Shapes: Seeing and Doing

with Shape Grammars

Joaquim Reis
Instituto Universitário de Lisboa (ISCTE-IUL)
ISTAR-Iscte
Lisboa, Portugal
joaquim.reis@iscte-iul.pt

Abstract — **This paper describes the visual interface of a configurable and extensible system to support generic work with shape grammars. Shape grammars allow the implementation of computational mechanisms to analyze and synthesize designs of visual languages and have been used to represent the knowledge behind the creative work of architects, designers and artists. This kind of grammars is inherently visual. The system described, a kind of universal machine for shape grammars, allows users to build their own shape grammars and experiment with them. It has been the focus of our past work, it mixes technological and artistic aspects and it has a specific computational architecture which includes a symbolic and a visual interface. The latter one is the subject of this paper.**

Keywords –Shape Grammars; Rule-Based Systems; Creativity.

I. SHAPE GRAMMARS, AN INTRODUCTION

We start by introducing the theme, and saying what shape grammars are, in the present section. A brief summary of the research is presented in section II, which is followed by a short description of our past work, focused in the GSG (Generic Shape Grammars) system, in section III. The following three sections describe components of the visual interface of GSG, related namely shapes, rules, and grammars. We finish with a concluding section which also addresses future work.

Shape grammars were introduced by George Stiny and James Gips in the 1970s [1] and have been a topic of research since then. The focus this research is, in a few words, in representing and applying knowledge about languages of design basically through the use of concepts from formal grammars and rule-based/production systems [2] [3], essentially to give support to creative activities of designers, architects and artists.

Basically, a shape grammar is composed of (1) a set of basic shapes (the shape alphabet), (2) a set of rules, and (3) a special shape, the initial shape, used to trigger rule application.

The mechanics of rule application and shape generation is as follows. In a rule, A→B, the left side, or antecedent, A, and the right side, or consequent, B, are shapes. A rule, when applied, substitutes the shape of the right side of the rule for the shape of the left side. Applicable rules may recursively be applied to a shape, until there are no more applicable rules or some termination condition holds. A shape computation, or shape derivation, is a sequence of shapes in which each shape, except for the initial shape, is generated from the previous by the application of a rule of the shape grammar. A rule A→B is applicable to a shape, or design, or composition, C, if there is a similarity geometric transformation T which, when applied to shape A makes A a part of C, *i.e.*, a transformation T such that $T(A) \leq C$, where $\leq$ denotes the sub-shape relation. Application of the rule results in a new design, C', that is computed subtracting from C the result of applying the transformation T to A, and then adding to C the result of applying T to B, *i.e.*, the resultant design will be $C' = (C - T(A)) + T(B)$, where + and – denote the shape sum and shape difference (or subtraction) operations. Details of the + and – operations, as well as the $\leq$ relation, which are part of the so-called algebras of design [4], are outside of the scope of this paper, but an introduction and simple examples can be found in [5] (in section 2). It remains to say that, using operations on shapes from a special kind of algebras called algebras of *maximal shapes*, the computational mechanism used to match the left side of a rule with parts of the composition can be made to detect and accommodate embedded *emergent shapes*, *i.e.*, shapes that were *not explicitly included* there, and this feature is, from an artistic perspective, very much appreciated.

This all seems like Artificial Intelligence knowledge representation rules applied to visual (*i.e.*, with shapes) grammars to support some kind of creativity. In fact, shape grammars are related to design languages as phrase grammars [6] are related to symbolic/textual languages. Both can be considered production systems, where replacement rules can recursively, and incrementally, generate phrases of a language. But note that although shape grammars may exhibit (shape) emergence, a feature that production systems typically don't have.

II. A SUMMARY OF SHAPE GRAMMAR RESEARCH

In very brief words, the research area of shape grammars has been focused in conceptual and theoretical aspects, as in [7] [4] [8], in analysis, *i.e.*, the development of specific shape grammars of languages of design extracted from corpuses of designs in architecture, product design or painting, as in [9] [10] [11], and in synthesis, *i.e.*, building specific shape grammars to define original languages of designs, as in [12] [13] [14]. Other research threads include the development of algorithms for shape manipulation and rule matching and application processes, as in [15] [16] [17], and appropriate interfaces and generic and reusable shape grammar interpreters, as in [18] [19] [20], including didactical purposes.

