Artificial Intelligence for Engineering Design, Analysis and Manufacturing

cambridge.org/aie

Shape grammars as design tools: an implementation of a multipurpose chair grammar

Sara Garcia[1] and António Menezes Leitão[2]

[1]CIAUD, Faculdade de Arquitetura, Universidade de Lisboa, Lisbon, Portugal and [2]INESC-ID/Instituto Superior Técnico, Universidade de Lisboa, Lisbon, Portugal

Research Paper

Cite this article: Garcia S, Menezes Leitão A (2018). Shape grammars as design tools: an implementation of a multipurpose chair grammar. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **32**, 240–255. https://doi.org/10.1017/S0890060417000610

Received: 25 October 2016
Revised: 18 July 2017
Accepted: 11 August 2017

Key words:
Design tool; multipurpose chair grammar; shape grammar implementation; user experience

Author for correspondence:
Sara Garcia, E-mail: sgarcia@fa.ulisboa.pt

Abstract

This paper presents a multipurpose chair grammar and its implementation in the design tool *ChairDNA*. This tool is oriented for the exploration of design alternatives in the early concept phase of the chair design process. This work addresses two shortcomings within the research area of shape grammars (SGs), namely, the lack of implementation of SGs applied to design domains, and the lack of practical applications in real-life design scenarios. To address these problems, a methodology is proposed for the implementation of a SG (more specifically, a set grammar) into a tool, comprising the translation of the grammar into user-interface elements oriented for design practitioners. By using the proposed tool, the user can add/delete chair components and edit shape parameters, while visualizing the effects on a three-dimensional digital model presented in a variety of CAD applications. Compared with other SG implementations, ChairDNA uses an approach that keeps under control the combinatorial explosion of rule applications, which simplifies the use of the tool by designers that do not have experience in SGs. The generative potential of the tool is demonstrated by generating chairs of different types, and its usability and utility in aiding the designer are evaluated by design students and design practitioners.

Introduction

A shape grammar (SG) is a formalism to describe and generate designs based on the application of if-then shape rules. As the formalism relies predominantly on visual reasoning, it has been claimed to be suitable for the creative process of diverse design areas (Flemming, 1989). In fact, SGs have been applied to several design domains, such as urban design (Duarte & Beirão, 2011), architecture (Stiny & Mitchell, 1978), and product design (Agarwal & Cagan, 1998). SGs also address different design tasks, such as, the generation of original designs, the generation of new designs within a style or a brand (Pugliese & Cagan, 2002), the generation of product families (Castro e Costa & Duarte, 2014), and the transformation of existing designs (Eloy & Duarte, 2012). They are also useful for providing guidelines and constraints to the generation process (Duarte, 2001), and for evaluation and optimization tasks (Barros, Duarte & Chaparro, 2015).

However, there is a lack of studies regarding the designer's perspective on the use of SGs as design tools, because usually SGs are tested by merely the author of the tool, which, despite being a potential user, does not provide sufficient validation. Moreover, although the implementation of SGs brings several benefits to the design practice, few grammars applied to design domains have been implemented so far, and the methodology for the implementation of such tools is unclear (Strobbe et al., 2016). Considering also the currently available design tools, there is a lack of application in the early concept phases of the design process (McKay et al., 2012), and existing rule-based design tools have inappropriate interfaces (Chase, 2005).

In order to address these problems, the aim of this paper is twofold: on the one hand, to present a methodology for implementing a SG (more specifically, a set grammar) as a digital design tool, and on the other hand, to assess the usability and usefulness of the tool by conducting tests with potential users. The digital tool is oriented for the concept phase of a specific design problem and its use does not require specific knowledge either on SG or on three-dimensional (3D) CAD modeling.

The methodology for developing the tool requires the translation of a set grammar into an executable program. Differently from SGs, set grammars do not comprise the property of emergence – the ability to recognize and use subshapes (Duarte, 2001); thus, instead of operating directly on spatial entities, they operate on symbolic representations of those entities. Set grammars are more amenable for implementation (Stiny, 1982), as they avoid the combinatorial explosion of rule applications and the corresponding performance, usability, and utility

© Cambridge University Press 2018

CAMBRIDGE UNIVERSITY PRESS

problems. However, they remove ambiguity, which is considered a key feature to widen the range of unpredictable solutions (Knight, 2003).

The methodology comprises the use of rules with a hybrid graphical/symbolic representation. For instance, rules whose right-hand side describe added lines are encoded into a graphical realization of those lines and, also, into a symbolic representation based on Boolean values that assert the existence of the lines and numeric values related to their lengths or angles. These values are then matched with the left-hand side of additional rules that depend on the presence of those particular lines and their geometric properties. This simplifies the implementation and also allows the use of a Graphical User Interface (GUI), using checkboxes to control the application of the set grammar rules and sliders to control the rule parameters. This strategy for implementing the set grammar hinders the ability to change the rules but better matches the architecture of current computers, being very efficient to execute, to the point of supporting interactive changes on the parameters with immediate feedback in terms of the generated model.

The methodology is applicable to the development of different design tools that are based on set grammars, and in this paper, we show how it was applied to the development of a particular design tool – ChairDNA – for the concept phase of multipurpose chair design. The tool allows the user to explore several design alternatives within one product class, guaranteeing a wide diversity of solutions (some of them unanticipated) while ensuring the necessary restrictions (all designs obey correct anthropometric standards). The generation is processed by adding/deleting chair parts and editing shape parameters while visualizing the outcomes on a 3D digital model presented in a variety of CAD applications.

To assess the usefulness and user-friendliness of the tool, ChairDNA was evaluated by both students and expert designers. From the user's perspective, the tool provides fast and accurate generation and visualization of designs, helps redesigning solutions, and helps overcoming creative blocks by generating unexpected solutions. On the negative side, because the designs are controlled by GUI elements representing fragmented components, some designers found that the tool lacked intuitiveness and flexibility when compared with freehand drawing.

The paper is divided into two main sections. In the first one, we present a theoretical discussion on the role of SGs as tools for design, analysis, education, and customization. In the second one, we present a methodology for the implementation of SGs in digital design tools, which we particularize with a specific set grammar – the Multipurpose Chair Grammar (MCG).

SGs as tools

SGs can provide different tools, including analysis tools, pedagogical tools, design tools, and customization tools, usable by historians, students, designers, and end-users. For each tool, we provide some examples of practical application and user evaluations. Considering the goal of this paper, we give a particular emphasis on SGs used as design tools, by discussing its advantages and disadvantages.

Analysis tool

Analytic SGs are helpful to understand and clarify a design language, to identify if a given design belongs to a language, and to generate new designs within the language (Stiny & Mitchell, 1978). They have proved to be useful in numerous cases: in identifying a coherent style, for example, in a designer's work (Knight, 1989), in measuring the influence of a language in a given design, for example, how deeply the pupil adopts the style of his master (Figueiredo et al., 2014), in the reconstruction of missing parts of damaged artifacts (Mamoli, 2015), and in constructing new designs that must be coherent with a pre-existing language, for example, for the visual coherence and identity of a district (Flemming, 1987a). Although there is a large number of examples of analytic grammars, few of them were evaluated by users beyond the grammar developer. Examples include tests with owners of products of a specific brand in order to identify which features of a language can be relaxed before the core identity is lost (Pugliese & Cagan, 2002), and with the author of an individual style to identify whether new designs belong to his own style (Duarte, 2001).

Pedagogical tool

Knight (1999) argues that SGs have many pedagogical advantages, such as to teach students about: (1) constructive design processes, (2) notions such as proportion and symmetry, (3) how to externalize design ideas, and (4) how to consider multiple solutions instead of just one. Analytic grammars can teach students about design languages and how complex designs can emerge from simple patterns. Earlier SG examples were planned for pedagogical purposes (Stiny, 1980). Others proposed the use of SGs in design classes, providing grammars to the students so that they could use and transform them to build other design languages (Flemming, 1989). Some experiments with architecture students are documented in Li (2001) and in Duarte and Beirão (2011).

Design tool

SGs have been used in the generation of art and design objects. SGs can bring several advantages to a design process (Stiny, 1980), such as: (1) generating complex and diverse designs through simple rules, (2) generating unforeseen solutions to the designer, thus, enhancing creativity, (3) clarifying the process and design knowledge in an understandable and shareable way, (4) generating a family of designs instead of one single design, (5) easily adapting a design to suit different contexts, and (6) generating new languages by modifying, removing, or extending rules.

Despite these advantages, there are two opposite perspectives regarding the creative capacity of SGs. On the one hand, Kirsch and Kirsch (1986) state that, in grammars, designs will not transcend the corpus, and thus, are predictable and not truly innovative. As an example, Agarwal and Cagan (1998) consider that their grammar only generates standard solutions. On the other hand, some authors claim that SGs can generate creative solutions, either given by emergence or other processes. For example, Orsborn et al. (2006) claim that their grammar can provide truly innovative and original designs. Knight (2003) state that creativity in SGs is a result of emergence, which allows four key features of the design process: (1) *interaction*: a continuous reinterpretation that promotes the reframing of rules and designs, (2) *novelty*: the generation of new solutions, (3) *unpredictability*: the generation of solutions that are not obvious from the study of the rules, and (4) *ambiguity*: as shapes can be decomposed in many different ways. Nevertheless, Grasl (2012) claims that emergence

https://doi.org/10.1017/S0890060417000610 Published online by Cambridge University Press

is not needed to build a valuable SG that reproduces an extensive corpus of designs. In fact, creativity may be given by combinatorial processes, or ultimately, by the human user of the system (McCormac et al., 2014).

In the product design domain, SGs have been applied in the design of specific product classes, such as coffeemakers (Agarwal & Cagan, 1998), office chairs (Hsiao & Chen, 1997), and cross-over vehicles (Orsborn et al., 2006). This last case also included a user evaluation where a design practitioner generated designs using the grammar.

Customization tool

SGs can be applied as configurators, with the ultimate goal of mass-customization. Duarte (2001) focused on the customization of housing and tested the grammar implementation with potential clients. Barros, Duarte and Chaparro (2015) designed an integrated automated process, from idea to manufacturing, for the customization of Thonet chairs. Castro e Costa and Duarte (2014) created an SG-based system for the customization of ceramic tableware that is flexible enough to adapt to both designers and end-users.

Despite the uses previously described and the fact that SGs are being developed since the early 1970s, their applications in design practice are still quite sparse (Knight, 1999; Theodoros, 2013). SGs are mostly used as a proof of concept within a restricted scope and, typically, with analytic or pedagogical purposes (Chase, 2005).

SG implementations

SG implementations can serve different purposes (Gips, 1999): (1) design, by generating designs according to a given grammar; (2) analysis, by determining if a given design belongs to a language, (3) inference, by automatically generating an SG from a given set of designs, and (4) development, that is, to assist the user in the design of SGs.

