Artificial Intelligence for Engineering Design, Analysis and Manufacturing

cambridge.org/aie

IM-sgi: an interface model for shape grammar implementations

Joana Tching[1], Joaquim Reis[1] and Alexandra Paio[2]

[1]Computer Science and Information Technology, University Institute of Lisbon (ISCTE-IUL), ISTAR-IUL, Lisbon, Portugal and [2]Department of Architecture, University Institute of Lisbon (ISCTE-IUL), ISTAR-IUL, Vitruvius FABLAB-IUL, Lisbon, Portugal

Research Article

Cite this article: Tching J, Reis J, Paio A (2019). IM-sgi: an interface model for shape grammar implementations. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **33**, 24–39. https://doi.org/10.1017/S0890060417000695

Received: 7 October 2016
Revised: 14 December 2017
Accepted: 19 December 2017

Key words:
Computational creativity; creative process; ergonomic criteria; interface model; shape grammar implementations; shape grammars

Author for correspondence:
Joana Tching, E-mail: joana.tching@outlook.com

Abstract

Information technologies are a driving force for progress in the design field, allowing new modes of creativity. However, most of the existing computational design tools are focused on the latest stages of the design process and especially directed to drafting operations. Conceptual design tools that support the designer in the creative and inventive early stages of the design project are still in their early development. Shape grammars (SG) were introduced by George Stiny in the 1970s, allowing the generation of designs according to a set of predefined rules. SG computational implementations have the potential to answer the need for tools that can assist designers, architects, and artists in the creative process, offering design alternatives, stimulating new ideas and encouraging the search for new design generation processes. Acknowledging this potential, a user-friendly interface seems essential for the adoption of these tools. Taking Scott Chase's interaction model as background, the aim of the present investigation is to define guidelines and begin to design a graphical-user interface for SG implementations. Inspection methods of human–computer interaction (HCI) were used to analyze existing SG implementations and understand usability issues. Subsequently, HCI ergonomic criteria for interface evaluation were adapted to establish guidelines for the design of an SG implementation interface, called IM-sqi. These guidelines take into account different user groups, adjustable interaction modes for each user group, and the nature of each task performed by the user.

Introduction

This paper presents IM-sgi – model of interface for shape grammar(s) (SG) implementations. The model illustrates the existing lack in the use of SG for architectural and design projects. Since the presentation of SG (Stiny & Gips, 1975), there are a considerable number of computational implementations, but few actually assist in the use of SG in the design field (Yue & Krishnamurti, 2014). SG is defined by a vocabulary of shapes and a set of rules that specify how to combine such shapes by recreating spatial relations defined between them (Stiny, 1977). These are similar to phrase structure grammars, with an alphabet of shapes that generate one-dimensional (1D) to *n*-dimensional shapes. SG use algorithmic processes for the representation and computation of shapes that organize specific knowledge for the exploration of designs (Krishnamurti, 1980).

The use of SG with computer applications enables the designer to take the full advantage of (1) synthesis and analysis of styles in design/architecture/art and (2) the creation of new forms. Several models of interaction with the user have been developed, as the ones developed by Scott Chase (Chase, 2002) and Haldane Liew (Liew, 2002) [see analysis in (Tching et al., 2016)]. These models seem to lack guidelines for a clear and efficient interface for SG implementations that translate the objectives of the existing models of interaction. The interface of SG implementation should take into account architects/designers who are trained and comfortable using CAD software systems – an interface well adopted and stabilized.

The complexity of CAD systems and the types of tasks associated with them require a high-quality interface. In this sense, the best way to ensure that SG implementation is well accepted and understood by an architect or designer is to include the basics of the user. IM-sgi aims to provide computing ergonomics and suitability to architects to work with SG implementations. It uses human–computer interaction (HCI) methods, with a focus on those for interactive design. Thus, the main focus is on user experience while using an application, addressing the usability problems in the process of design interface, with emphasis on cognitive and experiential factors.

In resume, this paper has two sections. The first addresses the need to move toward the use of SG by architects and designers, and the second presents IM-sgi, a model of user interface (UI) developed according to the principles of HCI. IM-sgi proposes that a well-focused

© Cambridge University Press 2018

CAMBRIDGE UNIVERSITY PRESS

interface will allow designers and architects to clearly understand SG implementations and their potential use in daily design practice.

Toward an effective use of SG

Digital design and CAD applications are widely used today in architectural projects and design. As far as creative areas are concerned, computational applications have meant much faster and more effective processes, as compared with those previously done. They have also allowed the production of more complex and ambitious projects, offering new ways of analysis, control, and representation that would not otherwise be available to designers, as more time and unaffordable resources would be required. Architectural projects, in contrast to other artistic areas, develop in different phases that can be described not only as a need to solve a large number of issues but also as the fulfillment of rules and constraints, whether they are legal, environmental, economic, or formal. It is the solution of all these issues that the final project comes out, and the architect shows his/her creativity, combining all the items in an aesthetic and functional product. Dividing the project into its elementary parts, we see that the architect elects, consciously or intuitively, a set of rules and makes choices that impact the final work (Tching et al., 2013).

Overall, the architect's intentions are guided by rules that are imposed by technical and legal needs of the project and by the artist's aesthetic and creative intentions.

It is common practice to use computational applications that reproduce the architect's manual design, and other technical aspects of a project (such as automatic measurements, thermal simulations, 3D visualization, etc.). However, to support earlier conceptual project stages, there is a demand for new CAD solutions that do not focus on the development of shapes and structures already well defined (McKay et al., 2012). It will be possible for the designer to use SG implementations in their common practice with new systems that need to support the designer's creative process, offering non-obvious design alternatives that were not originally defined by the designer and can be used as a solution or as a creative stimulation for new ideas. The use of SG implementations can lead to the optimization of ideas, taking advantage not only of CAD but also computational creativity (CC). The solutions given by SG implementation reflect a creative process of synergy between SG and the designer (McKay et al., 2009). This can potentially improve, provide uniqueness, or at least an alternative from what the architect/designer would accomplish without the use of SG.

The communication between the designer and the computational generating system is defined by the computational interface. In the creation of a computer application, the interface is a time-consuming task, and many times the programmer, or program designer, is not familiar with the true needs and limitations of the end-user. A model that specifies categories of SG users and their objectives, and that clarifies how each type of user will make use of SG and what barriers of communication need to be addressed will increase the possibilities of the success of computer SG implementation use. So, how to achieve good usability? According to Myers, there are three main points. First, know the users and their tasks through the analysis of these and contextualized investigations; second, ensure the adequacy of the design through prototypes tested by users with a participatory and iterative design; third, make the final product usable and efficient through the use of the interface, analyzing it through various methods, heuristics, and others (Myers, 2008).

The design of the UI is a creative process and often designers have difficulty thinking like end-users. The usability is linked to learning, efficiency, productivity, ease of memorization, satisfaction, and no errors. Good usability is important, as it reflects the notion of quality of the user's system. Good usability allows beginners to become effective more quickly, experts more efficient, and a reduction in errors. The true needs have been identified so that the application will be successful on the market (Myers, 2008).

IM-sgi: interface model for SG implementations

In section "Toward an effective use of SG", we showed that the interface is very important for the success of SG implementations, as it is the means of communication between the computational tool and the designer's goal. For that reason, IM-sgi is based on the analysis of the interaction model developed by Scott Chase (Chase, 2002).

Ultimately, a computer system is created for the function you want to perform, and the functionality of a system is defined by the set of tasks for its user (Karray et al., 2008). As the user meets his/her objectives when using the system efficiently, the value of the application is visible.

According to (Lewis & Rieman, 1994), the process to be followed when creating a new interface that focuses on the success of the task performance has the following steps:

1. Analysis of the target users and the tasks they will perform with the application;
2. Selection of the main tasks the application must perform;
3. Analysis of existing interfaces of similar applications;
4. Initial definition of the interface;
5. Evaluation of the interface proposed without users (by the designer of the interface or experts in the area);
6. Prototyping;
7. User test;
8. Iteration of prototype corrections;
9. Final development of the interface.

