Translating Algorithmic Design from CAD to BIM

Renata Castelo Branco [1], António Leitão [2]

[1,2]*INESC-ID, Instituto Superior Técnico, Universidade de Lisboa, Portugal.*

[1,2]*{renata.castelo.branco|antonio.menezes.leitao}@tecnico.ulisboa.pt*

Abstract. Nowadays, practitioners are embracing the BIM paradigm, as it allows a faster development of models using pre-modeled elements and automates time-consuming tasks such as the production of technical documents. Nevertheless, CAD tools still provide greater freedom in form creation, which is better suited for the early and exploratory stages of the project. To take advantage of both approaches, some practitioners begin designing in CAD environments and, when satisfied with the overall shape, transition to BIM. This paper addresses the in between stage of this transition from an Algorithmic Design standpoint. It explains the main differences between the two paradigms and proposes a modeling methodology, using a portable algorithmic design tool capable of generating the building's design in both CAD and BIM software, which facilitates the translation process of the building's description from one paradigm to the other.

Keywords. CAD; BIM; Algorithmic-based Design.

Introduction

Architectural design has seen many changes over time, and more so over the past few decades, namely regarding the representation methods used in its process (Kalay, 2004). After centuries of sketches and hand-made technical drawings, perspectives, and models, the representation archetype changed with the introduction of Computer-Aided Design (CAD) software. Further along, another shift occurred with the spread of Building Information Modeling (BIM), hailed as one of the most promising developments in Architecture, Engineering and Construction (AEC) industries (Eastman, et al., 2008), replacing the two-dimensional conception of projects into a fully three-dimensional representation workflow.

Parallel to these advances, an entirely different manner of conceiving architecture has been pressing forward behind these applications: Algorithmic Design (AD), a computational approach to architectural design. AD allows the user to create forms through algorithms (Terzidis, 2006), describing the shapes through a series of rules and constraints. Recognizing the advantages presented by this approach, many tools were created to support the development of AD programs, firstly within the CAD paradigm, and, more recently, within the BIM paradigm as well.

Both CAD and BIM paradigms have appeared in response to specific needs that have arisen over time. This means that both paradigms present distinct advantages to the modeling process, and the two have set their own role in the architectural agenda. Most practitioners are embracing the BIM paradigm, as it not only accelerates the modeling process with the use of pre-modeled elements, but also automates time-consuming tasks, such as arranging maps of quantities and costs. Nevertheless, CAD tools, as free-form surface modeling tools, provide greater freedom in form creation (Zboinska, 2015), which may justify a preference in developing the early stages of the project in these applications, such as form and concept experimentation. This paper addresses the in-between-paradigms stage of an architectural design process based on Algorithmic Design.

Algorithmic-based Design

AD allows the modeling of complex geometries that would pose challenges to a normal mouse-based approach. An AD approach entails a parametric modeling philosophy, meaning that the design can be manipulated through parameters. Multiplicity of scalar parameters (Meredith, 2008) offers a degree of freedom to design that cannot be achieved by simply using the software as it is provided to the common user. Hence, AD allows the architect to explore a wider range of possibilities with less effort than a mouse-based approach. Furthermore, it allows the user to go beyond what the tool's manufacturers intended, converting the normal tool-user into a tool-maker (Burry, 2011).

Besides transcending the limitations modeling tools might impose on their users (Terzidis, 2006), algorithmically describing buildings' designs is also proving to facilitate the transition between paradigms. Algorithmic descriptions, as mathematical abstractions of the design intent, possess portability qualities that transcend software particularity. Portable AD tools already available in the market are capable of connecting to both CAD and BIM applications. However, the two paradigms still require architects to model using different operations, and consider different sequences and methods for each archetype. The following sections address some of these differences.

Two different paradigms

Programming for CAD tools is more advantageous in an initial stage of the model, as these tools present a better performance when compared to BIM tools. CAD programs can generate geometry faster, since they do not deal with the semantics inherent to BIM objects. For this reason, an architect can test a wider range of solutions for his design in a shorter time span. Furthermore, not only do they allow for the generation of more complex geometries that some BIM tools cannot process, but they also enable a constraint free modeling workflow, where no sequences or precedents are imposed.