SG implementations can be divided into generic interpreters and specific grammar implementations. A generic interpreter can reproduce several grammars, and thus, it allows the user to introduce a grammar and to generate designs with it. It requires users to know how to develop SGs. The first SG interpreter is the one of Gips (1975), and since then several interpreters have been developed to address particular features, namely: subshape recognition (Krishnamurti, 1980), 3D shapes (Earl, 1986), intuitive visual interface (Tapia, 1999), curves (Jowers & Earl, 2011), parametric rules (Grasl & Economou, 2013), and display of design alternatives (Strobbe et al., 2015). However, there is no SG interpreter that fully supports all the features, and some features are clearly underdeveloped, for example, the capability to express descriptions (McKay et al., 2012). These limitations make it difficult to implement a complex SG that requires large amounts of information for domain-specific design tasks (Li, 2002). In fact, most SGs implemented in interpreters are simple abstract ones (e.g., Tapia, 1999; McGill & Knight, 2004), or limited in scope, as the examples of the Coca-Cola bottle grammar (Chau et al., 2004), the Palladian grammar (Grasl & Economou, 2013), and the fractions of the Prairie house grammar and of the Malagueira grammar presented in Strobbe et al. (2015).

Differently from a generic SG interpreter, a specific SG implementation only supports one pre-defined grammar, and therefore, it only allows the user to generate designs with that grammar. However, it may support users who do not know about SGs. Paradigmatic specific implementations include the Queen Anne Houses (Flemming, 1987b), the coffeemakers mentioned in Chau et al. (2004), the ice-ray lattice windows mentioned in Stouffs and Wieringa (2006), and the Chinese wood-frame buildings according to the *Yingzao fashi* manual (Li, 2002).

There are advantages and disadvantages in using a computer implementation of an SG when compared with the manual application of the same grammar. The advantages, according to Flemming (1989) and Strobbe et al. (2016) are: (1) rapid exploration of design alternatives; (2) interactive visual flexibility – by allowing multiple representations; (3) adaptability of the solution – by quickly editing the derivation; (4) documentation of the results – by saving the design history, which allows the replication of results; (5) test of the accuracy and efficiency of the rules – by revealing some underconstrained or ambiguous rules and undesired results, which allows the grammar improvement; (6) automation of certain procedures – by the enumeration of the possible rule applications and the solution space, which is particularly advantageous for grammars that have many rules and are too time-consuming for manual exploration; and (7) augmenting the impact of SGs in design practice – by making them more accessible.

On the other hand, there are also disadvantages. These are, according to Knight (1999) and Gips (1999): (1) the lack of contact with physical shapes and the lack of involvement of the user with the process, which may be particularly disadvantageous for pedagogical purposes; (2) it may require a reduction in the potential ambiguity present in the rules – as the computer's current capacity of interpretation is below human's, and the symbolic nature of the computer contradicts the visual nature of SGs; and (3) the implementation of emergence potentiates performance problems (including subshape matching complexity and combinatorial explosion), usability problems (in presenting to the user so many alternatives), and utility problems (as emergent shapes might not be useful). According to previous summaries (Gips, 1999; Chau et al., 2004), although there are several implementations that handle emergence (e.g., Krishnamurti, 1980; Tapia, 1999), there are more implementations that rely on set grammars (e.g., Heisserman, 1994; McGill and Knight, 2004), which do not support emergence.

SG implementations have been tested mostly within the academic field. In some exceptional cases, tests were made with design practitioners, as was the case of the interpreter of Li et al. (2009), oriented to support users in designing with grammars and tested by novice and experienced users of SGs. In fact, most SG implementations have not yet been seriously used in real design scenarios; exceptions include the interpreter of Heisserman, Mattikalli and Callahan (2004), which was applied in aircraft piping design (Chau et al., 2004), and *CityEngine*, an application for city modeling based on set grammars and inspired by L-systems (Lindenmayer, 1968), which is having commercial success in the movies and games industry (Müller et al., 2006).

To be useful, a design tool must be able to solve non-trivial design problems. In this paper, we present one such problem: the design of multipurpose chairs which, as will be explained in the next section, requires a considerable amount of design knowledge. To address this problem, we developed the MCG and the ChairDNA tool, a specific implementation of that grammar.

The ChairDNA tool

There are few examples of design tools specific to one product class. In the case of chair design, there is SketchChair (Saul

https://doi.org/10.1017/S0890060417000610 Published online by Cambridge University Press

et al., 2011) that provides a graphical editor where users sketch the outline of a chair and the application generates the profiles needed for laser cutting. However, there are no anthropometric constraints being verified during the sketching process, and thus, no guarantee that the result is a comfortable chair or even a chair.

SGs have also addressed the chair design problem. Hsiao and Chen (1997) proposed a workflow for building a design tool, including the creation of a semantic model (an ontology), with adjectives suitable for describing characteristics of the intended product, and the development of an SG, which they then apply to the case of office chairs. However, the resulting tool is non-parametric. Barros, Duarte and Chaparro (2015) also developed an SG for chairs, where selected designs were implemented as parametric chair models, each one with a fixed topology, which could then be used for structural optimization. Nevertheless, the work is focused on a very specific kind of chair, namely, Thonet chairs.

To address the previous limitations, we developed ChairDNA, a set grammar implementation for aiding designers in the concept phase of the design of multipurpose chairs. The name ChairDNA comes from an analogy to the genetic code, which determines the similarities inherent to a species among their individual differences. Figure 1 shows the ChairDNA GUI alongside the generated model of a chair in Rhinoceros 3D.[1]

The development of the tool comprised a set of requirements: (1) *applicability* – to provide a useful tool for designers; (2) *usability* – to provide a simple, intuitive, and easy to learn interface; (3) *performance* – to provide quick visual feedback in response to the user's inputs; and (4) *accessibility* – to be accessible to designers, and thus, it should work with different commercial CAD applications and should be able to export and import designs.

A tool for practical application needs to provide useful support for one design task, by addressing a set of criteria: (1) to clarify the aspect of design to be supported and what benefits are expected, (2) to consider the end-users from the beginning, (3) to test the tool with users in order to assert effectiveness, and (4) to constantly process improvements (Flemming, 1994). The ChairDNA tool was developed to support the early phases of the design process, allowing a controlled exploration of different design alternatives. In order to involve the user within the tool development, we employed an evolving cyclic process with four stages: planning, grammar, implementation, and evaluation. So far, this process was repeated during two cycles, and therefore, ChairDNA had two major versions: version 1.1 and version 1.2, for Rhinoceros, AutoCAD, and SketchUp. In both cases, the connection to the CAD applications is ensured by Rosetta (Lopes & Leitão, 2011). The first version of the tool is briefly documented in (Garcia & Romão, 2015); the second version will be discussed below.

Given that the ChairDNA tool may be used by designers that do not know about the shape or set grammars, it is important to clarify our methodology for translating the grammar into user-interface elements. To this end, we will explain the grammar, and at the same time, the user interface.

The Multipurpose Chair Grammar

The ChairDNA tool relies on a parametric set grammar named Multipurpose Chair Grammar (MCG). The goal of the grammar is twofold: to generate a wide context-independent solution space, containing a large number of chair types, and to be implemented as a design tool for the concept phase of chair design. The grammar was developed through the analysis of a corpus of 26 chairs representative of a large number of types (e.g., one to four-legged chairs, arm and armless chairs, etc.), and by an ontological classification of the parts and types of chairs. The selected objects are iconic chairs designed between 1859 and 2003 by 24 well-known designers influenced by Modernism, whose general principles are: simple and abstract forms, harmony of forms and structure, honest use of materials, functional and affordable products, and use of new technologies and materials (Fiell & Fiell, 1994). Among these general principles, the last 150 years are characterized by a great diversity of chairs, caused by significant breakthroughs in materials and technology (e.g., tubular steel, molded plywood, and injection molded plastics). The importance of this object in design history, as well as the inherent complexity concerning form and structure motivated the choice of this case study.

The design language defines a particular type of chair – the multipurpose chair, which is suitable to different uses (thus, it excludes chairs with a single purpose, such as office chairs, lounge chairs, and children chairs), does not include mechanisms and comprises minimum upholstery. Moreover, it only considers chairs with bilateral symmetry and disregards any decorative elements and accessories (e.g., chair feet and chair pads). The design language employs a simplified representation of the structure and shape of the chair; the chair frame is represented by lines and the chair panels by planar surfaces (with no openings). To this wire-frame structure is then assigned a solid shape with a constant thickness.

The rules successively add the parts of the chair, step-by-step until a solution is reached. Each part is assigned with a label; thus, along with the geometric representation, there is also a symbolic representation. The grammar is label-driven, as labels are used to control rule application. Although the MCG was decomposed into parts, they are inter-related, so that when one part is changed, others might be affected. The grammar is discussed below through a brief explanation of rule sets, rules, rule parameters, and an example of a derivation.

(1) *Rule sets*: Rules are classified into rule sets that deal with the six main functional areas of the chair: legs, seat, back, stretchers, base, and arms (Fig. 2).

(2) *Rules*: The grammar contains 108 different rules. A rule corresponds to the addition of one part of the chair. For each functional area, the rules are divided into three configurations: guides, frame, and panels. This is illustrated in Figure 3, where we show the rules that are concerned with the generation of the legs. The guides (depicted in a gray tone) represent the bounding perimeter of the area. The frame represents the linear elements and can be further decomposed considering the position of the parts, for example, in the case of the legs, the front and back legs. The panels represent surface-based elements, and in the case of the legs, they can be decomposed into front panel, back panel, side panel, and radial panel. The termination rule, which erases all the labels and guides, can be applied when the crucial parts of the chair are placed: seat, back, and legs.

The grammar allows the user some choice regarding the rule application sequence. Figure 3 illustrates that choice: starting with the initial shape (on the left-hand side of all the rules), one may opt for any of the rules of the legs. On the other hand, some rules can only be applied when others are already employed (Figs. 7–9, left).

