A Discursive Grammar for Customizing Mass Housing

The case of Siza´s houses at Malagueira

José P. Duarte

Masschusetts Institute of Technology U.S.A., now at Instituto Superior Técnico, Portugal
http://www.civil.ist.utl.pt/~jduarte/malag/

Abstract. The ultimate goal of the described research is a process for mass customizing housing based on computer-aided design and production systems. The current goal is the development of an interactive system for generating solutions on the Web based on a modeling approach called discursive grammar. A discursive grammar consists of a programming grammar and a designing grammar. The programming grammar generates design briefs based on user data; the designing grammar provides the rules for generating designs in a particular style, and a set of heuristics guides the generation of designs towards a solution that matches the design brief. This paper describes the designing grammar using Siza´s houses at Malagueira as a case study.

Keywords. Mass customization: housing; grammars; Siza; design automation.

Introduction

The ultimate goal of this work is the design and production of mass-customized houses. The current focus is on design aspects. The purpose of mass-customization is to provide high-quality housing at an affordable cost. Quality is defined as the satisfaction of user needs. Cost is controlled by using computer-aided manufacturing, which does not rely on exhaustive repetition. Traditionally, when a designer is faced with the design of a large development, the usual solution is to design a limited number of housetypes and then to repeat them based on market analysis. The envisaged process aims at overcoming such limitations by using computer-aided design and manufacturing processes. The idea is to give mass-produced houses some of the qualities associated with individually designed homes.

The design system includes an interactive program for generating housing solutions, and rapid prototyping and virtual reality techniques for visualizing these solutions. The user accesses the program on the Web. The program guides the user through questions that an architect would normally ask during an initial meeting, such as the family members' profile, their living habits, the rooms they want, the cost that they can afford, and so on. When the interview is over, the program generates the design brief or housing program, taking into account existing housing regulations. The user can then make changes to the initial requirements and the program will update the design brief. Once the brief is approved, the program generates a housing solution that satisfies the requirements, which the user can assess. At this stage, the user might want to change the initial requirements and proceed through another iteration of the design process. Once a solution is accepted, an order can be automatically issued to the housing factory. This order will include a detailed list of parts, and digital information to manufacture the parts using computer-aided manufacturing techniques. At the end of the

manufacturing process, these parts are transported to the site and assembled.

The research reported in this paper aims at the development of a mathematical model for the interactive program just described. This model needs to overcome three problems that correspond to the three computer-based methods identified by Radford & Gero (1988). First, it needs to provide a way of translating client data into design requirements, and to verify whether a design satisfies these requirements -- the simulation problem. In simulation, the computer manipulates a mathematical model that describes the design to evaluate the performance of a given design configuration against the design requirements. Second, it has to codify the rules of formal composition to design a house in a given style -- the generation problem. In generation, the computer is used to produce design configurations according to a set of rules. And third, it needs a mechanism to translate the design requirements into a housing solution -- the optimization problem. In optimization, the computer is used to generate design configurations that meet a performance goal. The number of solutions that satisfy multiple requirements is potentially very large. Thus, an important part of the model is a computational strategy capable of searching a potentially large design space, and providing insight into function-form relations for multi-criteria housing design.

The proposed model is illustrated with a case study that includes specific programming and designing grammars. This paper is mainly concerned with the designing grammar. Section 2 briefly explains the concepts of discursive, shape and description grammars. Section 3 briefly describes the programming grammar. Section 4 describes the designing grammar. Section 5 refers to the computer implementation, and Section 6 closes the paper.

