A METHODOLOGY TO AUTOMATICALLY GENERATE GEOMETRY INPUTS FOR ENERGY PERFORMANCE SIMULATION FROM IFC BIM MODELS

G.I. Giannakis[1], G.N. Lilis[1], M.A. Garcia[2], G.D. Kontes[1], C. Valmaseda[2] and D.V. Rovas[1,3]
[1]Department of Production Engineering and Management, Technical University of Crete, Chania, Greece
[2]Department of Energy, Fundacion CARTIF, Valladolid, Spain;
[3]Group of Energy Systems, Fraunhofer, Institute for Building Physics, Germany

ABSTRACT

Building Energy Performance (BEP) simulation model require significant effort for set up, limiting the potential utilization of modelling both in the design and operational phases. A methodology for semi-automated BEP simulation model creation could make this process much more expedient and as such lower to threshold for the use of such models. Building Information Models (BIM) are an information-rich repository that could be used to streamline and expedite the collection of such information. Concerning the building geometry, the Industry Foundation Classes (IFC) can provide static building information that include geometric configuration and material properties, but in a form that might not be directly usable for the generation of thermal simulation models due to the absence of 2^{nd}-level boundary information. In this paper, a three-step methodology for (semi-) automated generation of thermal simulation models is presented, including: a query on the building data model requesting geometry-related information; a processing of the acquired data by a 2^{nd}-level boundary identification algorithm, the Common Boundary Intersection Projection (CBIP) algorithm; and a transformation stage that converts the geometry information of IFC, along with the data obtained from the CBIP algorithm, to an EnergyPlus and/or TRNSYS input file, with the latter being the main topic of this paper.

INTRODUCTION

The accuracy of thermal simulation depends significantly on the correct representation of the building geometry, commonly being initially generated by a Building Information Model (BIM) authoring tool according to an architectural perspective that must be altered for energy performance simulation tasks (Bazjanac, 2010). Geometrical data extracted from BIM have to be transformed and combined with material properties to be entered as inputs to energy simulation engines, a process which is time consuming and error prone. Relevant BIM data schemas include the Industry Foundation Classes (IFC) (ISO16739, 2013) and the green-building XML schema (gbXML). Many commercial authoring tools (e.g. Revit™, Vasari™ and ArchiCad™) support export in one of these two BIM schemas. One of the problems is that this export is often not perfect: unlike what would be expected the resulting exports are of poor quality and therefore not directly usable.

The topic of automated data translation between BIMs and thermal simulation tools has received considerable interest as of late. The IDF Generator (Bazjanac, 2009) works in conjunction with the Geometry Simplification Tool (GST) and transforms IFC-format building geometry into EnergyPlus input-data file (IDF); GST simplifies the original building geometry defined in IFC-format and converts it into gbXML-format, while the IDF Generator converts the gbXML-format file into EnergyPlus input-data file. The resulting IDF file contains all information related to building geometry and constructions needed to run an EnergyPlus simulation. For complex geometries, the IDF Generator requires revisiting the generated files to include corrections to windows in curtain walls, missing floors and ceilings (O'Donnell et al., 2013). The RIUSKA (Jokela et al., 1997) uses the DOE-2.1 thermal simulation engine and imports the IFC-defined building geometry, utilizing the BSPro server middleware (Karola et al., 2002). Limitations of its IFC import exist, since RIUSKA ignores slabs in the IFC file and generates them internally, according to the size of the space defined by the bounding walls. Moreover, high quality of import results are achieved only when RIUSKA is used in conjunction with SMOG, while problems occur if other CAD tools are used to develop and export the IFC file.

The Green Building Studio (GBS)™ web service requires a gbXML-based description of the building which is converted into a DOE-2.2 or an EnergyPlus input file. Studies indicating problems that occur during the conversion process (Maile et al., 2007) are abundant: most notable errors include the incorrect shading surface definitions and omission of some walls. Google SketchUp along with its Openstudio and IFC2SKP plugins, is able to upload any gbXML or IFC well-formatted geometry and convert it into the EnergyPlus or TRNSYS17-format file. However, due to imperfections of the import tool, information related to floors and ceilings is neglected. Virtual Environment (VE)™, developed by Integrated Environmental Solutions (IES), is an integrated system that uses the Apache simulation engine. IES VE supports import of gbXML and IFC file formats. Nevertheless, the results depend on the correctness of 2^{nd}-level space boundary definitions contained in the IFC file, which currently are not exported properly by any BIM tool.