More recent research consists mainly on refinements on previous approaches as well as a diversification of the application, see [21] [22] [23], for instance, and see also the review in [24].

III. Previous Work Related to GSG

Our exploratory work preceding the GSG system can be seen in [25] [26] [27], but the central ideas were described in [28] [29] [5]. As is referred in [28] [29], to support the creative activity of different kinds of users (students, designers, architects, artists) in interacting with the generic reusable shape grammar interpreter, which is the underlying core of the system, the interface, in particular the visual interface, is an important component of the GSG computational architecture, shown in Figure 1.

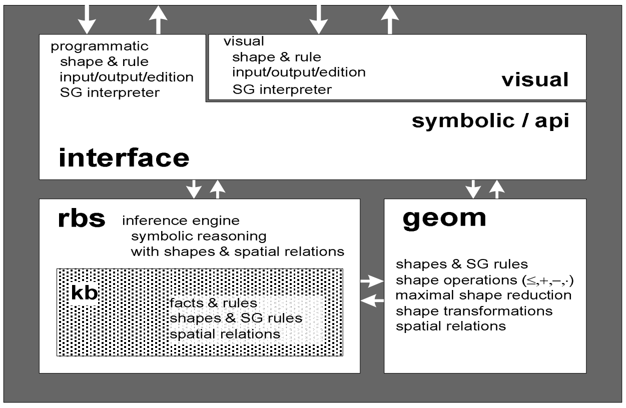


Figure 1. The GSG computational architecture.

Here we can see that the visual interface is a part of the interface layer of the system, together with the symbolic/API (programmatic) interface. A third kind of interface, the textual/file interface, not shown, is also available. A notable point in GSG is that all shape grammar objects, *i.e.*, shapes, rules, and grammars, that come to existence in the system environment may have an independent representation in three formats: the symbolic (through programmatic objects), the visual (through graphical windows) and the textual (with an appropriate text/file external representation) format.

In subsequent work about the usability of interfaces of shape grammar computational systems, see [30] [31], we have devised a set of requirements that can be used either to evaluate interfaces of existing shape grammar systems, or as a set of good rules to follow in the implementation of new ones. The requirements can be slightly different as users are either students/beginners in the field of shape grammars, or architect, designer or artist specialists, or have further additional expertise in programming, but, for simplicity, let us think we assume the student/beginner perspective and have didactical purposes. The requirements include the following minimal set of features, or abilities, of the visual interface:

1. Creation of grammar shapes;
2. Creation of grammar rules;
3. Grammar rule application;
4. Manipulation (*e.g.*, edition) of obtained shapes;
5. Alterations (*e.g.*, edition) to grammar shapes and rules.

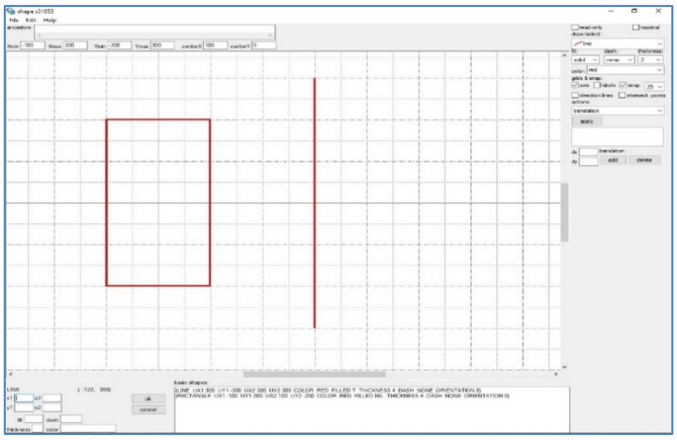
All these tasks can be performed through either the symbolic/API, or the visual interface. In the following three sections we show how they can be performed using the visual GSG interface.