BIM tools require the user to model an accurate virtual model of his design, from which technical drawings, like plans and sections, are automatically generated and updated whenever the model undergoes changes. Unlike a CAD model, that contains only the modeled geometry, BIM embeds the model with data needed to support construction, fabrication, and procurement activities (Eastman, et al., 2008). This means that, besides plans and sections, the programs are also capable of automatically elaborating other technical elements for construction management, such as quantity and cost charts.

When the project enters a phase of greater detail, shifting the scripting task to the BIM paradigm has proven to reduce the time and effort spent on the modeling process. In a BIM environment, the architect can take advantage of all the information available in the families or libraries, saving a lot of time, as he needs not to algorithmically model every single geometric element from scratch.

As each archetype better fits a specific stage of the design process, transitioning from CAD to BIM in the midst of a project is becoming more and more common. To this end, users may try importing the models generated in CAD into BIM tools, but the results are seldom satisfactory. BIM tools recognize the geometry created in CAD, yet they rarely succeed in embedding them with the right semantics. Hence, even if the geometry is successfully imported, its elements may still not be recognized as BIM objects. As a result, architects may end up having to rewrite the entire description of the model from scratch. This paper addresses a transition process with no model imports or exports. Instead, we promote a conversion of the scripted model for CAD applications to one capable of producing equivalent building elements in BIM applications.

There are several examples of projects that faced this need for translation. The Aviva Stadium began using McNeel’s Rhinoceros platform for the architects to quickly explore the stadium’s geometry, which was latter rebuilt within Bentley’s Generative Components (Shepherd, et al., 2011). The Shanghai Center project and Hangzhou Olympics Stadium are two other examples. Both were initially generated in Rhinoceros, using Grasshopper, for parametric design of the mass and the skin of the buildings, and then translated into BIM (Autodesk Revit) for detailed design and construction (Kensek & Noble, 2014).

Lessons learned regarding translation processes

Assuming the architect begins scripting his design in a CAD environment, so as to take advantage of a constraint-free workflow, and further along the process transitions to the BIM paradigm for a more detailed phase of the model, we propose a methodology for the transition stage between the two. This methodology focuses on Algorithmic Design and presents a set of rules for the conversion of a scripted design thought to be generated in CAD, into a rationalized concept that can be modeled with BIM objects. The following paragraphs expose the main challenges posed by the translation process.

While modeling for CAD, the architect can explore his design concerning himself with geometry only. Hence, taking advantage of the modeling operations that create geometric shapes, like circles and boxes, and that apply geometric transformations, including translations, lofts, extrusions and sweeps, he can explore several variations to the form of the various elements that compose the building.

When transitioning to BIM, the user must consider other aspects beyond geometry modeling. Namely, sequential arrangements of the elements according to construction logic, object semantics, and additional information. The BIM paradigm intends to facilitate the modeling task for the architect through the given pre-modeled objects, while restricting his modeling to buildable realities. All elements created must belong to a building category, with a specific role in the building’s structure, such as slabs, columns, beams, roofs, doors, windows, among others.

Furthermore, when modeling for BIM software, users are obliged to follow a specific order of modeling. While in CAD the user may model, for instance, doors or windows anywhere in space with no precedence needed, in BIM these elements cannot be created if a wall to host them is not provided first. Other imposition are present in the modeling operations, for instance the need to define levels, used as reference for the location of all elements.

As part of the transition process, we propose modeling in CAD having in mind intermediate abstractions that might help overcome some of the translation issues. These abstractions are implemented by the user on his program, using operations to model in CAD, yet already subdivided into construction categories, as well as following a construction logic. For instance, instead of using pre-defined operations, such as the creation of a box or a cylinder, the user should define intermediate functions, e.g., ‘slab’ and ‘column’ which then invoke the pre-defined operations.

Although the CAD tool may not actually benefit from this organization of the program, it not only helps the user organize the modeling tasks, but also eases the translation to BIM, as it compels him to think in constructive terms from the start. Furthermore, when using intermediate abstractions that correspond to BIM pre-defined operations, such as a slab, column, or wall, the transition to BIM may be as simple as deleting the CAD operation invocation inside the function and rearranging the parameters to fit the BIM primitive.