Discursive grammar

The simulation and generation problems mentioned above can be solved with the use of a description grammar (Stiny, 1993) and a shape grammar (Stiny and Gips, 1972), respectively. A grammar consists of a set of substitution rules that apply recursively to an initial assertion to produce a final statement. In description grammars, the assertions are symbolic descriptions, whereas in shape grammars, they consist of shape descriptions. In addition, description grammars deal with semantics, and shape grammars address form. The third problem is solved with a set of heuristics. Heuristics are used to choose a rule for application at each step of the design generation or to constraint choice to a small number of rules. Other heuristics assess the designs that would result from the application of each of the available rules, and then choose the one that takes the evolving design closer to the goal description in the design brief. This process is deterministic. At a micro-scale, a specific design context will lead to the application of a specific rule; at a macro-scale, a given context will lead to a given housing solution. I call this mathematical model a discursive grammar because it allows the generation of formally and semantically correct designs. Each house is like a piece of speech in the language that is appropriate for the context.

From the technical viewpoint, a discursive grammar consists of a shape grammar, a description grammar, and a set of heuristics. From the operative viewpoint, a discursive grammar consists of a programming grammar and a designing grammar. The programming grammar processes user data and site data to generate the housing program (design brief). The designing grammar uses the housing program to generate a housing solution (design). The programming grammar has a description part and an empty shape part, whereas the designing grammar has both a

description and a shape part. In theory, different programming grammars can be combined with different designing grammars to form various discursive grammars. In practice, one has to ensure that the contexts of both grammars match, so that one can find designs in the language defined by the shape grammar that match descriptions in the universe defined by the description grammar. In the specific discursive grammar developed, the programming grammar adapts the rules of the Portuguese housing program and evaluation systems (PAHP) (Pedro 1999, 2000), and the designing grammar (Duarte 1999) encodes the rules developed by the architect Álvaro Siza at Malagueira, a 1,200 housing development still in construction today. The proposed discursive grammar is, thus, called the PAHPA-Malagueira grammar, after the Portuguese acronym.

The PAHPA Programming Grammar

The programming grammar is concerned with the generation of a symbolic description of the desired house (the housing program) from user input data by manipulating only symbolic descriptions. A crucial issue in the development of symbolic descriptions is that of fixing the contents of the description, that is, which categories to include. The PAHPA grammar used a hierarchy of qualities based on Pedro (2000) performance criteria organized into a decision tree. The description includes a variable description (a) and a fixed description (b). The features of the variable description are organized into three main groups. The first group includes contextual features—plot size, urban context, and solar orientation—typological features—degree of customization, users' profile, number of bedrooms, and quality level—and morphological features –housetype, number of floors, and existence of balconies. This group is called constraints because the values of its elemental features are specified by the user and cannot be changed by the programmer. The exception is the quality level, which is updated by the programmer after the user changes other features. The second group includes function—spatiality: dwelling capacity, room capacity, room articulation, spaciousness; and topology—and aesthetics—whose single feature is proportion regarded by Siza as important. The user can assign weights to these qualities to express their relative importance, and to determine the overall quality. The third group includes only the construction cost. Constraints, qualities, and cost frame the problem as "design a house with the specified qualities, within the given constraints, without exceeding the cost." The features of the fixed description have fixed values that the user cannot change. Due to space constraints, it is not possible to show the full contents and the set of rules in the programming grammar, but more information can be found in Duarte (2002).

The Malagueira Designing Grammar

The designing grammar is concerned with the generation of a housing solution that matches the housing program by manipulating both symbolic and shape descriptions. To make it easier for the reader to understand the formal properties of the grammar, we show a very simplified set of shape rules and the partial generation of a layout using such rules in Figure 1.

In brief, the generation of a Malagueira design is based on the manipulation of rectangles using rules for dissecting, connecting, and extending rectangles, as well as rules for assigning and changing the functions associated with them. The generation of basic layouts with these rules comprises two steps. In the first step, the lot is first divided into the four functional zones—patio, living, service, and sleeping—thereby obtaining a basic pattern, and then a staircase is added thereby defining a stair pattern and the house-

type. In the second step, these zones are divided into rooms to obtain the layout. The labels "fn" denote the functions of the rooms that the rectangles represent. The dot $\sum$ is a label that identifies the last line placed and indicates on which side the next dissection may occur: on both sides (Rule A) or only one side (Rule B). In rules A and B, dissections are perpendicular to the bigger side of the rectangle, whereas in Rule C it is perpendicular to the smallest one. Rule D deletes the label $\sum$, preventing further dissections. Rule E concatenates two adjacent rectangles to form a larger room. Rule F, extends a room at the expense of an adjacent one. Rule G assigns a function to a room. Finally, Rule H permutes the function of two adjacent rooms.

