult to be reused for other projects. This reduces the automation capability of the model.

Akbaş (2004) introduced an approach for modeling and simulating construction processes based on geometric models and techniques, called GPM. In this research, Akbaş considered the geometric model as triangle meshes and described a construction process as a sequence of crews acting in geometric work locations, which are formed by one triangle element or more. The direction and speed of crew movements through work locations in performing tasks are defined by workflow strategies and production rates. A set of crews working at locations in a bounded space defines a subsystem. The interaction between subsystems depends on activity ordering, spatial crew ordering and nearby work. A discrete-event simulation is established by coupling of a set of subsystems and their interactions. This model is able to evaluate the strategies of crew organization, workflow, and production rate considering the limit of working area.

Unlike the other approaches above, this approach already paid attention to an automated decomposition of activities. By an automated decomposing, i.e., the breakdown of a project into the physical geometry of triangle meshes and the process into subsystems, this model has been able to decompose activities into finer ones. This model significantly contributes to the automated generation of subsystems for a discrete-event simulation system based on geometry.

Despite its significant contribution, the model of Akbaş still has not yet been able to reflect the workspace requirements of activities, especially for trades involved with discrete products. Representations of work locations in Akbaş's model include various different types, which are described in Figure 1.4. However, the automation works only based on triangle base meshes (Figure 1.4a, b). So the model is suitable to trades, whose products are continuous and have the same location as their working areas, such as earthworks or paving (Figure 1.4a). But for trades that have discrete products such as framing (Figure 1.4b), such an automated method cannot reflect the real workspace of an activity, but rather can only present the product positions.

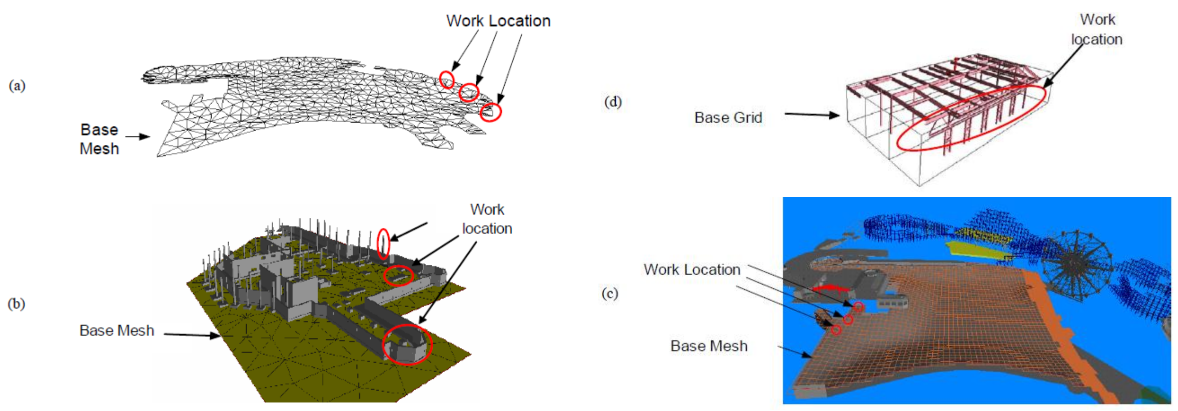


Figure 1.4: Examples for work locations and base structures in Akbaş's (2004) model

Besides this, Akbaş's model considers that an activity, or rather a subsystem in this discrete-event simulation system, is associated with only a single discrete building component (Akbaş 2004, p.70). Therefore, it cannot regard the relative positions of components in planning and also makes a schedule very cluttered. According to this model, another trade might occupy the nearby component first, so this trade must wait until the other finishes. Although this model proposed a nearby factor to consider the impact of nearby work on productivity, acceptance of this solution loses opportunities of potentially better alternatives. This modification can result in an accuracy of duration estimation for activities, but cannot resolve stacking trades and the idle time of crews.

1.5.4 Conclusion and research gap

The recent research on BIM has shown an effective way to automate the generation of schedules, especially, the establishment of a database of tasks and precedence relationships associated with BIM based components. However, the aforementioned research is limited to the generation of a schedule associated with a pre-identified zone. There is no attention paid to the automated generation of well-organized work breakdowns. Moreover the relationships between rough activities in these researches have not been specified precisely.

In terms of spatial scheduling, Figure 1.5 illustrates an overview of the evaluated research on detailed planning associated with workspace. According to this analysis, almost all of the presented approaches require similarly detailed levels of input and output. This means in order to create detailed planning, it is first required to manually decompose schedules, generate relationships between newborn detailed activities, create working areas, and assign workspaces to corresponding activities. Only the research of Akbaş is an exception. He already focused on how to decompose activities into finer ones. However, as analyzed in section 1.5.3, his research is not yet suitable to be applied for detailing activities involved in the finishing construction phase.

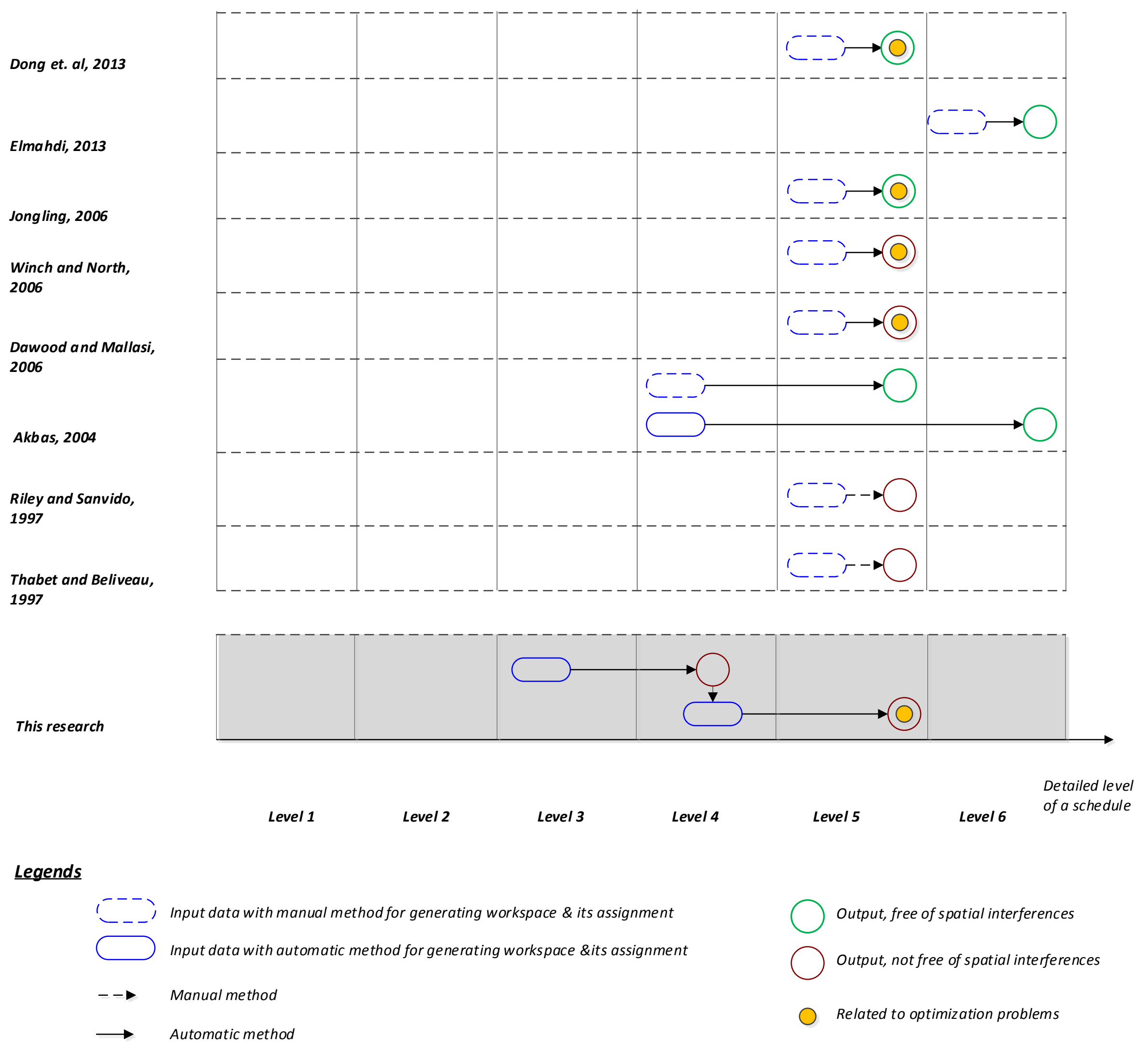

Dong et. al, 2013
Elmahdi, 2013
Jongling, 2006
Winch and North, 2006
Dawood and Mallasi, 2006
Akbas, 2004
Riley and Sanvido, 1997
Thabet and Beliveau, 1997
This research
Detailed level of a schedule
Level 1
Level 2
Level 3
Level 4
Level 5
Level 6
Legends
Input data with manual method for generating workspace & its assignment
Input data with automatic method for generating workspace &its assignment
Manual method
Automatic method
Output, free of spatial interferences
Output, not free of spatial interferences
Related to optimization problems

Figure 1.5: Overview of the research on detailed planning associated with workspace

Another shortcoming of current scheduling is the static status of relationships, which always exists in the course of scheduling. It should be mentioned that these relationships are created based on the structure and topology of components. So any change of component members of an activity can break or establish a new relationship of this activity to another. However, the current static activity cannot reflect this issue. A relationship once created will exist until the end of the schedule. A more dynamic relationship is therefore needed to be developed to overcome this issue and to enable an automated connection of sub-activities.

Finally, only one alternative given as output of thorough scheduling is still a disadvantage. It is well known that detailed planning is associated with a lot of constraints and very specific data, whereas construction projects are characterized by uncertainty, diversity and complexity. This means that there is hardly ever a “one size fits all”-solution, which will work well for all construction projects. So a model with only one solution reduces its applicability in reality. In order to improve the adaptability of the model to the diversity of construction projects, multiple alternatives should be proposed.

In conclusion, the above evaluated research proves a current shortcoming of an appropriate model for the automated detailing of a schedule in the finishing phase of execution. This shortcoming is related to the decomposition of activities associated with finishing trades, need of a set of more dynamic relationships between activities, and creation of multiple alternatives for resource conflict resolution.

1.6 Structure of the dissertation

This dissertation is divided into five chapters. This chapter has already presented an overview of the research. It includes the motivation for conducting the approach, the overall purpose, description of its framework and review of associated literature. The following chapters will proceed in detail of this work.

Chapter 2 describes the patterns of workspace generation. In this part, previous research on this area is first analyzed concerning its advantages and disadvantages. Afterwards, a new model is built by assimilating the advantages of previous models and providing other structures to overcome the disadvantages in order to meet the demands of the automation of detailing processes. Then the patterns of workspace generation for finishing trades are developed by analyzing their characteristics. A prototype implementation is finally presented to make the chapter more coherent.

Chapter 3 presents the automation of breaking down a schedule. This chapter first identifies essential relationships between activities to enable the process of the break down to become an automated system. Second, it focuses on developing the decomposition patterns of schedule activities for individual trades. Thereby, each trade is associated with a specific execution strategy to accomplish its products. Aside from trade type, these patterns are also established based on product positions and workspace. Third, the method for an automated breakdown of a schedule is proposed by integrating new advanced relationships and decomposition patterns. Finally, a prototype implementation is introduced to evaluate the results.

Chapter 4 demonstrates the model to find solutions overcoming possible workspace conflicts. This is the last step of the detailing process. In this chapter, the reason why the integration of simulation and Pareto-based optimization has been chosen to resolve this problem is first explained. Then simulators and optimization framework are described in detail. After that a prototype implementation is conducted and its results are illustrated. Finally a discussion of the efficiency of the model is presented based on these results.

Chapter 5 summarizes the research work. The contributions of the research, advantages and disadvantages of the proposed methodology, suggestions for application of the research model, and future recommendations are presented.

2 WORKSPACE GENERATION

"Lack of execution pace planning interrupts and badly affects the progress of construction activities. Also, in real situations, spatial congestion can severely reduce the productivity of workers sharing the same workspace, and may cause health and safety hazard issues."

--Dawood and Mallasi (2006)--

2.1 Overview and definitions of key terms

Workspace is necessary data that must be considered when detailing a schedule. This section analyzes previous research on generating workspace requirements for construction activities and reasons why current workspace generation has not yet fulfilled the automation needs of scheduling. Based on this analysis, a method of workspace generation is proposed to overcome these existing shortcomings. The proposed method in this research is an integration of the advantages of previous research and a newly-developed hierarchy of workspace allocation.

In order to make this chapter more coherent, the following presents the definitions of terms that appear and are used in this chapter, such as workspace, one-surface trade, two-surface trade, object and product.

Definition 2.1 *Workspace is the physical space that is required to conduct an assembly activity.*

Definition 2.2 *One-surface trade is a construction activity whose required workspace can only be located in one, and only one certain side of its associated object.*

Examples for one-surface trades are plastering, painting, screeds, ceiling finishes.

Definition 2.3 *Two-surface trade is a construction activity, whose required workspace can be located in either one side or the other side of its associated objects.*

Examples for two-surface trades are masonry and framing.

Definition 2.4 *Object is a building component, on which an activity performs its work.*

Definition 2.5 *Product is the performance of an activity.*

According to Definition 2.4 and Definition 2.5, the terms *object* and *product* are used in different meaning in this research. *Object* is used to refer to a general element such as wall, floor, and ceiling. In order to create an *object*, various trades are required, e.g. to produce a wall, the trades of masonry, plastering and painting must be carried out. What a trade creates is called a *product*. That means, for example, a *product* of the trade of masonry is a brick wall, a *product* of the trade of plastering is a plastered surface of the wall.

2.2 Literature review of workspace generation

In the past twenty years workspace planning has already received a lot of interest from researchers. However, many of them have just considered workspace requirements as available input data of individual activities and used them to support workspace planning. The research of Dawood and Mallasi (2006) and Bansal (2011) are examples for this case. Every workspace in these studies is formed by manually setting the coordinates of the workspace corners. Such a workspace formation requires a huge effort on behalf of the modelers in terms of data preparation. This is therefore not suitable with the automation process of scheduling.

In other studies, researchers have focused much more on workspace generation itself. They have put in an effort to establish a model that is able to automate this process (Akinci, Fischer, and Kunz 2002, Elmahdi 2013). Workspace in these studies does not stand independently from involved products. Instead, it is a derivation of the associated product. This method of workspace generation promises much more efficiency for an automation process. This section, therefore, focuses on analyzing these methods of workspace generation in order to identify their advantages as well as obstacles in their application to scheduling.

In the research of Elmahdi (2013), workspace generation is presented as a part of a simulation model for allocating workspaces and resolving time-space conflicts. Workspace concerned here is formulated at the micro level of detail. In order to establish the model of workspace generation, he undertook site interviews and observations of the interior trades and their work procedures to capture common characteristics. He examined practical execution strategies for individual trades at the interior construction phase, and their workspace arrangement strategies such as required workspace types and their occupation behavior over time. The aim was to enable a model to automate the generation of the laborer workspace size for multiple trades in a simulation model based on common characteristics. The analysis has shown that workspace size can be automated based on the building object size and the nature of the activities including the work procedures. Yet he considered human engineering in the determination of a laborer workspace. Workspace in his research depends not only on the size of building elements but also on the suitable size for human operation. This human engineering consideration enables a model to generate feasible workspaces in case the size of a building element is smaller than the human operation size. In summary, Elmahdi has advocated a knowledge-based system of micro workspace formation associated with individual trades in the finishing phase. However, workspace allocation to corresponding products and activities was still implemented as a manual method in this research.

The research of Akinci, Fischer, and Kunz (2002) was motivated by the difference of workspace among different construction methods. There, a specific product could be executed from different positions depending on the method applied. Therefore, the researchers focused on analyzing relative positions between a product and its associated workspaces to establish a structure of input data for the automated generation of workspace requirements, which can reduce the time for creating workspace input data compared to the manual methods of assigning actual coordinates. They determined that a workspace presentation must contain a qualitative description of its position such as outside, inside, below, above, and a quantitative description of its size with deterministic dimensions.

These aforementioned two researches formulated an advantage of workspace generation. According to them, the size and position of a workspace can be changed automatically depending on product geometry. These characteristics are compatible to the automated generation of workspace requirements based on their associated product positions.

Table 2.1: Generating workspace requirements for activities in previous research

WORKSPACE RESEARCH OF	INPUT DATA	WORKSPACE POSITION & SIZE		METHOD FOR GENERATING WORKSPACE
		changeable with product's position	***changeable with product's size***	
(1) Elmahdi	workspace size of resources; relative positions of resource's spaces to respective task's products; positions of products; resources required by tasks	YES	YES	manually
(2) Akinci et al.	workspace template of activities: qualitative description of position and quantitative description of size; type of activities	YES	YES	automatically generated based on available templates
(3) Dawood, Mallasi, Winch, North (VIRCON)	coordinates of potential working area; size of workspace	NO	NO	manually
(4) Bansal	coordinates of workspace	NO	NO	manually

However these methods are still not adequate enough to be directly applied to the automatic detailing of a schedule. Several shortcomings still remain. The following is an analysis on the disadvantages and determination of the obstacles that should be overcome to enable the workspace generation to be an automated process.

A workspace is formalized by a quantitative description of its size and a qualitative description of its positions (Akinci et al., 2002). In terms of workspace size, we can obtain the template associated with individual trades by applying the research of Elmahdi (2013). This means the size description of workspace can be automatically generated. However, the requirement of workspace orientation as input data poses various challenges in terms of logical presentation and thus puts an obstacle in the way of automation.

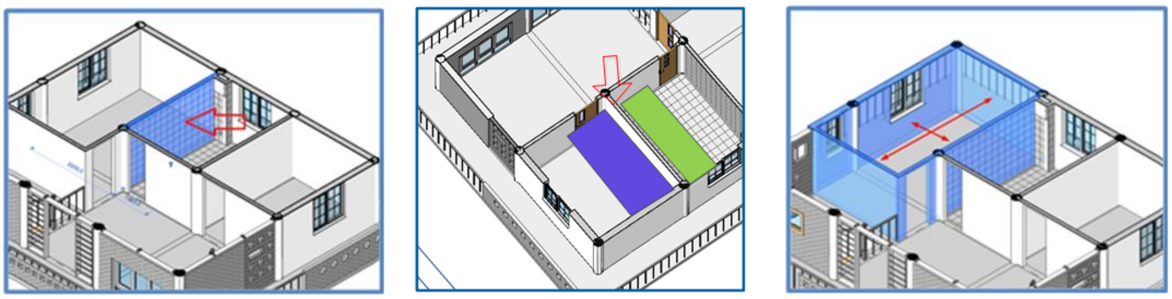
The first challenge expresses the difficulties in distinguishing the workspace orientations in respect of the reference object. It is relatively easy to catch the differentiation between distinguishable orientations such as *interior* vs. *exterior* or *below* vs. *above*. *Interior* vs. *exterior* surfaces of an object can be recognized by their typical features of materials and colors. They are also differentiated by referring to the boundary of a building. Similarly, the *above* vs. *below* can easily be differentiated according to the gravity coordinates of the objects. However, it is very confusing to identify the orientation of an object which has two equal surfaces. An interior wall with two surfaces in need of painting is an example of this case. Not in every case are we able to distinguish these two surfaces by using global directions such as *east*, *west*, *south*, or *north* because the wall positions and directions vary without any rule. They do not always merely stand in the four standard directions of east, west, south or north, but also stand in other directions such as southwest, northeast and so on. Besides, it is also impractical to differentiate them by using their typical features such as material or color since they can be the same on both sides. If using the left or right hand to distinguish them, schedulers could be confused since they must consider how they stand to identify the left or right side.

The second challenge is presented in the process of input data generation for activities associated with various objects. A schedule activity generally involves multiple components with differently located directions. Therefore, in various cases, it is impossible to find a common workspace orientation for all components relevant to the activity. If a common orientation for all objects of an activity could not be found, then the schedulers must put much effort into manually assigning individual orientations with corresponding objects. The workspace allocation for installing interior doors and painting interior walls are typical examples for this challenge when specific orientations must be identified manually.

In addition, the specific orientation for workspace required as input data does not only pose the challenge of establishing workspace representations, but this requirement also limits the model's ability in terms of flexibility. In several situations, we can locate the workspace on either one side of an object or the other side without any differentiation of a technical issue. The location of workspace for building interior brick walls or framing interior partition walls are examples of this case. No matter on which side of a wall we locate the workspace, its size and the equipment supporting the construction are the same. Therefore, in this case, we should create two possibilities of workspace locations corresponding to two sides of an object. On which side of an object the workspace should be assigned will be identified later according to organization strategies and the workspaces of other activities located nearby.

Furthermore, previous researchers have not considered the relative positions of components to generate a workspace associated with their group. These workspaces for a component group are simply a combination of the workspaces associated with the individual components. This might cause inefficiency of spatial organization and low productivity of workers. Let's assume an electrical crew works in a room to rough wires in the walls. In several cases, the combination of the workspaces required for the individual walls is smaller than the room. However, this electrical crew should occupy entirely the room to ensure safety and to avoid low productivity due to the interference of other crews. In another example, a framing crew might frame several wall segments. As mentioned before, there is more than one possibility of workspace positions to frame a wall. So the workspace for this crew should be selected from these possibilities such that the movement of the crew is most efficient and the interference with other crews is minimized. These examples show that merely combining predefined workspaces associated with individual components without considering the relation of their positions is not the suitable method to create the workspace for a component group.

In conclusion, previous research has already advocated a model that is able to establish a knowledge-based template for workspace size for individual trades (Elmahdi 2013) and a structure of workspace representation in respect to reference objects (Akinci, Fischer, and Kunz 2002). The structure of workspace identified by Akinci et. al must contain its orientation in relevance to reference objects and workspace size, which could be set with either fixed volume or variable volume. However, an integration of these methods is still inadequate for the automation of workspace generation. As summarized in Figure 2.1, current research has not yet provided an efficient method to identify the workspace orientation for one-surface trades, not yet considered multi-possibilities of workspace allocation, nor a well-organized formation of workspace associated with component groups.



Previous research is not able to recognize the workspace orientation for one-surface trades, such as painting without pre-identification

Previous research is not able to provide two possibilities of workspace allocation for two-surface trades such as masonry

Previous research has not focused on choosing workspace for a group of components

Figure 2.1: The gap of previous research on the automation of workspace generation

This research, in order to overcome the disadvantages of the previous research in terms of automation, will establish a template of workspace size by deriving the model of Elmahdi (2013) and create a new concept for workspace structure. The improved workspace structure will be able to recognize workspace orientation automatically without pre-identification, provide more than only one possibility for workspace allocation, set the workspace for a component group based on its relative positions, as well as consider room area as a workspace for a component group in feasible cases. Aside from this, it also analyzes characteristics of individual trades in the finishing phase in order to establish patterns for workspace generation. This helps the proposed model to be more efficient in terms of automation.

2.3 Framework of workspace generation

Workspace is a location where a construction activity is carried out. It can be a space for laborers working, a place on which equipment such as scaffolds are set up, or a specific area for storing material. Bargstädt and Elmahdi (2010) classified workspace into 13 types and assigned them to three groups (Figure 2.2). In my approach, however, only the type of workspace related to a process area is regarded. It also does not take into account the difference between the place for laborers, equipment or material. A workspace, in my research, is considered as the total area that is required for an activity. It includes spaces for laborers, in-place material, equipment, hazard spaces and areas for debris.

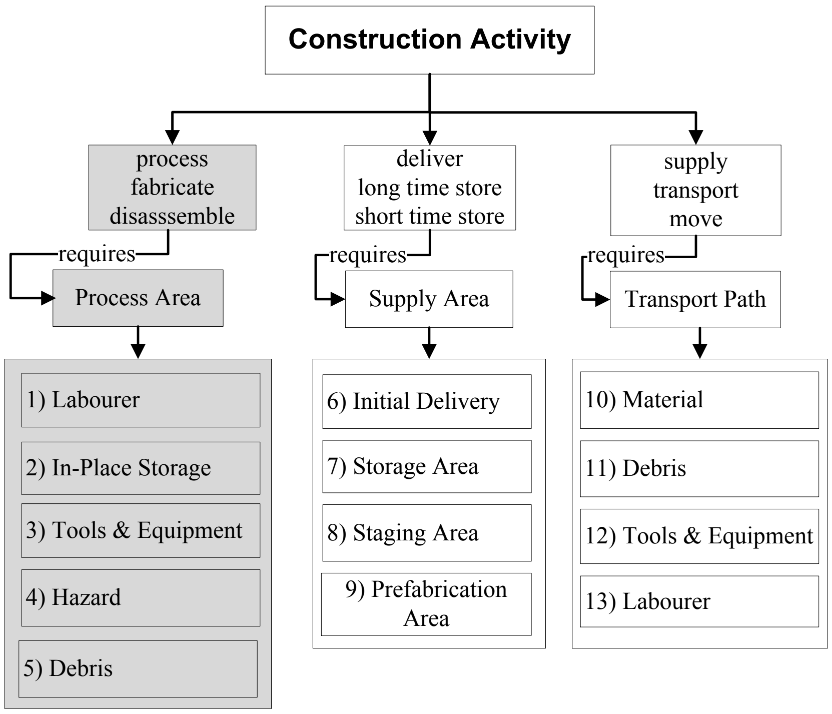


Figure 2.2: Workspace classification (Bargstädt and Elmahdi 2010)

A workspace can be defined in various ways with different levels of complexity. It can be identified with fixed or variable volumes, located at a static position during the whole time an activity is carried out, or at a dynamic position, which varies along a product length in the course of activity duration. Besides, some activities may have more than one possibility for a workspace position. And finally, no matter how the workspace is formulated, its position varies with the location of the product associated.

Figure 2.3 presents the proposed framework of workspace generation. The hierarchy of workspace generation is subdivided into two levels: workspace formulation and workspace allocation. Workspace formulation is the process for creating the size of the workspace; and workspace allocation is responsible for identifying the location where the workspace is located. In this model, an activity can have one or more possible locations to set up its workspace. This is defined by its trade type, or rather, whether it is a one-surface trade or two-surface trade.

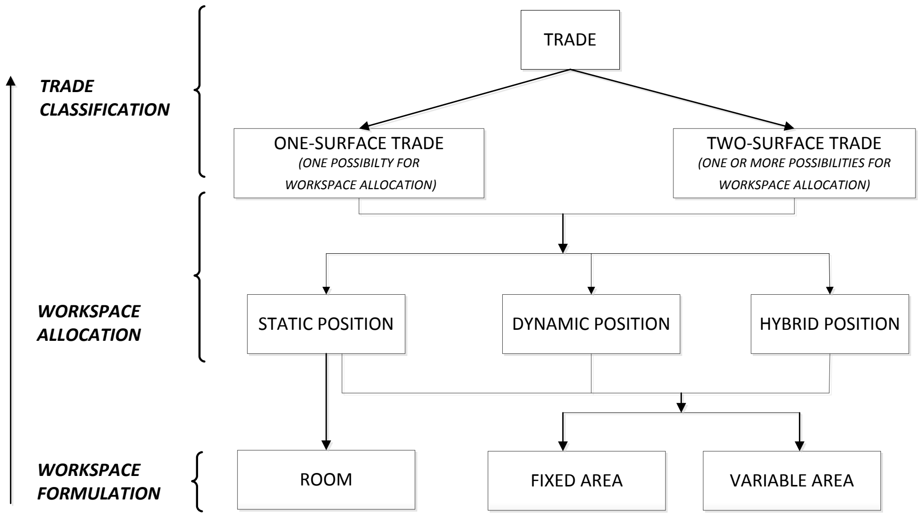


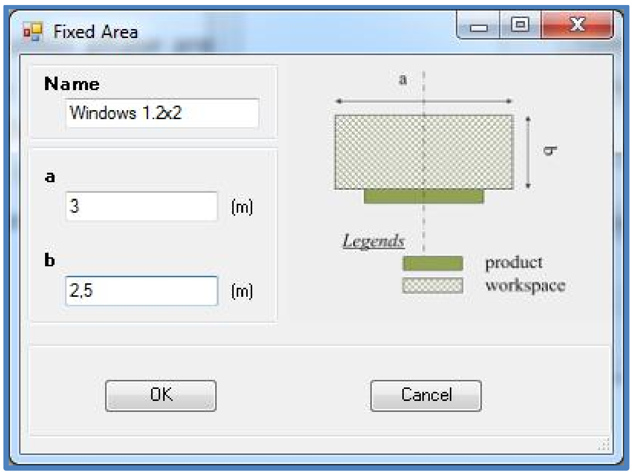
Figure 2.3: Framework of workspace generation

2.4 Workspace formulation

Workspace formulation focuses on how to formalize the size of a workspace. This approach provides three types of workspace formulations: fixed size, variable size and room. The suitable type to be used in workspace formulation for an activity is chosen according to the type of its associated objects.

2.4.1 Fixed size

In a finishing phase, several products have standardized sizes such as windows and doors. In order to install the same products with the same size, workspaces require the same area. In this case, a fixed size with a specific length and width is suitable to identify a workspace. Figure 2.4 presents a user interface to generate a fixed size. This formulation requires the width *a* and the length *b* to create a workspace.



Name: Windows 1.2x2

a: 3 (m)

b: 2,5 (m)

Figure 2.4: Fixed-size workspace

2.4.2 Variable size

Trades in the finishing execution phase are related much to the existence of walls, floors, and the like, which have different sizes even in the same project. Workspaces required for conducting these trades, therefore, have different dimensions depending on the size of products. In this case, variable sizes that can vary with the length and height of a wall, or length and width of a floor are more convenient for identifying their associated workspace.

Figure 2.5 depicts a user interface to create a workspace size for activities associated with vertical objects such as walls and windows. In this case, the size of the workspace varies with the length and height of the products. The length of workspace is calculated as the total of the product length *L* and the buffer length *2xa*. The width of the workspace is defined by the total of the product height *H* and the buffer width *b*. However, this model offers the *max* value to control the upper threshold of the workspace width. If the workspace width that is calculated by *(H+b)* exceeds the *max* value, it is set back to the *max* value.

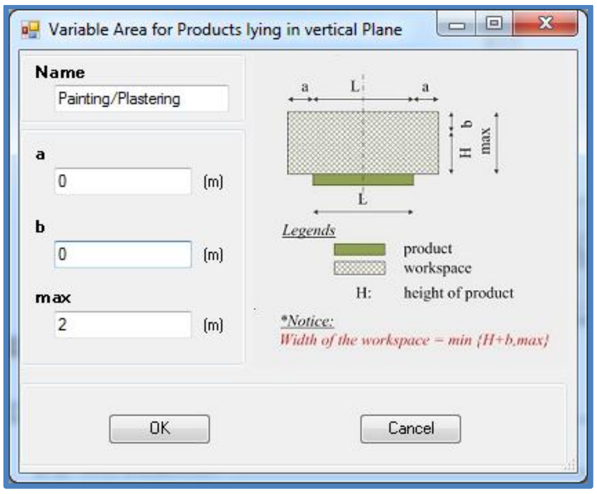


Figure 2.5: Variable size for products lying in vertical planes

Figure 2.6 depicts a user interface to create workspace sizes for activities associated with horizontal objects such as floors, ceilings and stairs. In this case, the size of workspace varies with the length and width of products. The area of workspace is calculated as the total of the product area and the buffer area, which is defined by the buffer length *a* and buffer width *b*.

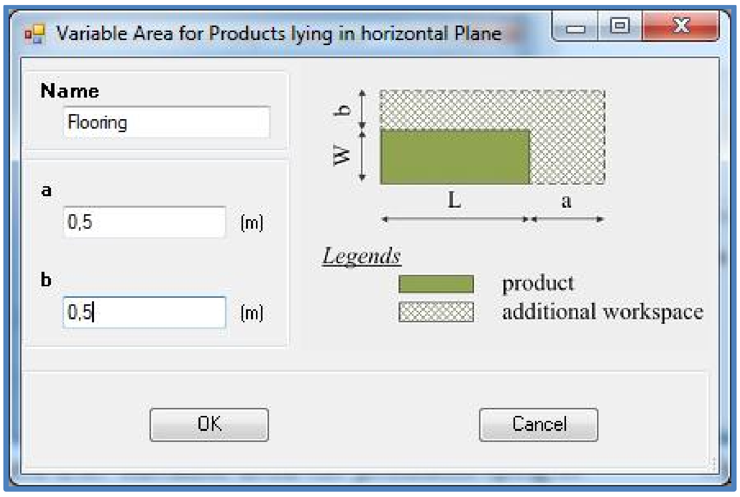
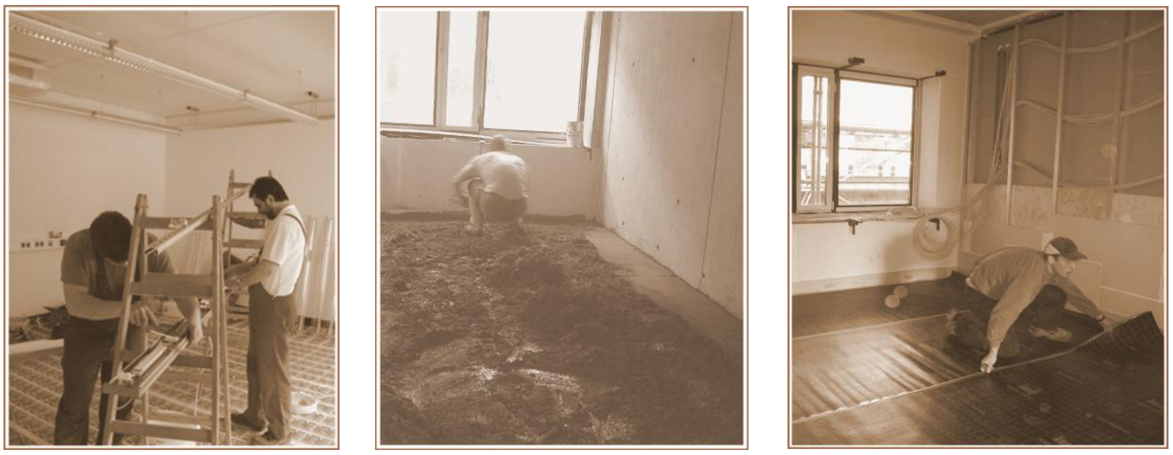


Figure 2.6: Variable size for products lying in horizontal or leaning planes

2.4.3 Room

In the finishing phase, working areas are normally isolated from each other by partition walls. Hence, various trades like painting, flooring, ceiling finishing, and MEP (mechanical, electrical and plumbing) once taking place in a room, usually occupy the entire room for their execution duration (Figure 2.7). No interruption by other crews is preferred in order to avoid interference and loss of productivity. It is beneficial, therefore, to collect a room to identify a workspace.


Electrical workers block entirely a room for their work

A worker must block a whole room for screeds

Laying waterproofing layers requires a whole room to be executed

Figure 2.7: Using a room as a workspace

2.5 Workspace allocation

Like other resources, workspace has a property of temporality. A given workspace does not exist for the whole duration of a project, but is required for just a short time for a task, even for a part of a task. This approach describes three types of workspace allocation: static position, dynamic position and hybrid position.

2.5.1 Static position

In this approach, allocating a workspace with a static position means that the position and dimension of the workspace is not changed in the course of the task duration. This type of workspace allocation is suitable in cases where products of a task do not stand far away from each other or the task duration is not so long.

✦ ***Workspace allocation for one-surface trades***

Trades like plastering, painting, flooring, ceiling finishing and the like, whose products can be carried out from just one certain side of their objects, are called one-surface trades in this approach. The workspace required to perform these trades is located at the position corresponding to the product side. Allocation of a workspace for a one-surface trade associated with a single segment is illustrated in Figure 2.8. This allocation method requires a room as a reference factor to identify the orientation of workspace towards its object, i.e., the orientation of workspace is located at the direction identified from the object to the room. Moreover, since a room is isolated by walls around, this research uses the reference rooms to limit the size of workspaces as default. If a workspace requirement is greater than its reference room, the workspace distributed to that segment is limited to the size of the reference room, of course a warning will be issued. After creating workspace related to single products, the workspace for an activity is then generated as the combination of workspaces associated with its involved products.

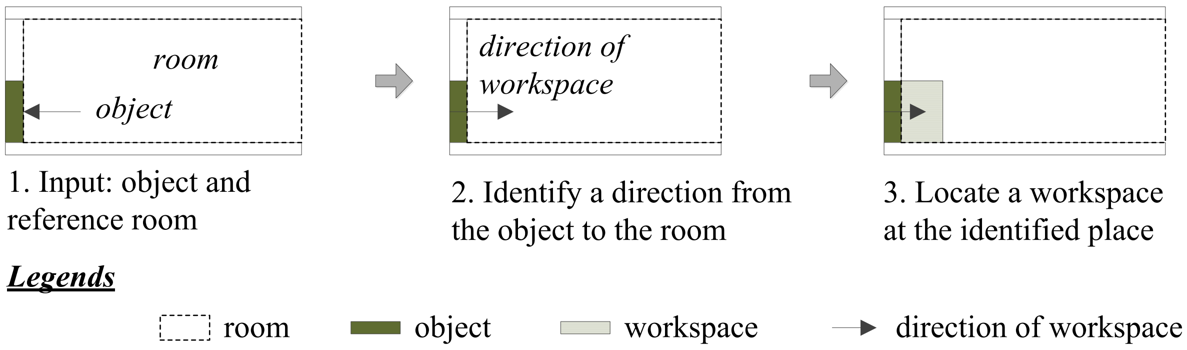

room
object
direction of workspace
1. Input: object and reference room
2. Identify a direction from the object to the room
3. Locate a workspace at the identified place
Legends
room
object
workspace
direction of workspace

Figure 2.8: Workspace allocation of a single segment associated with one-surface trades

Since one-surface trades are normally performed in isolated spaces such as rooms, each activity normally occupies the entirety of these rooms if they are small. This ensures avoiding low productivity due to interference with other crews. In this case, the workspace for an activity is a combination of reference rooms of relevant one-surface products.