[1]The ChairDNA Quick Start Tutorial is available at http://chairdna.wordpress.com/tutorial.

https://doi.org/10.1017/S0890060417000610 Published online by Cambridge University Press

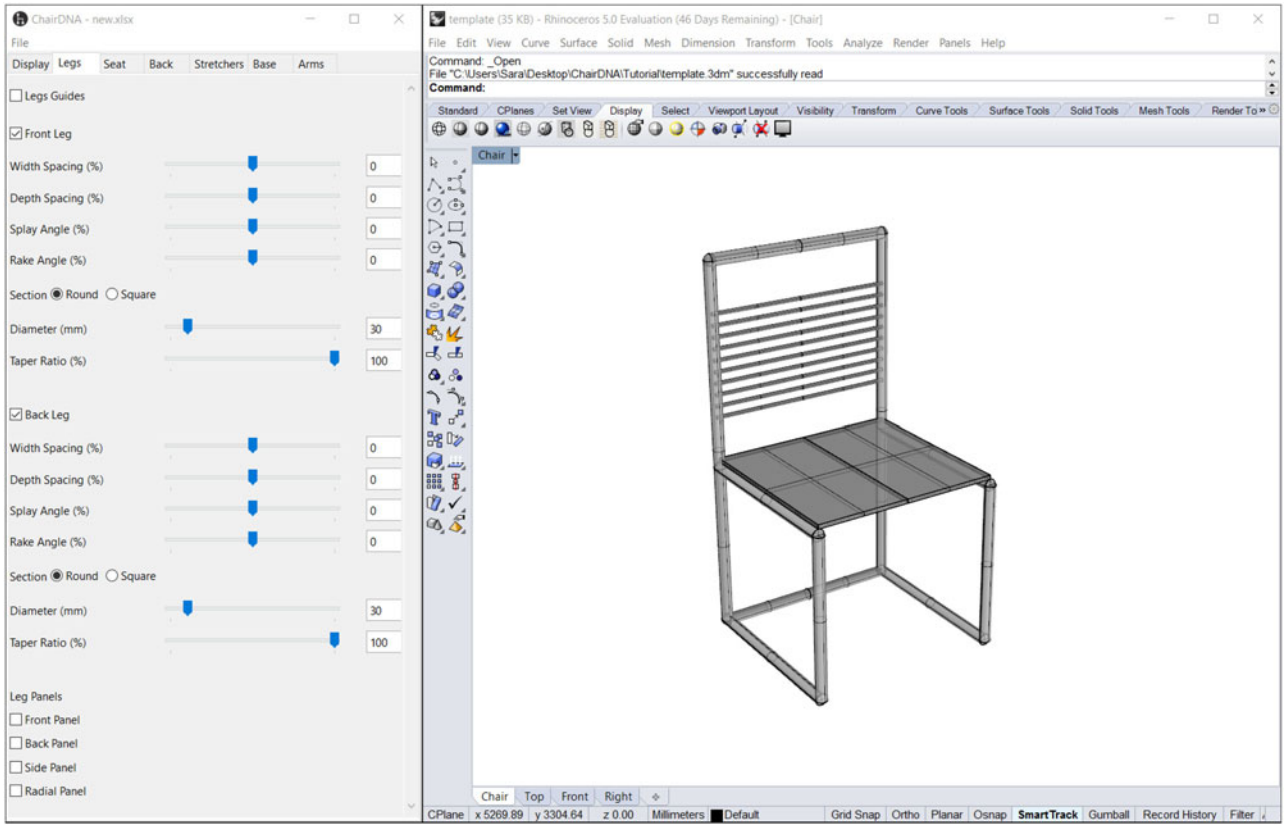


Fig. 1. Snapshot of the ChairDNA tool.

Beyond the shape part, there are additional rule application conditions, including (1) parametric conditions, where a rule is applied if certain parametric values in the left-hand side of the rule are observed; (2) geometric conditions, which impose relations between geometric entities in the left-hand side; and (3) negative conditions, where a rule is applied if a certain entity does not exist.

(3) *Rule parameters*: Most grammar rules have parameters, for example, dimensions, positions, curvatures, angles, etc. Some parametric ranges obey the anthropometric standards referenced in Tilley (2002). Figure 4 shows the parameters of the chair base and some possible variations. The parametric curve is controlled by five variables: width (*w*), depth (*d*), front radius (rf), rear radius (rr), and taper width (tw). Through the variations of values within the specified ranges, the curve can assume the shape of a square, a rectangle, a trapezoid, a triangle, a circle, or intermediate stages between them.

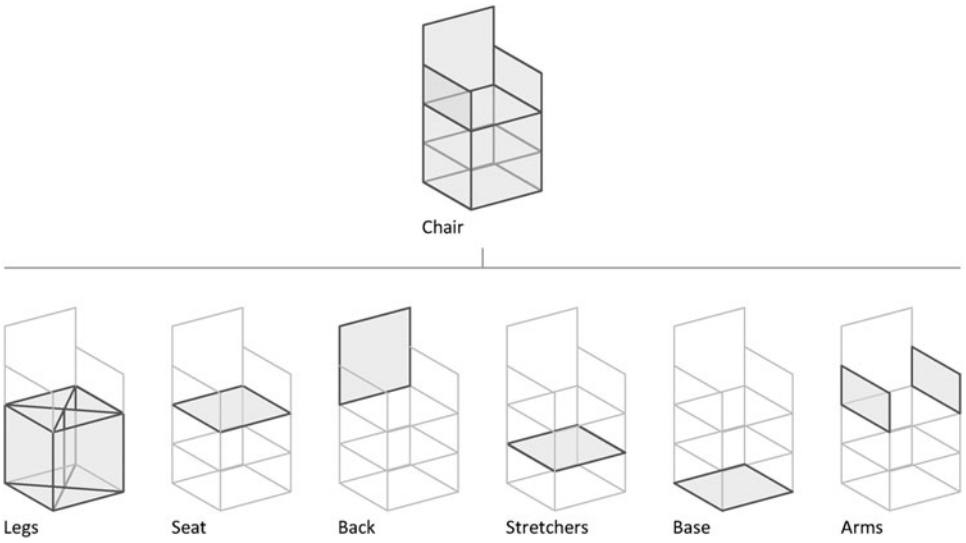


Fig. 2. Rule sets of the MCG.

https://doi.org/10.1017/S0890060417000610 Published online by Cambridge University Press

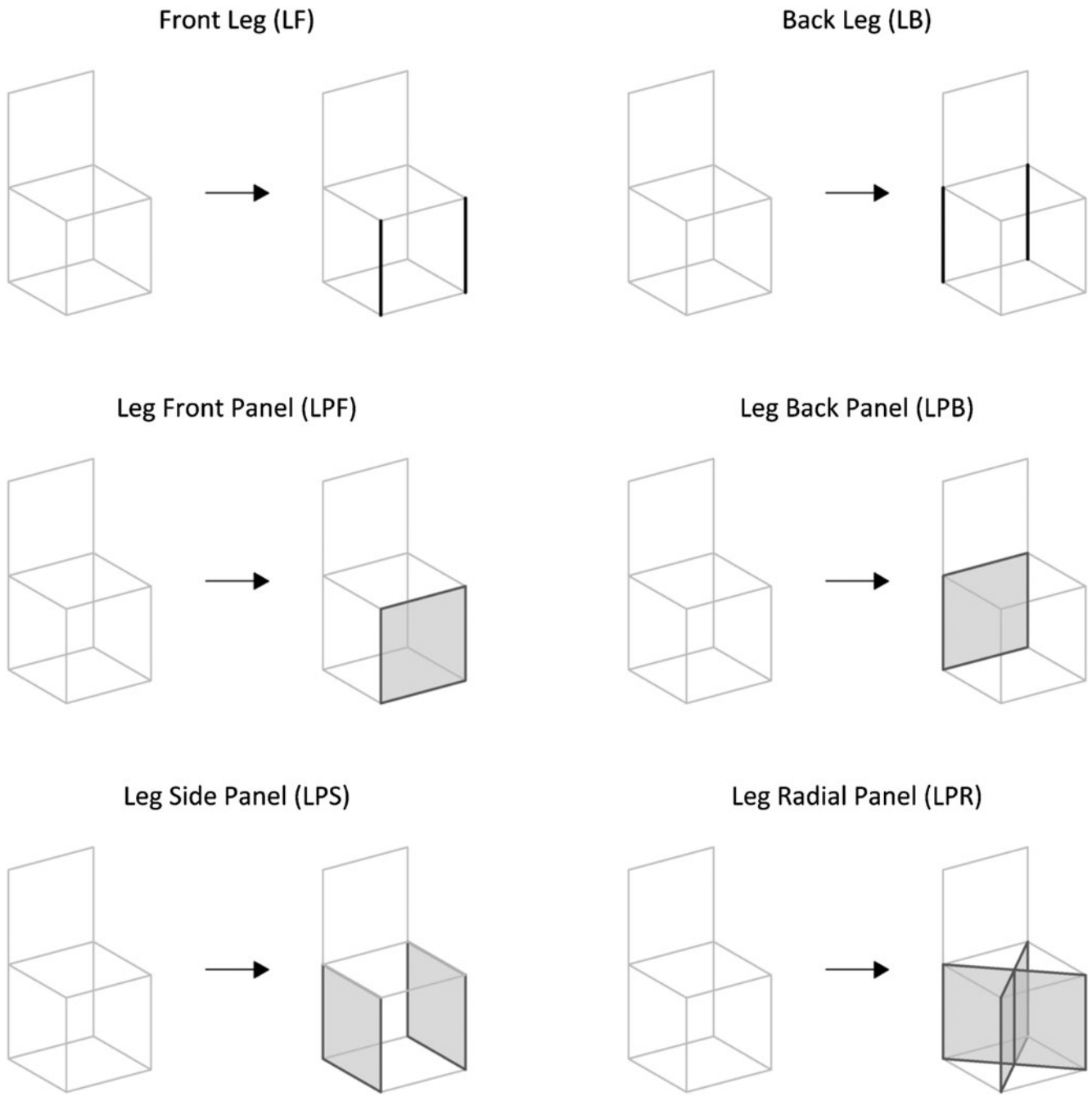


Fig. 3. Rules to place parts in the legs rule set.

(4) *Derivation*: Figure 5 illustrates a derivation of the Zig-Zag chair, designed by Gerrit Rietveld in 1934 (Fiell & Fiell, 1994), starting from the initial shape and ending in the final design. The first rule applied is the Leg Front Panel, shown in Figure 3.

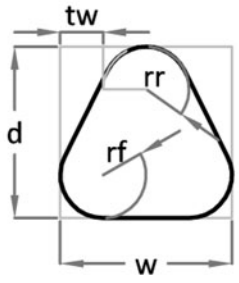
Implementation methodology

Given that ChairDNA users might not want to deal with grammar concepts, such as rule sets, rules, and derivations, we developed a methodology for translating a set grammar into GUI elements that control rule application and the derivation process. This will be illustrated by referring to Figure 6, which shows the implementation of a particular rule (Front Leg). The methodology is the following:

(1) Rule sets are translated into tabs, labeled with the name of the rule set. This means that, in the ChairDNA case, each tab represents a functional area of the chair.
(2) Rules are translated into checkboxes, labeled with the rule name. This is visible in the Front Leg rule and corresponding checkbox.
(3) When a rule is applicable (i.e., the left-hand side of the rule matches the current shape), the corresponding checkbox is enabled, otherwise, it is disabled. This last case is visible with the Seat Long Rail checkbox in Figure 7.

https://doi.org/10.1017/S0890060417000610 Published online by Cambridge University Press

Name	ID	Range	Default
Width	w	[1,100]	100
Depth	d	[1,100]	100
Front Radius	rf	[0,100]	0
Rear Radius	rr	[0,100]	0
Taper Width	tw	[0,100]	0