The actual designing rules are further more complicated as they include a shape part and a description part. To make it possible for the reader to grasp the complexity of the designing grammar, a more detailed rule but still simplified, is shown in Figure 2. The original shape part uses various shape algebras to represent several viewpoints—plans, elevations, axonometrics, but we only show three of such viewpoints—envelope and space (plan and axonometric). The descrip-

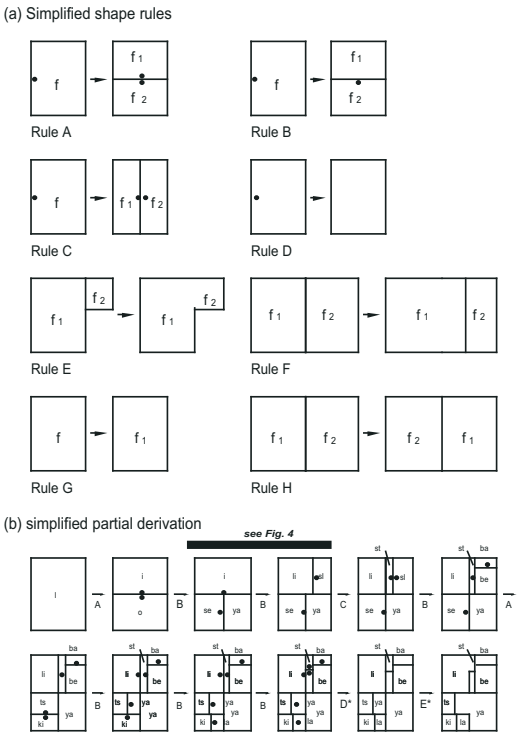


Figure 1. Simplified Malagueira shape rules (a) and partial derivation of an existing layout (b). There are rules for dissecting (A, B, and C), connecting (E), and extending (F) rectangles. The remaining rules are for deleting a marker (D), assigning a function (G), and permuting functions (H). Legend: l – lot, i – inside zone, o – outside zone, li – living zone, sl – sleeping zone, se – service zone, ya – yard zone, be – bedroom, ba – bathroom, ki – kitchen, ts – transitional space, la - laundry, pa – pantry, ci - circulation, st -stairs. The asterisk means that the same rule was applied several times.

tion part includes a program description and a design description. The design description includes the same features as the program description, but they are identified with the Greek letter d instead of a. Thus, for each program feature an there is a design feature dn. The derivation of a design in the grammar is controlled by the plan and the axonometric is used for visualization. It is possible to group the rules according to the type of operations that they perform in the derivation: start, locate functional zones, define circulation scheme, divide zones into rooms, introduce details, and introduce openings.

Figure 3 shows some of the rules used for locating functional zones in a very simplified manner as only the space view in the shape part is depicted. Rule 11 allocates the patio by dissecting the outside zone into patio and an unnamed zone. It locates the patio on one side or the other of the outside zone, depending on the urban context. Rule 12 allocates the zone of the remaining three zones whose allocation in the current design

