Shape Grammar Libraries of European Classical Architectural Elements for Historic BIM

Maurice Murphy[1] Eimear Meegan[1] Garrett Keenaghan[2] Alain Chenaux[2] Anthony Corns[3] Stephen Fai[4] Lara Chow[4] Yu Zheng[4] Conor Dore [5] Simona Scandurra[6] Andrew Tierney[7] Filippo Diara[8] Fulvio Rinaudo[8] Oriel Prizeman[9]

[1]Virtual Building Lab Dublin morris.murphy@TUDublin.ie
[2]Technological University of Dublin
[3]Discovery Programme Ireland
[4]CIMS Carleton University
[5]MMA Consulting
[6]Politecnico di Milano
[7] Trinity College Dublin
[8]Politecnico di Torino
[9]Cardiff University, PrizemanO@cardiff.ac.uk

KEY WORDS: HBIM, HGIS, Digital Architectural Heritage, CAD, Shape Grammars

ABSTRACT:

This paper proposes a design for libraries of European Classical architectural elements based on shape grammars. This design is based on a workflow which develops library objects from 3D CAD primitives using architectural rules to construct parametric representations of architectural elements. In the case of Classical architecture, the design and detail for the parametric objects are based on manuscripts ranging from Vitruvius to Palladio to the architectural pattern books of the eighteenth century. The generation of 3D objects for virtual reconstruction necessitates the application of computer algorithms and rules introduced by the user to generate objects, buildings and spaces from a grammar and vocabulary of shapes. Both the use of graphicly constructed and coded parametric libraries in formal and open-source platforms will be considered here.

1. INTRODUCTION

Historic BIM and GIS workflows involve the mapping of architectural heritage elements as parametric digital library objects within a 3D geometric framework, which uses historic and/or remotely sensed survey data to virtually represent whole buildings, collections of buildings, or objects. The purpose of the current paper is to consider how best such library objects might be generated. To do this, an approach based on shape grammars, which was developed by Dore and Murphy (2013) and which proposed the use of architectural rules to construct parametric representations of architectural elements from 3D CAD primitives, will be revisited and updated by a multidisciplinary group. In the case of the classical architectural examples discussed here, the design and detail of the parametric objects are informed by historic documentation drawn from the works of Vitruvius and Palladio, as well as the architectural pattern books of the eighteenth century, all of which record the advanced rules adhered by Renaissance architects and which strongly support the design of parametric library objects.

2. SHAPE GRAMMARS

2.1 Shape Grammars

The generation of 3D heritage objects necessities the application of computer algorithms in combination with architectural rules based on a grammar and a vocabulary of shapes. A high-level language based on the early concept of shape grammar which was introduced by Stiny and Gips (1971) is here proposed as a design approach. A set of procedures that considers shape in terms of a primitive transformed by a set of production rules leading to a realised construct in design underpins the language. This is illustrated in its simplest form in figure 1, which shows an initial non-terminal shape altered by a set of production rules to generate a terminal shape.

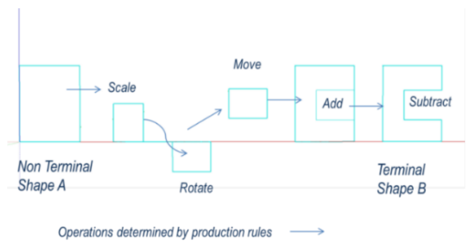


Figure 1. Shape Grammar Production Rules and Procedures.

The vocabulary of shapes comprises a finite set of non-terminal and terminal shapes that can be expressed as a a collection of points, lines and planes. In figure 1, the non-terminal shape A is replaced by terminal shape B through the application of a series of Euclidian transformation production rules, namely scale, rotate, add and subtract. The production process terminates when no more rules can be applied, and all non-terminal shapes have been removed. Production rules are applied in the form A→B where A and B are non-terminal and terminal shapes. When a rule is applied, the shape on the left-hand side is replaced by a new shape to the right. Shape rules are applied to a shape with an assignment of real values to the parameters and with additional transformations as required.

2.2 Parametric Shape Grammars

Building on his earlier work with Gips, Stiny subsequently (1980) proposed the concept of a parametric shape grammar (see figure 2) according to which nonterminal shapes are defined by coordinates at the vertices of a shape. These variable parameters associated with shapes can be adjusted and used to replace the non-terminal shapes.

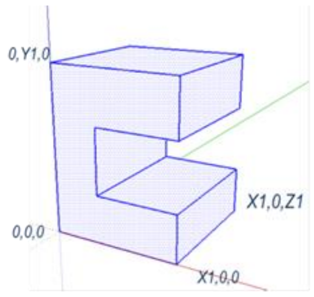


Figure 2. Parametric Shape Grammars.

2.3 Parametric Buildings as Library Objects

Much like their modern counterparts, when classical detail and ornament is removed from historic buildings, they can be represented by a series of blocks or cylinders with related window openings formed by holes or voids. Extruded building blocks can quickly create whole buildings or urban centres thus allowing for 3D mapping of cities or towns. The shape grammar in figure 3 (starting with the left side of the figure) shows the primitive block shape developed for facades without the addition of ornament, a logic which can be applied to whole buildings and streets etc. The window and door openings are multiplied according to the architectural rules and traditional building technology practices, while accuracy and level of detail can be improved by introducing additional shape grammars for curves and deformation in the block structures.