Base parameters

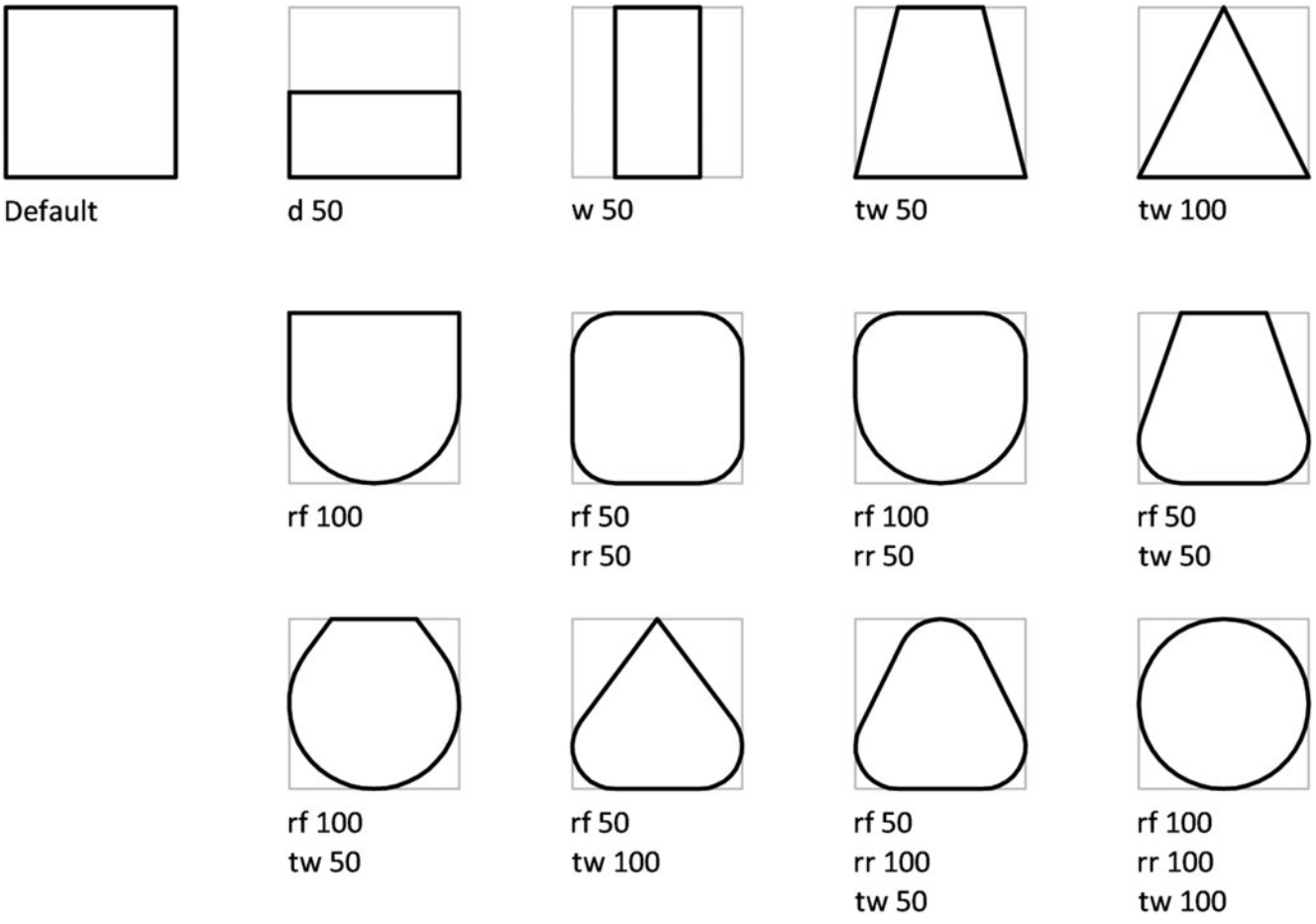


Fig. 4. Parametric variations of the chair base.

(4) Rule application (i.e., the effect of the right-hand side of the rule) is mediated by the corresponding checkbox: if the checkbox is unchecked, the rule is not applied, otherwise, the rule is applied and the corresponding part of the chair is inserted. In the example, the Front Leg rule is applied.
(5) Rule parameters are translated into sliders and text fields, whose values are limited by the range of the corresponding parameter. The example shows all the parameters related to the Front Leg, with the default values.
(6) When a rule is applicable more than once, the user interface depends on whether the rule uses the same parameter value on all applications or different values for each application. In the first case, there is just one interface element for the parameter, but it is complemented with a slider that specifies the number of applications, while in the second case, that number is used to inject in the user interface the corresponding number of rule parameters.
(7) Rules that have different left-hand sides but an identical right-hand side are merged into a single checkbox, which is enabled when any of the left-hand sides match the current shape.
(8) The current shape is always visible in the CAD program and is immediately updated whenever any user-interface element is changed. In the ChairDNA case, the shape can be viewed as a wireframe model or as a solid model, where the chair frame has a round or square section with user-specified dimensions.
(9) When the program starts, the initial shape of the grammar is used as current shape.

https://doi.org/10.1017/S0890060417000610 Published online by Cambridge University Press

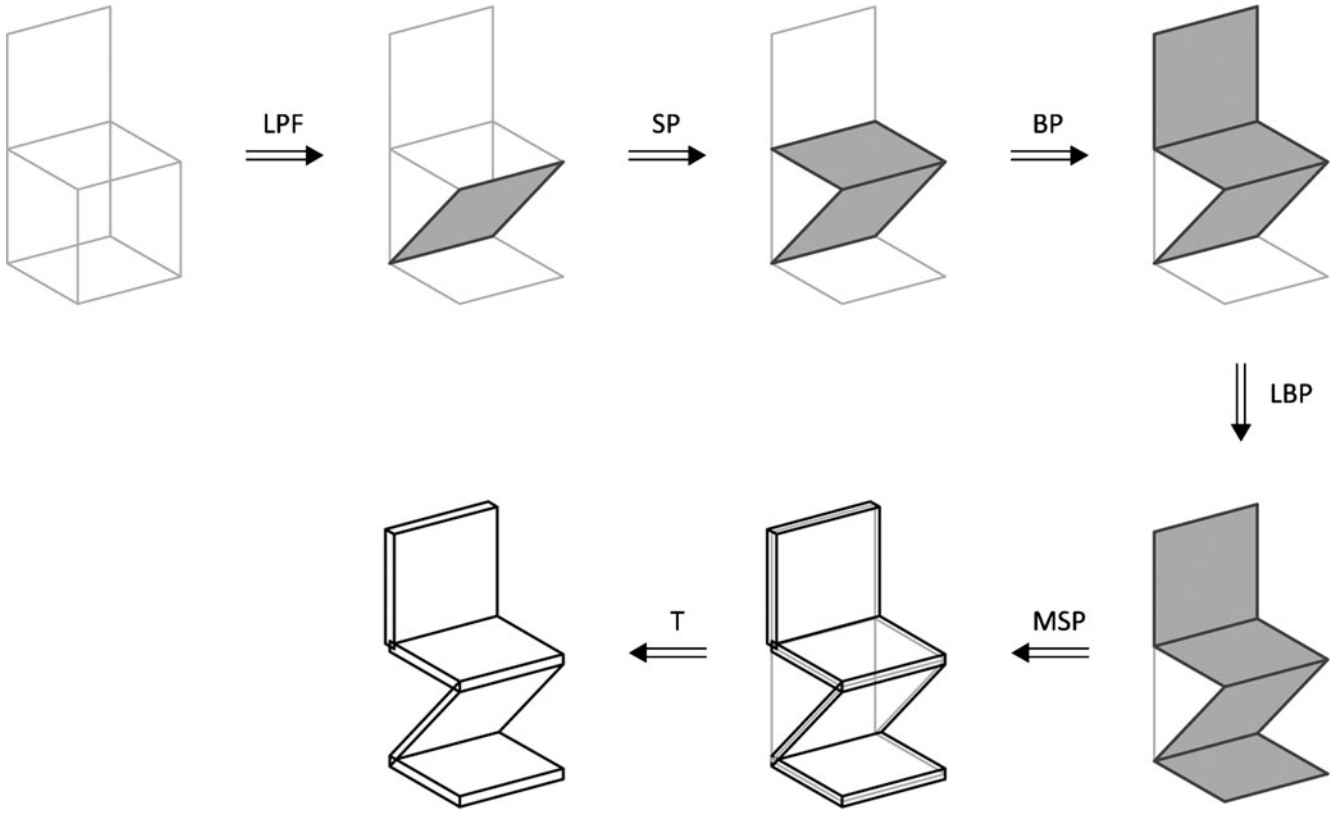


Fig. 5. Derivation of the Zig-Zag chair. Rule acronyms: LPF, Leg Front Panel; SP, Seat Panel; BP, Back Panel; LBP, Leg Base Panel; MST, Mode Solid – Thicken; T, Termination; MSP, Model Solid – Panels. Shape labels are omitted for clarity.

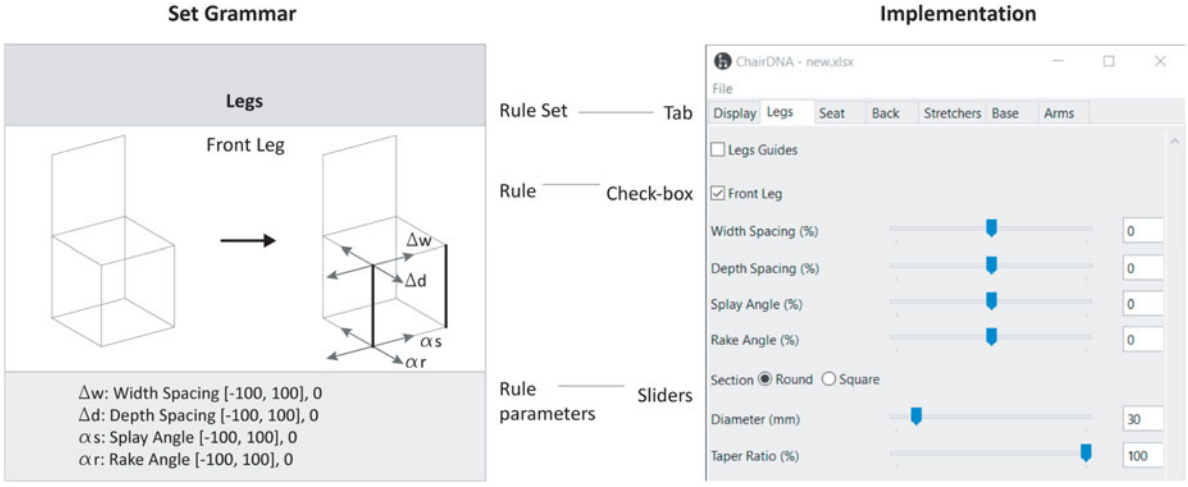


Fig. 6. Correspondence between the grammar and the implementation.

We implemented the ChairDNA user interface by applying the previous methodology to the MCG.

In order to faithfully support all the details of the MCG, the ChairDNA user interface contains seven tabs, 59 checkboxes, and 103 sliders. The difference in the number of grammar rules (108) and checkboxes is explained by the step 7 of the methodology. The implemented grammar is a slight variation of the original grammar because, as will be explained in the Evaluation section, we wanted the user to be able to finish the design process in intermediate steps of the derivation. To this end, the termination rule was not implemented.

