

# Decomposition of BIM objects for scheduling and 4D simulation

J. Tulke

HOCHTIEF AG, Essen, Germany

M. Nour & K. Beucke

Informatik im Bauwesen, Bauhaus-Universität, Weimar, Germany

**ABSTRACT:** This paper addresses the common problem of object splitting encountered when a Building Information Model (BIM) is used to support the creation and validation of construction schedules. A generalized splitting algorithm for boundary representations (b-rep) of BIM objects and an IFC based data management concept for refined object granularities are presented.

## 1 INTRODUCTION

As Building Information Models are getting more and more introduced into the building industry practice, the project scheduling can benefit from this new way of working. The well known 4D visualisations of completed schedules are one step in this direction (Heesom & Mahdjoubi 2004). Taking the bill of quantities into account, the BIM approach of working further eases project scheduling by supporting the calculation of individual task durations during the creation of the schedule (Aalami et al. 1998), (Tulke & Hanff 2007).

In both cases the granularity of geometry and quantity information needed is affected by the construction sequences in the schedule. In particular, if several construction schedule alternatives should be generated based on the same BIM, the object granularity must not be coarser than the greatest common partitioning needed by all schedules (Figure 1)

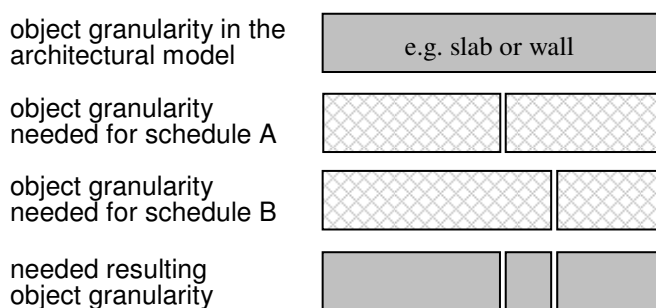


Figure 1: Scheduling impact on object granularity

This often leads to conflicts as also reported by (Haymaker & Fischer 2001), (Reinhardt et al. 2004) and (Aalami et al. 1998) since in common practice

the three information components (the CAD model, the bill of quantities and the construction schedule) are neither generated in a strict sequence nor by the same actors.

At first the CAD model representing the final state of the product is created by an architect or a draftsman. In this phase the only requirement towards the information granularity is the efficiency of model creation. In the second phase the quantity take-off (QTO) specialist adds missing information to the model and assembles the bill of quantities for cost estimation. The project scheduler reuses later these quantities during scheduling to predict the durations of single tasks. Meanwhile, he reuses also the CAD model to visualize scheduling results in a 4D simulation. Resources such as labour and equipment assigned during the scheduling process are later incorporated into the cost estimates (Figure 2).

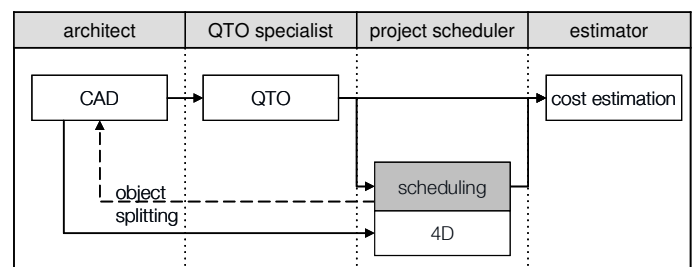


Figure 2: process dependencies

To enable this subsequent utilization of information, by the project scheduler, the model has to be held in a compatible object granularity. However this can not be guaranteed automatically. As a result several iteration loops in model creation and extensive coordination between the three different actors (architect/draftsman, QTO specialist and scheduler)

are needed to reach an object granularity that is suitable for the three purposes. The communication overhead related to this is extremely time consuming and requires round trip modifications between actors who do not know about each other's internal working peculiarities. Consequently, the reuse of information and the adoption of model based working in the scheduling process is difficult to achieve.