This process for a good interface definition requires a significant amount of time, but time is often scarce when a new application is being developed. This might be one of the reasons why existing SG applications, which require such complex development of algorithms and programming time, still have rudimentary interfaces, as analyzed in our previous work (Tching et al., 2016). This acknowledgment is the basis of our IM-sgi proposal. Together with IM-sgi, an interface model for SG implementations, we collected information for stages 1–3 and gathered the criteria needed for the development of stages 4–6, thus guiding the creation of SG interfaces with high levels of performance in terms of usability. The compilation of all this information in an interface model for SG implementation allowed us to develop and focus on the development of the application function. We then rapidly created an interface prototype following IM-sgi criteria that has all the needed considerations to be adjusted to SG users, allowing them to perform the tasks needed to work with SG. Following the Lewis and Rieman's process of nine stages listed above (Lewis & Rieman, 1994), IM-sgi definition of stages 1–3 has already been developed and presented in a previous article (Tching et al., 2016).

https://doi.org/10.1017/S0890060417000695 Published online by Cambridge University Press

In this paper, we present the criteria we developed based on ergonomic and usability standards. We applied these to SG tasks that can define good interfaces for SG applications. The vision here is that, with the correct interface, SG could be adopted by architects/designers in their project practice who can then take SG to the professional field, instead of staying only in the educational one (Tching et al., 2013). The first step was to understand the types of users that would be using SG and what tasks they would need to be able to perform. For this definition, Scott Chase developed an important model of interaction for SG systems (Chase, 2002). It addresses the types of SG users and the different ways they can relate to the SG applications, pointing out different scenarios that range from a higher control for the user of the system to a higher automation and passivity of the user. Studying these control scenarios, with the aim for architects and designers to be able to use SG in creative projects, we defined three groups of users for the IM-sgi, according to their level of control of SG and their needs for manipulation: students, designers/artists, and SG experts (Tching et al., 2016).

The general criteria of IM-sgi consider these users in a global form. It then defines a series of specific criteria for the three levels of expertise. Thus, the same interface can evolve according to the progress of the user expertise with the SG implementation, but keep the same ergonomic criteria (EC) and usability performance. To develop stage 2 (selection of the tasks the application must perform) and stage 3 (analyzing existing similar applications), the use of an inspection method in the field of HCI was applied: the cognitive walkthrough (CW). This method allowed the clear definition of the tasks that SG users should be able to perform if these tasks are successfully achieved in a group of existing SG implementations that we selected to be tested. The authors presented the full development of the CW in (Tching et al., 2016).

The conclusions of the CW allowed us to understand the interaction fragilities common in SG implementations that should be addressed by IM-sgi. The main one is the importance of the use of a graphical-user interface (GUI) that allows the permanent visualization of the shapes manipulated. As SG are mainly visual, as are most of the strategies used for early stages of creative projects, the graphic illustration of SG elements and results are essential to our investigation. The clear guidance to the user through the process of shape and rule creation must also be guaranteed so that SG can be created with success. The interface will then be self-explanatory and allows the user to create SG, focusing on the tasks to be performed, rather than concentrating on interpreting how windows, menus, and commands work.

IM-sgi criteria

IM-sgi defines the criteria that the interface for SG implementations should satisfy. Why should we have a specific model for these interfaces and not just follow general guidelines? Because it is not possible to define generally what a good interface is. This is dependent on the tasks to be performed and the users who will manipulate the interface. As stated before, usability measures the quality of the interface elements related to their usefulness. According to (Nielsen, 1994), usability is a quality attribute that allows the understanding of how easy UI is to use. To continue developing IM-sgi, after understanding the needs of the users, the next step is to define the criteria that will guide the interface to good usability. To achieve this goal, we gathered existing criteria to evaluate usability and ergonomics of the interface and interpreted them as keys that will be clearly defined by the model of interface. We then applied it to the tasks the user will need to perform in an SG implementation.

Since the development of HCI, countless authors have developed several general design guides, sets of guidelines, checklists, standards, and heuristics in order to help achieve good UI (Bastien & Scapin, 1995). Some examples are standards such as ANSI, DIN, and ISO, design guides from (Scapin, 1986; Schneiderman, 1987; Ravden, 1988; Brown, 1998), sets of guidelines from (Bodart & Vanderdonckt, 1995) and (Smith, 1986), style guides from (IBM, 1989) and (Apple, 1992), and heuristics like those from (Molich & Nielsen, 1990; Scapin, 1990; Bastien & Scapin, 1993; Nielsen, 1994). Despite the specificity of each set of guidelines, the main aspects of usability are learnability, memorability, low ratio error, efficiency, and satisfaction. This means that HCI has been sharing a path with cognitive theories that also focus on reducing the memory load from the user (Hollender et al., 2010). Also, both areas acknowledge the importance of the learner characteristics, showing that, to a certain extent, cognitive concepts and HCI approaches have been integrated.

Bastien and Scapin recognized the limitations and difficulties in evaluating an interface, using any HCI approach individually. So they proposed a set of EC for interface evaluation (Bastien & Scapin, 1993) that intends to be a compilation of the several existing guidelines, making their criteria very specific and oriented to avoid misinterpretations. The term EC shows that the authors were interested in the adequacy of the criteria to the users' characteristics, behaviors, and needs. Bastien and Scapin also developed a consistent model of testing these criteria to validate their adequacy, since the existing guidelines and heuristics from other authors have not applied them (Bastien & Scapin, 1995). These criteria have also been developed for use in different types of interfaces, such as virtual reality environments (Bach & Scapin, 2003).

As the purpose of IM-sgi is to gather the correct criteria for a very specific type of interface (SG implementation interface), Bastien and Scapin's EC was the basis to define IM-sgi criteria, together with the knowledge gained with a CW made with existing SG implementations and analysis according to the ISO standards (Tching et al., 2016).

By converting these evaluation criteria to define successful interfaces for SG implementations, IM-sgi demonstrates how each criterion should be addressed for a specific SG task so that the steps needed are successfully communicated by the interface. So, instead of general guidelines, IM-sgi clearly defines how the interface should communicate with the user in each task to be performed while working with SG. Bastien and Scapin defined 18 EC that are organized into eight main sections. For the definition of IM-sgi, Bastien and Scapin's criteria have been interpreted as guidelines and transformed into IM-sgi definition criteria. IM-sgi criteria, although based on Bastien and Scapin's EC, are presented in a different order. [For IM-sgi guidelines to the SG implementation developer in the interface definition, see (Tching et al., 2016).] IM-sgi criteria create specific definitions for how the interface should look to guide the user through the main tasks to be performed while working with SG (see Table 1 below). These five tasks were defined for and subsequently validated by the CW of a previous investigation (Tching et al., 2016). In the previous analysis, the main tasks the users will perform were defined according to a hierarchy of tasks that the user

https://doi.org/10.1017/S0890060417000695 Published online by Cambridge University Press

Table 1. Compatibility

General	SG-COMP	A) The users considered by IM-sgi are professionals in the artistic field, mainly architects and designers. Considering these users, three types were distinguished: STUDENTS – students of architecture or design that shall be able to use the SG implementation for educational purposes, exploring, and learning SG with the use of the tool ARTISTS/DESIGNERS – artists, architects, and designers that shall be able to use the SG implementation as a tool to develop their projects EXPERTS – SG experts that will be able to program the SG and the SG implementation in order to allow it to create newly desired results B) As the target users are characteristically users of CAD software, formats, tools, labels, messages, or instructions shall follow conventional CAD software that are perceived by users as being familiar. CAD software has highly sophisticated interfaces that shall be a source for these definitions C) A graphical user interface (GUI) shall be used and a permanent visualization of the results of the actions taken shall exist. All saved shapes, rules. or SG results shall have a graphic representation D) Units of measurement shall be metric and/or imperial
Task 1 Shape creation	SGSC-COMP	A) An Initial Shape shall be able to be created by drawing tools and with clear unit measurements as in conventional CAD software, by drawing in a window with zoom and pan abilities B) According to professional needs, importation of shapes to be used or importation of background images to serve as a guide to the drawing shall be available C) The use of a command line for drawing shall be available for most experienced users
Task 2 Rule creation	SGRC-COMP	A) Rules shall be able to be created by drawing tools and with clear unit measurements as in conventional CAD software, by drawing in a window with zoom and pan abilities B) According to professional needs, importation of shapes to be used as rules or importation of background images to serve as guides to the drawing shall be available C) The use of a command line for drawing shall be available for most experienced users
Task 3 SG application	SGAPP-COMP	A) SG results shall be able to be applied and visualized in a graphic window with zoom and pan abilities B) According to professional needs, importation of background images to merge and work with the SG results shall be available C) The exportation of SG results in compatible formats with conventional CAD software shall exist, allowing the use of SG for professional purposes
Task 4 SG manipulation	SGMAN-COMP	A) SG results shall be able to be manipulated, cycling through different alternatives and changing iterations number B) The manipulation of the results could be done by directly manipulating the drawing according to creativity purposes in the implementation itself
Task 5 SG alteration	SGALT-COMP	A) Users shall be able to manipulate any previous state of the SG, from Initial Shape to its Rules and results B) Any state and alteration shall be able to be saved or exported to be used later or to be developed in CAD software

will take to create and use an SG. These tasks are described in Table 1.