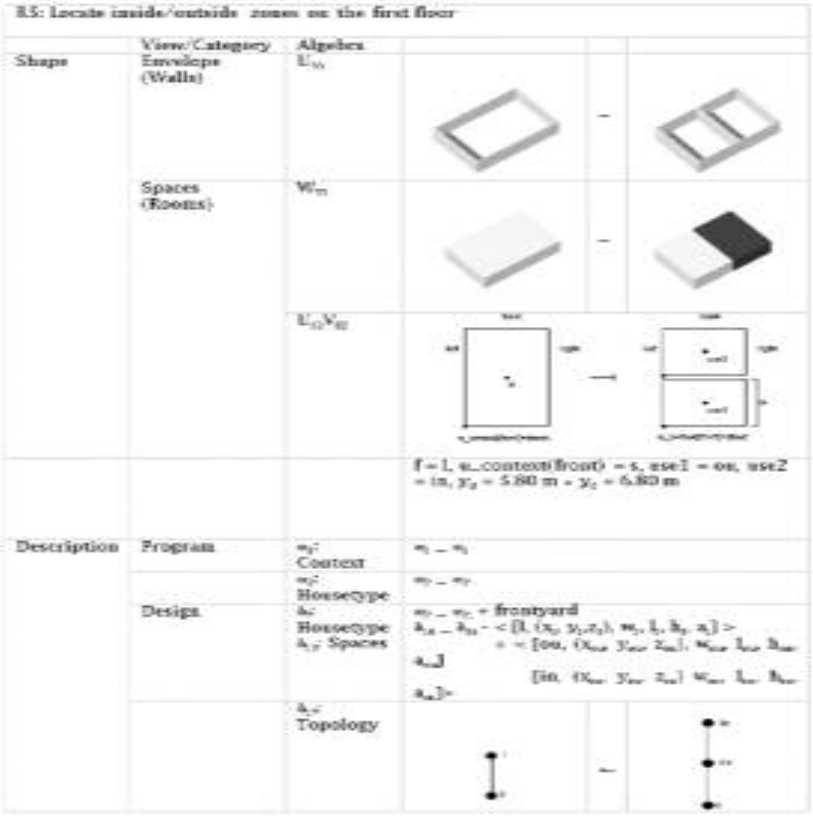


Figure 2. Simplified discursive rule combining shapes and descriptions. The shown rule includes only some of the views and some of the features in the original grammar and it depicts the case in which the outside zone is located at the front of the house, adjacent to the street—s. The feature "spaces" include [use (x,y) width length area] where use is the function associated with the space (e.g. lot—l, inside—in, or outside zone—ou); (x,y) is the insertion point, and w, l, h, and a are the dimensional features.

