Attribute Grammars for Controlling House Layout Customization

Wladyslaw Homenda, Krystian Kwieciński

Attribute Grammars for Controlling House Layout Customization

Władysław Homenda[1,2] and Krystian Kwieciński[3]

[1] Faculty of Economics and Informatics in Vilnius, University of Białystok
Kalvariju g. 135, LT-08221 Vilnius, Lithuania
[2] Faculty of Mathematics and Information Science, Warsaw University of Technology
ul. Koszykowa 75, 00-662 Warsaw, Poland
[3] Faculty of Architecture, Warsaw University of Technology
Koszykowa 55, 00-659 Warsaw, Poland

Abstract. This paper introduces a new framework for design automation. The discussion is focused on syntactic processing nested in a selected architecture problem of house layout customization. House layout is interpreted as multi dimensional concepts structure of domain knowledge. Syntactic structuring is based on context-free methods. We propose constructions of context-free grammars driven by concepts of multidimensional knowledge space. The formalism affords consistent representation of an approximate grammatical correctness of house layout. Furnishing grammars with attributes allows for information flow between domain's concepts supporting features disclosing and constrains verification. The approach is validated on some simple house layout and the directions for the future development of the system are outlined. The paper focuses on conceptual framework only and, as such, is intended to stimulate further research into the various implementation considerations that are prerequisite of large-scale applications.

Keywords: syntactic methods, context-free grammars, attribute grammars, design automation, house layout

1 Introduction

One reason for which design automation is of significant research interest is that it provides a basic reference for comparing different processing technologies. However, a more fundamental reason is that it focuses on the essence of intelligent information processing, the formation of abstractions. This paper provides a valuable insight into the methodologies that lead to comprehensive and interpretable results and that ensure the transparency of final findings. In one way or another there arises an issue of casting the results as information structuring conceptual entities that capture the essence of the overall data set in a compact manner. It is worth stressing that information structures not only support conversion of detailed data into more tangible information entities but, very importantly, afford a vehicle of abstraction that allows to think of structures as different conceptual structures. A context-free grammars were chosen due to its simplicity and possibility of augmenting them with the mechanism of attributes propagation. Clearly the task of information structuring is not a trivial one and it is dependent to a large extent on the application domain.

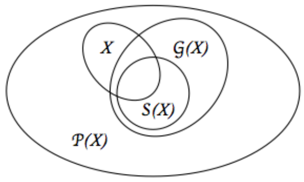

Fig. 1. Syntactic structuring and semantic analysis - an overall perspective.

Let us discuss the overall perspective on house layout design automation. Assuming that X is a set of basic design entities (living room, bedroom, kitchen etc.), we may express all possible arrangements of them as a power set (with repetitions of entities) $\mathcal{P}(X)$, as illustrated in Figure 1. We build a context-free grammar to generate arrangements of entities of X. Of course, only a small proportion of elements of $\mathcal{P}(X)$ represent arrangements, which conform to the rules of such a grammar. These represent a subset of $\mathcal{P}(X)$, referred to as $\mathcal{G}(X)$. And the grammatically correct arrangements that are meaningful are represented by $\mathcal{S}(X)$ that is a subset of $\mathcal{G}(X)$. The entities (X) themselves have a dual nature; on one hand they have grammatical meaning (parts of arrangements) and on the other hand they have semantics defined by the concepts they represent (function, size, neighborhood etc.). The essence of syntactical analysis is to discriminate whether a given arrangement belongs to $\mathcal{G}(X)$ or $\mathcal{P}(X) - \mathcal{G}(X)$. Syntactical analysis dissects the arrangements into higher level parts of a house (bay, private zone, semipublic zone etc.) and checks for conformity with the rules of grammar until the bottom-most level of individual entities is reached. A standard approach to syntactical analysis involves application of parsing techniques to check the conformity of a given arrangements with the rules of grammar. As adverted above, not all syntactic correct arrangements $\mathcal{G}(X)$ are meaningful, i.e. many of them do not conform to semantic constrains. Elimination of incorrect arrangements is accomplished with attributes, which furnish the context-free grammar. Attributes provide mechanisms of information flow in a derivation (parsing) tree of given arrangement and in this way allow to encounter semantic information in terms of supplementary rules and constrains verification. This study is motivated by fundamental ideas of syntactic methods employed to knowledge processing [1, 3] and their application in different domains, for example: intelligent man-machine communication [4, 5], music information processing [6] and human behavior [9].

