Beyond analysis and representation in CAD: a new computational approach to design education

By Maria Gabriela Caffarena Celani
M.Sc. in Architecture, University of São Paulo
Submitted to the Department of Architecture in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the field of Architecture: Design & Computation at the Massachusetts Institute of Technology
September 2002

© 2002 Massachusetts Institute of Technology
and Conselho Nacional de Pesquisa
e Desenvolvimento Tecnológico.
All rights reserved.

Signature of Author:

Department of Architecture, MIT
15 July 2002

Certified by:

William J. Mitchell
Professor of Architecture and Media Arts and Sciences
School of Architecture and Planning, MIT
Co-chair, Dissertation Committee

Terry W. Knight
Associate Professor of Design and Computation
Department of Architecture, MIT
Co-chair, Dissertation Committee

Accepted by:

Stanford Anderson, Professor of History and Architecture
Head of the Department, Department of Architecture, MIT
Chair, Committee on Graduate Students
To my daughters, Ale and Bia.
Dissertation Committee

Committee Co-Chairs

William J. Mitchell
Professor of Architecture and Media Arts and Sciences,
School of Architecture and Planning, MIT

Terry W. Knight
Associate Professor of Design and Computation,
Department of Architecture, MIT

Committee Members

George Stiny
Professor of Design and Computation,
Department of Architecture, MIT

Mitchel Resnick
Associate Professor,
Program in Media Arts and Sciences, MIT
Abstract

Beyond analysis and representation in CAD: a new computational approach to design education

By Maria Gabriela Caffarena Celani

Submitted to the Department of Architecture on July 15, 2002, in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the field of Architecture: Design & Computation, at the Massachusetts Institute of Technology.

This thesis aims at changing students' attitude towards the use of computer-aided design (CAD) in architecture. It starts from the premise that CAD is used mostly for analysis and representation, and not as a real design aide, and that architecture students have a bias against learning computer programming. For this purpose, a prototypical instruction system that mixes computer-aided design and computational design theory was developed, based on a series of fundamental concepts that are common to both fields. This system was influenced by Mitchell's (1987) The Art of Computer Graphics Programming and Stiny's (1976) shape grammars. Despite being based on solid theoretical foundations, CAD has progressively become an exclusively practical tool, since its origins in the 50's and 60's, while computational design theories have been mostly restricted to the academic circles. This thesis proposes an inversion in the present situation: the study of CAD theory, and the application of computational design into practice. The system proposed provides a conceptual framework that can be adapted to different circumstances, including course formats and resources, as well as students' background and technical training. It is based on seven fundamental concepts from computational design theories that are also important to the study of shape grammars: symmetry, recursion, rule-based compositions, parameterization of shapes, generative systems, algorithmization of design procedures, and shape emergence. These concepts are introduced within a CAD context, where their practical implementation and experimentation are possible, focusing the understanding of the computational nature of design. During this research, the proposed system was tested in two case studies with students from schools that had contrary orientations in terms of the importance of CAD in the architectural curriculum. In these experimental courses, students' activities evolved from using a commercial CAD tool in an innovative way, to the use of programming techniques for creating meaningful tools. Despite not having a statistical reach, the fieldwork allowed drawing preliminary conclusions about the proposed system's efficacy, since virtually all the students reported changing their understanding of the role of CAD in architecture, while some also acknowledged a conceptual influence in other subjects and in the way they see architecture.
Acknowledgement of sponsors

I would like to express my special thanks to CNPq, the Brazilian National Council for Scientific and Technologic Development, and to the Rosenblith Fund for their financial support.
Gabriela Celani, born (1967) and raised in São Paulo city, Brazil, holds a B.Arch. (1989) and an M.Sc. (1997) in Architecture and Planning from the School of Architecture and Planning of the University of São Paulo (FAU-USP). She taught at the University of the Paraiba Valley (UNIVAP), between 1993 and 1997, and worked as a research and teaching assistant at MIT for Professors William J. Mitchell, Terry W. Knight, Takehiko Nagakura and Michael Dennis from 1999 to 2001. She has also worked in architectural practice for many years in Boston, São Paulo, Rio de Janeiro and São José dos Campos.
Acknowledgements

There are many people I would like to thank for their support and help during the years I spent at MIT, and when I was doing the field research in Brazil. Some of them are, in alphabetical order:

Ana Vera Macedo Wynne
Birgul Kolakoglu, Ph.D. graduate in Design & Computation
Cynthia Wilkes, secretary of the Deans, School of Architecture and Planning, MIT
Doris Kowal{tionkowski, Professor of Architecture, FEC-UNICAMP
Flávia McLaughlin
George Stiny, Professor of Design and Computation, Department of Architecture, MIT
Jack Valleli, Department of Architecture, MIT, Administrative Assistant
João Rocha, Ph.D. candidate in Design & Computation
José Duarte, Ph.D. graduate in Design & Computation
Larry Sass, Ph. D. Assistant Professor, Department of Architecture, MIT
Marcelo Giacaglia, Professor of Architecture, FAU-USP
Maria Antonieta Celani, Professor of Applied Linguistics, PUCSP
Maria Ruth Amaral Sampaio, Dean, FAU-USP
Miranda McGill, M.Sc. in Architecture, MIT
Mitchel Resnick, Associate Professor, Media Laboratory, MIT
Renée Caso, Administrator of Academic Programs, Department of Architecture, MIT,
Riusuke Naka, Professor of Architecture, MIYAGI University
Roberto Okada, Professor of Information Technology, MIYAGI University
Susan Yee, Ph.D. Research Scientist, Department of Architecture, MIT
Takehiko Nagakura, Professor of Architecture, Department of Architecture, MIT
All the staff at MIT

Very special thanks to Terry W. Knight for her tireless and diligent reading of all my writing, endless patience and continuous support, and to William J. Mitchell for his brilliant insights that encouraged me to join this Ph.D. program to begin with and guided me through this dissertation process. Very special thanks also to Mitchel Resnick, for making me see computers in education from a whole new perspective and for his thoughtful suggestions in the final text, and to George Stiny, for being a continuous source of inspiration and always encouraging me to think deeper.

I would also like to thank all the students at UNICAMP and FAU-USP who participated in the experimental courses, all of my family and friends for their support and patience, and my husband, Ba, for always making me believe I could make it.
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>I. Introduction</strong></td>
<td>17</td>
</tr>
<tr>
<td><strong>II. Background</strong></td>
<td>25</td>
</tr>
<tr>
<td>II.a. CAD history</td>
<td></td>
</tr>
<tr>
<td>II.b. Computational Design</td>
<td></td>
</tr>
<tr>
<td>II.c. CAD in Brazilian architectural education</td>
<td></td>
</tr>
<tr>
<td>II.d. Research questions</td>
<td></td>
</tr>
<tr>
<td><strong>III. Methodology</strong></td>
<td>51</td>
</tr>
<tr>
<td>III.a. Research methods in this study</td>
<td></td>
</tr>
<tr>
<td>III.b. Sources of secondary research</td>
<td></td>
</tr>
<tr>
<td>III.c. Qualitative research</td>
<td></td>
</tr>
<tr>
<td>III.d. Case study versus experiments, survey and action research</td>
<td></td>
</tr>
<tr>
<td>III.e. Grounded theory</td>
<td></td>
</tr>
<tr>
<td>III.f. Reliability and validity</td>
<td></td>
</tr>
<tr>
<td><strong>IV. Prototypical System</strong></td>
<td>61</td>
</tr>
<tr>
<td>IV.a. Objectives</td>
<td></td>
</tr>
<tr>
<td>IV.b. A generic system</td>
<td></td>
</tr>
<tr>
<td>IV.c. Pedagogical issues</td>
<td></td>
</tr>
<tr>
<td>IV.d. Choosing concepts and examples</td>
<td></td>
</tr>
<tr>
<td>IV.e. Symmetry</td>
<td></td>
</tr>
<tr>
<td>IV.f. Recursion</td>
<td></td>
</tr>
<tr>
<td>IV.g. Design rules</td>
<td></td>
</tr>
<tr>
<td>IV.h. Parameterized shapes</td>
<td></td>
</tr>
<tr>
<td>IV.i. Generative systems</td>
<td></td>
</tr>
<tr>
<td>IV.j. Algorithmization of design procedures</td>
<td></td>
</tr>
<tr>
<td>IV.k. Emergent shapes</td>
<td></td>
</tr>
<tr>
<td>IV.l. Final discussion or project</td>
<td></td>
</tr>
<tr>
<td><strong>V. Case studies</strong></td>
<td>85</td>
</tr>
<tr>
<td>V.a. Two instances of the prototypical system</td>
<td></td>
</tr>
<tr>
<td>V.b. Pilot work</td>
<td></td>
</tr>
<tr>
<td>V.c. UNICAMP workshop</td>
<td></td>
</tr>
<tr>
<td>V.d. FAU-USP mini-course</td>
<td></td>
</tr>
<tr>
<td><strong>VI. Data analysis</strong></td>
<td>111</td>
</tr>
<tr>
<td>VI.a. Data organization and presentation</td>
<td></td>
</tr>
<tr>
<td>VI.b. Initial attitude towards CAD (assumption)</td>
<td></td>
</tr>
<tr>
<td>VI.c. Attitude towards programming</td>
<td></td>
</tr>
<tr>
<td>VI.d. Understanding design environments and tools</td>
<td></td>
</tr>
<tr>
<td>VI.e. Understanding the concepts</td>
<td></td>
</tr>
<tr>
<td>VI.f. Applying the concepts to architectural design</td>
<td></td>
</tr>
<tr>
<td>VI.g. Attitude-change towards CAD</td>
<td></td>
</tr>
<tr>
<td>VI.h. External reach</td>
<td></td>
</tr>
</tbody>
</table>
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>VII. Conclusions</td>
<td>VII.a.Confirmation of original assumptions</td>
<td>165</td>
</tr>
<tr>
<td></td>
<td>VII.b.Answering the original research questions</td>
<td></td>
</tr>
<tr>
<td></td>
<td>VII.c.Unexpected observations</td>
<td></td>
</tr>
<tr>
<td></td>
<td>VII.d.Critique</td>
<td></td>
</tr>
<tr>
<td></td>
<td>VII.e.Communication and Future work</td>
<td></td>
</tr>
<tr>
<td>VIII. Bibliography</td>
<td></td>
<td>175</td>
</tr>
<tr>
<td>IX. Appendix A</td>
<td>Students’ names and background</td>
<td>181</td>
</tr>
<tr>
<td>X. Appendix B</td>
<td>Description of exercises</td>
<td>183</td>
</tr>
<tr>
<td>XI. Appendix C</td>
<td>Excerpts from classroom journals</td>
<td>185</td>
</tr>
<tr>
<td>XII. Appendix D</td>
<td>Assessment questionnaires</td>
<td>189</td>
</tr>
<tr>
<td>XIII. Appendix E</td>
<td>Accompanying CD-ROM</td>
<td>201</td>
</tr>
</tbody>
</table>
I. Introduction

“What is a man if he is not even honest? And also, what is an architect, if he is not even logical?” (Rogers, 1979 p.27)

Assumptions

This thesis starts with the assumption that there is currently a general misconception about the application and utility of computers, in general, and computer-aided design (CAD) software, in particular, in architecture. After a period of mystification, when some people believed - while others doubted - that it was possible to solve all the problems by pressing a key, CAD technology has been used mainly for representational purposes, both for drafting and for realistic simulations. In other words, computer-aided design became computer-aided drafting.

There is evidence that the quick development of computer graphics and the popularization of computer hardware and software are some of the reasons that led to the present situation. As Mitchell (1990b) puts it,

"The negative effect of fourth-generation CAD was to establish a banal and simplistic conception of CAD functions and style of interaction in the minds of many architects(...) The theoretical foundations of these systems [established a quarter of century before] remained mostly unexamined, and the wider possibilities were largely ignored." (p.483)

Another assumption is that there is a bias against learning computer programming among architecture students, which often deprives them from having a higher control on the CAD tools they use. Finally, I also assume that most CAD software users do not have any knowledge about the origins of computer-aided design, which is another limitation to seeing applications of CAD that extend beyond pure representation and analysis.

At least two of the assumptions above were confirmed among the students who took part in the field-research experiments in this study. Because they all belonged to a particular country and region, it is impossible to generalize the assumptions to other cultures, but the results of the proposed system should be replicable in groups with similar characteristics.
I. Introduction

Definitions

The acronym CAD - meaning computer-aided design - is employed throughout this thesis as a synonym of the use of computer resources in the process of design in general, and particularly in architecture. Computer programs that aim specifically at helping designers are referred to as CAD software, which includes applications for representation (drafting, 3D-modeling, rendering, rapid-prototyping, movie-making, etc.), analysis, simulation, optimization, shape exploration, alternative generation, and automation of design tasks.

The expression computational design is used to refer to any kind of generative process of design based on the use of computation, and thus a computational design theory is any design theory based on computations. The word computation, in turn, is used as a synonym of performing logical operations, not necessarily in the computer, and not exclusively with numbers or other symbols, but also with shapes and other forms of analogical representation.

Aims and objectives

This thesis aims at encouraging architecture students to see CAD as a "design partner" (Negroponte, 1975a), a means of "augmenting" the design process (Negroponte, 1975b), and to make a better use of the features in which the computer is effectively good at. Its objectives are:

1. To propose a prototypical instruction system in which CAD and computational design theories are intertwined, the first bringing the second into practice, and the second justifying the first;
2. To test the proposed system in a "real world" context;
3. To search for preliminary evidence of the proposed system's efficacy in changing students' attitude towards the use of CAD in architectural design.

In order to test the system, two experimental courses were taught in two different architecture schools in Brazil. The two instances of the system constitute the case studies explained in detail in the body of this dissertation.

With the present thesis, I hope to contribute to the field of computer-aided design by:
I. Introduction

1. Suggesting an approach to CAD instruction that takes into account its underlying theoretical principles and is not exclusively concerned with representation;

2. Demonstrating practical applications of computational design theories that have been relegated to theoretical studies or implemented with the use of advanced techniques that are out of the reach of the common architect or the undergraduate student.

Scope

It is often difficult to explain what a thesis is not about. The issues below emerged from some of the initial questions and expectations presented by students of the two experimental courses. This thesis is not about:

1. Teaching programming, or the use of a particular CAD program or programming language;
2. Teaching design or a particular design methodology;
3. Presenting CAD and computational design techniques as the best or only way to design;

Although I have tried to be as inclusive as possible, the thesis also does not aim at showing all the possible applications of computational design and CAD. The concepts introduced in the prototypical system are without any doubt important ones in the underlying theoretical basis of CAD and computational design, but they certainly do not exhaust relevant topics in the study of each.

On the contrary, this thesis aims at helping architecture students (and designers in general) to see new possibilities in the use of computers in design, and at informing which are some of these possibilities, and how they can be applied with what they have at hand. Yet, the thesis does not imply that the quality of students’ designs will be necessarily improved with the new approach to CAD education proposed (which is, of course, very desirable).

Personal background

In my first architecture internship, while still attending college, I was often given the task of developing different alternatives for the same design problem. Design problems varied from optimization problems, such as generating different parking lot layouts, to more
open-ended, creative problems, such as suggesting different forms for a light fixture.

In my first job as an architect, I often felt mind-numbed when I had to hand-draw construction documents of different versions of the same bathroom for different clients (at that time I only used CAD for 3D presentation models and walkthroughs). When I started using CAD for drafting on a daily basis, such tasks became easier, for it was possible to reuse some of the lines from one drawing to the next, but I knew there was much more that could be done with the computer.

A few years later, as a CAD instructor for undergraduate architecture students, I had to focus on CAD software commands to fulfill students' needs. But at the same time I was already trying to give CAD a higher status by using three-dimensional modeling to help students visualize space while designing (that was the time of "virtual design environments"). Computer drafting and modeling soon became a prerequisite to some internships, but students found out that being experienced with CAD was not a synonym of making good architecture.

Digging out the history of CAD from its very origins, and getting in touch with computational design theories, I finally started seeing the connections between CAD software's geometric and generative principles and computational design theories. It is important, here, to make a clear distinction between "computational" and "computer-based" design. The former can - but does not necessarily need to - make use of computers, which we will be doing here. The latter always implies the use of computers, but not necessarily of a computational theory of form. In other words, by simply using the computer to manipulate shapes we are not necessarily designing within a computational thinking.

The first ideas about developing a prototypical system proposing an integration between CAD, programming instruction, and a computational study of form started in the [Northern Hemisphere's] winter of 2001, when I had the opportunity to co-teach a workshop on shape grammars with Professor Terry Knight and Miranda McGill (then an M.Sc. in Architecture student at MIT). The course focused on the use of computer applications for shape grammars, in a remote-collaborative design process between students at MIT and Miyagi University, in Japan (Celani, 2002). At that time, I realized
I. Introduction

how theoretical knowledge of computational design and technical training in CAD software and programming were complementary. Therefore, both are present in the instruction system developed here.

Sources of inspiration

Obviously, I do not claim to be the first person to think about an instruction system mixing computer-aided design and computational design theory. One of the pioneers of this approach was William Mitchell, one of my advisers. In his book with Ligget and Kvan (1987), Mitchell proposes a series of exercises using Pascal language, which are "as concerned with issues of design theory and visual aesthetics as [they] are with computer technology" (Mitchell, Ligget, & Kvan, 1987 p.vii). The art of computer graphics programming is a comprehensive book in terms of the technology it describes, but also very rich in beautifully illustrated architectural examples. Its conceptual exercises inspired me to propose the same kind of open-ended, interpretive questions that relate CAD programming to fundamental issues in design theory (and even to nature). As programming concepts are introduced, Mitchell proposes increasingly complex exercises on vocabularies of shapes, parameterized shapes, repetition, conditionals, hierarchical structures and geometric transformations. Table 1 shows some examples of his exercises, categorized under those concepts.

The system developed here is based on the same basic concepts proposed by Mitchell, but grouped in a different way and with different objectives. Mitchell's concepts, in turn, are very close to those in Stiny's shape grammars theory, another important influence to this thesis. Yet, unlike in Mitchell's book, the focus here is not on teaching programming, but rather on developing an understanding of different applications of the computer in architectural design. Mitchell used Pascal, a plain programming language, while I propose taking any CAD program that is at hand and using it in a new way, by customizing and developing programs to automate its functions. In the case studies in this research, the CAD software used was Autodesk's AutoCAD, a popular program among architects and schools, and the programming language, used sometimes like a script, was Visual Basic for application (VBA) edited in AutoCAD's own interactive development environment (VBAIDE). My approach focuses on the rapid return on the time invested in technical training and the practical application of the concepts.
<table>
<thead>
<tr>
<th>Concept</th>
<th>Parameters</th>
<th>Generation</th>
<th>Conditionals</th>
<th>Hierarchies</th>
<th>Recursion</th>
<th>Transformation</th>
<th>Nature</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Table 1:</strong> Some examples of exercises proposed in &quot;The Art of Computer Graphics Programming&quot; (Mitchell et al. 1987), separated by categories.</td>
<td><strong>Exercise</strong></td>
<td><strong>Write a parameterized procedure to generate the basic vocabulary element [for the plan Mies of Van der Rohe’s ‘Brick Country House’]. Use this in a program to replicate the plan. Then use it in a program to produce variations on this theme.&quot; (p.198).</strong></td>
<td><strong>Exercise</strong></td>
<td><strong>It has often been argued that the aesthetic success of a composition is a matter of appropriate balance between ‘unity’ (which may be established by regular repetition) and ‘variety’ (which may be introduced by changing parameters from instance to instance). Test this proposition by generating repetitive compositions with different degrees of variation. Provide a critical analysis of your results.&quot; (p.250).</strong></td>
<td><strong>Exercise</strong></td>
<td>**Many town plans consist of regular street grids interrupted at various points. Examine some plans of this type. What are the conditions in which the grid is interrupted? Write a set of conditional rules that could be used to produce plans of this type, and discuss their effects.&quot; (p.321)</td>
<td><strong>Exercise</strong></td>
</tr>
</tbody>
</table>
I. Introduction

I learned, which I consider crucial to the success of the system, given the assumptions described above.

I also do not claim originality when suggesting the use of programmable CAD software with generative purposes. Other people have already thought about it, among whom are, for example, Takehiko Nagakura and Peter Testa, who teach applied computational design courses at MIT. However, while their pragmatic approaches count on the technical and theoretical bases provided by other graduate courses at MIT, mine explicitly introduces both at the same time for the novice student.

Another thesis recently presented in this department (Yakeley, 2000) has also dealt with the question of CAD in architectural education, but with different objectives. While in that study programming was introduced as a way of helping students formalize their personal design methodology, here this formalization is considered as just one of the benefits of using CAD in the initial phases of design.

Parts of this thesis

This work is divided in seven sections:

1. In the Background section, I summarize the history of computer-aided design and introduce computational design concepts. I also comment on the current situation of the architectural curriculum in Brazil in terms of CAD and computational design, and ask preliminary research questions.

2. In the Methodology section, I explain the research methods that have been used in the secondary research, field research and data analysis.

3. In the Prototypical system section, I develop an instruction system based on a series of concepts and corresponding exercises. The choice of such concepts is justified in the light of the shape grammar theory of design, and pedagogical characteristics of the proposed system are explained. In each sub-section, sample programs developed in VBA that illustrate the respective concepts are described (corresponding digital files can be found in the accompanying CD-ROM).

4. In the Field research section, I describe the two instances of the prototypical system in practice and justify the choice of the course format in each case.
5. In the **Data analysis** section, I present the results of the two experimental cases above and group the different types of data gathered according to categories.

6. Finally, in the **Conclusions** section, I draw some preliminary conclusions based on the experimental courses described, preview possible future consequences, and suggest further work in the area.

7. The **Appendixes** present the names of students who took part in the experimental courses, the instruments used for data collection, and excerpts from the classroom journals. The accompanying CD-ROM contains the web sites and course materials from each case-study course, with most of the exercises presented by students, as well as the VBA applications I have especially developed to illustrate each of the prototypical system's concepts.

---

**I. Introduction**

As an architect and CAD instructor, I was able to make important observations that motivated this dissertation. Based on the assumptions that architecture students see CAD mainly as a drafting tool, have a bias against programming, and ignore the origins of CAD, I propose to change their attitude towards the use of computers in design. In this thesis I develop a prototypical CAD instruction system based on a computational design approach, inspired by Mitchell's (Mitchell et al., 1987) *The Art of Computer Graphics Programming* and Stiny's (1976) shape grammars. The system aims at encouraging architecture students to see CAD as a "design partner" (Negroponte, 1975a) and to make a better use of the features in which the computer is effectively good at. Two instances of the system were tested in different architecture schools in Brazil, leading to some preliminary conclusions about its efficacy.
II. Background

“All compositions of squares or of circles will be monotonous, and afford but little pleasure, because the means whereby they are produced are very apparent. So we think that compositions distributed in equal lines or divisions will be less beautiful than those which require a higher mental effort to appreciate them” (Jones, 2001, p.193).

II.a. CAD history

The development of CAD since its very origins to nowadays cannot be understood without looking at different fields evolving in parallel, but interconnected and affecting each other. For example, it is impossible to talk about CAD without citing the evolution of its practical medium, the computer. Computer Sciences, the theories behind computer technology, have received influences from many other fields (Figure 7). For example, the Boolean Algebra formalism from the 19th century made it possible to develop binary circuits; in the 1930's, British mathematician Alan Turing developed the concepts needed for the design of the machines; later, John von Neumann developed the concept of storing both instructions and data in the computer's memory. CAD itself has also received influences from different fields. Operations Research, Cognitive Sciences and Artificial Intelligence affected the development of CAD with "problem-solving" theory, wondering if the computer could think, and igniting the development of intelligent CAD. Although it is impossible to exhaust the number of interconnections or fields affecting CAD, it is still important to take into account such influences to understand how it has been developed, taught and used in architecture.

Areas that have influenced CAD

Computer technology has influenced the development of CAD since the very beginning. Hardware technology has always been a major constraint to CAD, both in terms of processing capabilities, memory and storage, and in what concerns input and output devices. Mitchell (1977) summarizes the history of four generations of computers from the 40's to the 70's. The very first, rudimentary electronic computer was developed with military purposes in the mid 40's. In the 50's, technical advances led to the first generation of commercially marketed computers, with vacuum tube technology and low speed.

The 60's saw the development of two new stages: a second genera-
tation of computers, from 59 to 65, with transistors and circuits assembled from independent components, allowing higher speed and the appearance of the first CAD systems; and a third generation, from 65 to 71, with integrated circuits, allowing better performance and miniaturization. Time-sharing was another important development in the third generation, dramatically reducing costs by allowing computers to be used by different operators simultaneously and opening the doors for the network concept.