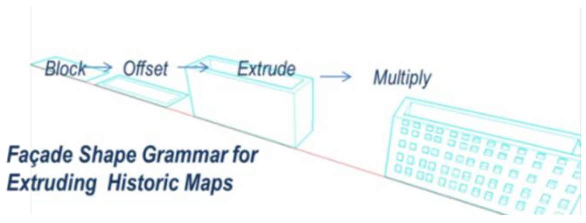


Figure 3. Applied Shape Grammars to facades and street Classical Architecture Dublin.

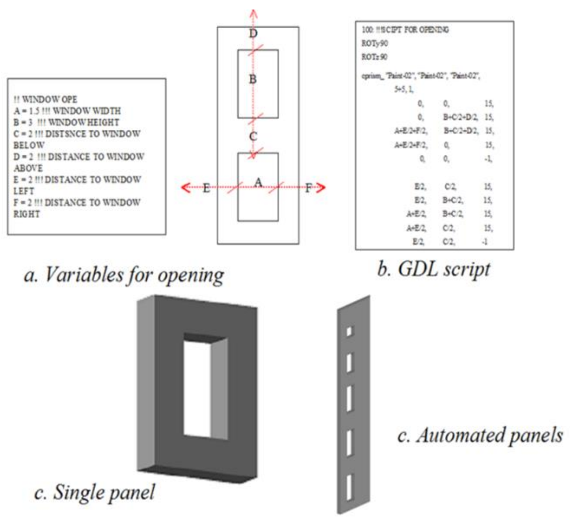
3. CONCEPTUAL DESIGN AND INTIAL IMPLEMENTATION

Informed by Dore and Murphy’s initial work (Dore & Murphy 2013, Dore 2017), the revised framework presented here is bolstered by more recent associated research, incorporating automatic mapping of remotely sensed data. Implementation of the Shape Grammar Libraries is, furthermore, carried out within Autodesk’s Revit and ArchiCAD (and associated plugins such as Dynamo GDL etc.), with the use of graphicly constructed and coded parametric libraries in formal and open-source platforms also considered.

3.1 Script Based Parametric Modelling using GDL

Geometric Description Language (GDL) is a programming language used to create parametric objects within ArchiCAD’s BIM software. Defined by a syntax similar to that of the BASIC programming language, GDL provides many functions for creating 3D parametric objects either using primitive shapes such as blocks, spheres, cones and ellipses or by generating shapes from 2D outlines. It also uses coordinate transformation commands stored in a stack to position multiple objects relative to each other, and allows for graphical editing of parameters, complex Boolean operations, various control statements and the use of mathematical functions in creating parametric objects. Also provided is the ability to script a specific user interface for objects and their parameters. Although designed for parametric modelling, the scope of its capability is such that it lends itself to the generation of shapes using shape rules and shape vocabularies of the kind determined by a shape grammar, as illustrated below using a wall panel as an example.

3.1.1 GDL Example – Wall panel: Within 3D space, a typical wall panel, as shown in figure 4, can be used as a tile, positioned and multiplied to make up a full façade, which, in turn, can then be multiplied to make a collection of building façades and so on. As in figure 2, the panel as a parametric object is defined by the coordinates at its vertices, variable parameters that can be adjusted to reflect multiple geometric variations. In the case of the illustrated example, the variables introduced include the width of Window A, the height of Window B, the distance between the windows A and B (C), the distance between Window B and that immediately above it (D), and the distance to the windows immediately to the left (E) and right (F) of Window A. The panel is represented by a block or prism with the openings created using a Boolean script and the wall thickness introduced as a variable. While discrepancies can occur between the façade as surveyed and the scripted geometry which is based on historic building technology practice, further detail can allow the single panels to be adjusted for any opening size or distance between openings (Murphy, 2012).



```
!! WINDOW OPE
A = 1.5 !!! WINDOW WIDTH
B = 3  !!! WINDOWHEIGHT
C = 2 !!! DISTSNCE TO WINDOW
BELOW
D = 2  !!! DISTANCE TO WINDOW
ABOVE
E = 2 !!! DISTANCE TO WINDOW
LEFT
F = 2 !!! DISTANCE TO WINDOW
RIGHT
```

```
100: !!SCIPT FOR OPENING
ROTy90
ROTz90

cprism_ "Paint-02", "Paint-02", "Paint-02",
            5+5, 1,
                  0,          0,          15,
                  0,          B+C/2+D/2,  15,
            A+E/2+F/2,        B+C/2+D/2,  15,
            A+E/2+F/2,        0,          15,
                  0,          0,          -1,

                 E/2,         C/2,        15,
                 E/2,         B+C/2,      15,
               A+E/2,         B+C/2,      15,
               A+E/2,         C/2,        15,
                 E/2,         C/2,        -1
```

Figure 4. Wall panel Sample GDL Script**.**

3.2 Visual Programming - Dynamo

In visual programming, procedures, operations and commands are executed through a graphical (or ‘visual’) user interface as opposed to using text bound by syntax. Dynamo is a visual programming tool that provides one such interface within Revit, thus allowing for access to the Revit API. Figure 5 shows the GDL-scripted wall panel discussed above, this time built in Revit, using Dynamo, as a parametric object, thus allowing for the modification of any constraints relating to its geometry. The most basic point in space can be parameterized using values of x, y, and z based on Cartesian coordinates. Such points can, furthermore, be connected to form lines, while the lines can be closed to form surfaces, and the surfaces can be extruded to form entities. In this way, any geometric shape can be parameterized by using the point coordinate value parameters and combining them with certain geometric rules. By further correlating the parameters, the coordinate parameters of the position of control points can be simplified to parameters with practical architectural meaning.