The paper is structured as follows. Next subsections introduce the subject (the domain of house layout design) and the tool used (context-free and attribute grammars). In Section 2 syntactic structuring of the target objects is outlined with context-free grammars employed. Section 3 attributes supplementing context-free grammars are pursued to accomplish semantic information.

1.1 The Subject: House Layout Customization

Automatic generation of house layouts has been previously studied both for historical architectural styles [10] as well as for the designs of contemporary living architects [2]. Previous researches utilized shape grammars in order to formalize architectural rules as a series of transformations of shapes.

An architectural design system proposed by Kwieciński [8], was intended to allow customers participation in the generation of customized houses. Proposed system, which consist of three modules: Site Planner, Home Planner and Facade Creator, was designed to produce single story or two story buildings, with a two bay house layouts and a gable roof. Home Planner, which is responsible for the generation and customization of house layout, collects architectural design principles responsible for the distribution of rooms into the bays on each floor. Spatial distribution of rooms takes place in relation to the external and internal context. For this purpose the orientation to the World coordinates (which affects indoor sunlight), orientation to the entrance to the plot and the privacy gradient (which suggests placing common areas, such as the living room, closer to the entrance, while private spaces, such as bedrooms farther away) are considered in the design system. Therefore the rooms are grouped into three zones: entrance zone, semipublic zone and private zone which constrain the freedom of layout configurations. Additionally some rooms are required to be placed next to each other like kitchen and dining room for example. Both of these rooms belong to semipublic zone together with living room, which is obligatory in every layout and home office which is optional and staircase which existence depends on the number of floors. If the house is not a single story building then localization of staircase is duplicated on the second floor and the generation of the first floor determines the generation of the second floor. When a planar layout of the spaces on each floor is completed it can be further customized by the user and further developed. Figure 2 presents exemplary two story building. Left side of the figure presents detailed floor plan with introduced walls and furniture and right side of the figure presents diagrammatic layout where rooms are represented by colored and named matrices of 60 by 60 cm modules. Building is having rectangular shape with only exception of garage which sticks out of the shape. Layout is divided in the middle into two part which represents different buildings bays. Bays are having same depth and equal length which is the same as the building length.

1.2 The Tool: Syntactic Structuring

Syntactic approach is a crucial stage and a crucial problem in the wide spectrum of tasks as, for instance, pattern recognition, translation of programming languages, processing of natural languages, music processing, etc., cf. [1, 4, 6]. By syntactic approach and syntactic methods we understand grammars, automata, algorithms used in processing languages. In this paper we employ context-free grammars to controlling house layout customization.

Context-free Syntactic Description The discussion on describing and controlling house layout is based on common definitions of grammars, context-free grammars and attribute grammars. We assume that the reader is familiar with the basic notions of mathematical linguistic. Therefore, we only recall them.

Let us recall that a system $G = (V, T, P, S)$ is a grammar, where: (a) V is a finite set of *variables* (called also *nonterminals*), (b) T is a finite set of