In summary, each IM-sgi criterion is divided into six groups of guidelines, one group of general guidelines and one group of guidelines related to each of the above tasks. The three types of users and interaction modes that the interface must consider are:

- Students–GUI mainly with drawing tools available and with immediate visualization of any action made. This interface should be prepared for a reduced control of the user over the implementation, which will give SG results in a more automated way. This will be considered a Beginner Mode.
- Artists/designers–GUI with drawing, editing, and exporting tools that follow conventional CAD software, allowing the user to have a good control of the implementation. They can then use it with possible integration with other CAD software so that SG can be defined to solve design projects, and its results can be used for further design project development/detailing. Command lines can be used to help the user clearly define the Rules of SG in order to use SG for specific problem solving (Tching et al., 2013). The user control should be higher and the degree of automation should be less than in the Beginner Mode, as this is an Intermediate Mode.
- Experts in SG–GUI combined with programming tools that allow the user to have full control over the implementation. They can even change its code so that it can create SG in new or different ways. Command lines or even windows dedicated to programming can be used. The user shall have full control over the application with a reduced degree of automation, as this is an Expert Mode.

Following the above descriptions of Bastien and Scapin's EC, target users, and task types, IM-sgi is presented here with the description of the original EC and the adaptation for IM-sgi criteria in the tables. It is important to notice that IM-sgi criteria were defined so that they can be applied to any type of SG implementation, although not addressing directly all the features that the implementation shall have. The idealized general SG

https://doi.org/10.1017/S0890060417000695 Published online by Cambridge University Press

implementation characterization made by (McKay, et al., 2012) points to seven requirements (form, orientation of shapes, semantics, definition interface, usability by designers/intuitive UI, automatic subshape detection, clear interpretation of resulting designs). Our model focuses on the definition interface and usability by designers with guidelines for these specific requirements. The criteria in IM-sgi can address different types of generation processes that can be applied to the several actions that the interface must allow for a specific SG implementation. Taking the generation process presented in (Trescak et al., 2012), several input parameters can be introduced to manipulate an SG, for instance, choosing the number of iterations, applying step-by-step iterations, and generating a list of possible next shapes to be chosen and the use of markers. For expert users, our defined criteria can provide different ways of defining rules, either by graphic shape rules or mathematical rule schemata (Economou & Kotsopoulos, 2014).

In summary, we developed IM-sgi with a total of 259 criteria, divided according to the 18 EC that Bastien and Scapin developed (Bastien & Scapin, 1993) described in Tables 1–18.3

These 18 EC groups provide very specific guidelines for an interface creation, explicitly defining how the interface communicates with the user through the tasks he/she will be performing for good usability. Every EC leads to the definition of six groups of criteria, one group with general guidelines and five groups according to the tasks previously defined. All criteria focus on assuring that the interface clearly guides the user on the actions he/she should perform to do the tasks and also guides him through the correct order of steps to successfully create and manipulate SG. The next stage of this investigation will be to use the criteria to define interfaces for SG implementations. According to [(Lewis & Rieman, 1994) referred to in section "IM-sgi: interface model for SG implementations"], we now have steps 4 and 5 completed and presented in a paper under development.

Table 2. Adaptability – users' experience management

General	SG-USEX	(A) The SG interface shall be prepared to adjust to three types of users, according to the definition made in this investigation. This adaptation can be achieved giving the user the chance to choose the level of expertise when opening the application (B) A Beginner Mode shall be available, focused for students of SG, with high-automated control by the implementation. This mode shall have mainly drawing tools available and with immediate visualization of any action made (C) An Intermediate Mode shall be available, focused on artists/designers, allowing the users to have a good control of the implementation and use it with possible integration with other CAD software, so that SG can be defined to solve design projects and its results used for further design project development/detailing (D) An Expert Mode shall be available, focused on SG experts allowing the user to have full control over the implementation and even changing its code so that it can create SG in new or different ways. The user shall have full control over the application with lower or no levels of automation
Task 1 Shape creation	SGSC-USEX	(A) In the Beginner Mode, an Initial Shape shall be created using predefined shapes, so that the automation is higher. Drag-and-drop actions are preferable (B) In the Intermediate Mode, an Initial Shape shall be created recurring in drawing tools that resemble conventional CAD software (C) In the Expert Mode, an Initial Shape shall have the possibility to be created by code, ideally in a command line
Task 2 Rule creation	SGRC-USEX	(A) In the Beginner Mode, Rules shall be created using predefined shapes, so that the automation is higher. Drag-and-drop actions are preferable (B) In the Intermediate Mode, Rules shall be created recurring in drawing tools that resemble conventional CAD software. There shall be the possibility to create several rules for one same SG and to be able to visualize them and reorder them. In this mode, Labels to complex Rules shall be available (C) In the Expert Mode, Rules shall have the possibility to be created by code, ideally in a command line. The use of labels and the general organization of the SG rules shall be able to be defined directly by code
Task 3 SG application	SGAPP-USEX	(A) In the Beginner Mode, a predefined iterations number could be applied to give immediate visualization of the SG created while manipulating the Rules. User shall also be able to increase or decrease iterations number with a simple button, without entering values (B) In the Intermediate Mode, the iterations number shall be entered manually (C) In the Expert Mode, the iterations number shall be entered manually or by code, ideally in a command line
Task 4 SG manipulation	SGMAN-USEX	(A) In the Beginner Mode, a small number of alternatives could be seen, preferably by cycling them in the SG window (B) In the Intermediate Mode, a good number of alternatives shall be visualized, preferably in a Drop-down menu or in a new window and dragged to the SG window for substitution (C) In the Expert Mode, the results shall be able to be controlled by code, ideally in a command line, allowing the application of alternatives
Task 5 SG alteration	SGALT-FLEX	(A) The user actions shall follow SGSC-USEX, SGRC-USEX, SGAPP-USEX, and SGMAN-USEX, giving the user the liberty to change any prior definition (B) In all expertise modes, any changes made to the SG shall be visualized immediately

https://doi.org/10.1017/S0890060417000695 Published online by Cambridge University Press

Table 3. Adaptability – flexibility

General	SG-FLEX	(A) SG implementation interface shall take into account conventional CAD software, as the target users are mainly from architecture and design fields, allowing the user to organize menus, toolboxes, windows, or others to resemble the software they are familiar with (B) Users shall be able to use buttons, menus, or commands, or perform the same action, using what suits their habits (C) Users shall be able to choose windows size, accordingly to their drawing or visualization needs
Task 1 Shape creation	SGSC-FLEX	(A) Initial Shape window shall allow resizing, facilitating the drawing actions (B) Drawing commands can be available using drawing tools, using predefined shapes selected or dragged to the window or by a command line (C) In Expert Mode, user shall be able to define Rules prior to Initial Shape drawing instead of following the general sequence of tasks
Task 2 Rule creation	SGRC-FLEX	(A) Rules window can allow resizing, facilitating drawing actions (B) Drawing commands can be available using drawing tools, using predefined shapes selected or dragged to the window or by a command line (C) Label tools shall be available, dragging symbols to the drawing or inserting them with button (D) In Expert Mode, user shall be able to define Rules prior to Initial Shape drawing instead of following the general sequence of tasks
Task 3 SG application	SGAPP-FLEX	(A) SG window shall have a big presence in the screen and allow resizing (B) User shall be able to choose the iterations number by adding the specific number or cycling through predefined numbers. The system can also have a predefined number applied automatically that can be changed
Task 4 SG manipulation	SGMAN-FLEX	(A) The manipulation of alternatives shall be possible according to SGAPP-FLEX (B) User shall be able to choose to see alternatives in a new window or cycle through them directly in the SG window, or Drop-down menu or even textually in a command line
Task 5 SG alteration	SGALT-FLEX	(A) The user choices shall follow SGSC-FLEX, SGRC-FLEX, and SGAPP-FLEX