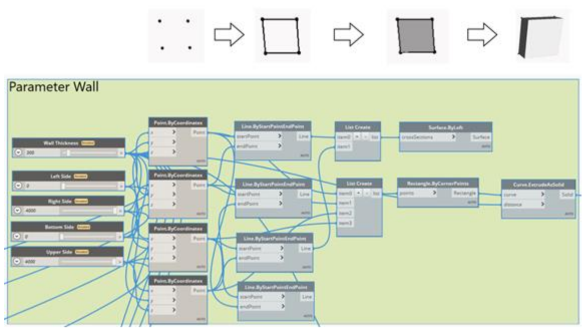


Figure 5. Detailed illustration of the Parameter Wall nodes group.

The parametrization of the wall panel as achieved using Dynamo’s node-based editor is laid out in Figure 5’s “Parameter Wall” node group. The column on the far left comprises five number slider nodes, the first of which allows for the modification of the thickness of the wall panel or the Y value of all points, while the remaining four control the x and z coordinates of the panel’s four corner points. The second column of nodes allows for the creation of these four corner points based on coordinate values, with the third column serving to connect these points or vertices to create four-line segments. The fourth and fifth columns of nodes create a rectangular surface from these line segments, and the final "extrudeAsSolid" node extrudes the rectangular surface into a 3 dimensional wall block. In the same way, a second, related node group labelled “Parameter Window Opening” governs the creation of window voids within the panel. In this case, the four corner point parameters of the opening are offset against the previous four parameters of the wall to control the distance between the window edge and the wall edge. By correlating these parameters using different Operator nodes, various parameter control methods can be set up, such as constraints between the parameters at a fixed distance or at a fixed proportion. Finally, the window void is removed from the wall by means of the "Solid.Difference" node, thus establishing the geometric shape of the wall panel. This geometry can be imported into Revit through the "ExportToSat" node, or exported as a FamilyType through the "FamilyType.ByGeometry" node.

4. SHAPE GRAMMAR PARAMETRIC DESIGN FOR HISTORIC FACADES

The basic elements that make up the vocabulary of shapes $\{S\}$ of a historic façade can be seen in figure 6. These include two wall tiles; one (TW) which contains a window opening and surrounding wall, and another (TD) which is intended to contain a door opening. Additional library objects relating to doors and door cases such as columns and pediments are linked with this latter shape. In addition to these tiles, other shapes include parametric library objects for windows (W) and a simple block (BL) that is used to create ashlar masonry detail.

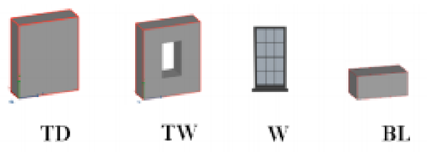
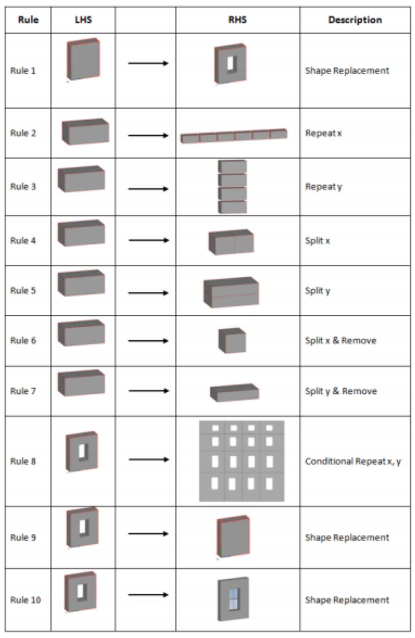


Figure 6. Basic shape vocabulary elements $\{S\}$ for parametric shape grammar design.

The shape rules $\{R\}$ applied to these objects are shown in Table 1, with the initial shape illustrated in the left-hand side column and the resulting shape, following the application of the rule, illustrated in the right-hand side column. Each rule is applied with an assignment of real values to shape parameters and transformations if required, such as translations, rotations, scaling or mirroring. Rule 1: replaces the initial shape $\{I\}$ with the shape TW by adding an opening to the solid shape. The new shape TW has new parameters to define the coordinates of the opening. Rule 2: repeats a shape along the x-axis. Parameters are used to control the number of repetitions or a distance which specifies the number of repetitions. Rule 3: is similar to Rule 2 but repeats shapes along the y-axis. Rule 4: splits a shape along the x-axis into smaller or separate shapes, with Parameters controlling the positions and number of splits. Rule 5: is similar to Rule 4 but splits shapes along the y-axis. Rule 6: splits a shape along the x-axis and removes one of the resulting segments. Rule 7: is similar to Rule 6 but splits a shape along the y-axis and removes one of the resulting segments. Rule 8: represents a conditional repeat (described below). Rule 9: replaces a window tile TW with a door tile TD Rule 10: adds a selected window W from the library to a window tile TW.
As illustrated in figure 7, Rule 8 serves as the primary rule in the creation of a façade arrangement. This rule repeats the wall tile TW in both x and y directions based on various parameter settings and architectural rules. This method contrasts to the concepts adopted in the CGA shape grammar (Muller et al. 2006) where a façade is split into smaller tiles. Rather, it repeats the input shape in the y direction for the first column, and then moves to the second column and so on until all tiles have been placed according to the specified parameters. Parameters for each repeated instance of the shape TW are calculated based on architectural rules and the position of that instance in a façade arrangement. Parameters for the “number of floors” and “number of columns” control the number of repeated instances along the x- and y-axes. The input for this rule is the shape TW and coordinates for this shape which are calculated and assigned as shapes parameters.