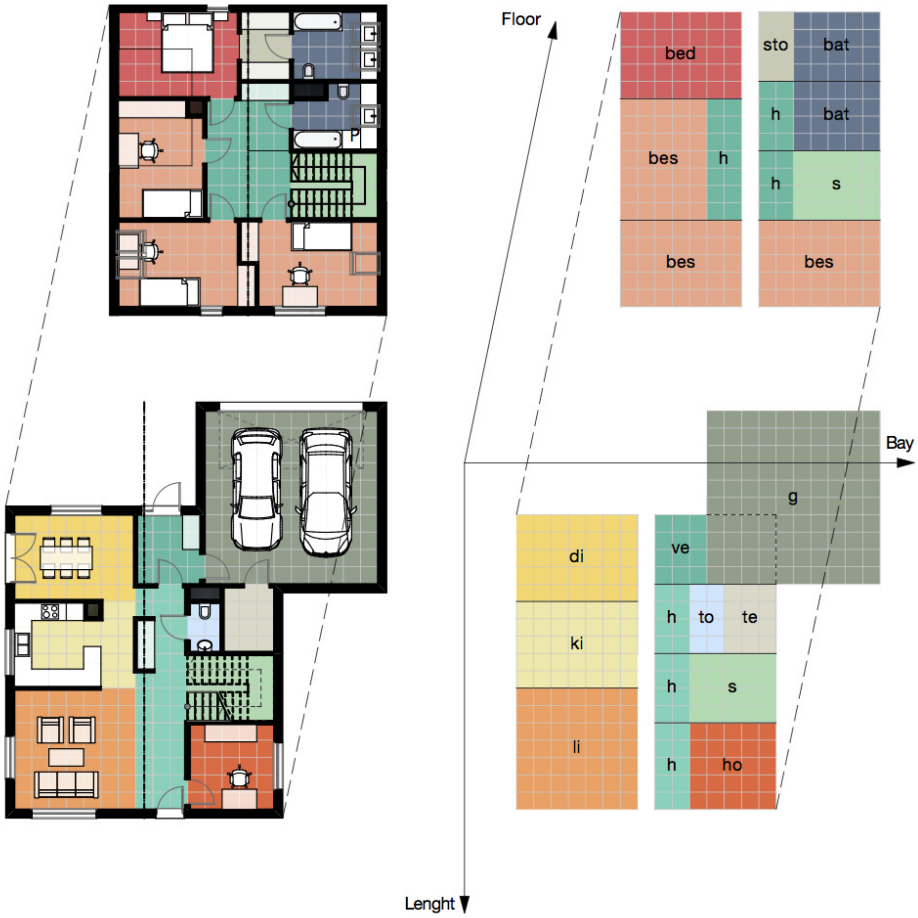

Fig. 2. Dimensionality of gablefront house layout: an example of two story two bay house layout and its decomposition in three dimensional space Floor-Length-Bay.

terminal symbols (simply called *terminals*), (c) a nonterminal S is the initial symbol of the grammar and (d) P is a finite set of productions. A pair (α, β) of strings of nonterminals and terminals is a production assuming that the first element α of the pair is a nonempty string. Productions are usually denoted $\alpha \rightarrow \beta$. Grammars having all productions with α being a nonterminal symbols are context-free grammars.

A derivation in a grammar is a finite sequence of strings of nonterminals and terminals such that: (a) the first string in this sequence is just the initial symbol of the grammar and (b) for any two consecutive strings in the sequence, the later one is obtained from the former one applying a production in the usual way, i.e. by replacing a substring of the former string equal to the left hand side of the production with the right hand side of the production. We say that the last element of the string is *derivable* in the grammar.

For a context-free grammar a derivation can be outlined in a form of derivation tree, i.e. (a) the root of the tree is labelled with the initial symbol of the grammar and (b) for any internal vertex labelled by the left side of a production, its children are labelled by symbols of the right side of the production.

Attribute Grammars An attribute grammar is a formal way to define attributes for the productions of a formal grammar, cf. [7]. The attributes are tied to symbols of productions (terminal and nonterminal symbols). The attributes are divided into two groups: synthesized attributes and inherited attributes. The attributes are assigned values at the derivation process. The values of the inherited attributes are passed down from parent nodes. The values of the synthesized attributes are the results of the attributes' evaluation rules applied to right hand side of the production or, for derivation tree, to children of a node. For simplification, the values of the synthesized attributes can be computed using inherited attributes and across from left neighbors.

In our approach, synthesized attributes are used to pass semantic information up the parse tree, while inherited attributes help pass semantic information down and across it. Attributes make it possible to validate semantic checks associated with a grammar, representing the rules of a language not explicitly imparted by the syntax definition. The strength of attribute grammars is that they can transport information from anywhere in the abstract syntax tree to anywhere else, in a controlled and formal way.

2 Syntactic Structuring - House Layout Customization

2.1 The Grammar

The initial productions of the grammar create the topmost level of hierarchy defining floors of the subjected house. Depending on the number of floors they are being placed in the Floor dimension. Floors of the house are having its length therefore they can also be placed in Length dimension. Building floors can be longitudinally divided into separate bays. If building is having rectangular shape then the floors are also rectangular and eventually bays are having equal length which is equal to the length of the building. Buildings might have various number of bays depending on its functional program, structural system and building typology. Therefore for the purpose of gable-front house presented grammar was simplified to the grammar generating just two bays of equal length on all the floors.

$<building> \rightarrow \ <floor> <buidling>$
$\qquad\qquad\quad \rightarrow \ <floor>$
$<floor> \rightarrow \ <bay1> <bay2>$

The fallowing part of the grammar divides Bays into zones. Zones allows to group rooms in order to prevent unallowable configurations of rooms. Rooms are grouped into three zones which correspond to degree of their privateness in the building. Lay out of zones create a sequence which begins with the entrance zone which is fallowed by semipublic zone and privet zone.