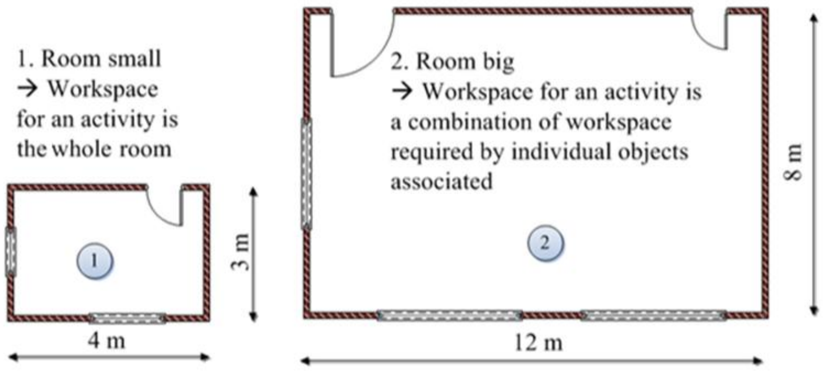
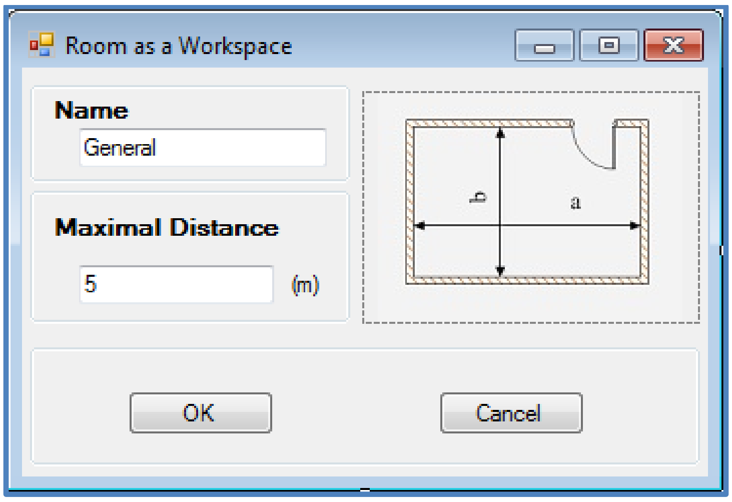

1. Room small
→ Workspace for an activity is the whole room
1
3 m
4 m
2. Room big
→ Workspace for an activity is a combination of workspace required by individual objects associated
2
8 m
12 m

Figure 2.9: Workspace allocation in relevant with room size

Figure 2.9 illustrates examples of workspace allocation in accordance with the room size. The crew for painting any of four wall surfaces around the room (1) will occupy the whole room during its duration since the room is small. If another crew also comes to occupy this room at the same time, they may disturb each other and cause loss of productivity. However if the room (2) is large enough, then more than one crew can work together without disturbing each other. So in this case, workspace for each trade is the summation of workspace required by the product concerned.


Room as a Workspace

Name: General

Maximal Distance: 5 (m)

Figure 2.10: The factor determines whether a room is considered *big* or *small*

In order to identify whether a room is *big* or *small*, a variable of the maximal distance from two objects inside a room will be considered (Figure 2.10). If the maximal distance between any two objects in a room is greater than the permitted value, then the room is *big*. In contrast, if the maximal distance between any two objects in a room is smaller than the permitted value, then the room is considered *small*.

Algorithm 2.1 depicts briefly an example of object identification and workspace generation for one-surface trades associated with a building. In this example, all rooms of the building are firstly identified. After that, the objects associated with one-surface trades will be identified by collecting objects around these rooms. The objects refer the respecting rooms as the reference parameter. The workspace for each object is generated with the algorithm presented in Figure 2.8 if the reference room is large. Otherwise, the workspace is assigned as the area of the reference room.

Algorithm 2.1 Object identification and workspace generation for one-surface trades

```
Input:  $M$:        building model
        $catList$:  Categories of objects that are associated with the considered trade
Output: $objList$: a list of objects and their associated workspace
$objList = \emptyset$;
$roomList$ = all $room \in M$;
foreach Room $room \in roomList$ do
    $oList$= all $obj$ around $room | obj$.Category $\in catList$;
    foreach Object $obj \in oList$ do
        $obj$.roomReference = $room$;
        if $room$ is small then /* see Figure 2.10                            */

            $obj$.Workspace = OneSurface-Workspace-Generation ($obj$); /* see Figure 2.8   */
        else
            $obj$.Workspace = $room$;
    $objList = objList \cap oList$;
return $objList$;
```

✦ ***Workspace allocation for two-surface trades***

Trades like masonry, drywall framing, installing doors and the like, whose products can be built from both sides of their objects, are called two-surface trades. Taking installing doors as an example, this kind of trade can take place on each side of the object or on both its sides as well (Figure 2.11).

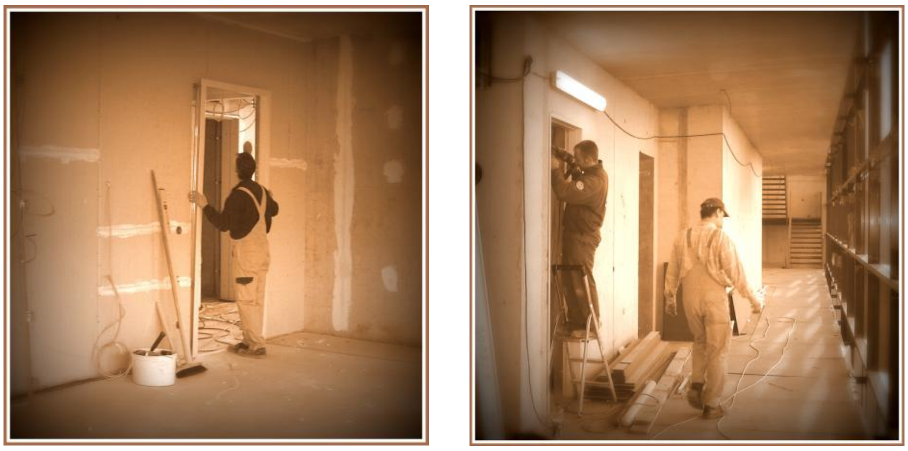

Installing doors can be executed from this side.. ..or from the other side

Figure 2.11: Possibilities of workspace allocation for installing doors

Figure 2.12 presents multiple possibilities for allocating workspace for such trades. In some cases, each side of an object plays an equal role in identifying workspaces. This means, no matter on which side of an object a workspace is located, the amount of its area is the same, such as workspace for installing interior doors. In other cases, however, they require different workspace sizes as well as supporting equipment. For example, the required workspace for building a perimeter brick wall can be allocated either on the outside or inside of a building. When a wall is built from outside, the width of workspace required is just the width of scaffolds, about 1.00 to 1.25 meters. However, if it is carried out from inside, maybe, up to 4 meters of width are used as the workspace.

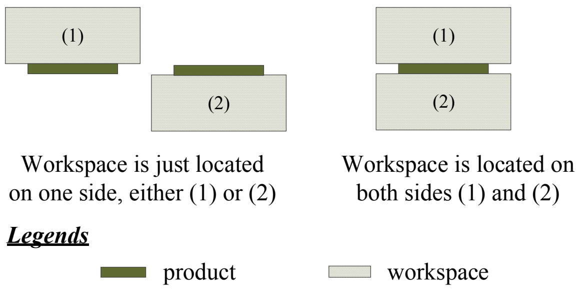

(1)
(2)
(1)
(2)
Workspace is just located on one side, either (1) or (2)
Workspace is located on both sides (1) and (2)
Legends
product
workspace

Figure 2.12: Possibilities of workspace allocation for a single segment associated with two-surface trades

Therefore, for two-surface trades, two data sets of workspace formulation must be defined when the role of object sides is not equal in setting up workspace. To distinguish from one another, a local coordinate system is used. Figure 2.13 depicts the local coordinate system of a wall. Here the local origin is located at middle point of *P1* and *P2*. The positive x-axis is defined by the direction from *P1* to *P2*. Based on this coordinate system, the workspace location marked with (1) will be located on the positive y-axis. In contrast, the location marked with (2) will be located on the negative y-axis.

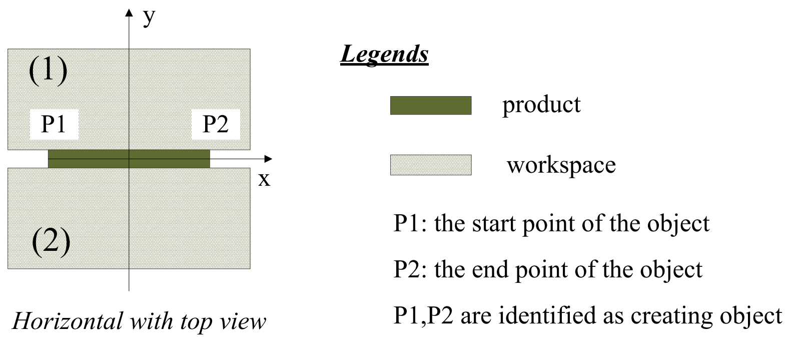

y
(1)
P1
P2
x
(2)
Horizontal with top view
Legends
product
workspace
P1: the start point of the object
P2: the end point of the object
P1,P2 are identified as creating object

Figure 2.13: Identification of the positions of workspace (1) and (2)

Workspace for an activity associated with two-surface trades is a combination of workspaces required by individual product segments. The method of workspace combination should, however, obey several practical rules to ensure an effective execution method. Generally, the number of practical workspace combinations is much fewer than the number of theoretical combinations.

In theory, the total number of possibilities of workspace allocation for an activity is obviously determined as the multiplication of the number of workspace possibilities associated with every product segment concerned. For example, activity *A* is associated with three objects a_1, a_2 and a_3, where let's assume that working on each a_i (with $i = 1$ *to* 3) has two possibilities of workspace allocation. As a result, *A* will have *2x2x2,* or rather, *8* possibilities for workspace allocation. In practice, however, this total number is much less due to the constraints of construction methods and organization. Let's consider the masonry work on the perimeter walls of a building as an example. It is well known that the masonry work on each perimeter wall has two possibilities of workspace allocation, i.e. inside or outside. As a result, an activity *A* theoretically will have 8 possibilities of workspace allocation if it is performed on three walls a_1, a_2 and a_3. However, in practice, if a_1 is constructed from outside, the others, a_2 and a_3, should generally be done from outside as well. This organization avoids the unproductive movement of crews from outside to inside and ensures the efficient allocation of scaffolding and temporary material storage. Therefore, due to this constraint, activity *A* in practice has only two possibilities for workspace allocation.

In this approach, the practical rule for the workspace combination is the minimization of workspace for an activity. This is to say that a schedule in this approach can provide all possibilities of the workspaces associated with individual objects. However, when the workspace for an activity should be presented, the minimum workspace will be shown as default. Aside from this, workspace associated with perimeter objects can be selected by the user, either outside or inside of the building.

2.5.2 Dynamic position

Workspace allocation with a dynamic position, in this approach, means that the position of workspace can be changed during a task's duration. From a practical perspective, relocation of workspace is planned on site either at the end of the day or at the end of the week. Thus, in order to produce a realistic interval time *t* for proceeding workspace positions and to reduce the complexity of the model, this research considers an interval time *t* as an integer value.

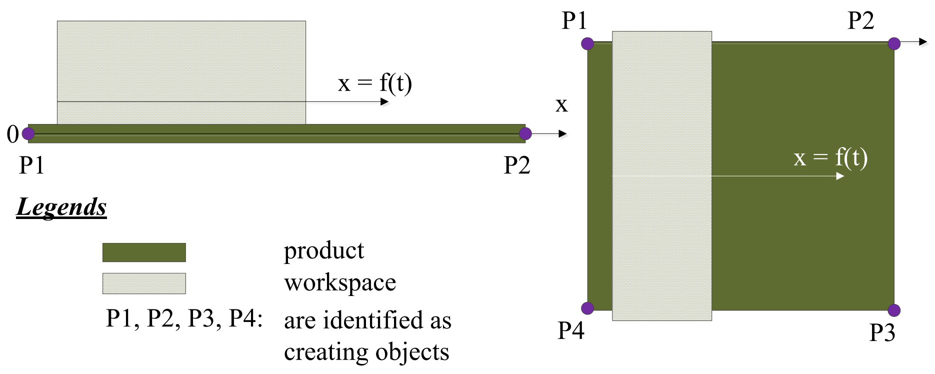


Figure 2.14: Sliding workspace

Figure 2.14 illustrates a sliding workspace that is changed along a direction. In this illustration, the size of workspace at a given time *t* is constant, but its position *x* is changed following the rule:

$$x = f(t) \tag{2.1}$$

There, *x* denotes the smallest corner coordinate of the workspace at the time point *t*; *x* is normally calculated in a local coordinate system associated with individual objects.

f is the user-defined function describing the change of *x* over time.

t denotes the time point.

This sliding workspace is suitable to be used to generate dynamic workspaces for activities associated with vertical objects such as walls.

In contrast, Figure 2.15 illustrates a two-direction moving workspace, whose position changes in both x-axis and y-axis. Similarly to sliding workspace, the size of this workspace type is also constant over time. However, the change of its position is defined by a couple of functions:

$$\begin{cases} x = f1(t) \\ y = f2(t) \end{cases} \tag{2.2}$$

There, *x* and *y* denote the smallest corner coordinates of the workspace. Normally, *x* and *y* are calculated in a local coordinate system associated with individual objects.

f1, f2 are respectively the user-defined functions describing the changing of *x* and *y* over time.

t denotes the time point.

This two-directionally moving workspace is suitable to represent the dynamic workspace for activities associated with horizontal objects such as floors and ceilings.

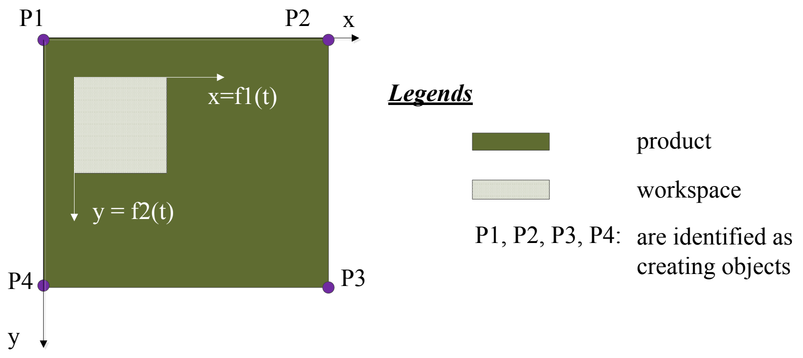


Figure 2.15: Two-directionally moving workspace

Dynamic positions for workspace allocation are suitable to tasks that are carried out to produce continuous products occupying a large area, and if the equipment and material involved are easily moveable over time. The trades of plastering, painting, flooring, wiring in a large room are good examples for this case.

Obviously, a dynamic position requires considerable complexity of data structures. This puts the dynamic position at a disadvantage. However, workspace allocation with a dynamic position offers several advantages in comparison to a static position:

- The dynamic position method creates fewer tasks in a detailed schedule, so the schedule is more controllable.
- The dynamic position method ensures the continuity of workflows. For example, if an activity associated with a very long wall is decomposed into smaller sub-activities that are involved in several segments, it is not clear that the wall is built continuously because of the independence of tasks in a schedule. However, if it remains as one

activity and uses the dynamic position method for allocating workspaces, the wall will be built continuously and the number of interruptions for the trade is controllable. The time for travel of laborers and relocating equipment therefore is reduced.

2.5.3 Hybrid position

Let's consider the following situation: a manager is faced with scheduling the construction of very long walls. The first possibility: the manager assigns the entire wall to only one masonry task with a static position. As a result, the use of workspace is insufficient since this task occupies a large area along the wall in the course of its duration, whereas it just needs a small part to perform its work for a specific time. The second possibility: he decomposes this wall into different segments and then assigns them with several fine activities. This choice might result in a disorderly sequence during the construction of the wall, unproductive movements of crews and the relocation of equipment. The third possibility: he creates workspace with a dynamic position and assigns it to a task. This method can address the shortcomings of the two possibilities above. However, the calculation and analysis process of this method is highly complicated.

This section, therefore, proposes another method, called the hybrid position, to overcome these problems. The hybrid position is proposed for workspace allocation that is able to reflect the continuity of the workspace occupation sequence (which is offered by the dynamic position method) by using a simple data structure (which is the positive feature of the static position method). The hybrid position is considered as an integration of execution strategy based relationships (EBRs) with static positions. In this method, a large product is at first decomposed into several activities. The sequence of execution is then arranged in a specific order by means of EBRs to reflect the technical requirements in terms of continuous construction. Notice that the EBRs are described in detail in section 3.3.4.

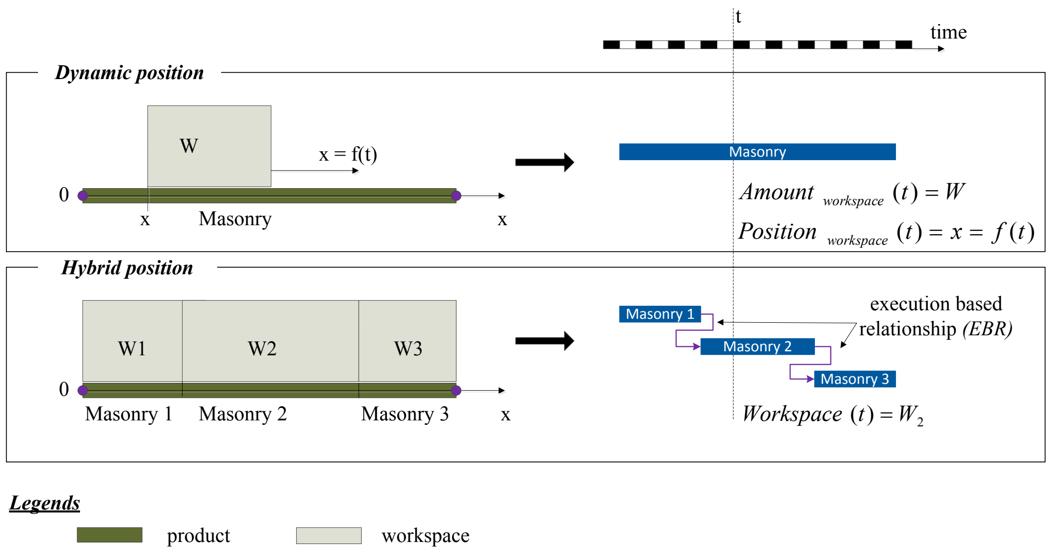


Figure 2.16: Using workspace allocation of hybrid position to replace dynamic position

With the above modeling, this method can present the workspace for big components sufficiently. Like the static position, the workspace associated with an individual activity of this method does not change over time, so the calculation is simple. Similar to the dynamic position, the sequence of execution is ensured to be arranged in specific order with the support of EBRs. Therefore, it is able to reflect the technical requirements in terms of continuous construction. With such advantages of the integration of static positions with EBRs, this research uses the hybrid position for workspace allocation later on instead of the dynamic position. Figure 2.16 describes an example of this replacement.

2.6 Patterns of workspace generation

This section suggests the patterns of workspace formulation and allocation for specific trades in the finishing phase. The workspace formulation and allocation for each trade can take more than one possibility, where one can be better than the others in different situations. So, depending on the characteristics of individual projects, the manager can decide his final choice.

Notably, trades that use rooms to formulate workspaces normally include more than one option for workspace formulation. A room might be a choice for the workspace of a group of objects, whereas the other possibilities are for creating the workspaces associated with individual objects. These other options will be used if a room is not sufficient to illustrate the required workspace. For example, if a room is too big, it should be further broken down into smaller workspaces. In this case, the workspace formulation with individual objects will be used.

It should be mentioned that the workspace allocation with the dynamic position is not included in this section. As mentioned above, the hybrid position would be implemented in this model instead of the dynamic position.

Table 2.2: Patterns of workspace generation

ID	**Trade Name**	**Trade Kind**		**Workspace Formulation**			**Workspace Allocation**	
		One-surface	Two-surface	Room	Fixed size	Variable size	Static position	Hybrid position
1	Framing (frame, brick wall)	-	o	-	-	o	o	o
2	Stairs	o	-	o	-	o	o	-
3	Rough HAVC	o	-	o	-	o	o	o
4	Rough Plumbing	o	-	o	-	o	o	o
5	Rough Electrical	o	-	o	-	o	o	o
6	Insulation	o	-	o	-	o	o	-
7	Drywall	o	-	o	-	o	o	-
8	Interior Plastering, Painting	o	-	o	-	o	o	o
9	Flooring	o	-	o	-	o	o	-
10	Exterior Plastering, Painting	o	-	-	-	o	o	-
11	Curtain Wall from inside	o	-	-	-	o	o	-
12	Curtain Wall from outside	o	-	-	-	o	o	o
13	Doors/Windows	o	o	o	o	o	o	-

****Note*: **o** denotes the coupling

2.7 Prototype implementation (4Dworkspace)

The module for workspace generation in this approach is called "4Dworkspace". 4Dworkspace is developed in Visual C# with the support of Revit API, and then embedded in Autodesk Revit®, a 3D-BIM-based environment, where we can extract and control the information of object geometry and their categories efficiently. 4Dworkspace involves more than 10 forms, 20 classes and 10,000 lines of code (excluding spaces and comments). The source code can be found in the CD attached with this dissertation.

This section describes the design goals, challenges, user interface and functionality of 4Dworkspace. The descriptions are illustrated through an example of generating workspace on a real building for various activities in the finishing phase of execution. In order to make the presentation clearer, this section is divided into three sub-sections. The first sub-section states briefly the design goals and challenges for 4Dworkspace. The second sub-section presents the generation of input data for 4Dworkspace. And the third sub-section discusses the performance of workspace entities.

2.7.1 Design goals and challenges

The implementation aims at automating the process of workspace generation in the finishing phase. It attempts to enable end users to put in as little effort as possible while working with 4Dworkspace. It also makes efforts to support them efficiently in checking the accuracy of the results. The design goals for 4Dworkspace, hence, have the following characteristics:

- ***Generality***: the implementation must be able to support workspace generation for various activities, product geometry, and model structures.
- ***Data reusability***: the data, once created, must be ready for reuse in various projects afterwards. This is the most important feature in order to enable the automation of workspace generation as well as reduce the working time of end users.
- ***Ease of use***: this is expressed through a friendly user interface and the ability of an easy evaluation of the accuracy of the input data as well as the model results.

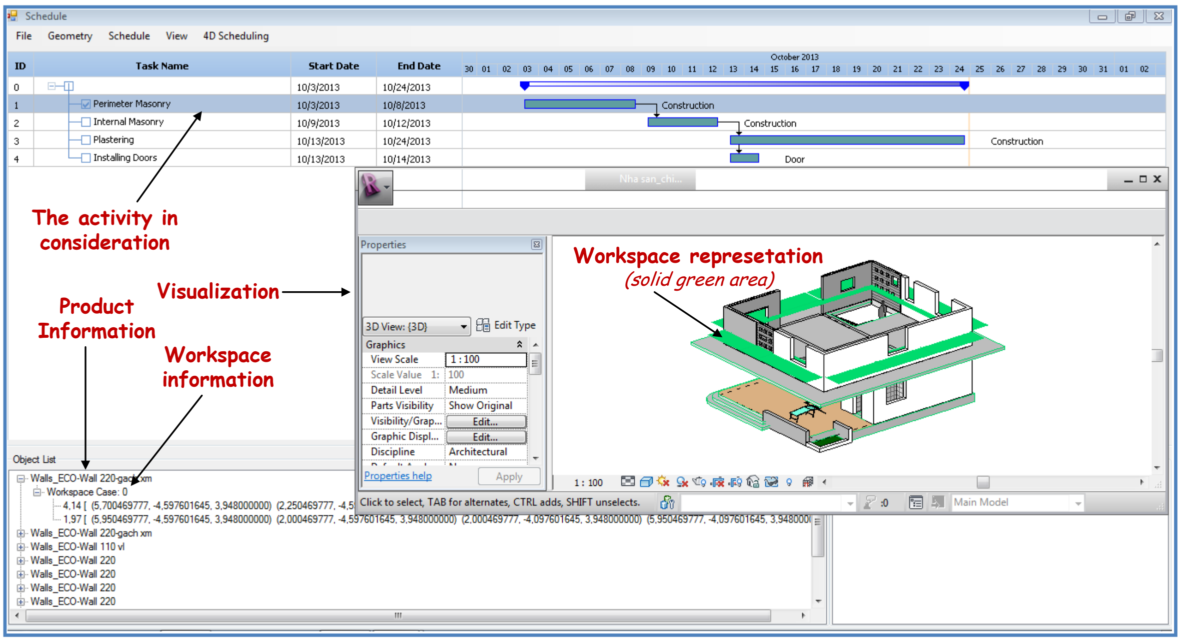
During the development of 4Dworkspace, various implementation challenges come with the given design goals. Specifically, much effort has been put into dealing with the variety and complexity of geometry as well as data structures of the model.

- ***Variety and complexity of geometry***: the 3D model contains different product categories. Each category has its own geometric shape and typical data structure. This raises the challenge of extracting and analyzing the necessary geometric information of individual products. Also, the interactions among different product components of a building are complex. These pose a challenge for finding an effective algorithm for geometry analysis, which can adapt to its variety and complexity.

- ***Complexity of data structures***: workspace generation involves various data, such as schedule activities with their relationships, 3D products, workspaces and other supported information. From an end user's perspective, this complexity of data structures challenges making an adequate data hierarchy so that the end user can clearly know about what he is doing, and then be able to control the data efficiently. From a developer's perspective, this complexity also confronts the extensibility of the implementation. The data must be organized in compact and independent classes in order to enable the developer to apply further extensions to functionality without great effort.

2.7.2 Functionality of 4Dworkspace

4Dworkspace supports two options for workspace generation, automatically and manually. Although the implementation attempts to provide the automatic option for various geometric models as well as characteristics of construction trades, it could never ensure the ability to cover all situations in every construction project. Hence, the manual option can support end users to adjust the workspace information whenever the results achieved from the automatic option are not accurate enough. It gives the end users more flexibility in using 4Dworkspace.



ID	Task Name	Start Date	End Date
0		10/3/2013	10/24/2013
1	Perimeter Masonry	10/3/2013	10/8/2013
2	Internal Masonry	10/9/2013	10/12/2013
3	Plastering	10/13/2013	10/24/2013
4	Installing Doors	10/13/2013	10/14/2013

Graphics	
View Scale	1 : 100
Scale Value 1:	100
Detail Level	Medium
Parts Visibility	Show Original
Visibility/Grap...	Edit...
Graphic Displ...	Edit...
Discipline	Architectural

Click to select, TAB for alternates, CTRL adds, SHIFT unselects.

```
Walls_ECO-Wall 220-gach xm
  Workspace Case: 0
    4,14 [ (5,700469777, -4,597601645, 3,948000000) (2,250469777, -4,5
    1,97 [ (5,950469777, -4,597601645, 3,948000000) (2,000469777, -4,597601645, 3,948000000) (2,000469777, -4,097601645, 3,948000000) (5,950469777, -4,097601645, 3,948000
Walls_ECO-Wall 220-gach xm
Walls_ECO-Wall 110 vl
Walls_ECO-Wall 220
Walls_ECO-Wall 220
Walls_ECO-Wall 220
Walls_ECO-Wall 220
```

Figure 2.17: User interface of 4Dworkspace

4Dworkspace is also able to analyze various categories and geometries of 3D products normally associated with the finishing phase, such as walls, floors, rooms, columns, doors, and windows. Moreover, it is able to differentiate the positions of product components to provide corresponding respondents. To express them more in detail, 4Dworkspace can distinguish between the components laying on the perimeter and those in the interior of a building. This differentiation enables it to provide the corresponding options for workspace allocation.

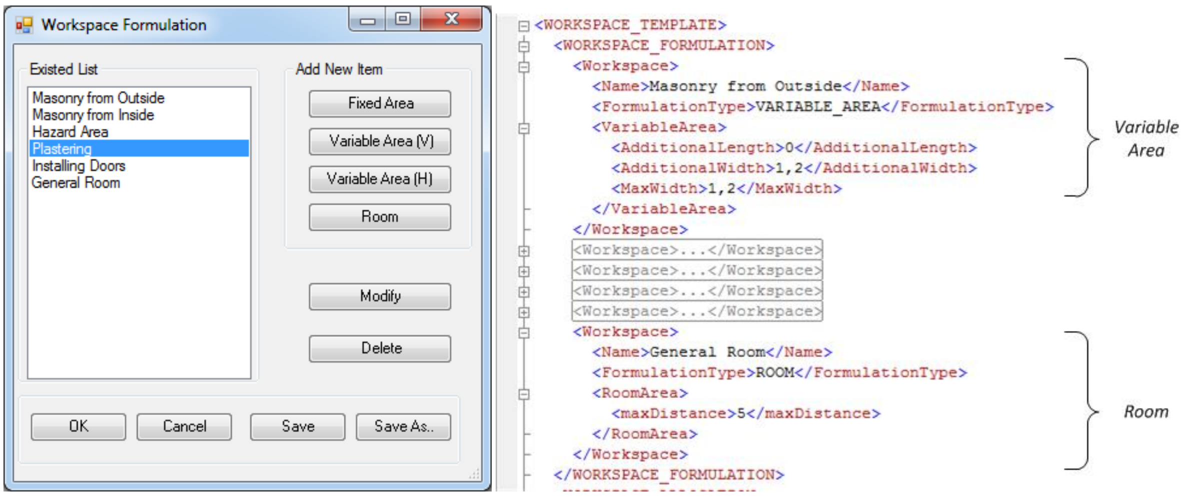
In addition, 4Dworkspace has a friendly user interface (Figure 2.17). It provides full interaction among schedule activities, 3D product components and workspaces. A schedule activity contains multiple product components and might have common workspaces, which are used for the whole activity. Each product component contains one workspace possibility or more. Each workspace possibility is composed of one or multiple discrete spaces. The user can isolate, show, hide, create, modify or remove the products and workspaces of a specific activity or workspaces of specific product components when conducting a given construction activity. This enables users to efficiently control data as well as evaluate the results.

In 4Dworkspace, a workspace is positioned in the 3D space in accordance with its associated product components. However, its representation is illustrated through plane components. Due to safety issues, there is almost no occurrence of a case in which a workspace for one activity is located above another on the same floor. In addition, representing a workspace as 3D components might complicate the view and thus confuse users. Therefore, a plane representation for workspace is suitable for providing information as well as efficiently investigating the workspace distribution.

2.7.3 Generation of workspace data

The input data for the 4Dworkspace can be created and accessed via a user interface. The data, after it is created and modified, can be saved as XML file format in order to be reused later. In 4Dworkspace, templates of workspace formulations are stored with the extension .WSF, templates of workspace allocations are stored with the extension .WSA, and templates of the combinations of workspaces, trades and product categories are stored with the extension .BAP.

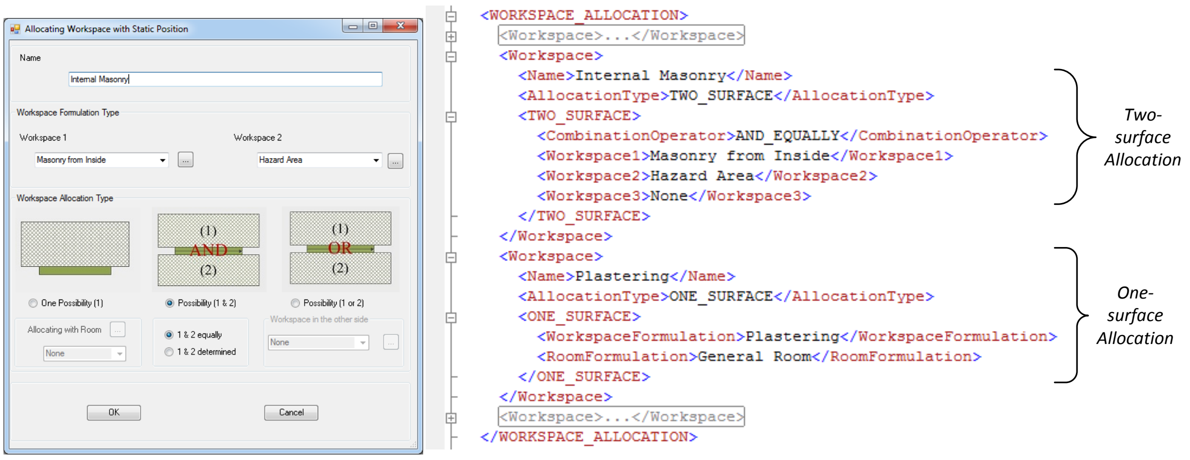
Workspace formulation is created by using the forms presented in Figure 2.4, Figure 2.5, Figure 2.6, Figure 2.10, and Figure 2.18. There, Figure 2.4 describes the form associated with the generation of a fixed-size workspace; Figure 2.5 and Figure 2.6 depicts the forms used for the generation of a variable-size workspace; Figure 2.10 presents the form used to determine whether or not a room should be used as a workspace; and Figure 2.18 illustrates the general form and the XML format of workspace formulation.



```
<WORKSPACE_TEMPLATE>
  <WORKSPACE_FORMULATION>
    <Workspace>
      <Name>Masonry from Outside</Name>
      <FormulationType>VARIABLE_AREA</FormulationType>
      <VariableArea>
        <AdditionalLength>0</AdditionalLength>
        <AdditionalWidth>1,2</AdditionalWidth>
        <MaxWidth>1,2</MaxWidth>
      </VariableArea>
    </Workspace>
    <Workspace>...</Workspace>
    <Workspace>...</Workspace>
    <Workspace>...</Workspace>
    <Workspace>...</Workspace>
    <Workspace>
      <Name>General Room</Name>
      <FormulationType>ROOM</FormulationType>
      <RoomArea>
        <maxDistance>5</maxDistance>
      </RoomArea>
    </Workspace>
  </WORKSPACE_FORMULATION>
```

Figure 2.18: User interface and XML file for workspace formulation

Workspace allocation is created by using the form presented in Figure 2.19. This allocation provides several types for the combination of workspace formulation. Depending on the type, the workspace required by an activity to complete a single product can contain either one or two possibilities of allocation.



Name: Internal Masonry
Workspace 1: Masonry from Inside
Workspace 2: Hazard Area
One Possibility (1) [✓ Possibility (1 & 2)] Possibility (1 or 2)
[✓ 1 & 2 equally] 1 & 2 determined
Allocating with Room: None
Workspace in the other side: None

```
<WORKSPACE_ALLOCATION>
  <Workspace>...</Workspace>
  <Workspace>
    <Name>Internal Masonry</Name>
    <AllocationType>TWO_SURFACE</AllocationType>
    <TWO_SURFACE>
      <CombinationOperator>AND_EQUALLY</CombinationOperator>
      <Workspace1>Masonry from Inside</Workspace1>
      <Workspace2>Hazard Area</Workspace2>
      <Workspace3>None</Workspace3>
    </TWO_SURFACE>
  </Workspace>
  <Workspace>
    <Name>Plastering</Name>
    <AllocationType>ONE_SURFACE</AllocationType>
    <ONE_SURFACE>
      <WorkspaceFormulation>Plastering</WorkspaceFormulation>
      <RoomFormulation>General Room</RoomFormulation>
    </ONE_SURFACE>
  </Workspace>
  <Workspace>...</Workspace>
</WORKSPACE_ALLOCATION>
```

Figure 2.19: User interface and XML format for creating workspace allocation

BIM for the combination of Trades, product Objects and Workspaces (bTOW) is generated by using the form presented in Figure 2.20. This BIM helps the model to identify which object and workspace type should be connected to a certain trade. Categories of product objects can be automatically listed referring to the objects available in the 3D model. Each trade can contain one or multiple object categories. However, it can contain only one type of workspace, which is referred to the template of workspace allocation.

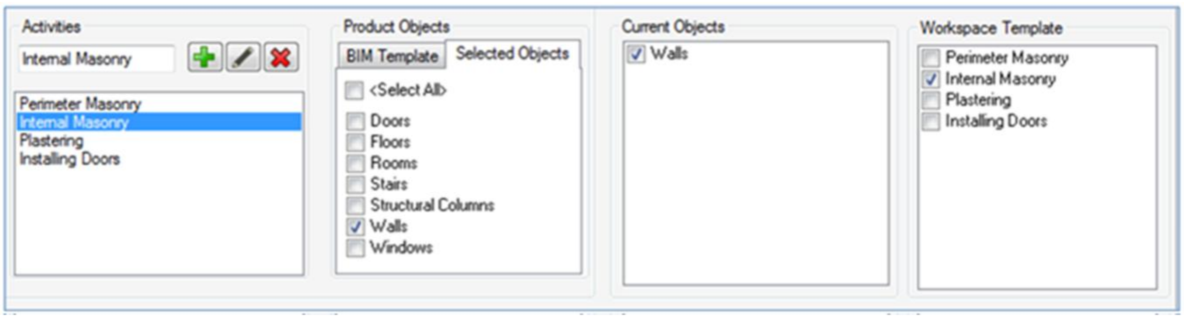
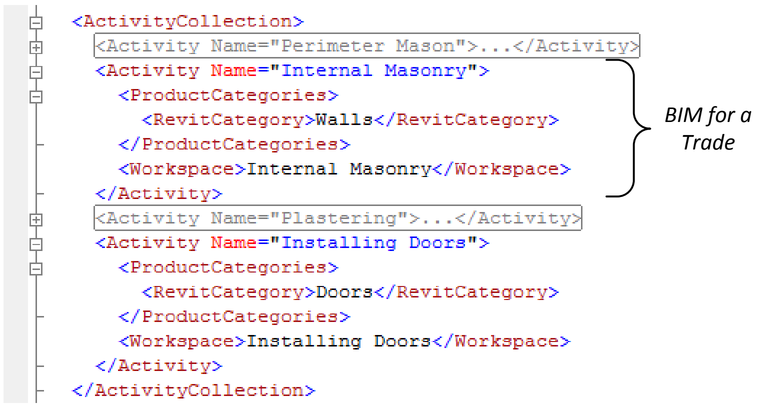


Figure 2.20: User interface for combination of trades, product objects and workspaces

Once a bTOW is generated, it can be saved as an XML file with the format structure shown in Figure 2.21 and then can be reused for other projects.