Rule	LHS		RHS	Description
Rule 1		→		Shape Replacement
Rule 2		→		Repeat x
Rule 3		→		Repeat y
Rule 4		→		Split x
Rule 5		→		Split y
Rule 6		→		Split x & Remove
Rule 7		→		Split y & Remove
Rule 8		→		Conditional Repeat x, y
Rule 9		→		Shape Replacement
Rule 10		→		Shape Replacement

Table 1. Shape rules $\{R\}$ for parametric shape grammar design.

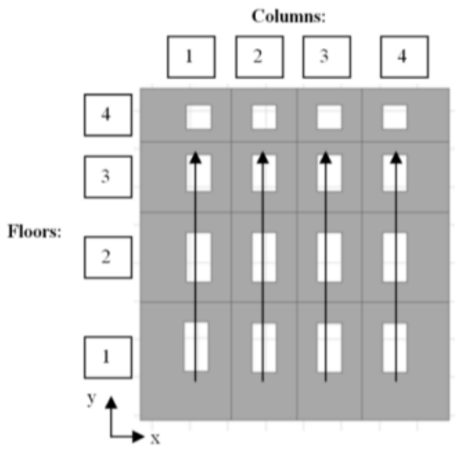


Figure 7. Repeated wall tile resulting from the application of Rule 8 to shape TW.

Figure 8 shows the order and application of some of the rules listed in Table 1 in the creation of a basic façade arrangement. Here, beginning with use of Rule 1 to create a single wall tile containing a window opening, Rule 8 is then applied to create a four-column, four-floor façade. The application of Rule 9 adds a door tile TD, the position of which is determined by a user defined parameter “windows to left of door”. The application of Rule 10 adds window objects to all window openings on the façade. Global parameters for window objects can be entered and set from the objects dialogue box. Specific parameters for a particular instance of a window object can be set using graphical parameter editing.

Another application of the rules in Table 1 can be used to create a parametric ashlar block wall that can be automatically combined with the parametric wall façade. Parameters allow the user to add ashlar block wall detail to the ground floor or all floors of the façade. Parameters of the block wall enable the user to change the individual block size, mortar spacing and texture.

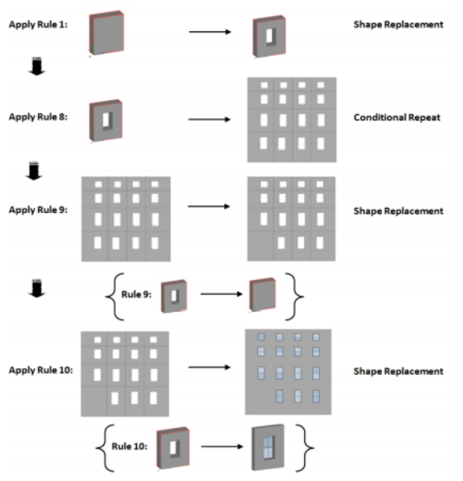


Figure 8. Repeated wall tile resulting from the application of Rule 8 to shape TW.

4.1.1 Procedural Modelling Whole Buildings

Procedural Modelling Whole Buildings: A second, procedural approach allows for the semi-automated modelling of existing buildings, with the required geometry first generated automatically and then manually refined to match to specific survey data. Crucially, unlike that outlined above, this second method facilitates the modelling of all faces of a building and not just a single façade. This requires the use of procedural rules capable of generating many types of building shapes. These rules are designed to interact with existing 2D building footprints which enables the automatic generation of any building shape. After the structure of a building is generated, other procedural rules can then be applied to split a building face into tiles and automatically add building components.

The first procedural rule is used to extrude a building footprint (FP) or user-drawn 2D polygon to create a mass model (MM) for a particular floor or building (figure 9). This allows the procedural modelling techniques and rules developed to be applied to any building shape. The volumetric mass model can be automatically generated from an existing building footprint, or a user can define the footprint by drawing a new 2D polygon. This rule can be applied to any closed polygon which can contain both lines and arcs. The rule will automatically generate a volumetric mass model that can contain planar and curved surfaces. The height at which the building footprint is extruded to is a parameter of this rule. Users can select a single building footprint or multiple building footprints (FP) to automatically generate any number of building models at once.

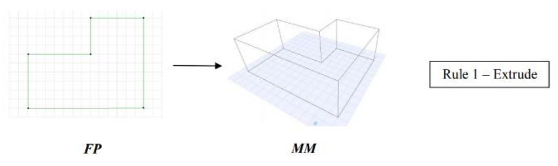


Figure 9. Rule 1 - extrudes a building footprint**.**

The input data automatically extracted from selected polygons include the number of selected polygons, the number of sides per polygon, the x- and y-coordinates of polygon nodes, the number of arcs in each polygon (if any), the x- and y-coordinates of arc start and end points and arc angles in radians (Figure 11). If any polygons contain arcs, then a number of initial calculations are performed to acquire data relating to these arcs for later operations.