<*bay1*> → <*zone_entrance*> <*bay1*>
→ <*zone_semipublic*> <*bay1*>
→ <*zone_private*> <*bay1*>
→ <*zone_semipublic*>
→ <*zone_private*>
<*bay2*> → <*zone_entrance*> <*bay2*>
→ <*zone_semipublic*> <*bay2*>
→ <*zone_private*> <*bay2*>
→ <*zone_semipublic*>
→ <*zone_private*>

The next part of the grammar generate rooms in the Length dimension inside entrance zone. Entrance zone is divided into entrance slot and technical slot. Slots are introduced to allow placing collection of rooms across bays. They allow to generate rooms in the same bay and at the same localization along Length dimension but one after the other. The placed rooms have the width of the slot (exception is garage which protrude rectangular building shape) and the sum of rooms depth is equal to the depth of the slot. It is also used to introduce hall in front of the room.

<*zone_entrance*> → <*entrance_slot*> <*technical_slot*>
<*entrance_slot*> → <*vestibule*> <*garage*>
<*vestibule*> → vestibule
<*garage*> → garage
<*technical_slot*> → <*hall*> <*toilet*> <*tech_room*>
<*hall*> → hall
<*toilet*> → toilet
<*tech_room*> → tech_room

The fallowing part of the grammar presents method for generating sequence of rooms inside semipublic zone. For the purpose of constraining generation of rooms in relation to each other s_slot and s_set which stands for semipublic slot and set were introduced. As it was described in the previous paragraph slot allows to place collection of rooms across bay at the same Length dimension. Therefore it was utilized to introduce hall in front of the room. Contrary set allows to force generation of certain rooms next to each other. In the presented grammar kitchen is always being placed next to the dining room. Moreover both of this rooms can either have or not have a hall attached to it. Therefore the set in this case is combined of the slots allowing to introduce hall. Presented method could be easily adopted for the purpose of privet zone therefore presentation of this part of grammar was omitted in this article.

<*zone_semipublic*> → <*s_slot*> <*zone_semipublic*>
→ <*s_set*> <*zone_semipublic*>
→ <*s_slot*>
→ <*s_set*>
<*s_slot*> → <*hall*> <*s_slot_room*>
→ <*s_slot_room*>
<*s_slot_room*> → living room | home office | stairs

<s_set> → <s_set_slot> <s_set_slot>
→ <s_set_slot>
<s_set_slot> → <hall> <s_set_slot_room>
→ <s_set_slot_room>
<s_set_slot_room> → kitchen | dining room

2.2 Derivation Trees

Derivation trees of language constructions in given grammar are best to identify syntactic structuring. Therefor, derivation tree of exemplary two story gablefront house, presented in Figure 2, is considerd. Figure 3 presents a part of a derivation tree of the house layout from Figure 2. The derivation tree covers second bay of the ground floor. The derivation is developed to the level of individual rooms area forming layout of the house.

3 Semantic Analysis - Design Control

To verify the design of the house layout the data regarding the total width of all the bays and the location, width, adjacency and accessibility of every room has to be verified. Therefore such data has to be provided in the grammar or - more precisely - in the derivation tree. To transfer needed data we can use attributive tools in the grammar, i.e. we use attributive grammars. Such grammar is obtained by furnishing productions with inherited and synthesized attributes.

Attributes are attached to the grammar considered in this paper. They are responsible for collecting and transferring important data between items of the information space. Complete grammar with all the attributes and rules of their evaluation is not being presented in this paper. Instead ideas of using attributes are being outlined.

3.1 Verification of room addition

In order to guarantee the completeness of building layout it has to be verified that all the required rooms were added. Figure 4 presents two methods for verifying this requirement. Left side of the figure presents an example of synthesizing the attributes of *count_s* which calculates the total number of stairs added to the building layout. The data collected by this attribute is being verified at the trunk of the derivation tree in order to check if the total amount of stairs meets the required amount. Right side of the figure presents an example of inheriting attributes of *count_s*. This attribute is also used to calculate the total number of stairs added to the building layout but the data collected by it is verified after any room is added to the layout. The second method allows to prevent invalid grammar productions before they are completed while the first one require to generate complete house layout which afterwards is being verified. Based on these methods the amount of all required rooms can be individually verified.