The fourth generation of computers, starting in 71, was characterized by the large scale integrated (LSI) silicon chip electronic circuit technology. The large-scale IC (integrated circuit) was developed by Intel Corporation, which also introduced the memory IC, allowing the development of microcomputers and the computerization of small electronic equipment with standard components programmed to do a specific job. The high demand for microprocessors quickly reduced their cost, making their use possible in applications for which electronic controls were too large and expensive before (such as automobiles and industrial robots), and promoting an even further miniaturization and cost reduction. This period made possible the widespread use of computers in science, engineering, and architecture, and coincided with the introduction of personal computers.

Since the early 80's, a fifth generation of computers has been characterized by very-large-scale integration (VLSI) technology (Phiri,
II. Background

1999 p.10), substantially increasing the density of microprocessor circuits. It made possible the mass production of personal computers (not only for offices and universities, but also for schools and homes) and of different computerized, programmable devices, ranging from children’s toys to automatic teller machines (ATMs). Another characteristic of this generation is the use of optical technology, which has dramatically increased computers’ capabilities of data storage and retrieval.

Table 2 partially shows Phiri’s (1999, p.6) summary of the major changes in size, processing speed and memory capacity of computers with different technologies.

It is possible to infer from the table that the performance of computers has increased at an incredible rate since its first development. The same happened in what concerns miniaturization. For example, while the first ICs from the 60’s contained about 10 components on a silicon chip 3 millimeters square, by 1970 the number had grown to 1,000 on a chip the same size at no increased cost. By the mid-1990s, the larger microprocessors contained 20 million units on a chip less than two centimeters square. Since CAD demands a great processing power for geometric operations and lots of random-access memory (RAM) for the interactive display of the results, it is possible to infer how these changes have dramatically facilitated its development over the decades.

The introduction of extremely light but powerful portable computers (laptops, notebooks and palms) and wireless communications technology since the early 90’s has started to provoke new changes in CAD. Computers now can go to the building site, allowing architects to make automatic updates to construction documents, whereas construction teams can keep up-to-date with the latest changes in the office. Especial versions of CAD have already been developed.
II. Background

Figure 9: Storage tube graphic display (Mitchell, 1977).

for such situations. In what concerns the future, with the incipient development of cluster computing, processing power will not be an issue any more. The use of remote computers in parallel distributed processing will soon become a commercial reality.

Although most input and output devices have influenced CAD, historically graphic displays have had a major effect in the interaction with the user. The very first CAD applications used little graphical interaction and performed mostly mathematical calculations or data management. With the development of graphic displays, CAD became more interactive and increasingly graphic-oriented.

At the beginning of CAD development, the unavailability of low-cost graphic displays was one of its major constraints. Until the 50's, interaction with computers had been done via punched cards and printouts. In the 50's and 60's, different kinds of vector, stroke or calligraphic displays were developed. The first commercially available displays were refreshed cathode ray tubes. They allowed smooth movement of graphics, but were very expensive and showed some flickering due to the need to constantly refresh the display. This kind of monitor could be used with a light pen for picking functions or drawing directly on the screen. In the mid 60's, storage cathode ray tubes eliminated the flickering, but were unable to represent a smooth movement of graphics (Mitchell, 1977).

In the 70's, raster displays, based on inexpensive television technology, "contributed more to the growth of the field than did any other technology" (Foley, van Dam, Feiner, Hughes & Phillips, 1977 p.8). In such displays, the buffer storage was made in terms of pixels, whereas in the older vector displays instructions were needed to repeat the light strokes that produced the graphics. Nevertheless, smooth movement of graphics depended on extra memory (RAM), which was the only limitation to making raster displays the dominant technology. For this reason, vector displays were still used until the mid 80's, when the memory required by raster displays became affordable. Some of the advantages of raster displays over vector displays are lower cost and the possibility to display areas filled with solid colors or patterns. On the other hand, the major disadvantage of raster displays is the pixelation and jagged lines that occur during the conversion of vector coordinates into raster-scan mode. Nowadays, computer graphics is still concerned with anti-aliasing techniques.
While interactive displays, with the use of light pens, were present in the 60's, at the very beginning of CAD development, with the introduction of the mouse and digitizers they were almost completely forgotten. Yet, there has been a growing interest in the development of displays that allow easier interaction and simultaneous use by more than one person. The MIT Media Laboratory has been researching applications for interfaces based on images projected in the physical space rather than inside the monitor (Ishii, 1999). This trend will probably produce significant changes in CAD interaction in the short run.

In the same way that Computer Graphics is closely linked to the development of graphic displays, Computer Graphics and CAD are so intertwined that the definition of one sometimes overlaps the other. The major difference between them is that computer-aided design necessarily implies some kind of help from the computer in the design process (architectural design, in our case), even if just its graphic representation, whereas Computer Graphics is related to the synthesis of images in the computer, not necessarily for design purposes. In practice, the idea of CAD is generally linked to mathematical models based on vectors, which can be rendered as raster images, and Computer Graphics includes necessarily raster images, based on pixels, although often using vector manipulation for their generation.

More recently, telecommunication technologies have influenced the development of applications in two main ways: by creating the possibility of remote collaboration and by dynamically feeding CAD systems with on-line information from manufacturers. The first trend has led to the development of remote collaborative CAD tools that allow users in different places to share drawings at the same time as they can see and talk (or chat) to each other. This technology has made possible the development of big international teams of architects and consultants spread throughout the world, taking advantage of different local skills and in some cases even of the time difference to get projects finished on time. There are two types of applications in this category: some have the specific purpose of sharing information and promoting collaboration, while others were added to existing CAD tools, which now serve for both production and collaboration. A typical system of the first type includes real time whiteboard and chat, file sharing, conferencing, viewing and annotating raster images, vector drawings and 3D models. The second category adds to that a publish-to-web func-
tionality, using XML format to allow team members to drag-and-drop content directly to on-line files.

Another way in which the Internet has influenced CAD is by making available a huge amount of information from manufacturers. Many CAD applications now have Internet capabilities, being able to retrieve updated information from on-line catalogs, such as product specifications and symbol libraries. Even the CAD package itself can be updated through the Internet, by automatically having new extensions delivered through subscription.

CAD development

Since the 60’s, there have been two main fields of development in CAD (Gero, 1994): (1) the "representation and production of the geometry and topology of designed objects", related to Computer Graphics and sometimes accompanied by the automation of repetitive tasks; and (2) the "representation and use of knowledge to support or carry the synthesis of designs" (p.vii), first with independent, then with intelligent / support systems. While the first has seen widespread use in offices, reaching large publics through commercial programs, the latter has been usually restricted to experimental and specific applications. Those include generative systems and grammars, and knowledge-based systems.

Different facts in architectural theory and practice induced the second trend above, such as the Design Methods movement and prefabrication. The Design Methods movement of the 60's started with the translation of techniques from Operations Research into design. Although not exclusively concerned with computer-aided design, CAD was among its points of interest, because it represented a possibility of dramatically changing the way design was performed. The Design Methods movement had four main purposes (Gregory 1971).

1. To design better, by understanding the design process, a purpose derived from the generalized dissatisfaction with the modern movement
2. To allow repetitive parts of the design process to be automated by the computer
3. To set up strategies for designers in new areas, such as industrial design
4. To externalize the design process, allowing large teams to collab-
II. Background

orrate from the early stages, and allowing design to reach a higher level of complexity

In the search for understanding design better, the Design Methods movement was also influenced by the Cognitive Sciences, applying protocol analysis and rigid problem-solving techniques to systematize design, and trying to reproduce human behavior in CAD systems. The movement intended to use computers in all phases of design, from the conceptual phase, with the generation and evaluation of alternatives, to the more mundane production of building documents.

According to Negroponte (1975a), the influence of the Design Methods movement on CAD was extremely malignant, as it completely eliminated intuition and established a rational trend in the development of CAD software, which still persists. This line of research continued even after the fading of the Design Methods movement. The application of CAD to design synthesis became itself a field in Artificial Intelligence and Pattern Recognition since

---

**Figure 13:** Methods for the different phases of the design process, suggested by Jones (Jones, 1980).
the late 70's, referred to as intelligent systems, and producing its own international conferences. In the 80's, the Design Thinking trend tried to lay down a theoretical basis for understanding design and developing CAD that was less rigid than before. The concept of local optimizations instead of an overall solution led to the automation of repetitive drafting tasks in the production of building documents, instead of attempting to control all phases.

Prefabrication is another architectural phenomenon that has influenced the development of CAD in the 60's. Although the system dramatically reduced the possibilities of design, this limitation was seen as an advantage in an "ill-defined problem" context, with the impossibility of the available computers to generate an endless design search space. Eastman (1975) points out that industrialized construction permitted greater integration between schematic designs and the production of building documents because all possible details could be produced in advance. Despite being too specific and limited, this was a very practical approach, and such systems still have applications in the prefabricated building industry. However, with the use of CAD/CAM technology for producing building parts since the 70's, prefabrication is giving way to the manufacturing of custom-made parts that can be assembled quickly and with great precision, eliminating the need to restrict designs to a combinatorial problem.

In the same way that the Design Methods movement and the prefabricated building systems strongly influenced CAD in the 60's, CAD has also influenced design and architectural education. For example, the possibility of manipulating solids in virtual space, where they are not subject to the physical laws of impenetrability and gravity, has also influenced the conceptual phases of design. In fact, in what concerns architectural education, much importance has been given to the teaching of geometric modeling and its application in the design studio. Nevertheless, instead of what had been predicted, CAD did not replace by-hand drawings in terms of design method; it only replaced certain kinds of design representations, such as the need for orthogonal views (Eastman, 1992). Rendering and animation packages made possible the production of sophisticated, photo-realistic images even in the early stages of the design process, but the impact of such representations has been stronger in the real-estate market and in the treatment of surfaces than in design concepts.
II. Background

**Seven generations of CAD**

Mitchell (1990b) summarizes the progress of five generations of CAD since the early 60’s, to which I have added two more in the 90’s and one more in the 2000’s (see Table 3). Although the 50’s had seen the birth of CAD at an experimental level, the first generation
II. Background

only started in the 60's. Ivan Sutherland's SKETCHPAD, presented as a doctoral thesis at MIT in 1963, is often cited as the very first interactive CAD system, which combined data-manipulation with graphic displays. Despite Sutherland's introduction of the concept of ill-specified, "loose" inputs, following developments had a different orientation, eventually turning CAD into an overly precise tool (Negroponte, 1975a). The first large-scale CAD systems were developed for the automobile and aerospace industries. With commercial mainframes, tailored CAD applications were installed at a few architectural offices by the end of the decade. At that time, CAD systems were operated by technicians and had little influence in everyday life.

In the 70's, with more affordable 16-bit minicomputers, a second generation of commercially available CAD systems began. Those were expensive, turnkey systems, providing the hardware, software, installation, training, and technical support. These second-generation CAD systems were used in public construction with repetition in Britain, and by large engineering and architectural firms in the U.S., still operated mainly by technicians. Some layout programs of this period aimed to completely eliminate drafting by automating the production of documents (Richens, 1992). Apart from the limitations imposed by the cost of computers, architects were still suspicious about the use of computers in design. According to Broadbent (1973), their attitude positions ranged from fear to mystification and resentment that computers would diminish humanity in architecture.

The 80's saw the parallel development of three different generations of CAD, and an intention to realize its potential through new methodologies and practices (Kalay, 1987), such as architectural expert systems. Those were "rule-based and/or frame-based systems that provided the means for codifying the problem-solving know-how of human-experts" (Schmitt, 1987 p.213). The third generation was a natural continuation of the previous two: with the new 32-bit super-minis, more large engineering and architectural firms acquired turnkey CAD systems. They were also introduced to some architecture schools, but still as a secondary activity.

At the same time, a fourth generation of simplified CAD was being developed to be used in the new 16-bit IBM and Apple Macintosh personal computers, finally making CAD affordable to small firms and independent architects. The PC's had appeared in the early 70's,
II. Background

but were not powerful enough for CAD applications until the 80's. In 84, Apple introduced the mouse-and-windows style, to which Microsoft responded with the Windows operational system. The new interaction system has been responsible for the widespread acceptance of computers since then.

The simplification of CAD systems for personal computers provoked a radical change in the CAD culture, which Mitchell identifies as an inflection point in the history of architectural CAD, after which "the wider possibilities were largely ignored" (Mitchell 1990b p.483).

In fact, the new standardized, general-use CAD applications for PC's did not aim to help architects from the early, conceptual stages of design. Such specialized applications remained restricted to academic research, while the great majority of offices kept the use of CAD restricted to drafting and representation.

The fifth generation of CAD was related to the development of a new kind of computer in the 80's: the graphic workstation. For these machines, the developers of both mainframe and PC CAD adapted their products, at the same time as special applications were being developed. Graphic workstations were installed in medium and large firms, as well as in many architecture schools, sometimes combined with networks of PC's. Their 3D-modeling capabilities led to a discussion about the role of solids modeling in the design process and in the replacement of traditional bi-dimensional design representation (Eastman, 1987).

By the end of the decade, there was no substantial difference between PC's and workstations both in price and in performance, giving rise to what can be perceived as a sixth generation of CAD. During the 90's, CAD became not only a pattern, but also a necessity in every architectural office, in the same way as word processors completely substituted typewriters. CAD became a synonym of productivity. The introduction of 3D modeling, rendering and communication capabilities to the originally simplified PC CAD systems was made possible by the increasing power of microprocessors. As opposed to the first, standardized systems, the new ones had different levels of complexity and capabilities, as well as different prices, to fit specific areas, such as drafting, graphic-lay-out, modeling, rendering, and surveying (not to mention applications out of the architectural scope, such as movie-making, GIS and

Figure 17: Off the shelf, "do-it-yourself" CAD.
II. Background

Finally, still during the 90's, a seventh generation of do-it-yourself or home design CAD packages, destined to dwellers, rather than architects, has allowed users to design American style houses based on standardized building materials. Although limited, they offer much more freedom than the early pre-fabrication CAD systems from the 70's. Those systems not only perform the structural analysis of the wood frame and produce complete building documents based on standard details, ready for printing, but also make the "designers" very proud of themselves.

In what concerns future developments, a clear direction is being taken towards web-based and wireless CAD that allow remote collaboration and on-the-go work, which is a characteristic of the eight and last generation. Learning capabilities, which are already present in programs such as text editors, will also be incorporated to drawing editors. Another probable line of development relates to interactive interfaces. For example, data gloves have been used experimentally to transform spatial gestures in 3D meshes, and projected displays are an alternative to monitors. As mentioned above, the MIT Media Laboratory's Tangible Interfaces has been working in this direction.

The drastic change in the primary objective of CAD from a problem-solver to a drafting, representation and communication tool, but also a convenient medium to explore design ideas, probably has many reasons, which are related not only to technical and economical questions, but also to philosophical ones. One of these reasons could be the fact that each architect has a personal way of designing, and it would be extremely hard to develop a conceptual design system broad enough to fit every architect's personal methodology, or even to develop customizable enough tools that could be adapted to each person. Another reason is that, over the years, CAD became increasingly complex, making it impossible for architects to develop their own applications. It takes large teams of software engineers, programmers, graphic designers, and sometimes architects to developed a CAD system.

In these 40 years of CAD, what have we, architects, gained? Mitchell (1990a) asserts that the advantages of computer-aided design over traditional drawings are "intelligibility, memory and structure" (p.5). On the other hand, he points out the difficulty in
II. Background

Figure 19: Yufei’s 3Dshaper: an example of a specially developed generative CAD application.

using CAD beyond representation, i.e., in design exploration, due to the computer’s inability to recognize emergent shapes. As Negroponte (1975a) puts it, the reason for the failure of CAD as a design partner is the same as in the failure of automated translation programs: the lack of context:

“It is much easier to work on problem solving, decomposition, and if-then-because than to tamper with issues of learning and meaning (…)” (p.38).

While truly helpful systems are not available, we can still try to use (and teach our students to use) the computer in the best way we can, taking advantage of what it is good at, and developing a critical attitude towards what the market offers in terms of software. Besides, we can re-introduce some design thinking, design generation and personal methodologies to a certain extent by customizing existing CAD packages and producing our own little tools. As we saw, systems that aimed at completely automating the design process never reached widespread acceptance. That very fact probably explains more about how we design than any of such programs.

II.b. Computational Design

Nowadays, the word computational inevitably brings up the idea of something done with computers, but the machine is certainly not essential for computations to occur. So, if computational design is not necessarily a design made with the help of a computer, what is it? According to the Webster’s New World Dictionary, calculating, a synonym of computing, can have three different meanings:

1. To find out by using mathematics (simple arithmetic in the case of computing):

Traditionally, arithmetic - the science of numbers - has dealt with numbers. Yet, since design is typically expressed through shapes, analogical representations of buildings, computational design should also include the synthesis of form through shape computation.

2. To find out by reasoning, to estimate;

In computational design, compositions should not be obvious, but organized according to some hidden logic that makes their inter-
II. Background

pretation dependant on reasoning and inference.

3. To plan or intend.

Computational design should not happen by accident. A design algorithm, for example, is a carefully planned description of procedures. Similarly, a rule-based design, another type of computational design, may lead to surprising shapes as a result of intentionally defined rules.

Computation has been used in architecture in many different ways, that are exemplified in the following pages: in the synthesis of designs (generation), in their modeling (representation) and in their analysis (evaluation). This distinction could imply a design system in which solutions are generated, then represented, and finally tested, which is not my intention; the separation has been assumed just for practical reasons. The cases presented here do not attempt to exhaust the uses of computation in design, but to show some significant examples of its use in architecture.

Representation

Representation can happen as an independent art, but it is also a fundamental tool for synthesis and analysis. In most fields of science, since it is impossible to build or test directly the object of study, problem resolution is based on the use of models, with both analytical and synthetic purposes. Models are abstract representations of real entities in a descriptive language, which can be analogical or symbolical. In architecture, drawings are an example of analogous representations, whereas force diagrams exemplify a symbolical representation of the structural behavior of beams.

The boundaries between representation and synthesis can sometimes be very loose. For example, a generative system in design (synthesis) can be based on an algorithmic description (representation) of shapes, where some values have been replaced by placeholders. When these placeholders or variables are replaced by different entities (such as numerical values or shapes) from a predetermined range of values or collection, the representation is completed and a new design is created. Because representation is so tightly connected to synthesis, it is often impossible to prevent the influence of representation systems in design.
Similarly, certain kinds of analytical processes are dependent on specific systems of representation. For example, space evaluation (analysis) in architecture is often based on (and sometimes confused with) a representation system called perspective. The case of perspective as a representational system is particularly interesting. Although it is considered an analogical type of representation, it can be created by both analogical (when done by hand, through descriptive geometry methods) and symbolical means (when done by the computer, through analytical geometry methods). Yet, CAD software users are often unaware of the thousands of computations needed to display a perspective on the computer screen.

**Analysis**

Analysis or evaluation is done by comparison of models in a symbolic or analogical system. Representations of designs are tested against parameters they are supposed to conform. Consequently, analysis can only be performed after design alternatives or a final design have been generated. In architecture, typical examples are structural, energy, lighting and acoustical analyses. Although those have usually been modeled in symbolic systems, more recently computer programs have been used for visual representations of such characteristics of designs. While light is the most obvious of them, forces acting on a beam or the frequency of the sound reaching different areas of a hall can also be mapped with the use of color scales. Analogical evaluation systems include checking designs for graphic standards or building codes.

An analytical system can also be part of a generative system that includes some device for testing possible solutions. For example, in evolutionary computation a fitness function tests offspring in every generation. The samples that do not conform the expectations are terminated, while the good ones are allowed to reproduce, as in natural selection. When the test device detects a satisfactory solution, the generative process can be stopped. In other cases, a device used to analyze a particular kind of design, such as a shape grammar, can be used as a generative tool to reproduce designs in the same style. By analyzing different Palladian villas, and inferring common compositional rules, Stiny and Mitchell (1978) demonstrated that they belonged to the same language of designs. Then, by re-applying the same rules, they were able to produce new villas that perfectly conformed to the Palladian style.
Synthesis

The use of computation in the generation of designs, also referred to as computational synthesis, deserves special attention because it is intended for the conceptual phases of design. After all, representation and analysis depend on the generation of something to be represented and analyzed. Computational synthesis can be divided in two main streams: problem solving, more structured and rational, and explorative/generative design, with concerns with creativity and originality.

In the 60's and 70's, Operations Research, Cognitive Sciences and A.I. introduced a new discussion on the generation of designs, known as problem solving. The interest was derived from the discussion about the ability of computers to "think", i.e., to reproduce human intelligent behavior. Protocol analysis was used to formalize such behavior in different situations, from chess playing to architectural design. Well-defined parts of the problem-solving process of design were subject to optimization, usually involving linear algebra. Ill-defined parts required sophisticated heuristic search techniques, such as backtracking, depth-first search, breadth-first search, means-end analysis, generate and test, hill-climbing or simulated annealing.

Many of the techniques above were implemented in CAD systems, such as space-planning programs based on graph-theory, and expert systems, which incorporated knowledge from a database to help in solving design problems. Nevertheless, the interest and acceptance of such design-aids in the so-called ill-defined parts of the problem faded away over the years, while well-defined parts are still solved by optimization techniques.

Evolutionary computation is an interesting case in between problem-solving and explorative generative design. It consists in the generation of all possible solutions and the search for the best one with the use of fitness functions, which simulate natural selection. Genetic algorithms, an evolutionary computation technique, have been used primarily for problem solving and optimization. On the other hand, evolutionary computation can generate new "breeds" of designs through mating, crossing-over and mutation. These forms of genotypic description recombination can generate diversity, which occurs at random. Evolutionists sustain that the fast, inexpensive generation of multiple design alternatives allows designers
to explore "numerous creative solutions" (Bentley, 1999 p.2), leading to originality and novelty.

Besides solving problems, computation can serve as a theme or method for design exploration and the generation of shapes. Most of these methods, such as symmetry, mystical numbers and harmonic proportions, have been used for thousands of years, while others, such as evolutionary computation, would not make sense before the invention of the computer. Generative design is not necessarily derived from project goals, which can give it a sense of randomness. On the other hand, the use of techniques that carry some kind of logic in the solving of ill-defined design problems can drastically reduce the number of alternatives and help justifying design decisions.

Mystic numbers, such as the "magical" number seven, logical sequences, such as the Fibonacci series, and numbers observed in nature, such as the golden section and human proportions have been used for thousands of years. March describes some types of proportions used in the Renaissance: rational, empowered, musical and composite (March, 1998). Le Corbusier (2000) proposed a general proportioning system, the Modulor. Palladio (1965) specified proportions for each part of the building, from doors to rooms. In what concerns computer applications, the ability of most programming languages to generate random numbers has been used in combination with other techniques to generate designs. Negroponte (1975a) describes a sample problem used in computer classes for architects in the 70's that simulates the growth of a three-dimensional community with the use of probabilistic numbers.

The very fact of using geometry in design already implies a form of computation. More than simply a form of representation, the use of geometry and its instruments has significant implications in designs. For example, with the objective of simplifying the complex Beaux Arts' drawing processes, Durand's (1802) design method implied that even the most complex designs could be generated from geometric primitives.

The correspondence between analytic and descriptive geometries makes it possible from analogical to symbolic, and from symbolic back to analogical representations in CAD software. A mouse click is an analogical input that is transformed into numbers, so operations can be performed in the symbolic level, and then the results
Symmetric design explores different combinations of motifs or basic units. Symmetries have had a major importance in the decorative arts, from Moorish ornaments to the Arts and Crafts movement. Although bilateral symmetry, generated by reflection, is historically the most common type of symmetry used in architecture, other types can be generated by identity, translation, rotation, glide-translation, glide-reflection and roto-reflection, combined in frieze, wall-paper and circular patterns.

March and Steadman (1974) assert that modern art and architecture rejected simple bilateral symmetry, reintroducing it in more complex ways:

"Le Corbusier and FLW were those who most understood symmetry as an abstract idea, Viollet-le-Duc's principe. (...) [they] even bring new life to symmetry in many of their projects." (p.40)

Computer programs can help in the design of symmetric compositions by, for example, dynamically showing how changes made to the motif affect the different symmetric patterns based on it. The computer thus becomes an exploratory medium for symmetric designs.

Parametric design consists in varying certain dimensions of a shape to adapt it to specific circumstances. Parameters or variables can be seen as placeholders for different values that may be used, allowing variations of a basic shape and therefore amplifying applicability and originality. For example, open-dimension representations of details in Architectural Graphic Standards can be applied to any situation by parametric matching. An interesting case of the achievement of originality by this technique is Niemeyer's Mondadori Headquarters in Milan, a parametric version of his earlier Itamaraty Palace for the Ministry of Foreign Relations in Brasília. The former introduced unexpected results that make the latter look as a rather conservative building. In the 80's, Mitchell's (1985) computational approach of CAD in UCLA included exercises for the parametric variation of geometry.