IV. Building and Editing Shapes

Building and editing shapes through the visual interface of GSG is done using the shape editor/viewer. This program component can be invoked either through the main GSG grammar system, or it can function independently. Each individual shape can be built and manipulated using an individual shape editor/viewer document window. As an example, this is shown in Figure 2, where a shape containing two basic shapes, a rectangle and a line, viewed in the graphical area of the window, is being shown for edition.

At the bottom center of the shape document window a list panel lists the basic shapes present in the shape, a rectangle and a line, in the case, in textual format (the same format of the textual/file interface). At the bottom left, text slots may show data related to a basic shape when one is selected. These can also be used to create new basic shapes textually, as these slots are, in fact, text input panes. Selection of basic shapes can be made either graphically, by pointing with the mouse pointer on the shape graphical area, or by pointing on the element of the list panel associated with the basic shape.

As seen in Figure 2 and, in detail, in Figure 3 a), the top right side of the shape document window has a series of display and control elements that allow the creation of new basic shapes, with options for the associated properties, like "color", "fill", "dash", "thickness", as well as changing, transforming, and deleting existing ones.

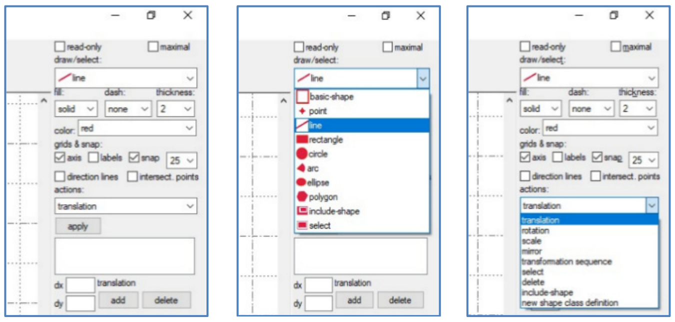


(LINE :UX1 300 :UY1 -300 :UX2 300 :UY2 300 :COLOR :RED :FILLED T :THICKNESS 4 :DASH :NONE :ORIENTATION 0)
(RECTANGLE :UX1 -100 :UY1 200 :UX2 100 :UY2 -200 :COLOR :RED :FILLED NIL :THICKNESS 4 :DASH :NONE :ORIENTATION 0)

Figure 2. A shape in the GSG shape editor/viewer.

Different two-dimensional basic shapes can be created, as shown in Figure 3 b), where the "draw/select" option pane is open. Besides points, lines (straight line segments), rectangles, circles and other plane figures, the choices include the possibility of user defined and pre-built/loaded from an image file basic shapes (the "basic-shape" and "include-shape" options, respectively). At the present stage of GSG, for the purpose of accommodating shape emergency ability, the most important basic shapes are limited to lines and points.

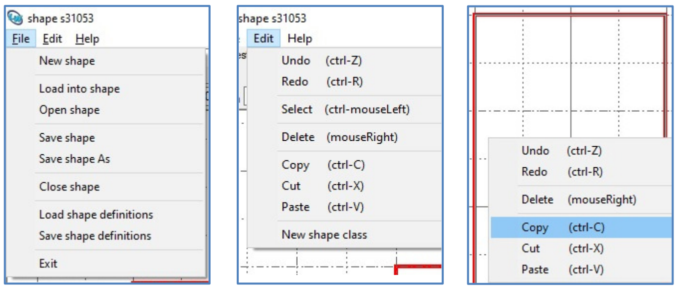
Options for controlling certain features of the graphical area, as "snap" and visibility of "axis" grids and "labels", "direction lines" and "intersection points", are also available, as seen on Figure 3 a). Actions like geometric transformation (translation, rotation, uniform scaling and sequences of these) can be defined for the shape and applied to a selected basic shape, see the action option pane in Figure 3 c). At the top, checkboxes for the options to make the shape "read-only" (immutable) and to use the "maximal" algebra operations are also available.



a) b) c)

Figure 3. a) Controls, b) Selectors, c) Actions.

In the menu bar, the "File" menu option, as seen in Figure 4 a), has different options for opening and saving shapes from/to files, including loading other shapes into the present shape, as well as loading and saving certain shape related definitions, like user defined shapes and geometric transformations. The "Edit" option, see Figure 4 b), displays editing options for any selected basic shape, some of which are duplicated in the context menu that can be made to appear on a selected basic shape, as depicted in Figure 4 c).



a) b) c)

Figure 4. a) File menu, b) Edit menu, c) Context menu.