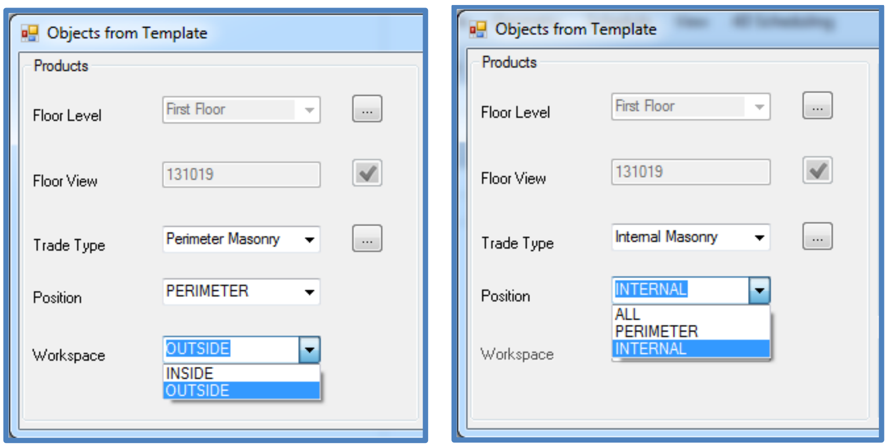


```
<ActivityCollection>
  <Activity Name="Perimeter Mason">...</Activity>
  <Activity Name="Internal Masonry">
    <ProductCategories>
      <RevitCategory>Walls</RevitCategory>
    </ProductCategories>
    <Workspace>Internal Masonry</Workspace>
  </Activity>
  <Activity Name="Plastering">...</Activity>
  <Activity Name="Installing Doors">
    <ProductCategories>
      <RevitCategory>Doors</RevitCategory>
    </ProductCategories>
    <Workspace>Installing Doors</Workspace>
  </Activity>
</ActivityCollection>
```

Figure 2.21: XML file for the combination of trades, product objects and workspaces

Assignment of bTOW into corresponding schedule activities is applied by using the form presented in Figure 2.22. Users first select an activity in the schedule to add the trade associated.

- *Trade type* in this form refers to elements of the bTOW. As mentioned above, each element of bTOW includes information of product object categories and workspace allocation in relation to the trade.
- *Position* in this form states the position of product objects in the building. For instance, if *PERIMETER* is selected, then the activity in question is just associated with the objects that stand on the perimeter of the building. Otherwise, if *INTERNAL* is selected, then it involves only objects inside. And if *ALL* is selected, then it involves all objects.
- *Workspace* in this form is used to choose the suitable option of workspace allocation. In general, the workspace required to produce perimeter objects of a building is determined firmly either as outside or inside beforehand, because this selection impacts much on the selected execution methods and supporting equipment. So if perimeter objects are involved, this parameter will be available to narrow down the combination possibilities of workspace allocation.


Objects from Template

Products

Floor Level: First Floor

Floor View: 131019

Trade Type: Perimeter Masonry

Position: PERIMETER

Workspace: OUTSIDE (INSIDE, OUTSIDE)

Objects from Template

Products

Floor Level: First Floor

Floor View: 131019

Trade Type: Internal Masonry

Position: INTERNAL (ALL, PERIMETER, INTERNAL)

Workspace

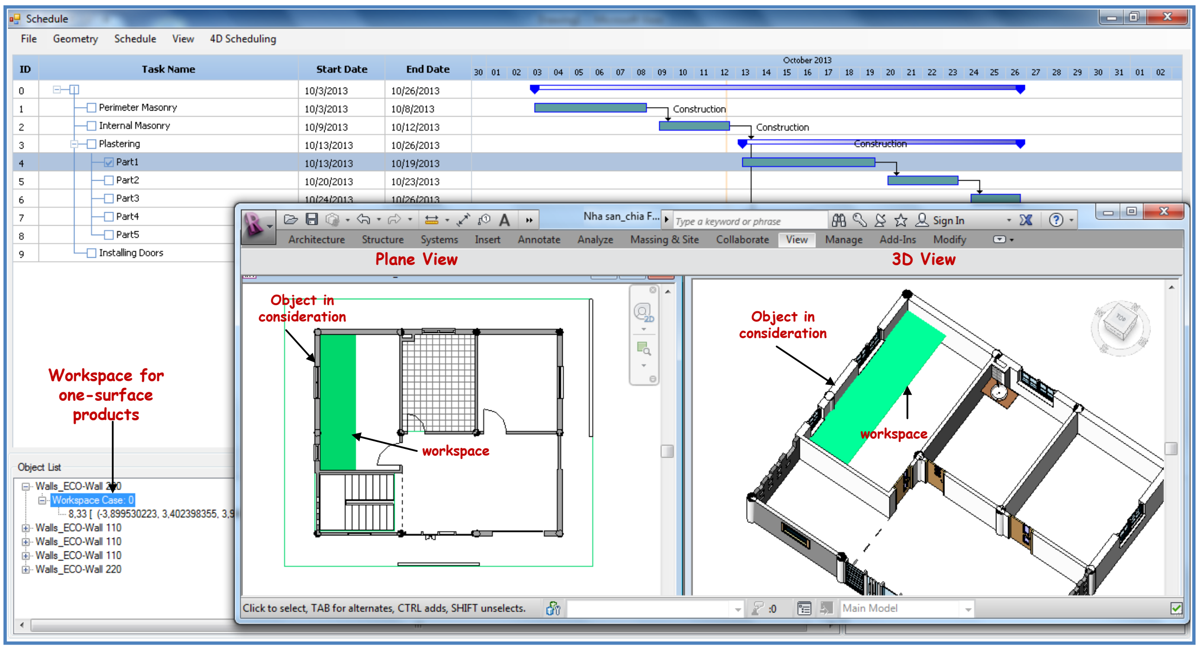
Figure 2.22: User interface for assignment of bTOW into schedule activities

2.7.4 Performance of workspace entities

After being generated, a workspace data is assigned to its associated activity and object. This implementation provides intuitional functions such as isolate, show, and hide to work with product objects and workspace components. Hence, users are able to easily evaluate the workspaces associated with individual product objects as well as check the combination of workspaces for individual activities. In addition, it also provides a function to add, modify, or remove workspace visually so that it enables users to adjust data after evaluation.

Workspace for one-surface trades is illustrated in Figure 2.23. The implementation uses room objects as references to identify the product orientation for individual activities. For example, every wall has two surfaces and the trade of painting must be applied to both of these surfaces. In order to identify which surface is carried out, the model uses a room connected with a wall. The surface in performance is the side of the object that touches the room.

Workspace for a one-surface product is its reference room if the room dimension lies within the maximum permitted value predefined in a template (Figure 2.10). In this case, the whole room is considered as a common workspace for the activity. Otherwise, the workspace would be identified as illustrated in Figure 2.8 and assigned to individual products. Figure 2.23 is an example for the latter case: when the room is considered big, then the workspace is distributed for individual products. So, the workspace for an activity is determined as the combination of its products' workspaces.



ID	Task Name	Start Date	End Date
0		10/3/2013	10/26/2013
1	Perimeter Masonry	10/3/2013	10/8/2013
2	Internal Masonry	10/9/2013	10/12/2013
3	Plastering	10/13/2013	10/26/2013
4	Part1	10/13/2013	10/19/2013
5	Part2	10/20/2013	10/23/2013
6	Part3	10/24/2013	10/26/2013
7	Part4		
8	Part5		
9	Installing Doors		

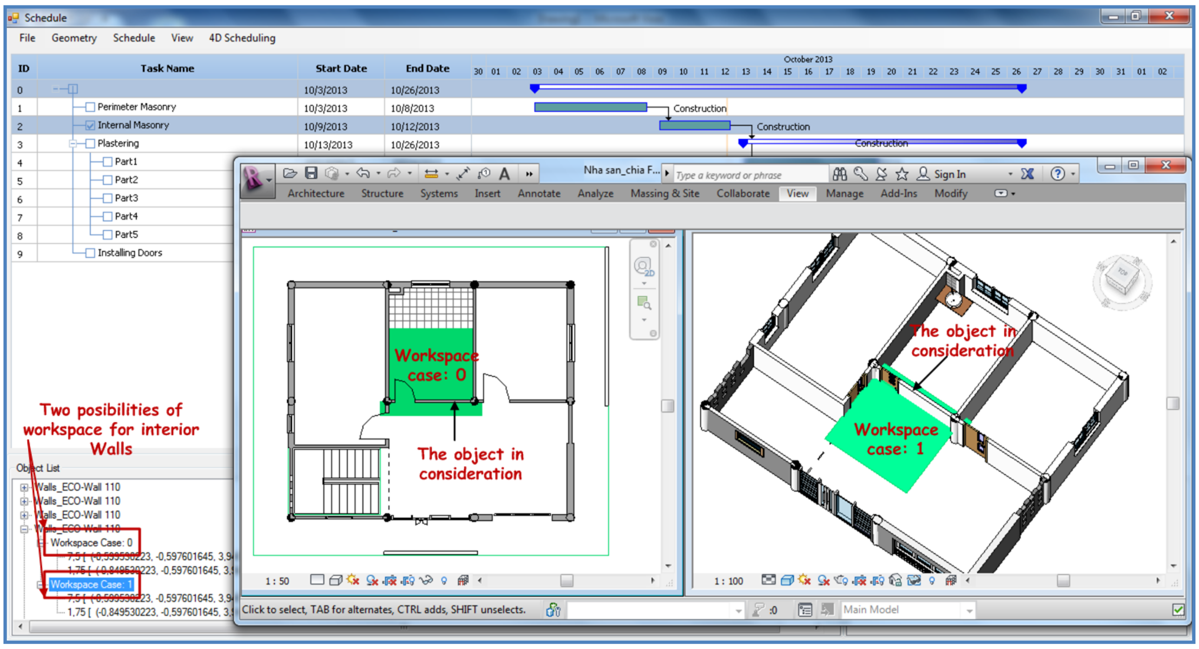
Object List
- Walls_ECO-Wall 200
  - Workspace Case: 0
    - 8,33 [(-3,899530223, 3,402398355, 3,9
- Walls_ECO-Wall 110
- Walls_ECO-Wall 110
- Walls_ECO-Wall 110
- Walls_ECO-Wall 220

Click to select, TAB for alternates, CTRL adds, SHIFT unselects.

Schedule
File Geometry Schedule View 4D Scheduling
October 2013
Construction
Architecture Structure Systems Insert Annotate Analyze Massing & Site Collaborate View Manage Add-Ins Modify
Plane View
3D View
Object in consideration
workspace
Object in consideration
workspace
Workspace for one-surface products
Main Model

Figure 2.23: Workspace for one-surface trades on a single object

Workspace for two-surface trades is illustrated in Figure 2.24 for the masonry work on interior walls, and in Figure 2.25 for the installation of doors.



ID	Task Name	Start Date	End Date
0		10/3/2013	10/26/2013
1	Perimeter Masonry	10/3/2013	10/8/2013
2	Internal Masonry	10/9/2013	10/12/2013
3	Plastering	10/13/2013	10/26/2013
4	Part1		
5	Part2		
6	Part3		
7	Part4		
8	Part5		
9	Installing Doors		

Object List
- Walls_ECO-Wall 110
- Walls_ECO-Wall 110
- Walls_ECO-Wall 110
- Walls_ECO-Wall 110
  - Workspace Case: 0
    - 7,5 [(-0,599530223, -0,597601645, 3,9
    - 1,75 [(-0,849530223, -0,597601645, 3,
  - Workspace Case: 1
    - 7,5 [(-0,599530223, -0,597601645, 3,9
    - 1,75 [(-0,849530223, -0,597601645, 3,

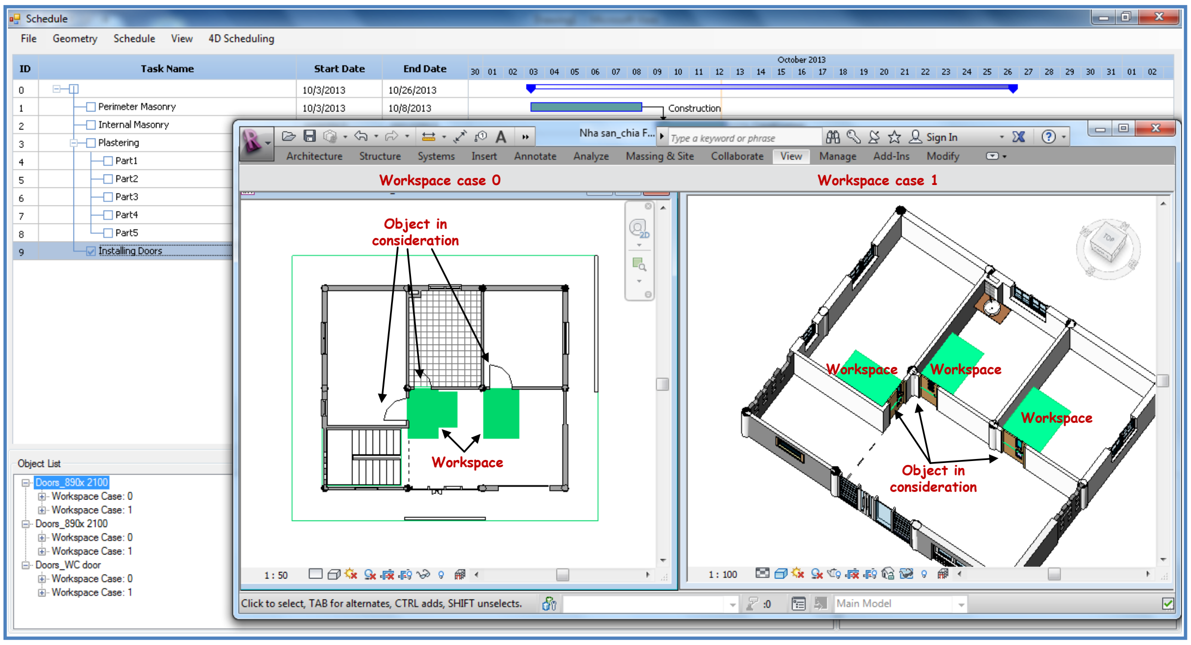
Click to select, TAB for alternates, CTRL adds, SHIFT unselects.

Schedule
File Geometry Schedule View 4D Scheduling
October 2013
Construction
Architecture Structure Systems Insert Annotate Analyze Massing & Site Collaborate View Manage Add-Ins Modify
Workspace case: 0
The object in consideration
The object in consideration
Workspace case: 1
Two posibilities of workspace for interior Walls
1 : 50
1 : 100
Main Model

Figure 2.24: Workspace for masonry on an interior wall

The trade of masonry requires workspaces located on both sides of a wall. One is the working area for crews, equipment and materials; the other side is for safety. No matter on which side of the wall the workspaces are located, their dimensions are still kept the same.

Therefore, in this case, two workspace dimensions are formulated, one is for the main working area, and the other is for the safety area. The workspace for the masonry on individual walls is then identified by a combination of these areas. There are two possibilities for locating them. This is implemented by using the operators AND_EQUALLY (Figure 2.19). Figure 2.24 shows two possibilities of workspace allocation to perform the masonry for an interior wall.



ID	Task Name	Start Date	End Date
0		10/3/2013	10/26/2013
1	Perimeter Masonry	10/3/2013	10/8/2013
2	Internal Masonry		
3	Plastering		
4	Part1		
5	Part2		
6	Part3		
7	Part4		
8	Part5		
9	Installing Doors		

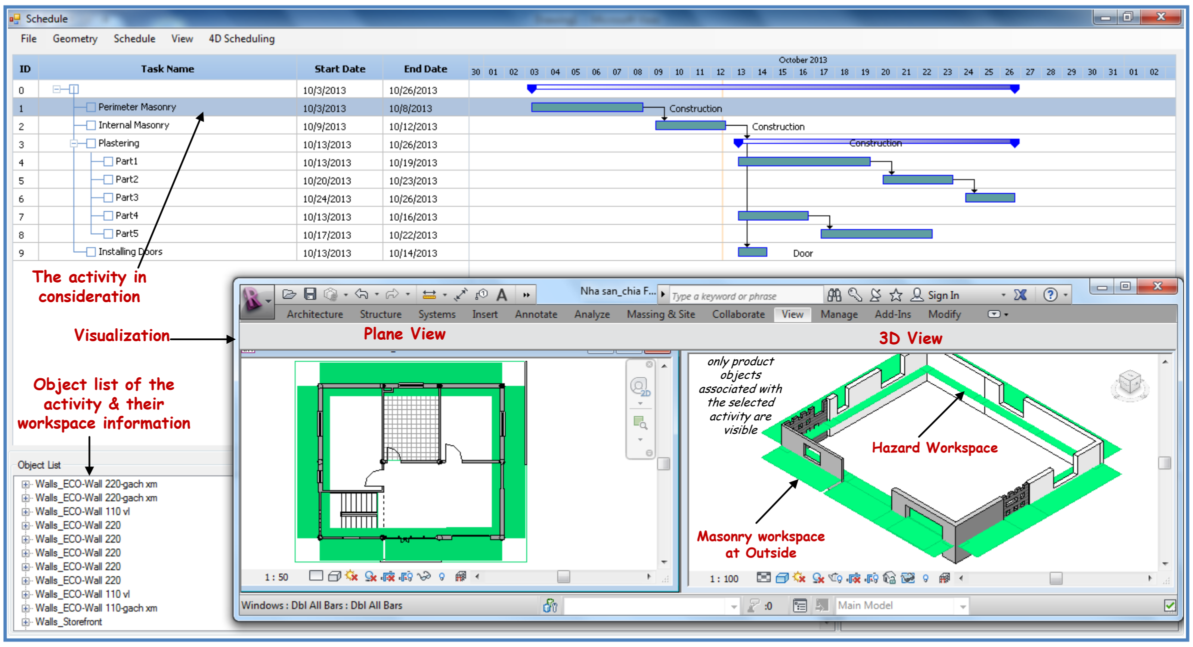
- Doors_890x 2100
  - Workspace Case: 0
  - Workspace Case: 1
- Doors_890x 2100
  - Workspace Case: 0
  - Workspace Case: 1
- Doors_WC door
  - Workspace Case: 0
  - Workspace Case: 1

Click to select, TAB for alternates, CTRL adds, SHIFT unselects.

Figure 2.25: Workspace for installing doors

Similarly, each product object associated with installing doors also has two possibilities of workspace allocation. Figure 2.25 presents the workspace for the activity of installing doors in two combinations. Notice that it does not mean that there are only two possible combinations. In this example, there are up to 8 different workspace combinations created by gathering workspace cases for installing three doors.

Workspace for product objects on the perimeter of a building also has two possibilities, which is the same as for different two-surface trades. However, users should beforehand identify either the outside or inside of the building, where the workspaces should be located, since this choice impacts the implemented execution method and depends on the capacity of supporting equipment for the project. For example, if a contractor has a glass robot, and if the inside of the building is large enough, he can choose curtain wall installation from inside of the building. But another contractor can choose the execution method from outside because he wants to use hoists or cranes to support the curtain wall installation.



ID	Task Name	Start Date	End Date
0		10/3/2013	10/26/2013
1	Perimeter Masonry	10/3/2013	10/8/2013
2	Internal Masonry	10/9/2013	10/12/2013
3	Plastering	10/13/2013	10/26/2013
4	Part1	10/13/2013	10/19/2013
5	Part2	10/20/2013	10/23/2013
6	Part3	10/24/2013	10/26/2013
7	Part4	10/13/2013	10/16/2013
8	Part5	10/17/2013	10/22/2013
9	Installing Doors	10/13/2013	10/14/2013

Schedule
File Geometry Schedule View 4D Scheduling
October 2013
Construction
Construction
Construction
Door
The activity in consideration
Visualization
Object list of the activity & their workspace information
Object List
Walls_ECO-Wall 220-gach xm
Walls_ECO-Wall 220-gach xm
Walls_ECO-Wall 110 vl
Walls_ECO-Wall 220
Walls_ECO-Wall 220
Walls_ECO-Wall 220
Walls_ECO-Wall 220
Walls_ECO-Wall 220
Walls_ECO-Wall 110 vl
Walls_ECO-Wall 110-gach xm
Walls_Storefront
Architecture Structure Systems Insert Annotate Analyze Massing & Site Collaborate View Manage Add-Ins Modify
Plane View
3D View
only product objects associated with the selected activity are visible
Hazard Workspace
Masonry workspace at Outside
1 : 50
1 : 100
Windows : Dbl All Bars : Dbl All Bars
Main Model

Figure 2.26: Outside workspace for masonry on perimeter product objects

In 4Dworkspace, users can choose the building side, either outside or inside, to perform two-surface trades on perimeter objects. Figure 2.26 presents the outside workspace for the masonry work on the perimeter walls. Here, the external workspace is the main operation area and the internal is for a safety region.

2.8 Conclusion

This chapter presented a framework and a tool to automate workspace generation and establish patterns of workspace allocation for trades in the finishing phase. According to the number of possibilities for workspace allocation, trades are divided into two categories, i.e. one-surface trades and two-surface trades. The key for the automation of workspace allocation for one-surface trades are reference rooms attached with the objects concerned. By using this key, no pre-defined orientation of workspace for one-surface trades is required. Therefore, the shortcomings of previous research related to pre-defined orientations of workspaces have been eliminated. This current approach also offers multiple possibilities of workspace allocation for two-surface trades. This support reflects the real possibilities of workspace allocation in the initial phase and provides multiples alternatives for workspace allocation to be evaluated during decision making.

3 BREAKDOWN OF SCHEDULE

"Of course, however, the living voice and the intimacy of a common life will help you more than the written word. You must go to the scene of action, first, because men put more faith in their eyes than in their ears and second, because the way is long if one follows precepts, but short and helpful, if one follows patterns"

--Seneca (Gummere 1917) --

3.1 Work breakdown structure and schedule

According to Kerzner (2009, p. 434), the first major step in the project planning process after project requirements definition is the development of the work breakdown structure (WBS). The WBS divides and subdivides a project and deliverables into smaller, more manageable work packages, whether by area, phase, responsibility, or other considerations. The level of detail for work packages varies according to the size and complexity of projects. Figure 3.1 gives an example of the highest levels of a WBS that might be used as a sample for the turn-key construction of ordinary apartment buildings.

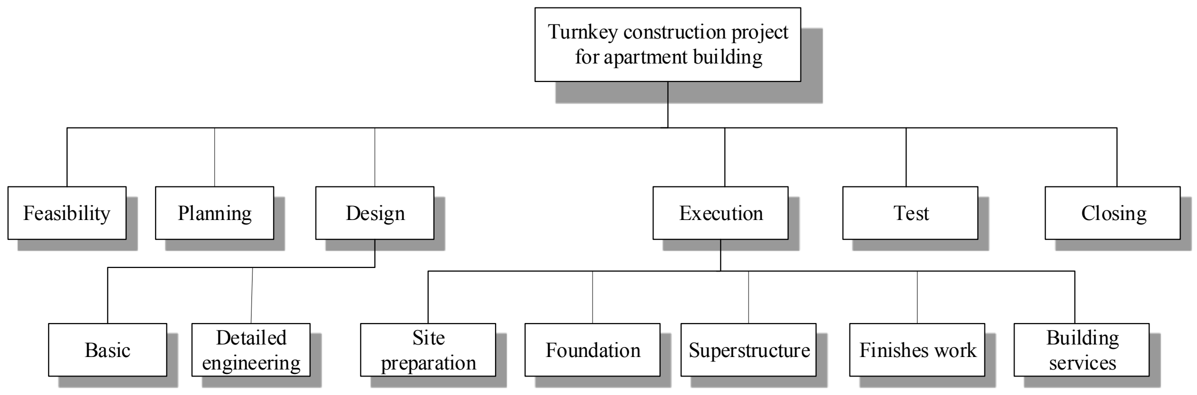


Figure 3.1: Example of a WBS for turnkey construction project of apartment buildings

On the basis of the WBS, the project manager can control and evaluate different objectives of a project, such as the responsibilities of stakeholders, schedule, cost flow, risk management, organization structure, and management coordination. Figure 3.2 shows the work breakdown structure for the control and evaluation of these objectives.

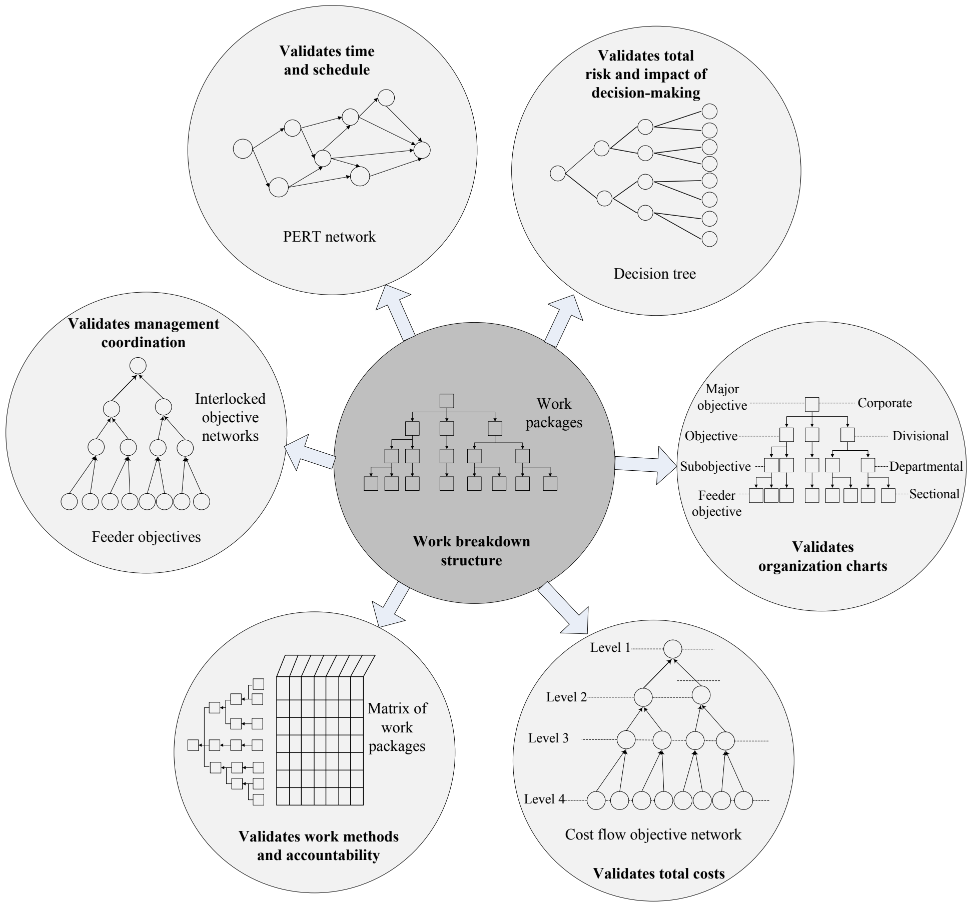


Figure 3.2: Work breakdown structure for objective control and evaluation (Kerzner 2009, p. 436)

As seen in Figure 3.2, the project time management is one of the key objectives of project management. This management is facilitated through a schedule model, which includes the project's activities, their durations, dependencies and other related information such as resources (Figure 3.3). According to Project Management Institute (2013, p. 149), the key benefit of defining activities is to break work packages down into activities that represent the work effort required to complete the work packages. The better activities are defined, the more sufficiently the project management works.

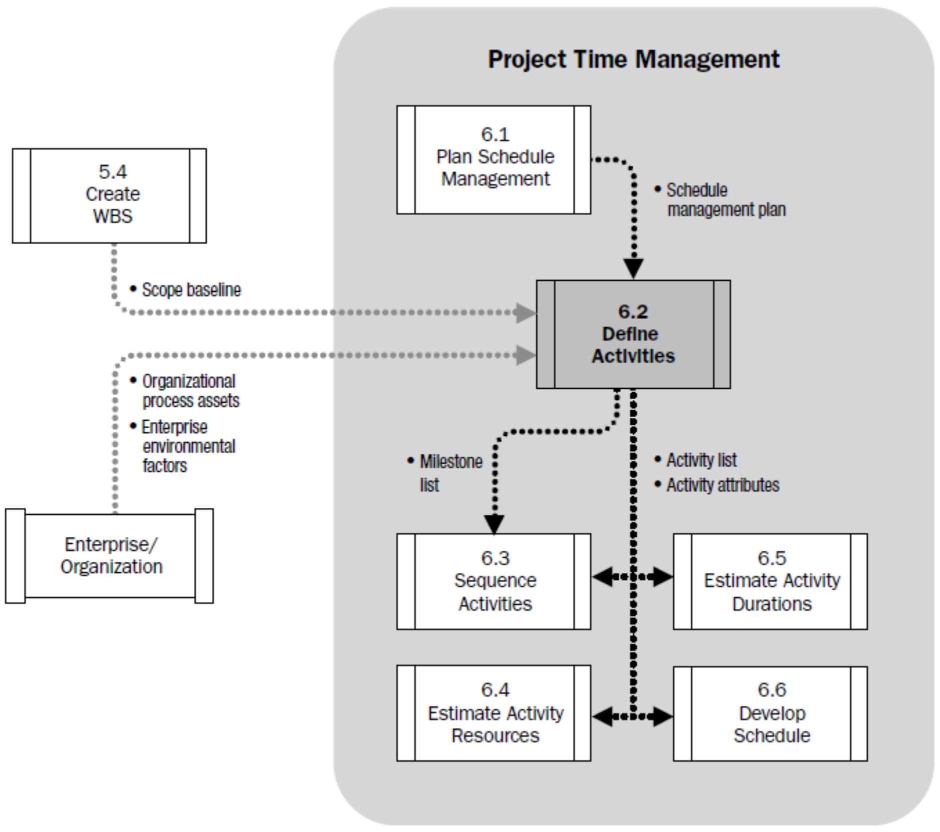


Figure 3.3: Define activities data flow diagram
(Project Management Institute 2013, p. 149)

3.2 Introduction to breakdown of schedule

Breakdown of a schedule is defined as a process including two steps: the decomposition of activities into finer and manageable sub-activities at the desired level of detail and the identification of dependencies among sub-activities based on the constraints in terms of construction techniques as well as resource distributions. The decomposition of activities can also be considered as the generation of a more detailed WBS.

In the past, several researchers have established knowledge-based systems to support this breakdown process. Examples of exemplary knowledge-based systems are the works of Echeverry (1991) and Riley and Sanvido (1995). Echeverry (1991) provided a knowledge-based system of scheduling logic. This system shows the constraints that govern the sequencing of activities, such as physical relationships among building components, trade interaction, path interference, and code regulation, and also determines the degree of flexibility associated with these constraints. Riley and Sanvido (1995) established the patterns of work-area. These patterns describe the directions and locations that units of work are completed for different trades in the finishing phase, such as linear pattern, spiral pattern, and so on. These knowledge-based systems are really valuable information on the breakdown of a schedule. However, perhaps because of the technology limitation at the research time, automation has not been regarded in these works.

The recent availability of 3D product models, which are based on building information modelling (BIM), has paved the way for improving the automation of scheduling. Several approaches have been proposed to extract information from a BIM-based drawing for making a general schedule automatically, such as the approaches of Tauscher (2011) and Kim et al. (2013). However, the automation of breaking down a schedule is still a novel area for researchers and also the urgent demand of practice.

Indeed, based on all the previous approaches analyzed in section 1.3, only the research of Akbaş (2004) has so far involved the automation of decomposing activities into finer ones. However, this automated model is not suitable for the finishing phase since its output is too detailed when an activity is only associated with a single product object. These too detailed activities are unmanageable on site and might cause a stacking of trades or much idle time for crews. Another problem of previous works on schedule generation is that they have not yet created precise and dynamic relationships between activities. Therefore, this chapter proposes a framework and develops a prototype to automate the process of breaking down a schedule, which is the core step of detailing a schedule. In particular, the scope of this chapter is limited to scheduling for the finishing work packages of building projects.

In this research, the two keys for the automation of breakdown are decomposition patterns and 4D relationships. Decomposition patterns define how to decompose 4D activities into efficiently finer tasks, while 4DRs enable these finer tasks to self-recognize their geometric constraints with others and then automatically convert these constraints to visible relationships.

3.3 Categorization of activity relationships

3.3.1 Need of relationship categorization

Relationship definitions are one key element of advanced scheduling. The updating and adjustment of schedules must be based on the transparent meaning of relationships. However, the current relationships are just able to show the results of the dependencies among activities. They cannot express the meaning within their definition. When looking at a schedule, for example, the user can just see the dependency between plastering and painting with a constraint that a plastering crew cannot start earlier than ten days after a painting crew begins. But the schedule cannot show the reasons for this relationship, whether it is a technical requirement or an organizational dependency. In various cases, this vague definition of relationships confuses the scheduler and prevents him from easily adjusting schedules.

This section identifies and categorizes the essential relationships among activities. These relationship categories can promote the automated breakdown of a schedule and make its meaning more transparent. Activity relationships are categorized according to the meaning of constraints, which these relationships reflect, namely 4D-relationships (4DRs), crew-based relationships (CBRs) and execution strategy-based relationships (EBRs). A 4DR represents a constraint between activities that is established based on the relation of their associated 3D products. A CBR reflects the effect of the limited crew capacity on scheduling activities. An EBR expresses the impact of a chosen execution strategy on sequencing relevant activities.

3.3.2 4D relationship (4DR)

Definition 3.1 *4D relationship (4DR) is a constraint type between two activities that illustrates their technical dependencies in terms of geometric structure over time.*

According to Definition 3.1, 4D relationships (4DRs) have been developed in order to present technical geometric constraints between activities. These 4D relationships are able to reflect the interaction of activities in terms of time as well as in terms of the spatial relation among their construction objects in a 3D structure. Unlike a traditional relationship, a 4DR does not have the same status in the course of a schedule. Instead, it is automatically adjustable over time to *active*, *inactive* or *released* according to which construction objects it is associated with, and also according to the completion status of these associated objects. Figure 3.4 shows an example for the status timeline of a 4DR.

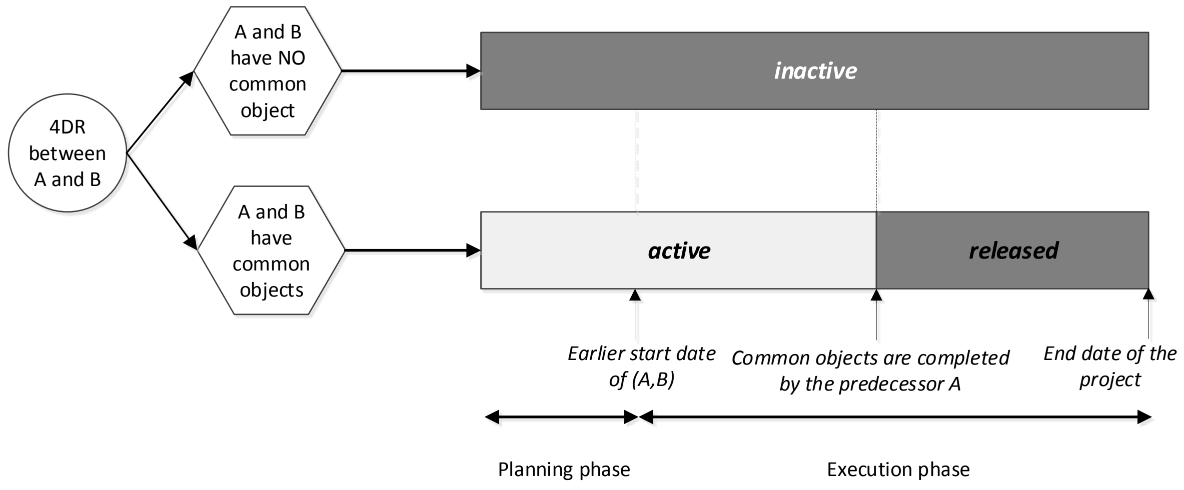

4DR between A and B
A and B have NO common object
A and B have common objects
inactive
active
released
Earlier start date of (A,B)
Common objects are completed by the predecessor A
End date of the project
Planning phase
Execution phase

Figure 3.4: Example for the timeline of a 4DR status

In 4DR modeling, a relationship is first established to reflect a constraint between a couple of trades. Afterwards, it controls the relationship between activities in accordance with their objects. A 4DR is *active* if either the products of its attached activities are associated with some common objects or their construction objects have a structural relation. Otherwise, if the construction objects of its attached activities are not related to each other at all, then the relationship is set to *inactive*. The *active* relationship is changed to *released* if the common objects or the objects in structural relation are accomplished by the predecessor activity. Thanks to this adjustable ability, 4DR can change its effect on the schedule once the objects associated with the activities or their completion status are changed and, thus, 4DR presents more dynamic working mechanism.