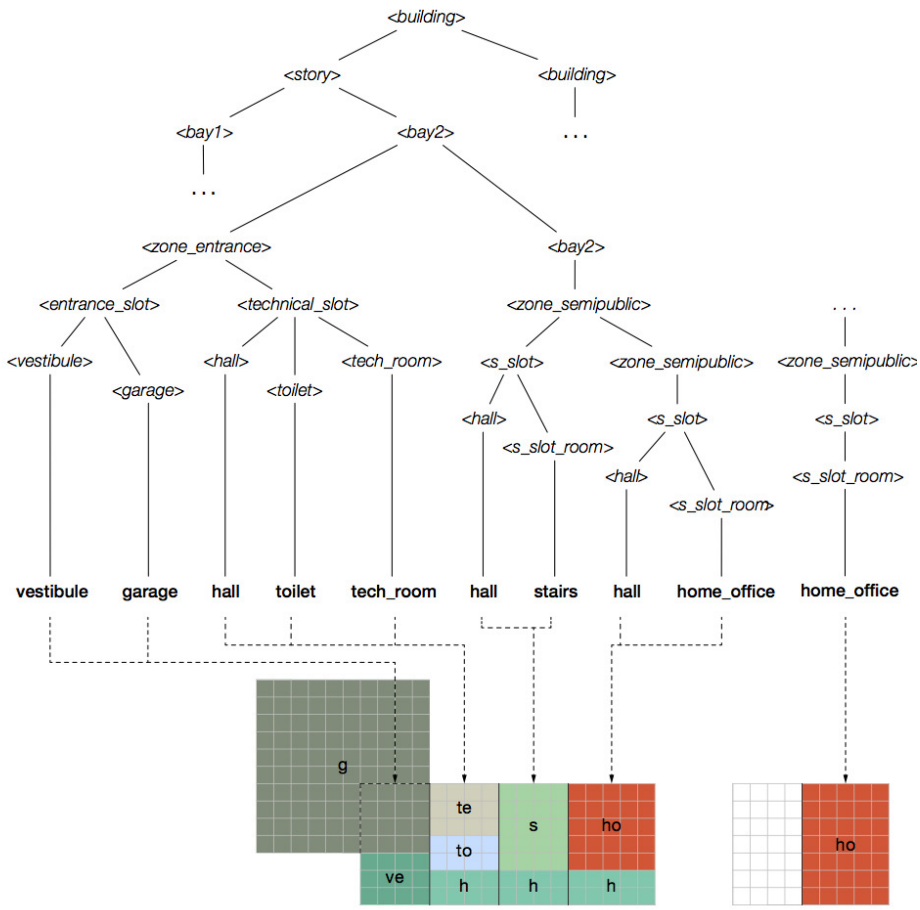

Fig. 3. Part of derivation tree in the grammar shown in section 2.1. A second bay of the ground floor of the house from Figure 2 (left part of the Figure) and exemplary room without hall (right part of the Figure).

3.2 Verification of widths

Verification of room width in order to fit to the allowed range is a trivial task and is being checked at the level of individual rooms (leaves). Additionally verification of the length of all the bays on all the floors also has to be checked, in order to guarantee rectangular shape of the building. Figure 5 presents an example of inheriting attribute of the building width and synthesizing the attributes of rooms width. Attributes of given nonterminal and their values are listed below nonterminal's name. Arrows with attributes names show direction of flow of values of attributes. For instance, the nonterminal $<bay2>$ has attribute: $free\_width$. This attribute get value from attributes of $<zone\_entrance>$: $free\_width$. Which in turn gets the value from synthesize attributes from generated elements of house