Among the two most popular BIM schemes, gbXML and IFC, IFC appears to be a suitable choice as its rich content, enables interoperability among different software environments and can be easily updated following building's life cycle (Hitchcock and Wong, 2011). Concerning the building geometry, IFC can provide static building information that include geometric configuration and material properties. In many cases

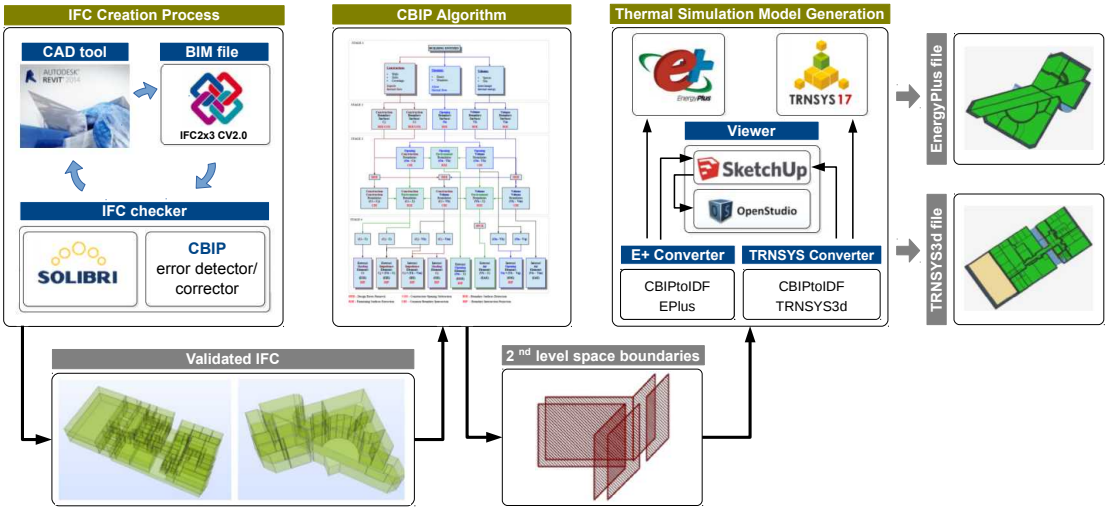

Figure 1: Geometry Transformation of an IFC file to a Building Energy Performance simulation model – the overall process

however, the necessary for energy simulations 2^{nd}-level space boundary data (O'Donnell et al., 2013), contained in IFC, are missing or are incorrect due to design errors or exporting software imperfections. Hence, a consistent way of transforming the building geometry information, contained in the IFC, to the 2^{nd}-level space boundary information is required.

In view of this, a 2^{nd}-level space boundary generation algorithm has recently been proposed (Rose and Bazjanac, 2013), based on graph theory to convert a three-dimensional architectural building model, without defined thermal space boundaries (2^{nd}-level space boundaries), to one suitable for import into a whole-building energy performance model. In this work, a different approach is developed to address the 2^{nd}-level space boundary generation requirement, namely the Common Boundary Intersection Projection (CBIP) algorithm (Lilis et al., 2014). Once such information has been generated, transformation rules are employed to convert data obtained from the CBIP algorithm to an EnergyPlus and/or TRNSYS input file.

The proposed multi-step methodology to extract IFC inputs and transform to inputs appropriate for a Building Energy Performance simulation model consists of different components, schematically shown in Figure 1 and explained in more detail in the following sections. Exemplary application of the proposed methodology to generate a simulation-ready TRNSYS input file is provided.

IFC DATA QUALITY

The IFC file schema manages to represent building's geometry with a very compact and precise way, accommodating multiple geometry representations in a single file. Although IFC has the necessary classes to support the description of building geometry, these classes sometimes contain incorrect data, mainly induced due to two reasons: either the designer has erroneously defined certain building entities, or the IFC-exporter software has flaws and exports incorrectly the geometrical data in the IFC file.

For the 2^{nd} -level boundary generation process, the input geometry should be of good quality; such errors can slow or make that process to fail. In O'Donnell et al. (2013), errors that affect the creation of properly defined 2^{nd}-level space boundaries are presented. Consequently, as a post-processing step, after the IFC file export, such errors should be detected and corrected either automatically or manually by communicating them back to the designer.

Concerning their detection, existing software packages such as the Solibri Model Checker™, perform detection (checking) of geometric inconsistencies, which are communicated back to the AEC software and corrected manually. In cases of exporter induced errors, it is hard to rectify them unless we work directly with the IFC file — something that requires quite often expert knowledge. Hence, automatic error detection and correction mechanisms, should be employed, however within limits; unless there are unambiguous conditions ("small" errors) which can be automatically corrected, these errors have to be manually corrected by the designer. Towards automatic detection and correction of such inaccuracies, detection and correction algorithms have been developed and could be performed, though their description is not the topic of this work.

IFC DATA QUERY

Although IFC files contain information referring to multiple building geometry entities, only some of them are required for building thermal simulations. These building geometry entities can be classified into three categories depending on their role in thermal

building simulations: Building Elements, Openings, and Volumes. IFC classes, which refer to building elements, belong to the abstract *IfcBuildingElement*.