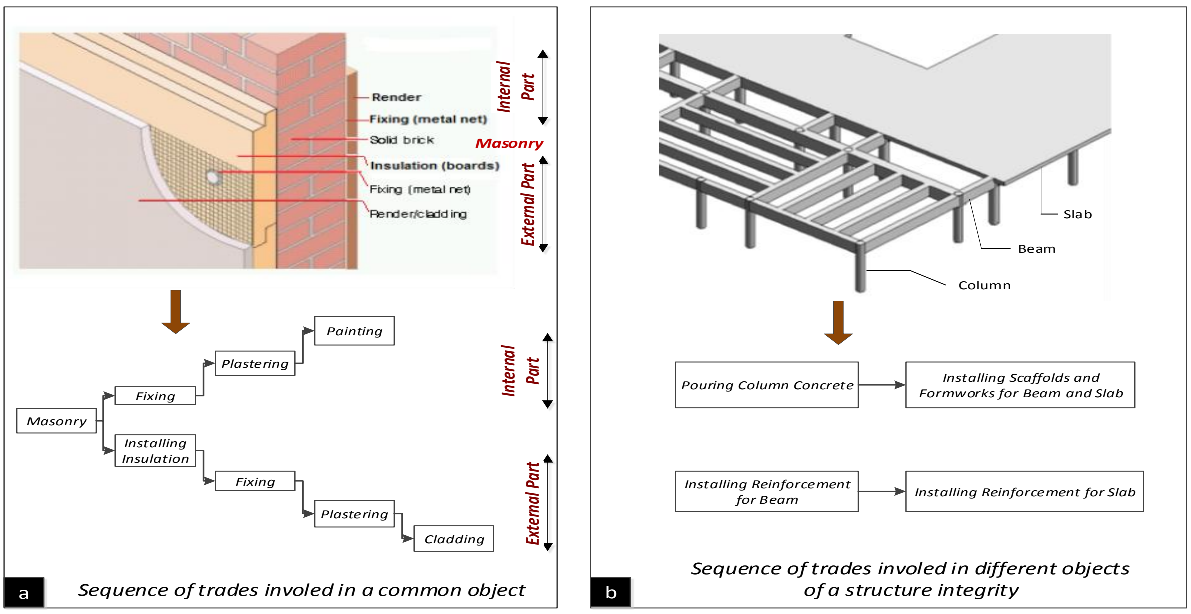

Render
Fixing (metal net)
Solid brick
Masonry
Insulation (boards)
Fixing (metal net)
Render/cladding
Internal Part
External Part
Masonry
Fixing
Plastering
Painting
Installing Insulation
Fixing
Plastering
Cladding
a
Sequence of trades involed in a common object
Slab
Beam
Column
Pouring Column Concrete
Installing Scaffolds and Formworks for Beam and Slab
Installing Reinforcement for Beam
Installing Reinforcement for Slab
b
Sequence of trades involed in different objects of a structure integrity

Figure 3.5: Sequence of trades created due to object/structure constraints

Figure 3.5 presents examples of sequencing trades to satisfy a geometric integrity. The sequence of trades is defined in order to ensure either the sequence of layers within an object or the structural stability of the building. Based on the type of object relations within 4DRs, 4DRs are further categorized into object-based dependencies (OBDs) and structure-based dependencies (SBDs). Definition 3.2 and Definition 3.3 describe the definitions of these dependencies.

Definition 3.2 *Object based dependency (OBD) is a domain of 4D relationship, which illustrates a constraint between two trades that must be conducted in a certain sequence on common objects.*

An OBD refers to a constraint between a pair of trades, which must be conducted in a specific sequence on common objects. For example, an OBD is created between the two trades of pouring concrete and installing formwork since they perform their work on the same columns. An OBD is also created to illustrate a constraint between plastering and masonry since their products are performed on the same walls.

Definition 3.3 *Structure based dependency (SBD) is a domain of 4D relationship, which illustrates a constraint between trades that are conducted on different objects, which must be created in a certain sequence to ensure a structural stability.*

In contrast to an OBD, an SBD refers to a constraint between trades that are conducted on different objects, which must be created according to a certain sequence to ensure a structural stability. For example, an SBD is created between the trades of installing doors and masonry since a door needs to be attached to a wall to be stable. For another example, an SBD is also created between the trades of installing slabs and installing beams since the slabs can only be installed if beams are already in place to ensure a structural stability.

In order to better understand the above concept and its implementation in practice, two examples are given (Figure 3.6). These examples present relationships between activities of the masonry and plastering trades. It is well known that the masonry and plastering trades have an OBD with the type of Finish-To-Start (*obd_FS*). In other words, the plastering can start on a wall just after the masonry trade on that wall has finished.

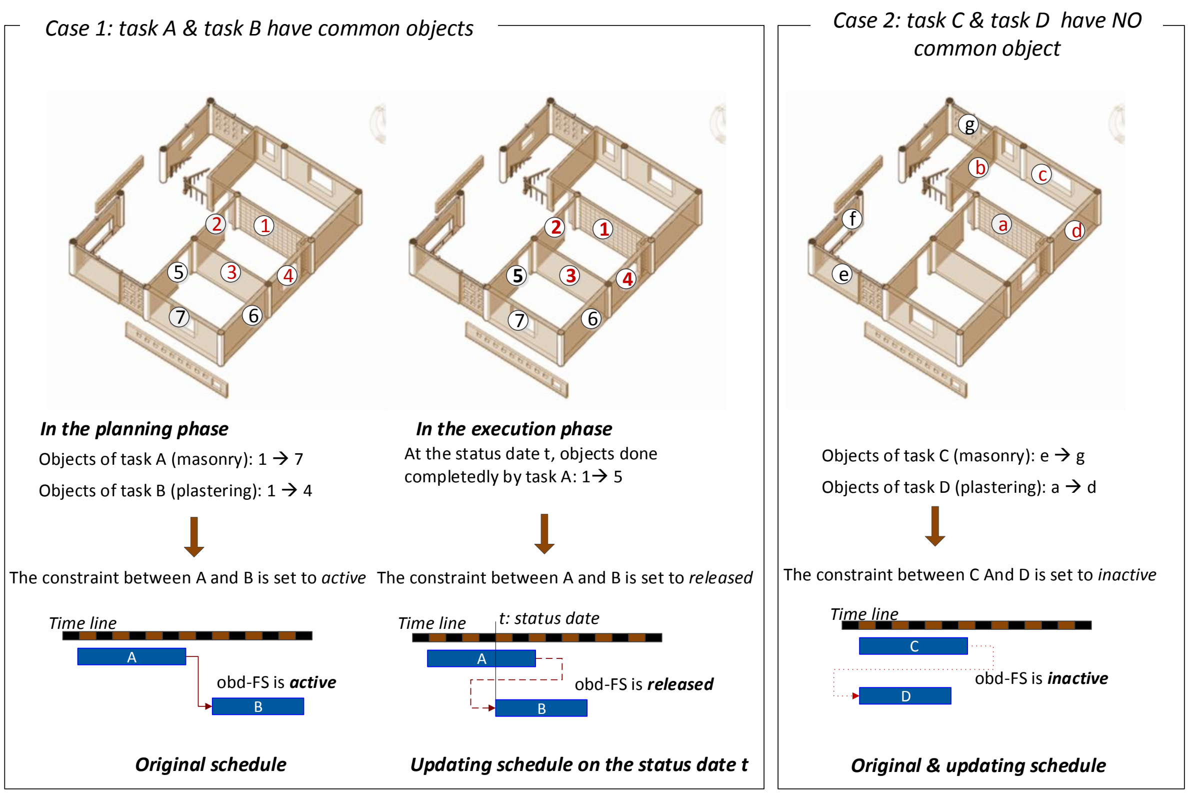

Case 1: task A & task B have common objects
Case 2: task C & task D have NO common object
1
2
3
4
5
6
7
a
b
c
d
e
f
g
In the planning phase
Objects of task A (masonry): 1 → 7
Objects of task B (plastering): 1 → 4
In the execution phase
At the status date t, objects done completedly by task A: 1→ 5
Objects of task C (masonry): e → g
Objects of task D (plastering): a → d
The constraint between A and B is set to active
The constraint between A and B is set to released
The constraint between C And D is set to inactive
Time line
t: status date
A
B
C
D
obd-FS is active
obd-FS is released
obd-FS is inactive
Original schedule
Updating schedule on the status date t
Original & updating schedule

Figure 3.6: Examples of statuses of OBDs

Figure 3.6 – Case 1 illustrates an OBD between activity *A* and activity *B*. There, *A* is associated with masonry and *B* is associated with plastering. In the planning phase activities *A* and *B* contain one or more common objects. In this example, these are objects *1*, *2*, *3* and

4. Therefore the status of this *obd_FS* is set to *active*. This means that the completion of *A* has a certain impact on *B*. In the execution phase, on an arbitrary status day *t*, the common objects of *A* and *B* are all accomplished by the activity *A*. So from this moment on, A and B can take place independently. Accordingly, the OBD between them is changed to the status *released*.

Figure 3.6– Case 2 describes an OBD between activity *C* and activity *D*. There, *C* is associated with masonry and *D* is associated with plastering. In this example, *C* and *D* involve two different parts of the building. C is performed on the objects *e*, *f* and *g,* whereas *D* is conducted on the objects *a*, *b*, *c*, and *d*. So these two activities in practice can be carried out independently from each other. Accordingly, the *obd_FS* between them is kept *inactive* in the course of the schedule. This means that although *C* and *D* are technically assigned with an *obd_FS* relationship, the completion of *C* does not have any influence on *D* because they are associated with different objects.

With a self-adjustable status according to the objects of attached activities, 4DRs are highly compatible with the automation of breaking down a schedule. Prior to the breakdown process, the rough activities, which are associated with individual trades, are assigned with 4DRs. During the breakdown process, the newborn sub-activities inherit the relationships from their summary activities. Finally, the real working status of an inherited relationship is automatically adjusted in accordance with the actual construction objects of the sub-activities. Sections 3.5.4 and 3.6.5 give more detail about the application of 4DRs in breaking down a schedule.

3.3.3 Crew based relationship (CBR)

Definition 3.4 *Crew based relationship (CBR) is a constraint type between two activities that is created in order to reflect the availability of trade crews or major devices of a project.*

According to Definition 3.4, CBRs must work as a traditional Finish-To-Start (FS) dependency since a crew or device can only conduct an activity at a given time. In this research CBRs are used as temporary relationships in order to roughly estimate durations of summary activities after the activity decomposition. These relationships are established based on the limitation of crews or major devices on site, such as cranes and hoists (Figure 3.7). Thanks to this consideration, a planner has an overview of how long a detailed schedule takes before other factors, such as workspace capacity, are taken into account. As mentioned above, this kind of relationship is temporary to roughly estimate duration. Thus, these relationships are released before launching a searching process for optimal schedules considering workspace and crew availability.

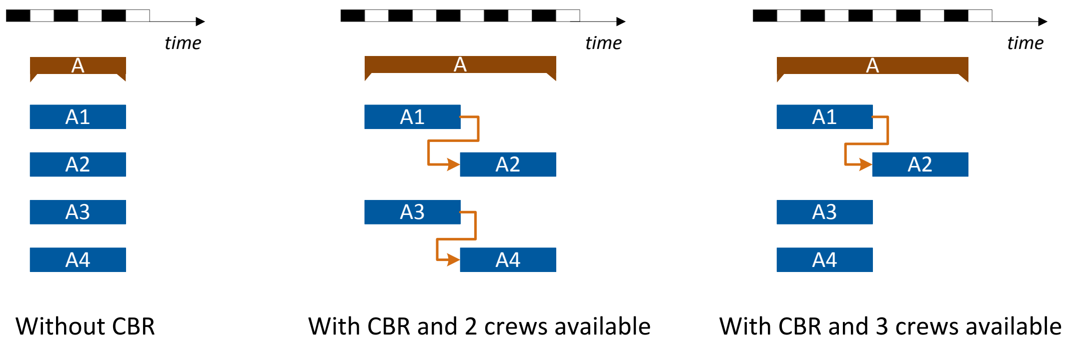

time
A
A1
A2
A3
A4
Without CBR
time
A
A1
A2
A3
A4
With CBR and 2 crews available
time
A
A1
A2
A3
A4
With CBR and 3 crews available

Figure 3.7: Examples about using CBR

3.3.4 Execution strategy based relationship (EBR)

After the decomposition of activities, a question can be raised here. In which order should the sub-activities be arranged? May every sub-activity have the same role, which makes the cost and duration of a schedule still remain constant regardless in which order they are executed? The answer depends on the properties of the products. For example, during the installation of doors, it is not important to put them in an order because the products are discrete elements and they have no relation to each other. Thus, the sequence of installing them is inconsequential. In a different scenario, installing façade should be arranged in a circular sequence so that cranes work more efficiently. In this model, execution strategy based relationship (EBR) is presented to illustrate the priority of activities in execution. EBRs work as traditional relationships and can be used as either hard constraints or soft constraints. Definition 3.5 presents the definition of an EBR.

Definition 3.5 *Execution strategy based relationship (EBR) is a constraint type between two activities that illustrates the priority hierarchy of activities according to execution strategies.*

3.4 Trade breakdown

3.4.1 Overview of trade breakdown

The breakdown of a construction project into activities is performed by combining two steps: the delegation of responsibility (trade contractors, work crews) and the work areas (Echeverry 1991, p. 29). Based on responsibilities, the project could be decomposed into earth works, structural works, or finishing works. At the finer level of detail, it could also be further decomposed into excavation works, reinforcement works, plastering or painting works. Based on work areas, however, the project could be broken down into the works for underground, the first floor, the second floor, and so forth. Evenly, it could be further decomposed into various rooms and regions on each floor.

This approach assumes that a project is already decomposed, based on responsibility, into various activities associated with typical trades. Thus, this approach focuses on establishing a set of decomposition patterns for individual trades. The imperative step of this section is to further break down these activities based on geometry. In this research, the geometry-oriented breakdown of an individual activity associated with a typical trade is called a trade breakdown (Definition 3.6).

Definition 3.6 *Trade breakdown is a geometry-oriented breakdown of an individual activity associated with a typical trade.*

Depending on the usage of schedules and properties of a building, different levels of detail for decomposition will be implemented. In this research, the common characteristics of execution strategies of trades are considered to subdivide the decomposition process into two levels: the rough decomposition and the detailed decomposition. The rough decomposition is the first image which comes to the schedulers whenever they start to decompose a building into sub-systems, such as multiple floors or multiple vertical surfaces. Afterwards, the detailed decomposition is performed to further subdivide the sub-systems into groups of components when the schedulers require a greater level of detail, such as rooms or regions. Figure 3.8 and Figure 3.10 illustrate the above definitions of rough and detailed levels of decomposition.

In this section, the patterns for the rough and detailed decompositions are sequentially established. On one hand, this approach focuses on the general definition of patterns and their application scope, which is partially consulted with the previous works of Echeverry (1991) and Riley and Sanvido (1995). On the other hand, it concentrates more on working with patterns in terms of automation, in which the relationships between the newborn sub-activities are considered and the complexity of building geometry in practice is acknowledged to propose methods to handle. Finally, algorithms for the pattern application in breaking down an activity associated with typical trades will be described in detail.

3.4.2 Rough decomposition patterns

✦ ***Floor-based decomposition***

In a multiple-story building, the working areas are typically isolated from each other by floors. The work on each floor is quite independent to the other floors in terms of mobilization space. Accordingly, the building is usually subdivided into floors for the first level of detail. This decomposition pattern is called floor-based decomposition and illustrated in Figure 3.8.

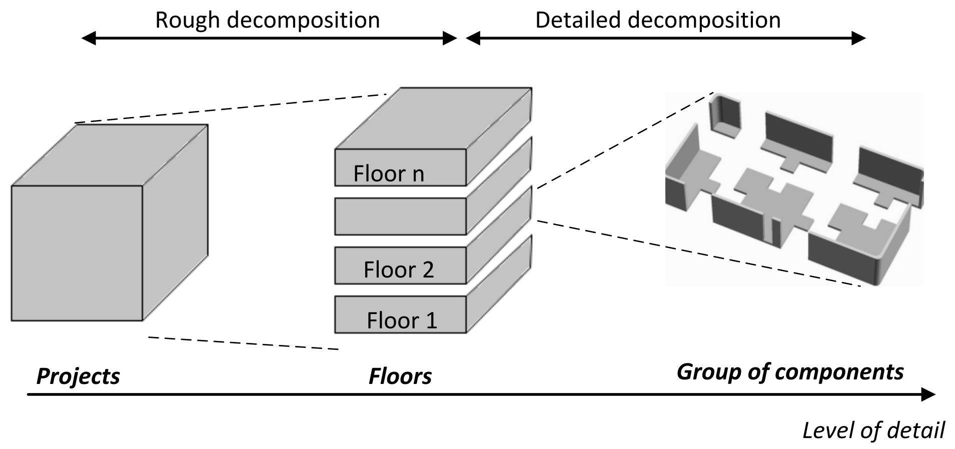


Figure 3.8: Floor-based decomposition

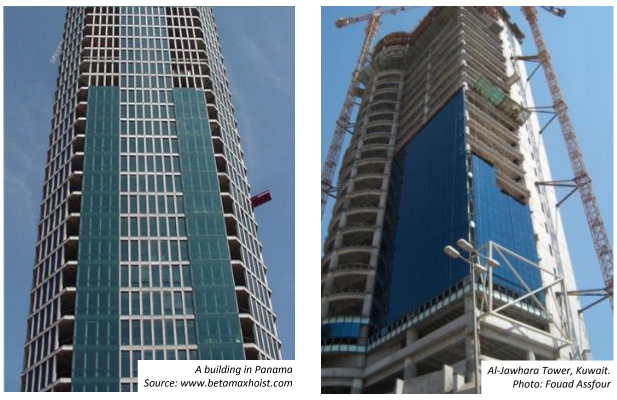
BIM-based 3D models provide benefits to the automation of such decomposition. In a BIM-based 3D model, every component contains in itself the information of the floor level it belongs to. This is very convenient to distribute the components into tasks associated with their corresponding floor.

After the decomposition of activities, the relationships between sub-activities must be established. In the fishing phase, the tasks associated with the same trade could take place independently on different floors in terms of construction techniques. No technical relationships are established between them. Therefore, the sequence of their performance is mainly defined by the constraints with activities associated with other trades. For example, the sequence of the painting walls on different floors is normally defined by the sequence of the trade masonry on those floors. This kind of relationship would be inherited from their direct summary activities. In addition, based on the limitation of resources such as crew capacity, the floors are normally performed by adopting a rolling wave strategy. Thus they are executed one after another. The EBRs or CBRs could be assigned to sub-activities in order to illustrate this strategy. Generally speaking, assignments of these relationships in this pattern follow either a bottom-up or a top-down sequence. This means that the activity associated with the floor (n) would be the predecessor of the one in charge of the floor (n+1) in the bottom-up sequence. In contrast, the activity associated with the floor (n+1) would be the predecessor of the one in charge of the floor (n) in the top-down sequence.

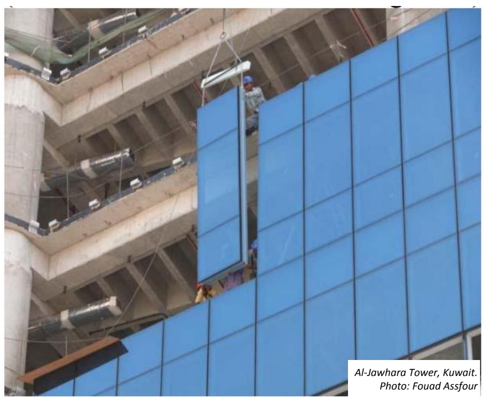
A floor-based decomposition pattern can be applied for almost all trades. The following are examples: the trades involved in the superstructure such as scaffolding, concrete, steel frame erection, etc.; the trades involved in interior construction and finishing, such as masonry, installing drywalls, MEP, plastering, painting, ceiling finishing, tiling; and also the trades involved in external finishing such as the installation of curtain walls, external wall tiling.

✦ ***Façade-based decomposition***

The trades associated with the external finishing of a building such as the installation of curtain walls, which require machines outside, e.g. a crane or hoist, to be assembled, are not isolated by floors. Nevertheless, these trades, in several cases, definitely could be decomposed by the floors as presented above. However, in various other cases, they could not follow the floor-based pattern because their work must depend on the positions of machines and material storage. For example, a hoist position might support well the installation of panels on one building surface, but it does not for other building surfaces. Therefore, the workflow should be organized according to the machine positions in order to reduce the number of machine changes. In addition, the variety of component types along the floor perimeter also makes the subdivision based on floors unsuitable. It would be better to assemble panel groups in a vertical direction. Thus, in such cases, a building should be subdivided into various exterior vertical surfaces instead of various floors. Figure 3.9 illustrates examples of installing curtain walls from one exterior vertical surface to another.



The assembly strategy of curtain walls depending on the machine position and variety of panel types



Close-up of panel assembly

Figure 3.9: Subdivision of a building into various faces

In another situation, various trades are required to ensure the continuity of performance and the consistency of color throughout individual building façades. So the performance sequence of this trade should be done from one vertical surface to another. This is the case of the trades associated with façade, e.g. exterior plastering, exterior painting or decorating external walls. In these cases, an activity should also be subdivided into individual vertical surfaces of the building instead of floors. This pattern is called façade-based decomposition. Figure 3.10 describes the process of façade-based decomposition of a building from rough to detailed level.

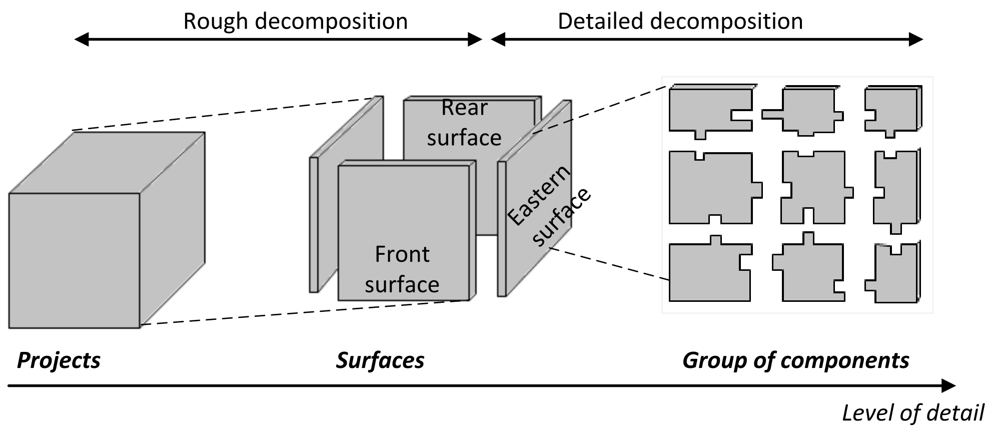


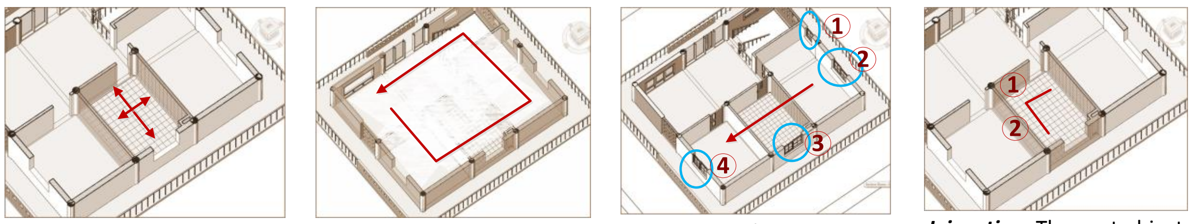
Figure 3.10: Façade-based decomposition

The decomposition for an exterior wall closure must pass through two steps. The first step is extracting the external components from all building components. The second step is subdividing these external components into different groups. The first step, named the extraction process, will be implemented floor by floor. On each floor, the algorithm illustrated in Figure 3.12 will be applied to get the perimeter components. After this process, all external components of a building are arranged in counter-clockwise order for each floor. The second step follows the bottom-up sequence. Beginning with a point at a certain corner of the bottom floor, the component collection for a building façade will take the adjacent component and the directly above component if their type is the same and their position is aligned to the previous. The process will be propagated until none of the components satisfies the above conditions. At this time, if there is any component remaining from the generated building façades, then a new façade will be created and assigned with the bottom component of the remaining group; the collection of a building façade spreads out as the same as the collection process described above. Otherwise, if no components remain, the process will end. The result returned are several groups of individual building façades, each of which contains a strip of typical components.

In order to make the movement of supporting tools efficient, the execution sequence of building façades is normally chosen as a circular sequence. Therefore, the relationships between the sub-activities in this pattern are assigned with EBRs according to a circular sequence.

3.4.3 Detailed decomposition patterns

According to execution strategies often applied in the finishing phase, four detailed decomposition patterns have been developed in this research. They are called room pattern, circular pattern, linear pattern and L-junction pattern. Figure 3.11 provides an overview of these patterns. The following describes them in detail.



Room: A sub-activity is in charge of all objects around a room

Circular direction: Objects are selected for an activity following a circular sequence

Linear: Objects are selected for a sub-activity following a linear direction

L-junction: The next object chosen is the one adjacent but not aligned to one of the objects chosen before

Figure 3.11: Detailed decomposition patterns

✦ *Room decomposition pattern*

The fact is that various trades, such as plastering, painting and ceiling finishing take place in an enclosed area or rather a room, which is separated from others by walls. Then they occupy a room completely to work. They move to other locations only after all of their associated objects around the room are accomplished. This ensures the fewest occurrences for changing and relocating equipment as well as temporary storages on site.

The room decomposition pattern imitates this case. According to this pattern, an activity is decomposed into sub-activities, each of which is associated with an individual room. Thus, a sub-activity is accordingly in charge of all objects around and inside its related room. Since rooms are enclosed and independent, the execution sequence of rooms is considered randomly. That means sub-activities of the same trade created with this decomposition pattern have no relation to each other. The relationships between them, if any, are normally CBRs to reflect the limited availability of crew numbers.

Algorithm 3.1 Room- Decomposition-Pattern

```
   Input:  $A$: an activity to decompose
           $d$: the desired duration of sub-activities
   Output: $S$: a list of sub-activities
 1 $S=\{\}$;
 2 $R_s = \{o \in A.objects : o \text{ is Room}\}$;
 3 foreach Room $r \in R_s$ do
 4     Create a new activity $a$;
 5     $a.objects = \{o \in A.objects : o \ is \ on \ r.boundary\}$;
 6     $A.objects = A.objects \setminus a.objects$;
 7     if $r$ is small then
 8         $S = S \cup \{a\}$;
 9     else if $A.secondaryPattern$ = LINEAR then
10         $S = S \cup$ Linear-Decomposition-Pattern$(a, d)$;
11     else if $A.secondaryPattern$ = ROUND then
12         $S = S \cup$ Round-Decomposition-Pattern$(a, d)$;
13 return $S$;
```

In case a room is so big that more than one trade can be carried out at the same time, this room can be further decomposed into smaller parts by using linear or circular patterns, which are presented in the following sections. The signal, which helps to recognize whether using a whole room for one activity or further breaking down it into sub-activities, was already discussed in section 2.4.3.

Algorithm 3.1 describes the method of decomposing an activity by using the room pattern. The result of this process is a list of newborn tasks and their objects accordingly. This pattern can be applied for one-surface trades such as plastering, painting, roughing MEP for brick walls, ceiling finishing and tiling.

✦ *Circular decomposition pattern*

In the circular pattern, objects are first arranged in an either clockwise or counter-clockwise round order depending on the selected execution strategy. They will then be collected into groups and assigned to sub-activities according to that sequence.

Figure 3.12 illustrates the algorithm to arrange objects in counter-clockwise order. If a clockwise order is chosen for the execution strategy instead of counter-clockwise, the result can be obtained by reversing the order of the output received above.

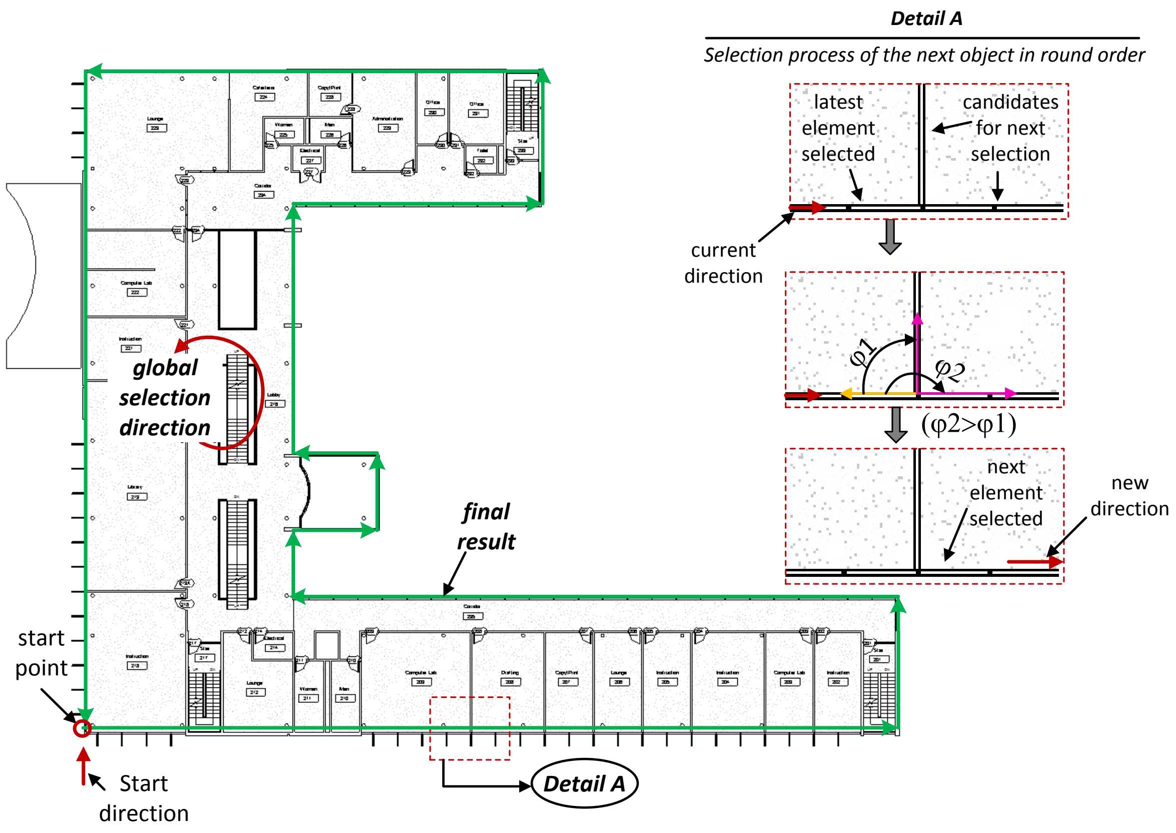


Figure 3.12: Illustration of sorting objects in a circular order of counter-clockwise

The process of a person searching for the outer path around a path matrix is used to briefly illustrate the idea of this algorithm. The searching process begins by putting that person at the bottom corner (respectively with the start point in Figure 3.12) of the matrix and looking straight ahead (respectively with the start direction with the red arrow). Once he sees two or more paths ahead, he will choose the path that is on the most right to his side (Figure 3.12 - Detail A), and then proceeds along that path. The searching process continues until either: 1) the path chosen was already passed along before (in the case of an enclosed route); or 2) no paths stand ahead anymore (in the case of an open route). After finishing his circular route, the sequence of the paths passed along will be achieved (the green arrows). In this illustration, a path denotes an object; and its direction denotes the length axis of that object. The advantage of this algorithm is that it does not only work for sorting out objects, but can also be used for extracting perimeter elements from a certain object group. So in this research, besides arranging objects in a circular order, this algorithm is applied as well for getting perimeter walls, which are important to identify objects associated with façade trades.

Algorithm 3.2 Circular- Decomposition-Pattern

```
Input:  $A$: an activity to decompose
        $d$: the desired duration of sub-activities
        $n$: the number of crews available
Output: $S$: a list of sub-activities
Sorting-in-Counter-Clockwise($A.objects$);
$S$={};
Create a new activity $a$;
foreach Object $o \in A.objects$ do
    if $a.Duration < d$ then
        $a.objects = a.objects \cup \{o\}$;
    else
        $S = S \cup \{a\}$;
        $Activity\ preTask = a$;
        $a = newActivity()$;
        $a.objects = a.objects \cup \{o\}$;
        Create a new predecessor $p$;
        $p.preTask = preTask$;
        $p.constraintType = EBR$;
        $a.predecessor = p$;
if $a.Duration > 0$ then
    $S = S \cup \{a\}$;
Remove all objects of $A$;
Relationship-Adjustment($S, n$); /* see Algorithm 3.3 */
return $S$;
```

Back to the circular decomposition pattern, Algorithm 3.2 describes the breakdown of an activity based on this pattern in detail. In this algorithm, the sub-activities born with this decomposition pattern are connected by EBRs since their sequence should follow a circular order. This ensures the efficient working of machines, such as a crane for installing façade elements, and also satisfies technical requirements in terms of continuity. In Algorithm 3.2, at first EBRs are assigned with an assumption that the trade is carried out with only one crew. Therefore, a sub-task is assigned with a constraint of EBR to the sub-task born directly prior to it. Afterwards, the number of crews available is considered to further sequence the sub-tasks by using Algorithm 3.3. In this algorithm, if there is more than one crew for that parent trade, then several relationships can be released so that several sub-tasks can take place concurrently. For example, if a parent activity is divided into 6 sub-tasks with two crews working for that trade, then the relationship between the sub-task #3 and #4 should be released. This release point ensures the continuous working area for each crew and avoids interference among different crews.

Algorithm 3.3 Relationship-Adjustment

```
   Input:  $S$: a list of activities
           $n$: the number of crews available
   Output: $S$: a list of activities with adjusted relationships
1  int $dA = S.Count \div n$;
2  for int $k = 1; k < n; k++$ do
3      Release the relationship between $S\{k * dA\}$ and $S\{k * dA + 1\}$;
4  return $S$;
```

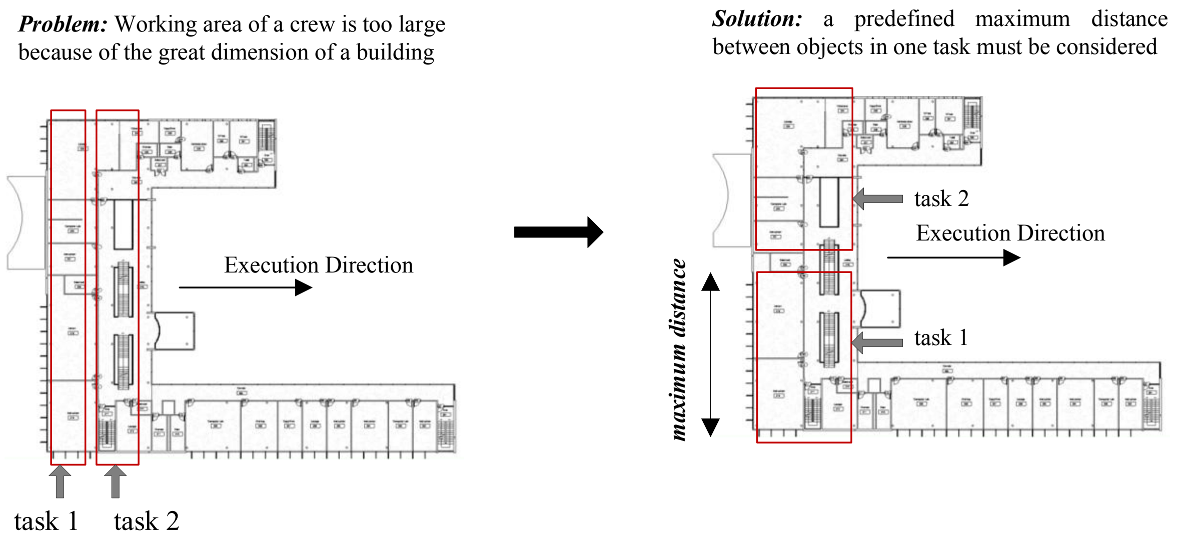
The circular pattern can be applied for the masonry trade if its objects are standing around the building perimeter. It can also be used for the trade related to façade construction. It can even be implemented for interior trades such as plastering, painting and wall tiling in case their objects are around a big room, which is in need of being further decomposed after using the room pattern.

✦ *Linear decomposition pattern*

Linear execution is a way to perform construction components along a linear direction. Therefore decomposition with a linear pattern is a kind of arrangement of objects in the order of a particular direction. It considers such an order as a priority hierarchy to select them into sub-activities. The objects of a sub-activity are close to each other and can be considered as a zone. This reduces the need of crew to move during work time and makes the project easier to be controlled.

This approach concerns two characteristics existing in the reality of large projects, which might avoid unfeasible solutions if only a coordinate hierarchy is regarded during decomposition. The first characteristic is that the building might be too large in two directions. So if one direction is the only factor considered to arrange objects in a sequence, the result could be a group of objects which are very far away from each other (Figure 3.13).

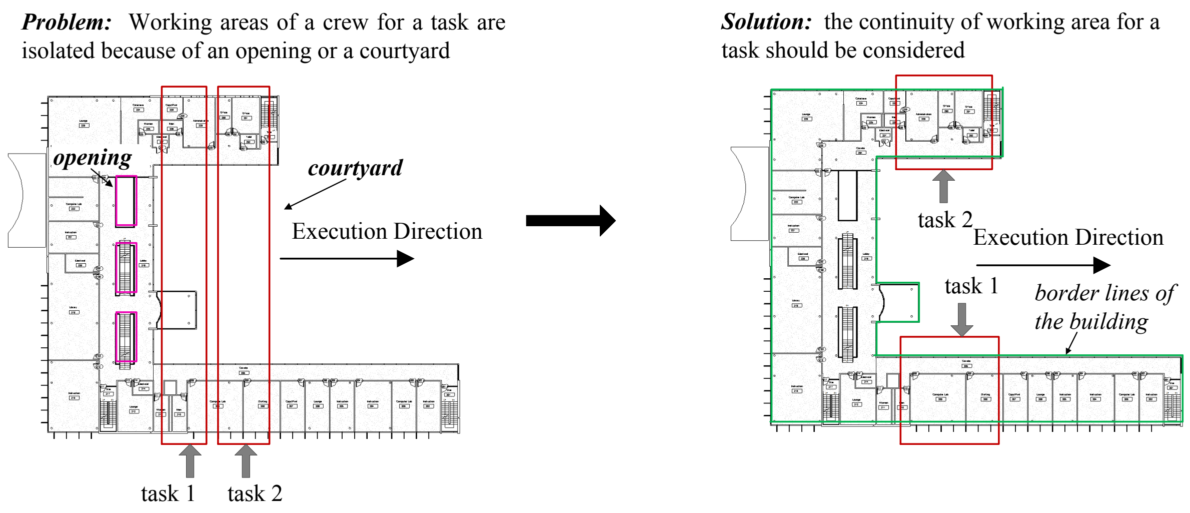
As consequence, the temporary storage of material on site is difficult to be sufficiently located and the associated crews must move a lot during working.