At the top of the shape editor/viewer window, a text slot may display a record of the "ancestors" of the shape, *i.e.*, the shapes from which the present shape has been derived (if there are any), and others show the x and y coordinate values of the extremes and center of the shape at its the present state.

V. Building and Editing Rules

Building and editing rules is done using the rule editor/viewer, a program component that can be invoked either through the main GSG grammar system, or can function independently. Each individual rule can be built and manipulated using an individual rule editor/viewer document window. A rule has the left, or pre-condition, side and the right, or consequent, side, and both are shapes and are displayed in the respective graphical window area. See Figure 5, where an example rule is being shown for edition, with a rectangle in the left side and, in the right side, an equal rectangle and a circle in a certain position related to it. This rule will have one possible application instance to each and any rectangle in the composition that can be found similar to the one in its left side. Similar means that it can be made equal, or, as we say, can match, when transformed through a geometric similarity transform (*i.e.*, a combination of translation and/or rotation and/or uniform scaling). And if this rule, being applicable, is applied, the application results in the circle, transformed with the same transform combination used to match the left side, being added to the composition.

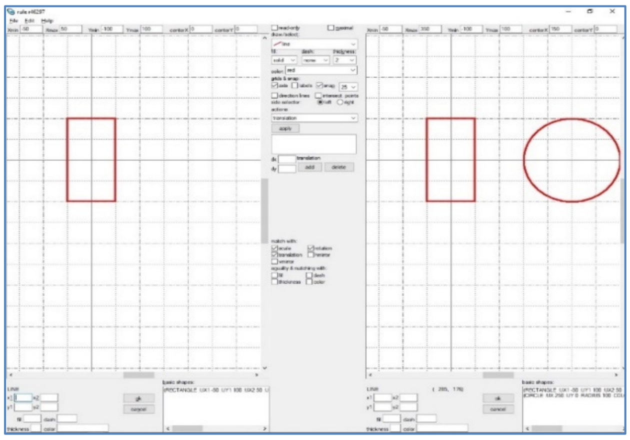


Figure 5. A rule in the GSG rule editor/viewer.

Most of the window display and control elements for each side of a rule in the rule editor/viewer window are similar to the ones in the shape editor/viewer window. The menu and the controls in the central area are shared by each side of the rule, and certain options and actions associated to them will apply either to the left, or to the right side depending on the selection of one of the two radio buttons, labelled "left" and "right".

Finally, in the central area, there is a set of checkboxes to allow for the control of the matching process. They control if scaling, rotation and translation (horizontal and vertical mirroring checkboxes are present but not used at this stage) are allowed in the matching process, and also if matching takes into account equality of the values of "fill", "dash", "thickness" and "color" properties between the shape of the left side and the composition.

VI. Building, Editing and Executing Grammars

Building and manipulating grammars, controlling the execution of instances of shape and rule editor/viewer as well as executing grammars through rule application to shapes is done using the main GSG grammar system, see Figure 6.

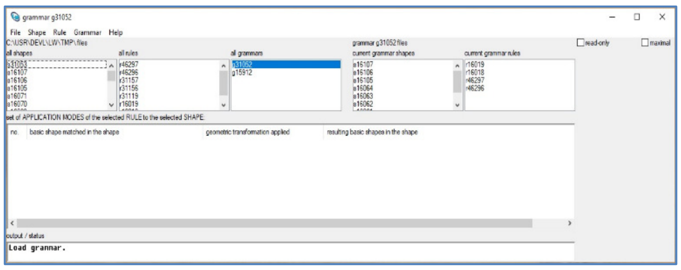
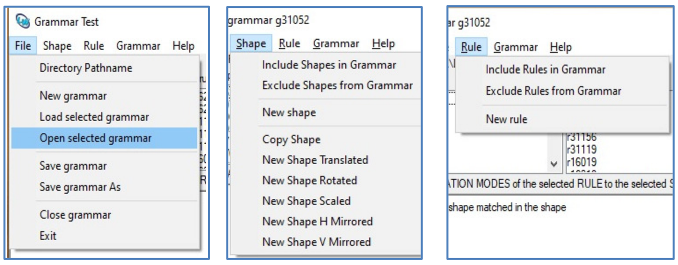

grammar g31052
File Shape Rule Grammar Help
C:\USR\DEVL\LW\TMP\files
all shapes
all rules
all grammars
grammar g31052 files
current grammar shapes
current grammar rules
read-only
maximal
set of APPLICATION MODES of the selected RULE to the selected SHAPE
no.
basic shape matched in the shape
geometric transformation applied
resulting basic shapes in the shape
output / status
Load grammar.