To overcome this problem the project scheduler needs to have a simple to use software functionality integrated into his tool set which enables him to adjust the object granularity to his needs without intervening in other domains' way of working. Using CAD software or estimating packages for this purpose is impractical since these software packages are rather complicated and are not tailored to the needs of a project scheduler. In addition, the refined object granularity is domain specific and does not have to be taken over by the architect or estimator. These stakeholders prefer to work with the original object granularity. Thus the adjustment has to be added as optional granularity or object decomposition.

Existing scheduling and 4D simulation software does not provide such a functionality to easily add an adjusted object granularity to the scheduling domain model which is connected to the architectural design model in order to be able to react to any design changes.

### 1.1 Adjustment of object granularity

From a scheduler's perspective five different types of relations between a task in the schedule and objects in the CAD model are possible (Figure 3):

- 1 one task matching no CAD object (e.g. a design activity)
- 2 one task matching a portion of a CAD object (e.g. portion of wall W1 between axis A and B or within a zone Z)
- 3 one task matching exactly one CAD object (one construction element, e.g. a wall)
- 4 one task matching exactly several CAD objects (e.g. all walls in storey three)
- 5 one task matching non to several CAD objects and one to several portions of CAD objects (e.g. all walls in storey three between axis A and B or zone Z)

Whereas type one and three are easy to handle, type four can be mastered by simple aggregation (grouping) of objects without any implication on the object granularity in the CAD model, type two and five can not be dealt with by existing 4D and scheduling software tools since in these cases a splitting of one or several objects is needed.

As the project scheduler reuses the geometry for visualisation and the items from the bill of quantities for the calculation of task durations, both informa-

tion parts have to be refined. The splitting of the geometry thereby is the dominant problem since the refined quantity information can be calculated based on the new geometry parts with the same rules as used during QTO with the original object. Attributes for the part objects like material, manufacture, storey affiliation, etc. can be inherited from the original CAD object. Geometry based attributes have to be recalculated.

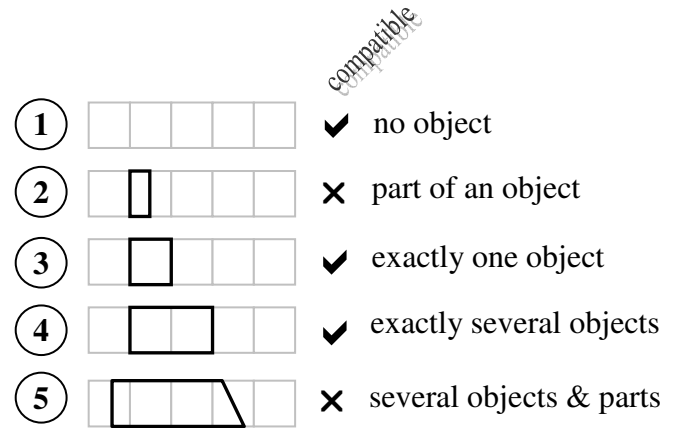


Figure 3: Possible object relations between CAD objects and a task in the schedule.

Once the information refinement is done, the new objects have to be incorporated into the BIM as optional, domain specific object granularity. An appropriate data management concept is needed for this.

### 1.2 Requirements on geometry splitting

Today based on 2D drawings, the project scheduler uses axial grids or zones to address parts of objects. This method is applied to all types of construction elements which are not constructed as a single unit e.g. walls or slabs. Different sections are surrounded with coloured polygons to visualize the construction sequence.

The BIM based scheduling process should support this way of working by enabling automated splitting of three dimensional CAD objects with reference to three dimensional zones. The used algorithm thereby has to be general so that it can be applied to any type of CAD object no matter what type of construction element it represents. The resulting parts of the original object have to be additionally classified as being inside or outside of the clipping zone. In this way the project scheduler is able not only to slice the product according to his needs but also to address the parts inside the clipping zone.

The need for additional user interaction compared to the 2D way of working should be kept minimal. E.g. to easily construct the zone objects, they could be defined by the end-user as a planar polygon. An additional entered extrusion direction and depth

automates the construction of the three dimensional zone objects.