The second rule (Figure 10) converts a mass model into a higher level of detail (LoD) model which is represented by walls with a uniform thickness. These walls are created from the mass model by defining an opening in it using an algorithm which offsets the 2D building footprint by a distance equal to the wall thickness, with the resulting offset polygon then removed. Wall thickness, which is the offset distance, is a parameter for this rule and can be altered by a user. The new lines and arcs created by this offset operation are not connected; rather they are either intersecting or do not meet. To overcome this, intersections between two lines, between individual lines and arcs or between two arcs, depending on whether two adjacent sides are lines of arcs, must be calculated.

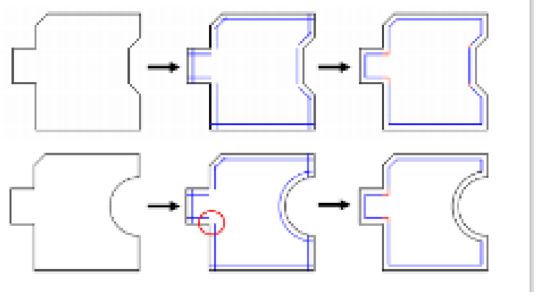


Figure 10. Rule 2 - converts a mass model into a higher level of detail (LoD) model.

The third procedural rule is a repeat rule that is used to repeat a wall instance for any number of floors specified by a user (figure 11). According to this rule, a loop is used to create new wall instances for each floor. By default, new wall instances created have the same footprint as the ground floor but, using Rule 1, it is possible to extrude different footprints on each floor. When the application of Rule 1 (Procedural Extrusion) is followed by that of Rule 3 (Repeat), the heights of each floor (extrusion height) are automatically calculated by applying the classical proportions and architectural rules. The inputs for this are the parametric wall objects created using previous rules, associated polygon data and architectural proportions which are used to setup initial floor heights with classical proportions. The outputs include parametric wall objects for any number of floors with classically proportioned floor heights and an interactive editing function that allows for the adjustment of the number of floors and floor heights (Figure 11).

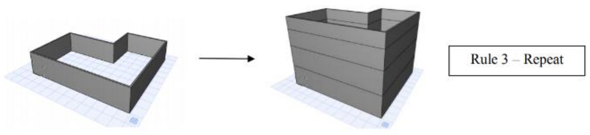


Figure 11. Rule 3 - Repeat Wall Instance for Any Number of Floors.

The fourth procedural rule is a split rule which is used to subdivide a façade or floor into any number of tiles which may contain openings (figure 12). Whereas the wall tiles of the previously described façade prototype are created by repeating tile instances, the procedural building prototype creates tile instances by subdividing existing wall geometry, with the number of tiles to be created serving as the parameter for this rule. Users can apply a split rule to a complete building, all floors on a particular building side or a specific floor on a building side. Different floors on a façade can also have different numbers of tiles.

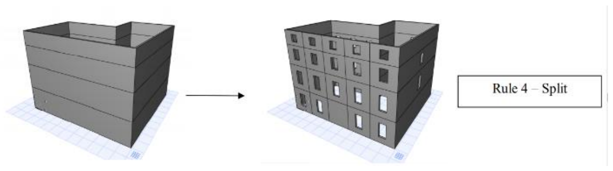


Figure 12. The fourth procedural rule is a split rule which is used to subdivide a façade or floor.

Once a building structure has been created using rules 1-4, additional parametric library objects and shapes can then be automatically added. The fifth procedural rule, a replacement rule, is used to replace existing geometry with new shapes or add new shapes to existing geometry. The referencing and semantic information attached to wall tiles allows additional detail and objects to be easily placed anywhere on the building. Objects can be added to a specific tile or groups of tiles with semantic attributes. When an object is placed on a tile, the semantic attributes describing the position of the tile are also associated with the instance of the object placed on that tile. An example of an application of this rule can be seen in Figure 13.

Procedural rules can also be applied to multiple building footprints at once, as illustrated in figure 14 where building footprints for an area of Dublin city centre have been used to automatically generate building models. To achieve this, an Ordnance Survey Ireland (OSI) dataset comprising 3,610 building footprints was imported into ArchiCAD and the relevant procedural modelling rules were applied.

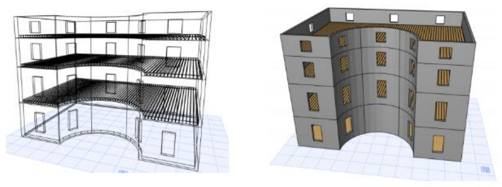


Figure 13. Rule 5 - Replacement Rule.

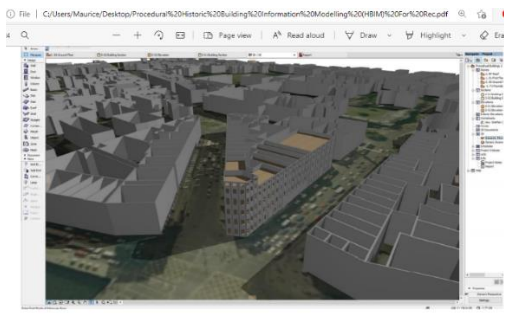


Figure 14. Procedural rules applied to multiple building footprints.