Table 4. Significance of codes

General	SG-SICO	(A) All window titles shall be clearly related to the tasks and results that will be available (B) Any abbreviations shall be clear and limited to the strictly necessary (C) Menu and button naming shall follow conventional CAD software and be clearly related to the actions (D) Codes to be used in command lines shall be meaningful and allow the clear definition of the actions
Task 1 Shape creation	SGSC-SICO	(A) The term “INITIAL SHAPE” shall be used in the title of the window for this purpose (B) This term shall be used for any message related to Initial Shape definition (C) Drawing tools shall have clear names and follow conventional CAD software
Task 2 Rule creation	SGRC-SICO	(A) The term “RULES” shall be used in the title of the window for this purpose (B) This term shall be used for any message related to Rule definition (C) Drawing tools available for Rule creation shall have the same naming as the ones for Initial Shape drawing
Task 3 SG application	GAPP-SICO	(A) The term “SHAPE GRAMMAR” shall be used in the title of the window for this purpose (B) This term shall be used for any message related to shape grammar application (C) The term “ITERATIONS” shall be used for the data entry field for iterations application
Task 4 SG manipulation	SGMAN-SICO	(A) A term related to “ALTERNATIVES” or “RESULTS” shall be used for menus or windows that allow the choice of shape grammar alternatives
Task 5 SG alteration	SGALT-SICO	Not applicable

https://doi.org/10.1017/S0890060417000695 Published online by Cambridge University Press

Table 5. Consistency

General	SG-CONS	(A) Window titles and formats shall follow the same format and placement in all windows (B) The screen format shall be identical in all expertise modes, adapting itself to the specifications of each one (C) Menu options, buttons, and command lines shall be identical in all different windows and expertise modes (D) All messages shall follow the same format and position on the screen
Task 1 Shape creation	SGSC-CONS	(A) Initial Shape window and tools shall be identical and recognizable in all expertise modes, independently of the additional tools in each one
Task 2 Rule creation	SGRC-S CONS	(A) Rules window and tools shall be identical and recognizable in all expertise modes, independently of the additional tools in each one
Task 3 SG application	SGAPP-CONS	(A) SG window and tools shall be identical and recognizable in all expertise modes, independently of the additional tools in each one
Task 4 SG manipulation	SGMAN-CONS	Not applicable
Task 5 SG alteration	SGALT-CONS	Not applicable

Table 6. Immediate feedback

General	SG-FEED	(A) SG implementation shall have a GUI that allows immediate visualization of created shapes, created rules, and SG results and alternatives (B) A graphical visualization of the created rules, either by drawing or data entry shall be immediately presented (C) SG iterations shall be immediately graphically visualized, along with a number of alternatives (D) Undo and Redo tools shall be available and its results immediately visualized (E) Messages guiding the user shall be delivered.
Task 1 Shape creation	SGSC-FEED	(A) Initial Shape window shall allow the graphic design with drawing tools, similar to conventional CAD software (B) Drawing tools shall be clearly identified and its results immediately visualized when used (C) Any change made to the shapes shall be easily reversible and all the results immediately visualized (D) If a command line exists to define shapes, the results shall be graphically seen in the Initial Shape window
Task 2 Rule creation	SGRC-S FEED	(A) Rules window shall allow the graphic design with drawing tools, similar to conventional CAD software (B) Drawing tools shall be clearly identified and its results immediately visualized when used (C) Any change made to the shapes and rules shall be easily reversible and all the results immediately visualized (D) The system shall allow saving and ordering the rules of the SG and allow the user to easily visualize and manipulate them, preferably in a dedicated Window or Drop-down menu with visualizations of the composed rules (E) If a command line exists to define rules, the results shall be graphically seen in the Rules window
Task 3 SG application	SGAPP-FEED	(A) The result of the number of iterations applied shall be immediately seen in the SG window (B) The system shall show alternatives of the SG with the same iterations, preferably in a dedicated Window or Drop-down menu
Task 4 SG manipulation	SGMAN-FEED	(A) The iterations tool shall allow immediate change of the number, with immediate visualization of the new result (B) The system shall graphically show alternatives of SG with the same number of iterations, preferably in a dedicated Window or Drop-down menu, which can be selected and worked on
Task 5 SG alteration	SGALT-FEED	(A) If changes are made in the Initial Shape, the new results shall be immediately graphically visualized (B) If changes are made in the Rules, the new results shall be immediately graphically visualized (C) Any alteration shall be easily undone or redone and its results visually seen

https://doi.org/10.1017/S0890060417000695 Published online by Cambridge University Press

Table 7. Guidance – grouping and distinction of items by location

General	SG-GDLO	(A) Three main drawing windows must exist, organized by order: 1 – INITIAL SHAPE or equivalent; 2 – RULES or equivalent; 3 – SHAPE GRAMMAR or equivalent They shall be organized linearly, left to right or up to down, or combinations of these, according to reading conventions (B) Any extra windows must be organized according to above reading conventions, next to the related window (C) The menu options to be used in each drawing window must be organized according to the object they apply to and according to the order of commands to be performed
Task 1 Shape creation	SGSC-GDLO	(A) Window for Initial Shape creation must be logically shown as the first to be used. All drawing tools for the Initial Shape drawing must be available in that window, in buttons or menus (B) Any new windows or menus that relate to Initial Shape must be gathered with the main Initial Shape window (C) If a command line exists for the Initial Shape creation, it shall be located on the bottom of the Initial Shape window
Task 2 Rule creation	SGRC-GDLO	(A) Window for Rule creation must be shown as logically the second one to be used. All the tools for the Rule definition must be available in that window, in buttons or menus (B) A window shall exist to show the list of existing rules, next to the Rule main window. Exact same logic if instead of a window for this purpose, a Drop-down menu is opened (C) Any new windows or menus that relate to Rule creation must be gathered within the main Rule window (D) If a command line exists for the Rule creation, it shall be located on the bottom of the Initial Shape window
Task 3 SG application	SGAPP-GDLO	(A) Window for SG application must be logically shown as the third one to be used. All the tools for the SG application must be available in that window, in buttons or menus (B) The button or data field to add the iterations number, to apply the SG, must be clearly situated in the SG window (C) Any new windows or menus that relate to SG application must be gathered with the main SG window
Task 4 SG manipulation	SGMAN-GDLO	(A) SG manipulation made by changing the iterations number must follow SGAPP-GDLO-B (B) A Window or Drop-down menu shall exist to show SG alternatives to be selected. This must be gathered with the main SG window (C) Tools related to save or export results shall be clearly located near the SG window
Task 5 SG alteration	SGALT-GDLO	(A) SG alteration made by changing the Initial Shape shall be clear by the application of SGSC-GDLO (B) SG alteration made by changing Rules shall be clear by the application of SGRC-GDLO

Table 8. Guidance – grouping and distinction of items by format

General	SG-GDFO	(A) Drawing windows for shapes and rule creation shall be graphically similar Drawing tools shall be the same in Shape and Rule window, so they can be easily recognized (B) SG windows shall be graphically identifiable as the main results and manipulation window (C) Secondary windows for Rules list or SG alternatives shall be graphically similar to be identifiable as secondary and have the same position relatively to the main window they refer to. If they are shown by a Drop-down menu, they shall be also similar graphically and in position in the window
Task 1 Shape creation	SGSC-GDFO	(A) All drawing tools for shape creation in the Initial Shape window shall be similar, either in a List Menu or in Buttons. These shall also be similar to conventional drawing tools CAD software (B) If a command line exists for shape definition, it shall be graphically identifiable as this kind of tool
Task 2 Rule creation	SGRC-GDFO	(A) All drawing tools for shape creation in the Rule window shall be similar, either in a List Menu or in Buttons. These shall also be similar to conventional drawing tools in CAD software (B) If a command line exists for rule definition, it shall be graphically identifiable as this kind of tool A secondary window for the list of Rules that compose the grammar shall be graphically identifiable as this kind of element. The same way if it is shown in a Drop-down menu
Task 3 SG application	SGAPP-GDFO	(A) The button or data field to add the iterations number shall be graphically relevant and predominant in the SG window
Task 4 SG manipulation	SGMAN-GDFO	(A) A secondary window for a list of SG alternatives shall be graphically identifiable as this kind of element. The same way if it is shown in a Drop-down menu
Task 5 SG alteration	SGALT-GDFO	(A) All windows, menus, and buttons shall be graphically identifiable according to SGSC-GDFO and SGRC-GDFO so that it is easy to identify the procedures to change the Initial Shape or Rules of the SG

https://doi.org/10.1017/S0890060417000695 Published online by Cambridge University Press