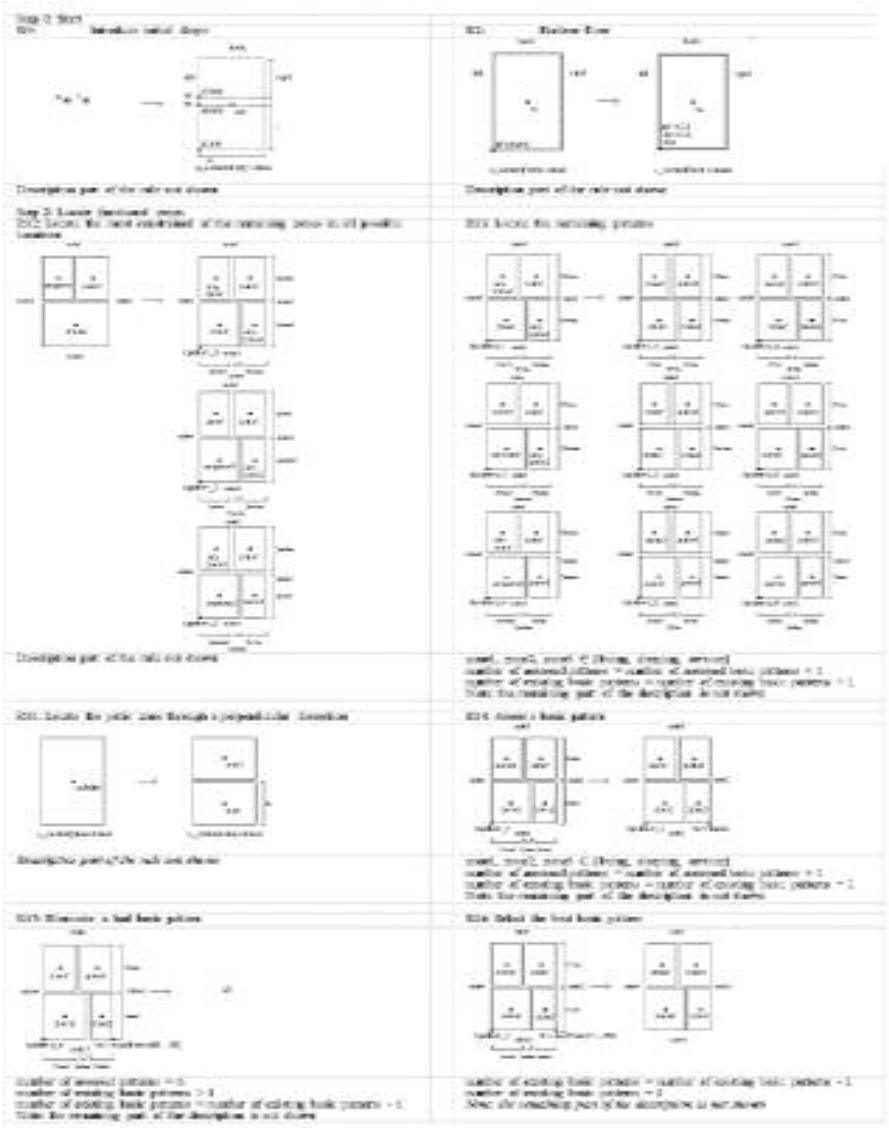


Figure 3. Simplified discursive rules. The shown rules include only the rooms view in the shape part.

context will likely lead to the design solution that is the closest to the goal. It locates the zone in all possible situations so that the resulting candidate solutions can be assessed afterwards. The heuristic for choosing such a zone is the importance of a zone and is explained below. Rule 13 locates the remaining two zones in all possible situations thereby generating basic patterns. Rule 14 assesses a basic pattern to determine its fitness. The heuristic for determining the fitness of a basic pattern is explained below. Rule 15 eliminates all the worst basic patterns, one by one, until only one is left, if all the six possible patterns have already been generated. Rule 16 chooses the best basic pattern to resume the derivation. Figure 4, shows a few moves in the generation of a design using these rules. These moves define a basic pattern and correspond to the part of the derivation highlighted in Figure 1. . The description part is not shown, but it should help to clarify in greater depth the relation between shapes and descriptions.

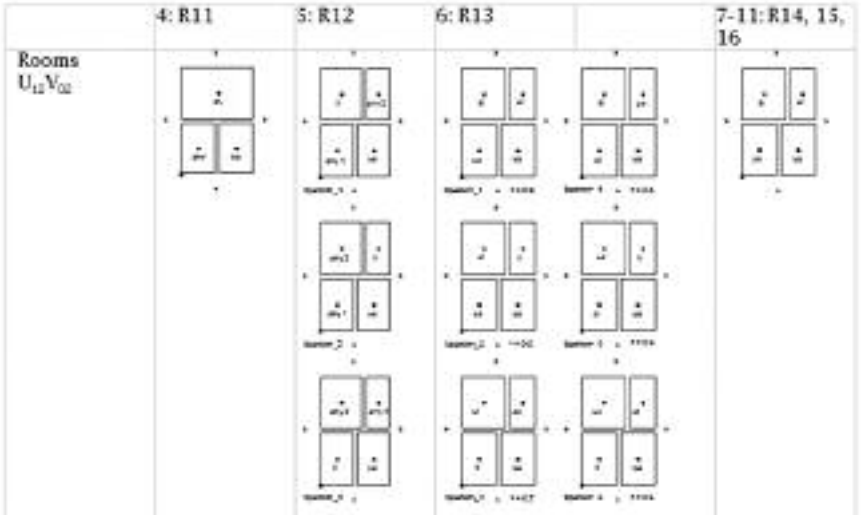


Figure 4. A few moves in the derivation of a house within the PAHPA-Malagueira grammar. (Please compare with Fig. 1). The description part is not shown.