### 1.3 Requirements on data management

As mentioned above, the refined object granularity of the CAD model is a domain specific issue but has implications on the granularity of quantity items. Other design disciplines continue to base their work on the original object granularity. But both object granularities have to stay in relation to enable the project scheduler to react on design changes. This leads to the conclusion that in general, it is not a good practice to save the refined object granularity within the BIM. A better approach would be to use an unevaluated model approach by saving the rule behind the object refinement rather than the new objects themselves. But due to the fact that currently available software packages support explicitly saved object models only the authors found no alternative other than supporting this approach.

To limit the amount of data being stored, it is convenient to store only one refined object granularity compatible with all schedule alternatives instead of saving one specific object granularity for each schedule alternative. That is, the greatest common partitioning derived by a consecutive splitting according to the different schedule needs is stored.

To support open collaboration between the stakeholders in a project the data should be saved based on the open data exchange format IFC.

## 2 OBJECT SPLITTING ALGORITHM

The splitting functionality developed to fulfil the requirements explained above is based on the algorithm for Boolean operations on polyhedral objects as presented by (Laidlaw et al. 1986). This algorithm is used in many constructive solid geometry (CSG) modellers and operates on two objects at a time. In the context of object refinement the first object is the CAD object and the second one is the zone object used to address a model portion.

### 2.1 Principle of Boolean modeller

Three steps have to be taken to calculate the boundary representation of the result of a Boolean operation performed on two objects (Laidlaw et al. 1986):

1. Making the surface meshes of both participating objects compatible by subdividing all intersecting triangles
2. Classification of all triangles as being inside, outside or on the surface of the other object. Triangles which are on the surface of the other object are further classified as having the same or opposite surface normal.

3. Assembling the cubature of the resulting object by selecting triangles from both objects based on their classification and the Boolean operation to be executed (Table 1|Table 1)

Table 1: Triangle selection for assembling of the resulting object of Boolean operations

Operation	Triangles from object A	Triangles from object B
$A \cup B$	outside, same	outside
$A \cap B$	inside, same	inside
$A - B$	outside, opposite	inside*

\* but inverted

As a result of a Boolean operation a single object is always received, even if its cubature consists of strictly separated parts.

### 2.2 Modification for object splitting

Step 1 and 2 within the Boolean modeller are independent of the actual operation and have to be executed only once, even if several Boolean operations are applied on the same objects.

For the object splitting functionality two operations are used on one CAD-zone object pair: The intersection operation is used to produce the object A representing the part of the original object which is inside the cutting zone. The difference operation is used to produce the object B representing the outside part. But still, both objects can contain several separated surface meshes (Figure 4).

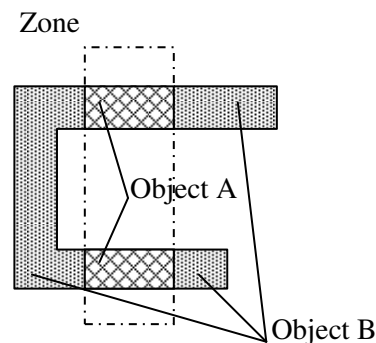


Figure 4: Objects representing the inside and outside parts in reference to the zone.

To fix this situation, the separated surface meshes within object A and B have to be automatically identified and transferred into separated objects.

Unfortunately the data structure commonly used for triangle meshes of b-reps allows direct access to triangle-vertices adjacency information only but not to triangle-triangle connectivity information as needed for identification of disconnected surface meshes.