4.2 Classical Orders and Architectural Rules

Ideally suited to algorithmic expression, generative shape grammars have been developed and used in the virtual reconstruction of both archaeology and architectural heritage. In figure 15, the defining characteristics of the Classical orders are shown, upon which were formulated the rules which govern the distribution and combination of elements. These rules of classical architecture can perhaps best be understood in terms of a grammar of ornament and composition and thus, ultimately, a shape grammar, with the elements comprising each order (mouldings, profiles, symbols etc.) forming an architectural vocabulary. This classical shape grammar is defined and represented by limited arrangements of basic shapes in two-dimensional Euclidian space (see top left side of figure 15).
Similar to structural elements, shapes are governed by replacement rules whereby a shape can be changed or replaced by transformations and deformations. The shape commands, combined with a library of primitives, allow for all configurations of the classical orders in relation to uniform geometry. Non-uniform and organic shapes are developed through a series of procedures that involve deformation and Boolean operations, while also attempting to maximise the parametric content of the objects.

In previous work, (Dore and Murphy 2013) we designed library objects based only on the various interpretations of the Classical orders. However, in many cases, the architects and builders that drew on these rules did not adhere strictly to them and, as such, the use of these formal libraries can lead to inaccurate representations of the real-world object.

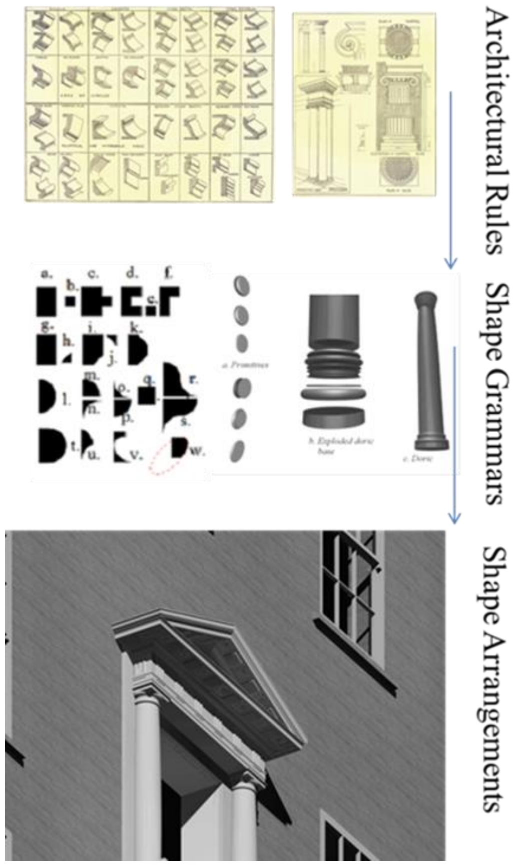


Figure15. Applied Shape Grammars to Classical Architecture.

4.2.1 Accuracy and Aesthetic principles in designing a reusable library of architectural elements.

As a prerequisite to designing a library, the myriad variations of the Classical orders in the bibliographic record will need a clear statement of its sources. The orders, as originally described by Roman architect Vitruvius in books 3 and 4 of his *Ten Books on Architecture*, presented architects from the fifteenth century onwards with a wide architectural grammar, which required interpretation and careful decision making (Morgan 1960). The relationship between temple plan, order, column height, and inter-columnation, explained in Latin, without illustrations, and with proportions in a dizzying array of fractions and subfractions, was a recipe for confusion and contradiction, which subsequent theorists and practitioners worked hard to resolve. It is worth bearing in mind that the first unabridged English edition of Vitruvius was not published in Britain until 1791, meaning that the only surviving architectural treatise from antiquity was largely filtered into architectural production via illustrated Renaissance and post-Renaissance texts such as Serlio, Vignola and Palladio, and a host of minor handbooks, that better served the practical needs of the architect and builder.

The representation of antique buildings in these texts, furthermore, was not always accurate. Antoine Desgodetz's late seventeenth-century survey of Roman buildings showed that Palladio's famous and influential *Quattro Libri*, published a century earlier, had misrepresented many details and tidied up irregularities in the proportions of many well-known Roman buildings (Loth 2014). Vitruvius' work presented similar problems, with comparative studies between the proportional principles in his text and surviving Roman buildings showing regular divergence from his proportional systems. One inhibiting factor was that the production of columns to a series of standard measures in Roman quarries reduced the degree of control architects had in the exercise of proportional systems (Bosman 2015). Indeed, Vitruvius himself often struggled to find examples of buildings in Rome that encapsulated the rules he was espousing. As a result, the proportions of columns shifted between the writings of Serlio (1537) and Philandrier (1544), becoming more slender to reflect observations in the field, and again in Vignola (1562) who introduced a more comprehensive system of proportion between all the parts based on the module of the radius, which he subdivided again into thirty parts to cover the proportions of all the more minor decorative mouldings. Following this, a variation of Vignola's approach was adopted by Palladio (Lemerle 2011, p. 3), and a further attempt to rationalise the orders was made by Claude Perrault (1683) who introduced yet another system with a module based on a third of the diameter of the column and subdivided by twenty parts (Lemerle 2011, pp 7-8). Beyond the work of theorists, practitioners could equally go their own way, with John Summerson describing the orders of the Roman Baroque architect Francesco Borromini as 'outrageous and extremely expressive inventions, entirely his own'.