Heuristics for choosing the zone to allocate

The importance of a zone Izone is the weighted average of its spatiality importance Ispatiality(zone) and its topology importance Itopology(zone) as expressed in equation (1), where wspatiality is the weight assigned to spatiality in the housing program, and wtopology is the weight assigned to topology in the housing program.

This equation implies an indirect comparison

between spatiality importance and topology importance. To make such a comparison possible, it is necessary to translate them into a similar scale. This is achieved by comparing the importance of a zone from each viewpoint relative to the importance of the zone with bigger demands from the same viewpoint. Therefore, the relative importance has a value between 0 and 1, from any of the viewpoints.

The relative importance of the zone's spatiality Ispatiality(zone) is given by equation (2), where a(spacei) is the area of a space included in the zone, wi is the average of the weights assigned to this spaces' spatiality features, a(spacej) is the area of the space included in the zone with bigger spatiality demands, and wj is the average of the weights assigned to this spaces' spatiality features.

$$I_{spatiality(zone)} = \frac{\Sigma\,(a(space_i))\cdot w_i \,/\, \Sigma\, w_i}{\Sigma\,(a(space_j))\cdot w_j \,/\, \Sigma\, w_j}$$

Similarly, the relative importance of the zone's topology Itopology(zone) is given by equation (3), where ri (spacez, spacen) is a relation that involves one of the zone's spaces and any other space included in the housing program, wi is the weight assigned to such a relation in the housing program, rj (spacezmax, spacen) is a relation that involves the space in the zone with higher number of topological relations involving its spaces, and wj is the weight assigned to such a relation in the housing program.

$$I_{topology(zone)} = \frac{\Sigma\, r_i\,(space_z,\ space_n)\cdot w_i \,/\, \Sigma\, w_i}{\Sigma\, r_j\,(space_{zmax},\ space_n)\cdot w_j \,/\, \Sigma\, w_j}$$

Heuristics for determining the fitness of a basic pattern

The fitness of a basic pattern fpattern is the weighted average between its spatiality fitness and its topology fitness as expressed in equation (4), where wspatiality is the weight assigned to spatiality in the housing program, Itopology(zone) is the pattern's topology fitness, and wtopology is the weight assigned to topology.

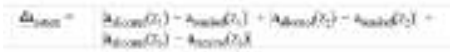
$$f_{pattern} = \frac{f_{spatiality(pattern)}\cdot w_{spatiality} + f_{topology(pattern)}\cdot w_{topology}}{w_{spatiality} + w_{topology}}$$

As for the importance of a space, to calculate the fitness of a basic pattern implies a comparison between spatiality fitness and topology fitness. This comparison requires their translation into a similar 0-1 scale by measuring the fitness of a pattern from a given viewpoint to the fitness of the best pattern from the same viewpoint.

The pattern whose zone areas are closer to their requirements in the housing program, will likely need smaller area exchange among its zones later in the derivation, and therefore is considered the best pattern from the spatiality viewpoint. The area requirements of a given zone arequired(z1) are given by equation (5), where a(spacei) is the area of each space included in the zone, wi is the average weight assigned to this space's spatiality requirements.

$$a_{required(z_1)} = \frac{\Sigma\,(a(space_i))\cdot w_i}{\Sigma\, w_i}$$

The area balance of a pattern dapattern is the sum of the area balances of its zones as shown in equation (6), where aallocated(zn) is the area allocated for zone n, and arequired(zn) is the required area for the same zone n.