For each of both objects A and B Boolean path algebra is used to calculate this relationship. Once

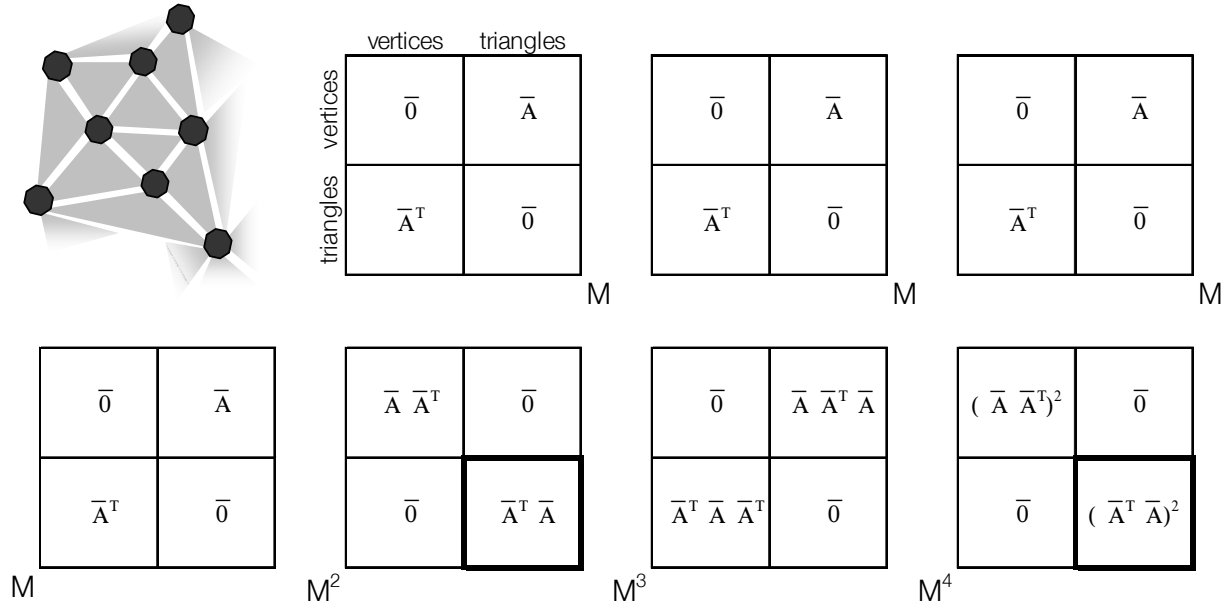


Figure 5: Principle for triangle-triangle connectivity calculation during mesh separation

the triangle-triangle connectivity is known, the set of strictly separated objects classified as inside or outside the zone can be returned as required by the splitting functionality.

### 2.2.1 Mesh separation for one object

The complete initial adjacency information of the object's surface mesh is described by the quadratic Boolean matrix  $M$ . The sub-matrix  $A$  of  $M$  with the dimension  $n \times m$  represents the triangle-vertex relationship (Figure 5). Whereas  $n$  is the number of vertices and  $m$  the number of triangles in the complete surface mesh. Each column of  $A$  represents the vertex adjacency of one triangle and therefore contains true for exact three vertices. Due to symmetry, the vertex-triangle adjacency sub-matrix can be derived as  $\bar{A}^T$ .

The vertex-vertex adjacency sub-matrix which can be derived from the edges of the triangles is not needed and thus neglected.

The triangle-triangle adjacency information is not directly available in a common b-rep data structure. Therefore in the initial adjacency matrix  $M$  this sub-matrix is set to false, too.

That is, triangles are considered to be connected to vertices only (sketch in Figure 5).

The transitive hull  $H$  of  $M$  calculated according to equation (1) contains the complete triangle-triangle connectivity information in the lower right sub-matrix.

$$H = M \cup M^2 \cup M^3 \cup M^4 \cup \dots \quad (1)$$

As can be seen from Figure 5 only an even power of  $M$  contributes to that sub-matrix. That is the triangle-triangle connectivity can be calculated more efficiently as hull  $H_D$ :

$$D = \bar{A}^T \bar{A} \quad (2)$$

$$H_D = D \cup D^2 \cup D^3 \cup D^4 \cup \dots \quad (3)$$

The order of triangles in  $\bar{A}$  can be chosen in a way that  $H_D$  contains true only in quadratic sub matrices around the main diagonal. Each of these quadratic sub matrices represents a separated surface mesh contained in the object.