Table 9. Guidance – prompting

General	SG-PROM	(A) All windows clearly named, numbered according to order of use, if needed (B) For data field entries, display-associated label (C) Display measurement units when drawing tools are used (D) Provide drawing status information [shape measurements, shape color, thickness, filling, shape geometry (opened or closed)] (E) Provide help tools
Task 1 Shape creation	SGSC-PROM	(A) Window for Initial Shape creation name: 1 – INITIAL SHAPE or equivalent (B) Display only available actions and in order of use: drawing tools for shape drawing, editing, and saving (C) If drawing is made by data entry, provide required formats and accepted values
Task 2 Rule creation	SGRC-PROM	(A) Window for Rule creation name: 2 – RULES or equivalent (B) Display only available actions: drawing tools for shape drawing, editing, and saving (C) If drawing is made by data entry, provide required formats and accepted values (D) Provide Drop-down menu, List, or Window with saved rules that will compose the SG
Task 3 SG application	SGAPP-PROM	(A) Window for SG application name: 3 – SHAPE GRAMMAR or equivalent (B) Display only available actions: number of iterations to be used, editing, and saving (C) Provide the required format and acceptable values for the iterations number to apply (D) Provide Drop-down menu, List, or Window with SG alternatives
Task 4 SG manipulation	SGMAN-PROM	(A) Display only available actions: changing number of iterations, choosing different alternatives, edit, save, and export.
Task 5 SG alteration	SGALT-PROM	(A) Display only available actions: add/change Initial Shape; add/change rule or rules order

Table 10. Guidance – legibility

General	SG-LEGI	(A) Window names shall be all equally formatted and aligned, preferably with upper case letters and aligned on the left (B) Menus in all the windows shall be distributed with the same inter-word spacing and preferably with upper case letters (C) Tools and menus shall follow conventional CAD software
Task 1 Shape creation	SGSC-LEGI	(A) Drawing tools shall have clear icons if they are buttons, and preferably, have an associated label (B) Drawing window shall have a contrasting background with the drawing lines. Ideally black background and white shapes or *vice versa*. A scalable grid can help understand shapes relations (C) If a command line exists to enter shape definitions, the text shall have a sans serif font and with a good readable size
Task 2 Rule creation	SGRC-LEGI	(A) Drawing tools shall have clear icons if they are buttons, and preferably, have an associated label (B) Drawing window or Rules list window shall have a contrasting background with the drawing lines. Ideally black background and white shapes or *vice versa*. A scalable grid can help understand shapes relations (C) If a command line exists to enter shape definitions, the text shall have a sans serif font and with a good readable size
Task 3 SG application	SGAPP-LEGI	(A) SG window shall have a contrasting background with the drawing lines. Ideally black background and white shapes or *vice versa*. A scalable grid can help understand shapes relations (B) If a command line exists to enter shape definitions, the text shall have a sans serif font and with a good readable size (C) Iterations button or data field shall have a velar sans serif font and with good readable size
Task 4 SG manipulation	SGMAN-LEGI	(A) If an SG alternatives' window exists, it shall have a contrasting background with the drawing lines. Ideally black background and white shapes or *vice versa* (B) Iterations button shall be as SGAPP-LEGI-C
Task 5 SG alteration	SGALT-LEGI	(A) Tools and menus needed to manipulate the SG shall be according to SGSC-LEGI, SGRC-LEGI, and SGAPP-LEGI

https://doi.org/10.1017/S0890060417000695 Published online by Cambridge University Press

Table 11. User workload – brevity – concision

General	SG-CONC	(A) Possibly existing data field shall allow exclusively accepted values (B) If a measurement unit is associated with a particular data field for drawing, include that unit as part of the field label (C) If codes are used in a command line to activate tools, use short codes with no more than five characters (D) In icon buttons, allow labels with the description to appear when hovering the cursor
Task 1 Shape creation	SGSC-CONC	(A) Drawing tools for Initial Shape drawing shall be the only ones available directly on the Initial Shape window (B) If a command line is used, drawing commands shall be small and clear. Measurements shall automatically have the measurement unit, instead of making the user add them
Task 2 Rule creation	SGRC-CONC	(A) Drawing tools for Rule drawing shall be the only ones available directly on the Rule window and be similar to the ones to create Initial Shapes (B) If a command line is used, commands shall be small and clear. Measurements shall automatically have the measurement unit, instead of making the user add them (C) The Window or Drop-down menu with list of rules shall allow them to be selected or reordered by simple click and drag with the cursor
Task 3 SG application	SGRC-CONC	(A) Iterations button or data field shall accept numbers exclusively and allow only the length acceptable by the implementation
Task 4 SG manipulation	SGMAN-CONC	(A) The Window or Drop-down menu with alternatives shall allow them to be selected by simple click with the cursor
Task 5 SG alteration	SGALT-CONC	(A) The tools to be used shall follow SGSC-CONC, SGRC-CONC SGAPP-CONC, and SGMAN-CONC

Table 12. User workload – brevity – minimal actions

General	SG-MIAC	(A) Reduce to the minimum necessary actions to create an SG (B) Reduce to the minimum the codes to be used in command lines
Task 1 Shape creation	SGSC-MIAC	(A) For the creation of the Initial Shape, only drawing tools shall be needed and they shall immediately be available in the Rules window (B) Manipulation of the shapes shall be done by click and drag with the cursor (C) If a command line exists for the creation of shapes, these shall be easily created by simple commands with the dimensions' introduction (D) Saving and editing tools shall be distinguished as secondary tools
Task 2 Rule creation	SGRC-MIAC	(A) For the creation of Rules, only drawing tools shall be needed and they shall be immediately available in the Rules window (B) Manipulation of the shapes shall be done by click and drag with the cursor (C) The Initial shape shall automatically appear in the Rules window to avoid repeating its drawing (D) When an SG is made from a list of rules, allow the creation of a new rule starting from the drawing of the previous one (E) The insertion of Labels shall be done by single commands, preferably a button with the label to be used (F) If a command line exists for the creation of shapes, these shall be easily created by simple commands with the dimensions' introduction (G) Saving a rule to the list shall be done by a single command that will add it to the Rules list
Task 3 SG application	SGAPP-MIAC	(A) A simple button or data field shall be needed to indicate the iterations number and apply the SG, and no other command shall appear as relevant in the SG window (B) Saving a result shall be done by a single command
Task 4 SG manipulation	SGMAN-MIAC	(A) Manipulation of the SG shall be easily done by changing the iterations number, according to SGAPP-MIAC-A (B) Choosing alternatives shall be done simply by click and drag with the cursor
Task 5 SG alteration	SGALT-MIAC	(A) The tools to be used shall follow SGSC-MIAC, SGRC-MIAC SGAPP-MIAC, and SGMAN-MIAC

https://doi.org/10.1017/S0890060417000695 Published online by Cambridge University Press

Table 13. User workload – information density

General	SG-INDE	(A) The implementation display shall exclusively show the windows referred in SG-GLO-A (B) Other windows shall be presented as secondary or opened only when needed (C) Command lines shall be integrated in the bottom of the corresponding window (D) Allow shapes, rules, and results to be permanently visible, so the user does not have to memorize and give them proper emphasis in the rest of the environment (E) Computation needed for the SG application shall be automatic and without any calculated entry done manually by the user
Task 1 Shape creation	SGSC-INDE	(A) Initial Shape window shall be a simple drawing area with only drawing tools available (B) A command line for shape creation can be shown only when needed, in the bottom of the Rules window
Task 2 Rule creation	SGRC-INDE	(A) Rules window shall be a simple drawing area with drawing tools available and a save command to create list of rules for SGs with more than one rule (B) A window with the graphical visualization of the rules can be opened when needed, instead of being always visible (C) A command line for rule creation can be shown only when needed, in the bottom of the Rules window
Task 3 SG application	SGAPP-INDE	(A) SG window shall be a simple drawing area with the iterations number tool as main command
Task 4 SG manipulation	SGMAN-INDE	(A) A window with the graphic visualization of alternatives can be opened when needed instead of being always visible
Task 5 SG alteration	SGALT-INDE	(A) The tools to be used shall follow SGSC-INDE, SGRC-INDE, SGAPP-INDE, and SGMAN-INDE