Evaluation

To evaluate our approach and exemplify some of the steps, we selected two case studies: the Wadala Tower in Mumbai from James Law Cybertecture, and the Astana National Library from BIG architects. Both projects present rather complex shapes that categorize them as buildings that clearly benefit from an algorithmic-based approach.

The following sections present an extended description of the steps followed to algorithmically model the constructive elements, such as slabs, beams, columns, walls and doors, in a CAD environment, and their consequent translation to BIM. We compare the modeling process used for either paradigm, their advantages, and their disadvantages regarding one another. Figure 1 shows the 3D models of the two mentioned case studies, generated in CAD and BIM applications.

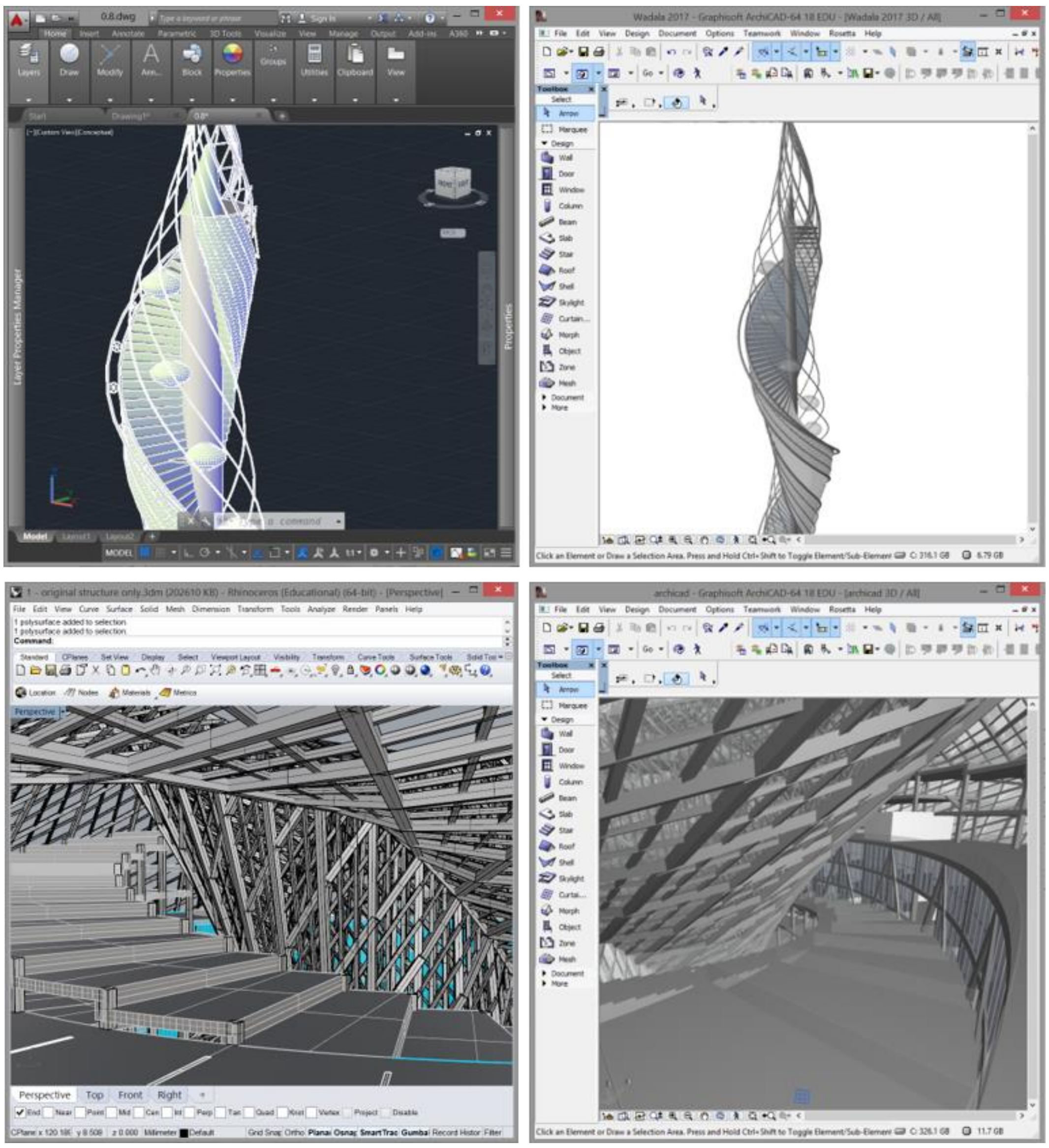

Figure 1
Wadala's 3D model generated in AutoCAD and ArchiCAD (on top) and Astana's 3D model generated in Rhinoceros and ArchiCAD (on the bottom)

Modeling tool

The case studies were implemented in Rosetta, a portable generative design tool capable of generating parametric models in both CAD and BIM tools (Feist, et al., 2016). Rosetta's abstraction layer contains the common functionalities amongst CAD tools, such as procedures to create geometric shapes, like circles and boxes, and procedures that apply geometric transformations, including lofts, extrusions and sweeps (Leitão & Lopes, 2011).

Rosetta also includes operations for modeling the parts of a BIM model, such as slabs, walls, columns, and beams, among others. These operations are common to the two currently integrated BIM tools, meaning that from the same algorithmic description, Rosetta is able to produce similar models in both tools. Nevertheless, many operations are only available in specific tools, and with this in mind, Rosetta also provides the possibility to work with specific functionalities of each tool. For this purpose, however, the user must relinquish the portability of his programs.

Slabs

The most common way of modeling a slab in a CAD paradigm would be to define its contour through a sequence of either lines or curves, create a surface from that contour and extrude it. In a BIM paradigm, the same contour can be used, but the lasts steps are unnecessary. The slab operation already entails the creation of a geometric element with a certain thickness, hence the user must only provide the contour. The thickness can be defined by the user or it can be left as the default definitions of the chosen family dictate.