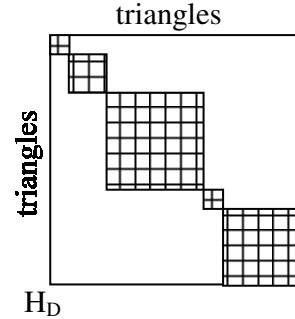


Figure 6: Separated surface meshes within  $H_D$

Thus for one row representing a specific triangle all triangles belonging to the same separated surface mesh can be found in the columns marked with true. The other surface meshes are found in the same way until all triangles have been processed.

### 2.3 Implementation

The splitting functionality was implemented in a test environment based on Java3D and a Boolean modeller implementation from (Castanheira 2003) according to the algorithm from (Laidlaw et al. 1986). The code was enhanced as described above. An extract is presented in Figure 7.

The transitive hull  $H_D$  according to equation (2) and (3) is calculated with the Floyd-Warshall algorithm.

```

Vector<Vector<Integer>> getSeparatedObjects(Shape3D mesh)
{
    ...
    int[] vertexIndices = new int[3*nrFaces];
    ...

    // build triangle-node adjacency matrix A
    boolean[][] tn_adjacence =
        new boolean[nrNodes][nrFaces];
    for (int i = 0; i < nrFaces; i++){
        for (int j = 0; j < 3; j++){
            tn_adjacence[vertexIndices[3 * i + j]][i] = true;
        }
    }

    // calculate adjacency matrix of triangles (AT*A)
    boolean[][] tt_adjacence =
        new boolean[nrFaces][nrFaces];
    for (int i = 0; i < nrFaces; i++){
        for (int j = 0; j < nrFaces; j++){
            for (int k = 0; k < nrNodes; k++){
                if (tn_adjacence[k][i] && tn_adjacence[k][j]){
                    tt_adjacence[i][j] = true;
                    continue;
                }
            }
        }
    }

    // calculate hull
    calculateHull(tt_adjacence);

    // separate objects
    Vector<Vector<Integer>> objects =
        new Vector<Vector<Integer>>();
    // outer vector = objects
    // inner vector = vertex indices (3 per triangle)

    HashSet<Integer> used = new HashSet<Integer>();
    for (int i = 0; i < nrFaces; i++)
    { // row of tt_adjacence
        if (used.contains(i))
            continue;
        Vector<Integer> vertices = new Vector<Integer>();
        for (int j = 0; j < nrFaces; j++)
        { // column of tt_adjacence
            if (tt_adjacence[i][j])
            {
                vertices.add(vertexIndices[3 * j]);
                vertices.add(vertexIndices[3 * j + 1]);
                vertices.add(vertexIndices[3 * j + 2]);
                used.add(j);
            }
        }
        objects.add(vertices);
    }

    return objects;
}

```

Figure 7: Java code example for object separation

In a BIM coordinates of a surface mesh are often formulated in local coordinate systems. In this case both objects (the CAD and zone object) have to be transformed into a common coordinate system before executing the splitting algorithm. Afterwards both have to be transformed back into their original coordination system.

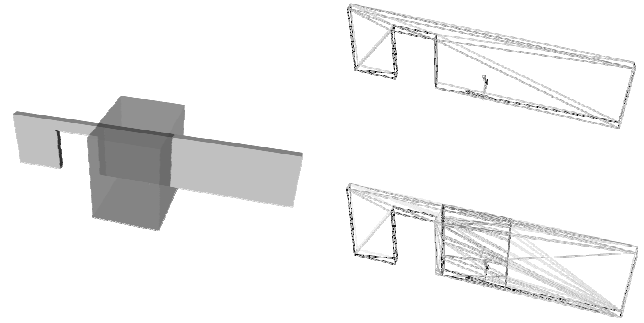


Figure 8: Object refinement of a wall

## 2.4 Remaining drawbacks

The current implementation is based on the algorithm from (Laidlaw et al. 1986). But (Hubbard 1990) already published a much more efficient version by avoiding the local approach of surface intersecting and a more efficient classification phase. The code provided by (Castanheira 2003) which was used as basis for the test implementation doesn't consider these improvements.