5. OPEN SOURCE PLATFORM FOR LIBRARY

Recently, unconventional solutions and workflows have been the focus of several studies exploring FOSS (Free and Open-Source Software) BIM platforms (Logothetis et al., 2016; Diara et al., 2018; 2020a). This experimentation has sought to implement and improve the applications' default tools as regards the modelling of heritage assets.

One such offering, FreeCAD offers a dynamic BIM suite that facilitates the generation of parametric architectural families through its implementation of smart macros and workbenches. In addition to its modelling capabilities, it also allows for the conversion of freeform models to parametric objects, although this process is time-consuming and requires the use of additional plugins (Diara et al., 2019).

Parametric families have been created in FreeCAD using an initial metric framework of historical architectural contexts (LiDAR surveys). This process also involved the simplification of shapes, making general BIM modelling of custom elements possible. Crucially, this simplification was informed both by the information to be included in the model and also the project’s overall goal.

Within FreeCAD, parametric modelling is achieved using a combination of the Draft Workbench and the Part Module. The latter comprises a set of tools that represent the core modelling instruments of the software, tools which are founded on the use of OCCT (OpenCASCADE Technology) geometric primitives. It is this module that enables users to create and combine custom primitive shapes and, it is worth noting here, where such users possess good programming skills and an advanced view of parametric modelling, it can also be implemented using some custom Python scripts.

FreeCAD’s Arch Workbench supports the attribution and classification of architectural elements, as well as the creation of standard architectural entities (walls, roofs, windows, and so on). Models built using the associated Part Module can here be converted into BIM models through semantic attribution and classification according to the IFC (industry foundation classes) standard format. Once the objects have been constructed and classified, the related property menu can then be filled with relevant and useful categories of information such as descriptions, custom materials, identification codes and strings labels. These data improve the semantic dimension of single parametric objects as well as providing information on the architectural context.

A simple plugin or macro (*HBIM_library*) is currently under development that will allow the inclusion of custom parametric families previously designed within FreeCAD using Part modelling tools. In this way, a smart container with custom parametric elements (a digital library) will be easily implemented inside the software.

Within this context, however, the creation of parametric families of historical architectural components requires further attention as regards semantic classification. This is since IFC classification is actually based on the standards set out in the BuildingSMART data dictionary used within the AEC (architecture, engineering and construction) industry. Unsurprisingly, the latter does not feature heritage assets but deals with modern architectural and construction components only. That said, it has been possible, using FreeCAD’s source code, to test a proposed (and local) extension of the default *ifcType* related to Python files of Arch elements (Diara et al., 2020b).

A graphical expression of this approach to classification can be seen in figure 16’s hierarchical schema which is informed by the need to embed custom entities and definitions within existing modules. These additions act as extensions of Arch modules (ifcWall, ifcWindow, and so on) and they inherit the main characteristics of generic Arch elements like walls, windows, roof (Diara et al., 2020b).

Despite adherence to IFC standards across different platforms (FOSS and proprietary), compatibility issues can and do arise. As such, the construction of parametric families using FOSS solutions – in this case FreeCAD – demands that consideration be given to the interoperability of 3D models (IFC files). For example, while semantic data interoperability issues can be solved by using external solutions or accessing the source code (Diara et al., 2020b), metric data (3D models) may not be read correctly or may be negatively impacted by render engines and serializers used by BIM software. It is important, therefore, that future research identifies and addresses any such methodological obstacles.

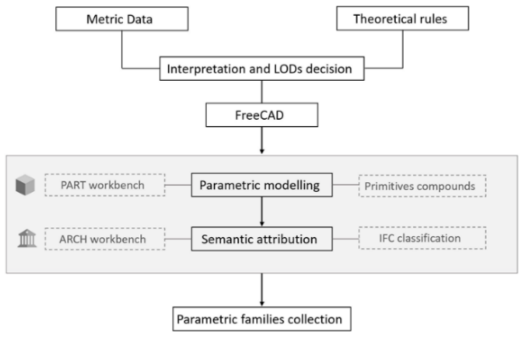


Figure 16. Design for Open-Source Platform.

6. DISCUSSION – FUTURE WORK

The potential for further automation of the process described here, and its expansion to include other epochs of building design which share standardised elements or components, is clear. In addition, the incorporation of regulations, such as the system of rates for buildings in London which determined eighteenth century window sizes, wall thicknesses and storey heights for the purposes of taxation and fire protection, may also enhance the capacity of heritage-related parametric library objects. As might the deployment of deep learning in relation to the information contained within historic building catalogues (Prizeman 2016; 2019, Pezzica et al 2019). This latter notion has previously been explored in the development of visual matching techniques for early 20th century Carnegie Library buildings (Prizeman et al 2018). Using architectural and construction histories in this way, to expand the recognition of such shared patterns, further offers the possibility of codifying, identifying and ultimately informing intelligent conservation and repair strategies including the assessment of building performance in terms of energy use and life cycle analysis (Prizeman et al 2020). Existing datasets in turn can be linked in some instances to manufacturers that are still in operation. Thus, a more nuanced reading of the existing building stock, critical to the safeguarding of not only buildings designed using classical proportions but all those that can be identified by reading patterns using machine learning, might be achieved.