A problem arises when Boolean operations are used to create the slabs in CAD. For the Wadala tower, the half-circle-shaped slabs were originally modeled through the subtraction of a parallelepiped to a cylinder. Converting them to the BIM paradigm required us to change the way we were conceiving them. We had to rethink the slab's shape and define its contour as a sequence of a line and an arc, which could be recognized by the BIM operation.

A similar issue was found in Astana's case, where the center slabs had a ring-shape. These were modeled subtracting a center cylinder to a group of broader ones. For BIM we had to implement a function capable of creating holes in a slab, but not via subtractions like in CAD. Instead, the function should receive two sequences of curves, one for the slab contour and another for the slab hole. In the BIM paradigm, the subtraction is implicit in the creation of an opening, since an opening in a slab is, by definition, a void.

Columns and Beams

Modeling vertical columns in a CAD environment follows a similar logic to the slabs: an extrusion of the columns' profile. If the columns are not vertical, however, the process would be closer to the one we might also use for beams: a loft between two sections positioned in space, a sweep of the section along a line virtually drawn between the column's or the beam's start and end point, or an extrusion as well, only using a vector as input instead of a single measure.

In a BIM paradigm, none of these operations are available in order to restrict the creation of columns and beams to what is buildable. Therefore, we must rethink our approach and adapt it to BIM functions.

The column function in a BIM program usually requires a base point, the specification of the level at which is it created and the level immediately above (assuming the column covers only one floor) and an angle at which the columns is inclined. The beam function requires only two spatial locations, however the geometry created by both functions is not the same. Columns have flat bases, always parallel to the ground floor, whereas the beam section remains perpendicular to its axis. Either function benefit from the set of profiles offered in the programs' libraries and family, most of which also have changeable parameters for the element's section.

Whereas in CAD there was little distinction between columns and beams, in BIM there is a considerable difference. In Astana's case, the cross-beams of the ring-volume were, in fact, modeled using columns as it only made sense for them to be sectioned by their top and bottom slabs, like they would be if they were constructed.