Structured or grammatical design refers to the "grammatical combination" (Mitchell 1990c p.131) of parts. Even a finite architectural vocabulary (such as floors, walls, roofs) could be combined in an
infinite number of designs, most of which would have no architectural meaning, if there were no compositional rules. An architectural grammar can restrict the possible combinations of building parts by specifying certain relations between them, in the same way that syntax can identify correct phrases in a natural language. The hierarchical relationships between architectural elements can be represented by a tree diagram, in a similar way to the syntactic structure of phrases. Subordinated phrases have their correspondence in sub-assemblies of building parts, such as the base, shaft and capital of a column, each of which can be composed by a sub-sub-assembly of even smaller parts.

Durand (1802) presented a good example of grammatical design: in the first part of his book, he described the "elements" of architecture, such as columns, walls and floors. In the second, he described the relations by which such elements were supposed to be combined. Another example of the use of this kind of computation in design is found in applications for pre-fabricated building systems, where each part can only be connected to certain other parts in particular ways. In some CAD systems, such as ArchiCAD, elements are categorized by function (such as slabs, walls, doors) and each object follows rules of intelligent behavior. Yet, since the program can't predict all categories, those systems become harder to use than generic systems (Richens, 1992). Another critique is often made to grammatical design: in the same way that syntactically correct phrases are not necessarily semantically correct, the blind use of an architectural grammars does not guarantee good design.

Algorithmic design can be considered a kind of encapsulated representation. Fractals are a good example, and can be described both in terms of symbolic or analogical rules. Other classical examples of algorithmic design are instructions for drawing building parts in different architectural treatises, such as Palladio's "Four Books on Architecture". Computer languages provide a perfect medium for experimenting with algorithmic design. Nevertheless, as computer-languages grew more complex and sophisticated, and as the time required to learn them has increasingly grown, most designers now see graphics programming as a challenge.

As mentioned before, evolutionary computation, originally developed in the 60's, has been used both in problem solving and in explorative/generative approaches. Evolutionary design is based on the concepts of phenotype and genotype (an actual shape and its
algorithmic description) and the combinatorial operations between genetic structures. Frazer's (1995) Reptile System is a simplified example, using the concepts of "seeds" and "growth". It consisted of a flexible enclosure system made of two folded-plate structural elements that could be combined in eighteen different ways (notice the use of symmetry combined with evolution here). The unit was used to "grow" structures of different sizes and shapes. In a later experiment in the 90's, Frazer incorporated environmental factors to affect the development of the "genetic code" (in a direct analogy between the shapes' symbolic descriptions and DNA), a developmental system, an evaluation module, a genetic algorithm, and a graphic output mapping procedure. The developmental system reacted to the environment and genetic code, providing data for the genetic algorithm to make selections. Peter Testa's Emergent Design Group implements evolutionary programming techniques in a CAD package with C++ to develop novel architectural forms. In this case, evolutionary computation is combined with grammars and emergent design.

Ambiguous and emergent designs allow open interpretations. Therefore, they can defy hermeneutics, i.e., prejudices, biases and expectations observers carry when perceiving designs. Coyne (1991) quotes Gadamer when discussing about projection in the process of understanding architectural meaning:

"A person who is trying to understand a text is always performing an act of projecting. He projects before himself a meaning for the text as a whole as soon as some initial meaning emerges in the text. Again, the latter emerges only because he is reading the text with particular expectations in regard to a certain meaning. The working out of this fore-project, which is constantly revised in terms of what emerges as he penetrates into the meaning, is understanding what is there." (p.125)

An emergent shape is a novel shape that was not defined or thought of explicitly at the start of a design process. On the other hand, the word ambiguous means having two possible meanings. Ambiguity in architecture usually derives from emergence: when new shapes emerge from a composition, it suddenly gains new possible meanings or forms of interpretation. The process of finding ambiguous and emergent shapes involves a complex kind of intellectual computation, an activity that designers do naturally when hand drawing (Stiny, 1990). As Dewey (1934) puts it,
"The artist does his thinking in the very qualitative media he works in, and the terms lie so close to the object that he is producing that they merge directly into it." (p.50)

Nevertheless, contrary to the other cases of use of computation in design, it is very difficult to implement this technique in computer-programs, because it requires the recognition of arbitrary shapes not explicitly defined by the programmer. The process of interpreting emergent patterns, or pattern-recognition, has been a recurrent concern in the artificial intelligence field, not only applicable to design, but also to different kinds of data structures and complex systems.

Shape ambiguity can be both formal and functional. Mitchell (1990c) exemplifies the second case with Labov's differently proportionate containers. Between a bowl and a cup, as well as between a cup and a vase, lie shapes that are hard to describe with precision. Similarly, a parapet can be turned into a bench or a cabinet, giving new interpretations to its original function. When working with pre-defined architectural elements, such as in an intelligent CAD program, a parapet is a parapet, and a bench is a bench. A designer would have to explicitly create a new description of a parapet-bench before he or she could use ambiguous interpretations. Stiny (1990) cites Charles Percier's 1er Grand Prix, described by Van Zanten, as an example of shape ambiguity in design. In Percier's plan, the use of a modular grid allows multiple readings of rectangles within rectangles.

Shape grammars (Stiny, 1977) are a good example of a technique that can combine almost all kinds of generative computations. They can employ numerical sequences to order the application of numbered rules; they certainly can employ geometry; they use symmetry as a way of amplifying the number of possible combinations; they can employ parametric values and give them conditions to control results; they can be combined with optimization methods, such as shape annealing, to produce optimized, innovative shapes (Shea, 1999). Best of all, they are explicitly concerned with emergence and ambiguity. Apart from their generative use, shape grammars can also be used deterministically, as a description device, or in reverse engineering to determine the rules of composition of a corpus of buildings from a specific style or artist.
II. Background

In conclusion, there are many ways in which computation can be used in the design process. This chapter did not aim at covering all the existing ones, and new ones will certainly be developed in the future. Yet, it tried to show that the only use that makes design genuinely "computational" is the one that occurs in its conceptual phases, before representation or analysis, i.e., computational synthesis or generative design. The reflection on legitimate and illegitimate forms of computational design leads to the discussion whether CAD software should still be called "computer aided design" systems, since the word design means synthesis, whereas most commercial programs offer exclusively representational and analytical tools.

II.c. CAD in Brazilian architectural education

In Brazil, the use and teaching of information technologies in architecture schools began in the late 70's, with the use of programs to calculate structures and statistics. In the late 80's and early 90's, with the popularization of graphic software, as well as the possibility of applying such technologies to many of the existing subjects, the computerization of Brazilian architecture schools acquired a new character.

In 1991 a series of Symposia entitled "Computer Graphics for Architecture, Engineering and related Areas" launched a movement aiming at discussing the applications of Information Technology in design. Since then, many congresses and seminars have been devoted to the issue, like the First Multimedia for Architecture and Urban Design Symposium (FAU-USP, 1994) and the NUTA U'96 - International Seminar Technology/Architecture/Urbanism (FAU-USP, 1996). In 1995, a group entitled O ensino da informática na arquitetura (The teaching of IT in architecture) also started a new series of seminars.

Not until 1994 the Education Ministry published new recommendations for Architecture and Urban Planning course curricula (Portaria 1770, Diretrizes Curriculares Gerais) that included the teaching of information technologies as a specific professional subject, and required its use throughout the curriculum.

A recent master degree thesis presented at Londrina University (Duarte, 2000) has studied the present situation of CAD instruction in ten Brazilian schools (UFBA, UFRJ, UnB, FAU-USP, Belas
II. Background

Artes SP, UNISINOS, UFPR, UFMG, UFSC and UFRGS), through interviews with 20 professors. The thesis gives a comprehensive idea of the present situation of CAD instruction for architects in Brazil, which I have summarized in Table 4:

Duarte (2000) reports an interesting phenomenon that is consistently occurring in Brazilian architecture schools, especially at FAU-USP. FAU's building has large studio spaces (Figure 39) that used to be full of students during the weeks preceding studio reviews. Nowadays, they are usually empty, because students either stay in the school's small CAD lab or work at their own computers at home, which happens more often.

In summary, the last ten years have seen the assimilation of CAD in the architectural curriculum and its use as a means of volumetric exploration and technical representation in design. However, it seems that this change has not brought any significant improvement in terms of quality of design. This fact may have produced skeptical attitudes towards the use of CAD in design by some educators, and may be one of the reasons for the apparent disregard in what concerns CAD education in some Brazilian architectural schools. Although this thesis does not aim at proving that a computational approach to CAD can effectively improve the quality of designs (as stated in section I, sub-item Scope), it suggests that this
could possibly be one of the benefits of the new system proposed.

II.d. Research questions

Stiny (1976), citing March and Matela, suggests that "the study of form itself should again become a central enterprise in architectural education" (p.187). He claims for a rigorous, "scientific" study of form instead of "letting form simply follow function". It is not in the scope of this thesis to discuss the reasons why computational design theories, such as shape grammars, have not been systematically incorporated in the design curriculum of architectural schools. Nevertheless, the incorporation of such theories in CAD instruction could open the possibility of showing architecture students some of their practical applications. This new approach to CAD education would have been unfeasible a few decades ago, when the resources and skills it requires were out of the reach of most students. Yet, nowadays, with the widespread use of CAD software and computers in most schools, the conditions seem quite favorable.

In conclusion, the study of CAD history, computational design theories and the present situation of CAD in the architectural curriculum have led to the following research questions:

1. Would it be possible to mix CAD practice and computational design theory successfully in the same instruction system?
2. How would technical and theoretical issues be balanced in this system?
3. Would students exposed to this system change their attitude towards the use of CAD in architecture?
4. Would students exposed to this system be able to apply the new concepts in other subjects and studios?
5. Would the teaching of CAD with a computational design approach give it a new relevance in the architectural curriculum, reverting the skepticism that is starting to emerge?
6. Would the new approach have any positive influence in design?

The questions above are obviously not easy to answer. In the first place, different schools have different attitudes towards the use of technology. In some places, it seems like design technology and computers in particular will always occupy a peripheral position in the curriculum, no matter how much "philosophy" is added to them. Secondly, many years of research would be required for the observation of emergent patterns of change in design and style due to a
new design approach based on computation. Nonetheless, it is unacceptable that architecture students stay unaware of the "wider possibilities" (Mitchell, 1990b, 483) and keep using CAD below its potential.

Summary

In this section, the history of CAD technology has been summarized, taking into account the different fields that have influenced its development, such as computer hardware, telecommunications technology and design theory. Computational design has been defined and examples of different types of the use of computation in design have been presented. Finally, the present situation of CAD instruction in Brazilian architecture schools has been described, and initial research questions have been asked.
III. Methodology

“A researcher should be able to say a lot about a little, ...avoiding the tempta-
tion to say a little about a lot.” (Silverman, 1995 p.3)

III.a. Research methods in this study

In this study, as discussed in the last part of the Background section, some of the initial questions were too broad to be answered. On the other hand, the answers to other questions were too subjective to be tested, because there was no existing situation to be evaluated. On the contrary, a new situation had first to be proposed and then instantiated in order to be tested. Therefore, the study evolved in the following steps:

1. It started with a background information research;
2. As a response to the background research, a CAD instruction system that incorporates Computational Design theory was developed at a conceptual level;
3. Next, two instances of the prototypical system were tested in two different architecture schools (case-studies);
4. Then, the data gathered in the two situations were compared, categorized and analyzed;
5. Finally, preliminary conclusions about what can be expected from the application of the proposed system were drawn.

The first part of the study required secondary research methods (literature review) as a foundation for designing the prototypical system. The field study (or primary research) was carried out under a qualitative, rather than a psychometric approach, with low selectivity, but high intervention. It considered a limited number of individuals among the population of architecture students, which characterizes a case study, as opposed to a statistical study. In what concerns data analysis and conclusions, grounded theory methods were used to induce theory from the results achieved.

III.b. Sources of secondary research

Secondary sources were consulted for the definition of the methods to be used, and as a background foundation for the prototypical system proposed. This included sources on the history of CAD, on computational design, on educational theories and on the present situation of CAD in Brazilian architecture schools.

In what concerned CAD history and computational design, MIT
III. Methodology

libraries were used. In the first case, it is common knowledge that this type of software had its origins in the institute, and that MIT owns one of the best collections of sources of information in the field. Most books used for that part of the research dated from the 70's, such as Eastman's Spatial systems in computer-aided building design (1975) and Mitchell's Computer-aided architectural design (1977).

On the other hand, in terms of computational design, different currents have been developed mainly in different universities in the United States, the UK and Australia, varying from artificial intelligence in design and optimization methods to evolutionary design and shape grammars. The main source of information came from periodicals, especially Environment and Planning B and Design Studies from the 70's, 80's and 90's, and conference proceedings by design methods, design thinking, artificial intelligence and expert systems in design groups, and by the ACADIA and CAAD Futures societies. Besides, I have consulted some of the books that are considered milestones in the development of the computational thinking in design, namely:

- Alexander's Notes on the synthesis of form (1966);
- Akin's Psychology of architectural design (1986)
- Broadbent's Design in architecture (1973)
- Eastman's Explorations of the cognitive processes in design (1968);
- March's The Architecture of Form (1976);
- March and Steadman's The geometry of environment (1974);
- Mitchell's The Logic of Architecture (1990c);
- Negroponte's The Architecture Machine (1970) and Soft Architecture Machines (1975a)
- Simon's The sciences of the artificial (1998).

Besides, the inspiring influence from MIT professors, such as Terry Knight, William Mitchell and George Stiny - all of whom are indispensable references in the field as well - has to be acknowledged.

In what concerns education, sources started with Dewey (1934; 1997) and Piaget's (1950) constructivism and ended with Papert and Harel's (1980; 1991) constructionism and the work developed at the MIT Media Lab on computer-based learning. Another important reference in teaching methodology was Schön (1983; 1987), with his "reflection in action" theory. Research methodology
sources included material on qualitative methods (Denzin, 1998; Silverman, 1995), action research (Nunan, 1992), research information management (Orna & Stevens, 1995) and grounded theory (Strauss & Corbin, 1990).

Sources on the characteristics of Brazilian architecture schools, and more specifically on the schools where the system was tried, were gathered at UNICAMP and FAU-USP libraries. They included an M.Sc. thesis (Duarte, 2000) and conference proceedings of national meetings on Architectural education and CAD and computer graphics in architectural education.

III.c. Qualitative research

According to Nunan (1992), qualitative research emphasizes "process and meanings", not necessarily measuring them "in terms of quantity, amount, intensity, or frequency", while quantitative studies "emphasize the measurement and analysis of causal relationships between variables, not processes." While quantitative research is based on statistics and quantification, qualitative analysis "can be used to uncover and understand what lies behind any phenomenon about which little is yet known" (Strauss and Corbin, 1990).

Such was the case in the testing of the prototypical system proposed in this study; all the principal characteristics of qualitative research, listed below, were present here:

1. The field research happened in a real world context, instead of in an artificially constrained setting
2. The quality of the analysis was given more importance than the recruitment of the sample population
3. No prior hypothesis was assumed or tested; it was rather generated by the findings of the research.
4. Different types of data were collected.

As Nunan (1992) points out, the choice of a qualitative orientation does not discard the use of some supportive quantitative methods. In this study, questionnaires with both multiple-choice and open questions were used to assess students' attitude and beliefs. In the second case, the results were organized according to the categories that emerged from the answers presented. The display of the questionnaire results in charts should not be interpreted as a strategy to
generalize the findings, but rather as a visual, easy-to-understand form of justifying the conclusions, whose external validity limitations are acknowledged.

Besides, ethnographic techniques such as peers' participant observation in class and classroom journals for documenting daily activities were used. In-depth interviews were carried out with observers at the end of each case study.

### III.d. Case study versus experiments, survey and action research

Initially, it was rather difficult to define if the two instances of the proposed system should be characterized as experiments, surveys, action research, or case studies. By definition, an experiment is a research conducted in a laboratory, under controlled conditions (Strauss and Corbin, 1990), which was certainly not the case in the present study. In an experiment, the researcher "manipulates variables to determine their casual significance" (Nunan, 1992), which again was not the case here. According to Nunan's (1992) definition, a "true experiment", as shown in Table 5, should include "pre- and post-treatment tests", "experimental and control groups" and "random assignment of subjects". In the present study, pre- and post-treatment tests were applied only in what concerned students' attitude towards the use of CAD. Due to time constrains, it was impossible to assess other types of repercussion, such as influences in studio work in the long run. Control groups were out of question, given the qualitative character of the study, and teaching the experimental system to a significantly representative number of students would have been unfeasible, requiring a larger number of researchers and a longer time span.

Differently from an experiment, a survey is always conducted in a real-life context, and can be descriptive or analytical (Strauss and Corbin, 1990). The former normally uses standardized questionnaires for describing phenomena, while the latter uses qualitative methods to find associations between variables and explanations.
Both include a representative sample of the population. Although some survey methods - such as questionnaires and assessment forms - were used in the present study, they have not been applied to a representative sample of the population studied, and did not have a psychometric purpose.

Action research is the result of an intervening observer and/or an active practitioner (Hamilton, 1998). Punch (1998) considers action research as one of the important developments responsible for a recent change in research ethics, pointing out that in this type of study, "subjects are seen as partners in the research process...in a constructivist paradigm" that seeks to "empower participants" rather than just use them. Another aspect of action research present in this study is the "form of an ongoing cycle ... in which the teacher reflects on, returns to, and extends the initial inquiry" (Nunam 1990, 18). After the first instance of the proposed system was applied, such reflection guided the changes that were incorporated in the second instance. Yet, the present study was not initiated by "a problem detected in a classroom by a teacher", nor did it have a specific phase for "observation of the existing situation", two of the seven steps in action research proposed by Nunan (1992) and shown in Table 6. In other words, it cannot be considered action research either.

On the other hand, a case study implies one or more sample situations, as in the two courses taught in this study, that are investigated by an observer or, as in the present case, a participant observer. According to Silverman (1995), research based on case studies focuses in "the quality of the analysis rather than the recruitment of the sample." Therefore, the field research in this study can best be described as the study of cases or "instances in action" (Adelman et al., cited by Nunan 92) of the prototypical system developed.

### Table 6: Steps in the action research cycle (after Nunan 1992 p.19).

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
III. Methodology

III.e. Grounded theory

There are different approaches to qualitative research, varying from the simple presentation of data in a way that "speaks for itself", to "theoretical informed interpretations" that build theory from the data gathered. One type of such approaches is "grounded theory", which is defined by Strauss and Corbin (1990) as something "discovered, developed and provisionally verified through systematic data collection and analysis of data pertaining to that phenomenon". In that kind of study, "one does not begin with a theory, then prove it", but rather "one begins with an area of study and what is relevant to that area is allowed to emerge" (p.23)

The grounded theory methodology was originally developed by Barney Glaser and Anselm Strauss, being highly influenced by Dewey and having a strong emphasis on change and process, which are also important characteristics of the present study. Since the initial questions (see section II.d) were too broad to answer, instead of departing from a pre-defined hypothesis, a theoretical system was initially proposed, and experimental actions produced the data from which theory could be derived in the concluding section.

Table 7 shows the basic procedures in grounded theory defined by Strauss and Corbin (1990) and the corresponding sections in the present study. Although their sequence was not followed exactly the way it was proposed (due to the non-linear nature of the research process), the method used is very similar to theirs.

The data analysis was done in a way that is similar to the three-step analytical method suggested by Strauss and Corbin (1990), summarized in Table 8.

---

Table 7: Procedures in grounded theory according to Strauss and Corbin (1990) and sections of this thesis where they can be found.

<table>
<thead>
<tr>
<th>Section</th>
<th>I. Introduction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>End of II. Background</td>
</tr>
<tr>
<td></td>
<td>II. Background</td>
</tr>
<tr>
<td></td>
<td>IV. Prototyp. System</td>
</tr>
<tr>
<td></td>
<td>V: Case Studies</td>
</tr>
<tr>
<td></td>
<td>VI: Data Analysis</td>
</tr>
<tr>
<td></td>
<td>VII: Conclusions</td>
</tr>
</tbody>
</table>

---
Contrary to what is usually the case in the human sciences, the data collected here were very heterogeneous in nature, ranging from graphic material to classroom notes, but also in terms of the kind of information each could provide, such as students’ attitude change or interest in programming. Therefore, data analysis was done in the following steps:

1. Data were first organized in preliminary groups, according to their method of collection (e.g. homework, interviews, etc.);
2. The type of inferences that different kinds of data pointed to was listed, and data were reorganized according to them (sometimes the same type of data provided "clues" related to different suppositions, and therefore was included in more than one inference group);
3. Within each inference group, categories of answers and design exercises were established, and data were subdivided within them;
4. Finally, the occurrence of categories in each inference group was used as evidence to support them, allowing conclusions to emerge.

It is important to note that certain kinds of data unveiled clues about unexpected issues, which have also contributed to the conclusions. For example, students’ preferences when choosing the type of final project among the given options in the UNICAMP workshop clearly showed an increase in their interest in computer programming, which was not an original research hypothesis or question.

### III.f. Reliability and validity

In order for a qualitative study to be creditable, it needs to be conducted with the same rigor as any other scientific method. Some important issues on validity and reliability, described by the different authors consulted, are summarized in Table 9.

In what concerns reliability, the present study did use uniform procedures for data collection in each group studied, such as paper-based and web-based questionnaires for recording students' profiles and assess lectures, written interviews with observers - open yet
Based on standard outlines, and principally standard exercises that allowed comparing students’ performance. When an open-ended exercise was proposed, it was with the deliberate intent of observing what topics most interested students, without influencing them. Such was the case with the UNICAMP workshop’s final project proposal. The presence of experienced observers, witnessing everyday activities and taking part in them at the same time as students, teaching assistants and reviewers, was another strategy used to increase the reliability of this study.

When dealing with instruction systems it is important to keep close to the real world. Not surprisingly, researchers in language learning do a lot of fieldwork. While in the natural sciences lab experiments are a common way of strengthening internal validity, in the humanities - particularly in education - they often lead to a “situation in which outcomes can not be generalized to non-experimental contexts” (Nunan, 1992 p.214). In the present research there was a deliberate intention to test the proposed system in a real world environment. I acknowledge the impossibility to prove that the study can be replicated with similar results in other cultures. Nevertheless, it is possible to generalize the findings to a certain extent, when students present the same characteristics as those described in the Assumptions section of this thesis. Besides, by testing the system in schools with opposite orientations in what concerns the use of technology, it was possible to prove its replicability in different environments.

III. Methodology

<table>
<thead>
<tr>
<th>Reliability</th>
<th>Internal</th>
<th>External</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validity</td>
<td>Internal</td>
<td>External</td>
</tr>
</tbody>
</table>

Table 9: Validity and reliability issues.
III. Methodology

Summary

In this section, I have presented the methods that have been used to collect data from secondary and primary sources, and to analyze them. The research consisted of the development of a prototypical CAD instruction system followed by the study of cases where the proposed system was instantiated in two different Brazilian architecture schools. There was no random assignment of subjects or control groups. Rather, the field research had a qualitative approach and the data gathered were analyzed under a grounded theory methodology, in order to derive theory. Research reliability and validity were secured through the systematic handling of data, detailed descriptions of subjects and contexts, and peer observation. The conclusions of this study are expected to be replicable in situations where the same assumptions can be observed.
IV. Prototypical System

“It has often been said that a person doesn’t really understand something until he can teach it to a computer, i.e., express it as an algorithm.” (D. Knuth, cited by Stiny, 1975 p.313)

IV.a. Objectives

Back in the 50’s and 60’s, CAD was initially developed on solid theoretical foundations, but with commercialization it was progressively turned into an exclusively practical tool. On the other hand, computational design theories have been mostly restricted to the academic circles, and have not been systematically applied to design education, despite their great pedagogical potential. The present prototypical instruction system proposes an inversion of this situation: the study of CAD theory, and the application of computational design theories into practice, with the objective of demonstrating applications of CAD in architectural design that go beyond analysis and representation.

Starting from the assumption that most architecture students see CAD as a simple representation and analytical tool, the expected consequence of the proposed system is to change their attitude towards the possible applications of the computer in design. By seeing a variety of examples, and putting them into practice within a familiar environment, students are expected to be able to start seeing new possibilities in CAD.

Mitchell (1975) considers architectural design as a special kind of problem that may be more or less specified. When design constraints are clearly stated, the problem is well defined, and the number of possible solutions tends to one. On the other hand, if only a few, or no constraints are determined at all, the search for suitable solutions is considered an ill-defined problem, and the number of candidates may tend to infinity. Still according to Mitchell (1975), representation and analysis are among the least ambitious forms of the use of computers in design. A slightly higher level of ambition would be the automated generation of design alternatives based on well-defined criteria, which would be then evaluated by a human designer. Even more ambitious would be the generation and evaluation of well-specified designs by the machine. The highest level of ambition in the use of the machine would be the generation and evaluation of ill-defined architectural problems. The present system does not propose the computer implementation of such a complex task, which would inevitably
fall into a simplistic representation. Nevertheless, it introduces the topic to encourage a discussion on this very contentious issue.