The relative spatiality fitness of a pattern fspatiality(current_pattern) is, therefore, given by equation (7), where dabest_pattern is the area difference of the pattern with the smallest area difference, and dacurrent_pattern is the area difference of the current pattern.

The topology fitness of a pattern ptopology(current_pattern) is the sum of the ratio between the length of the wall separating two zones and the weighted number of close relations between them; and the ratio between the weighted number of distance relations between them, and the length of the wall as shown in equation (8), where i, j Œ {patio, living, sleeping, service, front, left, back, right}, ptopology(current_pattern) is the topology performance of the current pattern, r ck (zonei, zonej) is a close relation either between a space in zone i and space in zone j, or between a space in zone i and zone j (e.g. living room adjacent to south), rdck (zonei, zonej) is a distance relation between a space in zone i and space in zone j, or between a space in zone i and zone j (e.g. living room adjacent to south), wk is

the weight associated with such a relation, and lwall(zonei, zonej) is the length of the wall between zone i and zone j.. The unit is added to the length of the wall to prevent the problem from becoming indeterminate when the zones do not share a wall.

The relative topology fitness of a pattern, ftopology(current_pattern) is obtained by comparing its absolute topology fitness with that of the best pattern from this viewpoint as shown in equation (9).

Conclusion

This paper shows the application of a modeling approach called discursive grammar to the development of an interactive computer system for exploring housing solutions within a given style with the goal of customizing mass housing. The discursive grammar uses a programming grammar to translate user requirements into a design brief, and a designing grammar to translate the brief into a design solution using heuristics. The approach was illustrated with the development of a specific grammar called PAHPA-Malagueira, which codifies the Portuguese housing design guidelines into a programming grammar, and the style of Álvaro Siza at Malagueira into a designing grammar. A computer implementation of the PAHPA-Malagueira called MALAG is being developed as a proof of concept. Currently, MALAG generates stair patterns (the lot divided into 5 functional zones), but its completion will be sought in the future. The computer implementation will automate the process of generating designs, as it will yield a design solution from user prompt data. Yet, it still has to be shown that the best solutions can be found by heuristic search only. Despite automation, the user can use

MALAG as a design assistant, as it can fiddle with weights and requirements and use it to explore different solutions. More information of the project can be found at: http://www.civil.ist.utl.pt/~jduarte/malag/ .

Acknowledgements

I thank Terry Knight, William Mitchell, George Stiny, and Patrick Winston for their support and guidance. Also thanks to Álvaro Siza for his cooperation. Funds for this research came from Fundação para a Ciência e a Tecnologia, Portugal.

References

Duarte, J. P.: 1999, Democratized Architecture: Grammars and Computers for Siza's Mass Housing, in Proceedings of the International Conference on Enhancement of Computational Methods in Engineering and Science. Macau: Elsevier Press.

Duarte, J. P.: 2002, A Descriptive Grammar for Generating Housing Briefs on Line, in Concurrent Engineering – Research and Applications, Proceedings of the.9th ISPE International Conference on Concurrent Engineering 2002, Cranfield, United Kingdom.

Pedro, J. B.: 1999, Programa Habitacional. Habitação. Colecção Informação Técnica e Arquitectura, nº5. Lisboa: LNEC.

Pedro, J. B.: 2000, Indicadores de Qualidade Arquitectónica Habitacional. Ph.D. Thesis, Faculdade de Arquitectura da Universidade do Porto. Porto: Edições FAUP.

Radford, A. and Gero, John: 1988, Design by Optimization in Architecture, Building, and Construction. New York: Van Nostrand Reinhold.

Stiny, G. & Gips, J.: 1972. Shape Grammars and the Generative Specification of Painting and Sculpture, in C V Freiman (ed), Information Processing 71, Amsterdam: North-Holland, pp. 1460-1465.

Stiny G.: 1981, A note on the description of designs, in Environment and Planning B: Planning and Design 8, pp. 257-267.