Problem: Working area of a crew is too large because of the great dimension of a building
Execution Direction
task 1
task 2
Solution: a predefined maximum distance between objects in one task must be considered
task 2
Execution Direction
task 1
maximum distance

Figure 3.13: Problem and solution for a large building

In order to deal with this problem, a pre-defined maximum distance should be considered. Before assigning an object to a task, it will be checked whether the distance between it and any other object existing in the task is within this acceptable distance. If yes, it will be added into the object group of the task. Otherwise, another object will be considered for the sub-activity in question.



Problem: Working areas of a crew for a task are isolated because of an opening or a courtyard
opening
courtyard
Execution Direction
task 1
task 2
Solution: the continuity of working area for a task should be considered
task 2
Execution Direction
task 1
border lines of the building

Figure 3.14: Problem and solution for a large opening and courtyard

The second characteristic taken into account is the existence of a large opening or a courtyard in a building (Figure 3.14). With this kind of separation, a crew has to move inefficiently during work if the working areas of a task are located on different sides of an opening or a courtyard. Accordingly, the productivity of crews decreases considerably. In order to solve this problem, large openings and border lines of a building will be concerned during distributing objects into sub-activities. A given object can join a sub-activity if it is not isolated from the objects existing in this considered sub-activity.

Algorithm 3.4 describes the method to decompose a task into finer sub-activities by using the linear pattern. Since sub-activities born with this pattern have no technical constraints to each other, this algorithm uses CBRs to reflect the relationships between sub-activities. Algorithm 3.3 must afterwards be applied to consider the availability of crews. This pattern can be applied to the trade of installing windows and doors. It can also be used for partial trades for drywalls such as roughing-in plumbing, roughing-in electrical, and installing drywall boards.

Algorithm 3.4 Linear-Decomposition-Pattern

```
Input:  $A$: an activity to decompose
        $d$: the desired duration of sub-activities
Output: $S$: a list of sub-activities
Sorting $A.objects$ according to the selected direction;
$S$={};
Create a new activity $a$;
$S = S \cup \{a\}$;
$o = A.objects[0]$;
while $A.objects.Count > 0$ do
    if $a.Duration < d$ then
        if $o$ is neither SO-FAR nor ISOLATED from any $obj \in a.objects$ then
            Move $o$ from $A$ to $a$;
            $o = A.objects[0]$;
        else
            $o = o.Next()$; /* get the object next to $o$ in $A.objects$ */
    else
        $a = new$ Activity();
        if $S.Count > 0$ then
            Create a new predecessor $p$;
            $p.preTask = S[S.Count - 1]$;
            $p.constraintType = CBR$;
            $a.predecessor = p$; /* assign a predecessor to $a$ */
        $S = S \cup \{a\}$;
        $o = A.objects[0]$;
return $S$;
```

✦ ***L-junction decomposition pattern***

The L-junction pattern focuses on generating a group of components which is self-stable by being formed in the L-junction shape. The collection of components for a task starts with a certain object. The next to be chosen is the one that is adjacent but not aligned to objects chosen before. If there are several objects satisfying that condition, then the most suitable object will be prioritized, whose workspace can be combined with previous workspaces of the task to form a minimal space. This ensures the minimal workspace required for a task. The collection of components for a task will end when its duration reaches the predefined duration for newborn tasks.

After collecting construction objects for a task, another new task will be created by repeating the above collection process for the remaining objects. The first object for a new task should be chosen following the hierarchical priorities: 1) its ability to join with a certain object, which has already been selected for previous tasks, for forming an L-junction shape, and 2) a certain minimum of overlap between its workspace and previous tasks. These priorities ensure the stability of the new object as being constructed as well as ensure the independence of the new task from others in terms of workspace.

Algorithm 3.5 L-junction-Decomposition-Pattern

```
Input:  $A$: an activity to decompose
        $d$: the desired duration of sub-activities
Output: $S$: a list of sub-activities
$S$={};
Create a new activity $a$;
$S = S \cup \{a\}$;
while $A.objects.Count > 0$ do
    if $a.Duration < d$ then
        $L = \{e \in A.objects : e$ is L-junction with a certain $obj \in a.objects\}$;
        if $L = \{\}$ then
            $L = \{A.objects\}$;
        $o = \{e \in L : \{e.workspace \cup a.workspace\}$ is minimal $\}$;
        Move $o$ from $A$ to $a$;
    else
        $a = new$ Activity();
        if $S.Count > 0$ then
            Create a new predecessor $p$;
            $p.preTask = S[S.Count - 1]$;
            $p.constraintType = CBR$;
            $a.predecessor = p$; /* assign a predecessor to a */
        L1={$e \in A.bojects : e$ is L-junction with a certain $obj \in S.objects\}$;
        if $L1 = \{\}$ then
            $L1 = \{A.objects\}$;
        $o = \{e \in L1 : \{e.workspace \cap S.workspace\}$ is minimal $\}$;
        Move $o$ from $A$ to $a$;
        $S = S \cup \{a\}$;
return $S$;
```

There is no technical relationship between the newborn tasks created in this pattern. However, they normally can not be carried out concurrently because of the limit of crew capability. So the crew based dependencies (CBRs) are used to make the constraints between them to ensure the number of crews working within the capacity. This hierarchy can be applied for installing frames, or for masonry, so that the products created are stable. Algorithm 3.5 and Algorithm 3.3 are sequentially applied to decompose and sequence activities based on the L-junction pattern.

3.5 Schedule breakdown

3.5.1 Definition of schedule breakdown

Definition 3.7 *A schedule breakdown is a combination of multiple trade breakdowns and the interactions among them.*

Definition 3.7 defines a schedule breakdown as a combination of multiple trade breakdowns and their interactions. The previous section already discussed the automated breakdown of a trade and the generation of internal relationships within them. So, at this point, the breakdown of each rough activity, including internal relationships between the sub-activities, has been accomplished, and the relationships between rough activities have been known. This section therefore focuses on the automation of generating the relationships among the sub-activities belonging to different trades.

There are three cases which might occur: 1) the predecessor is still singular but the successor is decomposed into a breakdown; 2) the predecessor is decomposed into a breakdown but the successor is still singular; and 3) both of the activities are broken down in detail.

3.5.2 Combination of singular predecessor and successor breakdown

Figure 3.15 illustrates a combination of a rough predecessor and the breakdown of a successor. In this case, the predecessor (A) is not decomposed into finer activities whereas the successor (B) is broken down into detailed activities. Since the preceding objects are still kept the same because of not being decomposed, the relationships between the predecessor and the sub-activities of the successor work through the relation of the successor summary activity. In this circumstance, the relationship is kept the same as before decomposition.

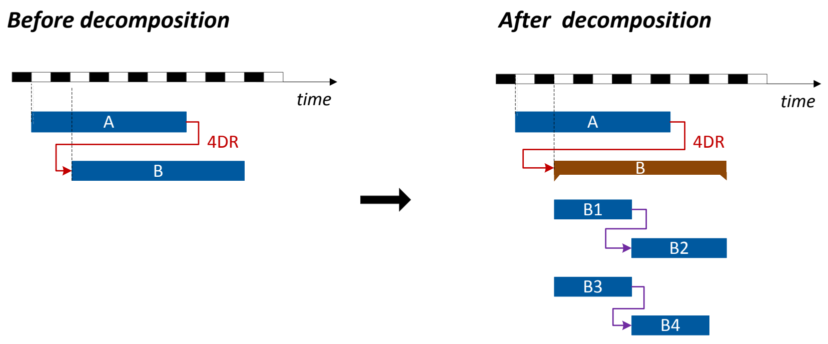


Figure 3.15: Combination of a rough predecessor and the breakdown of a successor

3.5.3 Combination of predecessor breakdown and singular successor

The predecessor (*A*) is broken down into detailed activities whereas the successor (*B*) is kept as rough as prior to the decomposition. Generally, all of the sub-activities of *A* have relations to *B*. So in this case, the relationship between *A* and *B* remains the same as previously. The adjustment of *B* will thus be controlled through the summary task *A*. This combination is depicted in Figure 3.16.

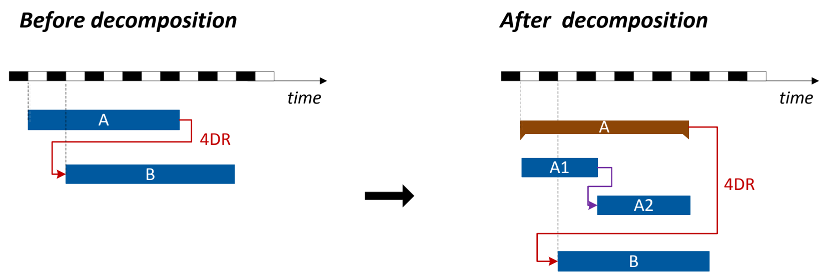


Figure 3.16: Combination of the breakdown of predecessor and the rough successor

3.5.4 Combination of two breakdowns

In this case, both the predecessor (*A*) and the successor (*B*) are decomposed into breakdowns. It is assumed that *A* and *B* have a technical relationship that can be illustrated through a 4DR. Since a sub-activity is just in charge of a specific group of objects, not every sub-activity of the summary *A* has a relation with all sub-activities of *B*. Accordingly, the relationships between the sub-activities among *A* and B are highly complex. They cannot be controlled just by a general relationship between the summary activities *A* and *B*.

In this research, when both *A* and *B* are broken down into detailed activities, the 4DR between *A* and *B* is transferred into their sub-activities to reflect the relationships efficiently.

Thanks to the self-adjustment of the 4DR status according to objects, the 4DRs will activate a relationship between two certain sub-activities that have a spatial constraint. Figure 3.17 illustrates this concept. At first, the 4DR between *A* and *B* is transferred to their children. Each sub-activity of *A* is assigned a 4DR to every sub activity of *B*. After that, these 4DRs self-detect the geometric constraints between their attached activities to set up their statuses. In detail, *A1* has geometric constraints with *B1* and *B4*, whereas *A2* has geometric constraints with *B2* and *B3*. Therefore the relationships of *A1* to *B1*, *A1* to *B4*, *A2* to *B2*, and *A2* to *B3* are set up with an *active* status, which are visible with red arrows and the legends for the 4DR. The other relationships, e.g. of *A1* to *B2*, *A1* to *B3*, *A2* to *B1*, and *A2* to *B4*, are all set up with an *inactive* status, which are invisible in Figure 3.17.

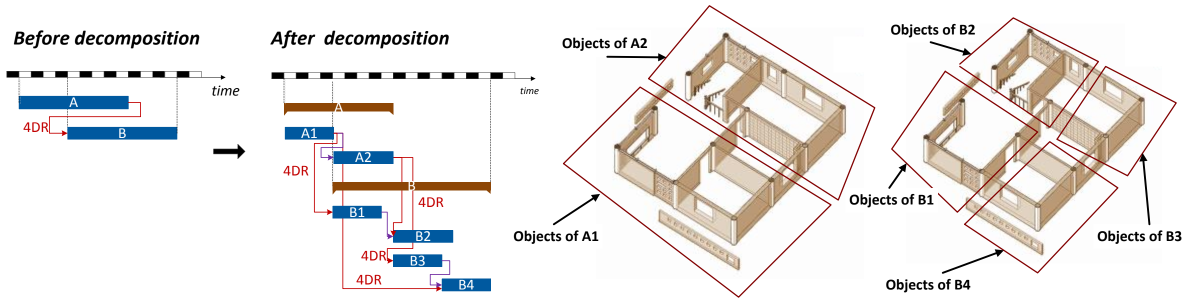


Figure 3.17: Combination of two breakdowns

Due to multiple new constraints being added between the sub-activities, the duration of the successor activity might be longer than planned. However, this time can be reduced by adjusting the CBRs of sub-activities. Since the sequence of sub-activities of the predecessor might already be defined by its predecessors, the adjustment here focuses on the sequence of sub-activities of the successor.

Figure 3.18 illustrates the algorithm of the adjustment of a sub-activity sequence that is defined by CBRs in order to reduce the duration of the summary successor. This algorithm aims to satisfy technical relationships first, i.e. 4DRs, before taking CBRs into account. The reason is that 4DRs are hard constraints, whereas CBRs can be transferred from one activity to another as long as at any point in time, the crew number is within the accepted capacity.

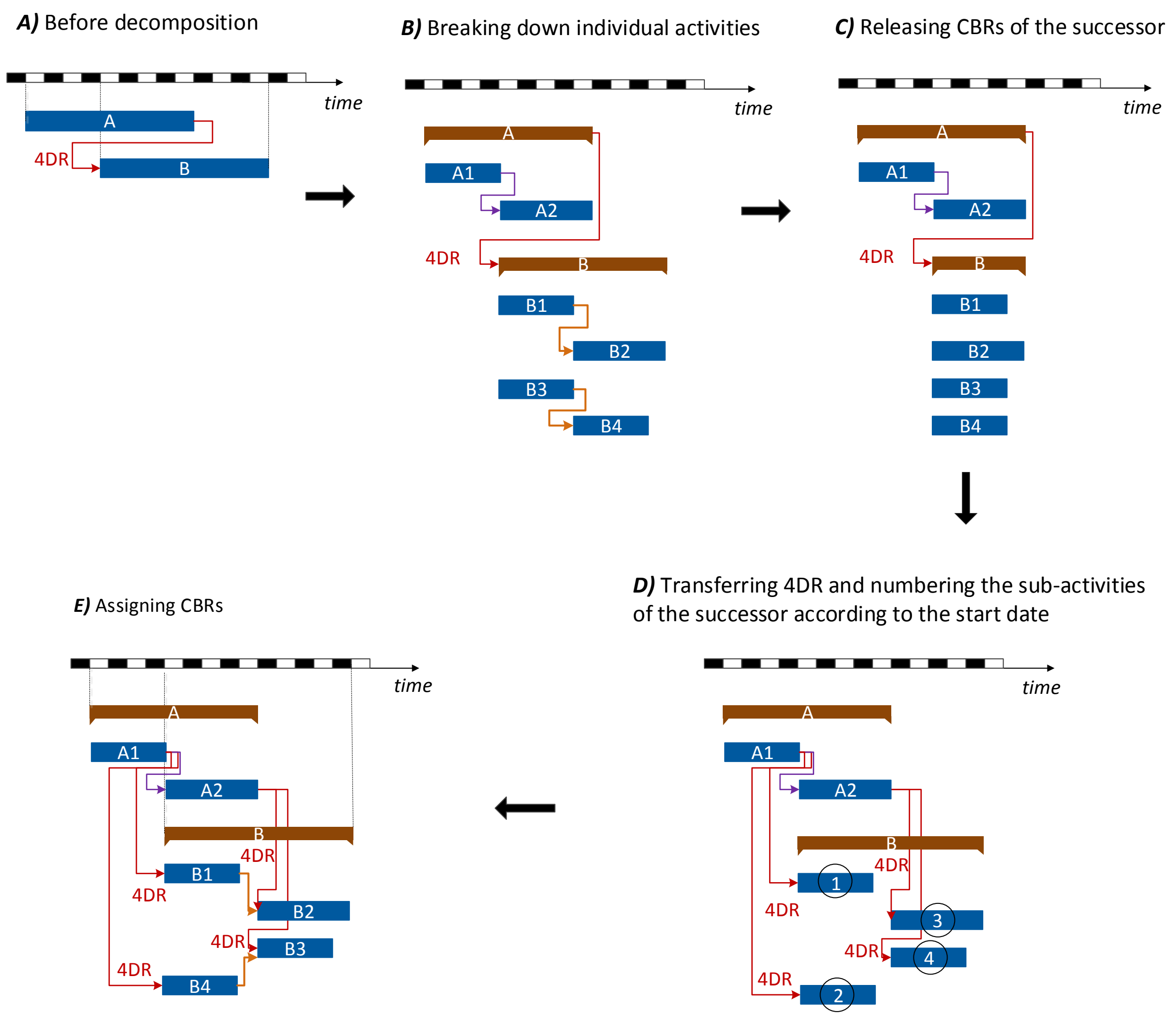
According to this algorithm, after the activities have been decomposed into two breakdowns, the CBRs between sub-activities of the successor would be removed. The next step is transferring the 4DR from the parent activities to their children. The start dates of these activities would be adjusted in accordance with the new relationships. Then the sub-activities of the successor would be assigned with ascending numbers depending on the ascending order of their start dates and their appearance in the schedule. Finally, the CBRs would be assigned to the sub-activities following the rule illustrated in formula (3.1):

$$S_p \rightarrow S_q \text{ if } \begin{cases} \text{number of } S_p \text{ is } (n-1) \times (i-1) + k \\ \text{number of } S_q \text{ is } (n-1) \times i + k \end{cases} \qquad (3.1)$$

Where,

→ denotes "is the predecessor of"

- *S* denotes sub-activity of the successor activity
- *p, q* denotes the ordinal number of *S* in its summary *(p, q = 1 ÷ m)*
- *m* denotes the number of sub-activities belonging to the successor activity
- *n* denotes the number of crews available for the successor activity
- *k* denotes the crew identity *(k = 1 ÷ n)*
- *i* denotes the ordinal number of activities associated with the k^{th} crew *(i = 1 ÷ m/k)*



A) Before decomposition
time
A
4DR
B
B) Breaking down individual activities
time
A
A1
A2
4DR
B
B1
B2
B3
B4
C) Releasing CBRs of the successor
time
A
A1
A2
4DR
B
B1
B2
B3
B4
D) Transferring 4DR and numbering the sub-activities of the successor according to the start date
time
A
A1
A2
B
4DR
1
4DR
3
4DR
4
4DR
2
E) Assigning CBRs
time
A
A1
A2
B
4DR
B1
4DR
B2
4DR
B3
4DR
B4

Figure 3.18: Adjustment of the CBRs for keeping the summary duration minimal

For example, in Figure 3.18, activity *A* is decomposed into two sub-activities *A1* and *A2*, activity *B* is decomposed into four sub-activities *B1*, *B2*, *B3* and *B4* (see figure A and B). After releasing CBRs (figure C) and considering 4DRs (figure D), *B1* and *B4* have the earliest start dates, following them are *B2* and *B3*. According to their start dates and their appearance sequence in the schedule, *B1* is marked with #*1*, *B4* with #*2*, *B2* with #*3*, and *B3* with *#4*. It is assumed that there are two crews available. Therefore, in this example, the number of crews is figured as *n=2*; so the identities of crews are *k=1* or *2*; and the ordinal number of an activity in each crew is *i=1* or *2*. Thus, the sequence of activities using the first crew, i.e. correspondingly *k=1*, is *#1* and number *#3* (correspondingly to $(2\text{-}1)\times(1\text{-}1)+1$ and $(2\text{-}1)\times(2\text{-}1)+1$), or rather activities *B1* and *B2*. Similarly, the sequence of activities using the second crew, i.e. correspondingly *k=2*, is number *2* and *4*, or rather activities *B4* and *B3* (figure E).

Let's compare the schedule before the adjustment of CBRs in Figure 3.17 to the schedule after the adjustment in Figure 3.18-E, obviously the schedule after the adjustment is 3 units of time less than before the adjustment. This is to say that the CBR adjustment according to formula (3.1) should be applied to achieve the minimum total duration.

3.6 Prototype implementation (4Dbreakdown)

The module of breaking down a 4D schedule in this approach is called *4Dbreakdown*. 4Dbreakdown is developed in Visual C#® and then embedded in Autodesk Revit®. 4Dbreakdown consists of more than 10 forms, 20 classes and 3,000 lines of code (excluding spaces, comments and the codes already mentioned in the 4Dworkspace). The source code can be found in the attached CD.

This section describes the design goals, challenges, user interface and functionality of 4Dbreakdown. They are illustrated through an application of 4Dbreakdown on a school building. In order to make the presentation clearer, this section is divided into four sub-sections. The first sub-section briefly describes design goals and challenges for 4Dbreakdown. The second sub-section describes the case study used in this implementation. The third sub-section presents the user interface and functionality of 4Dbreakdown during the creation of the input data. And the fourth sub-section discusses the user interface and functionality of 4Dbreakdown through results of the model.

3.6.1 Design goals and challenges

The implementation aims to automate the process of breaking down a 4D schedule. It attempts to facilitate end users to minimize handling operation as much as possible while working with 4Dbreakdown. It also aims to support them in efficiently checking the accuracy of results. Hence, the design goal of 4Dbreakdown must include generality, data reusability and ease of use, which were discussed briefly in Chapter 2. Since 4Dbreakdown provides a more detailed 4D schedule with new relationships between activities, its output should be transparent and able to make end users understand the meaning of these activity relationships. With this capability, the detailed schedule can be used efficiently to communicate among project stakeholders as well as to be updated in the course of the execution phase. In order to reach the proposed goals, much effort has been put into dealing with a variety and complexity of geometry as well as data structures of the model while developing 4Dbreakdown.

3.6.2 Functionality of 4Dbreakdown

4Dbreakdown is an automation of breaking down a 4D schedule. It goes with the results of 4Dworkspace to decompose 4D activities into finer and more manageable tasks as well as to create workspaces for them. After the decomposition of activities, it generates new suitable relationships between the sub-activities and then adjusts the schedule based on these relationships.

During the decomposition of 4D activities, 4Dbreakdown is able to extract and analyze geometries, structures and properties of 3D products in various categories, such as walls, floors, rooms, columns, doors, windows, etc. It also allows analyzing the geometric relations among objects and their workspaces in order to automatically collect the right objects into sub-activities. The durations of these sub-activities are then calculated by considering the ratio between their product volumes with parent-activities and the number of laborers assigned to them.

While the generation of activity relationships, 4Dbreakdown considers the relationships between summary activities, capabilities of crews as well as execution strategies to generate suitable relationships between the sub-activities. Notably, 4Dbreakdown categorizes relationships into four types according to their meaning and illustrates them by different colors, namely, 4DRs including OBDs and SBDs, CBRs and EBRs (see section 3.3). This categorization and illustration gives the output more transparent. Schedules are no longer only timetables of activities, but also provide meaning for the relationships between them. Hence, end users can easily understand the schedule and confidently adjust it in the course of the execution phase.

Finally, 4Dbreakdown has a user interface. It provides full interaction among schedule activities, activity relationships, 3D product components and workspaces. This promotes end users to efficiently control data as well as to evaluate the results.

3.6.3 Case study description

The user interface and functionality of 4Dbreakdown is demonstrated through its application on a three-story school building. Each floor of the building is about 1760 m^2 including about 27 rooms and laid out in a U-shape (Figure 3.19). The goal of this case study is to break down a rough schedule for the construction of this building into finer schedules at various levels of detail.

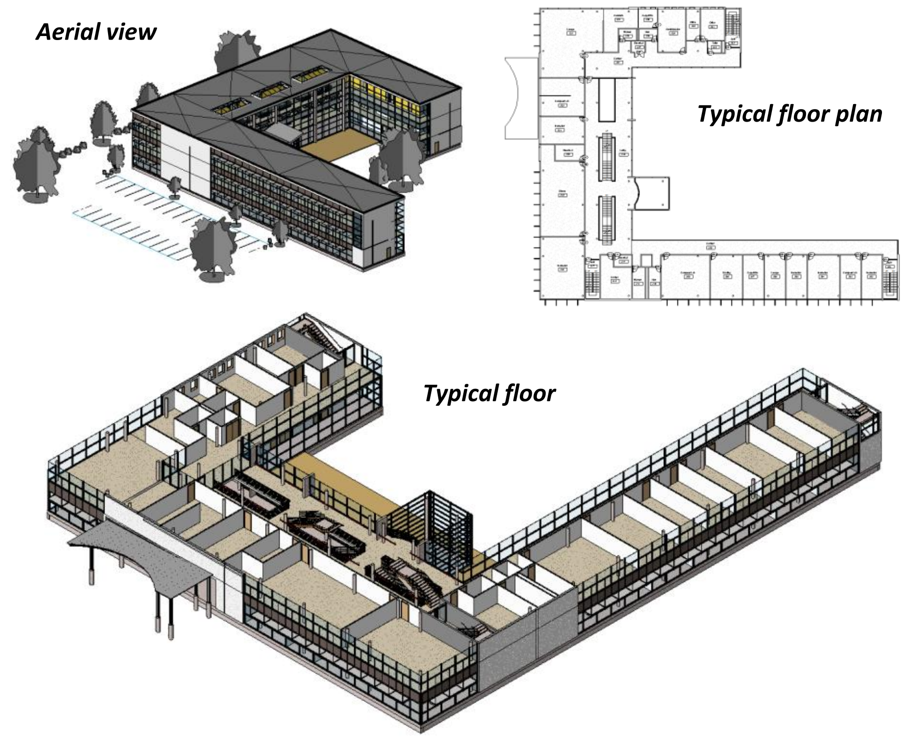


Figure 3.19: School building for the case study

In this case study, a rough schedule of ten activities is considered. A rough activity in the initial schedule associates with all the objects on which a trade must be performed. Table 3.1 describes the properties of activities and their relationships in detail.

Predecessor column presents information on predecessors of the activities. For example, the activity of drywall board (id. 4) is a predecessor of the activity of internal plastering (id. 5); the relationship constraint is based on object (OBD); the relationship type is Finish-to-Start (FS); and the work amount lag is roughly estimated as the amount of work for all the objects on one floor.

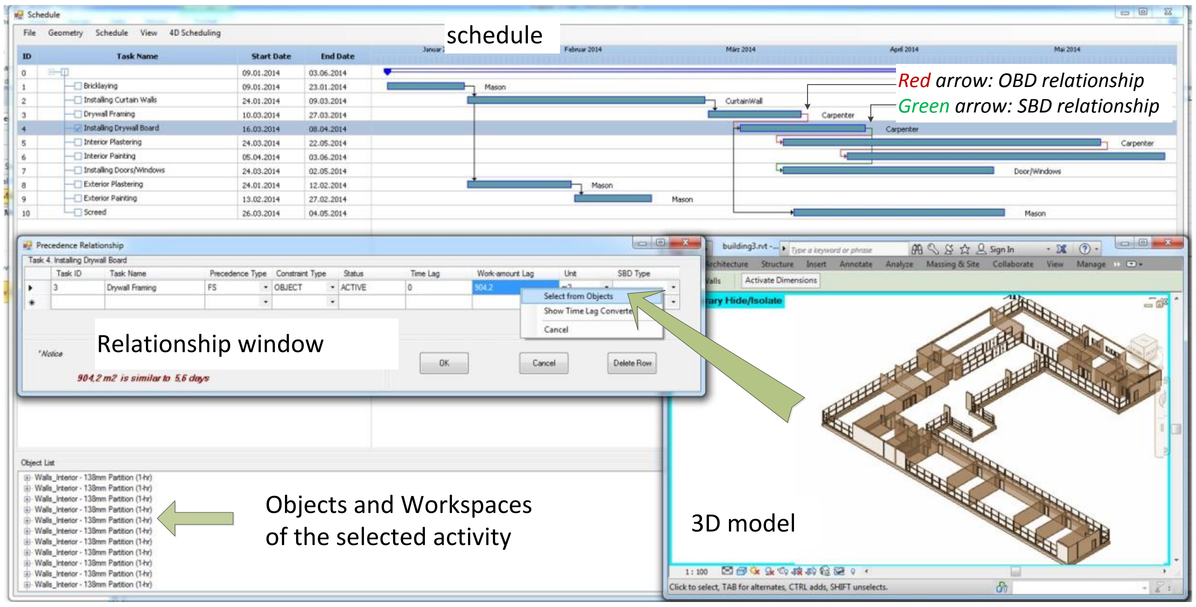
Object column describes the categories and types of objects associated with the trade. An upper-case word is the name of a category. A lower-case word, if any, presents a name of a specific type of category. In case no lower-case words appear, the activity will associate with all types of the corresponding category. Otherwise, the activity only involves the types specified by lower-case words. For example, the activity of bricklaying is only associated with the walls whose type is masonry; installing doors/windows is associated with all doors and windows, no matter which type they are.

Table 3.1: Properties and relationships of activities

ID	Task Name	Duration	Start	Finish	Predecessor	Worker Number	Objects
1	Bricklaying	15 days	09.01.14	23.01.14		4	WALL masonry
2	Installing curtain walls	45 days	24.01.14	09.03.14	1FS	4	WALL curtain
3	Drywall framing	18 days	10.03.14	27.03.14	2FS	4	WALL partition
4	Drywall board	24 days	16.03.14	08.04.14	3obd_FS+one floor	4	WALL partition
5	Internal plastering	60 days	24.03.14	22.05.14	4obd_FS+one floor	4	WALL partition
6	Internal painting	60 days	05.04.14	03.06.14	5obd_FS+1/2 floor	4	WALL partition, masonry
7	Installing doors/windows	20 days	24.03.14	02.05.14	4sbd_FS+one floor	2	DOOR; WINDOW
8	External Plastering	20 days	24.01.14	12.02.14	1FS	4	WALL masonry
9	External Painting	15 days	13.02.14	27.02.14	8FS	4	WALL masonry
10	Screed	40 days	26.03.14	04.05.14	4SS+10 days	4	FLOOR

3.6.4 Input data of 4Dbreakdown

A 4D-schedule is a combination between 3D-model and a traditional schedule. The input data for 4Dbreakdown can be created and accessed via a user interface. The data of the total model after being created and modified can be saved in XML format with the extension .SCH. This function provides reusable data. Advanced relationships in 4Dbreakdown work not only based on time as traditional relationships, but also based on the relation of activity objects. Indeed, 4DRs are used in this research in order to enable automated generation of constraints between activities.



ID	Task Name	Start Date	End Date
0		09.01.2014	03.06.2014
1	Bricklaying	09.01.2014	23.01.2014
2	Installing Curtain Walls	24.01.2014	09.03.2014
3	Drywall Framing	10.03.2014	27.03.2014
4	Installing Drywall Board	16.03.2014	08.04.2014
5	Interior Plastering	24.03.2014	22.05.2014
6	Interior Painting	05.04.2014	03.06.2014
7	Installing Doors/Windows	24.03.2014	02.05.2014
8	Exterior Plastering	24.01.2014	12.02.2014
9	Exterior Painting	13.02.2014	27.02.2014
10	Screed	26.03.2014	04.05.2014

Task 4. Installing Drywall Board

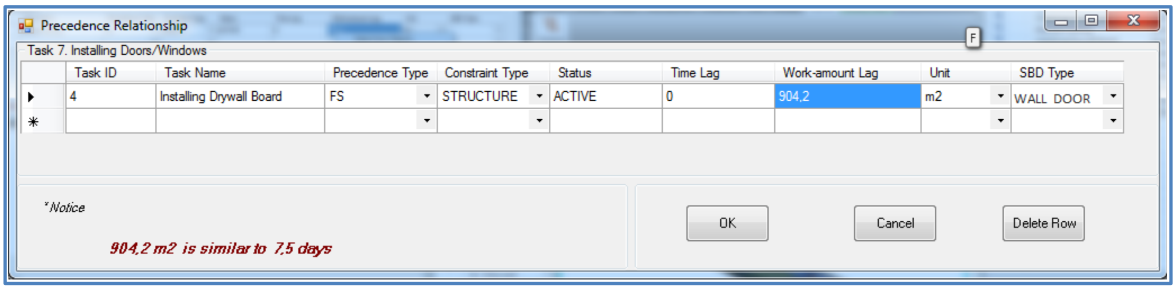
Task ID	Task Name	Precedence Type	Constraint Type	Status	Time Lag	Work-amount Lag	Unit	SBD Type
3	Drywall Framing	FS	OBJECT	ACTIVE	0	904.2	m2	

904,2 m2 is similar to 5,6 days

Walls_Interior - 138mm Partition (1-hr)
Walls_Interior - 138mm Partition (1-hr)
Walls_Interior - 138mm Partition (1-hr)
Walls_Interior - 138mm Partition (1-hr)
Walls_Interior - 138mm Partition (1-hr)
Walls_Interior - 138mm Partition (1-hr)
Walls_Interior - 138mm Partition (1-hr)
Walls_Interior - 138mm Partition (1-hr)
Walls_Interior - 138mm Partition (1-hr)
Walls_Interior - 138mm Partition (1-hr)
Walls_Interior - 138mm Partition (1-hr)

Figure 3.20: User interface for 4Dbreakdown

Figure 3.20 presents a general user interface for 4Dbreakdown. The 3D model is created and interacted with directly in Autodesk Revit®, while the initial schedule is imported from XML files, which are created from Microsoft Project. However, relationships between activities can be created and modified within 4Dbreakdown.


Precedence Relationship

Task 7. Installing Doors/Windows

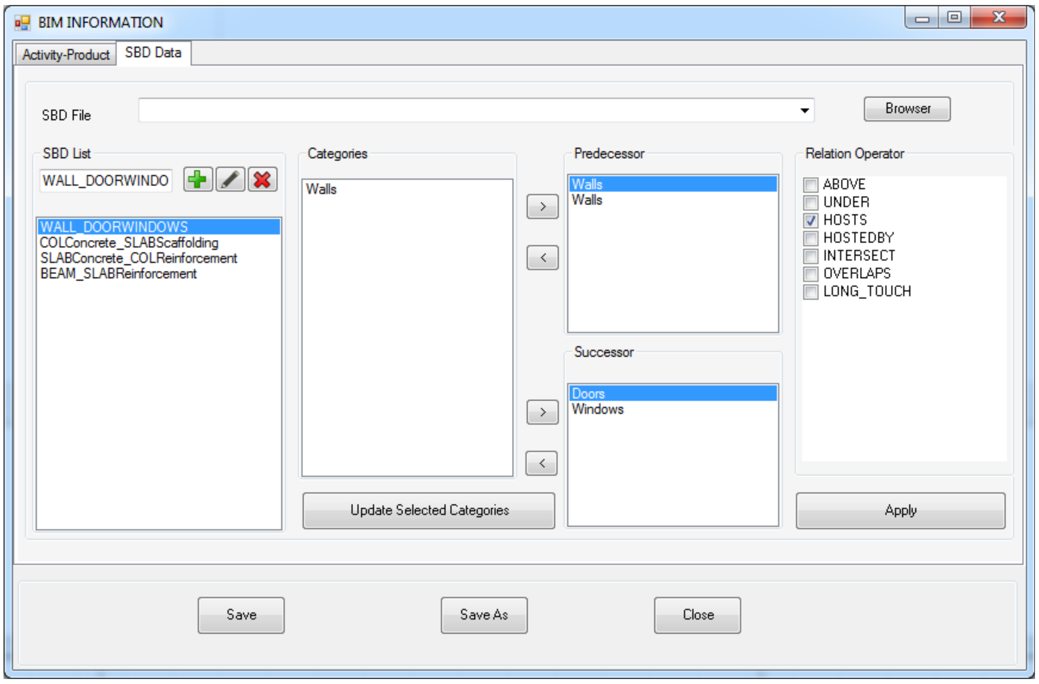
	Task ID	Task Name	Precedence Type	Constraint Type	Status	Time Lag	Work-amount Lag	Unit	SBD Type
▸	4	Installing Drywall Board	FS	STRUCTURE	ACTIVE	0	904.2	m2	WALL_DOOR
*									

*Notice

904.2 m2 is similar to 7.5 days

Figure 3.21: Properties of precedence relationship

Figure 3.21 illustrates an example of the properties of an activity relationship, which is between the activities of installing doors/windows and installing drywall boards. In reality, a door can only be installed after the wall that holds this door has been built. Therefore the precedence type of this relationship is defined as FS and the constraint type between these two activities is *STRUCTURE*. There, work amount lag is estimated as the amount of work for the objects on one floor, which is calculated similarly with 904.2 square meters. The SBD type of this structure-based relationship is defined as the selected type in Figure 3.22.


BIM INFORMATION

Activity-Product | SBD Data

SBD File

SBD List: WALL_DOORWINDO

WALL_DOORWINDOWS
COLConcrete_SLABScaffolding
SLABConcrete_COLReinforcement
BEAM_SLABReinforcement

Categories: Walls

Predecessor: Walls, Walls

Successor: Doors, Windows

Relation Operator:
- [ ] ABOVE
- [ ] UNDER
- [✓] HOSTS
- [ ] HOSTEDBY
- [ ] INTERSECT
- [ ] OVERLAPS
- [ ] LONG_TOUCH

Figure 3.22: A template for structure-based dependencies

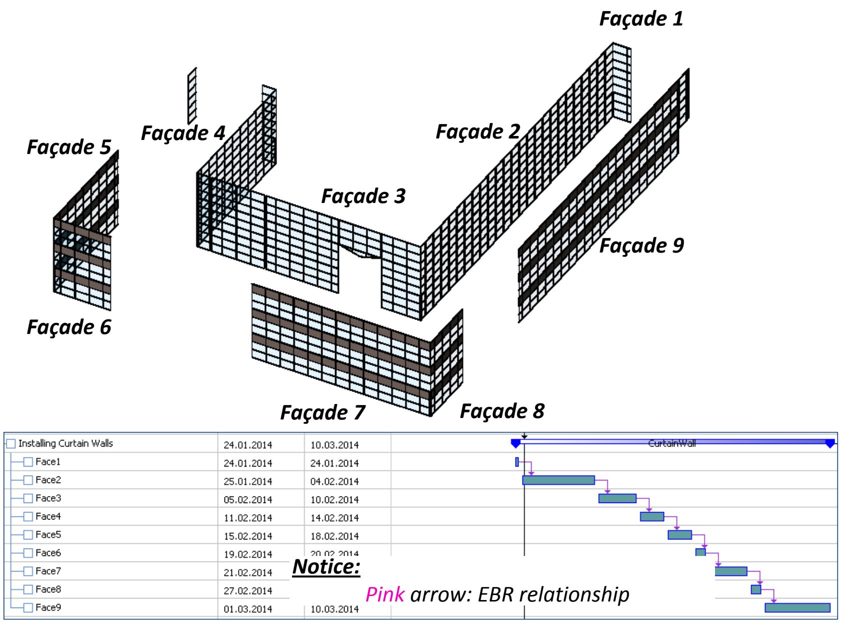
3.6.5 Rough breakdown

During a rough breakdown, 4Dbreakdown considers rough execution strategies for trades to break activities down. This factor defines the distribution of objects into sub-activities as well as the sequence for conducting them. Table 3.2 presents execution strategies for trades involved in this implementation.