Table 14. Explicit user actions

General	SG-EXUA	(A) Request from the user for a clear Enter action to initiate the SG process (B) Request from the user for a clear click on Menu and Drop-down menu choices (C) Request from the user for a clear Enter action when using command-line entries
Task 1 Shape creation	SGSC-EXUA	(A) When defining the Initial Shape, request a Finish command to declare that the drawing is finished and that the next task can be started (B) If the definition is made in a command line, request an Enter command after each code used
Task 2 Rule creation	SGRC-EXUA	(A) When defining Rules, request a Finish command to declare that the definition is finished, so that the rule can be added to the SG Rules list (B) If the definition is made in a command line, request an Enter command after each code is used (C) When all rules to be used by the SG are finished, request a Finish command to declare that the next task can be started
Task 3 SG application	SGAPP-EXUA	(A) When choosing the number of iterations to apply in the SG, request an Enter command for the action to be applied (B) To Save or Export a result, request an Enter command
Task 4 SG manipulation	SGMAN-EXUA	(A) To change the iterations number to apply to the SG, request an Enter command for the action to be applied (B) To select an alternative, preferably with Drag-and-drop from an alternative window or selection from it, request a Click on the desired shape to be dragged or selected to the SG window
Task 5 SG alteration	**IM-sgi EC abbreviation** SGALT-EXUA	(A) The tools to be used shall follow SGSC-EXUA, SGRC-EXUA, SGAPP-EXUA, and SGMAN-EXUA.

https://doi.org/10.1017/S0890060417000695 Published online by Cambridge University Press

Table 15. User control

General	SG-USCO	(A) Allow users to pace their entry, rather than controlling the time for each action (B) Allow users to Save the current state of the system to resume it in another time (C) Provide the user with Undo and Redo tools (D) Provide a Cancel option to take the user to the initial state of the system or action
Task 1 Shape creation	SGSC-USCO	(A) Allow the user to take its time to define the Initial Shape, allowing the drawing to be changed until the user considers it finished (B) Allow users to Save the Initial Shape to be used later
Task 2 Rule creation	SGRC-USCO	(A) Allow the user to take his/her time to define Rules, allowing the drawing to be changed until the user considers it finished (B) Allow users to Save Rules to be used later
Task 3 SG application	SGAPP-USCO	(A) Allow the user to see the iterations result without timing out (B) Allow users to Save results to be used later
Task 4 SG manipulation	SGMAN-USCO	(A) Allow the user to try several iterations numbers until reaching a desirable result (B) Allow the user to try alternatives and select them for further work without timing out (C) Allow users to Save results to be used later
Task 5 SG alteration	SGALT-USCO	(A) The tools to be used shall follow SGSC-USCO, SGRC-USCO, SGAPP-USCO, and SGMAN-USCO without timing out (B) Any new state shall be able to be saved and used later

Table 16. Error protection

General	SG-ERPR	(A) Assure that the SG implementation will deal properly with possible user error, including accidental inputs (B) Use display messages to warn the user about incorrect inputs (C) Display advisory messages before closing the SG implementation if all elements are not correctly saved or if there is a pending action (D) If the SG computation by the implementation is limited, user definitions that will create computational errors shall be prevented
Task 1 Shape creation	SGSC-ERPR	(A) Assure user creates an Initial Shape as first action, creating warnings if another tasks are attempted first (B) In the Expert Mode, where the user might start by Rules creation, a message shall appear to make the user confirm that action is first one (C) If Initial Shapes are defined by code, there shall be prevention of data entry errors, accepting only valid codes and warning if incorrect ones were tried
Task 2 Rule creation	SGRC-ERPR	(A) The SG implementation shall have the means to advise the user if the Rules attempted to be made have errors that will lead to a computation error of the SG (B) Unless the defined Rule is valid, it shall not be possible to save it and use it (C) If Rules are defined by code, there shall be prevention of data entry errors, accepting only valid codes and warning if incorrect ones were tried
Task 3 SG application	SGAPP-ERPR	(A) The iterations field shall only allow inputs of integer numbers and warn if incorrect data were input (B) If a limited number of iterations is possible for a given SG, there shall be a warning and only valid numbers accepted
Task 4 SG manipulation	SGMAN-ERPR	(A) The implementation shall allow the user to access alternatives to SG results without destroying the SG rules definition
Task 5 SG alteration	SGALT-ERPR	(A) The error prevention of user actions to change the SG shall follow SGSC-ERPR, SGRC-ERPR, SGAPP-ERPR, and SGMAN-ERPR

https://doi.org/10.1017/S0890060417000695 Published online by Cambridge University Press

Table 17. Quality of error messages

General	SG-QUEM	(A) If the user takes an incorrect action, this shall take no effect and an error message shall be displayed (B) Error messages shall have task-oriented wording (C) The error messages shall be specific and brief, informing clearly the error made and the correct action to take (D) Different error messages shall be used according to the expertise mode in function
Task 1 Shape creation	SGSC-QUEM	(A) Any error message regarding Initial Shape shall follow the next template, or equivalent: "INITIAL SHAPE: Please make sure to…."
Task 2 Rule creation	SGRC-QUEM	(A) Any error message regarding Rules shall follow the next template, or equivalent: "RULE CREATION: Please make sure to…."
Task 3 SG application	SGAPP-QUEM	(A) Any error message regarding the SG by its iterations application shall follow the next template, or equivalent: "SG ITERATIONS: Please make sure to…."
Task 4 SG manipulation	SGMAN-QUEM	(A) Any error message regarding the SG alternatives shall follow the next template, or equivalent: "SG ALERNATIVES: Please make sure to…."
Task 5 SG alteration	SGALT-QUEM	(A) Any error messages needed when the user changes prior definitions shall follow the corresponding templates, according to SGSC-QUEM, SGRC-QUEM, SGAPP-QUEM, and SGMAN-QUEM

Table 18. Error correction

General	SG-ERCO	(A) Users shall be allowed to edit an action before taking the explicit Enter entry, allowing corrections during composition (B) After an error message, the user shall be able to take the correct action or correct the error directly and immediately
Task 1 Shape creation	SGSC-ERCO	(A) Allow the user to make any changes to the Initial Shape drawing until a valid shape is accepted (B) Tools to Edit the drawing shall be available to allow the user any correction to the drawing needed (C) If the Initial Shape is done by code, its code shall be accessible and changeable at all times
Task 2 Rule creation	SGRC-ERCO	(A) Allow the user to make any changes to the Rules definition until a valid Rule is accepted. (B) Tools to Edit the drawing shall be available to allow the user any correction to the drawing needed (C) If the Rules are done by code, their code shall be accessible and changeable at all times
Task 3 SG application	SGAPP-ERCO	(A) Allow the user to change the iterations number if an incorrect entry is done
Task 4 SG manipulation	SGMAN-ERCO	(A) Allow the user to manipulate alternatives without losing visibility or access to the ones used before or the ones never used
Task 5 SG alteration	SGALT-ERCO	(A) Any prior action shall be accessible and changeable according to SGSC-ERCO, SGRC-ERCO, SGAPP-ERCO, and SGMAN-ERCO

IM-sgi application

Application of the IM-sgi criteria is needed to verify if the definition of suitable interfaces for SG implementations is accurate. Depending on the importance and complexity of interface design, many tools are available to make interface prototypes prior to the application development. Prototyping unveils and explores human needs. In the information technology (IT) market, prototyping helps both the client and the designer/developer to understand each other's requirements. The prototype plays a major role in giving clarity to a client who does not have prior IT experience. This initial, raw presentation of an interface has been named low-

https://doi.org/10.1017/S0890060417000695 Published online by Cambridge University Press

fidelity prototyping (Rudd et al., 1996). Unlike high-fidelity prototyping, this method requires less time and fewer specialized skills and resources. Its purpose is not to impress users, but to learn from them. The goal of low-fidelity prototyping is not connected to a client/programmer negotiation but has the users as its main focus. In a way, the low-fidelity prototyping technique facilitates listening, rather than selling. It opens a conversation in which users' needs, designers' intentions, and other stakeholders' goals are discussed and aligned.

In order to be able to test and validate IM-sgi in this investigation, we developed a low-fidelity prototype, so that we could analyze IM-sgi criteria before testing with real users and with a real SG implementation development. This prototype translates visually what IM-sgi criteria define – one of many options possible. For our purposes here, this simple prototype was defined for 2D SG, as this type of representation is adequate for the purpose of using SG in the architectural practice. Although architecture is a 3D area, 2D representations of shapes can have multiple meanings. The clarity of 2D plans remains the primordial architectural representation (Ligler & Economou, 2015). Thus, 2D SG are adequate for creative exploration in this field.