Comparison

In order to compare the MCG and its implementation in the ChairDNA tool, the following aspects will be discussed: shape, rules, generation capabilities, and display.

https://doi.org/10.1017/S0890060417000610 Published online by Cambridge University Press

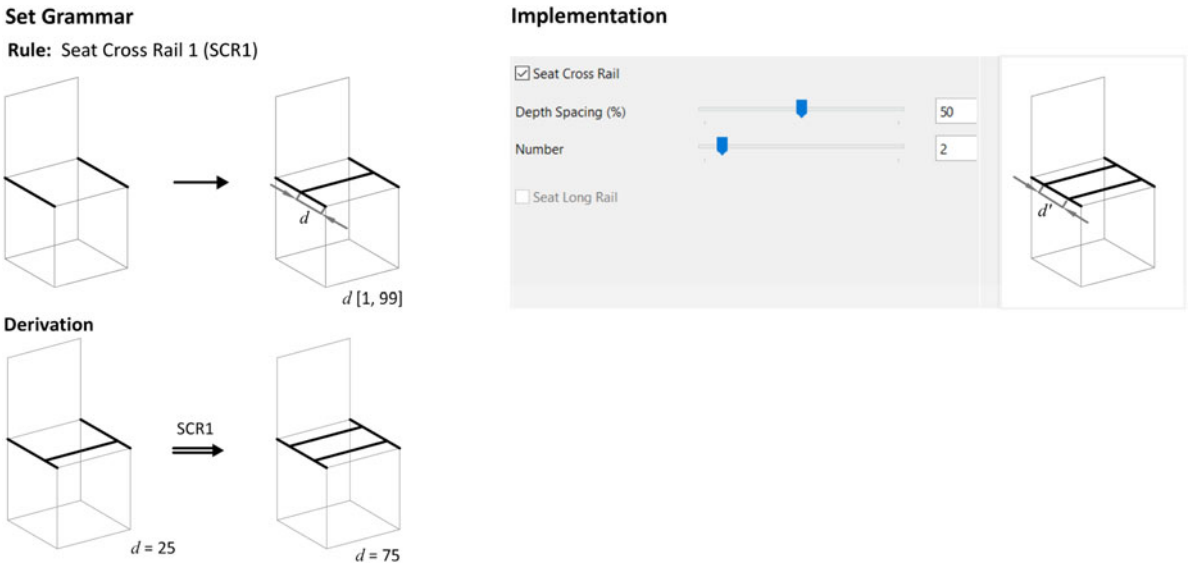


Fig. 7. A grammar rule (top left) and an example of its application (bottom left) and the corresponding implementation (on the right).

Shape

Based on the criteria presented in Garcia (2016) that capture the essential features of SGs, one can assert that both MCG and ChairDNA use lines (rectilinear and arcs), planes, and solids as shape basic elements; they both deal with parametric labeled shapes in 3D space, and they do not consider emergent shapes, nor transformations of shapes. The MCG uses weights by assigning thickness and color to shapes, while ChairDNA currently only considers color weights.

Rules

In the grammar, only additive rules are used. In order to undo a rule, one needs to go back in the derivation. On the other hand, in the implementation, one can use additive and subtractive rules at any time, effectively jumping between non-sequential derivation steps. As such, the user of the grammar is forced to use a bottom-up additive strategy, while the user of the implementation may opt to use a bottom-up or a top-down strategy. A top-down strategy may be employed because the user is able to import a given design from a library of pre-defined chairs, which will work as a template for further developments. Note that the chairs generated by the user may also become templates for further iterations. The tool exports and imports chair configurations using human-readable Excel files that store all the chosen parameters of a given solution, and that may be further edited by users. The Excel files may also be manually constructed by measuring and registering the values of a chair, a process that is further described in (Garcia & Barros, 2015).

Another difference comes from step 6 of the methodology, regarding the number of times a rule can be used. Although unlimited in the grammar, in the implementation the majority of rules can only be applied, at most, once, depending on the corresponding checkbox, and for those that can be applied more than once, the actual number of applications is provided by the user. On the left of Figure 7, we show an example of a parameterized rule where the parameter d can be different for each application of the rule. In its implementation, however, we use a slightly different parameter d', visible on the right of Figure 7, that is fixed regardless the number of applications specified by the user.

The implementation also combines grammar rules that have different left-hand sides but that insert the same part [according to step 7 of the methodology. In these cases, both rules are present in the implementation but their separate application is controlled by a single checkbox. This is visible in the example presented in Figure 8 (right), which shows the combined implementation of the rules Seat Cross Rail 1 (Fig. 7, top left) and Seat Cross Rail 2 (Fig. 8, left). The second rule is applicable only when the Seat Long Rail checkbox is checked.

Another problematic situation is visible in Figure 9, where the same rule may have two different applications. Typically, SG implementations that support this situation generate multiple derivations and then force the user to choose one among a potentially huge set. This approach has two problems: (1) it is inefficient from a computational point of view and (2) it makes it very difficult for the user to select the best option. To solve these problems, rules in ChairDNA do not generate alternatives. To this end, the implementation requires the elimination of rule ambiguity, and in the case of the rule in Figure 9, we decided that it always produces the second option.

Generation capabilities

The implementation supports one particular grammar and it does not allow the user to add or edit rules or labels. However, it allows for the manual or automatic generation of designs. The automatic generation creates designs based on a random selection of rules and parameters, which is not only useful for providing alternative templates for the user, but also as a generate-and-test method, where erroneous or inaccurate solutions are used for debugging and restricting the solution space of acceptable designs.

Display

In the grammar, rules are displayed visually with numeric parameters, while in the implementation, rules and parameters are displayed using user-interface elements, namely checkboxes, sliders, and text fields. The rules that are available at a certain stage of the derivation are more easily detectable in the implementation, as they are displayed as enabled user-interface elements; while in the grammar the user needs to check rule applicability for every

https://doi.org/10.1017/S0890060417000610 Published online by Cambridge University Press

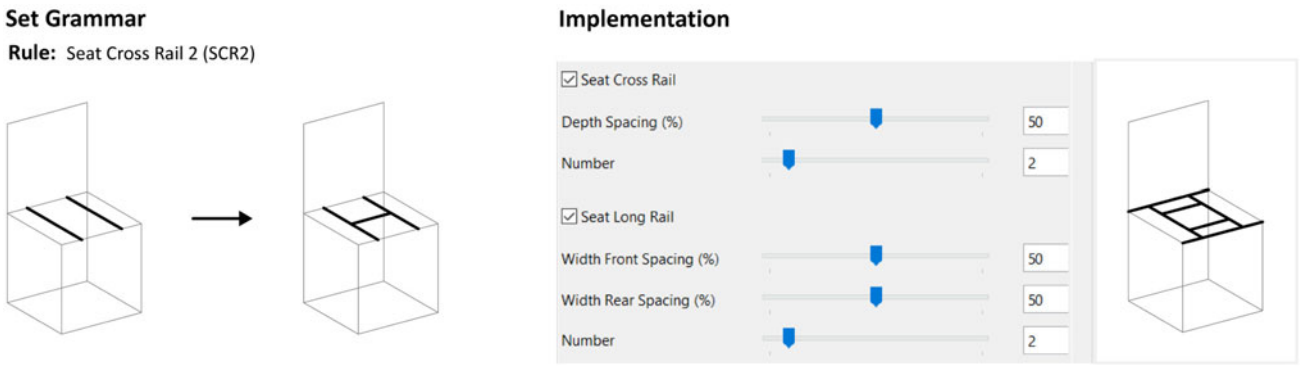


Fig. 8. A different grammar rule which places the same part (on the left) and its implementation (on the right).

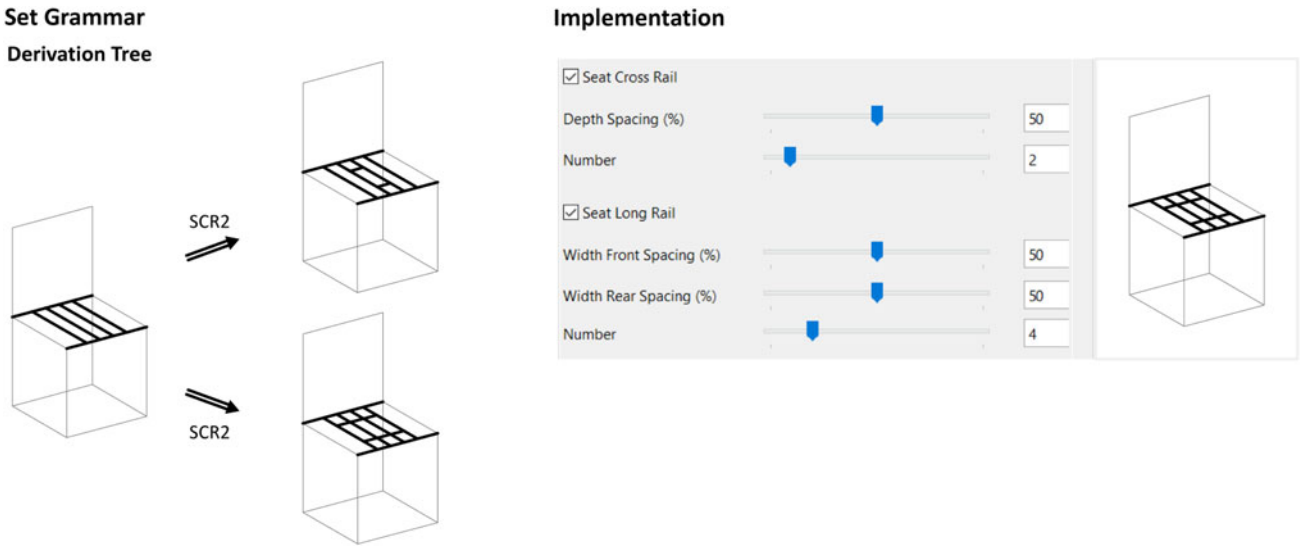


Fig. 9. Two possible rule applications (on the left) versus one single application in ChairDNA (on the right).

rule. Finally, it is worth mentioning that although the current shape is displayed visually in both systems, the implementation does not currently display the derivation, and there are no design alternatives when applying a rule.

Evaluation

ChairDNA was evaluated by employing the three tests mentioned by Stiny and Mitchell (1978) and summarized by Duarte (2001): the descriptive test, the analytic test, and the synthetic test. Moreover, and regarding the evaluation of ChairDNA as a design tool, we will detail a user evaluation oriented for synthetic purposes. In previous work, ChairDNA was evaluated as an analysis tool (Garcia & Barros, 2015), being able to restrict the solution space to one particular style, and as a pedagogical tool (Garcia & Romão, 2015), being evaluated by design students.

Descriptive test

The descriptive test evaluates whether the grammar generates the chairs of the corpus and, indeed, the ChairDNA tool can replicate all the chairs of the corpus; one example is given in design 1 of Figure 10, whose derivation was previously illustrated in Figure 5.