At this rough level of detail, where each activity is associated with one floor or one building vertical surface, sub-activities should be conducted according to a rolling wave strategy. In other words, all crews for a summary activity should be assigned to its sub-activities so that all crews must work together to complete a floor or a surface of a building before going on to another. Therefore, relationships between sub-activities *n* and *n+1* within the same parent are assigned with the precedence type of FS with the constraint type being EBR. These sub-activities have no technical relationships, however EBRs should be obeyed to minimize the movement of crews and the relocation of equipment as well as to make the schedule more manageable. In a special situation, this sequence can be changed if required without any conflict in terms of technical issues.

Table 3.2: Execution strategies for trades

ID	Task Name	Rough execution	Detailed execution	Construction method
1	Bricklaying	FLOOR-bottom up	round	
2	Installing curtain walls	FACE-counter clockwise	linear (bottom-up)	using hoist to install curtain walls
3	Drywall framing	FLOOR-bottom up	L-junction	
4	Drywall board	FLOOR-bottom up	linear (X direction)	
5	Internal plastering	FLOOR-bottom up	room	
6	Internal painting	FLOOR-bottom up	room	
7	Installing doors/windows	FLOOR-bottom up	linear (X direction)	
8	External Plastering	FACE-counter clockwise	linear (top-down)	
9	External Painting	FACE-counter clockwise	linear (top-down)	
10	Screed	FLOOR-bottom up	room	

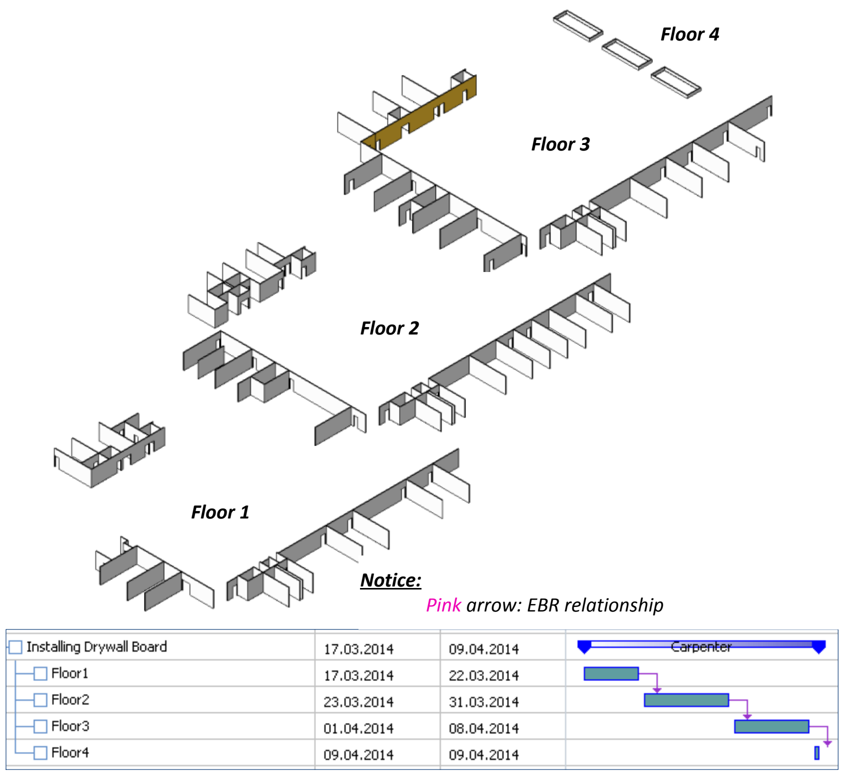


Installing Curtain Walls	24.01.2014	10.03.2014
Face1	24.01.2014	24.01.2014
Face2	25.01.2014	04.02.2014
Face3	05.02.2014	10.02.2014
Face4	11.02.2014	14.02.2014
Face5	15.02.2014	18.02.2014
Face6	19.02.2014	20.02.2014
Face7	21.02.2014	
Face8	27.02.2014	
Face9	01.03.2014	10.03.2014

Figure 3.23: Façade-based decomposition for installing curtain walls

The façade-based decomposition pattern is applied to decompose the activities of installing curtain walls, external plastering and external painting. Figure 3.23 illustrates the decomposition of installing curtain walls. Since hoists are supposed to be used for transporting curtain walls, the execution strategy for conducting this process is to install building façades one after another. The chosen execution direction is counter-clockwise. As a result, nine sub-activities are created and sequenced by EBRs.

The floor-based decomposition pattern is used for the other activities of the schedule. Figure 3.24 shows the decomposition of installing drywall boards. According to floor levels of the objects, the activity is divided into four finer sub-activities. The relationships between these activities are also EBRs.

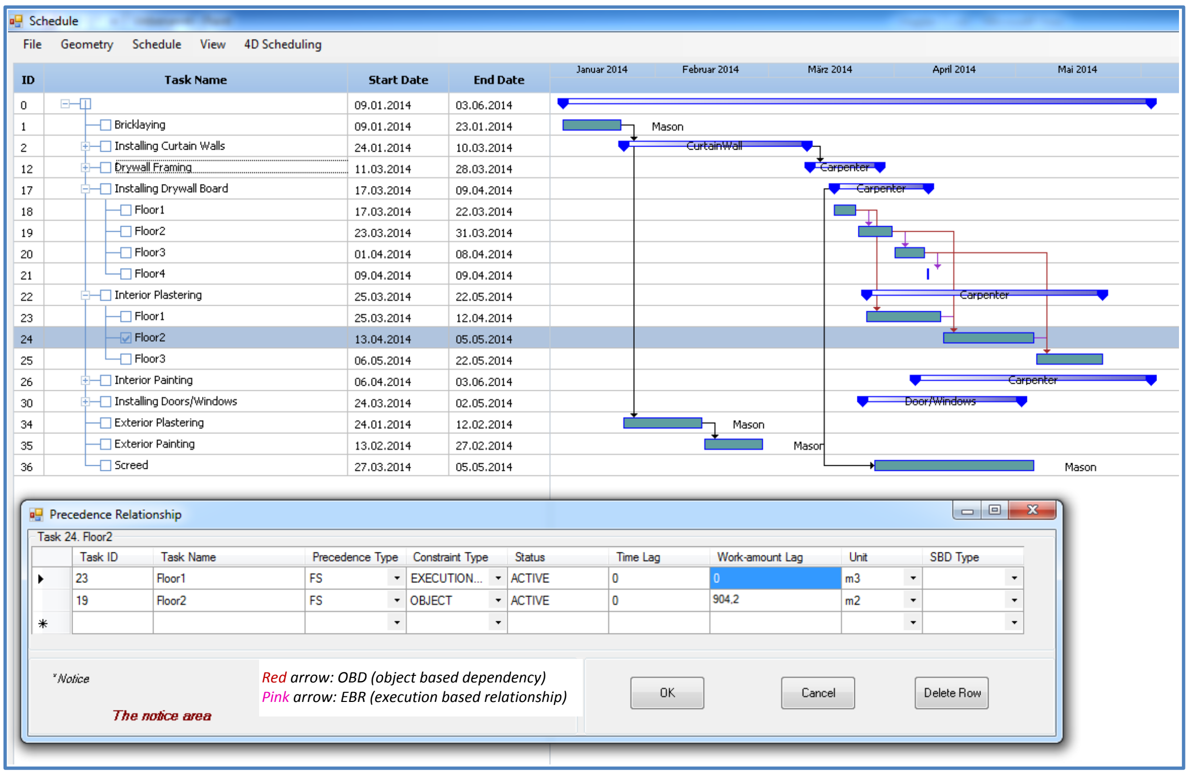


Installing Drywall Board	17.03.2014	09.04.2014	Carpenter
Floor1	17.03.2014	22.03.2014	
Floor2	23.03.2014	31.03.2014	
Floor3	01.04.2014	08.04.2014	
Floor4	09.04.2014	09.04.2014	

Figure 3.24: Floor-based decomposition for installing dry walls

Relationships between sub-activities are automatically generated. 4Dbreakdown sets constraints between sub-activities, which have the same summary activity, with EBRs as mentioned in the two above discussed examples. The sub-activities coming from different summaries, however, are assigned with different constraints based on the relationships of their summary activities. If their summary activities have a traditional relationship, the so-called time-based relationship in this research, the relationship would be kept at the summary level and hence, every sub-activity receives the same effect from this relationship. Otherwise, if their summary activities have a 4DR, then they can self-recognize which activities have constraints with them to generate corresponding relationships. Figure 3.25 and Figure 3.26 illustrate examples in the case that summary activities are constrained by 4DRs.

In Figure 3.25, the interior plastering has a 4DR to the installing drywall boards with the type of OBD. During the decomposition of activities, the relationships of the summary activities transfer to their sub-activities. However, OBDs are just active if their attached activities have at least one common object. As a result, an active OBD is automatically generated between plastering on floor1 and installing drywall boards on the same floor1; and the like happens for floor2 and floor3. No relations exist between plastering and installing drywall boards for activities on different floors.



ID	Task Name	Start Date	End Date
0		09.01.2014	03.06.2014
1	Bricklaying	09.01.2014	23.01.2014
2	Installing Curtain Walls	24.01.2014	10.03.2014
12	Drywall Framing	11.03.2014	28.03.2014
17	Installing Drywall Board	17.03.2014	09.04.2014
18	Floor1	17.03.2014	22.03.2014
19	Floor2	23.03.2014	31.03.2014
20	Floor3	01.04.2014	08.04.2014
21	Floor4	09.04.2014	09.04.2014
22	Interior Plastering	25.03.2014	22.05.2014
23	Floor1	25.03.2014	12.04.2014
24	Floor2	13.04.2014	05.05.2014
25	Floor3	06.05.2014	22.05.2014
26	Interior Painting	06.04.2014	03.06.2014
30	Installing Doors/Windows	24.03.2014	02.05.2014
34	Exterior Plastering	24.01.2014	12.02.2014
35	Exterior Painting	13.02.2014	27.02.2014
36	Screed	27.03.2014	05.05.2014

Task 24. Floor2

Task ID	Task Name	Precedence Type	Constraint Type	Status	Time Lag	Work-amount Lag	Unit	SBD Type
23	Floor1	FS	EXECUTION...	ACTIVE	0	0	m3	
19	Floor2	FS	OBJECT	ACTIVE	0	904,2	m2	

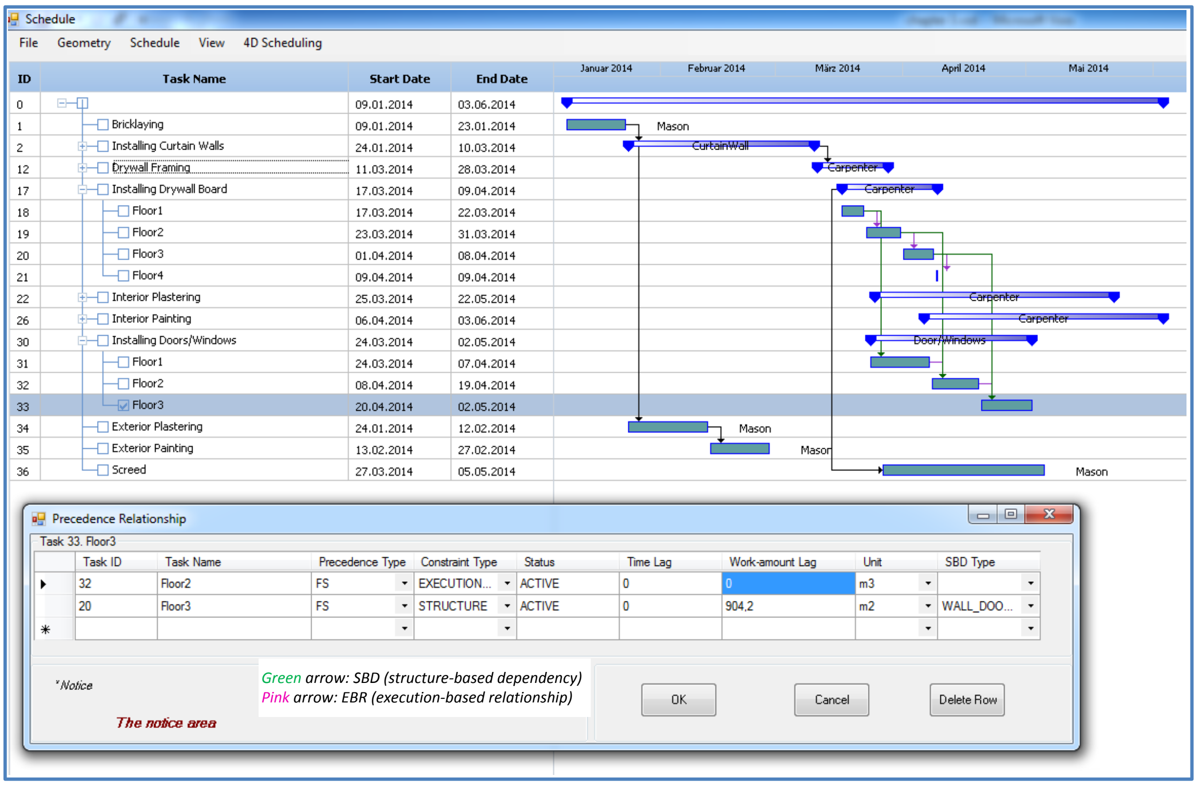
*Notice

The notice area

Red arrow: OBD (object based dependency)
Pink arrow: EBR (execution based relationship)

Figure 3.25: Automated generation of object-based dependencies between sub-activities

In Figure 3.26, the activity of installing doors/windows has a constraint with the installation of drywall boards. This constraint is 4DR with the type of SBD specific to WALL_DOORWINDOW. This constraint is also transferred to their sub-activities after the decomposition. However, this specific SBD is just active when there is at least one wall of the installation of drywall boards that holds either a door or window associated with the activity of installing doors/windows. As a result, SBDs are automatically generated between sub-activities performing on the same floor.



ID	Task Name	Start Date	End Date
0		09.01.2014	03.06.2014
1	Bricklaying	09.01.2014	23.01.2014
2	Installing Curtain Walls	24.01.2014	10.03.2014
12	Drywall Framing	11.03.2014	28.03.2014
17	Installing Drywall Board	17.03.2014	09.04.2014
18	Floor1	17.03.2014	22.03.2014
19	Floor2	23.03.2014	31.03.2014
20	Floor3	01.04.2014	08.04.2014
21	Floor4	09.04.2014	09.04.2014
22	Interior Plastering	25.03.2014	22.05.2014
26	Interior Painting	06.04.2014	03.06.2014
30	Installing Doors/Windows	24.03.2014	02.05.2014
31	Floor1	24.03.2014	07.04.2014
32	Floor2	08.04.2014	19.04.2014
33	Floor3	20.04.2014	02.05.2014
34	Exterior Plastering	24.01.2014	12.02.2014
35	Exterior Painting	13.02.2014	27.02.2014
36	Screed	27.03.2014	05.05.2014

Task 33. Floor3

Task ID	Task Name	Precedence Type	Constraint Type	Status	Time Lag	Work-amount Lag	Unit	SBD Type
32	Floor2	FS	EXECUTION...	ACTIVE	0	0	m3	
20	Floor3	FS	STRUCTURE	ACTIVE	0	904,2	m2	WALL_DOO...

*Notice

Green arrow: SBD (structure-based dependency)
Pink arrow: EBR (execution-based relationship)

The notice area

Figure 3.26: Automated generation of structure-based dependencies between sub-activities

3.6.6 Detailed breakdown

A detailed breakdown normally has the level 5 of detail. In 4Dbreakdown, the desired duration of sub-activities is pre-defined by the schedulers. Besides this, several factors also influence the detailed breakdown of activities. They are detailed execution strategies, workspace requirements, crew capacity, and in some cases, starting object as well.

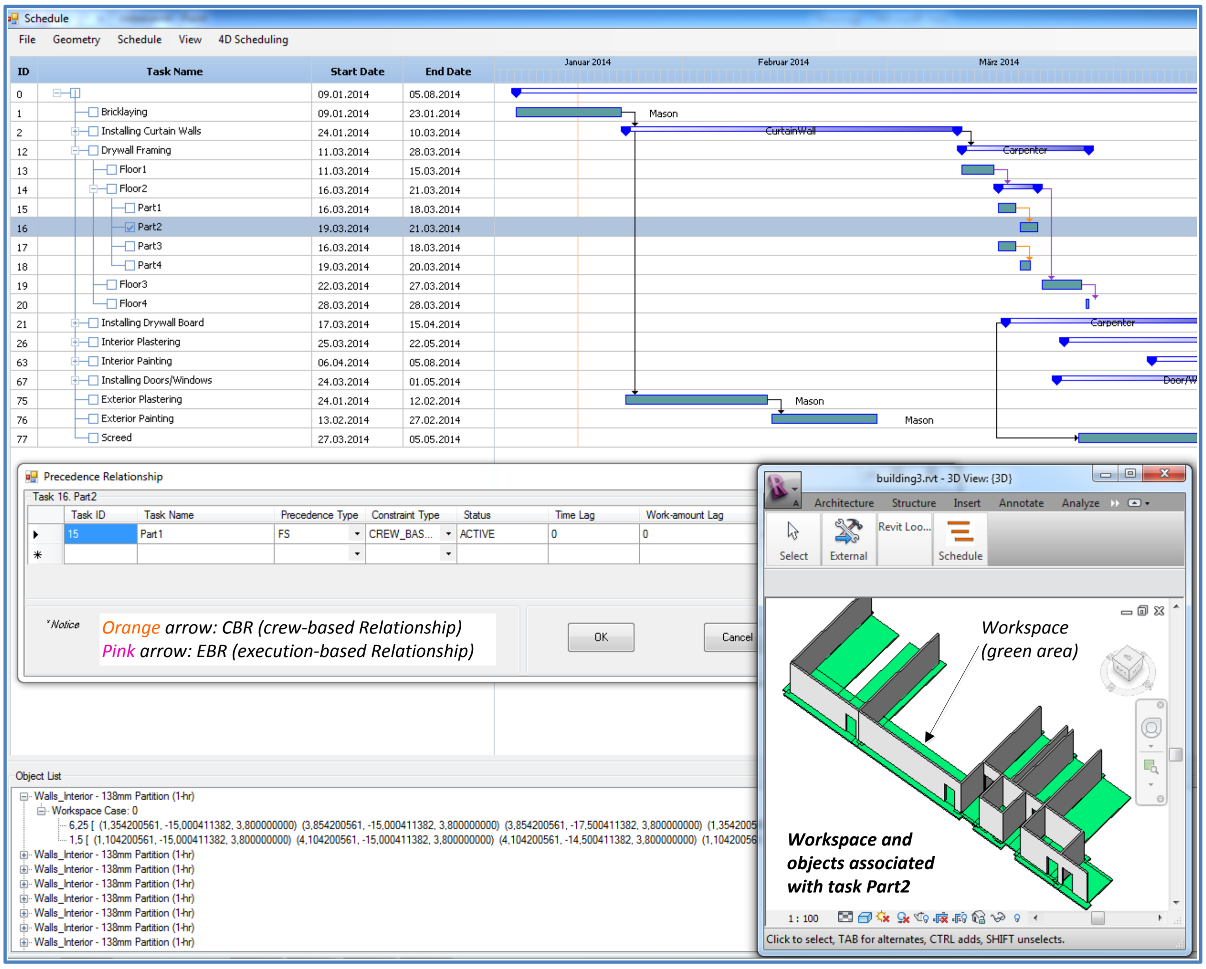
Desired duration for a sub-activity defines the detailed level for a decomposition of a task. This duration can indirectly be defined by the desired number of segments, into which the task should be decomposed. During detailed breakdown, objects are sequentially selected into each sub-activity until the corresponding duration of this sub-activity reaches the desired duration.

Detailed execution direction is a property of the associated trade of an activity. An execution direction can be linear, circular, room or L-junction. This direction defines how objects should be grouped into sub-activities to ensure a well-organized schedule. In some cases, e.g., when a circular execution direction is chosen, it also determines the relationships between sub-activities by using EBRs.

Workspace is originally generated for every single object associated with an activity. This factor defines the secondly grouping priority of objects after the detailed execution direction. If several objects have the same grouping priority according to a selected execution direction, their workspace will be considered to further prioritize them. In this case, 4Dbreakdown sets a higher priority on the object, whose workspace can combine with the existing workspace of the considered sub-activity to create a smaller workspace. In the event that an object has two or more possibilities for workspace allocation, for example for two-surface trades, the chosen possibility is also the one that makes the workspace combination of the sub-activity minimal.

Crew capacity is assigned to activities as input data (see Table 3.1). This factor defines the temporary relationships (CBRs) among sub-activities after a breakdown process. Thanks to these relationships, the sub-activities are sequenced such that their crew requirement is within capacity.

Starting object is an optional input data. Starting objects define the first objects that will be conducted by individual crews. If this data is not set, 4Dbreakdown will, by default, choose the object that has the smallest coordinate as the starting object for the first crew.



Schedule
File Geometry Schedule View 4D Scheduling
Januar 2014
Februar 2014
März 2014
Mason
CurtainWall
Carpenter
Carpenter
Door/W
Mason
Mason
building3.rvt - 3D View: {3D}
Architecture Structure Insert Annotate Analyze
Select External Revit Loo... Schedule
Workspace (green area)
Workspace and objects associated with task Part2
1 : 100
Click to select, TAB for alternates, CTRL adds, SHIFT unselects.

ID	Task Name	Start Date	End Date
0		09.01.2014	05.08.2014
1	Bricklaying	09.01.2014	23.01.2014
2	Installing Curtain Walls	24.01.2014	10.03.2014
12	Drywall Framing	11.03.2014	28.03.2014
13	Floor1	11.03.2014	15.03.2014
14	Floor2	16.03.2014	21.03.2014
15	Part1	16.03.2014	18.03.2014
16	Part2	19.03.2014	21.03.2014
17	Part3	16.03.2014	18.03.2014
18	Part4	19.03.2014	20.03.2014
19	Floor3	22.03.2014	27.03.2014
20	Floor4	28.03.2014	28.03.2014
21	Installing Drywall Board	17.03.2014	15.04.2014
26	Interior Plastering	25.03.2014	22.05.2014
63	Interior Painting	06.04.2014	05.08.2014
67	Installing Doors/Windows	24.03.2014	01.05.2014
75	Exterior Plastering	24.01.2014	12.02.2014
76	Exterior Painting	13.02.2014	27.02.2014
77	Screed	27.03.2014	05.05.2014

Precedence Relationship

Task 16. Part2

Task ID	Task Name	Precedence Type	Constraint Type	Status	Time Lag	Work-amount Lag
15	Part1	FS	CREW_BAS...	ACTIVE	0	0

*Notice *Orange* arrow: CBR (crew-based Relationship)
Pink arrow: EBR (execution-based Relationship)

OK Cancel

Object List
```
Walls_Interior - 138mm Partition (1-hr)
  Workspace Case: 0
    6,25 [ (1,354200561, -15,000411382, 3,800000000) (3,854200561, -15,000411382, 3,800000000) (3,854200561, -17,500411382, 3,800000000) (1,3542005
    1,5 [ (1,104200561, -15,000411382, 3,800000000) (4,104200561, -15,000411382, 3,800000000) (4,104200561, -14,500411382, 3,800000000) (1,10420056
Walls_Interior - 138mm Partition (1-hr)
Walls_Interior - 138mm Partition (1-hr)
Walls_Interior - 138mm Partition (1-hr)
Walls_Interior - 138mm Partition (1-hr)
Walls_Interior - 138mm Partition (1-hr)
Walls_Interior - 138mm Partition (1-hr)
Walls_Interior - 138mm Partition (1-hr)
```

Figure 3.27: Detailed breakdown of drywall framing on a floor

Figure 3.27 illustrates the detailed breakdown of drywall framing on floor2. Drywall framing is a two-surface trade. So there are two possibilities of workspace allocation for conducting each object (Figure 3.28). These possibilities are generated and assigned to objects of an activity before decomposition is carried out. However only one possibility is chosen during the decomposition based on the rule of workspace minimization for a task. The final workspace requirement for drywall framing at part2 on floor2 is also seen in Figure 3.27.

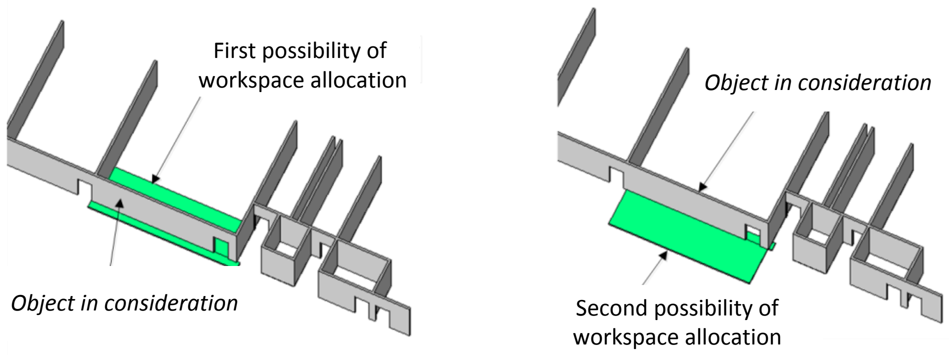
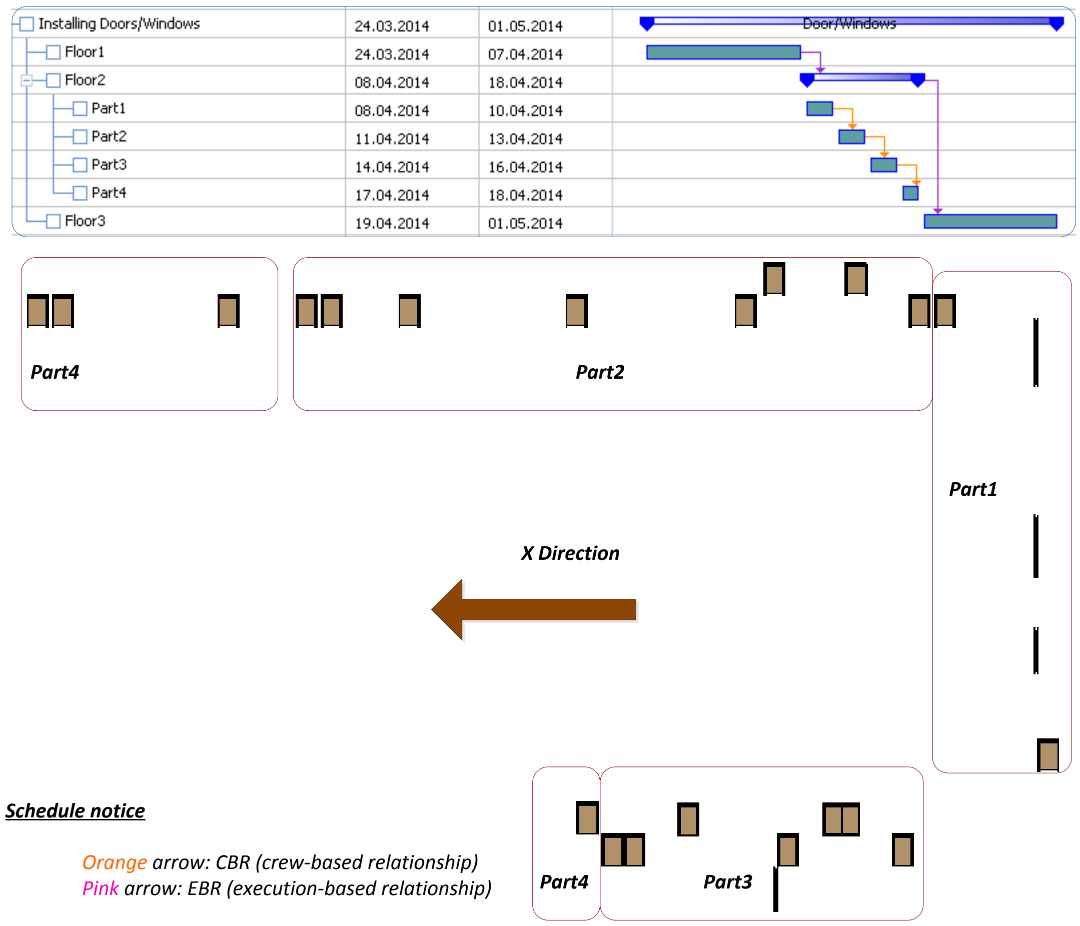

First possibility of workspace allocation
Object in consideration
Object in consideration
Second possibility of workspace allocation

Figure 3.28: Possibilities of workspace allocation for an object associated with drywall framing

As indicated in Table 3.1, four carpenters are assigned to the task of drywall framing. 4Dbreakdown considers by default that a crew contains two workers. So there are two crews taking part in this task. According to the L-junction pattern, 4Dbreakdown considers crew capacity to create relationships between sub-activities from the same parent. Therefore the relationships between them are assigned with CBRs.

Figure 3.29 presents the detailed breakdown of installing doors/windows on floor2. This task is decomposed based on the linear pattern and assigned to two workers, similarly to one crew. Since sub-activities decomposed with this pattern are independent, the constraints between them are only based on crew capacity (CBR). Notably, this decomposition pattern is not purely linear. During decomposition, 4Dbreakdown also considers the isolation among objects to distribute them into sub-activities. The grouping priority in this case is that objects in the same certain activity should not be isolated by courtyards or openings on a floor. As a result, the sub-activities of part2 and part3 are generated very feasibly. The activity of part4 is created in the end, so it takes part in the rest of the remaining objects. However, in this case, schedulers can get a better decomposition by keeping the decomposition pattern of linear X, but with its reverse direction.



Installing Doors/Windows	24.03.2014	01.05.2014	Door/Windows
Floor1	24.03.2014	07.04.2014	
Floor2	08.04.2014	18.04.2014	
Part1	08.04.2014	10.04.2014	
Part2	11.04.2014	13.04.2014	
Part3	14.04.2014	16.04.2014	
Part4	17.04.2014	18.04.2014	
Floor3	19.04.2014	01.05.2014	

Part4
Part2
Part1
X Direction
Part4
Part3
Schedule notice
Orange arrow: CBR (crew-based relationship)
Pink arrow: EBR (execution-based relationship)

Figure 3.29: Detailed breakdown of installing doors/windows on a floor

3.7 Conclusion

This chapter has presented a framework and a prototype implementation for an automated breakdown of a 4D-schedule into various levels of detail. With the support of the rough and detailed decomposition patterns, the decomposition of a geometry building has been performed well. Besides this, thanks to the advanced relationships developed in this model, namely 4DR, CBR and EBR, the linking of sub-activities is also automated to reflect geometric constraints, as well as the limited capacity of resources and execution strategy. These relationships also make the schedule more transparent and understandable.

It should also be noticed that this chapter does not involve optimization. The detailed schedule generated by 4Dbreakdown does not focus on solving spatial conflicts. 4Dbreakdown purely decomposes activities into more manageable sub-activities, chooses suitable workspaces for them and assigns relationships to them in order to reflect technical constraints as well as crew capacity. Solving spatial conflicts is considered in Chapter 4.

4 CONFLICT RESOLUSION

*"**One:** Heuristics are groundless decisions which have no mathematical proofs. They give us the results which are only good enough for practice, but they are not the best ones.*

***The other:** No! Heuristics are decisions in a field irrelevant to the subject and competence of mathematics. The results of heuristics are often much better than those which can be obtained from a formalized approach."*

--Ivakhnenko (1970)--

4.1 Introduction to conflict resolution

The realization of construction projects involves various contractors. Communicating schedules and strategies among them are important tasks on a construction site. This is especially serious in the finishing stage of execution, when many stakeholders, such as constructors, electricians, sanitation engineers, facility coordinators and other contractors compete with each other for a working place in a limited space. Besides, tasks during this period of execution are open for time flexibility, so that there are various alternatives for the contractors to arrange for the schedule and adapt to their own conditions. If a detailed schedule including the mapping of subcontractors' workspaces over time is not created beforehand by the site manager, space disputes between entities will occur. Therefore, generating a sufficiently detailed short-term schedule is necessary for the finishing period, when several trades compete for a limited space.

Moreover, a challenge for any efficient scheduling method is its adaptability. Every construction project has its own individual characteristics such as its contractor's efficiency and capability to mobilize resources. In some cases, for example, in order to resolve a problem, compressed schedules can be considered as a solution if site managers are able to mobilize extra crews as well as provide extra financing. In other cases, however, it is easier to deal with the problem by leaving some activities suspended to give their workspaces to others first, if extra slack time is acceptable. Therefore, it is difficult to find one specific solution which fits every context. This section, thus, proposes a method which generates various alternatives for arranging activities to deal with spatial conflicts. From this, construction managers can choose the solution which best suits their situation.

The proposed model is an integration of simulation and heuristic optimization. The simulators analyze the behavior of schedules considering workspace and crew requirements. They investigate how workspace and crew conflicts may impact the schedule and accordingly evaluate an activity sequence. The optimization engine generates a set of activity sequences that are near the Pareto front based on the evaluation results of the simulation process. Finally, the feasibility and flexibility of the proposed solutions are analyzed via an example application.

4.2 Previous research on workspace planning

Many researchers have been interested in space planning and confirm the necessity of workspace management during planning and scheduling in terms of detecting spatial conflicts. The integration of workspace management within a schedule allows users to be aware of possible workspace conflicts prior to their occurrences on site. This facilitates their informed decision making. Along with the research that just focuses on generating workspace requirements and detecting workspace clashes (Akinci et al. 2002), plenty of other research considers searching for strategies to overcome these conflicts. Currently, there are two directions in dealing with spatial conflicts. The first direction is to create a schedule which eliminates all possible spatial congestions (Jongeling and Olofsson 2007, Zhang et al. 2007, Elmahdi, Wu, and Bargstädt 2011, Akbaş 2004); and the second direction is to adjust a schedule to minimize congestion (Mallasi 2006, Winch and North 2006, Bansal 2011).

4.2.1 Conflict-free research on workspace planning

In the conflict-free direction, the application of discrete-event simulation (DES) to eliminate possible spatial conflicts should be acknowledged as advanced research. Akbaş (2004) and Elmahdi, Wu, and Bargstädt (2011) have applied DES to solve workspace conflicts. An activity will be checked if the workspace associated with it is still free before allowing it to be conducted. If the associated workspace is not free, the activity must wait until another activity releases this workspace. Moreover, in order to make workflows smooth, Akbaş has considered execution strategies to sequence activities.

Another application, also acknowledged as a successful method to eliminate conflicts, is the application of Line-of-Balance (LOB) such as the research of Jongeling (2006a). Indeed, LOB has been evaluated as a simple, transparent and understandable method which ensures a smooth movement of crews through construction sites with minimal conflicts as well as decreases in idle time for crews and equipment (Arditi and Albulak 1986). This method is very useful with repetitive and linear scheduling. It is also a good choice for scheduling activities associated with a fixed chain of trades. Therefore, research associated with the application of LOB is successful in being applied for scheduling earthworks and super structural construction. However, in other cases, such as in the finishing phase, when various activities can take place at the same time and have certain independences, LOB expresses inefficiency. Just take a look at the following example for scheduling three activities: A, B and C (Figure 4.1). In this example, B and C can take place parallel in terms of technique. As a result, the optimized solution with LOB, which has a fixed and continuous sequence of workflow, has a longer duration than the global optimized schedule, in which activity C is carried out with another location sequence.

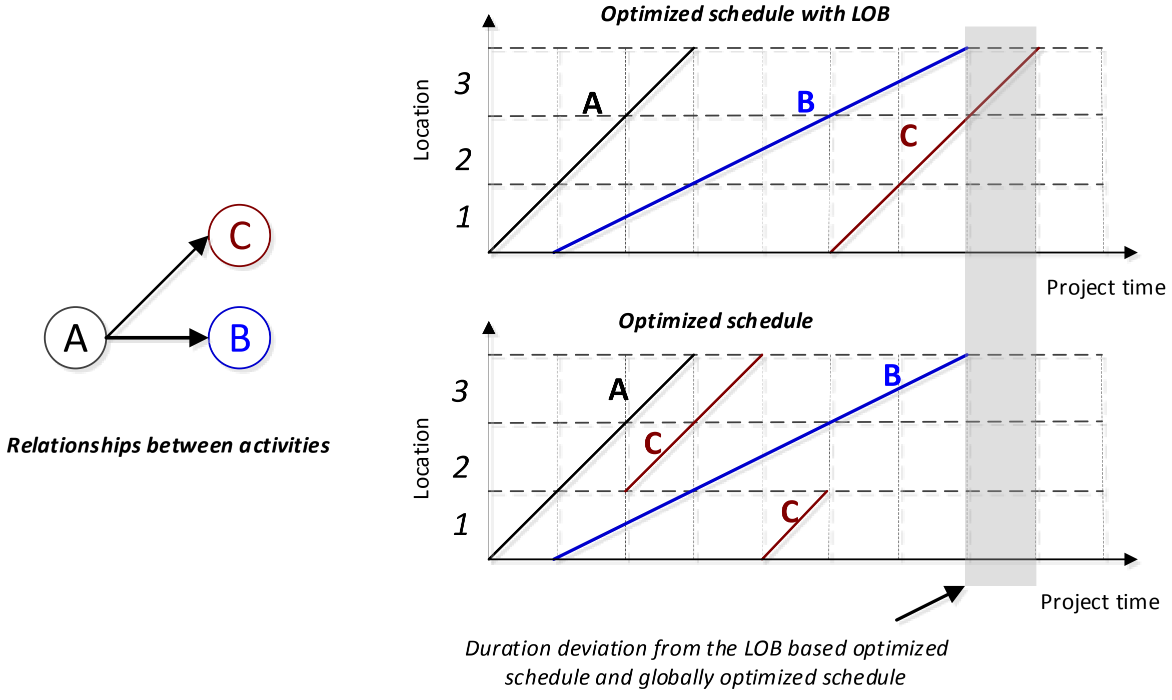

Optimized schedule with LOB
Location
3
2
1
A
B
C
Project time
Optimized schedule
Location
3
2
1
A
B
C
C
Project time
C
A
B
Relationships between activities
Duration deviation from the LOB based optimized schedule and globally optimized schedule

Figure 4.1: Disadvantage of LOB as applied for scheduling independent activities

Aside from this disadvantage, the decomposition of a working area into the same zones for different trades in LOB cannot reflect the reality of construction in the finishing phase, because the workspace depends much on the equipment and number of laborers required. Accordingly, they vary much from one trade to another. Moreover, organization with multiple crews working concurrently for the same trade does not go well with LOB. In order to avoid the over-crowding of laborers, a trade could be organized into more than one crew working in different areas at the same time. If two crews or more are involved in a trade, schedulers must manually create scenarios of crew-space assignments to evaluate them and then choose the best one in LOB. Thus, these assignments are very time-consuming. In conclusion, LOB does not fit trades associated with the finishing phase of execution since this method requires the same discrete workspace system for different trades. In addition, this method does not work well for the organization of multiple trade crews.