Walls

Modeling walls in CAD, much like slabs and vertical columns, usually requires the extrusion of the wall's base profile. An alternative approach, using a sweep of the wall's vertical profile along the wall's base line, might be preferred if the wall's position is not parallel to any of the axis.

In BIM applications, the wall function is available, waiting to receive two locations correspondent to the beginning and end of the wall, the level at which it is created and the level above. The wall profile can, naturally, be selected from the available families or libraries. This would seem like a smoother transition than the ones described before, yet our case studies presented quite some challenges.

The walls of Astana Library are arrayed in a circular path (see Figure 2). We created a function to distribute the two wall sets along the frame's spacing and the four floors. In our CAD approach, they were built from vertically sweeped rectangles (intended to be the wall profile) along virtual lines created in plan from the beginning to the end location of each wall. For a BIM application, we had no need to model the wall profile, as we did for CAD, since the program assumes the family values by default, such as thickness and material. Hence, the program was significantly simplified.

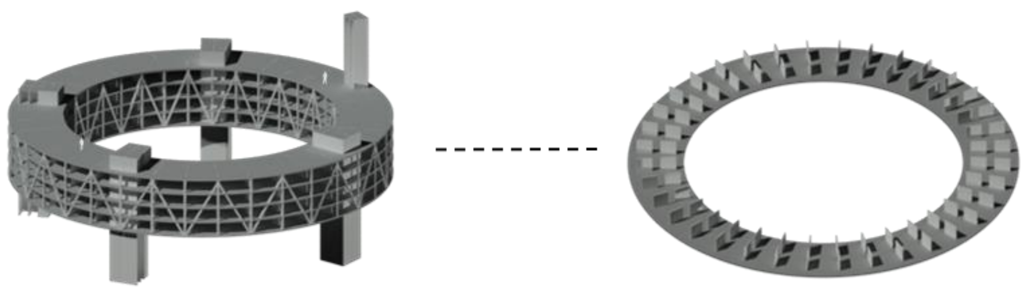

Figure 2
Astana's interior central volume (on the left) and a detail of the wall placement on one of the floor (on the right)

Glass Walls

Other commonly modeled elements are glass walls, that in a CAD paradigm do not differ much from a normal wall, except with a smaller thickness to represent the glass perhaps. If the user so wishes, he may also model the metal framings that hold together the glass panels. However, in a BIM paradigm, he may take advantage of the pre-modeled elements, which already present all this detail and with changeable parameters. The curtain wall command allows the creation of sequences of rectangular glass panels framed by metal mullions and transoms. For different BIM applications, the same element might require a different order of the parameters, but generally speaking, to model a curtain wall one would need to supply the guiding points of the wall, the number of panels in the grid and their respective lengths, and, optionally, the sizes for the framings.

Modeling Wadala tower, we had to introduce a glass wall on the edge of every half-circle slab. For CAD applications, these were made via the extrusion of polygonal surfaces perpendicular to the slabs (Figure 3. left). Transitioning to BIM, however, we could take advantage of the pre-modeled elements, namely the curtain wall, in this case, automatically instilling more detail in the model that we did not have to model ourselves (Figure 3. right). The exact same process occurred in the glass wall of Astana National Library.

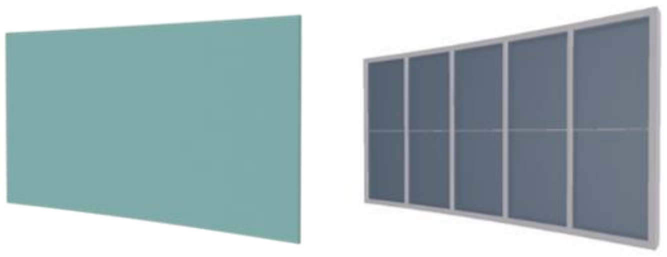

Figure 3
Glass Wall surface in CAD (on the left) and Curtain Wall object in BIM (on the right)

Doors

Delving in to more detail, we come to doors. In a CAD environment, in an initial modeling phase, one might model a door as a hole on the wall. On a more advanced stage, the user might want to give it some more detail. However, the more detail we want, the more work we need to put into the modeling task.