Analytic test

The analytic test evaluates whether the grammar generates existing designs that are not in the corpus but belong to the language. To this end, we estimated the extent to which the grammar can generate different designs in the same language, through the following procedure:

(1) Consider the implementation of a set of designs (A) in a generative system (B) with an abstraction level (C). In this test, A are the six most recent multipurpose chairs referenced in Design Museum (2010), that do not belong to the corpus; B is ChairDNA; and C considers the following characteristics: curves in the corner of the areas, constant thickness of parts (except in legs), round or square section shape, and intermediate stages between round, square, and trapezoid area shapes.
(2) Count the reproducible features (Fr): the features of a design of A that are reproducible by B, different from the default values.
(3) Count the irreproducible features (Fi): the features of a design of A that are not reproducible by B and are part of C (e.g., dimensions out of the range).
(4) The completeness of the reproduction of a design of A is given by the formula Fr/(Fr+Fi). The completeness is

https://doi.org/10.1017/S0890060417000610 Published online by Cambridge University Press

Table 1. Degree of completeness of the grammar

Chair name	Designer	Year	Fr	Fi	Completeness
Omkstak	Rodney Kinsman	1971	29	5	0.85
Wiggle	Frank Gehri	1972	20	7	0.74
Queen Anne	Robert Venturi	1985	37	2	0.95
Louis 20	Philippe Starck	1991	24	0	1.00
FPE	Ron Arad	1997	21	3	0.88
Air	Jasper Morrison	1999	16	1	0.94
				Mean	0.90
				SD	0.07

measured on a scale between 0 and 1, in which 1 means that B fully reproduces the design and 0 means that none of the features of the design are reproducible by B.

(5) The process is repeated for all designs of A. The arithmetic mean of the completeness of the designs gives an estimation of the completeness of B in reproducing designs in a language, in a defined abstraction level (C). For the completeness to be 1, it would be necessary for the system B to anticipate all the solutions within a language.

Table 1 presents the result of this evaluation, where the degree of completeness is 0.90. We also include the Standard Deviation (SD). Two of the evaluated designs are also reproduced in Figure 10 (designs 2 and 3).

Note that, as explained in the Implementation methodology section, the ChairDNA tool is also capable of generating designs that do not belong to the language of the MCG, as is the case of stools, generated as chairs without backrest. This is visible in Figure 10, design 4. Given that these designs were deemed relevant by the users, we currently consider this capability as an advantage.

Synthetic test

The synthetic test evaluates whether the grammar generates new designs that belong to the language. In fact, due to the fine-grained control over chair dimensions, the ChairDNA tool can generate chairs that are not present in the corpus, for example, by generating an interpolation between two different types. Figure 10 shows four chairs that are not in the original corpus: design 5 is a design generated by the grammar developer, design

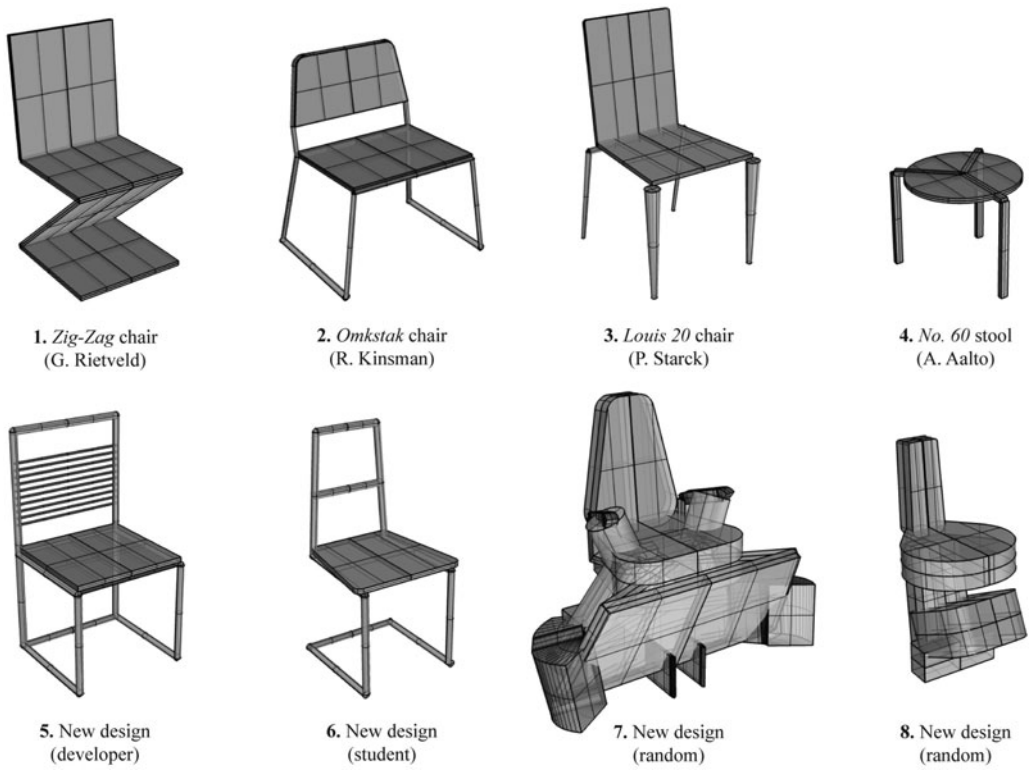


Fig. 10. Chair solutions produced by the ChairDNA tool.

https://doi.org/10.1017/S0890060417000610 Published online by Cambridge University Press

6 is a design generated by a design student, and designs 7 and 8 are randomly generated.

It should be mentioned that, although the tool only generates designs that obey the anthropometric standards, some of the designs might not be viable, as is the case of design 8, which seems to be unstable. These kinds of problems are perfectly acceptable in the concept phase, as trial-and-error is a typical process applied in creative problem solving, and intermediate stages of solutions are often incomplete, imperfect designs (Lawson 2008). In fact, as suggested in the next section, mistakes may provide the emergence of new ideas. If ChairDNA addressed the final phases of the design process, where detail becomes critical, it could either prevent errors to occur or notify the user when they occur.

User evaluation

In order to assess the applicability of ChairDNA in design practice, a user evaluation of the tool was conducted. The main goals of this evaluation were: (1) to evaluate the perceived usefulness of the digital tool as an aid for the designer in the concept phase of the chair design process, and (2) to evaluate the usability of the interface.

Method

The evaluation was carried out individually by one evaluator and one participant, at the participant's workplace. The test average length was approximately 1 h 45 min. The experiment was divided into three parts:

(1) A pre-test interview, oriented to gather information about the participant's profile and background as a chair designer.
(2) A user test, where after visualizing a ChairDNA video tutorial, the participant had to use the tool to accomplish four tasks: (1) a free exploration of the tool; (2) a reproduction of an existing chair; (3) a set of modifications upon an existing chair; and (4) the design of an original chair. During the test, the screen was being recorded, and the participants were encouraged to think aloud while the evaluator was observing and annotating their comments.
(3) A post-test questionnaire, where the participant's feedback was collected. Most answers were given on a five-point Likert scale, together with some qualitative ones.

Participants

The population was composed by design practitioners with recognized experience in chair design. The study population was restricted to Portuguese chair designers, based on the selection criteria of convenience sampling (Marshall, 1996). The individuals were found across five sources, comprising winners of two international design competitions, and authors represented in three Portuguese design exhibitions. From the study population of 31 individuals, ten were available for participation (32%). This is an acceptable number as, according to Macefield (2009), 10–12 participants is a sensible baseline range for a study to produce a relatively high percentage (82%) of qualitative problem discovery and descriptive statistics findings.

The participants were aged 26–56 (mean of 39), eight males, and two females. All participants were graduated, four had a Master degree and two had a PhD degree. The participants comprised different work environments: studio (four), freelance (one), industry (one), agency (one), university (two), and a technology company (one). One designer had more than 3 years of experience and nine had more than 5 years of experience.

Two participants did not have any experience in 3D CAD applications. The remaining eight used software mostly in development (25%), detail (24%) and production (24%) phases, and less in the early concept (19%) and research phases (8%). The design tasks accomplished with the programs are mainly 3D modeling (19%), rendering and ideation (13% each), and analysis, digital fabrication and communication (11% each).

Results

The results in this section will be presented in the format of (mean/SD).

Some usability metrics were taken from the user evaluation. The participants evaluated the easiness in accomplishing the tasks in a five-point scale, being task 3 the easiest (4.3/0.67), followed by task 1 (3.8/0.79), task 4 (3.4/1.07), and task 2 (3.0/0.71). Task 2 required the longest time (0:19:57/0:07:14), followed by task 1 (0:16:36/0:06:22), task 4 (0:13:11/0:07:41), and task 3 (0:03:47/0:00:42).

The designs produced in task 4 are presented in Figure 11. Different goals were considered by the participants: one reproduced an existing chair; three redesigned existing chairs; two developed a pre-conceived idea; and four generated designs without a previous specific goal.

Participants, in general, revealed a positive overall appreciation of the ChairDNA usability (Fig. 12, left). The results are, from best to worst: ease of learning (4.1/0.74), overall coherence of the tool (4.0/0.47), ease of use (3.7/0.48), use experience (3.7/0.82), the efficiency of the response (2.9/1.29), and flexibility to adapt to the user needs (2.9/1.29). Moreover, the inclusion of graphical explanations of the rules in the interface was highly recommended (4.6/0.70).

Despite a large SD, on average the participants provided a good appreciation of the utility of ChairDNA as a design tool (Fig. 12, right). On the one hand, they claimed that ChairDNA limited decision-making (3.5/1.08), but on the other hand, the tool also allowed the generation of solutions that they have not considered at the beginning (3.2/1.03). They revealed a median satisfaction about the achieved solution (2.9/0.88), and considered adequate the tool's generation logic (3.4/1.17). Few participants considered that they learned something about the shape and structure of the chairs (2.1/0.99). The utility of ChairDNA in the design process was positively evaluated (3.7/0.95), and the majority of the participants stated that they would use the tool in future projects (3.7/1.06).

In what concerns the usage per design phase, participants highlighted the concept phase (38%) and the earlier research phase (31%), followed by the development phase (19%), and lastly the detail and production phases (6% each).