Furthermore, in the current implementation the splitting functionality presuppose closed polygons (zones) as cutting objects. But in practice axial systems consisting of a set of open polygons are also used to specify cutting locations. The algorithm should be enhanced to support these cutting objects by automatically generating three dimensional zones defining the area between two specified axis.

In contrast to geometry, the splitting of object attributes could lead to an ambiguous problem e.g. for an attribute like the percentage of tiles on a wall surface. In that case user interaction is needed but a defined product ontology could help to classify attributes to be inherited from the parent, calculated from the geometry or ambiguous.

Another problem could occur when splitting compound objects which already consist of strictly separated surface meshes as e.g. windows or furniture. In this case, because of the object separation, the splitting operation produces many small parts which may be unwanted by the end-user.

Finally, to further reduce the additional effort for the end-user the object splitting and selection functionality could be encapsulated in a computer interpretable command or query language which allows to form expressions similar to the description tags used today for activities in the schedule (e.g. slab in zone A). For high rise buildings with nearly identical structures in each storey also a template based generation of those query expressions would further ease the creation of schedules and 4D simulations.

### 3 MAPPING OF REFINED OBJECT GRANULARITIES TO IFC

Parts of subdivided CAD objects are integrated into the IFC model as components of their parent through the decomposition relationship. The `IfcRelDecomposes` and any of its subtypes are hierarchal and acyclic. Any Object can be included in a single aggregation or nesting relationship. However, transitive decompositions (aggregate or nesting) are allowed. The main aim is to be able to navigate through the (whole/part) hierarchy of the object. In the meantime, a set of CAD objects that is allocated to an activity in the construction schedule is grouped in the IFC model through a grouping relationship.

#### 3.1 Decomposition of Objects, Layering Vs Splitting

Decomposition of CAD objects in the IFC model has two dimensions. First is the Layering dimension, where different layers can represent different work tasks (e.g. masonry work, plastering and painting of a wall). On the other hand, there is another dimension which is the portion or amount of work in each individual work task regardless the underlying material layers.

The first dimension is mapped into the IFC model through the `IfcMaterialLayerSet` which defines the relative positioning of individual layers relative to an axis (IAI, 2006).



Figure 9: The multi-layering system of materials in IFC

Figure 9 shows how different material layers are positioned in reference to each other. Meanwhile, the `IfcMaterialDefinitionRepresentation` entity allows for multiple presentations for the same material for different geometric representation contexts that suit the 4D simulation requirements.

#### 3.2 Grouping and Splitting

On the second dimension there is a need to be able to subdivide objects into smaller components or group them to form larger units according to the corresponding work task in the construction schedule

on the basis of volumes or clipping zones regardless of the material layers.

##### 3.2.1 Grouping of Objects

Grouping of several objects to be allocated to one work task is done by using the IFC relationship (`IfcRelAggregates`). It is worth mentioning that this relation is also capable of grouping objects that are not of the same type. The only prerequisite is the logical dependency. If the logical dependencies between the objects fail to exist, then the more general grouping concept of the IFC model can be used (`IfcGroup` and `IfcRelAssignsToGroup`).

##### 3.2.2 Splitting algorithm

Splitting of CAD objects in IFC model is much more awkward than grouping. The decomposition relationship that is used in the splitting process is the `IfcRelNests` relationship, as shown in Figure 10. It only allows for nesting objects of the same type. This means that a wall must be decomposed to walls, a beam to beams and so forth as shown in the EXPRESS-G diagram in Figure 10.

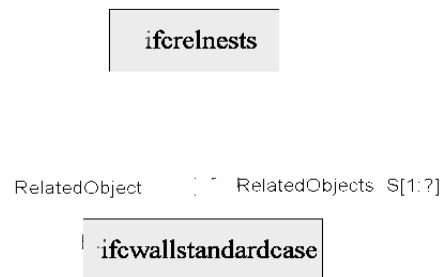


Figure 10: An EXPRESS-G diagram showing the nesting of objects

In the case of mixing objects to suit the objects within a given work task, it is mandatory to use the grouping relation `IfcRelAggregates` as the nesting relationship does not permit mixing different types of objects.