Figure 6. The main GSG grammar system window.

In Figure 6, the first three list panels on the left list all the shape, rule and grammar file names in the working directory. A grammar was opened, the one with the name selected in the grammar list panel, and its shapes and rules were automatically loaded and are listed in the fourth and fifth list panel. This was made with the "Open selected grammar" option of the "File" menu, in which different options for creating and saving grammars are also available, see Figure 7 a).



a) b) c)

Figure 7. a) File menu, b) Shape menu, c) Rule menu.

Also, as seen in Figure 7 b) and c), both the "Shape" and the "Rule" menu include options to create new shapes and rules, which invoke the shape and the rule editor/viewer, respectively, to include and exclude shapes and rules in/from the present grammar, and also, in the case of the "Shape" menu only, some operations on whole shapes.

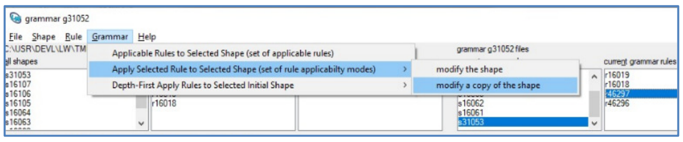

grammar g31052
File Shape Rule Grammar Help
C:\USR\DEVL\LW\TM
all shapes
Applicable Rules to Selected Shape (set of applicable rules)
Apply Selected Rule to Selected Shape (set of rule applicabilty modes)
Depth-First Apply Rules to Selected Initial Shape
modify the shape
modify a copy of the shape
grammar g31052 files
current grammar rules
s31053
s16107
s16106
s16105
s16064
s16063
r16018
s16062
s16061
s31053
r16019
r16018
r46297
r46296

Figure 8. Grammar menu.

The last, and very important, is the "Grammar" menu. This has an option to find which rules of the present grammar are, at any moment, applicable to a selected shape, taken as the initial shape, an option to apply to a selected shape a selected (applicable) rule, and an option to automatically apply rules in a depth first strategy and viewing the resulting shapes (other different strategies are being developed).

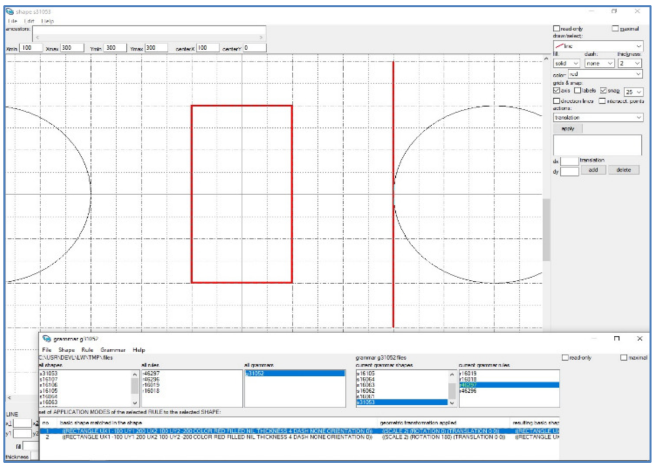


Figure 9. Application modes of selected rule to selected shape.

For reasons of space we will only show an example of the second option mentioned above. In Figure 8 we are about to apply a selected rule, the same in Figure 5, to a selected shape, the same in Figure 2, generating a copy of the original shape. The application modes, or instances, of the rule to the shape are just two, in this example, *i.e.*, the rule has two possible matching instances, with 0 and 180 rotation degrees and the appropriate translation and scaling transforms. This is shown in Figure 9, both textually, in the list panel at the bottom part of the main GSG grammar system window, and graphically, in the original shape (which will be automatically open) by the two phantom circle shapes at the right and left of the rectangle.