Although teaching programming is not among the objectives of this prototypical system, the new approach to CAD proposed depends on some intervention on the software used by students. This intervention can be achieved with the use of programming, but also with the use of existing CAD commands in an unorthodox way or simple customization. Depending on students' technical skills and interest, programming can be used throughout the entire course, or it can be restricted to the topics where it is indispensable, such as in generative systems and design algorithms. The advantage of simply using existing commands in a new way in comparison to teaching inexperienced students to program is obviously the learning curve: the first method has many limitations but can be assimilated instantly, while the second has much more flexibility, but takes longer to teach. Another option is the use of ready-made programs specifically developed for each concept, so novice students will not be limited to using the simple applications they can develop. In this case, the programs should at least be explained and their computational logic should be made explicit to students. The three methods may also be combined, depending on students' background and experience.

**IV.b. A generic system**

The prototypical system proposed here aims at being a generic, abstract description of a CAD instruction system for architects that incorporates computational design theory. It is not specifically oriented towards graduate or undergraduate architecture students. Rather, it consists of a conceptual framework that can assume specific characteristics depending on where and to which public it is instantiated. Details such as which CAD software or programming language to use can be decided at instantiation time, depending on availability of software and students' previous experience. Yet, the CAD package chosen should allow a certain level of customization with scripting or programming.

Although highly desirable, the implementation on computers is not mandatory for all the exercises proposed. For example, when studying the algorithmization of design processes, it is perfectly acceptable to have students implement simple programs, and then develop complex ones only conceptually. Learning to describe an
IV. Prototypical System

architectural design procedure algorithmically is the most important step before it can be implemented, which can be done by interdisciplinary groups, not necessarily by the architect him- or herself. In addition, the system does not require a specific time span. According to the time available, it can assume the form of a short workshop or a full-semester course; what is important is that students have the opportunity to experiment with at least some of the hands-on exercises.

In what concerns pre-requisites, the proposed system does not assume any previous background in programming, but it does assume at least a basic knowledge of the CAD software that will be used for developing the proposed exercises. Adaptations to other fields of design, such as industrial and graphic design, can be easily made, by simply changing the examples given and the design exercises. The system can probably be useful to any field in which there is a search for an extended, meaningful use of computer technology in design.

IV.c. Pedagogical issues

Theory and practice have often been segregated in the teaching of computational design. Reading about heuristics, generate-and-search, graph theory, evolutionary design, shape grammars and other computational approaches to design may seem very interesting, but students may never apply such techniques in their actual work without proper training. On the other hand, just learning to operate a CAD tool or a programming language does not necessarily make an architect think computationally about design. Therefore, the first premise of the prototypical system developed here was to include both theory and practice, in an inter-related relationship, in accordance to Dewey's (1945) constructivist principle of "learning through personal experience". The objective of the proposed exercises is not only to put theory into practice, but also to prove how most computational design techniques gain efficacy, speed and complexity when implemented in the computer (even though most of them can be demonstrated and applied "by hand").

When theory and practice come along, there is the question of whether concepts should be presented in an abstract or concrete way. According to Knight, abstract approaches "do not limit possible interpretations of the designs they generate" (Knight, 2000). On the other hand, as she points out, novice architecture students "may
IV. Prototypical System

not be able to see beyond, or generalize from" abstract shapes. Although Knight was referring specifically to shape grammars, it is possible to extend her suppositions to the teaching of computational design in general. The concepts introduced in this prototypical instruction system are abstract ideas, but they become concrete through the discussion of their occurrence in the architectural repertoire. Similarly, practical exercises should evolve from abstract to concrete applications.

The proposed system does not imply any previous knowledge of a particular CAD package or programming language by students. Nevertheless, it should incorporate specific characteristics depending on students' background. For example, for students without any experience with computers, the system can include basic CAD and programming instruction so that students can develop simple exercises on each concept discussed. For experienced CAD operators or programmers, the course would obviously propose more advanced exercises to illustrate each concept, and in this case, CAD and programming basic instruction would not be necessary. The important point is to provide students with the needed technical knowledge, so they can be put the new concepts into practice, even if only in the most simple way.

Another important assumption of the proposed system is that it should always depart from the experience students already have. As pointed out by Dewey (1997 p.74) previous experiences should provide "the starting point for all further learning". For example, if students are already used to a specific CAD package for drafting, it would be helpful to start with exercises in which they use the very same software in a new way, while putting in practice the newly learned computational design principles. Already known commands may be used in an unorthodox way, which can help start changing the paradigm of the possible uses of CAD in architecture.

Next, more advanced techniques can be introduced, such as scripting in the same CAD package to make it even more powerful. As suggested by Dewey, "unless a given experience leads out into a field previously unfamiliar no problems arise, while problems are the stimulus of thinking" (Dewey 1997 p.79). Nevertheless, the introduction of programming for architects has to be done in a very "soft" way. Negroponte (1975a) suggests that, instead of long instruction periods without practical results, students should be able to immediately apply computer programming techniques
learned in short instruction sessions.

Besides Deweys' constructivist approach, Harel and Papert's (1991) constructionist principle of learning by building "personally meaningful tools" is also present in the proposed system. In this approach, creating design tools or environments that help the architect to design becomes an important part of the design process.

Another important pedagogical aspect of the proposed system is the concern with suggesting ways of applying concepts in the design of new architectural objects and in the analysis of existing ones. Shape grammars, for example, have been successfully used for both analytical and synthetic purposes (sometimes the synthetic use even deviates from a previous analysis) proving that one strategy is complementary to the other.

IV.d. Choosing concepts and examples

The present system consists of the progressive introduction of concepts that allow to relate the logic of architecture to the logic of computers. Nevertheless, it also deals with concepts that emphasize the importance of not-so-logic procedures in the design process, suggesting ways in which computers can help in this case as well. There is an inevitable overlap between the chosen concepts, which makes exercises seem redundant sometimes. However, this overlapping is pedagogically important because it reinforces, through different angles, the same fundamental idea: the abstract notion of computational design.

Table 10 shows the seven inter-related, fundamental concepts from computational design theory that are introduced in the system. The seven concepts are divided into informal sub-categories, as a strategy to help students relating them to examples from nature and the architectural repertoire. The concepts are introduced from the more specific to the less specific one, except for algorithmization, which is closely related to alternative-generation, requiring the use of similar programming techniques. For example, a symmetric composition can be generated by a recursive process, by design rules, by parameterized copies of a motif, by a generative system or by a design algorithm, and may allow the emergence of new shapes. Yet, designs generated by each of these methods are not necessarily symmetric. Similarly, recursive compositions can be generated by design rules, but not all design rules generate a recursive composi-
tion (because they can be applied in a free sequence), and so on. Finally, emergence may occur in almost any kind of compositions, notwithstanding the generative methods employed.

The concepts above, explained in detail in the following sections, are also key points in the study of shape grammars:

1. In Stiny's Kindergarten grammar (Stiny, 1980a) symmetry is a key factor when determining the number of different ways in which shapes can be combined.
2. In "The grammar of paradise" (Stiny and Mitchell, 1980), charbagh garden designs are obtained through recursive subdivisions.
3. Design rules are one of the four components of a shape grammar, as defined by Stiny (1980b).
4. Parametric shape grammars are a special kind of shape grammar, defined by Stiny (1980b).
5. If a generative system is a system that "can be operated to produce a variety of potential solutions" (Mitchell, 1975), then a shape grammar can be considered a generative system for shapes belonging to a specific family of shapes.
6. Knight (2000) states that "shape grammars are spatial, rather than textual or symbolic, algorithms" that generate designs.
7. Emergence, "the ability to recognize and ... operate on shapes that are not predefined in a grammar" is possible in shape grammars because they "treat shapes as non-atomic entities" (Knight, 2000).

Therefore, although the system proposed here does not intend to teach shape grammars directly, it provides some theoretical and practical foundation for that field.

Finding appropriate examples to illustrate each concept above was
not an easy task. Almost all the examples that suited one particular category had at least one or more features belonging to other categories - if not to all of them at once. On the other hand, this very difficulty was at the same time a proof that all the proposed categories were tightly connected. For this reason, examples were chosen according to their capacity to emphasize each concept studied. Each concept was also divided in sub-categories, and specific examples for each of them were recruited from nature, the decorative arts, and the architectural repertoire.

Finally, small programs developed by me with the purpose of demonstrating practical applications of each concept are presented at the end of some of the following items. The programs were written in Visual Basic for Applications (VBA) programming language in AutoCAD's own development environment, VBAIDE. They are embedded in DWG files and can be loaded in AutoCAD 2000 or compatible versions. All the sample programs are included in the accompanying CD-ROM (see Appendix E: Accompanying CD-ROM).

IV.e. Symmetry

Symmetry is one of the oldest computational concepts in architecture. Yet, contrary to what most people believe, it is not restricted to bilateral correspondence. Vitruvius defined symmetry as "the proportioned correspondence of the elements of the work itself, a response, in any given part, of the separate parts to the appearance of the entire figure as a whole" (Rowland and Howe, 1999 p.25). In algebra and geometry, symmetrical operations are those that introduce no change when applied to a number or shape. Similarly, in design, patterns can be generated by the recursive application of symmetrical operations. In The geometry of environment, March and Steadman (1974) describe different types of compositions generated by rotation, reflection, translation, and combinations of the above, and how architects have used them throughout history.

Examples of symmetry can be easily found in nature and architecture (Figure 42 and Figure 43). Bilateral symmetry, the most common type, is present in most animals, as well as in classical buildings. Row house plans present the same type of frieze symmetry that can be observed in some climbing plants. "Cobogós", a frequent element in Brazilian modernist façades, resemble the wallpaper symmetry in a snake's skin pattern, and gothic roses have the
IV. Prototypical System

Note: Technically, bilateral symmetry is a special kind of dihedral symmetry with a rotation of 0°. Yet, it will be treated as a special type of symmetry here, due to its common association to the concept of symmetry in general.

The reason why symmetric compositions have been so common throughout time and the different cultures is a question commonly raised by students. Explanations have been suggested, for example, by Jungian psychology. According to the German psychologist, concentric designs - also called mandalic structures - are often related to the search for an interior order (Jung, 1964).

In most CAD systems, symmetry can be easily obtained through the use of operations such as mirroring and copying objects in rectangular or circular (sometimes called polar) arrays, both in 2D and 3D. Yet, such systems do not include built-in features that allow dynamically experimenting with different symmetric arrangements. For this reason, as a practical exercise, students could be asked to develop strategies for using CAD to explore symmetric design. One possible way for this is through the use of programming. Nevertheless, it is also possible to prepare a dynamic design environment without much effort through the simple unorthodox use of standard features of an existing CAD program. A good example is the arrangement of AutoCAD viewports (polygons containing views of the model that can be individually set to different zoom factors, perspective angles, viewpoint rotations, etc.). In this example, rotated and mirrored views of the same model were overlapped (top) laid out in regular patterns (middle and bottom), producing circular (top left and center), bilateral (top right), frieze (middle), and wall-paper (bottom) symmetry compositions. In this configuration a motif - a basic unit that is repeated to generate a pattern - can be modeled inside one of the views, while the other views are dynamically updated.

IV.f. Recursion

In mathematics, a recursive function is characterized by the use of variables that are generated during the process, through the repetitive application of a routine. Recursive operations with shapes can include different types of transformations that are cumulatively applied to each new added element. If the transformations are restricted to translation and/or rotation, the resulting compositions
IV. Prototypical System

Figure 46: Symmetry Assistant

Figure 47: Innovative use of AutoCAD viewports for symmetric design.

Figure 48: Different types of recursive compositions: concentric and branching.

Symmetry Assistant

In order to demonstrate how CAD can help in design beyond simple representation, I have developed an environment called Symmetry Assistant (Figure 46). The environment consists of a pre-set AutoCAD file that includes overlapping viewports with mirrored and rotated views at different angles. The environment allows seeing multiple circular symmetry versions of any shape drawn in the upper left viewport. The effect is similar to a kaleidoscope, with the difference that it is possible to simultaneously see different symmetric compositions based on the same motif. Any of the resulting designs can be "frozen" and saved as a separate file at any given moment, which is done by an embedded VBA routine.

may have the same characteristics as some types of symmetries. Nevertheless, if scaling is also used, the result will not be symmetric. Similarly, in shape grammars, a transformation rule may be recursively applied to an initial shape to generate new designs, which characterizes a "set grammar".

To help students identify the different occurrences of recursive processes in architectural design, the concept of recursion can be subdivided into some informal categories. For example, concentric recursion can be obtained with the exclusive use of scaling around a central point. Branching does not necessarily need scaling, but needs that at least two new instances of the original shape be inserted. Internal subdivision implies successive division into smaller parts inside the original shape, necessarily including scale and translation transformations. Nuclear recursion can be characterized by the insertion of multiple, smaller instances around the original shape, resembling new nuclei, with scale, rotation and translation transformations. Figure 48 and Figure 51 show some natural and architectural examples of the categories listed above: recursive subdivision in gothic windows and ferns; nuclear recursion in the Garden City and dandelion; concentric recursion in gothic door entrance and tree rings; branching in Calatrava's gallery structure and trees.

Fractals can be cited as examples of the branching and the subdivision categories above. Fractals, such as the Koch curve and island (Figure 49), can be generated by a recursive process based on a single rule composed by an initiator and a generator (Batty, 1994).
In order to allow students who are less experienced in programming experimenting with recursive design, I have developed a special VBA application (Figure 50). The program allows to define a single rule based on a spatial relationship between an initial shape and any number of instances of it. The rule is recursively applied a number of times defined by the user, who has the options of keeping or deleting old generations.

The program is based on a very simple principle, which can be easily understood even by novice programmers. When the initial shape is defined in the appropriate viewport, in Layer 0, it is converted to an AutoCAD block named "Shape1". When the user inserts instances of the initial shape in the Spatial Relation viewport, he or she is actually inserting the AutoCAD block at specific insertion points, equal X, Y and Z scale factors (which means that reflections are not possible) and rotation angles, in a special layer called "Shape2", which is automatically made current. When the recursive process is run, the code searches for any instances of block Shape2 in Layer Shape2, and groups them all in an array of objects.

For each block in this array, the code will do the following:

· Match the coordinate system's origin with the block's insertion point;
· Match axis Y's rotation angle with the block's rotation angle;
· Insert the same number of blocks in the array, at the same insertion points and rotation angles, but multiplying their scale factor by the index of the current derivation.
· Proceed to the next block in the array.

After the last block in the array is used, the code will do the following:

· If the user chose to delete parents, all the blocks in the array are erased and replaced by the newly inserted blocks;
· Else, the new generation replaces the old one in the array, without erasing them from the drawing.

The sequence above is applied recursively until the specified number of derivations specified is reached.
Batty (1994) notes that an interesting aspect of the Koch curve and island is that they challenge Euclidian geometry, because while the curve's perimeter tends to infinity, the island's area is bounded within an almost circular limit.

Recursive functions (disregarding those that create symmetry, such as circular and rectangular arrays) are usually not present in CAD programs. Nevertheless, they can be added with scripting or programming. Another way of demonstrating a recursive process in a CAD program is by creating nested blocks (Figure 90). Any change in the block definition will be recursively applied to all the different levels of block insertion, in a process that resembles Stiny and Mitchell's (1980) Mughul Garden grammar.

**IV.g. Design rules**

The concept of design rules is directly related to the concepts already introduced in the previous topics. For this reason, students should be encouraged to compare the rules that generated the recursive and symmetrical compositions seen so far. This concept is also important because it makes a transition between the mechanical application of a single rule, and the more complex process of intentionally choosing rules from a collection and applying them to generate a specific design. In other words, it is important to make a clear distinction between pre-set and free sequences of rules, and the consequences in terms of computer implementations. With free sequences, instead of running on its own, the computer needs constant input from the user.

The relevance of the study of design rules in terms of design education is clearly stated by Stiny (1980a):

"Rules shift the emphasis in design away from individual designs to languages of designs. By thinking about rules, a designer can concentrate on the constructive basis for his designs and thus develop an explicit awareness of their properties and structure. This understanding allows the designer to use his past experience in new situations" (p.461).

In what concerns the role of CAD in design, pre-stored rules can help the designer take advantage of his previous experiences in new designs, combining them in new ways. Computer programs can also apply the rules under new sequences or even under any possi-
ble sequence. They can be considered, thus, generative systems. Stiny and Mitchell's Palladian grammar (1978) provides a good illustrative example of how a large number of rules can be used to generate different designs in the same language (Figure 52).

In most CAD packages, it is possible to define simple design rules with the help of a programming language and to apply them to selected shapes. The question of how a shape is interpreted, however, is a complex one, involving emergent shape recognition, a concept that will not be discussed until the end of this chapter, and that requires advanced programming techniques in order to be implemented.

IV.h. Parameterized shapes

In mathematics, a parameter is a variable for which the range of possible values identifies a collection of distinct cases in a problem. Similarly, in design the use of parameters - or parametric descriptions of shapes - may increase the number of possible variations of a shape.

Parameters are also a fundamental concept in the definition of families of architectural elements. The definition of shapes based on topological spatial relations and parameters can be considered a generative system: by changing the values of each parameter it is possible to explore the variations of the shape.

Parameters have been present in all the shape-generation proce-
IV. Prototypical System

Stiny's comparison between shape grammars and parametric shape grammars (Stiny, 1980b), provide a good distinction between the simple application of design rules and their more powerful parametric variations. According to him, "shape grammars are best used to define languages of shapes with proportional relationships determined by arithmetic or geometric series", while "parametric shape grammars can be used to define languages of shapes with proportional relationships determined in any way whatever." As in fractal geometry, parametric grammars challenge Euclidean geometry, introducing unexpected elements in the design process.

To help students search for examples of parametric shapes, the concept can be subdivided in three different categories:

1. Parametric variations of whole objects;
2. Parametric variations of an element within a single object;
3. Parametric variations of an element that appears in different kinds of objects.

Figures 53, 54 and 57 show examples of the three categories above in architecture and nature. The first one compares Brazilian modernist building profiles to the skeletons of different dog breeds. In both cases, the very same structure is instantiated at particular proportions, generating similar, yet different individuals. Figure 54 shows three different examples of basic elements that are repeated,
at different parameters, within single individuals. In nature, basic units can form the petals of heliconia’s flowers, the internal divisions of seashells, or the multiple legs of small animals. Similarly, basic forms are repeated in Aalto’s Neue Vahr apartments, Calatrava’s Lyon airport station trusses, and F. L. Wright’s spire elements. Note that, while the different shapes in heliconia’s petals and Aalto’s apartments display some randomness, the sizes of the shapes in the other examples increase in an orderly fashion. Figure 57 shows examples of parametric variations of the same type of element in different individuals, such as the human eye. Similarly, Louis Kahn instantiated his typical windows in different buildings, with flat or arched lintels, longer or shorter underneath slits, and different overall proportions.

In most CAD programs, user-defined components (such as blocks or cells) can be inserted in different positions, with different rotation angles and X, Y and Z scales, generating parametric variations of the original shape. Nevertheless, this strategy results in an overall scaling of the object, and does not allow changing different features of the object independently. For the parameterization of complex shapes, programming techniques can be used.

After finding out the limitations of X, Y and Z scale variations of a pre-defined shape, students should be encouraged to develop small parametric design applications. This type of exercise should start with the description of an architectural object in terms of topological relationships (e.g. line a ends where line b begins). Next, the parameters required to instantiate the conceptual shape should be identified, and constraints for each of them established. Finally, families of shapes could be created by instantiating the shape at different parameters.

**Parametric Arches Explorer**

I have developed two simple VBA applications for AutoCAD to demonstrate the parameterization of an architectural shape.

Parametric Arches Explorer helps designing arches with different proportions, as in Niemeyer’s Mondadori building (Figure 55). The difference between using this application and simply inserting AutoCAD blocks at different X and Y scales is that in the former the user can control a higher number of variables than in the latter, such as height of the impost and column width.
IV. Prototypical System

Figure 56: Modernist Profile Explorer.

This application allows creating 3D models of buildings based on Reidy's Museum of Modern Art at different parameters, in the same way as Reidy himself and Artigas did (Figure 56) to design new buildings.

**IV.i. Generative systems**

This topic is closely related to the previous ones, because some of the possible ways to generate design alternatives are by different symmetric arrangements, by the application of rules in different combinations, and by the variation of parameters within ranges of values. The grammatical system developed by Durand (1802), in which buildings are composed by parts that can be combined according to pre-established principles, is a typical example of a generative system from the history of architecture.

According to Mitchell (1975), the objective of a "problem solver" is to "obtain some existing tangible object" by "firstly obtaining an appropriate candidate object and secondly verifying that it matches the given goal description." The problem of course is harder to solve when the "candidate" does not pre-exist, and needs to be produced, so "the question of how potential solutions may be produced for consideration arises" (p.128). This can happen through a trial-and-error strategy (which designers usually employ), or by the systematic enumeration of solutions by a generative system (exhaustively, or from a constrained number of possibilities, which drastically reduces the number of candidates). While there may be a single solution to well-defined problems, ill-defined problems can usually be satisfied by a large number of different solutions. In that case, choosing the best option may be a question of subtle aesthetic judgment, which is better performed by a human designer.

Table 11 summarizes Mitchell's levels of ambition in the division of tasks between a human designer and a machine. Rows 1 and 2 (representation and analysis) are the most common uses of computers in architecture and civil engineering. All the other uses are still very rare. As the author points out, "the most ambitious potential level of use of the machine is to attempt to develop systems capable of dealing intelligently and flexibly with ill-defined problems, that is of displaying the capabilities characteristic of a good
human designer." (p.148). The implementation of an application with such specifications will not be discussed in this prototypical system, because it would imply the use of advanced programming and A.I. techniques. Besides, the practical applications of such systems have not even been demonstrated so far.

The objective of the exercises developed for this topic is to develop pedagogical examples of systems with what Mitchell considers a "medium" level of participation of the machine in the design process. Although some CAD programs already include pre-set routines for creating building elements (such as staircases) at given parameters, none of them can enumerate all their possible variations within sets of constraints. For that purpose, students should be encouraged to develop special applications with CAD scripting or

<table>
<thead>
<tr>
<th>Level of ambition</th>
<th>Smallest</th>
<th>Medium low</th>
<th>Medium</th>
<th>Medium high</th>
<th>Highest</th>
</tr>
</thead>
<tbody>
<tr>
<td>Criteria for generation</td>
<td>Well- or ill-defined</td>
<td>Ill-defined</td>
<td>Well-defined</td>
<td>Ill-defined</td>
<td>Well-defined</td>
</tr>
<tr>
<td>Criteria for evaluation</td>
<td>Well- or ill-defined</td>
<td>Well-defined</td>
<td>Ill-defined</td>
<td>Well-defined</td>
<td>Ill-defined</td>
</tr>
<tr>
<td>Examples</td>
<td>Data-base systems, computerized perspectives, working drawings, specifications, etc.</td>
<td>Structural, thermal, acoustic, lighting, cost evaluations</td>
<td>Systematic enumeration of floor plans</td>
<td>Optimization programs</td>
<td>Negroponte’s Architecture Machine</td>
</tr>
</tbody>
</table>
programming. Students should also try modeling similar solutions directly, before using their computer applications for the generation of alternatives, in order to compare the "manual" and automated systems. By transforming a common CAD tool into an alternative-generator, students are expected to completely rethink the possible uses of this type of software. A discussion about which are the best methods for searching for solutions in different architectural situations should also be raised.

In summary, it should become clear with this topic that no magic is possible when creating designs with the help of a computer program. A framework must be previously defined, so that variations can be generated within some limits and then selected. The present topic should end with a discussion about the "division of tasks between human designer and machine" (p.127), as proposed by Mitchell (1975).

**IV.j. Algorithmization of design procedures**

Design algorithms were used in the previous topic with the purpose of generating design alternatives, when design criteria were partially defined. Here, they are used in the automation of perfectly well-defined design problems where criteria, assumptions and default values have been previously established. In this type of situation, as Stiny puts it, "languages of designs defined by rules need not lead to the combinatorial difficulties of far too many possibilities, ...