Openings are building entities described by the *IfcOpeningElement* class. The *IfcOpeningElement* class contains information associated with building openings, such as window's and door's openings.

Building volumes are categorized as follows: (1) *Building spaces* refer to the air volumes of rooms or room partitions (separated by air boundaries) — Building spaces are defined by the *IfcSpace* class; (2) *Building site* refers to the surrounding ground volume, encompassing the building under consideration — Building site is defined by the *IfcSite* class. IFC classes related to Volume entities, belong to the abstract *IfcBuildingSpatialStructureElement*.

Regarding their representation, all these entities are considered products which are related to the abstract *IfcProduct* class. The representation data are contained in the *IfcGeometricRepresentationItem* class, related to the *IfcProductDefinitionShape* subclass of the *IfcProduct* class.

There are four main solid geometrical representations and respective sub-classes of the *IfcGeometricRepresentationItem* class, that can be handled by the CBIP algorithm: (1) *Face based surface model representation*, described by *IfcFacedBasedModel* class; (2) *Solid model representation*, described by *IfcSolidModel* class; (3) *Half Space Solid representation*, described by *IfcHalfSpaceSolid* class; and (4) *Boolean result representation*, described by *IfcBooleanResult* class.

CBIP ALGORITHM

CBIP algorithm is a four-step methodology consisting of: (1) the Identification stage that corresponds to the IFC data query step; (2) the Boundary Surface Extraction stage; (3) the Common Boundary Intersection stage; and (4) the Boundary Intersection Projection stage.

Boundary Surface Extraction

CBIP operates on the boundary surfaces of the building entities' polyhedrons and extracts the space boundary surfaces, i.e. the common surfaces among spaces and other construction polyhedrons. An essential input requirement of CBIP algorithm is that all of the involved building entities must have an outward oriented B-rep (Jackson, 1995), condition. B-rep theory is adopted in order to describe each polyhedron by its corresponding boundary polygons.

Representations that do not contain the desired B-reps for CBIP, require geometric calculations, which are preformed in the stage of the algorithm called Boundary Surface Extraction (BSE) process.

Common Boundary Intersection

The Common Boundary Intersection (CBI) process determines the Common Boundaries (CB) surfaces shared by two polyhedrons of respective building entities. CBI is applied on the B-reps $\partial\mathbf{A}$ and $\partial\mathbf{B}$ of polyhedrons $\mathbf{A}$ and $\mathbf{B}$, and outputs the set of CB surfaces $\mathcal{CB}_{AB}$, shared by the two polyhedrons using Boundary Space Partitioning tree (BSP-tree) polyhedral representations (Thibault and Naylor, 1987).

Boundary Intersection Projection

The CB surfaces are forwarded to the final stage of the algorithm, the *Boundary Intersection Projection* process, to generate the required geometry elements of a Building Energy Performance (BEP) model. These elements, are essentially *surface pairs*, which include thermal material properties and other relative information. The BIP process can be described by two geometrical operations: (1) the *projection* of one of the common boundaries on the plane of another; and (2) the *intersection* of the projection with the other element.

The projections of BIP process are applied to four types of CB, derived from the CBI stage: C-V (Construction - Volume), C-E (Construction - Environment), O-V (Opening - Volume) and O-E (Opening - Environment). In case a C-V, C-E, O-V or O-E common boundary is not projected into a C-V, C-E, O-V or O-E common boundary, it remains as a CB (denoted as Remain Surface (RS)) and is associated with a specific building entity (wall, slab, etc.).

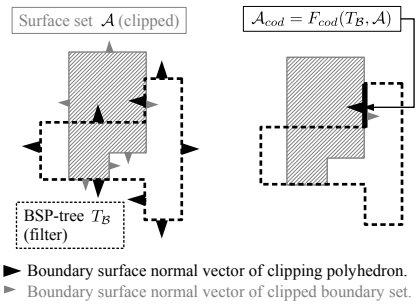

Figure 2: Geometric illustration of the clipping function

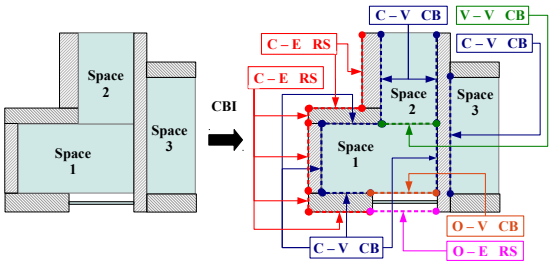

Figure 3: Geometrical illustration of CBI process

Concerning the operations required by the CBI and the BIP stages, a clipping function (F_{cod}) is defined. From a general point of view, this function is applied on a surface set $\mathcal{A}$ using the BSP-tree T_B of another surface set $\mathcal{B}$ and returns surfaces or parts of surfaces of $\mathcal{A}$, which are coplanar with the surfaces of $\mathcal{B}$ and have opposite normal vectors – see Figure 2).