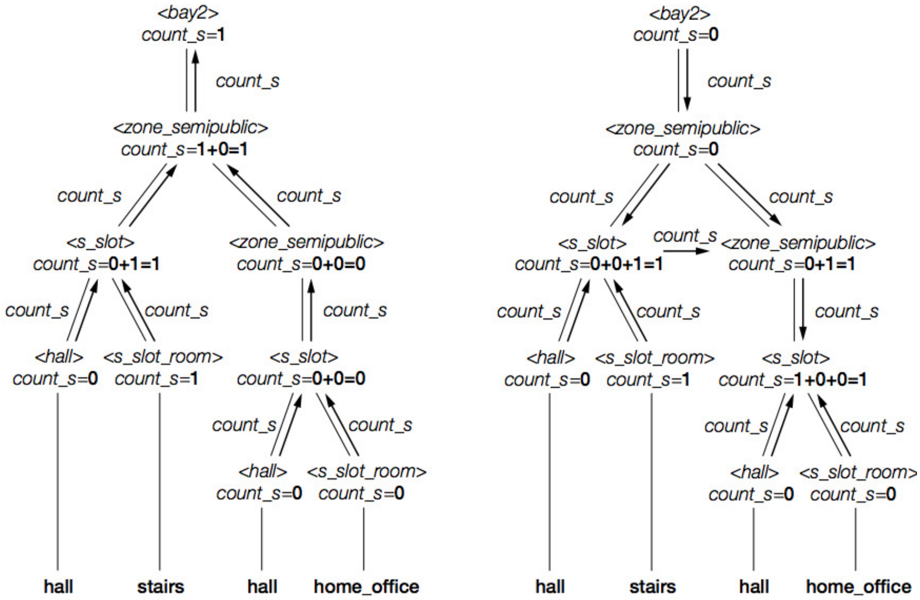

Fig. 4. Synthesized attributes (left part of the Figure) and inherited attributes (right part of the Figure).

layout *slot_width* and *slot_width* and inherit attribute from <*bay*2> $free\_width$. The date collected at the nonterminal <*zone_semipublic*> of the derivation tree allows to further calculate total free bay's width which is left to allocate next room before it is being added. If attribute $free\_width$ is equal zero then no additional room can be added and the length of bays are equal and they are equal to the width of the building. If it is bigger then zero there is still left space to add another room. When it is lower then zero then the added room is too big and it's size has to be reduced or this room has to be removed. This verification allows to control that the house is in a rectangular shape.

3.3 Verification of zones connectivity

As it was described in Section 2 bays are being divided into zones, which group rooms in order to place them next to each other. Therefore different bays could be differently divided into zones. In order to provide continuity of zones in the building layout the connectivity of this zones laying in different bays has to be verified. Figure 6 presents an example of information flow in the derivation tree of an attributive grammar which allows to control connectivity of semipublic zones on the ground floor. For instance, the nonterminal $< bay2 >$ has attributes: *bay*2*_s_start* and *bay*2*_s_end* which represents the beginning and the end localization of the semipublic zone placed in bay2. The attributes get the value of <*zone_semipublic*>: *start* and *zone_width*. The *start* value is being inherited from the nonterminal <*zone_entrance*> *zone_width*. The *zone_width*