**Alternative Generator**

This is a pedagogical example of an application that enumerates design alternatives (Figure 59). In this application, three different methods are available to explore different alternatives for a composition with three oblongs (Figure 58):

- Slider bars that dynamically update the composition (trial and error);
- A button that generates all the different combinations of proportions, with dimensions varying between 1 and 2 units (exhaustive enumeration);
- A button that generates all the different combinations of proportions, with dimensions varying between 1 and 2 units, as long as the central block is larger than the two other blocks (selective enumeration).
Although teaching design directly is not among the objectives of this study, the present topic inevitably leads to a formalization of the steps involved in a routine design situation, such as the determination of objectives and constraints, and the definition of if/then rules. This kind of procedure gives consistency to design. Architecture students often find it difficult to express design intentions verbally, but these must be made explicit before they are converted into algorithms. Palladio's (1965) first book of architecture provides a good example of detailed explanations of different design procedures.

An algorithm is a systematic procedure that can produce - in a finite number of steps - an answer to a question or the solution of a problem. It is also similar to a parametric equation, in the sense that it can generate different results according to the parameters fed to it. According to Stiny, "because it must be specified in such detail, just the attempt to construct an algorithm for a given process provides an excellent means of exploring the process in all its aspects and features" (1978 p.208).

Back in the 60's, CAD and design-methods researchers used protocol analysis to find out how architects thought, with the purpose of emulating the design process in the computer. As with any ill-defined problem, it is a very complex task to completely automate design. Yet, some specific steps in the process can be easily turned into algorithms and performed by a CAD program. The enterprise starts with a reflection on how each decision is taken, the possible
alternatives and the range of values parameters can take.

It is possible to define algorithms that generate symmetric designs, recursive designs, or even that automatically generate parametric versions of a shape. Yet, the algorithms discussed here are concerned with concrete design operations, the type of procedure that we often need to repeat day after day to solve the same design problem in different contexts. As an exercise for this topic, students should be encouraged to look at a simple design process or a specific part of a more complex design process, and try to carefully describe it. They should start by determining:

1. The variables of the problem;
2. The constraints to each variable;
3. The initial conditions assumed;

Next, the sequence of steps that generate the design must be described, incorporating, if necessary, conditional and repetition structures (such as if/then and do/while loops). All the possible solutions must be taken into account. Figure 61 shows a graphic representation of the sequence followed in the determination of a straightforward restroom layout design. Finally, the code's efficacy must be tested in different situations, and an interface capable of capturing the user's particular needs for each case, and at the same time suggesting standard values, must be developed.

The method described above is naturally used by architects in routine design problems in which the same type of procedure is applied to different space situations. Automating this type of procedure can free architects from repetitive tasks, leaving more time for the creative ones. It can also help them achieve more consistency in large

IV. Prototypical System

Figure 61: The graphic process of laying out a restroom plan, step by step.
IV. Prototypical System

Figure 62: Restroom Layout Assistant interface.

Restroom Layout Assistant

This VBA application automates the process of drafting a restroom layout in a very simple, pedagogical way. The design algorithm is based on the formalization of the design procedures an architect would follow in a similar situation. The design assumptions, criteria and variables in the program are:

Assumptions:
The space will be always rectangular;
The door will be always close to a corner.

Criteria:
When entering the room, a person should not see the stall's doors;
When entering the room, a person should see the sinks;
Stalls and sinks should be arranged against each of the longer walls;
In case the plan is square, the walls perpendicular to that containing the door will be considered the longer ones.
The first stall starts at a given distance from the entrance door, forming a small hall in front of it;
Stalls will have a constant width, but any left over centimeters are added to the stall that is next to the restroom's entrance door;
The number of sinks is the same as the number of stalls that can be fit in the space;
Sinks start being inserted at a given distance from the wall opposite to the entrance hall, with a fixed distance between each other.

Variables:
L = restroom width
C = restroom length
P = passage in front of the entrance door and in front of the sinks
1,2,3,... = door position
l = stall width
c = stall length
p = counter depth
d = distance between sinks

Design constraints:
3m >= L >= 4m
3m >= C >= 7m
1.20 >= P >= 0.80
Default values:
p = 40/45/50cm
l= 90/100/110cm
c = 100/120/130cm
d = 50cm

In the version included in the accompanying CD-ROM, the definition of some variables was left for students to complete.
projects involving different designers but similar situations. This topic may be a good opportunity to introduce programming techniques to students not familiar with them, although conceptual programs are also a good way to start thinking "computationally" about design. By writing scripts or linear programs within a CAD tool instead of developing applications from scratch (and therefore taking advantage of many commands already available), this type of program becomes feasible for most architects. Novice programmers can start by developing this type of simple applications, and then be encouraged to modify them, adding more variables and decision nodes (code loops) to the process.

Algorithmization exercises should always be followed by a discussion about the advantages and disadvantages of developing computer programs for routine design tasks, and how they can incorporate optimization and style objectives. Optimization can be achieved with the inclusion of mathematical functions, while stylistic options may be based on different pre-set routines from which users can choose, depending on the style wanted.

**IV.k. Emergent shapes**

Emergence in design can be described as the ability to define novel shapes that emerge from the addition, subtraction and intersection of other shapes. Emergence is often the method by which architects reach innovation: they can take well-known shapes, combine them in unusual ways, look at them from a different perspective, perceive ambiguities, and then suddenly start seeing something new. In what concerns interpretation by others, different possible readings - taking into account either the basic units or the emergent whole - can make designs more interesting. Figure 63 shows an emergent star resulting from the intersection of structural roof beams. Figure 64 shows how the same shapes can be present in different Moorish patterns, depending on how they are interpreted.

Finding emergent shapes is not always so quick and evident, even for the trained eye. As Mitchell (1990) points out "not surprisingly, then, computer algorithms for performing similar tasks turn out to be computationally complex" (p.102) Much has been speculated about possible computational methods for finding emergent shapes, which has been a frequent preoccupation of the Artificial Intelligence and Pattern Recognition fields. Having architecture students, usually programming novices, search for a solution to this
problem would probably be a frustrating task. For this reason I believe that, instead of trying to develop programs that implement this concept, students can use standard CAD commands.

Although the process described above happens naturally for most designers, students can be trained to find emergence in different situations, and CAD software may be an appropriate environment to do it. One interesting exercise is to use 2D pattern filling or 3D solid operations on top of symmetric or rule-based compositions developed in the previous lessons. Figure 66 shows an example in which different new shapes were picked up from a rule-based composition created with 2D Rules Assistant, while Figure 65 shows a form resulting from the application of Boolean operations on a composition generated by the 3D version of the same program.

It is not by accident that the topic of emergence is left for last in the proposed system. As proposed by Yakeley (2000), I also believe that it is possible to have ambiguity of interpretation in the computer, despite its "unambiguous nature". In the exercise proposed above, students intervene in the designs created by their own rules in order to find novel shapes the computer would never "see". Maybe, as Yakeley suggests, the "detachment from the resultant image encourages freer interpretation." A comparison between finding emergent shapes in a CAD program and with by-hand methods, such as pencil drawings and clay models, should be encouraged. While the machine is faster and allows more complexity and precision, hand drawings and models can have loose boundaries that allow different interpretations. Once more, the stage is set for a final discussion about the roles of the machine and the human designer within the same design process, a process in which both contribute in a unique way, and the computer becomes a real "design partner" (Negroponte, 1975a).

IV.1. Final discussion or project

The proposed instruction system should end either with a final project, in which students can have the opportunity to relate two or more of the concepts learned, or with a wrap up discussion about the role of CAD in architecture. In the first case, a discussion based on students’ projects should also be encouraged. Final projects can range from computer implementations to conceptual programs, with alternative-generation, automation or shape-exploration purposes.
IV. Prototypical System

Summary

In this chapter, I have developed a prototypical CAD instruction system that incorporates computational design theories, putting them in practice with a programmable CAD package. The system is based on seven basic concepts that are present in different computational design theories: symmetry, recursion, parameterized shapes, design rules, generative systems, algorithmization of design procedures and emergent shapes. Architectural examples and exercises on each concept have been described, and some applications developed by me have been presented. I have also addressed pedagogical aspects of the system, stressing the importance of creating practical applications for each concept, and the need to ignite discussions among students on the roles of CAD in architecture.

Figure 66: Different shapes "picked up" from a 2D rule-based composition.
V. Case Studies

"The important pedagogical point is ... simply the rapid return on a small investment of time and knowledge... this immediacy is crucial in an introduction to computer sciences." (Negroponte, 1975a p.191)

V.a. Two instances of the prototypical system

In order to test the prototypical system described in Chapter IV, two instances of it were created in real-life contexts. The purpose of instantiating it twice was not only to increase the number of students observed, but also to allow comparisons between students from two directly opposed environments in terms of attitude towards technology, and thus demonstrate the replicability of the study.

The two case studies have also created the opportunity to test the prototypical system in two different formats and for two different types of public. The system proved being an adaptable framework capable of assuming different characteristics that met each "client's" needs. This flexibility is very important when a new approach that aims at changing things is proposed, and has to start being put in practice in any available opportunity. As pointed out by Orma and Stevens (1995), in such situations one is "dependent on the institution's goodwill" and therefore should be as little demanding as possible.

The first premise for choosing schools was that they should belong to the Brazilian public system, which was highly recommended by CNPq, the Brazilian governmental agency sponsoring this study. Besides, as the result of a highly competitive admissions system, public universities have a good reputation in the country, both in terms of students and faculty. Due to practical reasons, two schools in São Paulo state were chosen: both the newest and the oldest in the state:

1. The recently created (1999) architecture course at UNICAMP, Sao Paulo State University in Campinas, with subjects offered by FEC, the School of Civil Engineering, and other departments;
2. FAU-USP, the School of Architecture and Planning at Sao Paulo State University in Sao Paulo city, which became independent from the Escola Politécnica's civil engineering course in 1968.

Although part of the public at the UNICAMP workshop was formed by graduate students and faculty, both courses were offered...
V. Case Studies

as undergraduate subjects. This does not mean that the proposed system could not be taught at the graduate level. In principle, it should be applicable to any given situation.

The architecture undergraduate degree at FAU-USP is a 5 year-long course, four of which are full-time (8:00am-5:00pm). Although FAU-USP has formed some of the most important modernist architects in the Brazilian scope, the school can be considered very conservative in what concerns the use of computers in the design activity. An evidence of this is the fact that, despite governmental recommendations in favor of universal CAD instruction, only about one fourth of the students have access to the two highly disputed elective CAD courses, usually in their last year at the school. Of course, most of the other students do look for complementary instruction elsewhere, learning to operate AutoCAD, the same program that is used in the two courses offered by the school, and a requisite for finding a good internship. Moreover, the two computer laboratories at FAU-USP are almost obsolete, lack the appropriate software, and the machines available are not enough for the more than 750 students.

The Architecture Course at FEC-UNICAMP is a 6 year long, part-time course. Classes are held in the evening, in order to give students the opportunity to work during the day. At the undergraduate level, three mandatory subjects are offered in the IT area, starting the very first semester, including the instruction on various types of graphic software, from Corel Draw to AutoCAD and 3DStudio. Besides, students are required to use CAD software for presentation in some studios, and other elective CAD subjects are offered by the civil engineering school at the graduate and undergraduate level. However, none of these subjects includes any computational design theory; computers are used exclusively for representation (modeling and drafting) and analysis (mainly in engineering). Contrary to what happens at FAU, FEC-UNICAMP's CAD laboratory is well equipped and maintained, with a variety of programs available.

Despite FAU's apparent disregard in what concerns CAD instruction, Prof. Maria Ruth Amaral Sampaio, the school's director, was very receptive to the idea of changing the way that CAD is taught, immediately putting me in contact with one of the CAD instructors, Prof. Marcelo Giacaglia, who agreed in ceding me six out of his 16 sessions in the immediately following semester. Not surprisingly, at UNICAMP, the reaction was even more favorable, and a special
Summer Course was especially created for my experiment, under the supervision of Prof. Doris Kowaltowsky, one of the founders of the architecture course.

Although the resources at FAU-USP in terms of equipment and supporting staff were inferior to UNICAMP's, the two schools had at least one thing in common: both used the same software, Architectural Desktop 2 (an architectural version of AutoCAD 2000), which made the preparation of the exercises much easier.

The experiment at UNICAMP happened between December 2001 and February 2002, before the experiment at FAU-USP, which happened between April and May 2002. As a result, after observations during the first instance, it was possible to incorporate some improvements to the second, in a spiral movement that is typical of action research.

V.b. Pilot work

After the prototypical system was developed, specific exercises for each theoretical concept were especially designed for undergraduate students, using the CAD software available at FAU-USP and UNICAMP. One of such exercises was pilot-tested at FAU-USP in November 2001, in a single 4-hour session, as a "guest lecture" in one of FAU's elective CAD subjects, AUT-535. The survey questionnaires completed for this pilot work also served as an additional source of information about architecture students' background and attitude towards CAD.

The "guest lecture" took place at the computer laboratory where classes are normally held. The lab was equipped with 20 Unisys CWP5757-Pentium 75 MHz microcomputers, with 48 MB RAM and 1 GB HDD and a 15" monitor. All machines were on Windows NT 4.0, running AutoCAD Architectural Desktop v.2. A high definition multimedia projector (800x600points) was used for presenting slides and demonstrating the exercises.

Except for one student, who was a fourth grade student, the other eleven students taking part in this pilot work were in their last year (officially the course takes five years, but most students finish it in six or more years). In what concerns students' familiarity with the CAD package used, AutoCAD, it suffices to say that they were at the end of a full-semester, 4-credit introductory AutoCAD elective
Each student received a handout with an introduction to the topic (in the case of this lecture, the use of symmetry in tile design), detailed descriptions of the exercises, as well as pre- and post-treatment questionnaires. All the material was printed on paper. The exercise directions were carefully explained, so that even the least skilled students would be able to follow them. Each step in the execution of an exercise had a headline explaining what had to be done and a body where all the interaction with the software was detailed, such as the name of menus and commands, what had to be typed at the prompt line, and so on. More advanced students should be able to work out the exercises just by reading the headlines.

The pilot work started with a two-hour lecture introducing the history of CAD, the concept of design rules (with Stiny and Mitchell's (1978) Palladian grammar), and an introduction to the study of symmetries. Next, a series of four exercises, well explained in a handout, were completed by students using the CAD tool they were familiar with, AutoCAD. The proposed exercises used viewports in an unorthodox way in order to create environments for developing symmetric designs. The objective was to displace the focus from the unit to the whole while designing with symmetries, and show students how CAD could help them during the creative process. Instead of using programming techniques, the exercises proved that it was possible to develop design environments with the use of simple techniques already known by the students. Finally, Symmetry Assistant, a more complex ready-made symmetric design environment, was presented and experimented by them. Students seemed to understand all the exercise directions, and to enjoy the paradigm shift introduced by the unorthodox use of viewports, but showed some amazement when the ready-made environment was introduced. For this reason, the only change made to the symmetry exercises after the pilot work was the introduction of a detailed explanation of how Symmetry Assistant had been created. In the second case study, at UNICAMP, the ready-made environment was not even used.

Despite students' apparent understanding of the design environments' concept, the compositions developed in class were not very elaborated, which was probably due to the fact that students were not very experienced with the medium, or did not have enough time to reflect on the exercises. In fact, the very same exercises produced
V. Case Studies

more interesting results in the experimental courses at UNICAMP and FAU-USP, where students had more time to develop them.

Students were asked to fill out a paper-based questionnaire before and after the introductory lecture and classroom exercises, so their original attitude towards CAD and possible change in it could be measured. The survey (see Appendix D: Assessment questionnaires) also helped establishing students' average profile and interests, which helped in the design of the experimental courses.

The questionnaires consisted of open and multiple-choice questions. The latter type was used whenever students were asked about something that was new to them, to suggest answers that they probably would not think about. For example, an open question such as "what is CAD used for?" in the pre-treatment questionnaire could have restricted the answers to "drafting" and "3D modeling", whereas a question like "how do you think CAD can help you being more creative?" would suggest something that students did not necessarily agree with. For this reason, students were asked to complete the phrase "CAD can help you..." with as many alternatives as they found applicable, from those listed below:

a. ... being more efficient.
b. ... being more creative.
c. ... finding novel shapes.

In this way, being efficient, creative or finding novel shapes was not taken as a necessary truth, but just suggested as a possibility, with which students could agree or not. Nevertheless, it was still possible to notice, by some of the answers and a few explicit comments, that some questions in the questionnaires were not quite clear to students. This fact was taken into account when the other questionnaires in this study were prepared, pointing to the need to consult specific sources on questionnaire design.

The results of the pre- and post treatment questionnaires showed that students were in general more concerned about using CAD for drafting than as a true design aide. All of them associated the word "CAD" to the particular computer program they were using (AutoCAD) instead of a more general idea of using computers for creating designs. After the lecture, most students stated a change in their opinion either about the meaning of the word CAD or the use of CAD in architecture (Chart 1). Contrary to what was expected,
most students reported being interested in learning programming techniques with the purpose of generating architectural forms (Chart 2). In what concerns the session evaluation, the majority of the students considered the introductory lecture clear, well illustrated and easy to follow (Chart 3).

In summary, the pilot work at FAU-USP, with actual architecture students, was an important step in the development of exercises that were interesting to the target public and at an appropriate level of difficulty, considering their background and original attitude towards CAD. The pilot work also helped in determining the time required for the exercises that were tested.
V. Case Studies

V.c. UNICAMP workshop

Resources

The workshop took place at FEC’s computer laboratory, equipped with 35 Microtec computers with PIII 866 processors and 128 MB of RAM, connected by a 100 Mbit/s Ethernet network. The CAD software used was AutoCAD, which is officially used and taught in both the Architecture and Civil Engineering courses, as well as in most Brazilian schools. Finally, the programming language used was Visual Basic for Applications, edited on AutoCAD’s own development environment. In what concerns image-editing software, Corel Draw was available, but it was not used, because civil engineering students were not familiar to that software.

Public

The group of students attending the UNICAMP workshop was very heterogeneous, as shown in Table 12. Nine students presenting final projects had architectural background, while the other six were civil engineering students, therefore not pertaining to the target

<table>
<thead>
<tr>
<th>Students’ profile</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Undergraduate architecture</td>
<td>5</td>
</tr>
<tr>
<td>Undergraduate civil engineering</td>
<td>6</td>
</tr>
<tr>
<td>Graduate students in the Edifications program (all of whom with architecture background)</td>
<td>2</td>
</tr>
<tr>
<td>FEC’s architecture course faculty members (3 holding a Ph.D. and the other 3 presently pursuing their doctorates)</td>
<td>0</td>
</tr>
<tr>
<td>M.Sc. in multimedia with architectural background</td>
<td>1</td>
</tr>
<tr>
<td>Professional architects</td>
<td>1</td>
</tr>
<tr>
<td>Total with architectural background</td>
<td>9</td>
</tr>
</tbody>
</table>

Table 12: UNICAMP students’ profile.
group originally defined for this research. Nevertheless, some of the projects developed by them were analyzed in this research. They provided evidence for interesting conclusions that, although not originally in the scope of this thesis, are certainly worth mentioning.

Pre-treatment questionnaires were applied before and after the introductory lecture in December, but at that occasion only 6 students were present. For this reason, the data gathered in that occasion could not be considered representative of the group and was not used as evidence in the study. Classroom observations provided information about students’ familiarity with AutoCAD, and their knowledge of programming languages. It is possible to say that undergraduate architecture students had a good knowledge of 2D and 3D AutoCAD, but had no programming experience at all. Civil engineering undergraduate students, on the contrary, were only experienced with AutoCAD 2D but had some knowledge of computer programming (Pascal). All the other students had some prior experience in AutoCAD, but not necessarily in 3D modeling. None of the students in the class, except for two faculty members, had prior experience in VBA programming.

Materials

The course materials consisted of handouts with directions for the proposed exercises, VBA programs and selected readings (papers or book chapters) directly related to each topic. All the course material was available for download from the class website. Handouts and selected readings were available in digital format, and all VBA programs were embedded in DWG files. Printouts of each session’s handouts and a questionnaire page were distributed to students at the beginning of every new topic.

The class website (Figure 70), which was also used to display the results of the workshop, is contained in the accompanying CDROM.

Format

The experimental course reported here was offered to FEC’s undergraduate and graduate students as a Summer workshop between January and February 2002. The entire course consisted of:
An introductory lecture about the history of CAD and notions of Computational Design theories.

- A two-week (ten sessions) workshop in a computer laboratory with daily classes lasting from 9:00AM to 1:00PM.
- A 10-day interval for the development of final projects.
- A final review.

The course calendar was determined by FEC's requirements for Summer courses: forty-eight hours of classes were required, with a minimum time span of 60 days between the first and the last session. Consequently, the introductory lecture had to happen in December, and it was missed by most students (only 7 students were present).

**Description**

Due to the unexpectedly large number of students and their diversity, the original plan for the UNICAMP workshop had to be slightly changed. With 25 students instead of the 7 or 8 pre-registered, and no teaching assistants, some exercises had to be eliminated and
others (especially the ones involving programming) had to be adapted to the large group. For example, when ready-made programs were presented and explained, some students asked for a step-by-step explanation about how to create very simple applications from scratch.

Each session started with a lecture in which concepts were presented and buildings from the architectural repertoire were analyzed in terms of each concept. For practical reasons, each text in the course's bibliography was attributed to a specific student, who presented a summary to the class. Text presentations happened in general after the lectures. The second part of the sessions was always used for classroom exercises (see Appendix B: Description of exercises). Because sessions happened every day, students were not given any formal homework, except for the preparation of text presentations (just one text for each student in the entire course of the workshop). Students were also asked to bring images with architectural examples of the concepts introduced, as a volunteer contribution to the class. Most students used part of their afternoons to finish classroom exercises in the same computer laboratory where classes were held, where I was usually available to help them.

For the first topic (Symmetry), students used AutoCAD’s commands in an innovative way. With the use of multiple, dynamically updated views of the design space, students were able to design wallpaper patterns in an environment that emphasized the whole instead of the units. During the study of topic 2 (Recursion), exercises were based on the use of Recursion Assistant, described in Chapter IV.

In topics 3 through 5 (Parameters, Generation and Search and Algorithmization), small programs that incorporated each theoretical concept were first presented and used by students, then thoroughly explained, and finally modified by them. Among such programs were Modernist Profile Explorer and Alternative Generator, also described in the previous Chapter.

VBA programming was introduced in these 3 sessions, starting with the explanation of simple, pedagogical, sample programs, and the concepts behind them. After those explanations, students developed very simple applications on their own, in classroom exercises where each step was carefully explained. This type of procedure was not without its risks. It often happened that a few students, in...
V. Case Studies

a classroom with 25, got stuck. Since there were no teaching assistants, I had to interrupt the process to help them reach the point where I had stopped.

In topic 6 (design rules) students used 2D and 3D Rules Assistant, which allowed them to generate compositions based on a single rule. Finally, in topic 7 (Emergent shapes) a standard command of the CAD package was again used to unveil novel shapes from a rule-based composition.

As a final project, students were given different options, to be developed in a ten-day period, individually or in small groups. The options included:

1. A small design project making explicit use of the computational design concepts introduced in the workshop.
2. A simplified computer application for automating a simple design procedure, optimizing a shape, generating different design alternatives or automating a drafting procedure.
3. A conceptual program for automating a more complex design procedure or a sequence of procedures.
4. An analysis of some work of art or architecture containing the computational design concepts introduced in the workshop.
5. Extending one of the programs presented in the workshop (such as adding more variables or options).

During the vacation week between the last session of the workshop and the final review, the only help students had from me was by e-mail, and the day before the final presentation I met some of them to solve their last questions. The final review took four hours and was attended by most students, even those not presenting final projects. Four reviewers were present, among which were two professors from UNICAMP and two professors from other architecture schools. The review started with a presentation of all the work developed by the students during the workshop. Next, each student or group of students were given approximately 10 minutes to present their final project and the reviewers had 5 minutes to comment on them.

Problems

With a daily schedule, time was too short for students to keep up to date with the reading list and exercises. To solve this problem, each
Results

The result of the 2-week workshop was an impressive number of beautiful compositions and thoughtful programs. The developed exercises can be separated in four categories:

1. Creative abstract compositions.
2. Small programs developed in tutorial exercises, in general not particularly creative, but a necessary step in the learning of the new programming technique. Some students were able to propose interesting variations of the programs provided, and even to develop new programs.
3. Images of architectural examples of symmetry, recursion and parametric shapes, which provided a clue to how well students were assimilating the concepts introduced in each session.
4. Final projects.