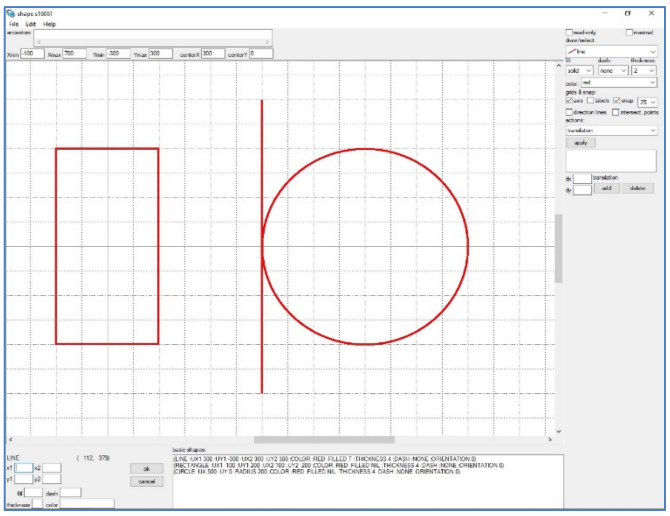


Figure 10. Shape resulting from rule application.

If we take the first choice mentioned (the one associated to the 0 rotation degrees), by double clicking on the corresponding option in the list panel, we have, as a result, the new automatically generated shape in Figure 10, shown in a newly automatically opened shape editor/viewer window.

VII. Conclusions and Future Work

In this paper we described the visual interface of GSG, a generic shape grammar system that allows users to build their own shape grammars, experiment with them, and test and refine them. The options taken in the implementation of in the interface of this program were guided by a set of requirements, found necessary, for this kind of systems, in past work we made. The GSG visual interface was built using the CAPI graphics component of the LispWorks® IDE system.

In the future, we plan to refine and expand the system, including working on components at its core, in particular related to the implementation of maximal algebras and ability to recognize shape emergence. But this is, in fact an already work in process.

Acknowledgment

This work was undertaken at ISTAR-Information Sciences and Technologies and Architecture Research Center from ISCTE-Instituto Universitário de Lisboa (University Institute of Lisbon), Portugal, and it was partially funded by the Portuguese Foundation for Science and Technology (Project "FCT UIDB/04466/2020").

References

[1] G. Stiny e J. Gips, “Shape Grammars and the Generative Specification of Painting and Sculpture,” *Information Processing,* pp. 71, 1460-1465, 1972.

[2] G. Stiny, “Introduction to Shape and Shape Grammars,” *Environment and Planning B,* pp. 7(3), 343-351, 1980.

[3] G. Stiny, Shape: Talking about Seeing and Doing, Cambridge, Massachusetts, USA: MIT Press, 2006.

[4] G. Stiny, “The algebras of design,” *Research in Engineering Design,* pp. 2(3) 171-181, 1991.

[5] J. Reis, “Crossing Lines in GSG,” em *ISDOC 2014, Proceedings of the International Conference on Information Systems and Design of Communication, pp. 105-112.*, Lisboa, Portugal, 2014.

[6] N. Chomsky, Syntactic Structures., London: Mouton and Co., 1957.

[7] G. Stiny, “What is a Design?,” *Environment and Planning B,* pp. 17, 97–103, 1990.

[8] T. Knight, “Computing with Emergence,” *, Environment and Planning B,* pp. 30, 125-155, 2003.

[9] G. Stiny e W. J. Mitchell, “The Palladian Grammar,” *Environment and Planning B,* pp. 5, 5-18, 1978.

[10] G. Koning e J. Eisenberg, “The Language of the Prairie: Frank Lloyd Wright’s Prairie Houses,” *Environment and Planning B,* pp. 8, 295-323, 1981.

[11] J. P. Duarte, “Towards the Mass Customization of Housing: The Grammar of Siza's Houses at Malagueira,” *Environment and Planning B,* pp. 32, 347-380, 2005.