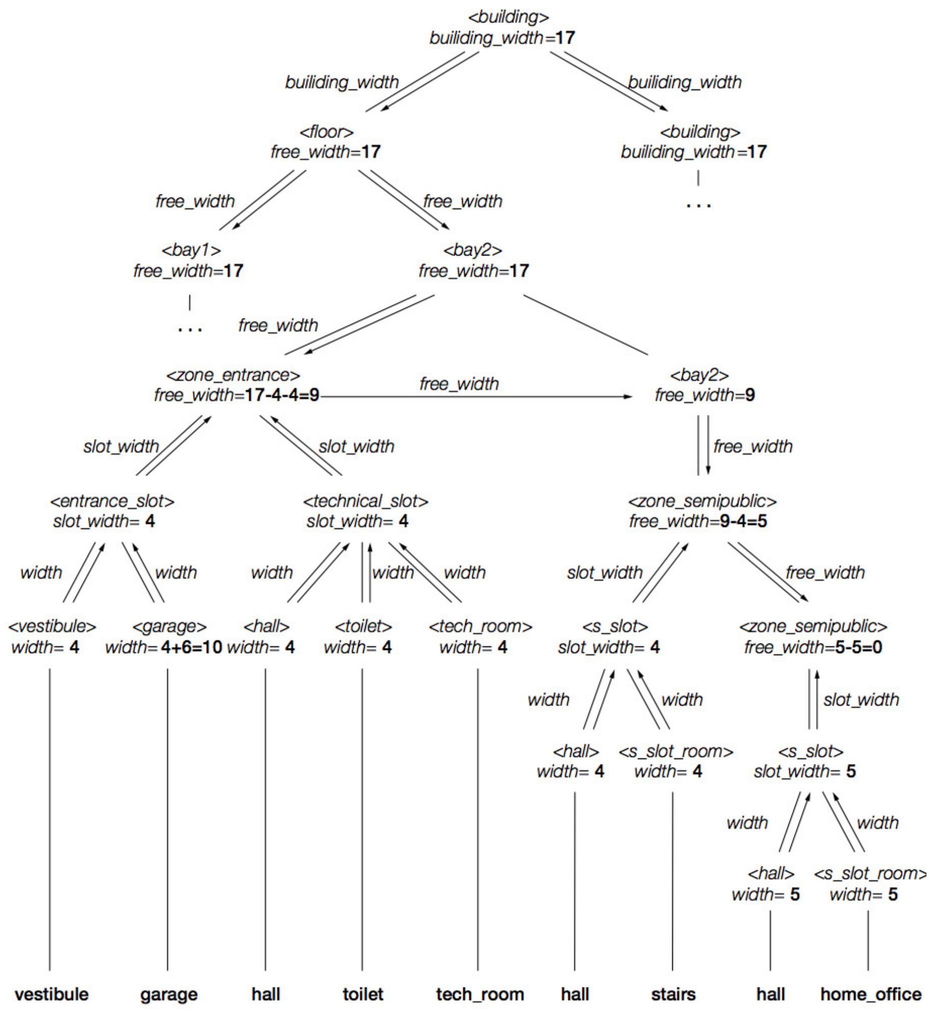

Fig. 5. Inherited attributes controlling equality of lengths of bays.

gets the value from synthesize attributes from generated elements of house layout $slot\_width$ and $zone\_width$. The date collected at the nonterminal $<floor>$ of the derivation tree allows to calculate if the semiprivate zones placed in different bays overlap for at least 2 modules which guarantees their connectivity. This method can also be utilized to control rooms adjacency. Division of bays into zones allow to control if the rooms are grouped correctly. Additionally sets of rooms allow to control if the required rooms are placed next to each other in the same bay. But these conditions are not enough to ensure that the required rooms are placed next to each other when they are added to the different bays. Therefore presented in a previous paragraph method allows also to control adjacency of rooms being placed in two different bays.