The overall satisfaction with ChairDNA was positive (3.6/0.7). ChairDNA strengths and weaknesses pointed out by the participants are summarized below:

(1) *Visualization*: overall, the quick visualization of the design during the generation process, in real size and proportions, was considered a benefit for 60% of the users. One participant mentioned the risk of seeing the model as an end product.
(2) *Generation*: participants highlighted the ability of ChairDNA to generate designs quickly and effectively, and to freely explore the solution space. The parametric aspect, together with the restriction to anthropometric standards, was considered very helpful. Participants did not agree on the benefit of

https://doi.org/10.1017/S0890060417000610 Published online by Cambridge University Press

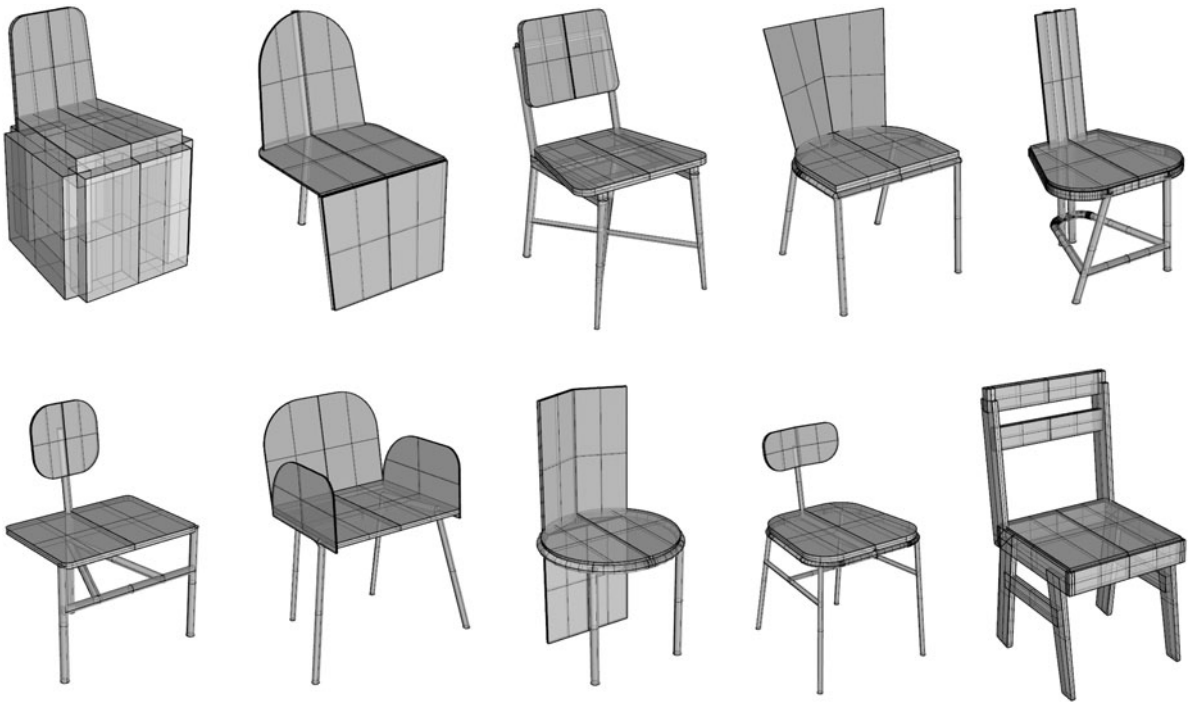


Fig. 11. Chair solutions produced by design practitioners with the ChairDNA tool.

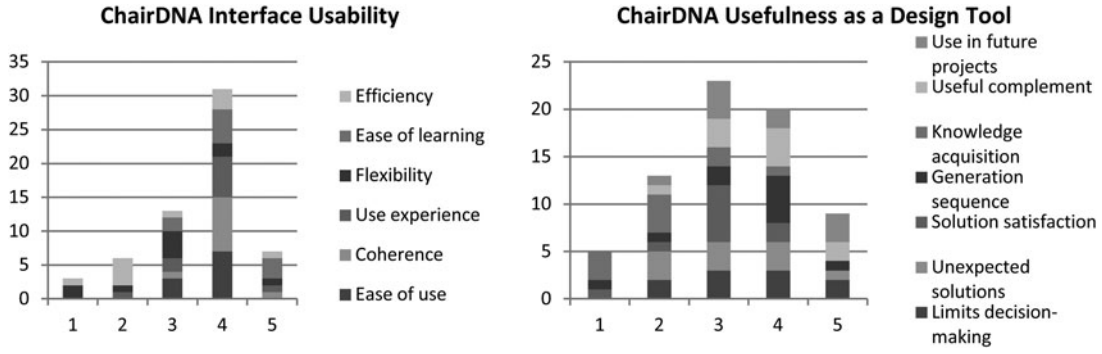


Fig. 12. Major results of the user experience with ChairDNA. The horizontal axis represents the Likert scale, while the vertical axis represents the number of people that selected each element of the scale.

using existing chairs as an inspiration; both in the case of templates (3.5/1.43) as well as of random designs (2.8/1.40).

The participants provided helpful insights for future versions of the program. They suggested including analysis tasks, such as structural analysis and movement simulations. The inclusion of other types of seat furniture, for example, asymmetric chairs and lounge chairs, was considered a key issue for the designers to purchase ChairDNA. They also requested additional features such as curves, libraries of joints and materials, and they highly recommended improving the interface, namely to allow the user to directly manipulate the chair shape.

Discussion

In this section, we discuss (1) the methodology for the implementation of set grammars and (2) the user evaluation of ChairDNA.

Regarding the methodology for implementing set grammars, it is important to acknowledge that SGs operate as a blueprint for the design process, and despite the implementation difficulties (Gips, 1999), they are recognized as excellent intellectual devices for explaining the design process (Knight, 1999). Moreover, for designers that do not have extensive programming experience, SGs are easier to understand and modify than the programs typically needed for implementing a design tool. To this end, one important contribution of our work is a methodology that guides and simplifies the creation of design tools that are based on set grammars.

As mentioned in the Introduction section, emergence is considered an important property of SGs. However, in the case of the MCG presented in this paper, we did not find it useful as, on one hand, supporting emergence requires more complex implementations that are also less efficient, and on the other hand, we were looking for a more deterministic design process

https://doi.org/10.1017/S0890060417000610 Published online by Cambridge University Press

based on an ontology of chairs, which is more difficult to achieve in the presence of emergence. This does not mean that the designs produced by the MCG become predictable, as there is still considerable room for the unexpected by selecting different rules and changing rule parameters, as demonstrated by the examples presented in Figures 10 and 11.

In order to fulfill the usability requirement, we decided to implement our grammar in a way that prevents multiple outcomes from each rule application. To this end, for every grammar rule, its application generates only one new state, thus avoiding ambiguity. This can be seen as an advantage because it makes it easier to move forward in the design task, or as a disadvantage, because it limits alternatives. However, it should be noted that implementations supporting rules with many different outcomes tend to produce a combinatorial explosion of alternatives that make it difficult for the user to select the next state to explore. This is also one of the reasons why we decided to develop a specific grammar implementation instead of relying on existing SG interpreters.

During the implementation of ChairDNA, despite the efforts employed in the development of the MCG, we found that it still contained rules that were too vague and imprecise to be directly usable by a computer. In fact, one important side effect of the creation of an SG implementation is the considerable improvement it causes in the grammar rules as, in order to have a correct implementation, the grammar needs to become much more consistent and detailed. It is our belief that most SGs, in order to be implemented in a computer, must be redefined in a much more stringent form. As this redefinition is a complex and time-consuming task that requires considerable human ingenuity, it is rarely done, which partially explains why there are few digital tools based on the implementation of SGs (Chase, 2005).

In principle, we would prefer that the implemented grammar faithfully follows the original grammar. However, due to the fact that the user of the implementation can stop the generation process at any time, independently of applying the termination rule of the grammar, it is possible for him to accept designs that are in an intermediate state. We can handle this situation by informing the user that the generation process is not yet complete. However, in the user evaluation, this situation was considered an advantage, and thus, we decided that the user should have the final word regarding the design. This made it possible to generate stools that were not possible in the MCG.

In this paper, we focused on the applicability of ChairDNA in a real-life design scenario, and to this end, the tool was evaluated by design practitioners. It is important to mention that these practitioners did not have any knowledge of SGs, although some were familiar with parametric modeling. This being said, they considered the generation logic adequate and praised the parametric aspect, with one exception that mentioned the inadequacy of the chair fragmentation into parts. We also verified that inexperienced users of 3D CAD applications could effectively use the program.

Participants addressed the task of designing a multipurpose chair in quite different ways: by free exploration, by developing a pre-established idea, and by redesign. Although the results present a wide diversity of chair types, users also found some limitations, as they expected a higher level of shape detail, the ability to reproduce curves, and to choose the alignment of solid parts.

The results demonstrated a positive evaluation of the usability. Participants found ChairDNA easy to learn and use, and coherent as a whole. On the negative side, they considered that the prototype was not very efficient (which was more evident as the parts were being added) and that it could be more user-friendly.

The participants considered ChairDNA especially useful for the concept phase of the chair design process, although it was also suggested that it could be used in all the other phases. It allowed a quick generation and visualization of designs, the generation of unexpected solutions, and an unrestricted exploration of the solution space while always obeying correct anthropometric standards. Allowing "impossible" designs was considered an added value for creativity. On the other hand, some participants were afraid that, by being based on existing designs, the program might restrict the solution space. Participants also highlighted its potential applicability for pedagogical and industrial purposes.

There was no consensus regarding the usefulness of templates and random designs. On the positive side, templates were considered useful for redesign processes, and the random command useful as a creative unlocker. On the negative side, the participants considered that the use of existing designs was not advantageous when they had a well-defined idea of what they want to generate, and that the random command could produce odd designs (e.g. Fig. 10, designs 7 and 8). Participants also suggested that ChairDNA should support templates for independent parts and combination of templates.

Conclusion

This paper presented the implementation of the MCG in the design tool ChairDNA. The work addressed two shortcomings in the current research of SGs: (1) the lack of implementations of SGs in digital environments, and (2) the lack of practical applications in real-life design scenarios.

The ChairDNA tool contributes to overcoming these shortcomings because: (1) it efficiently replicates the 3D parametric rules of the grammar; and (2) it was considered useful by a group of ten expert designers. To this end, the tool is easy to learn, its use does not require any knowledge of SGs or programming, it allows for the quick generation and edition of designs, and it provides directions that the designer did not think of in the first place, as demonstrated by the experiences with design practitioners.

The contributions of this paper are twofold: (1) a methodology for the implementation of set grammars, which was illustrated with the case of the MCG but can be generalized to other cases; and (2) an evaluation of an actual set grammar implementation that, contrary to many implementations, is oriented for design practitioners. The proposed methodology addresses the concerns presented by Chase (2005) regarding the difficulty in developing useful interfaces.

This work also makes a broader contribution to the artificial intelligence field, by presenting a digital assistant for a design problem that requires creative intelligence. This assistant collaborates with the human designer, by suggesting the design rules that are applicable at each moment, by automatically computing their application, and by validating constraints that govern the design. These features also allow the automatic generation of designs, which made some of the designers consider the assistant as being creative.

In the near future, we plan to take advantage of the evaluation results to make further developments in the tool, including additional rules and parameters that allow for a higher level of detail, and improvements on the GUI and in the composition of templates. More importantly, we plan to combine ChairDNA with a structural analysis tool that provides feedback regarding the stability and strength of the generated chair, and with an

https://doi.org/10.1017/S0890060417000610 Published online by Cambridge University Press

optimization tool that, based on the previous feedback, optimizes the design by making changes in the user-selected parameters. Another planned feature is the ability to export the generated chair for digital fabrication. Further evaluations will address the use of ChairDNA as a customization tool (with end-users) and as an analysis tool (with design historians).