The results of CBI and BIP operations for a naive example of three spaces' floor plan, are depicted in Figures 3 and 4, respectively.

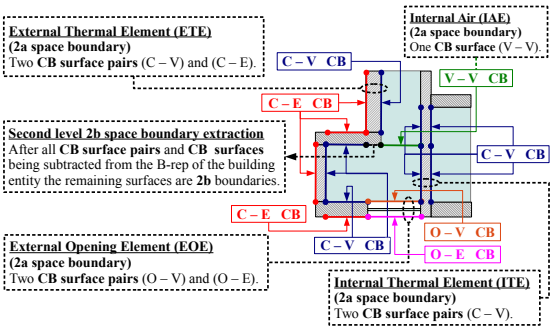

Figure 4: Geometrical illustration of BIP process

Second-Level Space Boundary Generation

After completion of the BIP process on all building elements of interest, CBIP surface pairs are extracted. These CBIP surface are related to the 2nd-level space boundaries of type 2a (as defined in (Bazjanac, 2010)). Apart from the 2nd-level boundaries of type 2a, those of type 2b (see Figure 4), 2c and 2d (Bazjanac, 2010), are also extracted. These special cases of 2nd-level space boundaries can be ignored or be entered as adiabatic surfaces in a thermal simulation model.

The surface pairs obtained by CBIP can populate the *IfcRelSpaceBoundary2ndLevel* classes, as displayed in Figure 5.

Each boundary surface's geometry (surface polygon) is pointed by the *ConnectionGeometry* item to a respective geometric representation. The boundary's location, with respect to other building entities, is indicated by the *InternalOrExternal* item, which can potentially receive the following values: INTERNAL; EXTERNAL; EXTERNAL_EARTH.

If the boundary surface refers to a building construction (wall, slab , etc.), the item *PhysicalOrVirtual* receives the *PHYSICAL* value (boundaries #101, #102, #103, #104 in Figure 5); otherwise, it refers to a surface separating building spaces, the PhysicalOrVirtual receives the *VIRTUAL* value (boundaries #105 and #106 in Figure 5).

CBIP's surface pairs are defined by the *CorrespondingBoundary* attribute. For example, the external wall of Figure 5 contains a thermal element, defined by two boundary surfaces (#101, #102), which forms a pair indicated by the CorrespondingBoundary attribute (the Corresponding boundary of #101 is #102 and vice versa).

If a boundary surface contains openings (doors, windows, etc.), these openings are indicated by the *InnerBoundary* attribute of the boundary surface, which contains the boundary surface pairs of these openings. In Figure 5 for example, the boundary surface #102 contains an opening indicated by the InnerBoundary #103. In the same manner, for the inner space boundaries, the space boundaries they belong to are indicated by the attribute *ParentBoundary* (as indicated in Figure 5, inner boundaries #104 and #103 have boundaries #102 and #101 as parent boundaries, respectively).

If the boundary surface refers to an internal boundary attached to a specific building space, this space is indicated by the *RelatingBuildingSpace* attribute which points to the respective IfcSpace class. For instance, in figure 5, boundaries #102, #103 and #105 indicate space #1 as their internal space in figure 5.

Finally, the building element in which the boundary surface corresponds to, is indicated by the *RelatedBuildingElement*attribute. If the boundary surface is a virtual boundary, such an attribute does not exist. In Figure 5, the boundaries #101 #102 refer to an external wall and the boundaries #103 and #104 refer to an external window.

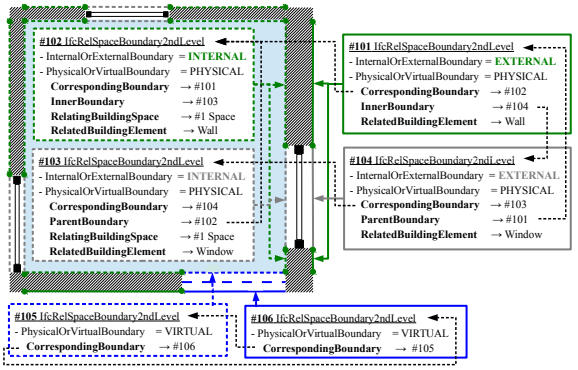

Figure 5: Example of IFC4 classes referring to 2nd-level space boundaries

Data Requirement Addressed by CBIP

According to (Maile et al., 2013) and referring strictly to building's envelope simulation, we infer that the following data are required: (1) opaque surfaces' thermal properties; (2) reflectance properties of surrounding surfaces; (3) transparent surfaces' glazing and thermal properties; (4) surface type (internal or external); (5) surface area; (6) convection coefficients (inside and outside); (7) 3D position of surfaces (to calculate correctly the solar radiation effects); (8) normal vector of each surface (to determine the inner most and the outer most layer of surface' construction); (9) relationship between surfaces and spaces (to define the flow paths of each zone); (10) relationship between materials and the surfaces (to define the construction of a building element); (11) relationship between two opposite surfaces (for internal heat transfer); and (12) relationship between parent and child surfaces (for openings hosted to a wall).