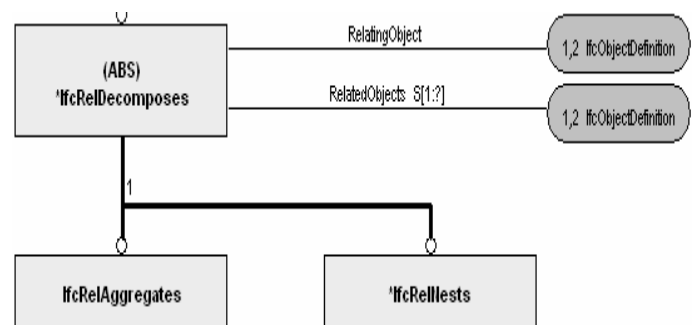


Figure 11: An EXPRESS-G diagram showing the nesting and aggregation relationships

Figure 11 shows the decomposition mechanism in the IFC model. Both the nesting and aggregation relationships are derived from the abstract entity (IfcRelDecomposes).

Figure 12 shows how the nesting relationship of walls is exchanged through the IFC STEP-21 (ISO 10303-P21, 1994) format.

```

DATA;
#114= IFCWALLSTANDARDCASE
('10iC4Sf6zBK9yE8ZwTDQPu',#56,'Wall_A2',
'SecondChild',,$,$,$,$);
#40= IFCPERSON ('personID', 'personFamily-
Name', 'personGivenName', $,$,$,$,$);
#42= IFCORGANIZATION ('Organiza-
tionID', 'OrganizationName', 'Description', $
,$);
#135= IFCRELNESTS
('1gd13QG7L1KhRdK7oR9i15',#56,'Decompositio
nRelation','This Relation Manages the Nest-
ing',#76, (#114,#95));
#95= IFCWALLSTANDARDCASE
('0$Hmk2osb6GeTsZHV3PCcE',#56,'Wall_A1',
'FirstChild',,$,$,$,$);
#56= IFCOWNERHISTORY
($,#53,$,..NOCHANGE..,$,$,$,1178130842);
#76= IFCWALLSTANDARDCASE
('1_nE7A_FvFRuRxElrUeAo6',#56,'Wall_A0',
'ParentWall',,$,$,$,$);
#53= IFCAPPLICATION
(#42,'TestVersion','TestApplication','Test
ID');
ENDSEC;

```

Figure 12: A STEP-21 example showing the decomposition of a wall “A0” to two children walls “A1” and “A2”

It is clear from Figure 13 that the parent IfcWall-StandardCase (A0) is subdivided into two children walls (A1 and A2). To reduce the complexity of splitting objects, it has been decided to include only one level of split children objects. If the 4D simulation and the optimisation of the construction schedule require a deeper degree of granularity, then the new granularity replaces the old one and acts as the degree of granularity that can serve the production of more simulation alternatives (Figure 13).

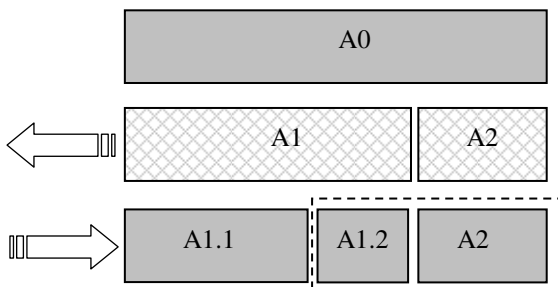


Figure 13: Replacing the domain specific object granularity in the IFC STEP file

Figure 13 shows how the IFC model is updated to include a deeper degree of granularity of the split CAD object. Object A1 is substituted by both A1.1

and A1.2. In the meantime, object A2 remains as it is. If a single work task addresses A2 and A1.2, then a new grouping is formed to include both of them. This grouping is allocated to the work task (IfcTask) through the IFC relationship IfcRelAssignsToProcess.