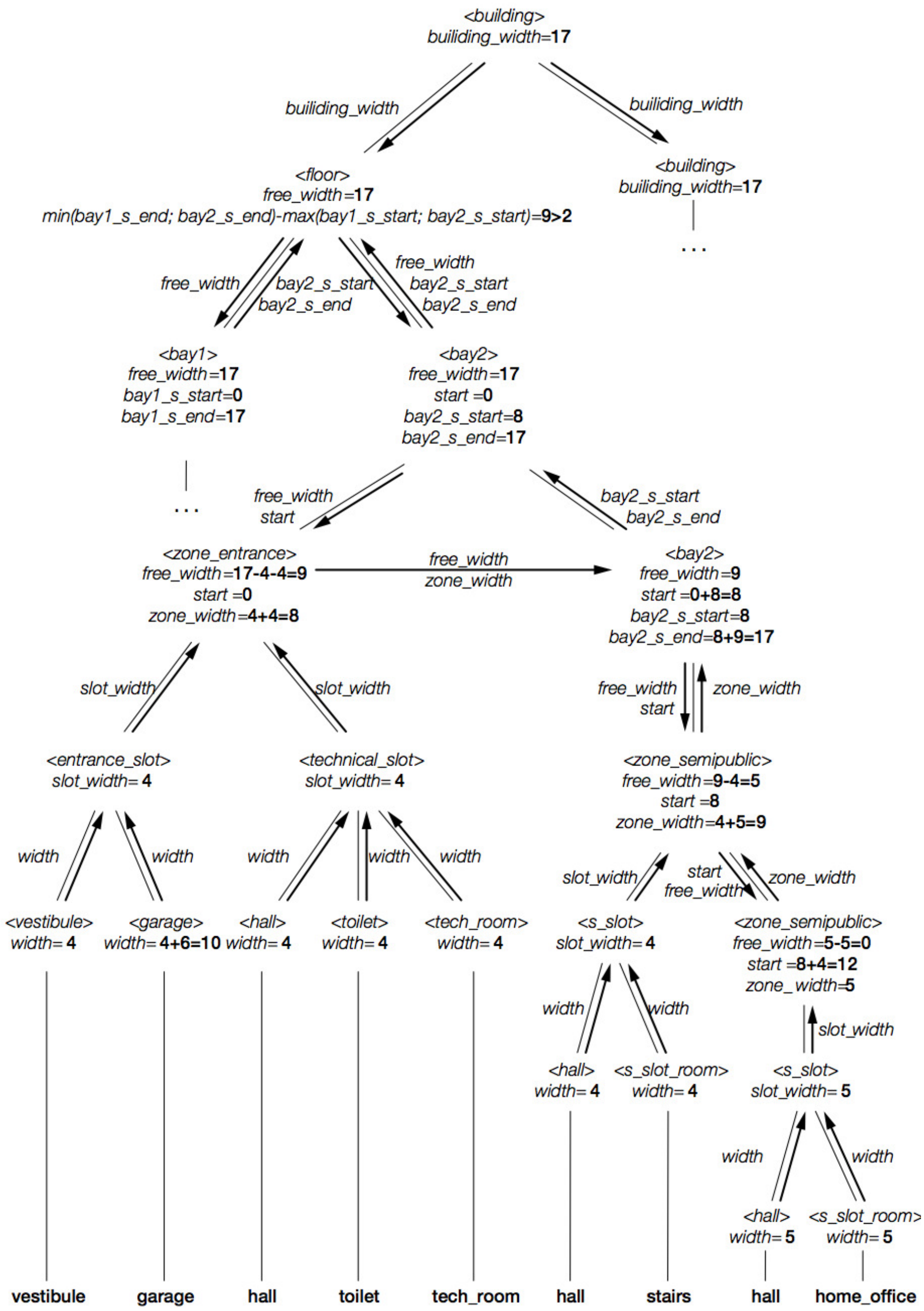

Fig. 6. Information flow in the derivation tree of an attributive grammar: control of bays length and zones connectivity.

4 Conclusions

This paper introduces a new framework for design automation, which is a case of more general problem of building tools for operating on knowledge structures. The discussion is focused on syntactic processing nested in a selected architecture problem of house layout customization. House layout is interpreted as multi dimensional concepts structure of domain knowledge. Syntactic structuring is based on context-free methods. We propose constructions of context-free grammars driven by concepts of multidimensional knowledge space. The formalism affords consistent representation of an approximate grammatical correctness of house layout. Furnishing grammars with attributes allows for information flow between domain's concepts supporting features disclosing and constrains verification. The approach is validated on some simple house layout and the directions for the future development of the system are outlined. The paper focuses on conceptual framework only and, as such, is intended to stimulate further research into the various implementation considerations that are prerequisite of large-scale applications.

Future directions for the introduced ideas are both theoretical and practical. The former ones include generalizing concepts to be more applicable in different domains. The later ones are intended to develop methods and tools directly applicable in design automation of architectural domain.

References

1. Bargiela A., Homenda W., Information structuring in natural language communication: Syntactical approach, Journal of Intelligent & Fuzzy Systems, 17(6), 575-581, (2006)
2. Duarte, José Pinto. Customizing mass housing: a discursive grammar for Siza's Malagueira houses. Diss. Massachusetts Institute of Technology, (2001).
3. Higuera C. A bibliographical study of grammatical inference, Pattern Recognition 38, 1332-1348 (2005)
4. Homenda W., Integrated syntactic and semantic data structuring as an abstraction of intelligent man-machine communication, Proc. of the ICAART - International Conf. on Agents and Artificial Intelligence, 324-330 (2009)
5. Homenda W., Automatic data understanding: a necessity of intelligent communication, Rutkowski L. et al. (Eds.), LNAI 6114, 476-483 (2010)
6. Homenda W., Rybnik M., Querying in Spaces of Music Information, Proceedings of the IUKM, Symposium on Integrated Uncertainty in Knowledge Modelling and Decision Making, LNAI 7027, pp. 243-255 (2011)
7. Knuth D. E., The genesis of attribute grammars. Proceedings of the Int. Conf. on Attribute Grammars and their Applications, LNCS 461, pp. 1-12, (1990)
8. Kwieciński, K., Słyk, J.: Interactive design system for provisioning of customized houses. In: eWork and eBusiness in Architecture, Engineering and Construction, pp. 649–655. CRC Press, (2014)
9. Rosani A. et al., Human behavior recognition using a context-free grammar, Journal of Electronic Imaging 23(3), 033016-1-12 (2014)
10. Stiny, George, and William J. Mitchell. "The palladian grammar." Environment and planning B 5.1 (1978): 5-18.