Shifting to the BIM paradigm, we find several pre-modeled door types, so there is no need to model them. Moreover, if further ahead we had wished to change the door type, in BIM we only need to specify the type, whereas in CAD we would have to model a new design from scratch. Figure 4 presents three examples of door types changed with little effort in the BIM model. Furthermore, the insertion of a door in a wall does not require the creation of the hole in the wall, as it did in CAD. That process is implicit in BIM, although some limitations are imposed, for instance, as we may not create a door without a wall to host it. This mandatory modeling order sequence does not exist in CADs, where the user is allowed to model doors wherever he so desires, with no precedencies needed.

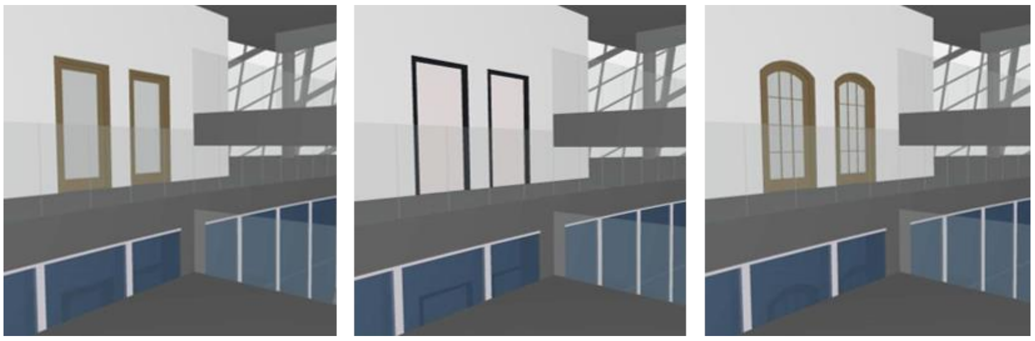

Figure 4
Astana's cores with three possible door types

Building's skin

The modeling elements that usually present the biggest difficulties in the CAD-BIM transition are the ones we design without real consciousness of what structural elements they might represent. In such cases, the true challenge is the rethought of the design elements as construction elements. Our chosen case studies brought up some interesting situations, particularly regarding the buildings' skin.

The skin bars of Wadala Tower were modeled in CAD using sweeps: circular sections with the bars' radiuses would be dragged along path-curves, which twisted around the building core as they rose along the building height. Transitioning to the BIM paradigm, we had to convey structural meaning to these bars, and the solution we envisioned was to convert them into columns. Instead of path-curves, we used lists of points calculated at each floor height to generate a set of columns at each floor in replacement of the sweeps. However, the column function in the BIM paradigm required, not two points like in CAD, but the base-location and an angle to successfully incline the column. Hence, we implemented a column function of our own that, given two spatial locations, calculates the angle between them, along with the base point and the level heights. Figure 5 shows the façade skin with a set of columns highlighted in white.