Acknowledgments. This work was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UID/CEC/50021/2013 and PhD grant SFRH/BD/77927/2011.

References

Agarwal M and Cagan J (1998) A blend of different tastes: the language of coffeemakers. *Environment and Planning B: Planning and Design* **25**, 205–226.

Barros M, Duarte JP and Chaparro BM (2015) A grammar-based model for the mass customisation of chairs: modelling the optimisation part. *Nexus Network Journal* **17**(3), 875–898.

Castro e Costa E and Duarte JP (2014) Generic Shape Grammars for Mass Customization of Ceramic Tableware. In: Design Computation and Cognition DCC'14. 2014, pp. 437–454.

Chase S (2005) Generative design tools for novice designers: issues for selection. *Automation in Construction* **14**, 689–698.

Chau HH, Chen X, McKay A and Pennington A (2004) Evaluation of a 3D Shape Grammar Implementation. In Design Computing and Cognition'04. 2004 Dordrecht Kluwer Academic Publishers, pp. 357–376.

Design Museum (2010) *Fifty Chairs That Changed the World. Fifty*. London: Conran Octopus.

Duarte JP (2001) *Customizing Mass Housing: A Discursive Grammar for Siza's Malagueira Houses*. PhD Thesis, Massachusetts Institute of Technology.

Duarte JP and Beirão JN (2011) Towards a methodology for flexible urban design: designing with urban patterns and shape grammars. *Environment and Planning B: Planning and Design* **38**(5), 879–902.

Earl CF (1986) Creating design worlds. *Environment and Planning B: Planning and Design* **13**(2), 177–188.

Eloy S and Duarte JP (2012) A transformation grammar-based methodology for housing rehabilitation. In: Design Computing and Cognition '12. 2012, pp. 301–320.

Fiell C and Fiell P (1994) *Modern Chairs*. Köln: Taschen.

Figueiredo B., Sousa L., Duarte J.P. and Krüger M. (2014) *Alberti Digital on Portuguese Architecture: Shape Grammar Transformations as a Computational Framework to Determine the Influence of Alberti Legacy on Portuguese Renaissance Churches*. Joelho: Revista de Cultura Arquitectonica, pp. 42–51.

Flemming U (1987a) More than the sum of parts: the grammar of Queen Anne houses. *Environment and Planning B: Planning and Design* **14**, 323–350.

Flemming U (1987b) The role of shape grammars in the analysis and creation of designs. In Yehuda E. Kalay (ed.) *Computability of Design. Principles of Computer-Aided Design*. New York, Wiley Interscience. pp. 245–272.

Flemming U (1989) Syntactic Structures in Architecture: Teaching Composition with Computer Assistance. In: CAAD futures Digital Proceedings. 1989, pp. 31–48.

Flemming U (1994) Get with the program: common fallacies in critiques of computer-aided architectural design. *Environment and Planning B: Planning and Design* **21**(7), 106–116.

Garcia S (2016) Classifications of Shape Grammars. In: *Design Computing and Cognition '16*. John Gero. [Online]. Springer, pp. 243–262.

Garcia S and Barros M (2015) A Grammar-Based System for Chair Design: From Generic to Specific Shape Grammars. In: Real Time – Proceedings of the 33rd eCAADe Conference. [Online]. 2015 Vienna University of Technology, pp. 427–436.

Garcia S and Romão L (2015) A Design Tool for Generic Multipurpose Chair Design. In: Gabriela Celani, David Sperling, & Juarez Franco (eds.). Computer-Aided Architectural Design: The Next City – New Technologies and the Future of the Built Environment: 16th International Conference, CAAD Futures 2015, São Paulo, Brazil, July 8–10, 2015. Selected Papers. [Online]. Berlin, Heidelberg, Springer, pp. 600–619.

Gips J (1975) *Shape Grammars and Their Uses: Artificial Perception, Shape Generation and Computer Aesthetics*. Basel: Springer Basel AG.

Gips J (1999) Computer Implementation of Shape Grammars. In: Workshop on Shape Computation. [Online]. 1999 MIT, Cambridge, USA. p.

Grasl T (2012) Transformational Palladians. *Environment and Planning B: Planning and Design* **39**(1), 83–95.

Grasl T and Economou A (2013) From topologies to shapes: parametric shape grammars implemented by graphs. *Environment and Planning B: Planning and Design* **40**, 905–922.

Heisserman J (1994) Generative geometric design. *IEEE Computer Graphics and Applications* **14**, 37–45.

Heisserman J, Mattikalli R and Callahan S (2004) A grammatical approach to design generation and its application to aircraft systems. In: Proceedings of the Generative CAD Systems Symposium '04. 2004 Pittsburgh, PA. p.

Hsiao S-W and Chen C-H (1997) A semantic and shape grammar based approach for product design. *Design Studies* **18**, 275–296.

Jowers I and Earl C (2011) The implementation of curved shape grammars. *Environment and Planning B: Planning and Design* **38**(4), 616–635.

Kirsch JL and Kirsch RA (1986) The structure of paintings: formal grammar and design. *Environment and Planning B: Planning and Design* **13**, 163–176.

Knight T (1999) Applications in Architectural Design, and Education and Practice. Report for the NSF/MIT Workshop on Shape Computation.

Knight T (1989) Transformations of De Stijl art: the paintings of Georges Vantongerloo and Fritz Glarner. *Environment and Planning B: Planning and Design* **16**, 51–98.

Knight T (2003) Computing with emergence. *Environment and Planning B: Planning and Design* **30**, 125–155.

Krishnamurti R (1980) The arithmetic of shapes. *Environment and Planning B* **7**, 463–484.

Lawson B (2008) How Designers Think. Oxford: Architectural Press.

Li AIk (2001) Teaching Style Grammatically, with an Example From Chinese Architecture. The Proceedings of Mathematics & Design 2011. pp. 270–277.

Li AIk (2002) A prototype interactive simulated shape grammar. In: Connecting the Real and the Virtual – design education: 20th eCAADe Conference Proceedings. 2002, pp. 314–317.

Li AIk, Chen L, Wang Y and Chau HH (2009) Editing Shapes in a Prototype: Two- and Three-dimensional Shape Grammar Environment. In: Computation: The New Realm of Architectural Design: 27th eCAADe Conference Proceedings. 2009 Istanbul, Turkey: Istanbul Technical University, Faculty of Architecture, pp. 243–250.

Lindenmayer A (1968) Mathematical models for cellular interactions in development. *Parts I and II. Journal of Theoretical Biology* **18**, 280–315.

Lopes J and Leitão A (2011) Portable Generative Design for CAD Applications. In: Integration through Computation. 2011 Calgary, Canada, pp. 196–203.

Macefield R (2009) How to specify the participant group size for usability studies: a practitioner's guide. *Journal of Usability Studies* **5**(1), 34–45.

Mamoli M (2015) Library Grammar: A Shape Grammar for the Reconstruction of Fragmentary Ancient Greek and Roman Libraries. In: Proceedings of the 33rd eCAADe Conference. 2015 Vienna University of Technology, Vienna, Austria, pp. 463–470.

Marshall MN (1996) Sampling for qualitative research. *Family Practice* **13**(6), 522–526.

McCormac J, Bown O, Dorin A, McCabe J, Monro G and Whitelaw M (2014) Ten questions concerning generative computer art. *Leonardo* **47** (2), 135–141.

McGill M and Knight T (2004) Designing Design-Mediating Software: The Development of Shaper2D. In: 22nd eCAADe Conference Proceedings. 2004, pp. 119–127.

McKay A, Chase S, Shea K and Chau HH (2012) Spatial grammar implementation: from theory to useable software. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **26**, 143–159.

Müller P, Wonka P, Haegler S, Ulmer A and Van Gool L (2006) Procedural modeling of buildings. *ACM Transactions on Graphics (TOG) – Proceedings of ACM SIGGRAPH 2006* **25**(3), 614–623.

https://doi.org/10.1017/S0890060417000610 Published online by Cambridge University Press

Orsborn S, Cagan J, Pawlicki R and Smith RC (2006) Creating cross-over vehicles: defining and combining vehicle classes using shape grammars. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **20**, 217–246.

Pugliese M and Cagan J (2002) Capturing a rebel: modeling the Harley-Davidson brand through a motorcycle shape grammar. *Research in Engineering Design* **13**, 139–156.

Saul G, Lau M, Mitani J and Igarashi T (2011) SketchChair: An All-in-one Chair Design System for End-users. In: Proceedings of the 5th International Conference on Tangible and Embedded Interaction 2011. 2011 Funchal, Madeira, Portugal, pp. 73–80.

Stiny G (1980) Kindergarten grammars: designing with Frobel's building gifts. *Environment and Planning B: Planning and Design* **7**, 409–462.

Stiny G (1982) Spatial relations and grammars. *Environment and Planning B: Planning and Design* **9**, 113–114.

Stiny G and Mitchell W (1978) The Palladian grammar. *Environment and Planning B: Planning and Design* **5**, 5–18.

Stouffs R and Wieringa M (2006) The generation of Chinese ice-ray lattice structures for 3D façade design. In: Conference Proceedings of the Joint International Conference on Construction Culture, Innovation and Management (CCIM). 2006 BUiD, Dubai. pp. 416–424.

Strobbe T, Eloy S, Pauwels P, Verstraeten R, De Meyer R and Van Campenhout J (2016) A graph-theoretic implementation of the Rabo-de-Bacalhau transformation grammar. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **30**, 138–158.

Strobbe T, Pauwels P, Verstraeten R, De Meyer R and Van Campenhout J (2015) Toward a visual approach in the exploration of shape grammars. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **29**, 503–521.

Tapia M (1999) A visual implementation of a shape grammar system. *Environment and Planning B: Planning and Design* **26**(1), 59–73.

Theodoros D (2013) Some Notes on the Incompleteness Theorem and Shape Grammars. In: Global Design and Local Materialization. Communications in Computer and Information Science, pp. 368–375.

Tilley A and Henry Dreyfuss Associates (2002) *The Measure of Man and Woman: Human Factors in Design*. New York, John Wiley & Sons.

Sara Garcia is a designer and a doctoral researcher at the Faculty of Architecture of the University of Lisbon (FA/UL). She graduated in Design and holds a Master degree in Product Design, both from FA/UL. Her research interests include computational design approaches, particularly in the domain of shape analysis and generation, and grammar-based design tools for product design.

António Menezes Leitão has a BSc in Mechanical Engineering, an MSc in Electronics Engineering, and a PhD in Computer Science and Engineering, all from Instituto Superior Técnico (IST) of the University of Lisbon. Currently, he is an Assistant Professor at the same university, Scientific Coordinator of the Software Engineering Group at INESC-ID, and Coordinator of the Architecture and Computation Group, teaching, lecturing, and researching on bringing together the fields of computer science and architecture.

https://doi.org/10.1017/S0890060417000610 Published online by Cambridge University Press