Contrasting Figure 5 and the enumerated list of the data requirements discussed above, it is obvious that, while most data requirements can be fulfilled by data available through the CBIP algorithm output, missing data for the material thermal properties is a major drawback.

BEP MODEL GENERATION

For a BEP model generation, a transformation process which converts CBIP output, to a thermal simulation engine's input file is required. Rules embedded in this transformation process are presented in this section.

EnergyPlus IDD–classes

In EnergyPlus (DOE, 2014, 2013), model input data are defined by two ASCII (text) files: the Input Data Dictionary (IDD) and the Input Data File (IDF). All possible EnergyPlus classes, and a specification of the properties each class has, are defined in the IDD file. Each version of Energyplus has a different IDD file. IDF file consists of all the necessary IDD–classes' objects to properly define a thermal simulation model of a certain building. Each thermal simulation model has a different IDF file.

In order to utilize all the information encapsulated in these files, a Matlab script has been developed that identifies the version of the IDF file, parses the appropriate IDD file, and creates a library (MatlabIDD$_{xx}$, where xx is the EnergyPlus version) of Matlab classes, corresponding to EnergyPlus classes.

For a building's envelope thermal simulation, objects of the following IDD classes must be determined: (1) Version – to define the proper version that the IDF is created for; (2) SimulationControl – to specify what kind of calculations will be performed; (3) Building – to describe parameters that are used during the simulation of the building; (4)/(5) SurfaceConvectionAlgorithm: Inside/Outside – to determine the convection algorithm used for surface convection calculations at the inside and outside face of all the heat transfer surfaces in the model; (6) HeatBalanceAlgorithm – to determine the conduction transfer model; (7) Timestep – to specify the simulation timestep; (8) RunPeriod – to describe the elements necessary to create a weather file simulation; (9) Material – to define the thermal properties of opaque materials; (10) WindowMaterial:SimpleGlazingSystem – to define thermal and glazing properties of a glazing material only by the U-factor and the SHGC (Solar Heat Gain Coefficient); (11) Construction – to specify the material layer bedding of building elements; (12) GlobalGeometryRules – a description of geometric parameters before the surface objects are explained in detail; (13) Zone – to set-up the thermal zones of the building; (14) BuildingSurface:Detailed – to describe each of the opaque surfaces; and (15) FenestrationSurface:Detailed – to describe each of the transparent surfaces.

CBIPtoIDF – EnergyPlus

Simulation input parameters, described by objects of classes: 1 trough 8 and 12, can be set to default values or be entered by users. For all the other classes a transformation of the IFC and CBIP output data must be performed. This transformation process follows seven major rules which are described in this section. The energyPlus classes which are transformed to MatlabIDD$_{8.1}$ classes referring to zones, constructions, surfaces and materials are presented in tables 1, 2, 3 and 4.

Rule 1: Although IFC has the necessary data structures to support information referring to thermal properties of building construction materials, current IFC-export tools export partially such information. As a result, Material and WindowMaterial:SimpleGlazingSystem classes properties are set manually by the user and only material names and layer thicknesses are obtained from the IFC file.

Rule 2: In IFC each construction is defined as an IfcMaterialLayerSet and is mapped to a Construction class object (table 2), for which the layer bedding is a list of IfcMaterialLayers.

Rule 3: Each building space, defined as an IfcSpace class object, is mapped to a new Zone MatlabIDD$_{8.1}$–class object (table 1), with Name value set equal to the IfcSpace Global ID (GID). The space boundary surfaces, not including the ones with relating space external earth, are used to populate BuildingSurface:Detailed and the FenestrationSurface:Detailed class objects 3.

Table 1: Zone MatlabIDD$_{8.1}$ class properties

Zone class properties
1. *Name* – a unique name associated with each building thermal zone.
2. *DirectionofRelativeNorth* – Zone North Axis (relative to the Building North Axis).
3. *XOrigin, YOrigin and ZOrigin* – X, Y and Z coordinates of a zone origin.
4. *Type* – not used, always left empty.
5. *Multiplier* – always set to 1;
6. *CeilingHeight* – zone's ceiling height.
7. *Volume* – zone's volume.
8. *FloorArea* – zone's floor area.
9. *ZoneInsideConvectionAlgorithm / ZoneOutsideConvectionAlgorithm* – name of the algorithm used to calculate the inside / outside zone surface convection coefficients.
10. *PartofTotalFloorArea* – If not specified, 8-10 are auto-calculated.