[12] G. Stiny, “Kindergarten Grammars: Designing with Froebel’s Building Gifts,” *Environment and Planning B,* pp. 7, 409-462, 1980.

[13] J. Heisserman, “Generative Geometric Design,” *, IEEE Computer Graphics and Applications,* pp. 14, 37-45, 1994.

[14] M. Agarwal e J. Cagan, “A Blend of Different Tastes: The Language of Coffeemakers,” *Environment and Planning B,* pp. 25, 205-226, 1998.

[15] S. C. Chase, “Shapes and Shape Grammars: From Mathematical Model to Computer Implementation,” *Environment and Planning B,* pp. 16, 215-242, 1989.

[16] R. Krishnamurti, “The Maximal Representation of a Shape,” *Environment and Planning B,* pp. 19, 267-288, 1992.

[17] T. Trescak, M. Esteva e I. Rodriguez, “A shape grammar interpreter for rectilinear forms,” *Computer-Aided Design,* vol. 44 (7), pp. 657-670, 2012.

[18] M. Tapia, “A Visual Implementation of a Shape Grammar System,” *Environment and planning B,* pp. 26, 59–73., 1999.

[19] M. Agarwal e J. Cagan, “On the Use of Shape Grammars as Expert Systems for Geometry-Based Engineering Design,” *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing,* pp. 14, 431-439, 2000.

[20] S. C. Chase, “A model for User Interaction in Grammar-Based Design Systems,” *Automation in Construction,* pp. 11, 161–172, 2002.

[21] T. Grasl e A. Economou, “GRAPE: using graph grammars to implement shape grammars,” em *, SimAUD '11: Proceedings of the 2011 Symposium on Simulation for Architecture and Urban Design, Pages 21–28, April 3 - 7, 2011*, Boston, Massachusetts, 2011.

[22] J. Tching, A. Paio e J. Reis, “A Shape Grammar for Self-Built Housing,” em *Proceedings of the SIGraDi 2012, pp. 486-490.*, Fortaleza, Brasil, 2012.

[23] J. Tching, J. Reis e A. Paio, “Shape Grammars for Creative Decisions in the Architectural Project,” em *CISTI 2013, Proceedings of the 8th CISTI, Vol. I, pp. 389-394.*, Lisboa, Portugal, 2013.

[24] S. Eloy, P. Pauwels e A. Economou, “Advances in Implemented Shape Grammars: Solutions and Applications,” *AI EDAM, Volume 32, Special Issue 2 (May 2018): pp. 131 - 137.,* vol. 32, pp. 131-137, 2018.

[25] J. Reis, “Agents with Style – Multi-Agent Visual Composition with Shape Grammars,” em *Proceedings of the Third Joint Workshop on Computational Creativity, Aug. 2006*, Riva del Garda, Italy, 2006.

[26] J. Reis, “Using Rules for Creativity in Visual Composition,” em *Proceedings of the SIGDOC 2008, pp 207-214.*, Lisbon, Portugal, 2008.

[27] J. Reis, “A Rule Language to Express Visual Pattern Generation,” em *Proceedings of the SIGDOC 2008, 26th ACM Int. Conf. on Design in Communication, September 2008*, Lisbon, Portugal, 2008.

[28] J. Reis, “GSG, A Tool for Knowledge-Based Visual Creativity,” em *CISTI 2013, Proceedings of the 8th CISTI, Vol. I, pp. 358-363.*, Lisboa, Portugal, 2013.

[29] J. Reis, “A Shell Tool for Visual Creativity Support,” em *ISDOC 2013, Proceedings of the International Conference on Information Systems and Design of Communication, pp. 56-63.*, Lisboa, Portugal, 2013.

[30] J. Tching, J. Reis e A. Paio, “A Cognitive Walkthrough towards an Interface Model for Shape Grammar Implementations,” *Computer Science and Information Technology,* vol. 4(3), pp. 92-119, 2016.

[31] J. Tching, J. Reis e A. Paio, “IM-sgi – an Interface Model for Shape Grammar Implementations,” *AIEDAM, Artificial Intelligence for Engineering Design, Analysis and Manufacturing,* pp. 33, Issue 1, February 2019, 24-39, 2019 .