SG implementation interface prototype

To calibrate the level of expertise of the users, IM-sgi was developed for the three types of users described in section "IM-sgi criteria". In the prototype created, a landing page, shown in Figure 1, is proposed to make the user choose the level of expertise that he/she believes better suits his/her SG knowledge. The name of the different expertise levels follows the level of expertise of the three types of users defined above. The choice of each level will originate the loading of the interface that should communicate the user's characteristics. Another option for this differentiation of the interface according to the level of expertise could be chosen, that is, a predefined interface could be loaded automatically, and then it asks the user if he/she would like to use a different one.

The screen in Figure 2 below shows a GUI according to the IM-sgi criteria SG-COMP (see Table 1), representing the users' characteristics. As the interface is directed mainly for the users of design areas, a GUI that resembles existing CAD software will be easier to be understood and accepted by this group, as stated by the IM-sgi criteria SG-FLEX-FEED.

Three drawing windows are shown according to the IM-sgi criteria SG-FLEX (see Table 3) and SG-SICO (see Table 4). The windows are numbered and named according to the IM-sgi criteria SICO (see Table 4), SG-GDLO (see Table 7), and SG-PROM (see Table 9). The first task for the beginner is to define the Initial Shape. In this level of expertise, a toolbox with shapes is shown so that the user may drag them to the Initial Shape window, according to SGSC-USEX-A (see Table 2) and SGSC-GDLO-B (see Table 7). When the user manipulates a shape by drag and drop, the Rules window will update in real time, as all the other windows, showing the shape to be used as the basis for Rule definition. This function enables the users to get an immediate, real-time response to their actions. In the Beginner Mode, the use of messages indicating user steps is desirable. This mode may play the role of a tutorial for the use of the application (see Tables 1–18 for competing explanation of criteria abbreviations).

The Rules window is the second one to be used, after the Initial Shape window, according to SGRC-USEX (see Table 2) and SGRC-GDLO-A (see Table 7). A toolbox with a group of Rules is graphically shown for the selection of rules from previously saved ones, or predefined by the application, is shown below the Rules window. This is called the Rules List, according to SGRC-GDLO-B (see Table 7) and SGRC-GDLO-C (see Table 7). The use of the SG is made by selecting the iterations number. A number can be predefined to allow a solution to appear immediately as the Rule is created, according to SGAPP-USEX (see Table 2). As the SG is applied, a number of alternatives will be shown in the Other Results tool bar. These can then be selected for work in the SG window, according to SGALT-GDLO (see Table 7). Generally, all criteria that can be verified by a graphic prototype were considered, except error management.

In our prototype, the Intermediate and Expert Modes follow the same organization as the Beginner Mode. The Intermediate Mode is for artists/designers who use conventional CAD software. Following this, the toolbox with predefined shapes to be selected is eliminated, giving more space for general drawing in the Initial

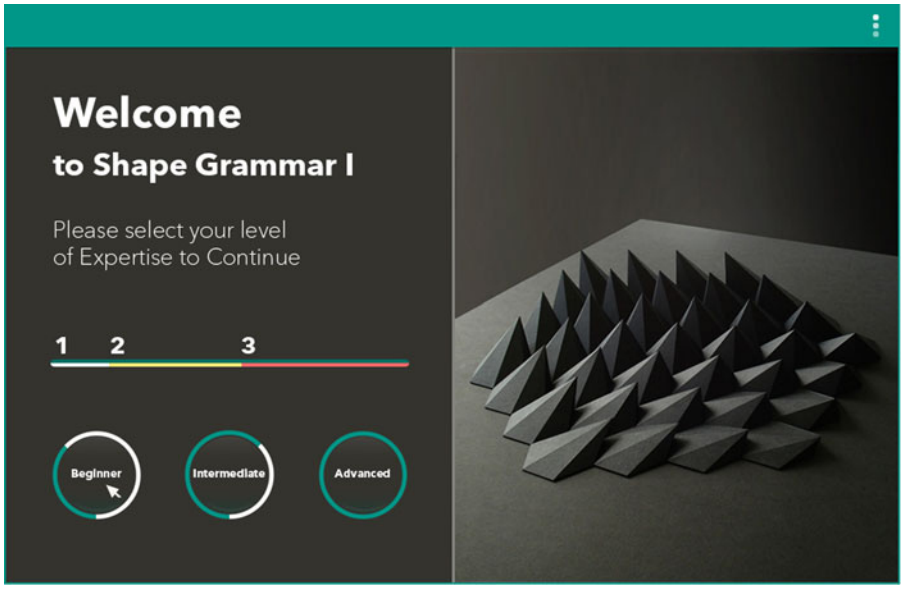


Fig. 1. SG Welcome Screen.

https://doi.org/10.1017/S0890060417000695 Published online by Cambridge University Press

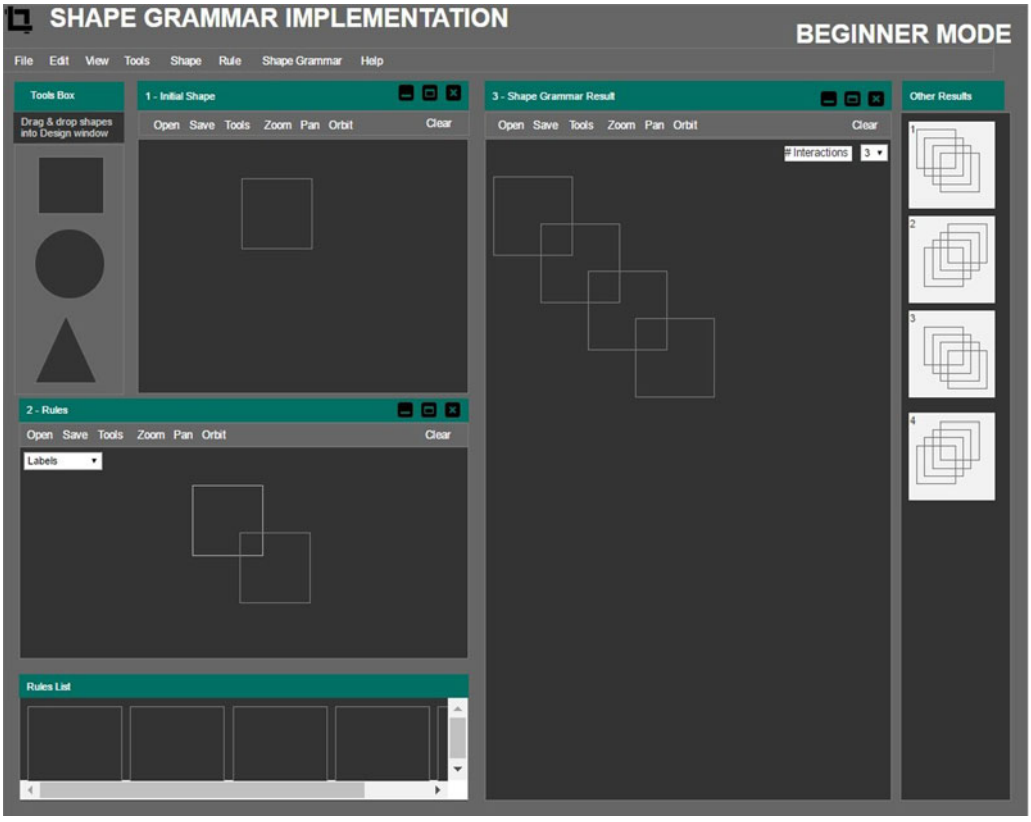


Fig. 2. SG Beginner Mode screen.

Shape window. The general display and functioning is similar to the previous mode so that the user can naturally evolve and use the next Expert Mode without having to interpret the functioning of the application again, according to SG-CONS (see Table 5). Also in the Intermediate Mode, according to SGRC-USEX-B (see Table 2) and SGRC-FLEX-C (see Table 3), a drop-down list exists in the Rules window for the use of Labels when creating the rules.

The Expert Mode allows availability of command lines to define Rules or change the SG through code. These command lines can be directly in the window of the task to be performed, or at the bottom of the entire interface, usable for any task. In this mode, the user is an expert in SG and/or in the development of the implementation itself, so expert tools are available, according to SG-USEX-D (see Table 2). The command lines will allow the Rule definition to be made with Rule Schemata or other input method needed/defined (Economou & Kotsopoulos, 2014).

As stated in section "IM-sgi application" above, this simple prototype is just an example. The prototype was defined for 2D SG, as this type of representation is adequate for the purpose of using SG in the architectural practice (Ligler & Economou, 2015). Further work involves prototypes that also consider 3D SG.