Table 2: Construction MatlabIDD$_{8.1}$ class properties

Construction class properties
1. *Name* – a unique name associated with each building construction.
2. *OutsideLayer and Layer2 - Layer10* – each layer of the construction is a material name listed in order from outside to inside.

Rule 4: For every surface i, which does not have a parent boundary (i is not an opening), a related BuildingSurface:Detailed class object is generated, with Name

and ZoneName values, the space boundary GID and the space boundary relating space GID, respectively. The normal vector n_i of i is calculated based on the space boundary coordinates, obtained from CBIP's output.

Rule 5: For every class related to surface i, adhering to rule 4, the following apply: If $n_i(3) = 0$, the SurfaceType value is set equal to Wall and the ViewFactortoGround value is set equal to 0.5; else if $n_i(3) = 1$, the SurfaceType value is set equal to Floor and the ViewFactortoGround is set equal to 1; else the SurfaceType value is set equal to Roof/Ceiling and the ViewFactortoGround value is set equal to 0. Moreover, if i has 2^{nd}-level space boundary of type 2a and its corresponding space boundary does not have a relating space, the OutsideBoundaryCondition, SunExposure, WindExposure property values of the related surface class, are set equal to Outdoors, SunExposed and WindExposed, respectively. In case the corresponding space boundary of i has as relating space the earth, the OutsideBoundaryCondition, SunExposure, WindExposure property values of the related class are set equal to Ground, NoSun and NoWind, respectively. If the corresponding space boundary of i has a relating space different from the earth, the OutsideBoundaryCondition value, is set equal to Surface, instead of Ground, while the OutsideBoundaryConditionObject value is set equal to the corresponding space boundary GID. In any other case, (i has 2^{nd}-level space boundary of type 2b) the corresponding BuildingSurface:Detailed class can be neglected or be defined with an Adiabatic OutsideBoundaryCondition. The generation of the BuildSurface:Detailed class object is completed when the NumberofVertices and coordinates of each vertex are populated using the space boundary coordinates obtained from CBIP.

Rule 6: If surface i, has a parent boundary (i iss an opening), a respective FenestrationSurface:Detailed class object is generated.

Rule 7: For every class related to surface i, adhering to rule 6, the following apply: If the construction of i consists of opaque Matetials, then the SurfaceType value of the related class is set equal to Door; else the SurfaceType value is set equal to Window. If the BuildingSurfaceName value is set equal to the parent boundary surface GID, the ViewFactorToGround value is set equal to autocalculate. If the corresponding space boundary of i has a relating space, the OutsideBoundaryConditionObject value is set equal to the corresponding boundary GID. The FenestrationSurface:Detailed NumberofVertices and coordinates of each vertex are populated using the space boundary coordinates obtained from CBIP.

CBIPtoIDF – TRNSYS3d

A plug-in called TRNSYS3d in Goggle SketchUp is available for TRNSYS (TRANSSOLAR, 2010), which facilitates the geometric data input to the building model. The TRNSYS3d plugin exports an IDF file consisting of Zone, BuildingSurface:Detailed and FenestrationSurface:Detailed IDD-class objects.

Table 3: Surface MatlabIDD$_{8.1}$ class properties

BuildingSurface:Detailed class properties
1. *Name* – surface name. 2. *SurfaceType* – Wall, Floor, Ceiling, Roof or Surface; 3. *ConstructionName* – surface's construction name. 4. *ZoneName* – surface's zone name. 5. *OutsideBoundaryCondition* – surface, adiabatic, outdoors or ground. 6. *OutsideBoundaryConditionObject* – if the Outside Boundary Condition is Surface, then this property's value is the surface name whose inside face temperature will be forced on the outside face of the base surface. 7. *SunExposure* – SunExposed or NoSun. 8. *WindExposure* – WindExposed or NoWind. 9. *ViewFactortoGround* – the fraction of the ground plane (assumed horizontal) that is visible from a heat transfer surface. 10. *NumberofVertices* – the number of vertices describing a surface and Vertex–v–X, –Y, –Zcoordinate – the X, Y, Z coordinate of vertex v, respectively.
FenestrationSurface:Detailed class properties
1. *Name* – fenestration name. 2. *SurfaceType* – Window or Door. 3. *ConstructionName* – surface's construction name. 4. *BuildingSurfaceName* – fenestration's surface name. 5. *OutsideBoundaryConditionObject* – if fenestration is interior, then this property's value must be the fenestration surface name whose inside face temperature will be forced on the outside face of the base surface. 6. *ViewFactortoGround* – the fraction of the ground plane that is visible from a heat transfer surface. 7. *ShadingControlName* – the name of the window shading control for this surface. 8. *FrameandDividerName* – window frame divider's name. 9. *Multiplier* – the number of identical items on the base surface (always set to 1). 10. *NumberofVertices* – the number of vertices describing the fenestration surface and Vertex–v–X, –Y, –Zcoordinate – the X, Y, Z coordinate of vertex v, respectively.