Among the final projects, there were only two analytical studies. All the other final projects consisted of VBA applications for different purposes, such as an automatic layout generator (based on a program provided in the workshop), a generator for land use alternatives, a stair-case 3D model generator, and programs for generating parametric versions of famous buildings. The fact that none of the students presented a design project can be interpreted in two different ways. It may be a sign of a skeptical attitude towards the use of computational design techniques in creative design (giving preference to its use for automation and optimization) or simply a consequence of students’ enthusiasm after learning of a new technique (VBA programming) and the resulting interest in applying it.
V. Case Studies

One of the best final projects was presented by an M.Sc. student with a background in architecture, who compared two designs by Frank Lloyd Wright (a church and a synagogue) and developed a program to model both using the same algorithm, but taking different parameters. This student was able to analyze the two buildings from a computational point of view, relate their construction processes, and generate a common, parameterized description of both. Another interesting project, presented by an engineering student, was the analysis of the work of an Italian-Brazilian painter, Alfredo Volpi, from a computational point of view. Different kinds of symmetries, recursive procedures and emergent shapes were pointed out in Volpi's paintings.

Assessment and communication

The UNICAMP workshop was assessed by the students and Prof. Kowaltowski, as a participant observer (see Appendix D: Assessment questionnaires). Before and after the introductory lecture, students filled out questionnaires about their background, course expectations, and attitude-change after the first lecture. These questionnaires were carefully planned and designed. They were mostly based on multiple-choice questions, to help determining students attitude towards CAD before and after the lecture. Unfortunately, most students who later took part in the rest of the workshop were absent that day, which made it impossible to generalize the findings. In the following sessions, after the introduction of each new concept, students were asked to fill out paper-based, anonymous forms with questions that assessed their understanding of the topics presented. These forms were never longer than a single page, consisting of three or four open or multiple-choice questions.

The very last questionnaire consisted of a short evaluation of the course, filled out by 12 students after the final review. Chart 4 summarizes the overall satisfaction with the workshop. Chart 5, based on the same questionnaire, indicates that there was a slight preference for exercises in comparison to the lectures and final project, but the three different activities were significantly cited as the best ones. It is also important to note that the great majority of students did not point to "worst" aspects of the course, but only to "best" ones.

Three months after the end of the workshop, students were asked to
answer a follow-up questionnaire by e-mail about external influences of the course during the semester, such as the use of the concepts introduced in other subjects or internships. They were also asked if they would be interested in giving continuation to the experience in a future workshop, to which all students replying the questionnaire answered yes.

Professor Kowaltowski, the participant observer, was interviewed by e-mail in the same period. Her major concern was with the need to give continuation to the project, by having other instructors, especially in studios, take advantage of the theoretical and technical contents assimilated by students.
How did you like the course? Did it meet your expectations? Do you have any critique or would you suggest any modifications?

I think the objectives were fully achieved. Judging by the work presented by students and especially taking into account the fact that it was a Summer course (when students usually don’t work as hard), I can say it was superior to my expectations. The important aspect is, no question about it, to demonstrate that CAD can be a creative partner in the design process (and on top of everything it is fun !!!). I’m sure some students will be able to take advantage of this experience and motivation in the development of future projects in undergraduate school, and even doing research in graduate school. My opinion is that the activities have been well chosen. Both conceptual and practical questions were addressed, and the results were very close to what is expected in architectural design. The reading of papers along with the creative and practical activities provided an interesting environment, motivating students to do research on their own.

(…) I’m concerned with the application of IT in the design process. I see many continuity problems between conventional CAD courses and studios, and a difficulty in the creative use [of CAD] in design by students.

Prof. Doris Kowaltowski

The results of the workshop could be evaluated in terms of the assimilation of the concepts, attitude change in what concerns the role of CAD in design, and influence in other subjects.

Besides the course-evaluation questionnaire, the good acceptance of the workshop could be also confirmed by students’ comments and compliments in class, in e-mail messages and during the final review discussion. Paula Roberta Baratela, an architecture undergraduate student, for example, confessed that she had always hated CAD and computers, and had registered for the course without being aware of its contents. Nevertheless, she declared having completely changed her mind about her attitude towards CAD during the workshop.

The final review jury made very positive comments on the work presented, emphasizing the appropriateness of the buildings cho-
V. Case Studies

Computationally parameterized in some of the programs, due to their computational nature. On the other hand, the extended bathroom layout program, presented by Marina Otaviano (Figure 76), was very criticized by three of the jury members, who happened to be studio instructors, but not by Prof. Coeli, a CAD and programming instructor. The studio instructors warned students about the risk of generalizing a design solution and applying it to any given case. The layout program was obviously meant to be applied to very specific situations, and the reasoning about the design process and building constraints the student went through were certainly a beneficial design exercise. Nevertheless, it was interesting to see the negative reaction of studio instructors to that kind of automation, which was not observed when other projects were presented, such as in the automation of stairs.

The only project that caused that kind of reaction was coincidentally the only one which presented a higher control over the design process. Even though design decisions in the program were based on simple if/then loops, the overall concept might have been seen as a challenge to the architect's control of the process. Surprisingly, many students defended the program, which can be interpreted as a sign of their understanding of the need for that kind of CAD application in architecture. This use sometimes includes automation, which always starts with a deep understanding and formalization of the design process.

In order to communicate the results of the workshop to other students in the school, the exercises were posted on a class web site, and posters were posted in the department to encourage people to visit it.

V.d. FAU-USP mini-course

Resources

The mini-course took place in one of the two computer laboratories at FAU-USP, equipped with 20 Unisys CWP5757 microcomputers with Pentium 166 MHz processors, 96 MB RAM, 1 GB hard drives, running Windows NT 4.0 in a 10Mb/s Ethernet network. The CAD software used was AutoCAD Architectural Desktop v.2 e v.2i, which is officially used and taught at the school, as in most Brazilian architecture courses. The programming language used was Visual Basic for Applications, edited on AutoCAD's own VBA.
Editor. No image-editing software was available.

Public

The group of students attending the experimental course at FAU consisted of 19 undergraduate architecture students, all of whom regularly registered in the fifth or higher grades. All the students had prior experience in AutoCAD, not necessarily in 3D modeling, and only three had some experience in programming, but none of them in VBA.

Materials

Instead of paper-based handouts, the course material was all made available in a specially designed web site, which included:

1. Course readings (consisting of papers or book chapters related to each session's topic, all of which had been OCR'ed and put in HTML format);
2. Detailed directions to all the exercises;
3. Lectures' slides;
4. Downloadable DWG files with specially prepared environments and VBA applications embedded in DWG files;
5. Instructions on how to operate the VBA applications provided.

As the course evolved, all the exercises presented by students were gradually posted in the web site for classroom review. The class web site is included in the accompanying CDROM.

Format

Due to administrative restrictions, it was not possible to create a stand-alone course with experimental purposes at FAU-USP. Therefore, the course was ministered as a series of guest lectures inserted in the second - and more advanced - of the two elective CAD courses in the school, thanks to the support of Prof. Marcelo Giacaglia. This second course, AUT-545-Advanced Graphic Resources in IT, is a 6-credit full semester course, which "aims at emphasizing the interdisciplinary character of advanced resources of computer graphics throughout all phases of design, from volumetric conception to urban insertion, including three-dimensional representation, verification of spatial results and walkthrough simulation." The course description statement also points out the use of
a "new communication language" that can help architects present their ideas.

The contents taught in the course before the six guest lectures consisted of the following topics: UCS's, axonometrics, solid primitives, extruded solids, Boolean operations, solid slicing, 3D-modeling, paper space, and an introduction to Auto Lisp.

The experimental course was taught in 6 weekly sessions of 4 hours each (8:00am-12:00pm), between April and May 2002. The topics of the sessions were:

1. Introduction and symmetry
2. Recursion and design rules
3. Parametric shapes
4. Shape generation
5. Algorithmization of design processes
6. Emergent shapes and wrap up

Each four-hour session was divided in three parts. The first part was devoted to a discussion about the previous week's exercises. Then, a new concept was presented, followed by architectural examples and some technical instruction. After a short interval, students started developing classroom exercises on their own, guided by online directions and assisted by me, with the help of Prof. Giacaglia (see Appendix B: Description of exercises).

As homework, students always had a small research question, a small architectural design exercise, a reading and summary, one or two questions to induce reflection on the week's topic and an assessment questionnaire. The purpose of the architectural design exercise, which had not been proposed at the UNICAMP workshop, was to give students an opportunity to apply each new concept in the design of a specific building type.

The number of sessions and their positioning within AUT-545's calendar took into account the following aspects:

1. Prof. Giacaglia's consent in ceding no more than half of his sessions;
2. The need to wait until students had some training in AutoCAD 3D;
3. The need to avoid the end of semester, when students are over-
whelmed with final design reviews and therefore not so open to learning new content.

The course happened, therefore, in six sessions in mid-semester, between April 21 and May 28, according to the calendar shown in Table 14. The time constraint made it necessary to condense two topics in some of the sessions.

**Description**

As in the UNICAMP workshop, in the first topic (symmetry), students used the CAD package's native commands in an innovative way. Besides the bilateral, wall-paper and circular symmetry exercises, FAU-USP students also developed a frieze composition. On the other hand, Symmetry Assistant was not used. In session 2, exercises started with the use of recursive viewport environments, which had not been used at UNICAMP. Next, Recursion Assistant was used.

In topic 3, parametric shapes, a new strategy was tried, instead of the Modernist Building Generator used at UNICAMP. After analyzing a building design by Niemeyer, students developed similar compositions with parametric variations of arches, using two different methods: AutoCAD blocks inserted at different X and Y scales, and a special VBA application, Parametric Arches. The use of the two methods allowed comparing the efficacy of the pro-

<table>
<thead>
<tr>
<th>Date</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>04/21</td>
<td>4hs</td>
</tr>
<tr>
<td>04/28</td>
<td>4hs</td>
</tr>
<tr>
<td>05/05</td>
<td>4hs</td>
</tr>
<tr>
<td>05/13</td>
<td>4hs</td>
</tr>
<tr>
<td>05/21</td>
<td>4hs</td>
</tr>
<tr>
<td>05/28</td>
<td>4hs</td>
</tr>
</tbody>
</table>

Credits: 1/3 of a 6 credits-subject = 2 credits
The strategy for teaching programming tried at FAU-USP was slightly different from that at UNICAMP. Instead of looking directly at ready-made programs and deriving concepts from them, it started with a formal explanation of programming concepts, and their comparison to similar ones in architecture. For example, the concept of classes was compared to the different building typologies, and the concept of an object (a class instantiation) was exemplified by "my" house - an instance of the class "houses". Although "my" house had some characteristics that were common to all houses, such as having bedrooms and a kitchen, it also had some specific characteristics (or properties) that were decided at instantiation time, such as the color of the walls and the building materials. Next, students developed very basic programs, following an on-line tutorial on their own, asking for individual help whenever necessary. For topics 4 and 5 (generative systems and algorithmization), simple, pedagogical programs were developed by students from scratch. Only after that, ready-made applications were used by students, as in the UNICAMP workshop.

Finally, in topic 6 (emergent shapes) standard AutoCAD commands were used to unveil novel shapes from rule-based compositions developed in Rules Assistant. While at UNICAMP this exercise was restricted to compositions in the plane, at FAU-USP students also developed 3D emergence exercises. For that purpose, instead of using area filling commands, they used Boolean operations, such as solid union, subtraction and intersection, to discover emergent volumes.

Problems

The main problem at FAU-USP concerned computer resources. Fortunately, the computer lab received Internet access, a fundamental need for the use of on-line course materials, a week before the start of the course, and the machines were upgraded in speed and memory especially for the experimentation. Nevertheless, it was impossible to obtain a proper multimedia projector from the school, and the first two lectures were taught without that resource. A portable projector was borrowed from a third party for the remaining sessions.

An unexpected fact in the FAU-USP mini-course was students’
complaints about the amount of homework proposed, and their apparent lack of time to develop exercises from one session to the next. Most students did not turn in their weekly exercises until the end of the course. Students rarely showed up during office hours, probably due to their tight schedule.

As it had been previewed, most students lacked enthusiasm and some even complained about the programming lessons. The programming exercises presented did not accomplish any more than the very minimum requirements, with a few exceptions.

Finally, because they lacked appropriate software and experience, students were not able to turn in their exercises in a raster format, as in the UNICAMP workshop. Therefore, converting all the CAD files into images and posting them on the web site took a considerable amount of time.

Results

Due to the tight schedule and also to FAU-USP students' lack of enthusiasm with programming, the results of this mini-course were less impressive than at UNICAMP, in terms of computer applications. Nevertheless, there were interesting results in terms of the application of concepts to architectural design, and a surprisingly high rate of change in attitude towards CAD could be measured.

The exercises developed can be separated in four categories:

1. Creative abstract compositions with symmetry, recursion, parametric shapes and emergence developed as classroom exercises;
2. Small VBA programs, resulting from classroom tutorials. Only two students were able to develop programs even slightly different from the examples suggested;
3. Images and sketches of architectural examples of the concepts studied, the result of research homework;
4. Architectural design exercises, in which students were asked to develop a study for a specific type of building using each concept learned, as a homework assignment.

Unfortunately, not all students presented the last type of exercise, but those who did seemed to have developed a better understanding of the related concepts and their application in architectural design. This type of applied exercise had not been proposed at UNICAMP.
because part of the public there was not familiar with architectural design.

Due to time constraints, it was impossible to ask students to develop a final project at FAU-USP.

**Assessment and communication**

The course at FAU-USP was assessed by the students and Prof. Giacaglia, as a participant observer (see Appendix D: Assessment questionnaires). In this case study, all the assessment material was based on on-line forms that students could easily fill in and submit from school or from home. These forms included text fields for open questions, and check boxes, radio buttons and drop lists for multiple choice questions, which made their completion more dynamic than paper-based questionnaires. Moreover, data were electronically sent in digital format, eliminating the need for interpreting hand-writing and transcribing answers. Because the turning in of questionnaires was considered mandatory and was computed as part of students' grades, a return-rate of almost 100% was achieved in most sessions, contrary to what happened at UNICAMP.

The pre-assessment questionnaire was filled out right after the course's first session by most students. Because the submission of the on-line questionnaires was mandatory and identification was required, it was possible to determine the average profile of students in what concerned previous experience with CAD software and programming, interests and expectations. As in the previous case study, the questionnaire included open and multiple-choice questions, yet, in this case, the suggestion of answers was avoided, except for questions asking for quantitative values, such as "in architecture, computer programming is... very important/ somewhat important/ not important."

After each session, students were asked to fill out on-line, non-anonymous forms with questions about the quality of the lectures and their level of interest in the topics presented. Here, again, only a couple of multiple-choice questions were asked to determine quantitative values of satisfaction and interest in the topics, and a file was included for volunteer comments. Questions that assessed the understanding of the concepts were asked as part of the homework, along with a small design exercise. The level of satisfaction
with the lectures was usually very high, but the interest decreased slightly with the introduction of programming concepts in sessions 4-Generation and 5-Algorithmization, and with the use of 3D commands in session 6-Emergence (not all students were experienced in 3D modeling).

At the end of the course, students were surveyed for the overall quality of the course instruction and materials, and the relevance of the topics introduced. This last questionnaire included an open question about students' intention to apply the concepts learned in other subjects or projects outside the school, to which most student answered yes, suggesting different uses, mainly in the initial phases of design.

Prof. Giacaglia attended all sessions, limiting his interventions to a few comments and questions during the lectures. On the other hand, when students were developing classroom exercises, he provided important support to them whenever I was busy with other students. In an interview at the end of the course, Prof. Giacaglia showed a concern with the limited use of IT by other instructors in the school, who, in general, have a negative opinion about the use of computers in architectural design. He acknowledged the importance of trying to make a better use of CAD in the creative phases of design with what is at our hand, while the "perfect CAD is still not available", which, according to him, depends on the development of new interfaces. Prof. Giacaglia has also explained how the contents of his present course is in part determined by "the job market pressure" for example, in what concerns the choosing of AutoCAD instead of other CAD packages.

The data collected at the FAU mini-course could be evaluated in

**Chart 6:** Students' evaluation of individual sessions and interest in each topic in the FAU-USP workshop (multiple-choice question), on a 0-10 scale.
terms of the assimilation of new architectural concepts, attitude change in what concerns the role of CAD in architectural design, and the application of the concepts in architectural design in some homework assignments. However, the time schedule did not allow assessing possible external influences of the mini-course in design projects for studios or internships.

Although not central to this study, the quality of the course methodology, materials, and teaching were also assessed in the final questionnaire (Chart 8). Students were in general very impressed not only with the possibility of accessing the course contents on-line, but mainly with the fact that their exercises were posted in the web site for classroom discussion every week. This type of immediate feedback seemed to encourage them to complete

Chart 7: Students' assessment after the FAU-USP mini-course (multiple-choice question): "Which were the best aspects of this course? (Choose 3)".
V. Case Studies

While I was counting on students’ extra-class time from one week to the next to compensate for the small number of sessions, according them one of the worst aspects of the mini-course was the amount of homework.

At the end of the course, all the exercises presented were posted in the "presentation" section of the class web site, and a poster was created to inform the URL to other students at FAU-USP.

Summary

The prototypical system described in Chapter IV was tested in two different architecture schools in Sao Paulo state. Due to particularities of context, students’ profiles, computer resources and administrative issues in each school, the system assumed different characteristics in each case, the first being taught as an independent Summer workshop and the second as an inset within an existing elective subject. A preliminary pilot work and the two case studies have been described in this Chapter. Problems encountered, results obtained and assessment techniques were also presented.
VI. Data Analysis

“The notion that the designer need not fully understand how the computer generates the answers (the 'black box' concept) is potentially very damaging. Without knowing what is taking place in the black box, the designer can never understand how much he can trust the answer; all he can do is accept the answer on faith. This significantly reduces his chances of utilizing his own judgement in the use of the computer-generated answer.” (Vladimir Bazjanac, 1975 p. 25)

VI.a. Data organization and presentation

Different kinds of data were collected during the field research. Grouped according to the methods by which they were collected, they consist of:

1. Classroom exercises (CAD drawings, VBA programs);
2. Homework exercises (CAD drawings, hand sketches, architectural images, answers to open questions and text summaries);
3. Final projects (only at UNICAMP; VBA applications and images);
4. Assessment questionnaires (on paper, on-line and sent by e-mail);
5. Classroom journals;
6. Interviews with observers (in person and by e-mail).

In what concerns their nature, the data collected were grouped under the following categories:

1. Charts and tables (with the results from assessment questionnaires);
2. Text (answers to homework questions, text summaries, classroom journals);
3. Computer programs (VBA applications from exercises and final projects);
4. Images of architectural examples (from research homework);
5. Hand sketches (from design and research homework);
6. CAD drawings (from abstract and concrete design exercises).

Nevertheless, in this chapter, the different types of data will not be presented according to their nature or method of collection, but regrouped according to the kind of information they provide. For example, different types of data provided clues about students' initial attitude towards CAD and programming, their understanding of design environments and programmed tools, their understanding of the concepts introduced and their application in architectural design, and finally the change in their attitude towards CAD.”
Table 15 summarizes the issues analyzed and corresponding sources at UNICAMP and FAU-USP, showing which data types were used in each case. Sometimes, the same data provided information on different issues.

### VI.b. Initial attitude towards CAD (assumption)

A few specific questions helped defining the type of attitude students had towards CAD prior to the workshop. At UNICAMP and in the pilot work conducted at FAU-USP, questionnaires were filled out just before the introductory lecture, assessing students’ attitude before any contact with the course contents. Unfortunately, though, few students were present at the introductory lecture at UNICAMP. At the FAU-USP experiment, the first questionnaire was filled out by students right after the introductory lecture, which, they admitted, influenced their answers, as stated by a student:
Table 16 (below): Pre-treatment questions concerning students' initial attitude towards the role of CAD in architecture.

Table 16 shows some of the pre-treatment questions asked, and a summary of the answers given at the FAU-USP course and the pilot work. The results from the pilot work at FAU have also been included here, increasing the sample of students assessed.

Do you think computers can help beyond representation in architecture?

"You are showing me that there are great possibilities of CAD helping in the creative process, because I used to see it simply as a technical tool. I had seen simulations of illumination or insulation, some technical aspects, but I had never thought about it as a help for the creative process of design. It's obvious that the technical aspects change the design process somehow, but the kind of help you talk about is different."

Rodrigo Azevedo, FAU-USP.

In your opinion, what is CAD?

Do you think CAD could help you be more efficient, be more creative or find out new shapes?

How can computers help you in design, beyond representation?

Do you think it is important to learn CAD at your school? Why?

Do you think CAD could be used in other subjects?
The first question asked to help determine students' attitude towards CAD was "what is CAD"? It was interesting to notice how some students mistook the word CAD for AutoCAD, the program used at school, giving characteristics of this particular product, instead of speaking about computers as tools for design in general. Another question asked was if CAD should be taught at architecture schools, to which almost 100% of the students answered yes. Nevertheless, the main reasons given for this were drafting, visualization and the job market in both cases. None of the students related the use of CAD in schools to shape exploration, to the formalization of the design process, or even to the automation of repetitive procedures in the design process.

One of the answers given by a student at FAU-USP is representative of the general view:

**Why is CAD instruction important in an architecture school?**

"I think it's very important because it's an almost indispensable pre-requisite for a student to get into the job market, since the great majority of students starts with CAD jobs. Besides, CAD allows multiple advantages, making easier the search for solutions for designs and allowing an easy representation of the alternatives, because it is not necessary to start making new representations of the same project from scratch (like in paper and ink drafting, for example)."

**Stella Tomiyoshi, FAU-USP.**
This attitude coincides with a pattern revealed by Duarte (2000) when surveying Brazilian architecture schools' faculty on the importance of including CAD in the curriculum. Most of them cited "job market demands" as the main reason (see section II.c) for it, which apparently coincides with students' expectations. Unfortunately, "job demands" almost always mean drafting - 3D modeling and visualization in the best case - but nothing related to a creative use or even automation.

The last question that provided additional clues about students' attitude towards CAD was related to their expectations about the experimental courses, which, in both cases (FAU-USP and UNICAMP), had been advertised as an innovative approach to CAD. The very fact of choosing to enroll speaks about their interest to start changing the way they used CAD, despite their fear of programming, as stated by a student from FAU-USP:

"One thing is to be conscious of the creative possibilities of the tool, another thing is knowing how to use it with that consciousness. I've always had a 'lack of patience' with creative methods with the use of computers, although I know they offer great possibilities as a help for the creative process of design. I've been trying to break up my attitude towards computers, which is one of the reasons that brought me to this course."

Chen Chih Cheng, FAU-USP.
In summary, it was possible to confirm the type of attitude towards the role of CAD in architecture that had been originally assumed: most students used to see CAD mainly as a tool for representation, but also as a job market demand. On the other hand, some students showed a surprising openness to change this attitude.

VI.c. Attitude towards programming

Different types of data helped determining students’ attitude towards programming. Results pointed to different types of attitude among students in each school taking part in the study. While FAU-USP students had more difficulty learning basic programming techniques, UNICAMP students - with both architectural and civil engineering background - seemed at ease with them and were able to develop more advanced applications.

The programming language used in both experimental courses was VBA - Visual Basic for Application - for AutoCAD (Figure 85). AutoCAD 2000 and higher versions include a VBA interactive development environment - VBAIDE. Although VBA is an object-oriented language, it is possible to use it as a scripting language, by simply sending AutoCAD standard commands to the drawing editor's prompt (Figure 86). Although this method is certainly not the most efficient and has its limitations, it is very easy to understand, providing a great opportunity to break up with any initial barriers, and providing the important “quick return on the time invested”, as suggested by Negroponte (1975a).

Another advantage of VBA, especially if compared to Auto LISP, the original programming language for AutoCAD, is the easiness to create graphic interfaces based on user forms and message boxes. VBAIDE contains a toolbox with control elements (such as buttons, slider bars and list boxes) that can be simply dragged and dropped on the forms. During the experimental courses I could observe how even the least experienced programmers became enthusiastic about the possibility of designing interfaces for architectural design.

The great majority of architecture students in this study had no prior programming experience, as shown in Chart 11 (students in the pilot work were also included, as an extra evidence). It must be noted that some of the architecture students who reported having prior experience in programming mentioned "HTML", which does...
VI. Data Analysis

Chart 11: Undergraduate architecture students in the study without any prior experience in programming.

Not have a typical programming language structure, including variable definition and code loops.

Despite being unexperienced in programming, most students in the FAU-USP pilot work showed some interest in learning a programming language with the objective of generating shapes or automating drafting (Chart 12 and Chart 13). Similarly, in a pre-treatment questionnaire at the FAU-USP mini-course, most students considered learning a programming language important to their profession and expressed their interest in it (Chart 14 and Chart 15). When asked about the difficulty to learn a programming language, most students said they were able to do it (Chart 16).

Not being experienced in a technique is obviously not evidence of bias against it. Rather, it is often simply due to a lack of opportunities to practice it. It was not by accident that all civil engineering students at UNICAMP did have experience in programming while architect students did not. The former had a mandatory Pascal subject in their curriculum, while programming classes are rare in most architecture courses. All novice programmers in the study reported difficulties and frustrations while developing classroom exercises, which is normal and was expected. At FAU-USP, this became obvious in post-treatment questionnaires and e-mails sent by students during the mini-course.