In this manner, the project scheduler can simulate different combinations and routes of construction for optimization reasons.

## 4 CONCLUSIONS AND FURTHER RESEARCH

To reduce the inter process communication between different actors during the creation of 4D simulations a general object splitting and related data management concept was developed to enable project schedulers to address model portions independent from the original CAD object granularity. User defined, three dimensional zones are used to specify these model parts which should be linked to a task in the construction schedule.

Based on such a request, the boundary representation (geometry) of the affected CAD elements is split automatically into a lower object granularity. The new objects are added to the model as parts of the parent object and can be used to calculate lower granular quantities which are needed for scheduling. The concept thereby allows supporting several alternative construction schedules through a greatest common object granularities approach which is managed within IFC and exchanged through IFC-STEP ISO 10303-P21 files. Because of the parent child relationship the new domain specific objects can be integrated in update cycles.

The main outcome of this research work is expected to be an interactive BIM editor dedicated to the project scheduler which in particular allows to easily slice the product model according to the needs of the construction schedule. The needed tools for parsing, interpreting and navigating the IFC model in an interactive 4D viewer with object splitting functionality and the ability to instantiate the IFC model with the new elements in relation to the parent elements are developed by the authors.

It is obvious that such a tool speeds up the creation time for schedules and 4D simulations. By giving the ability to investigate several different schedules for the same CAD model it also enables project optimization and thus will further promote the use of BIMs also during scheduling.

## 5 ACKNOWLEDGEMENT

This research relates to InPro, an integrated project within the 6th EU Framework Program for Research and Development ([www.inpro-project.eu](http://www.inpro-project.eu)).

## REFERENCES

- Aalami F.B., Fischer M.A. & Kunz J.C. 1998. AEC 4D Production Model: Definition and Automated Generation. CIFE Working Paper #52 Stanford University
- Castanheira D.B.S. 2003. Geometria construtiva de sólidos combinada à representação b-rep. The Boolean Set Operation Project  
<http://www.geocities.com/danbalby/>
- Floyd R. & Warshall S. Algorithm for transitive hull calculation.  
[http://de.wikipedia.org/wiki/Algorithmus\\_von\\_Floyd\\_und\\_Warshall](http://de.wikipedia.org/wiki/Algorithmus_von_Floyd_und_Warshall)
- Haymaker J. & Fischer M. 2001. Challenges and Benefits of 4D Modeling on the Walt Disney Concert Hall Project. CIFE Working Paper #64 Stanford University
- Heesom D., Mahdjoubi L. 2004. Trends of 4D CAD applications for construction planning. Construction Management and Economics, 22:171-182.
- Hubbard P.M. 1990. Constructive Solid Geometry for Triangulated Polyhedra. Technical Report CS-90-07. Department of Computer Science Brown University
- IAI 2006. Industry Foundation Classes, IFC2X Edition 3. International Alliance for Interoperability.  
[http://www.iai-international.org/Model/R2x3\\_final/index.htm](http://www.iai-international.org/Model/R2x3_final/index.htm)
- ISO 10303-21 STEP, Industrial Automation Systems and Integration — Product Data Representation and Exchange — Part 21: Implementation Methods: Clear Text Encoding of the Exchange Structure, ISO 10303-21:1994 (E)
- Laidlaw D.H., Trumbore W.B., Hughes J.F. 1986. Constructive Solid Geometry for Polyhedral Objects. ACM SIGGRAPH Computer Graphics. 20(4):161-170.
- Reinhardt J., Garrett J. H., Akinci Jr & B. 2004. Si-DaCoS: Product and Process Models on Construction Sites. Conference proceedings ICCCBE Weimar.
- Tulke J. & Hanff J. 2007. 4D construction sequence planning – new process and data model. CIB W78, 24<sup>th</sup> Conference, Maribor.