Consequently, the transformation rules described in the CBIPtoIDF – EnergyPlus section related to

Zone, BuildingSurface:Detailed and FenestrationSurface:Detailed IDD-classes can be adopted here (rules 3-7) with the following alteration in rule 5:

Table 4: Material MatlabIDD$_{8.1}$ class properties

Material class properties
1. *Name* – a unique name associated with each opaque material. 2. *Roughness* – one of the following choices can be used: VeryRough, Rough, MediumRough, MediumSmooth, Smooth, and VerySmooth. 3. *Thickness* – the thickness of the material. 4. *Conductivity* – the thermal conductivity of the material. 5. *Density* – the density of the material. 6. *SpecificHeat* – the specific heat of the material. 7. *ThermalAbsorptance* – the fraction of incident long wavelength radiation that is absorbed by the material. 8. *SolarAbsorptance* – the fraction of incident solar radiation that is absorbed by the material. 9. *VisibleAbsorptance* – the fraction of incident visible wavelength radiation that is absorbed by the material.
WindowMaterial:SimpleGlazingSystem class properties
1. *Name* – a unique name associated with each window material. 2. *UFactor* – the value for window system U-Factor. 3. *SolarHeatGainCoefficient* – the value for window system SHGC. 4. *VisibleTransmittance* – the value for window visible transmittance; optional.

Rule 5: For every class related to surface i, adhering to rule 4, the following apply: If i has 2^{nd}-level space boundary type 2a and its corresponding space boundary does not have a relating space, then the OutsideBoundaryCondition, OutsideBoundaryConditionObject, SunExposure, WindExposure values of the respective class are set equal to Outdoors, EXTERNAL, SunExposed and WindExposed, respectively. In case the corresponding space boundary of i has earth as relating space, then the OutsideBoundaryCondition, OutsideBoundaryConditionObject, SunExposure, WindExposure values of the related class are set equal to Ground, BOUNDARY = INPUT 1*TGROUND, NoSun and NoWind, respectively. However, if the corresponding space boundary of i has a relating space other than the earth, then the OutsideBoundaryCondition value of the related class is set equal to Zone, instead of Ground, while the OutsideBoundaryConditionObject vlaue is ADJACENT and set equal to the relating space GID of the corresponding space boundary.

ILLUSTRATIVE EXAMPLE

The proposed methodology has been implemented on the geometrical data of real building, which are queried from the respective IFC file exported by the Autodesk Revit IFCexporter.

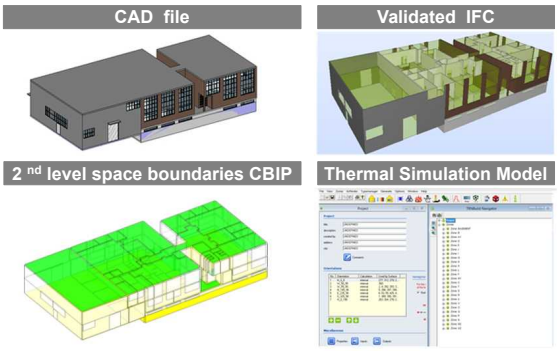

Figure 6: BIM to BEP simulation model: Overall process applied to an office building

The geometrical data of the surface pairs, results of the CBIP algorithm, were collected and transformed in IDD-classes objects in order to create the building model for EnergyPlus and TRNSYS, applying the CBIPtoIDF – EnergyPlus and CBIPtoIDF – TRNSYS3d transformation processes, respectively. Since, almost identical transformation rules are adopted to develop the BEP simulation geometry in both processes, the resulted geometry is the same (see Figure 7), differing in the boundary condition properties' definition.

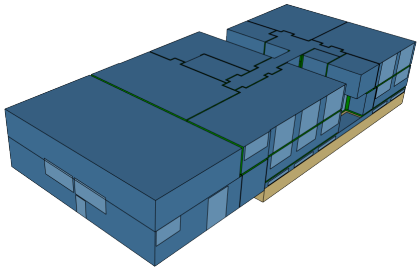

Figure 7: CARTIF building – BEP simulation model geometry

As Figures 6 and 7 demonstrate, CBIP correctly partitions the building walls and slabs according to the topology of the building spaces. Moreover, since almost identical transformation rules are adopted to develop the BEP simulation geometry in both processes, the resulted geometry is the same, differing in the boundary condition properties' definition. The run time of the overall process on a pc is: including error detection and correction processes 473 seconds and without detection and correction, 448 seconds.