The comments below clearly expose the frustrations mentioned before. Nevertheless, most of them show also an appreciation of the contents of the lectures and an understanding of the need to learn programming. The last two comments are examples of some type of aversion to computer programming.
VI. Data Analysis

**Chart 14:** FAU-USP pre-treatment question (multiple-choice): "What is your opinion about programming in the architect's profession?"

- Harmful: 5%
- Very important: 32%
- Somehow important: 63%
- Not interested: 5%

**Chart 15:** FAU-USP pre-treatment question (multiple-choice): "Are you interested in programming?"

- Interested: 28%
- Too hard for me: 5%
- Hard: 21%
- Easy: 0%
- I can learn: 74%

**Chart 16:** FAU-USP pre-treatment question (multiple-choice): "How hard do you think it is to learn a programming language?"

- Hard for me: 5%
- Easy: 0%
- I can learn: 74%

Comments sent by e-mail or assessment forms during the FAU-USP mini-course:

"With the notions of programming introduced in this class [session 3], exercises became very hard to solve, at least for me; but the lecture was still interesting."

**Min Kyung Lee, FAU-USP**

"The lectures as a whole have been extremely interesting and you are extremely competent, otherwise the absence and drop-out rate would have increased, because the VBA programming field is extremely arid and I believe few students will develop any work on it after your course."

**Marcos Donizete, FAU-USP**

"[The worst part of the course were] those awful VBA codes."

**Marcos Donizete, FAU-USP**

"Your lectures are very clear, I can understand the general idea of your explanations, but when I try to solve the exercises I often get stuck in small procedural details of the programming lines and the program does not work."

**Chen Chih Chen, FAU-USP**

"I'm conscious that your classes pass a content of great importance. I can visualize the general idea and the possibilities in the creative process, but when I try to solve the exercises I always end up stuck in a silly detail because I don't know the machine [logic] so well."

**Chen Chih Chen, FAU-USP**

"I think it is hard to do the exercises. Maybe if there was more time for the classes and if we could learn more steps in the process it would become easier. As time is short and only an introduction to each topic is shown, those who have no previous programming experience, like me, feel a little lost."

**Daniela Costa, FAU-USP**

"The design process can certainly gain efficiency with the use of VBA programming - as long as I am not the one in charge of programming!!!"

**Renata Figueiredo, FAU-USP**
VI. Data Analysis

Chart 17: FAU-USP mini-course post-treatment assessment (open question) “Which were the worst aspects in this course?”

Chart 17 shows that almost half of the students considered "programming" one of the worst aspects of the course in a post-treatment questionnaire at FAU-USP. On the other hand, a closer look at the answers reveals that one probable reason for students’ disliking the programming parts of the course were their own anxieties and fears to learn a new technique. Even Fabio, a student who had reported having some prior programming experience, complained about it, and Ana Carolina, a student who had no problems with programming, mentioned the difficulties experienced by her colleagues.

As explained in Chapter V. Case Studies, different methodologies were used to teach VBA programming in the two experimental courses. Due to the differences in the public, and the course formats, it is impossible to infer conclusions about the efficacy of the different methods. On the other hand, it was very surprising that all architecture students presented VBA applications as final projects at UNICAMP, when that was not a requirement (other options included developing an architectural design with the techniques learned and developing a conceptual program). For that assignment, most students worked in small groups and received some help by e-mail during the week before the final review. As a result, even the least experienced programmers were able to develop thoughtful programs. The opportunity to develop a final project probably worked as a second chance, after the classroom exercises, to put the programming learned into practice. Their use of programming was more related to the parameterization of shapes than to the automation of design processes or the automatic generation of alternatives, but it was still a good beginning.
VI. Data Analysis

Which were the worst aspects of the course?

"I think the programming exercises were too hard because I had no idea of what I was doing."
Laila Milek, FAU-USP.

"There was too much programming; my previous programming experience turned out to be obsolete and I could not even meet the minimum requirements of each [programming] exercise..working with programming, even in a 'light' way, is not respecting the different technical background of students."
Fabio Bellini, FAU-USP.

"I did not see bad aspects, maybe just the difficulty other students had in learning programming. I did not have any difficulties because I had some notions of web programming, which...has a similar reasoning."
Ana Carolina Salomão, FAU-USP.

"I think not knowing AutoCAD programming has jeopardized my learning somehow. I had difficulties developing some ..[exercises]."
Daniela Costa, FAU-USP.

It is possible to conclude that there was no real bias against computer programming among architecture students; they simply were not familiar with it, which resulted in anxieties and fears that are common to people learning any new technique, but the students were in general well aware of its importance, contrary to my original assumptions.

VI.d. Understanding design environments and tools

Another important observation that could be made in the field research is concerned with students' understanding of the environments used for symmetric, rule-based, parametric and recursive design. This understanding, on its turn, is directly related to the acknowledgment that CAD software can be used in the creative phases of design.

Not all participating students seemed to fully understand the help
provided by the design environments created with viewports. Besides, when using Recursion Assistant and Rules Assistant, most students limited their exploration to the most basic spatial relations. One possible explanation is that they were inexperienced in design. This was true of the civil engineering students at UNICAMP, but not at FAU-USP, where all students were architecture seniors. Another possible explanation was that students simply did not have enough time to reflect on and properly explore the design environments. Yet, answers to assessment questionnaires and homework questions revealed that most students were actually able to understand the concepts behind them.

**Viewport environments for symmetric design**

To assess students' achievement in the use of symmetric design environments, certain characteristics in the compositions developed were considered, in this study, as a synonym of their "correct" use, and therefore of their understanding:

1. Continuity between neighboring fields;
2. The use of solid areas instead of fine lines (which was interpreted as a sign of seeing CAD as an overly precise tool);
3. The use of asymmetric motifs (since the resulting symmetries should be a consequence of the environments themselves);
4. The emergence of new, overall patterns.

Table 17 to Table 20 show some of the presented exercises grouped according to the criteria above in both case studies (except for those that were carried out only in one of the courses, such as the frieze exercise). Although the expected criteria were not always met, the important point is that students had the opportunity to experiment with the new use of CAD. The fact that an exercise presents symmetric basic units or discrete forms does not necessarily imply that it is incorrect or not interesting as a composition. In the first case, it just means that it could have been achieved more easily with the use of extra viewports. In the second case, the composition probably does not present shape continuity and emergence.

In the wallpaper symmetry exercise, a FAU-USP student from the mini-course presented an interesting sequence she went through when developing her composition (Figure 88). The images show how, departing from an orthogonal scheme - a direct consequence
VI. Data Analysis

Table 17: Bilateral symmetry exercises.

<table>
<thead>
<tr>
<th>Continuous shapes and emergent patterns</th>
<th>Symmetric (or almost) basic units</th>
<th>Discrete (or almost) shapes</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Continuous shapes" /></td>
<td><img src="image2.png" alt="Symmetric units" /></td>
<td><img src="image3.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image4.png" alt="Continuous shapes" /></td>
<td><img src="image5.png" alt="Symmetric units" /></td>
<td><img src="image6.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image7.png" alt="Continuous shapes" /></td>
<td><img src="image8.png" alt="Symmetric units" /></td>
<td><img src="image9.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image10.png" alt="Continuous shapes" /></td>
<td><img src="image11.png" alt="Symmetric units" /></td>
<td><img src="image12.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image13.png" alt="Continuous shapes" /></td>
<td><img src="image14.png" alt="Symmetric units" /></td>
<td><img src="image15.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image16.png" alt="Continuous shapes" /></td>
<td><img src="image17.png" alt="Symmetric units" /></td>
<td><img src="image18.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image19.png" alt="Continuous shapes" /></td>
<td><img src="image20.png" alt="Symmetric units" /></td>
<td><img src="image21.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image22.png" alt="Continuous shapes" /></td>
<td><img src="image23.png" alt="Symmetric units" /></td>
<td><img src="image24.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image25.png" alt="Continuous shapes" /></td>
<td><img src="image26.png" alt="Symmetric units" /></td>
<td><img src="image27.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image28.png" alt="Continuous shapes" /></td>
<td><img src="image29.png" alt="Symmetric units" /></td>
<td><img src="image30.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image31.png" alt="Continuous shapes" /></td>
<td><img src="image32.png" alt="Symmetric units" /></td>
<td><img src="image33.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image34.png" alt="Continuous shapes" /></td>
<td><img src="image35.png" alt="Symmetric units" /></td>
<td><img src="image36.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image37.png" alt="Continuous shapes" /></td>
<td><img src="image38.png" alt="Symmetric units" /></td>
<td><img src="image39.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image40.png" alt="Continuous shapes" /></td>
<td><img src="image41.png" alt="Symmetric units" /></td>
<td><img src="image42.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image43.png" alt="Continuous shapes" /></td>
<td><img src="image44.png" alt="Symmetric units" /></td>
<td><img src="image45.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image46.png" alt="Continuous shapes" /></td>
<td><img src="image47.png" alt="Symmetric units" /></td>
<td><img src="image48.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image49.png" alt="Continuous shapes" /></td>
<td><img src="image50.png" alt="Symmetric units" /></td>
<td><img src="image51.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image52.png" alt="Continuous shapes" /></td>
<td><img src="image53.png" alt="Symmetric units" /></td>
<td><img src="image54.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image55.png" alt="Continuous shapes" /></td>
<td><img src="image56.png" alt="Symmetric units" /></td>
<td><img src="image57.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image58.png" alt="Continuous shapes" /></td>
<td><img src="image59.png" alt="Symmetric units" /></td>
<td><img src="image60.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image61.png" alt="Continuous shapes" /></td>
<td><img src="image62.png" alt="Symmetric units" /></td>
<td><img src="image63.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image64.png" alt="Continuous shapes" /></td>
<td><img src="image65.png" alt="Symmetric units" /></td>
<td><img src="image66.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image67.png" alt="Continuous shapes" /></td>
<td><img src="image68.png" alt="Symmetric units" /></td>
<td><img src="image69.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image70.png" alt="Continuous shapes" /></td>
<td><img src="image71.png" alt="Symmetric units" /></td>
<td><img src="image72.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image73.png" alt="Continuous shapes" /></td>
<td><img src="image74.png" alt="Symmetric units" /></td>
<td><img src="image75.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image76.png" alt="Continuous shapes" /></td>
<td><img src="image77.png" alt="Symmetric units" /></td>
<td><img src="image78.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image79.png" alt="Continuous shapes" /></td>
<td><img src="image80.png" alt="Symmetric units" /></td>
<td><img src="image81.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image82.png" alt="Continuous shapes" /></td>
<td><img src="image83.png" alt="Symmetric units" /></td>
<td><img src="image84.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image85.png" alt="Continuous shapes" /></td>
<td><img src="image86.png" alt="Symmetric units" /></td>
<td><img src="image87.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image88.png" alt="Continuous shapes" /></td>
<td><img src="image89.png" alt="Symmetric units" /></td>
<td><img src="image90.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image91.png" alt="Continuous shapes" /></td>
<td><img src="image92.png" alt="Symmetric units" /></td>
<td><img src="image93.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image94.png" alt="Continuous shapes" /></td>
<td><img src="image95.png" alt="Symmetric units" /></td>
<td><img src="image96.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image97.png" alt="Continuous shapes" /></td>
<td><img src="image98.png" alt="Symmetric units" /></td>
<td><img src="image99.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image100.png" alt="Continuous shapes" /></td>
<td><img src="image101.png" alt="Symmetric units" /></td>
<td><img src="image102.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image103.png" alt="Continuous shapes" /></td>
<td><img src="image104.png" alt="Symmetric units" /></td>
<td><img src="image105.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image106.png" alt="Continuous shapes" /></td>
<td><img src="image107.png" alt="Symmetric units" /></td>
<td><img src="image108.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image109.png" alt="Continuous shapes" /></td>
<td><img src="image110.png" alt="Symmetric units" /></td>
<td><img src="image111.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image112.png" alt="Continuous shapes" /></td>
<td><img src="image113.png" alt="Symmetric units" /></td>
<td><img src="image114.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image115.png" alt="Continuous shapes" /></td>
<td><img src="image116.png" alt="Symmetric units" /></td>
<td><img src="image117.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image118.png" alt="Continuous shapes" /></td>
<td><img src="image119.png" alt="Symmetric units" /></td>
<td><img src="image120.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image121.png" alt="Continuous shapes" /></td>
<td><img src="image122.png" alt="Symmetric units" /></td>
<td><img src="image123.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image124.png" alt="Continuous shapes" /></td>
<td><img src="image125.png" alt="Symmetric units" /></td>
<td><img src="image126.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image127.png" alt="Continuous shapes" /></td>
<td><img src="image128.png" alt="Symmetric units" /></td>
<td><img src="image129.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image1210.png" alt="Continuous shapes" /></td>
<td><img src="image1211.png" alt="Symmetric units" /></td>
<td><img src="image1212.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image1213.png" alt="Continuous shapes" /></td>
<td><img src="image1214.png" alt="Symmetric units" /></td>
<td><img src="image1215.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image1216.png" alt="Continuous shapes" /></td>
<td><img src="image1217.png" alt="Symmetric units" /></td>
<td><img src="image1218.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image1219.png" alt="Continuous shapes" /></td>
<td><img src="image1220.png" alt="Symmetric units" /></td>
<td><img src="image1221.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image1222.png" alt="Continuous shapes" /></td>
<td><img src="image1223.png" alt="Symmetric units" /></td>
<td><img src="image1224.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image1225.png" alt="Continuous shapes" /></td>
<td><img src="image1226.png" alt="Symmetric units" /></td>
<td><img src="image1227.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image1228.png" alt="Continuous shapes" /></td>
<td><img src="image1229.png" alt="Symmetric units" /></td>
<td><img src="image1230.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image1231.png" alt="Continuous shapes" /></td>
<td><img src="image1232.png" alt="Symmetric units" /></td>
<td><img src="image1233.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image1234.png" alt="Continuous shapes" /></td>
<td><img src="image1235.png" alt="Symmetric units" /></td>
<td><img src="image1236.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image1237.png" alt="Continuous shapes" /></td>
<td><img src="image1238.png" alt="Symmetric units" /></td>
<td><img src="image1239.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image12310.png" alt="Continuous shapes" /></td>
<td><img src="image12311.png" alt="Symmetric units" /></td>
<td><img src="image12312.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image12313.png" alt="Continuous shapes" /></td>
<td><img src="image12314.png" alt="Symmetric units" /></td>
<td><img src="image12315.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image12316.png" alt="Continuous shapes" /></td>
<td><img src="image12317.png" alt="Symmetric units" /></td>
<td><img src="image12318.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image12319.png" alt="Continuous shapes" /></td>
<td><img src="image12320.png" alt="Symmetric units" /></td>
<td><img src="image12321.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image12322.png" alt="Continuous shapes" /></td>
<td><img src="image12323.png" alt="Symmetric units" /></td>
<td><img src="image12324.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image12325.png" alt="Continuous shapes" /></td>
<td><img src="image12326.png" alt="Symmetric units" /></td>
<td><img src="image12327.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image12328.png" alt="Continuous shapes" /></td>
<td><img src="image12329.png" alt="Symmetric units" /></td>
<td><img src="image12330.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image12331.png" alt="Continuous shapes" /></td>
<td><img src="image12332.png" alt="Symmetric units" /></td>
<td><img src="image12333.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image12334.png" alt="Continuous shapes" /></td>
<td><img src="image12335.png" alt="Symmetric units" /></td>
<td><img src="image12336.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image12337.png" alt="Continuous shapes" /></td>
<td><img src="image12338.png" alt="Symmetric units" /></td>
<td><img src="image12339.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image123310.png" alt="Continuous shapes" /></td>
<td><img src="image123311.png" alt="Symmetric units" /></td>
<td><img src="image123312.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image123313.png" alt="Continuous shapes" /></td>
<td><img src="image123314.png" alt="Symmetric units" /></td>
<td><img src="image123315.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image123316.png" alt="Continuous shapes" /></td>
<td><img src="image123317.png" alt="Symmetric units" /></td>
<td><img src="image123318.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image123319.png" alt="Continuous shapes" /></td>
<td><img src="image123320.png" alt="Symmetric units" /></td>
<td><img src="image123321.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image123322.png" alt="Continuous shapes" /></td>
<td><img src="image123323.png" alt="Symmetric units" /></td>
<td><img src="image123324.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image123325.png" alt="Continuous shapes" /></td>
<td><img src="image123326.png" alt="Symmetric units" /></td>
<td><img src="image123327.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image123328.png" alt="Continuous shapes" /></td>
<td><img src="image123329.png" alt="Symmetric units" /></td>
<td><img src="image123330.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image123331.png" alt="Continuous shapes" /></td>
<td><img src="image123332.png" alt="Symmetric units" /></td>
<td><img src="image123333.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image123334.png" alt="Continuous shapes" /></td>
<td><img src="image123335.png" alt="Symmetric units" /></td>
<td><img src="image123336.png" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image123337.png" alt="Continuous shapes" /></td>
<td><img src="image123338.png" alt="Symmetric units" /></td>
<td><img src="image123339.png" alt="Discrete shapes" /></td>
</tr>
</tbody>
</table>

UNICAMP workshop

FAU-USP mini course
### Table 18: Frieze symmetry exercises.

<table>
<thead>
<tr>
<th>Continuous shapes and emergent patterns</th>
<th>Symmetric (or almost) basic units</th>
<th>Discrete (or almost) shapes</th>
</tr>
</thead>
</table>

FAU-USP mini course

### Table 19: Circular symmetry exercises.

<table>
<thead>
<tr>
<th>Continuous shapes and emergent patterns</th>
<th>Symmetric (or almost) basic units</th>
<th>Discrete (or almost) shapes</th>
</tr>
</thead>
</table>

UNICAMP workshop

FAU-USP mini course
### Table 20: Wall-paper symmetry exercises

<table>
<thead>
<tr>
<th>Continuous shapes and emergent patterns</th>
<th>Symmetric (or almost) basic units</th>
<th>Discrete (or almost) shapes</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Pattern 1" /></td>
<td><img src="image2.png" alt="Pattern 2" /></td>
<td><img src="image3.png" alt="Pattern 3" /></td>
</tr>
<tr>
<td><img src="image4.png" alt="Pattern 4" /></td>
<td><img src="image5.png" alt="Pattern 5" /></td>
<td><img src="image6.png" alt="Pattern 6" /></td>
</tr>
<tr>
<td><img src="image7.png" alt="Pattern 7" /></td>
<td><img src="image8.png" alt="Pattern 8" /></td>
<td><img src="image9.png" alt="Pattern 9" /></td>
</tr>
</tbody>
</table>

**UNICAMP workshop**

<table>
<thead>
<tr>
<th><img src="image10.png" alt="Pattern 10" /></th>
<th><img src="image11.png" alt="Pattern 11" /></th>
<th><img src="image12.png" alt="Pattern 12" /></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image13.png" alt="Pattern 13" /></td>
<td><img src="image14.png" alt="Pattern 14" /></td>
<td><img src="image15.png" alt="Pattern 15" /></td>
</tr>
<tr>
<td><img src="image16.png" alt="Pattern 16" /></td>
<td><img src="image17.png" alt="Pattern 17" /></td>
<td><img src="image18.png" alt="Pattern 18" /></td>
</tr>
</tbody>
</table>

**UNICAMP workshop**

<table>
<thead>
<tr>
<th><img src="image19.png" alt="Pattern 19" /></th>
<th><img src="image20.png" alt="Pattern 20" /></th>
<th><img src="image21.png" alt="Pattern 21" /></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image22.png" alt="Pattern 22" /></td>
<td><img src="image23.png" alt="Pattern 23" /></td>
<td><img src="image24.png" alt="Pattern 24" /></td>
</tr>
<tr>
<td><img src="image25.png" alt="Pattern 25" /></td>
<td><img src="image26.png" alt="Pattern 26" /></td>
<td><img src="image27.png" alt="Pattern 27" /></td>
</tr>
</tbody>
</table>

**FAU-USP mini course**

<table>
<thead>
<tr>
<th><img src="image28.png" alt="Pattern 28" /></th>
<th><img src="image29.png" alt="Pattern 29" /></th>
<th><img src="image30.png" alt="Pattern 30" /></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image31.png" alt="Pattern 31" /></td>
<td><img src="image32.png" alt="Pattern 32" /></td>
<td><img src="image33.png" alt="Pattern 33" /></td>
</tr>
<tr>
<td><img src="image34.png" alt="Pattern 34" /></td>
<td><img src="image35.png" alt="Pattern 35" /></td>
<td><img src="image36.png" alt="Pattern 36" /></td>
</tr>
</tbody>
</table>
of the arrangement of tiles - she arrived at an angled composition with good continuity between neighboring tiles.

The use of Symmetry Assistant in the pilot work showed that some students had difficulty understanding how that environment functioned. For that reason, at UNICAMP it was explained in detail, and at FAU-USP it was not even used. Instead, students created their own circular symmetry environments in one of the topic's exercise. The overlap of viewports was not well understood by most students. Moreover, in circular symmetry compositions, many of them insisted in using symmetrical motifs, generating dihedral compositions, which could also have been obtained with the use of even more complex viewport arrangements, as shown in Figure 89.

The symmetry exercises presented show that not all students were able to take full advantage of the design environments when designing. One possible reason for this is that most students are not used to this type of quick design exercise. Most academic projects are carried out slowly throughout the semester, giving them time to reflect on and change their decisions as many times as needed. While that type of exercise is very important, I believe that quick, instant-result exercises are a good way to develop good design skills. On the other hand, those students who could understand the concept behind the symmetric design environments were certainly able to see new uses of CAD, well beyond simple representation.

In a homework question at the FAU-USP mini-course, students were asked specifically about the use of the viewport environments for designing with symmetries (see next page). Most of them, like Leandro and Daniela, acknowledged the efficacy of using the viewport environments. Fernando de Mello also showed a clear understanding of shape continuity. Stella Tomyioshi pointed out how this type of technique is more appropriate for the initial phases of design - which was exactly what I wanted to demonstrate. Renata Figueiredo called attention to the risk of getting stuck with the pre-established symmetric scheme, since they were established a priori and it was only possible to change the motifs after that. Laila Milek and Daniela Vaz seemed a little skeptical about the unorthodox use of viewports, which clearly shows an attachment to the rules of "good" CAD drafting.
"Discuss the advantages and disadvantages of using the viewports environment to develop a symmetrical design."

"The main advantage is to visualize a design in its totality while changes are made in just one viewport."
Leandro Robles, FAU-USP

"The viewports facilitate the development of symmetric designs because it is possible to immediately verify the results of the symmetric operations applied to the model, working in real-time."
Daniela Costa, FAU-USP

"The viewport environments can be useful with modules that allow a continuity between each other, as in the classroom exercises."
Fernando de Mello, FAU-USP

"With the use of viewports we can see the results quicker ...the overall visualization, as well as any update or change are also immediate. Nevertheless, I believe this is an advantage only in a specific phase of the design process - the preliminary study - after that, there is the need to a more specific detailing."
Stella Tomyioshi, FAU-USP

"A viewport environment in a symmetric design gives agility to the process, because once the basic element is created, it is already possible to visualize the final overall composition. The disadvantage is that the number of solutions could be restricted to the pre-established symmetry scheme. Any variation would be impossible in this scheme."
Renata Figueiredo, FAU-USP

"I'm not sure if I would know how to use a viewport environment to design, but I still think it's a valid way to create compositions."
Laila Milek, FAU-USP

"Designing with viewports is something new for me, I only used that as a presentation resource, but it seems to me that the new use could jeopardize the overall visualization of the model..."
Daniela Vaz, FAU-USP
Recursion Assistant and viewport environments for recursive design

At UNICAMP and FAU-USP, students developed two different types of exercises before experimenting with Recursion Assistant. In the first case, they used AutoCAD blocks recursively "nested", automatically updating their designs as the blocks were redefined with progressively more complex levels of detail (Figure 90). At FAU-USP, they used the viewport environments similar to those used for symmetric compositions.

The criteria established to judge achievement in the recursion exercises were:

1. Shape continuity and the avoidance of loose, discrete forms;
2. Use of non-obvious spatial relations;
3. Use of asymmetric motifs and avoidance of spatial relations that generated frieze, wall paper and circular symmetries already seen in the previous topic;
4. Emergence of new shapes or patterns, i.e., the arising of shapes in the final compositions not originally defined as basic units or motifs explicitly.

Table 21 and Table 22 show some of the presented exercises grouped according to the criteria above. While in the recursive viewport environments the spatial relation was pre-established and basic shapes could be changed in real-time, in the Recursion Assistant exercises more attention could be given to spatial relations. After an interesting spatial relation was found, initial shapes could be easily changed. With that in mind, some students developed different designs based on the same composition rules (Figure 91).