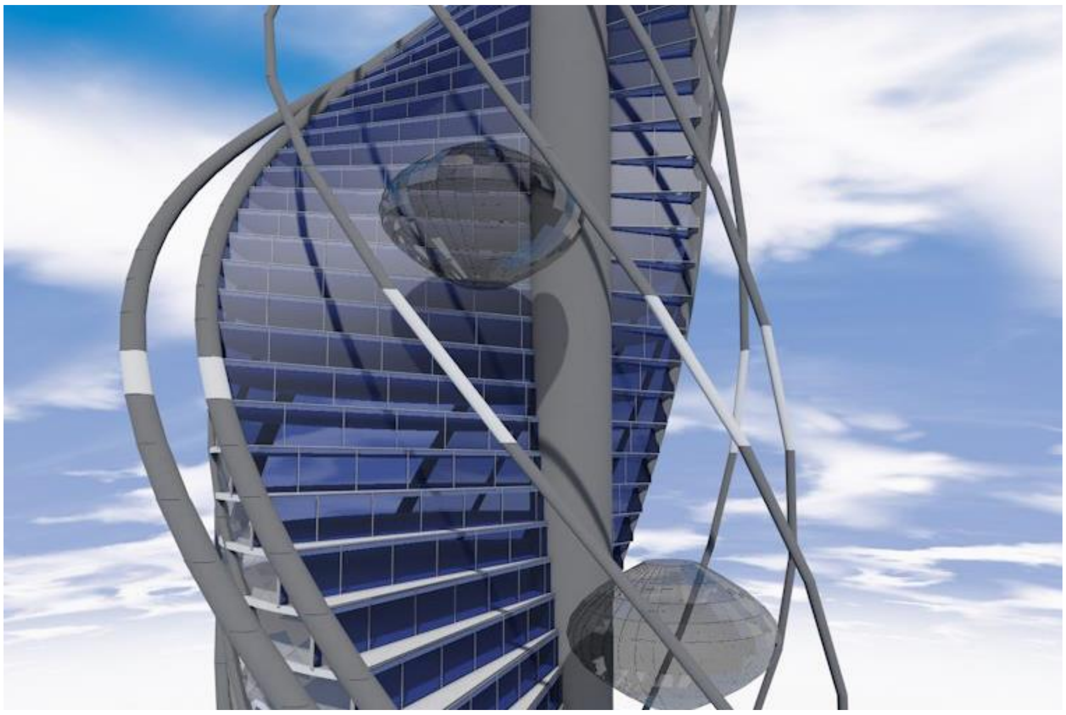

Figure 5
Highlighted columns (in white) from Wadala's façade skin

In Astana's case, sweeps were also being used to generate the façade's structure. Only, instead of a circular section, a quadrangular one was dragged along multiple rectangles. In this case, we believed beam objects would be more appropriate as these elements had no apparent relation to the existing floor organization. We used the matrix of points already calculated during the CAD approach to generate the beams as well.

However, a problem arose in a specific BIM application: ArchiCAD, in defense of the constructive logic, does not allow the production of vertical beams nor horizontal columns. This presented an obstacle to our approach: as the façade completed a whole loop in its twisting movement, some of the structural elements assumed vertical positions.

To surpass this issue, we implemented a function capable of determining if the given two locations for the creation of one beam aligned vertically. If they did, a column would be created instead of a beam.

Façade Glass

A more challenging case is presented by curved façades covered in glass panels. These panels commonly have triangular shapes, to allow the grid to adjust to the façade. In a CAD environment, they are easily represented by triangular surfaces placed in space, possibly extruded to glass thickness. The user may also model the glass framing using lofts or sweeps, if he so wishes.

However, in the BIM paradigm there is no specific category of objects containing these elements, and the only elements that allow anything close to freeform modeling are morphs or masses. These elements, however, will not grant the object the desired semantics. Belonging to a glass or curtain wall category, for instance, would be a more appropriate semantic for a façade panel. To this end, the user would have to model new families adequate for the project.

Due to the triangulated grids of glass panels forming their façades, both case studies suffered from this problem. For CAD, they were modeled as triangular surfaces, while for BIM we began using morph panels with glass for material (see Figure 6). The projects are still in development, as is the chosen tool to model them, Rosetta. In the future, the tool is planned to allow the modeling of project specific elements, when no object provided by the BIM library fits the design purpose.

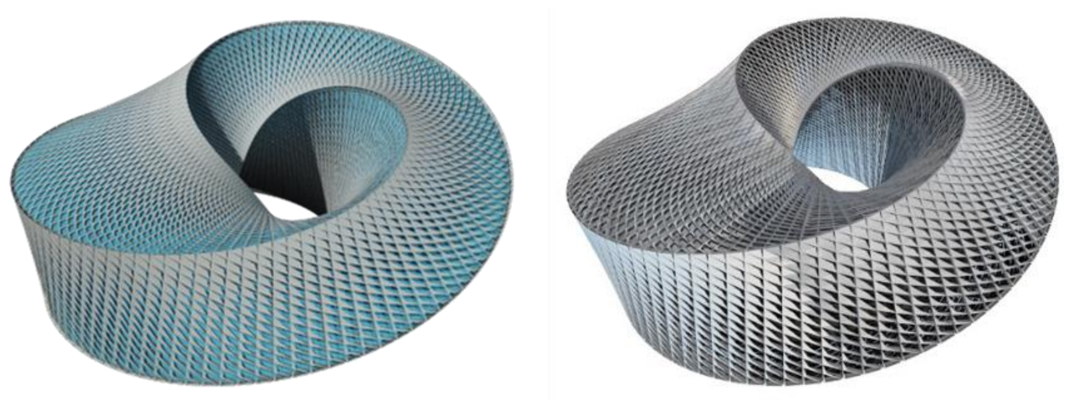

Figure 6
Astana's final model in CAD (on the left) and in BIM (on the right)