CONCLUSIONS

In the present work, a three-step methodology for (semi-) automated geometry's generation of thermal simulation models was proposed, including: a query

on the building data model requesting geometry-related information of the data scheme by an open-source BIM repository; a processing of the acquired data by a 2^{nd}-level boundary identification process called CBIP algorithm; and a transformation stage that converts the geometry and material properties information of IFC, along with the data obtained from the CBIP algorithm, to an EnergyPlus and/or TRNSYS input file. Obviously the translation process to each simulation is specific to the dictionaries used, it might be conceivable that an intermediate data model (e.g. Simmodel) is populated and then one generator for each specific simulation engine is developed. For this case, most of the transformation rules described in this paper would be applicable requiring only minor modifications.

The proposed methodology was applied on a real building and the results demonstrated the ability in handling non-convex geometries and generating all the possible thermal, opening, shading and virtual elements. The second level space boundaries were identified and their space connectivity information was obtained accurately. Additionally, the proposed process also translates any local shading surface data obtained by CBIP, to related shading group EnergyPlus classes. In conclusion the methodology facilitates significantly the overall process of energy simulation model creation from IFC geometric data. Of importance is the quality of the input IFC files; should geometric errors or other inaccuracies exist it might be hard to describe this process. It is for this reason that model-checking to ensure good quality of the IFC file is an important prerequisite.

ACKNOWLEDGMENTS

This research has been partially funded by the European Commission: FP7-ICT-2011-6, ICT Systems for Energy Efficiency #288409 (BaaS) and Innovative design tools for refurbishing of buildings at district level H2020-EeB-05-2015 #680676 (Opteemal).

REFERENCES

Bazjanac, V. 2009. Implementation of semi-automated energy performance simulation: building geometry. In *26^{th} int. conf. CIB W78*, pages 595–602, Istanbul, Turkey.

Bazjanac, V. 2010. Space boundary requirements for modeling of building geometry for energy and other performance simulation. In *27^{th} int. conf. CIB W78*, Cairo, Egypt.

DOE 2013. EnergyPlus: Guide for Interface Developers. U.S. Department of Energy, California.

DOE 2014. Energyplus: Input-output refernce. U.S. Department of Energy, California.

Hitchcock, R. and Wong, J. 2011. Transforming ifc architectural views bims for energy simulation. In *12th conf. of IBSPA, Sydney, 14-16 November*, pages 1089–1095.

ISO16739 2013. ISO 16739: Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries. *European Committee for Standardization (CEN), Brussels*.

Jackson, D. J. 1995. Boundary representation modelling with local tolerances. In *3^{rd} ACM symposium on Solid modeling and applications*, pages 247–254, Salt Lake City, USA. ACM.

Jokela, M., Keinänen, A., Lahtela, H., Lassila, K., and Granlund, O. 1997. Integrated building simulation tool RIUSKA. In *Building Simulation conf.*, Prague, Cezch Republic.

Karola, A., Lahtela, H., Hänninen, R., Hitchcock, R., Chen, Q., Dajka, S., and Hagström, K. 2002. BSPro com-server interoperability between software tools using industrial foundation classes. *Energy and Buildings*, 34(9):901–907.

Lilis, G., Giannakis, G., Kontes, G., and Rovas, D. 2014. Semi-automatic thermal simulation model generation from IFC data. In *European Conf. on Product and Process Modelling (ECPPM)*, pages 503–510, Vienna, Austria.

Maile, T., Fischer, M., and Bazjanac, V. 2007. Building energy performance simulation tools-a life-cycle and interoperable perspective. *Center for Integrated Facility Engineering (CIFE) Working Paper*, 107:1–49.

Maile, T., O'Donnell, J., Bazjanac, V., and Rose, C. 2013. BIM-geometry modelling guidelines for building energy performance simulation. In *13th conf. of IBPSA*, pages 26–28.

O'Donnell, J., Maile, T., Rose, C., Mrazovi, N., Morrissey, E., Regnier, C., Parrish, K., and V., B. 2013. Transforming BIM to BEM: Generation of building geometry for the NASA ames sustainability base BIM. Technical report, Simulation Research Group, California, USA.

Rose, C. and Bazjanac, V. 2013. An algorithm to generate space boundaries for building energy simulation. In *Engineering with Computers*, pages 1–10. Springer.

Thibault, W. and Naylor, B. 1987. Set operations on polyhedra using binary space partitioning trees. In *14^{th} conf. on Computer graphics and interactive techniques*, volume 21, pages 153–162. ACM.

TRANSSOLAR 2010. *A TRaNsient SYstems Simulation Program, TRNSYS 17.00.0019, 2010*. TRNSYS Coordinator Thermal Energy System Specialists, LLC 22 North Carroll Street suite 370 Madison, WI 53703 U.S.A.