Most importantly, however, the application of a conflict-free method for look-ahead planning might not be feasible. In order to maintain flexibility in planning, detailed schedules are normally developed not more than three to four weeks ahead as an expansion of a corresponding part of an overall execution plan. At this time, the time frame of a project is already determined. Manipulating activities to eliminate all workspace conflicts might then cause serious delays in the project. This makes the conflict-free method insufficient if solely applied.

4.2.2 Conflict-minimal research on workspace planning

The conflict-minimal method has been proposed by Mallasi (2006), Winch and North (2006) and Bansal (2011). This method attempts to keep congestions minimal within an acceptable time frame of the project. In order to search for schedules with minimal congestion, researchers have used different algorithms. Mallasi integrates a simple genetic algorithm with different work rates and execution patterns to find the best solution. However, Mallasi considers fixed start dates of activities. This assumption makes the approach inflexible to benefit from a time buffer of activities in the finishing phase. Winch and North use the “brute force” algorithm. That means that they investigate almost all possibilities of adjusting the schedule before choosing the best one. Bansal suggests a model, in which users can manually adjust a schedule and the spatial requirements, or split activities to find a suitable solution. These two latter models are time-consuming and not compatible with large-scale searching with approaches to scheduling problems.

4.2.3 Conclusion and research gap

All of the approaches mentioned above, except the manual model of Bansal (2011), do not allow an activity to be interrupted. This means that whenever an activity starts, it will occupy resources, e.g. workspace, until it finishes. This reduces the number of potential schedules during the search process and cannot reflect the priority of critical tasks. In reality, sometimes tasks must be interrupted in order to give additional resources to other tasks belonging to the critical path if needed. Another shortcoming of the aforementioned researches is the limitation of the number of results proposed. All of these provide only one solution. This limits the adaptability of the research to the variety of construction projects.

The approach in this dissertation therefore attempts to overcome the current shortcomings of workspace planning through an integration of simulation and Pareto-based optimization. This model is able to give out multiple solutions to improve the adaptability of the proposed results. It also considers suspending an activity as a solution to solve spatial conflicts.

Moreover, since this workspace planning is carried out as a look-ahead schedule, whose duration frame is already predefined, the approach would follow the second direction of resolving potential conflicts, which aims to keep conflicts as minimal as possible within an acceptable time frame.

4.3 Input data of the model

Input data for the model is a 4D detailed schedule including the crew capacity of a project and workspace requirements for activities. In detail, the data of an activity includes four types: 1) information of a schedule such as earliest start and end dates, free slack, total slack, constraints with other activities, along with the kind and number of crews required; 2) information of geometry data such as location of products and location of the products' workspace corresponding; 3) task properties which define whether or not a task can be interrupted; and 4) the range of acceptable delay duration.

The types 1 and 2 of the input data inherit from the results of the model proposed in chapter 3. However, it should be mentioned that the crew-based relationships (CBRs) in the detailed schedule created in chapter 3 must be released before launching this process since they are generated just in order to roughly estimate the schedule duration considering the limited crew capacity.

4.4 Simulation of conflict resolution

4.4.1 Need of simulation for conflict resolution

A significant portion of detailed activities in the finishing phase can be carried out concurrently in terms of technique. However, they cannot take place at the same time, but must instead be sequenced in a given order due to the limited capacity of resources, such as workspace and manpower. This proposed model takes over evaluating a scenario of such an activity sequence.

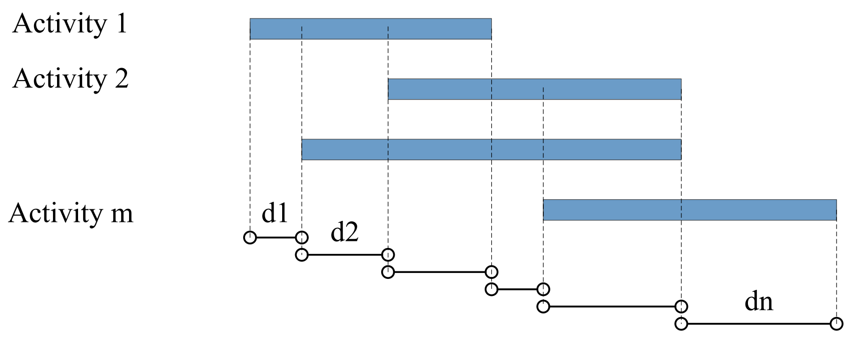
It must be said that the adjustment of activities in a schedule due to workspace conflicts as well as limited capacity of manpower and the evaluation of its results are a dynamic and complex process. Once a workspace conflict occurs, a decision of how to distribute that workspace must be made. This decision then leads to a change in the relations of remaining activities. This complex process therefore makes the adjustment and evaluation not able to be done by an analytical model. Instead, it should be solved by an application of simulation, which is defined, according to Shannon (1998), as the "process of designing a model of a real system and conducting experiments with this model for the purpose of understanding the behavior of the system and/or evaluating various strategies for the operation of the system".

Therefore, the goal of this section is to establish a simulation engine, which is able to handle workspace conflicts and crew overruns whenever they occur, i.e., to decide which activities can occupy the resources competed for and which activities must be suspended or delayed. After this handling, this engine evaluates the goodness of an activity sequence on the basis of several objectives, which are defined in section 4.3.

4.4.2 Simulation framework

In this simulation model, whenever a state of workspace conflict among activities or crew overrun occurs, the model needs an intervention of adjustment. Since the workspaces used in the model are allocated to activities as static positions in the course of its duration, the state of the model, i.e., conflict or non-conflict, only changes when a certain activity either starts or finishes. The state of the simulation model, therefore, is just updated when either the event of starting an activity or the event of finishing an activity occurs.

TIME INTERVAL



STATE AND ACTION

States require an action:

- *workspace conflict*
- *crew overrun*

Actions:

- *move an activity to a late time*
- *interrupt an activity*
- *record a conflict*

Figure 4.2: Properties of the simulation process

During a simulation process, workspace and crew are considered as soft constraints. However, these constraints are not violated unless obeying them will cause an unexpected project delay. This means, generally, a specific workspace and a crew are assigned to only one activity at a given time. If two or more tasks, according to the schedule, require the same space at the same time, then only one task is allowed to occupy this place and the other tasks must be moved to a later time. Only if these tasks do not have enough time to be moved due to the overrun of their total slack and the accepted delay defined in the input data, the conflicts must remain there and the workspace conflict will be considered in the procedure of the schedule evaluation. Similarly, if at a point in time, a crew requirement is beyond a crew's capacity, some tasks must be moved in order to deal with this inadequacy unless they have no time buffers. However, such an algorithm causes problems. Conflicts sometimes seem to pile up at once, where the tasks have no time buffer to be shifted. Of course, a schedule with piled-up congestion is not a feasible solution. Therefore, in order to diversify the results and feasibly distribute conflicts, this approach proposes two simulators: 1) the tense conflict simulation (TCS); and 2) the distributed conflict simulation (DCS). These are described in sections 4.3.1 and 4.3.2. In both TCS and DCS, tasks with earlier start dates will keep a priority of receiving workspace if disputes occur, unless tasks with later start dates have no time buffer.

Figure 4.2 illustrates properties of the simulation process. Time in this process is not discretized into constant intervals. Instead, the interval duration can vary over time depending on the start and end dates of activities involved. It is defined as the duration between the current simulated point and the nearest start or end date of the investigated activities. During the simulation process, there are two states, i.e., workspace conflicts and crew overruns, which require an action of the model to be handled. Once these states occur, the model must decide either to resolve the conflicts by moving or interrupting the other activities to a later time or to record the conflict that cannot be handled.

4.4.3 Tense conflict simulation (TCS)

Several activities in the TCS are interruptible, if needed. Conflicts just occur if the corresponding tasks no longer have time-buffers to be moved. The simulation process can be presented in detail as Figure 4.3.

```
currentDuration is assigned as d1 (figure 4.3);
do
{
    Step 1. List all tasks investigated taking place in currentDuration
    Step 2. If any couple of tasks in the list requires the same workspace & none of them has not been moved to later
    & at least one of their (total slack + acceptable delay) is greater than zero, then
        a. choose the task to be allowed to occupy the workspace based on the following priority
            i. for the case: at least one of them is interruptible
                1. Its property is not interruptible & its start date is earlier than currentDuration
                2. Its (total slack+accepted delay) is zero
                3. Its start date is earlier
                4. Its (total slack + accepted delay) is smaller
                5. If they have the same (3) and (4) values, then choose one of them randomly
            ii. for the case: both of them are not interruptible & at least one of their start dates is within
            currentDuration
                1. Its start date is earlier
                2. Its (total slack + acceptable delay) is smaller
                3. If they have the same (1) and (2) values, then choose one of them randomly
        b. move the not-completed part of the not-chosen task at step (a) to the later time, so that after this step its
        (total slack + acceptable delay) is not smaller than zero and its new start-date is not later than one day from
        the end-date of the chosen one.
        c. update the whole schedule based on its constraints after moving the not-chosen task.
    Step 3. List all tasks investigated taking place in currentDuration (some of tasks have been moved in step 2).
    Step 4. If any crew group is inadequate, then list all the tasks being conducted in this period requiring this
    resource, after that
        a. take a task out of the list if
                i. either its property is not interruptible & its start-date is earlier than currentDuration
                ii. or its (total slack + accepted delay) is zero
        b. for the tasks still existing in the list, choose some tasks to be moved so that the crew requirement is not
        greater than its capability and the number of tasks moved is minimum. In the case that the requirement is
        still greater than its capability after moving all tasks in the list, the crew overruns will be generated.
    Step 5. currentDuration is assigned as nextDuration (figure 4.2)
} while (currentPoint is not yet the lastPoint)
```

Figure 4.3: The simulation process of TCS

4.4.4 Distributed conflict simulation (DCS)

Like the TCS, several activities in the DCS are also interruptible, if needed, during a simulation process. Unlike the TCS, however, a spatial congestion and a crew overrun in the DCS is allowed to occur even if the corresponding tasks still have time buffers to be moved. In order to produce this kind of result, a change within the TCS scheme has been made. In step 2, before choosing a task being allowed to occupy the disputed place, a random number will be created with 20 % probability that they will stay together at that time; hence, the congestion will be reported. Of course, the number of the probability can be adjusted to be less or greater than 20 % but for the experiments so far, 20 % has proven to be the most suitable number in this case. Figure 4.4 illustrates the difference between the results of TCS and DCS.

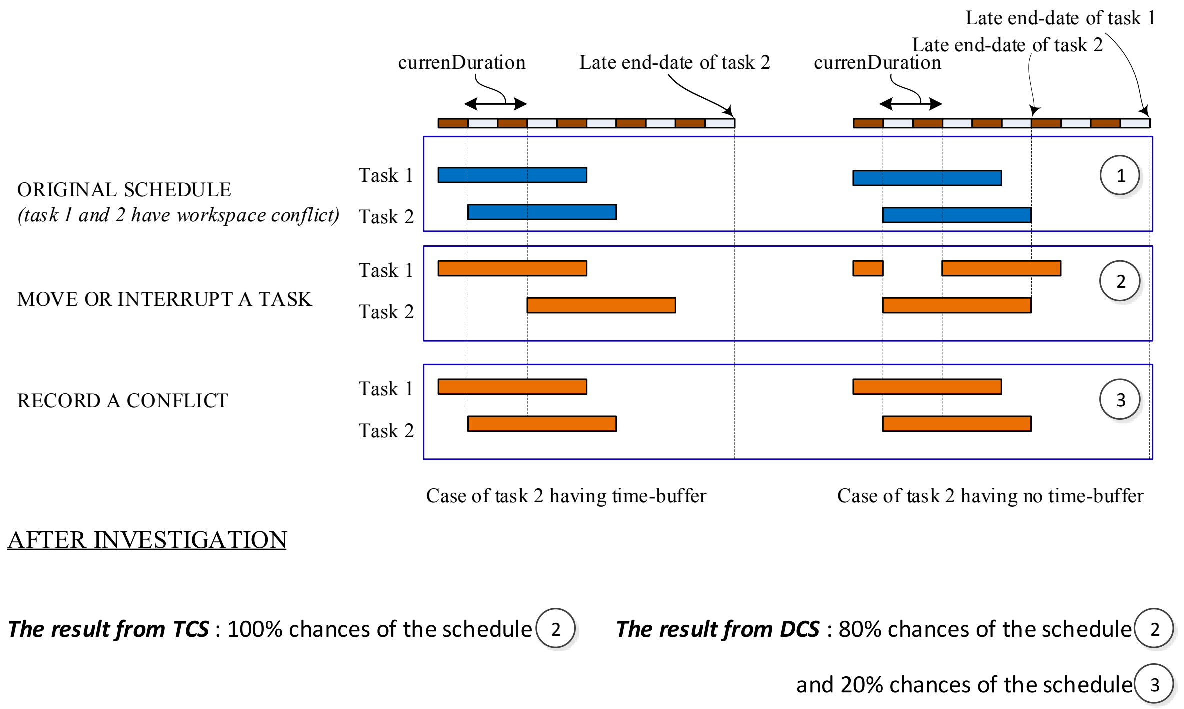

Late end-date of task 1
Late end-date of task 2
currenDuration
Late end-date of task 2
currenDuration
ORIGINAL SCHEDULE
(task 1 and 2 have workspace conflict)
Task 1
Task 2
1
MOVE OR INTERRUPT A TASK
Task 1
Task 2
2
RECORD A CONFLICT
Task 1
Task 2
3
Case of task 2 having time-buffer
Case of task 2 having no time-buffer
AFTER INVESTIGATION
The result from TCS : 100% chances of the schedule 2
The result from DCS : 80% chances of the schedule 2
and 20% chances of the schedule 3

Figure 4.4: Results of the TCS vs. DCS

4.5 Pareto-based optimization for conflict resolution

4.5.1 Choice of optimization model

The optimization model tries to adjust the schedule in order to find the solution which has acceptable values for their objectives, such as project lead time, conflict duration, and number of workers. In order to deal with multiple objectives, there are three principle methods (Mumford-Valenzuela 2005).

(1) Combine all the objectives into one singular scalar value by using weighted factors corresponding to objectives, and optimize for the scalar value.
(2) Arrange the objectives in a priority order, optimize for the first objective, then if there is more than one solution, optimize these solutions for the second objective, and repeat for the third, etc.
(3) Consider all objectives equivalent; find a set of non-dominated solutions, in which when attempting to improve an objective further, the other objectives suffer as a result. This is called Pareto optimal and the set of non-dominated solutions is called Pareto front (Figure 4.5).

The methods (1) and (2) only give one solution. This may ignore many good solutions which do not have the best value for a particular objective, but which make sense if all objectives are considered together. With the idea of providing the best information to site managers, the presented approach chooses the method (3). Such a method involves no pre-judgment, but it produces a set of viable alternatives from which a decision maker can take an informed selection at a later stage.

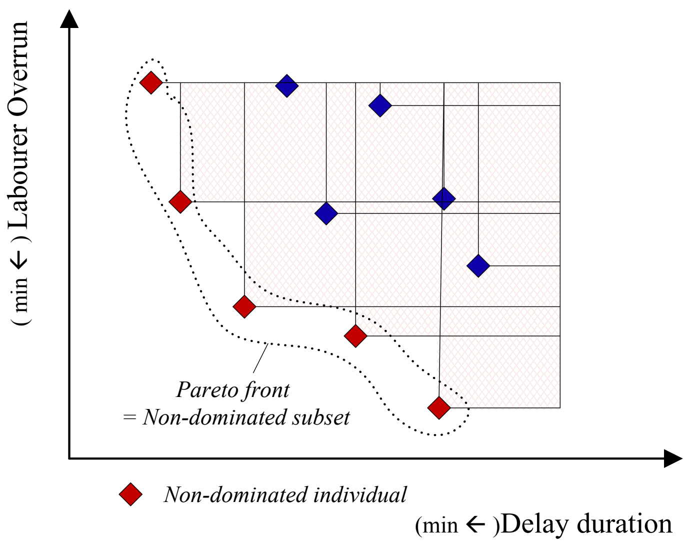


Figure 4.5: Pareto optimality

4.5.2 Definition of objectives

Five parameters are taken into account in this approach. They can be categorized into two groups. The first group, called *project properties* contains: project lead time, conflict duration and crew overrun. The second group, called *feasibility properties*, contains: split number and conflict number. The optimization process aims to keep the values of these five parameters at a minimum.

$$Objective = \text{ to minimize} \begin{cases} -\, project\ properties & \begin{cases} project\ lead\ time\ (1) \\ conflict\ duration\ (2) \\ crew\ overrun\ (3) \end{cases} \\ \\ -\, feasibility\ properties & \begin{cases} split\ number\ (4) \\ conflict\ number\ (5) \end{cases} \end{cases}$$

✦ ***Project lead time***

Project duration is one of the most important factors in construction management. In this approach, project duration is indirectly regarded as being under the parameter *project lead time***.** This parameter, as defined in equation (4.1), is considered an important item to evaluate the adequacy of a schedule. The value of the lead time always lies within a range of acceptable delay duration which should be given as an input data before the investigation process. The longer acceptable delay duration is given, the larger a searching domain is investigated.

***Project lead time** = actual end date of project – as planned end date of project* (4.1)

✦ ***Conflict duration***

Although the manipulation of an activity during simulation is conducted to resolve workspace conflicts, spatial congestion still exists in a schedule depending on the acceptable delay duration. In this research, the *conflict duration* is regarded as the second objective, which must be taken into account by the optimization process. This value is counted as the number of days in which the schedule shows workspace conflicts. A better solution is associated with smaller conflict duration. Equation (4.2) describes the calculation of the conflict duration.

$$\boldsymbol{conflict\ duration} = \sum_{k=1}^{n} d_k \text{, where:} \tag{4.2}$$

- n is the number of occurrences of workspace conflicts
- d_k is the number of days of the k^{th} conflict occurrence.

✦ ***Crew overrun***

In the evaluation process, overrun of labor is also important to evaluate in a schedule. If this information is not taken into account, a good theoretical solution can be achieved by letting all activities of the same trade and with different workspace requirements take place at the same time. Such a schedule, however, is not practical since the number of crews is not infinite. In reality, dealing with the limitation of the number of workers is also a serious problem in construction management on site. Therefore, in this approach, crew overrun is considered as one factor for evaluating solutions. It is defined as the percentage ratio of the total number of laborers exceeding capacity divided by the total number of available laborers.

$$\boldsymbol{crew\ overrun}\ (\%) = \max\left\{\frac{CR_i^k - CC_i^k}{CC_i^k} \times 100\right\};\ i = 1 \div n;\ k = 1 \div m \tag{4.3}$$

Where,

- n is the number of periods in which the laborer requirements are exceeding capacity.
- m is the number of crew groups whose capability is smaller than required.
- CR_i^k is number of Crew Requirements of the k^{th} crew group at the i^{th} duration
- CC_i^k is number of Crew Capacity of the k^{th} crew group at the i^{th} duration

✦ ***Split number***

Beside the three above objectives which present clearly negative consequences of a solution, some other objectives do not but they are very important to identify the feasibility of a schedule. One of them is relevant to the splitting number of a task. An activity once split means that it has to be suspended in order to give its workspace to another task with a higher priority. This may, consequently, lead to a necessary reallocation of equipment and material. Therefore, if a task underlines too many interruptions, the corresponding schedule is not chosen as a feasible solution. In order to take into account this problem, the parameter *split number* will be considered.

$$\textbf{\textit{split number}} = \max\{the\ interruption\ number\ of\ the\ task\ i\}, i = 1 \div n \tag{4.4}$$

There, n is the number of tasks investigated.

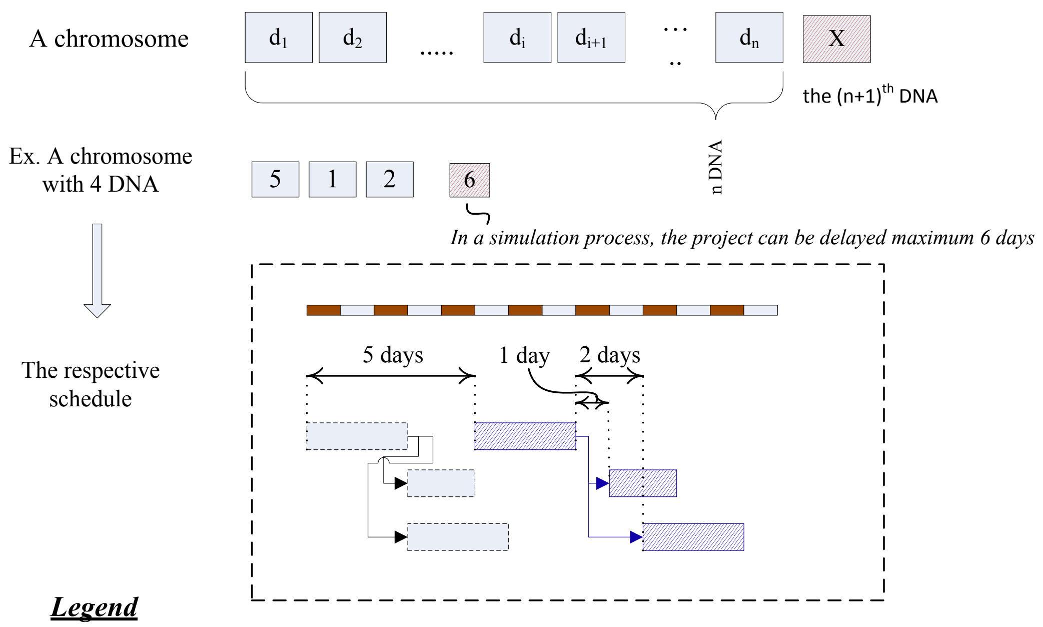
✦ ***Conflict number***

Besides the split number, the number of tasks conflicting at a given time should be kept to a minimum in order to make a solution feasible. The more tasks dispute each other for the same workspace, the more difficulty managers encounter dealing with them. This parameter is called *conflict number*.

$$\textbf{\textit{conflict number}} = max\ \{(number\ of\ tasks\ having\ the\ same\ workspace\ at\ duration\ i)\text{-}1\}\ (i=1\div n) \tag{4.5}$$

There, n is the number of investigated durations, as shown in Figure 4.2.

4.5.3 Chromosome definition



Legend

d_i: the duration counted from the early start date to the actual start date of task i
X: the acceptable delay duration of the project in the simulation process

⬚	*Tasks at the original early start date*
▨	*Tasks at the actual start date*

Figure 4.6: Chromosome

A chromosome is a string of DNA and it is used in this approach to represent an actual part of a schedule (Figure 4.6). The number of DNA in a chromosome is one unit greater than the number of tasks which must be investigated considering workspace and crew requirements. Except the last DNA (denoted by X), which contains information about the acceptable lead time of the project for the simulation process, the others provide the information about the duration which is counted from the early start date to the actual start date of tasks. In this approach, an individual is also considered as a chromosome.

4.5.4 Original generation

A chromosome in the initial generation is created by taking *(n+1)* random numbers. There, n is the number of tasks which is needed to be investigated. For the first n DNA, d_i is defined by a random number which lies within zero and the total slack of the task *i*. For the last DNA, X is identified by a not-negative random number which is not greater than the acceptable lead time of the project which is predefined as an input data.

According to the presented definition of a chromosome, the way to generate a chromosome may cause the project delay duration greater than the X value, after having updated d_i in order to obtain actual dates of tasks. If this occurs, the X will be assigned again with the value of this project delay duration if it is not greater than the acceptable lead time of the project. Otherwise, this chromosome will be taken out of the population. By using the X variable for each chromosome, it assures the diversity of population when the acceptable lead time is great. It should be noted, however, that the greater the acceptable lead time is, a larger the scale of the population should be.

4.5.5 Crossover operator

A simple crossover operator is applied to generate an offspring. From two parent schedules, a crossover position is randomly chosen, the first parent provides the first part of schedule for the offspring, and the second provides the rest. The offspring also takes the last DNA from either of its parents. In this approach, all of the chromosomes in the population join the crossover process. After this process, the offspring has a 10 % probability of being followed by a mutation process before being analyzed and evaluated. Figure 4.7 depicts an example of a crossover process.

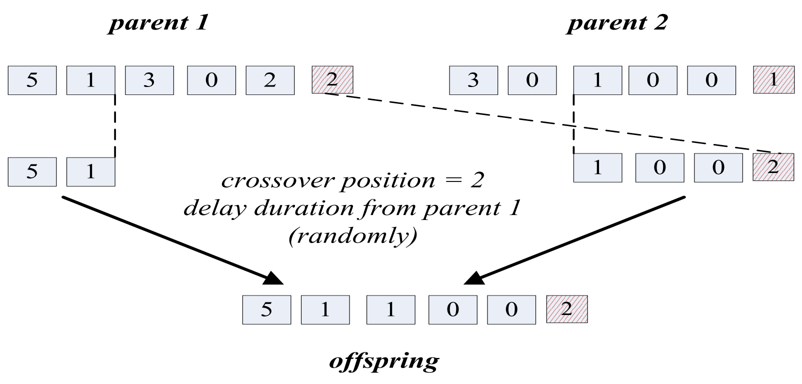


Figure 4.7: An example for crossover process

4.5.6 Mutation operator

In the mutation operation, a position (x), number of tasks (n) and the day to be changed (t) will be randomly created first. Notice that (t) may be either negative or positive. The mutation operator will then be applied for n continuous activities by moving them from the position x by t days uniformly. Figure 4.8 presents an example of a mutation process on a chromosome.

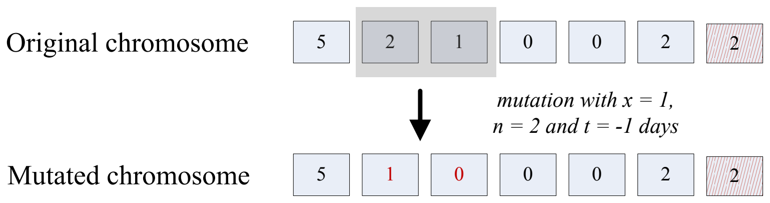


Figure 4.8: An example for mutation process

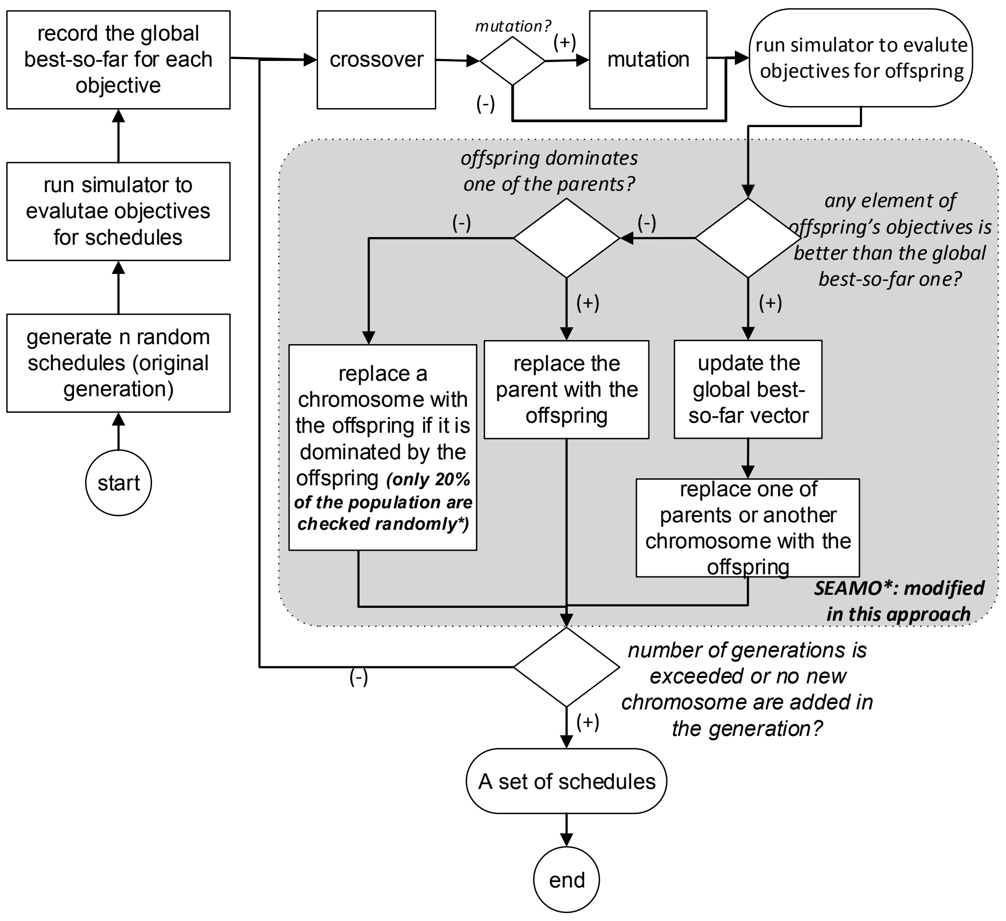
4.5.7 Selection process

In order to generate successive generations which are moved ever closer to the Pareto front, the simple evolutionary algorithm for multi-objective optimization (SEAMO) (Figure 4.9), which was developed by Mumford (2010), is adopted. This algorithm uses a replacement strategy to move the solutions during the searching process ever closer to the Pareto front, and to widen the spread of the solution set. Like its name, this algorithm is kind of simple and as such this property suits the approach. Honestly, the simulation process, which is used to evaluate a chromosome (or rather a schedule in this approach), takes some time. Therefore, a strategy, which is simple but able to generate diversified solutions in the end, has been chosen for this proposed approach.

It should be mentioned that this algorithm however has been modified for this approach. Instead of checking all chromosomes of a population in order to know whether or not a chromosome exists, which is dominated by the offspring, the approach takes randomly the maximum 20 % of a population. In this way, the modification gives chromosomes an equal probability of being replaced regardless of their positions in a population.

4.5.8 Working mechanism of optimization engine

The optimization engine working with the support of simulators is described in Figure 4.9. The process commences with generating an original generation. For each chromosome in this generation, the simulation process is then applied to evaluate its value. The record of the global best-so-far for objectives is created by taking the minimum number of its values from the chromosomes evaluated. After this, all chromosomes take part in the crossover process and 10 % of them continue with mutation to generate offspring. Once an offspring is born, it will be evaluated by a simulator and pass through the selection process to know whether or not it can remain by replacing an existing chromosome in the population. The optimization ends if the possible number of evaluated generations is reached, or if no further new chromosome comes into the population.



start
generate n random schedules (original generation)
run simulator to evalutae objectives for schedules
record the global best-so-far for each objective
crossover
mutation?
(+)
(-)
mutation
run simulator to evalute objectives for offspring
offspring dominates one of the parents?
any element of offspring's objectives is better than the global best-so-far one?
(-)
(-)
(+)
(+)
replace a chromosome with the offspring if it is dominated by the offspring (only 20% of the population are checked randomly*)
replace the parent with the offspring
update the global best-so-far vector
replace one of parents or another chromosome with the offspring
SEAMO*: modified in this approach
number of generations is exceeded or no new chromosome are added in the generation?
(-)
(+)
A set of schedules
end

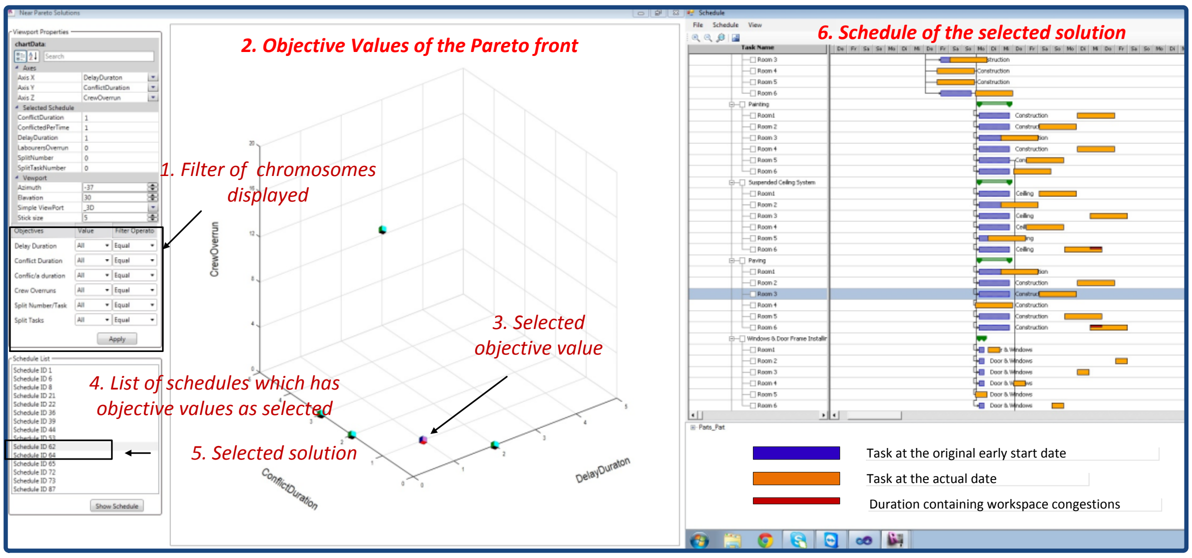
Figure 4.9: Optimization Engine

It should be stated again that two simulators are applied in the approach (TCS and DCS). If both of them are used for one population, for example by choosing a simulator to analyze a schedule randomly, the population required must be large and it takes a long time to obtain the converged solutions. Therefore, two independent populations have been used. One uses TCS and the other uses DCS in the analyzing and evaluating process of a schedule. With these two parallel optimization processes, the results will then be combined and a filtering process will be applied in order to take dominated chromosomes out of the population. As a result, a set of schedules which is considered near Pareto front is presented.

4.6 Prototype implementation (4Dconflict)

4.6.1 Introduction of 4Dconflict

Referring to the method developed in sections 4.4 and 4.5 a prototype implementation is developed to resolve possible spatial conflicts, which is called “4Dconflict”. 4Dconflict is conducted based on the result proposed by 4Dbreakdown. This module is written in C# with about 2,000 lines of code that can be found in the attached CD with this dissertation. Besides resolving conflicts, 4Dconflict also attempts to provide a user interface to evaluate proposed schedules with a three dimensional graph and an objective filter. Figure 4.10 depicts a user interface that 4Dconflict offers to evaluate the feasibility and suitability of proposed solutions.



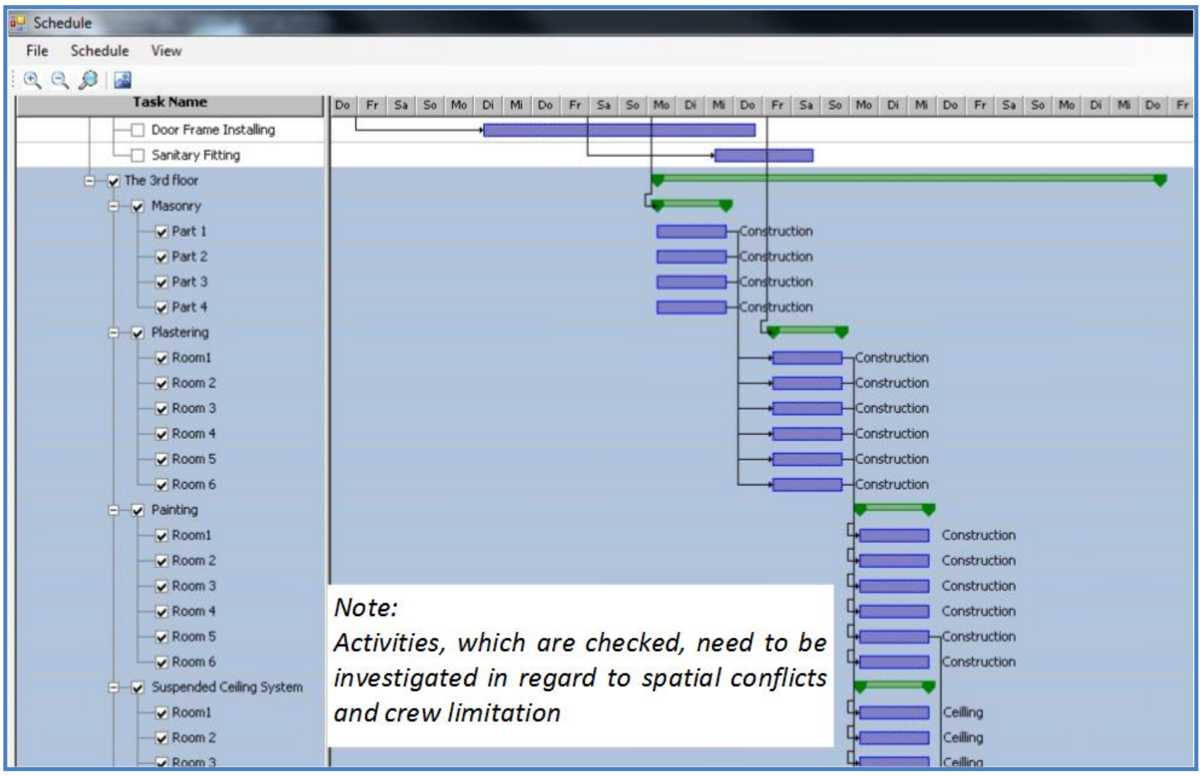
2. Objective Values of the Pareto front
1. Filter of chromosomes displayed
3. Selected objective value
4. List of schedules which has objective values as selected
5. Selected solution
6. Schedule of the selected solution
CrewOverrun
ConflictDuration
DelayDuraton
Task at the original early start date
Task at the actual date
Duration containing workspace congestions

Figure 4.10: User interface for investigating proposed solutions

4.6.2 Case study description

The proposed method for conflict resolution is experimented in the finishing period of a building floor. The trades involved in this experiment include masonry, plastering, painting, installing suspended ceiling systems, installing windows and doors, paving and installing sanitary facilities. The masonry trade contains 4 activities which correspond to 4 regions of the walls; the sanitary fitting is completed with only one activity; the others are divided into 6 activities corresponding to 6 different rooms. In summary, 35 activities are investigated.