Conclusion

In this paper we presented a methodology for CAD-BIM transition in the context of Algorithmic Design. We compared the two paradigms and the advantages and drawbacks each presents to the design process of an architectural creation. We exemplified the application of our methodology with two case studies we have modeled in a portable generative design tool, and that have undergone this translation process. We thoroughly explained both modeling phases, CAD and BIM, and the changes the program suffered in order to be ported to the second paradigm. We proposed a modeling sequence suitable to the BIM paradigm and we addressed some interdependencies that characterize the BIM logic for object creation.

During the translation process we verified a general simplification of the algorithmic description of the design. Shifting to the BIM paradigm, most geometry is implemented in the program through intermediate abstractions (such as slabs, columns, beams, etc.), since the user can benefit from pre-modeled objects' descriptions. The translation process also presented some limitations, however, as the CAD paradigm allows for higher levels of modeling freedom, when compared to BIM. As such, sometimes the user must envision more complicated solutions in order to convert his geometry into BIM elements.

Finally, we concluded that the ability to create new families is of crucial importance to the architect. The use of pre-modeled elements should be an advantage to the modeling process, but never a limitation to creativity. Furthermore, the architect should not have to feel compelled to change the design only to adapt to the existing objects. Hence, the possibility of altering or even creating our own families is essential to the full and correct use of the BIM paradigm. So far, Rosetta has missed to include this feature due to the significant differences between the different BIM tools, but future developments will focus on making this possible.

Acknowledgments

This work was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UID/CEC/50021/2013.

References

Burry, M. (2011) *Scripting Cultures*, United Kingdom: John Wiley & Sons Ltd.

Eastman, C., Teicholz, P., Sacks, R. and Liston, K. (2008) *BIM Handbook: A Guide to building Information Modeling for Owners, Managers, Designers, Engineers, and Contractors*, Hoboken, New Jersey: John Wiley & Sons, Inc.

Feist, S., Barreto, G., Ferreira, B. and Leitão, A. (2016) 'Portable Generative Design for Building Information Modelling', Living Systems and Micro-Utopias: Towards Continuous Designing, Proceedings of the 21st International Conference of the Association for Computer-Aided Architectural Design Research in Asia CAADRIA 2016, Hong Kong, 147–156.

Kalay, Y. (2004) *Architecture's New Media: Principles, Theories and Methods of Computer-Aided Design*, Massachusetts: The MIT Press.

Kensek, K. and Noble, D. (2014) *Building Information Modeling: BIM in Current and Future Practice*, Hoboken, New Jersey and Canada: John Wiley & Sons, Inc.

Leitão, A. and Lopes, J. (2011) 'Portable Generative Design for CAD Applications', ACADIA 2011: Integration through Computation: Proceedings of the 31st annual conference of the Association for Computer Aided Design in Architecture (ACADIA), Banff, 196-203.

Meredith, M. (2008) 'Never Enough (transform, repeat ad nausea)', in Sakamoto, T. and Ferré, A. (ed.) *From control to design : parametric/algorithmic architecture*, Barcelona, New York: Actar-D.

Shepherd, P., Hudson, R. and Hines, D. (2011) 'Aviva Stadium: A parametric success', *International Journal of Architectural Computing*, vol. 09, no. 02, pp. 167-185.

Terzidis, K. (2006) *Algorithmic Architecture*, Abingdon and New York: Routledge, Taylor & Francis Group.

Zboinska, M. (2015) 'Hybrid CAD/E platform supporting exploratory architectural design', *Computer-Aided Design*, no. 59, pp. 64–84.