Conclusion

The present work aims to help computer SG applications effectively enter the design practice and become a relevant way of exploring ideas and project solutions. Designers and architects would be able to see SG as a real potential design technique and the computer as a partner. The architect has been changing the way of doing his job since he has had the chance to use the computer. Computational applications allowed reproduction of the architectural drawings, in addition to simulating 3D models. It became possible to explore multiple hypotheses, reduce project errors, and spend less time in searching for more complex solutions, achieving more ambitious, innovative, and creative projects. Applications based in SG can be tools that offer the designer the potential for Artificial Intelligence (AI) and CC.

We started with a brief introduction to SG and HCI. Then we presented how SG can lead to new ways of exploring project solutions and how we think this can happen with the correct interface for the computational SG implementations. Finally, describing the development of this investigation, IM-sgi is presented as our proposal of an interface model for SG implementations.

The beginning of the work in progress is presented here by an interface prototype for SG implementations, which allows an example of use of our established criteria for an interface definition. IM-sgi is defined by 259 criteria divided into 18 sections. In each section, there are six fields that organize the criteria between general and five directed for each task previously defined to work with SG. In order to validate the IM-sgi model, a prototype and its testing and evaluation are under development and will be presented in a subsequent paper. The objective is to show how IM-sgi criteria apply to different SG implementations, according to the different features they might possess (2D or 3D shapes, shape emergence, or other) and also to different types of interface and input, allowing newer techniques, such as eye tracking, to be used (Jowers et al., 2013).

https://doi.org/10.1017/S0890060417000695 Published online by Cambridge University Press

References

Apple CI (1992) *Macintosh Human Interface Guidelines*. Reading, MA: Addison Wesley.

Bach C and Scapin DL (2003) Adaptation of ergonomic criteria to human-virtual environments interactions. *INRIA*.

Bastien J and Scapin D (1993) Ergonomic criteria for the evaluation of human-computer interfaces. *INRIA*.

Bastien JM and Scapin DL (1995) Evaluating a user interface with ergonomic criteria. *INRIA*, n° 2326.

Bodart F and Vanderdonckt J (1995) *Guide ergonomique de la présentation des applications hautement interactives*. Presses Universitaires de Namur.

Brown MH (1998) Perspectives on algorithm animation. *ACM CHI'88 Conference on Human Factors in Computing Systems*, pp. 33–38.

Chase S (2002) *A Model for User Interaction in Grammar-Based Design Systems* (Vol. Automation in Construction 11). Elsevier.

Economou A and Kotsopoulos S (2014) From shape rules to rule schemata and back. *Design Computind and Cognition*. doi: 10.1007/978-3-319-14956-1_22

Hollender N, Hofmann C, Deneke M and Schmitz B (2010) Integrating cognitive load theory and concepts of human-computer interaction. *Computers in Human Behavior* **26**(6), 1278–1288.

IBM. (1989) *IBM System Application Architecture, Common User Access: Advanced Interface Design Guide*. International Business Machines. doi: SC26-4582-0

Jowers I, Prats M, McKay A and Garner S (2013) Evaluating an eye tracking interface for a two-dimensional sketch editor. *CAD Computer Aided Design* **45**(5), 923–936.

Karray F, Alemzadeh M, Saleh JA and Arab MN (2008) Human-computer interaction: overview on state of the art. *International Journal on Smart Sensing and Intelligent Systems* **1**(1), 137–159.

Krishnamurti R (1980) *The Arithmetic of Shapes*, Vol. **7**. Environment and Planning B: Planning and Design.

Lewis C and Rieman J (1994, 6 16) Task-Centered User Interface Design. Available at http://hcibib.org/tcuid/chap-4.html#4-1

Liew H (2002) Descriptive conventions for shape grammars. *ACADIA*.

Ligler H and Economou T (2015) Lost in translation: towards an automated description of John Portmans's domestic architecture. *SIGRADI* **2015**, pp. 657–661. doi: ISBN: 978-85-8039-133-6

McKay A, Chase SC, Garner SW, Jowers I, Prats M, Hogg DC, Lim S (2009) Design synthesis and shape generation. *Designing for the 21st Century: Interdisciplinary Methods and Findings*, pp. 304–321.

McKay A, Chase SC, Shea K and Chau HH (2012) Spatial grammar implementation: from theory to useable software. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **26**(2), 143–159.

Molich R and Nielsen J (1990) Improving a human-computer dialogue. *Communications of the ACM*, pp. 338–348.

Myers B (2008) *A Quick Overview of Human-Computer Interaction*. Human Computer Interaction Institute, Carnegie Mellon University, pp. 1–36.

Nielsen J (1994) Usability Inspection Methods. *Conference Companion *CHI'940*, Boston, Massachusetts, USA, pp. 413–414.

Ravden S (1988) Ergonomic criteria for design of the software interface between human and computer. *CIM International Journal of Computer Applications in Technology*, pp. 35–42.

Rudd J, Stern K and Isensee S (1996) Low vs. high-fidelity prototyping debate. *Interactions* **3**(1), 76–85.

Scapin DL (1986) *Guide ergonomique de conception des interfaces homme-machine*. Technical Report No.77, Institut National de Recherche en Informatique et en automatique, Rocquencourts, France.

Scapin DL (1990) Decyphering human factors recommendations. *Ergonomics of Hybrid Automated Systems II*, pp. 27–34.

Schneiderman B (1987) *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Massachusetts: Addison-Wesley.

Smith SM (1986) *Guidelines for Designing User Interface Software*. Mitre Corporation.

Stiny G (1977) Ice-ray: a note on the generation of Chinese lattice designs. *Environment and Planning B* **4**, 89–98.

Stiny G and Gips J (1975) *Shape Grammars and Their Uses: Artificial Perception, Shape Generation and Computer Aesthetics*. Munich: Birkhauser, p. 1. doi: 10.1007/978-3-0348-5753-6.

Tching J, Reis J and Paio A (2013) Shape grammars for creative decisions in the architectural project. *CISTI*.

Tching J, Reis J and Paio A (2016) A cognitive walkthrough towards an interface model for shape grammar implementations. *Journal of Computer Science and Information Technology* **4**(3), 92–119.

Trescak T, Esteva M and Rodriguez I (2012) A shape grammar interpreter for rectilinear forms. *CAD Computer Aided Design* **44**(7), 657–670.

Yue K and Krishnamurti R (2014) A paradigm for interpreting tractable shape grammars. *Environment and Planning B*. doi: 10.1068/b3910 7

Joana Tching (joana.tching@netcabo.pt) PhD Student in the Department of Sciences and Technologies of Information at ISCTE-IUL. Architect at FA-UTL, Lisbon. Experienced Architect in Building Information Modeling (BIM), Autodesk Revit Instructor. Her present research interests are Computational Design and Creativity, Artificial Intelligence, and Shape Grammars.

Joaquim António Marques dos Reis (joaquim.reis@iscte.pt) Assistant Professor in the Department of Sciences and Technologies of Information at ISCTE-IUL. Mechanical Engineer, MSc in Mechanical Engineering at Instituto Superior Técnico, Lisbon. PhD in Sciences and Technologies of Information at ISCTE-IUL (thesis "A Multi-Agent Scheduling Model in the Extended Enterprise"), Lisbon. Researcher and member of the Scientific Committee of ADETTI-IUL, an ISCTE-IUL investigation unit, and member of APPIA, the Portuguese Association for Artificial Intelligence. His present research interests fall in the areas of Artificial Intelligence, Computational Creativity, and Shape Grammars. Currently, he teaches the Artificial Intelligence and Technologies for Intelligent Systems course units (in the Telecommunications and Informatics Engineering course) and the Computational Creativity MSC course unit.

Alexandra Paio (alexandra.paio@iscte.pt) Assistant Professor in the Department of Architecture and Urbanism at ISCTE-IUL. Director and Researcher of VitruviusFabLab-IUL and Researcher at ADETTI-IUL. Co-Director of Advanced Studies Course in Digital Architecture (ISCTE-IUL + FAUP). PhD in Urban Design at ISCTE-IUL, entitled "urbanGENE: An Urban Grammar for Portuguese Settlements from the 16th to the 18th century". Research at "OIKOnet. A global multidisciplinary network on housing research and learning" (co-financed by European Union). Her main research interests are Computational Design, Digital Tools and Processes to support the creative design, Interactive Architecture, Shape Grammars, and Digital Fabrication.

https://doi.org/10.1017/S0890060417000695 Published online by Cambridge University Press