ACKNOWLEDGEMENTS

Several of the ideas presented in this paper were refined in the course of discussions that took place during meetings of the 3D CRAFT Network, an interdisciplinary collective funded by the UKRI-AHRC and the Irish Research Council under the ‘UK-Ireland Collaboration in the Digital Humanities Networking Call (Grant numbers AH/V002333/1 and IRC/V002333/1).'

REFERENCES

Bosman, L. 2015 ‘Proportion and Building Material, or Theory versus Practice in the Determination of the Module.’ Architectural Histories, 3(1): 10, pp. 1-10, DOI: http://dx.doi.org/10.5334/ah.cm

Casey, Christine. Books and builders: a bibliographic approach to Irish eighteenth-century architecture. Unpublished PhD thesis. Trinity College Dublin, 1991.

Chambers, William. A Treatise on Civil Architecture. London: J. Haberkorn, 1759.

Diara F., Rinaudo F., 2018. Open source HBIM for Cultural Heritage: a project proposal. In: The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences; Volume XLII-2, 303-309. https://doi.org/10.5194/isprs-archives-XLII-2-303-2018

Diara F., Rinaudo F., 2019. From reality to parametric models of Cultural Heritage assets for HBIM. In: The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences; Volume XLII-2, pp. 413–419.

Diara F., Rinaudo F., 2020a. Building archaeology documentation and analysis through open source HBIM solutions via NURBS modelling. In: The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences; Volume XLIII-B2-2020, 1381-1388.

Diara F., Rinaudo F., 2020b. IFC classification for FOSS HBIM: open issues and a schema proposal for Cultural Heritage assets. In: Applied Sciences, Special Issue: BIM and HBIM: Principles, Applications, and Standardization/Interoperability Issues, 2020, 10(23), 8320, MDPI.

Dore, C. (2017). Procedural Historic Building Information Modelling (HBIM) For Recording and Documenting European Classical Architecture, (Doctoral Thesis, Dublin Institute of Technology). https://arrow.dit.ie/builtdoc/17/

Dore, C., Murphy, M. 2013. Semi-automatic generation of as-built BIM façade geometry from laser and image data, ITcon Vol. 19, pg. 20-46, http://www.itcon.org/2014/2

Dore, C., Murphy, M. (2017). Current State of the Art Historic Building Information Modelling. Int. Arch. Photogrammetry, Remote Sens. Spat. Inf. Sci. 42, 185–192.

Lemerle, Frédérique. Orders and Proportions from Serlio to Perrault. Proportional Systems in the History of Architecture, Mar 2011, Leiden, Netherlands. ffhalshs-01912730f

Logothetis, S., Stylianidis, E., 2016. BIM Open Source Software (OSS) for the documentation of Cultural Heritage. In: Virtual Archaeology Review, 7(15): 28-35

Loth, C., 2014. Can we trust Palladio? Antoine Desgodetz details Palladio’s inaccuracies, June 4, https://www.classicist.org/articles/can-we-trust-palladio-antoine-desgodetz-details-palladios-inaccuracies/.
Accessed 05 June 2021.

Morgan, M. H., 1960. Vitruvius: The Ten Books of Architecture. New York: Dover.

Murphy, M, McGovern, E & Pavia, S 2013, 'Historic Building Information Modelling – Adding intelligence to laser and image-based surveys of European classical architecture', ISPRS Journal of Photogrammetry & Remote Sensing, vol. 76, pp. 89-102.

Pezzica, C., Schroeter, J., Prizeman, O., Jones, C., Rosin, P. 2019. Between Images and built form: automating the recognition of standardised building components using deep learning. ISPRS Annals, Vol. IV-2 123-132.
Prizeman, O. 2016. HBIM and matching techniques: considerations for late 19th and early 20th century buildings. J. of Architectural Conservation 21(30):145-159.

Prizeman, O., Jones, C., Parisi, M, Pezzica, C. 2018. How Can Century-old Architectural Hierarchies for the Design of Public Libraries be Re-interpreted and Re-used?. J. of Cultural Heritage Management and Sustainable Development 8(4): 481-494.

Prizeman, O. 2019. Asserting Adequacy: The Crescendo of Voices to Determine Daylight Provision for the Modern World. In: Manfredi, C. ed. *Addressing the Climate in Modern Age's Construction History*. Springer AG: 171-190.

Prizeman, O., Pezzica, C., Taher, A., Boughanmi, M. 2020. Networking Historic Environmental Standards to Address Modern Challenges for Sustainable Conservation in HBIM. Applied Sciences 10(4): 1283. (10.3390/app10041283)

Summerson, John, The Classical Language of Architecture, 1980 edition, Thames and Hudson World of Art series, ISBN 0500201773

Stiny, G., (1980). Introduction to shape and shape grammars. Environment and Planning B: Planning and Design 7 343-351
Vignola, Giacomo Barozzi, Regola delli cinqve ordini d'architettura (1563)

Vignola, Giacomo Barozzi, Regola delli cinqve ordini d'architettura (1563)

Wilson Jones, Mark. Principles of Roman Architecture. Yale University Press, 2003