Figure 4.11 describes a part of the original schedule. This schedule is derived from the process of detailed breakdown of a schedule. All of CBRs were released before launching the 4Dconflict to resolve possible spatial conflicts. The optimization will be conducted with an acceptable lead time of 4 days, a population including 50 chromosomes and 5 generations.



Note:
Activities, which are checked, need to be investigated in regard to spatial conflicts and crew limitation

Figure 4.11: The original schedule

4.6.3 Alternatives for conflict resolution

Alternatives for conflict resolution are presented as a set of schedules. Each value vector of objectives may contain several schedules. It should also be noticed that an objective vector contains 5 parameters, but the graph can only show the maximum of three values in the three dimensional space at a given time. Therefore, it is necessary to use the filter and the axis data setting to enable investigation of all aspects of the solutions.

According to the objective vectors in the graph, the near Pareto-front solutions have the minimum delay duration of zero and the maximum delay of two days. With two days of delay, the schedule has no workspace conflicts and crew inadequacies. The following resulting schedules are considered in order to analyze the efficiency of solutions from the perspective of a site manager.

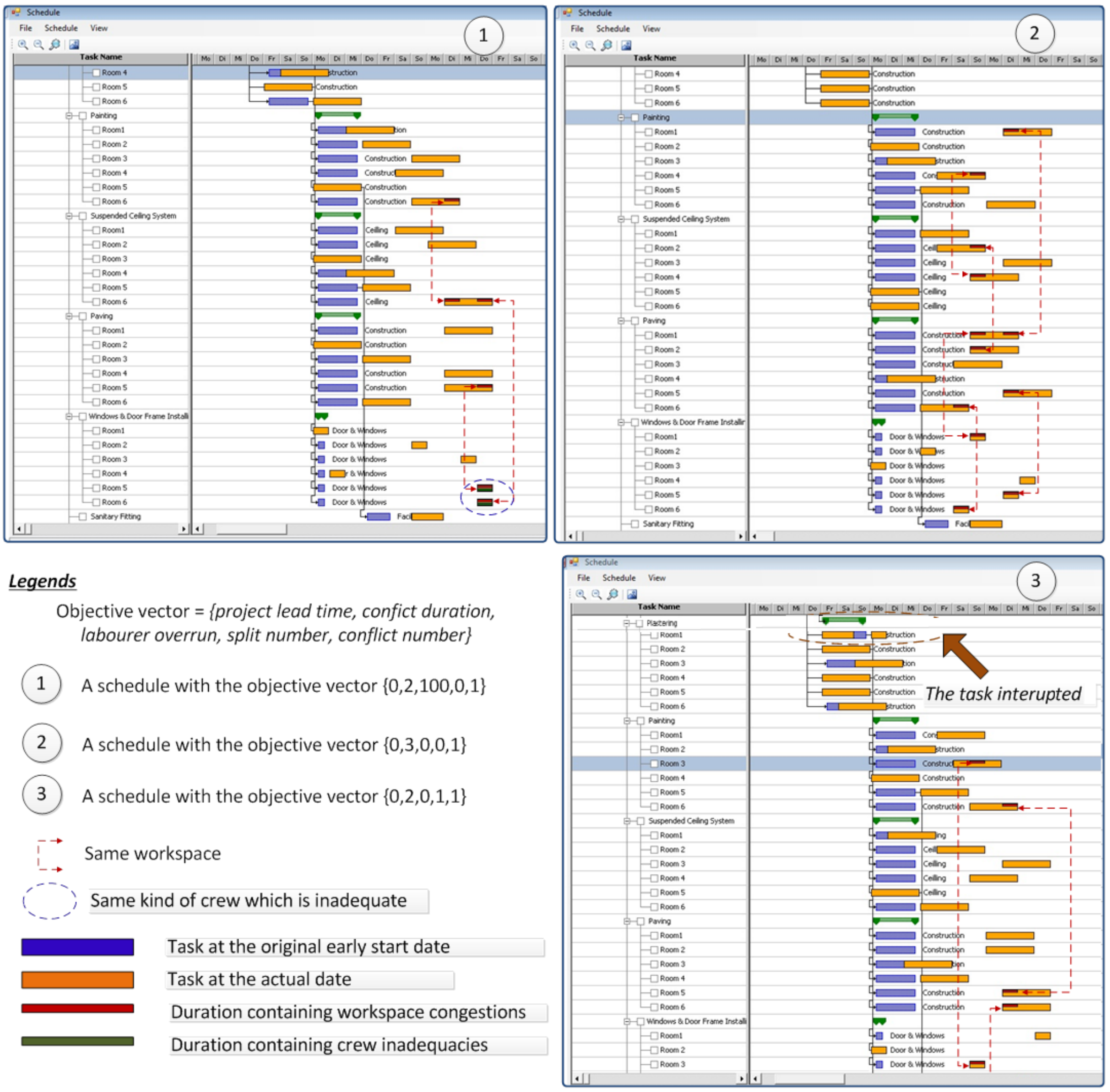

Schedule
File Schedule View
Task Name
1
2
3
The task interupted
Legends
Objective vector = {project lead time, confict duration, labourer overrun, split number, conflict number}
1 A schedule with the objective vector {0,2,100,0,1}
2 A schedule with the objective vector {0,3,0,0,1}
3 A schedule with the objective vector {0,2,0,1,1}
Same workspace
Same kind of crew which is inadequate
Task at the original early start date
Task at the actual date
Duration containing workspace congestions
Duration containing crew inadequacies

Figure 4.12: Solutions without delay

Case 1: the delay duration is zero (Figure 4.12)

For the schedule number 1, the problems which managers face include both workspace congestion and the inadequacy of the crew for installing windows and doors, exceeding the capacity by 100 % (which in this case means two laborers). However, if the trade plastering at room 1 could be interrupted, as shown in *the schedule number 3*, then the crew overrun would not occur. Then, the only problem to be dealt with is the workspace dispute.

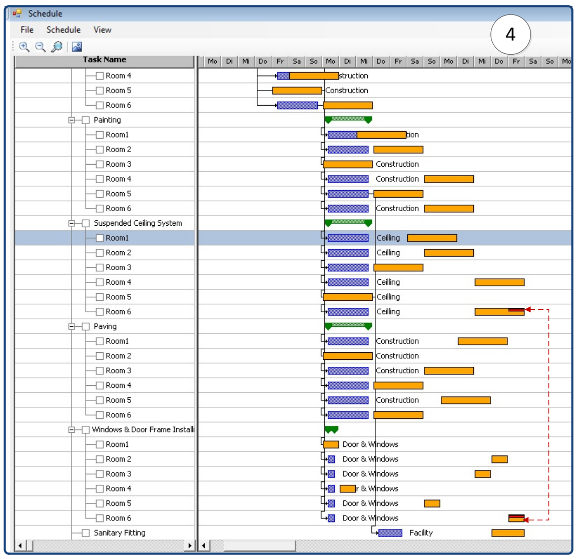
For the schedule number 2, the crew requirements always remain within their capability, i.e., no interruption of tasks is required. However, this schedule brings many spatial disputes, especially, the conflict between installing suspended ceiling systems and paving. This conflict is very serious since both of them normally require a whole room for their work. So this solution is not really feasible.

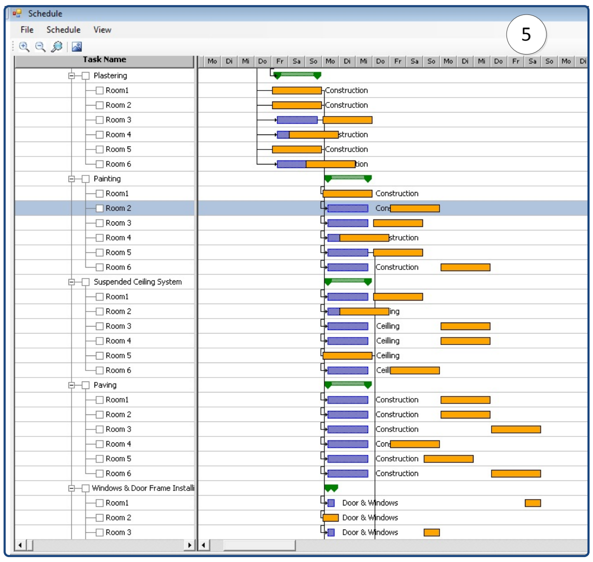
Case 2**:** the delay duration is one day (Figure 4.13, the schedule number 4).

For the schedule number 4, congestion occurs only on one day between the trades installing suspended ceiling and installing windows and doors. If a one-day delay can be accepted, this solution is favorable since this kind of congestion can be solved on site.

Case 3: the delay duration is two days (Figure 4.13, the schedule number 5)

For the schedule number 5, in the event that a two-day delay for the project is acceptable, this schedule is also feasible. It has no inadequacies of crews, no tasks must be interrupted and no time-space conflicts occur during the construction process.





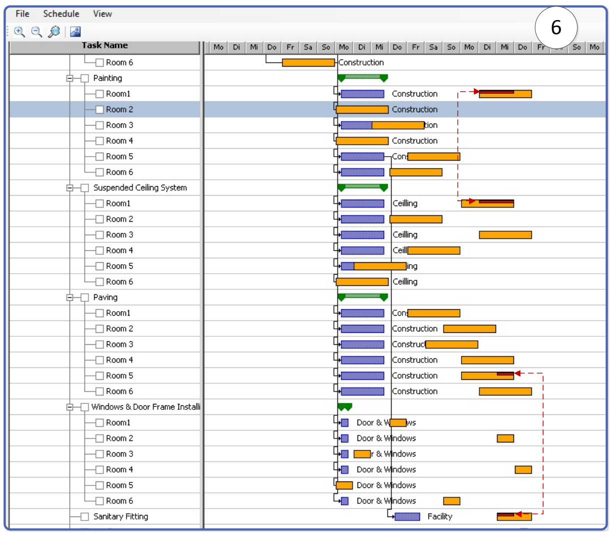
4 One of schedules having the objective vector {1,1,0,0,1}

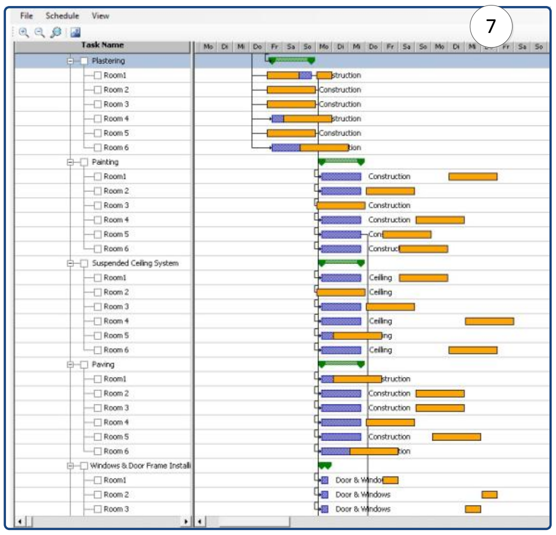
5 One of schedules having the objective vector {2,0,0,0,0}

Figure 4.13: Solution with one day delayed (4) and two days delayed (5)

4.6.4 Evaluation of 4Dconflict

The case study presented here has been experimented with just 5 generations for the evolutionary process. The result would be better if this case were to be carried out with 10 generations. Figure 4.14 depicts two schedules proposed by the 10-generation experiment. These two schedules are better than the results proposed by the 5-generation experiment, i.e., the schedule number 6 dominates the schedules number 1, 2 and 3; and the schedule number 7 shows a more feasible solution than the schedule number 4 although they have the same on-day delay.





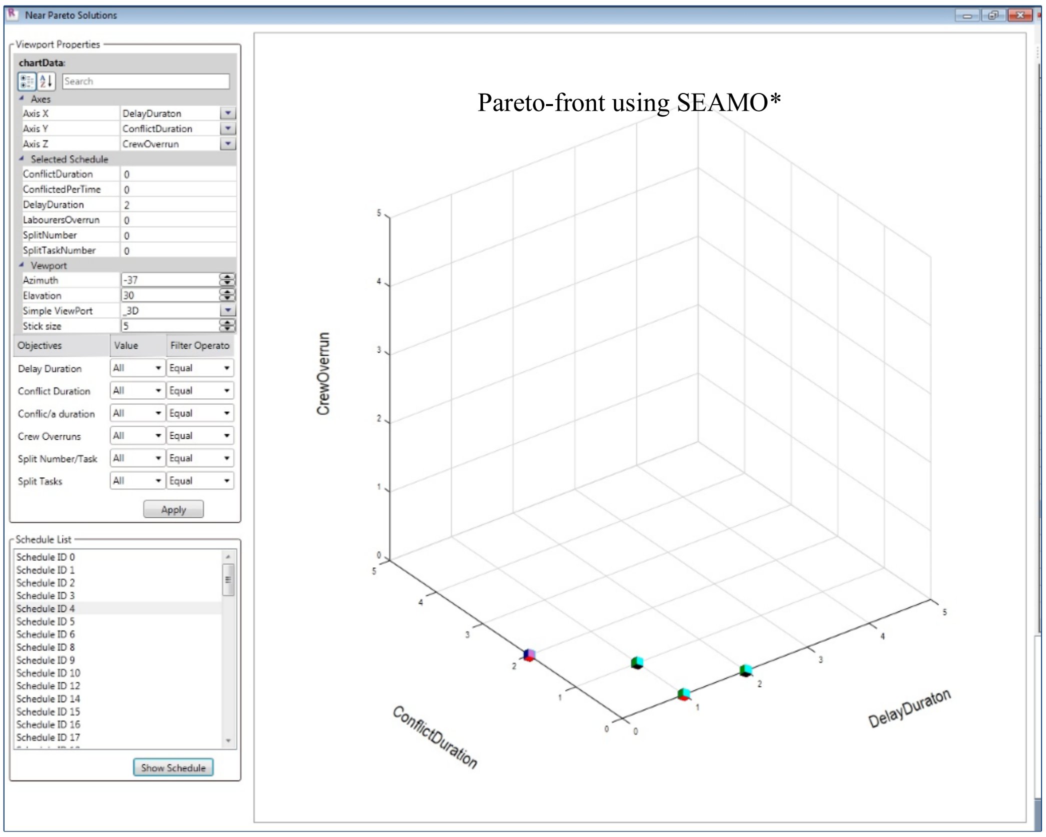
6 One of schedules having the objective vector {0,2,0,0,1}

7 One of schedules having the objective vector {1,0,0,1,0}

Figure 4.14: Solutions of an optimization with 10 generations

Another issue, which should also be regarded here, is how better the SEAMO* has been worked into the approach compared to a random searching algorithm. An experiment with a random search has then been implemented to reveal this matter. For the SEAMO*, the experiment has been conducted with a population containing 50 chromosomes for 10 generations. For random searching, the experiment has been conducted with a population of 12250 chromosomes; this number is equal to the maximum number of chromosomes, which has been checked in the experiment with SEAMO*.

The results have confirmed the efficiency of using SEAMO* (Figure 4.15) compared to a random search (Figure 4.16). With the SEAMO*, the results have converged and the maximum delay duration is just two days in order to get other parameters' values to zero; whereas by using random searching, the corresponding delay is three days. In addition, in case the delay duration and laborer overrun remain zero, schedules generated from SEAMO* show two days of conflict duration; however, with the delay duration zero, the minimum conflict duration in random searching comes up to five days. With just two objective vectors extracted from the results, it is enough to recognize that the optimization using SEAMO* definitely works more efficiently than random searching.


Near Pareto Solutions

Viewport Properties

chartData:

Axes

Axis X	DelayDuraton
Axis Y	ConflictDuration
Axis Z	CrewOverrun

Selected Schedule

ConflictDuration	0
ConflictedPerTime	0
DelayDuration	2
LabourersOverrun	0
SplitNumber	0
SplitTaskNumber	0

Vewport

Azimuth	-37
Elavation	30
Simple ViewPort	_3D
Stick size	5

Objectives	Value	Filter Operato
Delay Duration	All	Equal
Conflict Duration	All	Equal
Conflic/a duration	All	Equal
Crew Overruns	All	Equal
Split Number/Task	All	Equal
Split Tasks	All	Equal

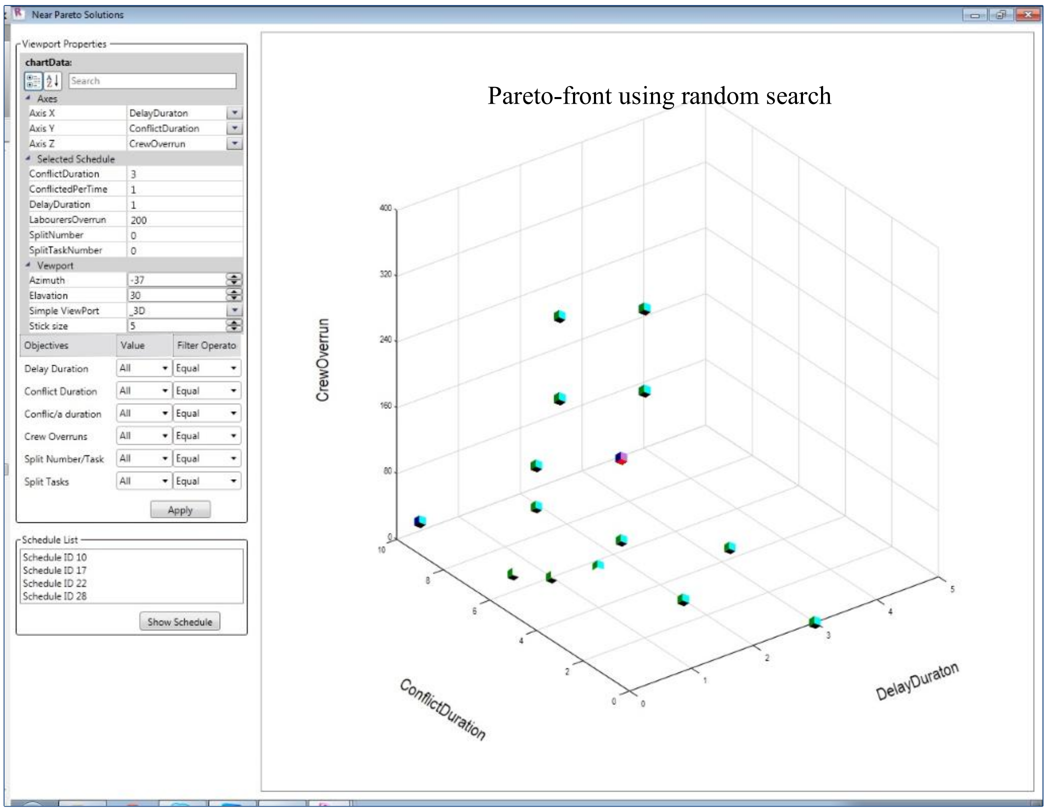
Apply

Schedule List

Schedule ID 0
Schedule ID 1
Schedule ID 2
Schedule ID 3
Schedule ID 4
Schedule ID 5
Schedule ID 6
Schedule ID 8
Schedule ID 9
Schedule ID 10
Schedule ID 12
Schedule ID 14
Schedule ID 15
Schedule ID 16
Schedule ID 17

Show Schedule

Figure 4.15: Pareto-front based on SEAMO*


Near Pareto Solutions

Viewport Properties

chartData:

Axes

Axis X	DelayDuraton
Axis Y	ConflictDuration
Axis Z	CrewOverrun

Selected Schedule

ConflictDuration	3
ConflictedPerTime	1
DelayDuration	1
LabourersOverrun	200
SplitNumber	0
SplitTaskNumber	0

Vewport

Azimuth	-37
Elavation	30
Simple ViewPort	_3D
Stick size	5

Objectives	Value	Filter Operato
Delay Duration	All	Equal
Conflict Duration	All	Equal
Conflic/a duration	All	Equal
Crew Overruns	All	Equal
Split Number/Task	All	Equal
Split Tasks	All	Equal

Apply

Schedule List

Schedule ID 10
Schedule ID 17
Schedule ID 22
Schedule ID 28

Show Schedule

Figure 4.16: Pareto-front based on a random search

4.7 Recommendation for using conflict resolution framework

The goal of the approach was to find a set of feasible strategies which resolve time-space conflicts and limits of crews. The integration of simulation with evolutionary algorithm has been successfully achieved. Like other results from random searching techniques, different schedules generated by the proposed method in this research are sufficiently diversified and the searching process also converges quickly. Therefore, decision makers can choose the suitable solutions depending on their individual conditions such as crew size, and material quality and quantity, etc.

Besides, site managers are able to evaluate solutions efficiently and make their decisions based on the proposed methodology either for the whole schedule or for a selected part of the schedule (concerning short term activities). However, the use of this methodology is recommended for short term activities. Investigating a whole project is quite time consuming and still requires much detailed information that is difficult to be exact at a far too early stage. Moreover, a detailed schedule for a whole project in one run restrains the flexibility as well as reactions to changed conditions, and therefore is not feasible in practice.

5 CONCLUSIONS AND FUTURE WORK

„Everything is possible. The impossible just takes longer"

-- Dan Brown--

This dissertation developed a 4D approach for the automation of detailing a schedule. Based on the assessment of scheduling techniques and the limitation of state-of-the-art theory and practice, this dissertation focused on spatial organization for multi-story buildings in the finishing phase of execution. As a result, the automated frameworks for generating workspaces, breaking a schedule down, and resolving workspace conflicts with their prototype implementations were proposed. This chapter makes a summary of the contributions of this research, discusses their limitations and recommends various directions for future work.

5.1 Research contributions

5.1.1 Contribution on workspace generation

This research has developed a novel structure and a framework for workspace allocation. According to the possible number for workspace allocations, it has categorized trades into one-surface and the two-surface types. That is to say, the way to allocating the workspaces to objects is not only based on the objects themselves, but is also based on the types of trades associated with activities.

The workspace generation for one-surface trades, such as plastering and painting, has faced the challenge of identifying a workspace orientation. Previous research had used global and local coordinate systems to promote the identification of a workspace orientation to its associated object, which required users to define the orientation for each task manually. Taking into account hundreds of activities in a schedule, this method is cumbersome and does not support the automation of workspace generation. In contrast, this dissertation has proposed using rooms as a reference parameter to identify the orientation of workspaces to their objects in the generation of workspaces for one-surface trades. As a result, the orientation of workspaces must not be created in input data. Instead, it can automatically be realized via the 3D model.

The workspace generation for two-surface trades, such as masonry and drywall framing, provides two possibilities for allocating required workspaces for each object. Such a methodology promotes planners to plan and utilize the spatial organization for the crews involved in working on a group of objects efficiently. In doing so, work interruption, disturbances and time-space conflicts can be avoided, reduced or resolved beforehand.

5.1.2 Contribution on schedule breakdown

This research has developed a novel method and a tool for breaking down a schedule. The breakdown process includes the decomposition of activities into smaller sub-activities and the connection of sub-activities by relationships. During the break down process, spatial organization must be considered in order to ensure the sub-activities are more manageable. Furthermore, this research indicated in an earlier section that the current relationships used in scheduling had not supported the automation of breaking down a schedule. Thus, the combination of the decomposition patterns concerning spatial organization and the set of advanced relationships have been developed to promote the automation of breaking down a schedule.

The decomposition patterns have been developed for various levels of detail. In each pattern, this dissertation analyzes the characteristics of trades, the shapes and positions of associated objects as well as the prevalent workflows on site. This analysis provides an informed overview of execution strategies, which can be applied for the decomposition process, and relationships, which can be used to reflect the constraints between sub-activities. Notably, the impact of the special geometric characteristics of buildings, such as the existence of courtyards and openings, is also taken into account to ensure that the sub-activities created after the decomposition process are executed in a more productive manner.

The 4DRs have been developed to enable the automation of sequencing sub-activities. The decomposition of constrained activities in terms of geometry might result in the release of constraints among their sub-activities. However, traditional relationships cannot reflect this case since they do not take into account the associated construction objects. This research has developed 4DRs that are able to consider the 3D objects as a parameter. As a result, they can automatically realize which constraints between sub-activities must be set to *active* and which ones must be set to *inactive*.

5.1.3 Contribution on conflict resolution

This research has proposed a framework and a tool for workspace planning to resolve workspace conflicts. Obviously, resolving time- space conflicts is hardly to be stated as a matter of a "one-size-fits-all" solution, but rather should be enunciated as multiple alternative solutions in order to increase the adaptability of results to the uncertainties of construction projects. Therefore, this research has proposed an integration of simulation and Pareto-based optimization for resolving workspace conflicts and providing the multiple possible solutions. It should be noted that the proposed framework cannot only be used for resolving time space conflicts but can also be applied in resolving conflicts among other resources.

The simulators developed in this research manipulate activities whenever a spatial conflict or crew overrun arises. In order to resolve possible conflicts, the system can delay an un-started activity or suspend an already-started activity by splitting it and then delaying the remaining part. If a conflict is not resolved by the manipulation of execution dates of activities due to the limitation of floating time, the conflict is recorded as a negative point of the schedule and is considered during the evaluation process.

The optimization algorithm developed in this research is a kind of evolutionary algorithm (EA). In order to produce good and feasible schedules, the set of five objectives that must be considered during optimization has been proposed: 1) project lead time; 2) conflict duration; 3) crew overrun; 4) split number; and 5) conflict number. The optimization uses the result of simulation processes to evaluate the effectiveness of a schedule based on these proposed objectives. The results of the application example in Chapter 4 have proved the efficiency of the optimization engine in moving the population further close to a Pareto-front. As a result, a set of executable schedules is proposed to provide informed solutions to site managers.

5.1.4 Contribution on scheduling techniques

This research has proposed a novel set of relationships: 4DRs, CBRs and EBRs. Here, 4DRs illustrate the geometric constraints, CBRs reflect the limited capacity of crew and EBRs present a sequence of activities created due to an execution strategy. This relationship set makes a schedule more understandable and also improve 4D-scheduling techniques.

Understanding the relationships between activities has been a challenge for site managers. Since a schedule is developed by a schedule generator which presumes a set of assumptions about the technical and organizational aspects of a project, schedule users and participants on a construction site will face big challenges in understanding the overall activity relationships in a schedule. As a consequence, in various cases, it is not easy to adjust a schedule whenever a deviation between as-planned and as-built occurs. Therefore, the application of the set of 4DRs, CBRs and EBRs promises to overcome this shortcoming and makes a schedule more transparent and understandable.

The 4D-scheduling technique has still so far relied on the traditional relationships, which have parameters merely based on time. As a result, if the objects of two activities are adjusted, the relationship between them cannot reflect this change. This research has proposed 4DRs, which are able to consider the objects of the attached activities as a parameter. Thus, any adjustment of these objects over time might result in a change of the working status of this relationship. This ability improves the 4D-scheduling technique to bring the power of 3D models into full play. This is to say that a 4DR is not only useful in the automation of breaking down activities, but it can also be applied in 4D-scheduling to facilitate the automation of updating a schedule.

5.2 Limitation and future work

This research has proposed an automated model to detail a schedule considering spatial organization. However, scheduling in construction projects does not only involve workspace availability, it is also concerned with financial and other resource limitations such as materials, manpower and equipment. It is stated at an earlier stage of this research that workspace availability is hardly changed regardless how strong contractors are. So workspace constraints should be taken into account first before considering other resources. Thus, workspace constraints are also the key priority of this research. The contribution of this research therefore is the necessary first step towards the automation of generating a feasibly detailed planning from a rough schedule.

Nevertheless, the proposed method in this research is able to be applied to handle conflicts associated with other resource limitations. It is also extended to find good schedules to adapt to the capacities of other factors by further filtering the proposed results. Otherwise, an expansion of the considered objectives during optimization, such as inquiring into further material and financial capacities, can be conducted to adapt its results to more constraints.

In a further direction, an approach of the automated generation of a rough schedule should be improved based on the models. The combination of such an approach and this proposed research promises a totally automated generation of schedules at different levels of detail from a BIM-based 3D model in a near future.

REFERENCES

Akbaş, Ragıp 2004. "Geometry-based Modeling and Simulation of Construction Process."Doctoral Dissertaion, Department of Civil and Environmental Engineering, Standford University.

Akinci, Burcu, Martin Fischen, Raymond Levitt, and Robert Carlson. 2002. "Formalization and Automation of Time-Space Conflict Analysis." *Journal of Computing in Civil Engineering* 16 (2):124-134.

Akinci, Burcu, Martin Fischer, and John Kunz. 2002. "Automated Generation of Work Spaces Required by Construction Activities." *Journal of Construction Engineering and Management* 128 (4):306-315.

Arditi, D., and M. Albulak. 1986. "Line-of-Balance Scheduling in Pavement Construction." *Journal of Construction Engineering and Management* 112 (3):411-424. doi: doi:10.1061/(ASCE)0733-9364(1986)112:3(411).

Bansal, V. K. 2011. "Use of GIS and Topology in the Identification and Resolution of Space Conflicts." *Journal of Computing in Civil Engineering* 25 (2):159-171.

Bargstädt, Hans- Joachim, and Amir Elmahdi. 2010. "Simulation von Bauprozessen - ein Qualitätssprung in der Arbeitsvorbereitung." 8. Grazer Baubetriebs- und Bauwirtschaftssymposium Graz Tecnhnische Universität Graz, Germany.

Billaut, Jean-Charles, Aziz Moukrim, and Eric Sanlaville. 2008. *Flexibility and Robustness in Scheduling*: John Wiley & Sons, Inc.

Dang, Trang, Amir Elmahdi, and Hans Joachim Bargstädt. 2012. "Generating Workspace Requirements in a finishing execution phase." 12th International Conference on Construction Application of Virtual Reality, Taiwan.

Dawood, Nashwan, and Zaki Mallasi. 2006. "Construction Workspace Planning: Assignment and Analysis Utilizing 4D Visualization Technologies." *Computer-Aided Civil and Infrastructure Engineering* 21 (7):498-513. doi: 10.1111/j.1467-8667.2006.00454.x.

de Vries, Bauke, and Jeroen M. J. Harink. 2007. "Generation of a construction planning from a 3D CAD model." *Automation in Construction* 16 (1):13-18. doi: http://dx.doi.org/10.1016/j.autcon.2005.10.010.

Dong, Ning, Martin Fischer, Zuhair Haddad, and Raymond Levitt. 2013. "A method to automate look-ahead schedule (LAS) generation for the finishing phase of construction projects." *Automation in Construction* 35 (0):157-173. doi: http://dx.doi.org/10.1016/j.autcon.2013.05.023.

Echeverry, Diego. 1991. "Factors for generating initial construction schedules." Ph.D., Civil and Environmental Engineering, University of Illinois at Urbana-Champaign.

Elmahdi, Amir. 2013. *Grid Based Simulation Model for Workspace Management and Analysis*. Germany: Bauhaus Universität Weimar. Ph.D. Dissertation.

Elmahdi, Amir, and Hans- Joachim Bargstädt. 2011. "Knowledge-based Simulation Framework for Workspace Modelling." International Conference on Computing in Civil and Building Engineering, Moscow.

Elmahdi, Amir, I-Chen Wu, and Hans- Joachim Bargstädt. 2011. "4D Grid-based simulation framework for facilitating workspace management." 11th International Conference on Construction Applications of Virtual Reality, Weimar Germany.

Gummere, Richard M 1917. Epistle 6. On sharing knowledge. In *Seneca Epistles 1-65*: Harvard University Press.

Hanna, Award S., Jeffrey S. Russell, and Erik O. Emerson. 2008. "Stacking of Trades." In *Construction Productivity: A Practical Guide for Building and Electrical Contractors*, edited by Eddy M. Rojas, 75-110. J Ross Publishing.

Hendrickson, Chris. 2008. "Project Management for Construction: Fundamental Concepts for Owners, Engineers, Architects and Builders." Accessed 07.04.2014. http://pmbook.ce.cmu.edu/09_Construction_Planning.html.

Ivakhnenko, A. G. 1970. "Heuristic self-organization in problems of engineering cybernetics." *Automatica* 6 (2):207-219. doi: 10.1016/0005-1098(70)90092-0.

Jongeling, Rogier. 2006a. *A Process Model for Work-flow management in construction: Combined Use of Location-based scheduling and 4D CAD*. Sweden: Luleå University of Technology. Doctoral Dissertation.

Jongeling, Rogier. 2006b. "A Process Model for Work-flow management in construction: Combined Use of Location-based scheduling and 4D CAD."Doctoral Dissertation, Department of Civil and Environmental Engineering- Division of Structural Engineering, Luleå University of Technology.

Jongeling, Rogier, and Thomas Olofsson. 2007. "A method for planning of work-flow by combined use of location-based scheduling and 4D CAD." *Automation in Construction* 16 (2):189-198. doi: 10.1016/j.autcon.2006.04.001.

Kerzner, Harold. 2009. *Project Management: A System Approach to Planning, Scheduling, and Controlling*. Canada: John Wiley & Sons, Inc.

Kim, Hyunjoo, Kyle Anderson, SangHyun Lee, and John Hildreth. 2013. "Generating construction schedules through automatic data extraction using open BIM (building information modeling) technology." *Automation in Construction* 35 (0):285-295. doi: http://dx.doi.org/10.1016/j.autcon.2013.05.020.

König, Makus, and Michael Baling. 2010. "Wissensbasierte Planung von Bauprozessen." In *Mefisto: Management-Fürung-Information-Simulation im Bauwesen*, edited by Raimar J. Scherer and Sven-Eric Schapke, 67-78. Dresden, Germany.

Mallasi, Zaki. 2006. "Dynamic quantification and analysis of the construction workspace congestion utilising 4D visualisation." *Automation in Construction* 15 (5):640-655. doi: 10.1016/j.autcon.2005.08.005.

Mallasi, Zaki, and Nashwan Dawood. 2001. "Assessing space criticality in sequencing and identifying execution patterns for construction activities using VR visualisations." ARCOM doctoral research workshop: Simulation and modelling in construction, Edinburgh University, UK.

Mumford-Valenzuela, Christine L. 2005. "A simple approach to evolutionary multiobjective optimization." In *Evolutionary Multiobjective Optimization - Theoretical Advances and Applications*, edited by Lakhmi Jain Ajith Abraham, Robert Goldberg, 55-104. United States of America: Springer.

Mumford, Christine. 2010. "A multiobjective framework for heavily constrained examination timetabling problems." *Annals of Operations Research* 180 (1):3-31. doi: 10.1007/s10479-008-0490-3.

Patrick, Charles. 2004. *Construction Project Planning and Scheduling*. United States of America: Pearson Education, Inc.

Project Management Institute, Inc. 2007. *The practice standard for scheduling*. United States of America: PMI Publication.

Project Management Institute, Inc. 2013. *A Guide to the Project Management Body of Knowledge: PMBOK Guide*. 5 ed. United States of America: PMI Publication.

Riley, David R., and Victor E. Sanvido. 1995. "Patterns of Construction-Space Use in Multistory Buildings." *Journal of Construction Engineering and Management* 121 (4):464-473.

Riley, David R., and Victor E. Sanvido. 1997. "Space Planning Method for Multistory Building Construction." *Journal of Construction Engineering and Management* 123 (2):171-180.

Shannon, R. E. 1998. "Introduction to the art and science of simulation." Simulation Conference Proceedings, 1998. Winter, 13-16 Dec 1998.

Tauscher, Eike. 2011. "Vom Bauwerksinformationsmodell zur Terminplanung: Ein Modell zur Generierung von Bauablaufplänen."Dotoral Dissertation, Department of Civil Engineering and Construction, Bauhaus Universität Weimar.

Thabet, ., and . Beliveau. 1997. "SCaRC: Space-Constrained Resource-Constrained Scheduling System." *Journal of Computing in Civil Engineering* 11 (1):48-59. doi: doi:10.1061/(ASCE)0887-3801(1997)11:1(48).

Thabet, W., and Y. Beliveau. 1994. "HVLS: Horizontal and Vertical Logic Scheduling for Multistory Projects." *Journal of Construction Engineering and Management* 120 (4):875-892. doi: doi:10.1061/(ASCE)0733-9364(1994)120:4(875).

Tulke, Jan. 2010. "Kollaborative Terminplanung auf Basis von Bauwerksinformationsmodellen."Doctoral Dissertation, Bauhaus Universität Weimar.

Winch, Graham M., and Steve North. 2006. "Critical Space Analysis." *Journal of Construction Engineering and Management* 132 (5):473-481.

Zhang, Cheng, Amin Hammad, Tarek M. Zayed, Gabriel Wainer, and Hong Pang. 2007. "Cell-based representation and analysis of spatial resources in construction simulation." *Automation in Construction* 16 (4):436-448. doi: 10.1016/j.autcon.2006.07.009.

ABBREVIATIONS

3D	three dimensional
4D	four dimensional = 3D + time
4DR	4D relationship
4Dbreakdown	name of the prototype for breakdown of a schedule
4Dconflict	name of the prototype for conflict resolution
4Dworkspace	name of the prototype for workspace generation
BIM	Building Information Model
CAD	Computer-aided design
CBR	Crew-based relationship
DCS	Distributed conflict simulation
EBR	Execution strategy-based relationship
GPM	The process modeling and simulation approach based on geometric models and techniques
LOB	Line of Balance
OBD	Object-based dependency
OSS	operation-specific spatial
SBD	Structure-based dependency
SEAMO	Simple evolutionary algorithm of multi-objective optimization
TCS	Tense conflict simulation

ALGORITHMS

FIGURES

TABLES