UNICAMP and FAU-USP students wrote about the use of Recursion Assistant (respectively in anonymous questionnaires and in homework questions). In the answers quoted in page 127, it is possible to notice a great enthusiasm among UNICAMP students with the finding of a creative application of computers in architectural design that was completely new to them. Student D acknowledged the fact that he or she would never have found those shapes by traditional methods. In other words, the computer can lead to a complexity that is hard to achieve with by-hand methods. On the other hand, as pointed out by student E, it may be dangerous to use
VI. Data Analysis

Table 21: Recursion exercises with viewport environments, only at FAU-USP.

<table>
<thead>
<tr>
<th>Continuous shapes and emergent patterns</th>
<th>Symmetric (or almost) basic units</th>
<th>Discrete (or almost) shapes</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Image" /></td>
<td><img src="image2" alt="Image" /></td>
<td><img src="image3" alt="Image" /></td>
</tr>
<tr>
<td><img src="image4" alt="Image" /></td>
<td><img src="image5" alt="Image" /></td>
<td><img src="image6" alt="Image" /></td>
</tr>
<tr>
<td><img src="image7" alt="Image" /></td>
<td><img src="image8" alt="Image" /></td>
<td><img src="image9" alt="Image" /></td>
</tr>
<tr>
<td><img src="image10" alt="Image" /></td>
<td><img src="image11" alt="Image" /></td>
<td><img src="image12" alt="Image" /></td>
</tr>
</tbody>
</table>
Table 22: Exercises developed with Recursion Assistant.

<table>
<thead>
<tr>
<th>Continuous shapes and emergent patterns</th>
<th>Discrete shapes or symmetric compositions</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Continuous shapes" /></td>
<td><img src="image2" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image3" alt="Continuous shapes" /></td>
<td><img src="image4" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image5" alt="Continuous shapes" /></td>
<td><img src="image6" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image7" alt="Continuous shapes" /></td>
<td><img src="image8" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image9" alt="Continuous shapes" /></td>
<td><img src="image10" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image11" alt="Continuous shapes" /></td>
<td><img src="image12" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image13" alt="Continuous shapes" /></td>
<td><img src="image14" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image15" alt="Continuous shapes" /></td>
<td><img src="image16" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image17" alt="Continuous shapes" /></td>
<td><img src="image18" alt="Discrete shapes" /></td>
</tr>
<tr>
<td><img src="image19" alt="Continuous shapes" /></td>
<td><img src="image20" alt="Discrete shapes" /></td>
</tr>
</tbody>
</table>

UNICAMP workshop
### VI. Data Analysis

**Table 22: Exercises developed with Recursion Assistant (cont.)**

<table>
<thead>
<tr>
<th>Continuous shapes and emergent patterns</th>
<th>Discrete shapes or symmetric compositions</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Diagram" /></td>
<td><img src="image2" alt="Diagram" /></td>
</tr>
<tr>
<td><img src="image3" alt="Diagram" /></td>
<td><img src="image4" alt="Diagram" /></td>
</tr>
<tr>
<td><img src="image5" alt="Diagram" /></td>
<td><img src="image6" alt="Diagram" /></td>
</tr>
<tr>
<td><img src="image7" alt="Diagram" /></td>
<td><img src="image8" alt="Diagram" /></td>
</tr>
</tbody>
</table>

FAU-USP mini course
"Do you believe the techniques shown in this class [Recursion Assistant] could help you in the creative exploration of architectural forms?"

"The techniques shown in this class stimulate our creativity to the extent that even during coffee-break we were impelled to stay in front of the computer, testing new colors and spatial relations in order to obtain those surprising results, which would hardly be obtained in another way.”

Student A, UNICAMP.

"I believe it can help, or rather, stimulate creativity when we are designing."

Student B, UNICAMP.

"Yes, this technique allows the creative exploration of different combinations of shapes, leading to a final result that is novel and creative."

Student C, UNICAMP.

"Yes, offering possibilities that I wouldn’t have thought about without the help of the computer."

Student D, UNICAMP.

"I believe this technique stimulates creativity and helps dealing with shapes, but I also think [the resulting compositions] are a little complex to be used in architecture."

Student E, UNICAMP.

overly complex compositions in architectural design (although I believe he or she meant plans or masses, while the design of structures could probably take better advantage of this technique).

Although FAU-USP students seemed less enthusiastic about the technique, they did acknowledge the advantages of computer implementations in comparison to by-hand techniques. Like student A at UNICAMP, Leandro Robles pointed out the possibility of finding certain complex shapes exclusively with the help of the computer, but also the risks of disregarding important issues related to building-performance. Stella Tomyioshi and Daniela Costa pointed out the applicability of this technique to the creative phases of the design process.
VI. Data Analysis

"Discuss the advantages and disadvantages of using Recursion Assistant to develop architectural designs."

"The system presents numerous advantages for the creative process, with spatial possibilities that would hardly be created with pencil and paper. Yet, we should be very careful about lighting and ventilation."

Leandro Robles, FAU-USP.

"The advantages of the use of applications like this are the immediate visualization of the proposal and the easiness to edit it. Nevertheless, this is only valid in a preliminary phase of the design process, when studying the mass plan or volumetric scheme, which reinforces the [course's] original suggestion to use CAD as a generative tool."

Stella Tomiyoshi, FAU-USP.

"Recursion Assistant allows visualizing almost instantly an idea and quickly analyzing and changing it. It's also possible to create novel, unexpected shapes, which can be good solutions, which means CAD is being used in the architectural creative process."

Daniela Costa, FAU-USP.

Rule-based design environments

Another type of recursive design environment very similar to Recursion Assistant was used at the UNICAMP workshop under the topic of "design rules." The only technical difference between Rules Assistant and Recursion Assistant is that the former allows the insertion of a single instance of Shape2, while the latter allows multiple insertions of Shape2 in the spatial relation window. Besides, Rules Assistant allows unequal scaling of Shape2, and was available in both 2D and 3D versions (for students more and less advanced in 3D modeling), while Recursion Assistant was only available for 2D compositions.

The original idea was to demonstrate the definition of shape rules and their application in design. Nevertheless, the computer application used, Rules Assistant, was very limited, only allowing students to generate "set" rule-based designs, where a single rule defined by users is recursively applied without their interference. For this reason, there was some confusion between this design environment and the previous one. Some of the designs developed with Rules Assistant are presented on Table 23, according to the same criteria used to analyze designs in the recursion topic.
Table 23: Rule-based composition exercises developed by UNICAMP students.

<table>
<thead>
<tr>
<th>Continuous shapes and emergent patterns</th>
<th>Discrete shapes or symmetric compositions</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Diagram 1" /></td>
<td><img src="image2.png" alt="Diagram 2" /></td>
</tr>
<tr>
<td><img src="image3.png" alt="Diagram 3" /></td>
<td><img src="image4.png" alt="Diagram 4" /></td>
</tr>
<tr>
<td><img src="image5.png" alt="Diagram 5" /></td>
<td><img src="image6.png" alt="Diagram 6" /></td>
</tr>
</tbody>
</table>

![Diagram 7](image7.png)
The most interesting aspect of the rule-based design exercise was to see how some students experimented with applying different rules to the same initial shapes and the same rule to different initial shapes, obtaining completely different results. Nevertheless, because Rules Assistant was limited to the use of a single "set" rule, there was not enough difference between this and the recursive design exercises, which resulted in some confusion. For this reason, students’ were not asked specifically about this program. In future experiments, more advanced programs, allowing the definition and free application of different rules, should be used or developed by students whenever possible.
Parametric shapes

The concept of parametric variation of shapes was introduced both at UNICAMP and FAU-USP with exercises directly related to architectural examples, shown in Figure 92. At UNICAMP, students started by inserting AutoCAD blocks at different X ans Y scales to create compositions with arches, like in the Mondadori building. Next, they were asked to generate parametric variations of Reidy's Museum of Modern Art using a ready-made VBA application, Modernist Profile Explorer. At FAU-USP, instead, students tried out the two different methods for generating variations of the same shape: first inserting an arch at different X and Y scales, and then using a VBA application for the same purpose, Parametric Arches Explorer. In the blocks exercise, instead of the simple arch used at UNICAMP, students inserted a more complex shape (compare Figures 93 and 94), which emphasized the limitations of the technique. The VBA application (Figure 95), instead, allowed a better control of the arches' many attributes, such as impost height and frame width.

The sequence followed at FAU-USP resulted in a better understanding of the help provided by computers in parametric design, because:

1. The ready-made application used (Parametric Arches Explorer) was simpler in terms of programming - and could be better understood by students;
2. It was possible to compare a method students were already familiar with (AutoCAD blocks) and a new one, based on a programmed tool, for the same design problem.

Some of the presented exercises are shown in Table 24. In both case studies, a few students were even able to develop more advanced applications of parametric shapes using VBA programming, which are shown in Table 24.

In summary, students' response to the proposed exercises did not always display a sign of a good understanding of the design environments, when evaluated by the criteria described above. Yet, students' answers to questions about the usefulness of such environments revealed that most of them could understand their purpose and were able to see their applicability in the early phases of the design process.
 VI.e. Understanding the concepts

Specific questions were asked as homework (at FAU-USP) or in assessment questionnaires (at UNICAMP) in order to gather evidence about students’ understanding of the concepts introduced. The questions asked after the introduction of each concept, and some of the answers given are presented below. Another type of
VI.Data Analysis

homework, searching for architectural examples of the concepts studied, provided extra clues on this issue. Some examples of the images submitted by students as examples of symmetry, recursion and parameterization are also shown at the end of this section.

VI.e.1. Symmetry

In general, students in this study reported being surprised not only with the new possibilities opened by the use of CAD to explore symmetric designs, but also with the new types of symmetries presented to them, such as the different kinds of circular symmetries. At FAU-USP students answered a specific question about the use of symmetry in architectural design.

It was possible to conclude that most students at FAU-USP associated symmetry to monotony. For this reason, they suggested using it in a less strict way, with "rhythm" and "movement", and pointed out the importance of knowing the different types of symmetry. Some students also mentioned that the repetition in symmetric designs can result in building economy.

"What are the advantages and disadvantages of using symmetry in architectural design?"

"Symmetry is an interesting aesthetical resource, and doesn't need to be used in a strict way. Breaking up symmetry can enrich the design."

Rodrigo Azevedo, FAU-USP.

"The use of symmetries ties the building parti to a certain type of organization of the elements in a design. The aesthetical result may be interesting, as long as it has some rhythm, some movement, avoiding monotony, which often happens in symmetric designs."

Stella Tomiyoshi, FAU-USP.

"If an architect is not familiar with the different possibilities, the use of symmetry can limit creativity."

Marcelo Nakazaki, FAU-USP.

"The use of symmetry in a project can make construction more rational, simple and quick. It may have also aesthetical advantages, since we can get very interesting designs by using symmetrically displayed basic units."

Fernando Mello Vargas, FAU-USP.
At UNICAMP, students answered a similar question about symmetry in anonymous assessment questionnaires. Some of them pointed out how the use of symmetry can lead to interesting solutions, with complex compositions based on simple units. One student stated that the different types of symmetries and examples presented worked as stimuli for his creativity, opening new possibilities for shape exploration.

In summary, it was possible to infer that most students were surprised to find the creative uses of the "new" types of symmetry presented, which they probably had not considered before. At the same time, the possibility of developing symmetric designs with the help of a custom-made, CAD-based environment seemed to encourage them to try out some of the otherwise hard-to-figure-out composition schemes.

VI.e.2. Recursion

Besides the designs developed in class, FAU-USP students also showed their understanding of "recursion" in a homework question (see next page).

It is interesting to see how Marcelo Nakazaki related the shapes generated in Recursion Assistant to Frank Ghery's designs. Fábio Bellini and Daniela Costa, on the other hand, emphasized the question of scale, although this is not necessarily a characteristic of recursive design. All the three students were able to understand the advantages in using shapes that refer to each other, both in terms of space perception and intelligibility of the design. Thiago Lessa stated his concern with building performance, a typical weak point in exclusively formalistic designs.
"Discuss the advantages and disadvantages of using recursive processes in architectural design."

"The use of a recursive process in a design project can be interesting if, for example, the architect likes to play around with scales, volumes, etc., like Frank Ghery. In what concerns the perception of space, it is easier to understand a space that displays a certain pattern, which becomes even more interesting if we understand what [pattern] the architect had proposed for his building. From an aesthetical point of view, a building designed with a recursive process is less monotonous."

**Marcelo Nakazaki, FAU-USP**

"The perception of space becomes much more powerful; the user can 'understand' that a smaller space makes direct reference to a bigger one and vice-versa."

**Fábio Bellini, FAU-USP**

"I can't understand the advantages in terms of detailing, but in what concerns the construction I believe the relationship of the parts with the whole allows to define modules and easily reproduce them changing only the scale factor. In what concerns space perception,... it is possible to make users establish relationships among the different parts of the building."

**Daniela Costa, FAU-USP**

"The recursive process generates interesting spaces, but this could favor aesthetics instead of building performance aspects."

**Thiago Lessa, FAU-USP.**

**VI.e.3. Parametric shapes**

FAU-USP students stated their comprehension of parametric shapes in the homework question on next page. Daniela Costa and Fábio Bellini stated their understanding of parametric shapes as a way to create a language of designs. On the other hand, Stella Tomyioshi's understanding of the concept was based on one of the architectural examples she had presented for the research assignment (Niemeyer's Copan building), which resembles some of Aalto's apartment buildings (see Figure 97) shown in this topic's lecture. Both Stella and Renata demonstrated a concern with the pragmatic side of parametric shapes by mentioning the possibility of using special applications to automatically draft parametric building details.
"Discuss the advantages and disadvantages of using parametric shapes in design."

"We can make users perceive differences in ceiling heights, for example, inducing sensations of space enlargement or confinement, and relate the different parts of the building... We can compare different buildings that are parametric variations [of a basic shape]... or generate new buildings that are clearly related to the previous ones."

Daniela Costa, FAU-USP

"It is easier, for the user, to identify 'patterns' of buildings; visual identification is much stronger and obvious."

Fábio Bellini, FAU-USP

"The use of parametric shapes in design may be interesting, for example, in an apartment building plan with a regular structure, but not completely orthogonal, where the number of rooms is the same in all the units, despite their different shapes (such as the Copan building, for example). In this case, the variation of dimensions is small and the user will seldom note the difference. In what concerns the construction, it will be more complicated. In terms of detailing, the use of computer applications can facilitate the work, if compared to traditional drafting methods."

Stella Tomiyoshi, FAU-USP

"In what concerns detailing, a parameterized design will be easier to develop because details are similar... A VBA application can be created to draft details according to the given parameters."

Renata Figueiredo, FAU-USP

VI.e.4. Design rules

Due to time constraints, specific exercises with rule-based designs could not be developed at the FAU-USP mini-course, but the concept of design rules was introduced along with the recursion topic. At UNICAMP, rule-based design was restricted to the application of "set" rules with 2D and 3D Rules Assistant. Nevertheless, in both cases the Palladian Grammar was presented and explained as an example of the generation of designs through the application of rules from a collection of rules that defines a language. The implementation of true rule-based design systems in the computer is a complex task that involves recognizing emergent shapes and matching them with pre-stored ones. This technique increases exponentially the number of situations in which design rules can be applied, but was not within the scope of the proposed instruction system.
VI.e.5. Generation

After the session on generative systems at the FAU-USP minicourse, students were asked about the advantages and disadvantages of generating multiple design alternatives with the help of a CAD application.

"In your opinion, what are the advantages and disadvantages of using a CAD program for automatically generating design alternatives to an architectural problem?"

"The advantage is the variety of solutions that are automatically offered. The disadvantage is the time required to study all these options when the objectives are not clearly stated."

Thiago Lessa, FAU-USP.

"The computer's capacity to generate a good solution is directly linked to the user's capacity to define conditions and constraints of the problem...It can generate possible solutions within the constraints, some of which would not even be considered by the user. Therefore, it frees the user from the repetitive process of drafting [design alternatives], leaving more time for the 'more noble' parts of the design process."

Fabio Bellini, FAU-USP.

"For me, using CAD in that phase [of the design process] could be interesting, but I've never missed it when developing my own projects...maybe because I didn't know I could make studies in that way in a CAD program."

Laila Milek, FAU-USP.

"I believe that using CAD to generate alternatives is only worth it in a very preliminary phase [of the design process]."

Marcos Donizete, FAU-USP.

"If the number of alternatives is too large, the CAD program will crash. Also, in the case of too many constraints, it is not worthwhile developing a special application for the generation of alternatives."

Stella Tomiyoshi, FAU-USP.

"I would find it very bizarre if an architect could find, with the help of the computer, a better arrangement than the one he had thought."

Marcelo Nakazaki, FAU-USP.

"It is important to use it with care, without the assumption that [the computer] will do everything on its own."

Daniela Costa, FAU-USP.
The session on the computer generation of design alternatives was one of the most polemical ones at the FAU-USP mini-course, resembling UNICAMP final reviewers' reaction, when the bathroom layout program was presented. Answers to the question above were in general long and presented both positive and negative aspects. For this reason, only the most significant fragments were quoted here.

According to Thiago Lessa, in the case of ill-defined problems the number of possible solutions produced may be too large, requiring too much time to analyze them. Fábio Bellini stressed the direct relationship between the automatic generation of meaningful alternatives and "the user's capacity to define conditions and constraints on the problem". Students had the opportunity to experiment this also in the following session, with a design algorithmization exercise. Laila Milek's statement shows the importance of informing architecture students about all possible uses of the computer in design, while Stella Tomyioshi's answer shows her understanding of the technique's limitations. Marcos Donizete pointed out the fact that the technique could be used in the early phases of the design process. On the other hand, Marcelo Nakazaki still seemed a little skeptical about the automatic generation of alternatives, and Daniela Costa feared the assumption that every problem could be solved with the help of computers.

Students' responses to the question above show a concern with the definition of design constraints. Although constraints have been traditionally seen by architects as a restriction to creativity in design, in the case of automatic generation, students acknowledged the advantages of clearly defining them beforehand. As seen in the classroom exercise with Alternative Generator, this strategy reduced the number of alternatives to a manageable value.

The fact that some students criticized the automatic generation of alternatives does not mean that they did not understand the concept behind it. Neither does it eliminate the need to introduce and discuss this topic in class. On the contrary, it stresses the importance of developing more meaningful, practical exercises to demonstrate that automated generative systems may effectively lead to the discovery of more creative solutions.

**VI.e.6. Algorithmization**

After the algorithmization session, FAU-USP students were asked
to formalize, in a small conceptual program, the criteria, variables and constraints that are present in a routine design process:

**Assignment:**

"Develop a small conceptual program for a design situation in which the same steps are always followed. Base your program on the script provided below:

- Establish initial assumptions;
- Define design criteria;
- Determine the system's variables;
- Attribute default values and constraints to each variable;
- Think about all possible design decision paths, identifying any node points in the process;
- Describe all the procedures involved in the process;
- Finally, test your ‘program’, attributing arbitrary values and proving its efficacy."

**Student A:** Parking lot design program

Draw a rectangle
Choose the rectangle's base
Place parking spots at 90° against the rectangle's base, starting a 5m from the beginning (parking lot's entrance). From that point on, parking spots will be at least 2.5m wide and 5m long.
Place other rows parallel to the first at a 5m distance from each other, until they reach the opposite side of the rectangle, and keeping 5m from each side.

**Student B:** Movie theater auditorium - layout program

The project is for a movie theater [auditorium], which must be rectangular, with minimum dimensions of 20x50[m]. The screen will be placed on the shorter wall.

The [system] variables are the length and breadth [of the rectangle], in the first place, and the screen's size, which varies from 18 to 30 meters. The door must be located on the left corner, opposite to the screen. In front of the door is a passageway, whose width varies from 1.5 to 2.5 m., running beside the chairs.

The fire exit is on the left side of the screen, on the same wall where the screen is, in a place to be determined by the program's user.
The rows [dimensions] are determined according to the room's dimensions, according to the following criteria:

- Chair dimensions: 50x40[cm]
- Distance between rows: 50 cm
- Passage way: minimum 1.5m (one at each side of the rows)
- Door width: from 2 to 3m.

The screen is on the opposite side of the entrance, in the middle of the wall, with the maximum dimension of 2m less than the length of the wall, with a minimum width of 18m and a maximum width of 30m.

**Student C:** Program for the automatic drafting of opening frames (doors and windows) in architectural plans

Initial assumptions:
- Frames can be made of wood or metal
- Frames may have standard or custom dimensions
- Draw sills (windows) and thresholds (doors)

Variable constraints:
- Wall thickness: 10 to 30 cm.
- Opening width: standard or click on two points [user-defined]
- Frame thickness: 1.5 to 3cm. (metal); 3 to 5 cm (wood)
- Door thickness: 1.5 to 3 cm.
- Trim: Yes/No

Options:
- Doors: sliding/swinging
- Windows: sliding/swinging

Decision-process nodes:
- Door: if single: opens to the right/ if double: opens to the left
- Sliding window: if two panels: [just] glazed/ if three panels: [include] Venetian blinds

Procedures:
- Define if it's a door or a window
- Choose a material: wood or metal
- Draw threshold/windowsill? (Yes/No)
- Standard dimensions? (Yes/No)
- How many panels? (Door: 1 or 2; window: 2 or 3)
- Sliding/swinging
- If 1 panel door, [then specify if it] opens to the left or right
- If window, [then specify if ] glass or Venetian blinds

**VI. Data Analysis**
VI.Data Analysis

The examples above show that the mere attempt to formalize a design process or part of it is not an easy task. Most of them lack important variables and design options. For example, student A could have added some conditional requirements in the parking lot layout conceptual program, such as:

In case the parking spots are laid out at...
- ...90°, the passage should be at least 5m wide.
- ...45°, the passage should be at least 4m wide.
- ...30°, the passage should be at least 3m wide.

For this reason, this type of exercise should be encouraged to help students develop a clear understanding of design alternatives, criteria and constraints. Unfortunately, due to time limitations and students' lack of more advanced programming skills, FAU-USP students were not able to evolve the preliminary scripts above into actual programs. At UNICAMP, a student was able to improve the sample program presented (Bathroom Layout Generator), by introducing new variables and design alternatives to it.

VI.e.7. Emergence

After the session on emergent shapes at the FAU-USP mini-course, students were asked to compare the finding of emergent shapes in a CAD software with traditional methods (see page 143).

Some students, like Stella Tomyioshi, Daniela Vaz and F.H.T., stressed the different nature of the emergent shapes found in the CAD program, in comparison with the by-hand methods. According to them, those were more "accidental" than in hand sketch. Their opinion did not contradict Juliana Beividas's, who pointed out the "intention" given to the tracing in hand drawings, as a result of something that is already on the designer's mind. For Marcelo Nakazaki, the process occurs "naturally". In other words, the use of the CAD-based techniques may induce the discovery of more unexpected shapes, while in hand drawings the designer somehow orients these "findings" with his or her original intentions or instincts. That was probably the reason why Laila Milek mentioned her difficulty in fitting the architectural program when using CAD to find emergent shapes.

Table 25 shows some emergence exercises from both schools, with symmetric or recursive compositions on the right side column. The 3D exercise (only at FAU-USP) is particularly interesting, based on the use of AutoCAD’s built in Boolean solid operation commands.
### VI. Data Analysis

#### Table 25: Emergence exercises.

<table>
<thead>
<tr>
<th>2D at UNICAMP</th>
<th>Unexpected shapes</th>
<th>Symmetric or recursive shapes</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image4.png" alt="Image" /></td>
<td><img src="image5.png" alt="Image" /></td>
<td><img src="image6.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image7.png" alt="Image" /></td>
<td><img src="image8.png" alt="Image" /></td>
<td><img src="image9.png" alt="Image" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2D at FAU-USP</th>
<th>Unexpected shapes</th>
<th>Symmetric or recursive shapes</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image10.png" alt="Image" /></td>
<td><img src="image11.png" alt="Image" /></td>
<td><img src="image12.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image13.png" alt="Image" /></td>
<td><img src="image14.png" alt="Image" /></td>
<td><img src="image15.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image16.png" alt="Image" /></td>
<td><img src="image17.png" alt="Image" /></td>
<td><img src="image18.png" alt="Image" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3D at FAU-USP</th>
<th>Unexpected shapes</th>
<th>Symmetric or recursive shapes</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image19.png" alt="Image" /></td>
<td><img src="image20.png" alt="Image" /></td>
<td><img src="image21.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image22.png" alt="Image" /></td>
<td><img src="image23.png" alt="Image" /></td>
<td><img src="image24.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="image25.png" alt="Image" /></td>
<td><img src="image26.png" alt="Image" /></td>
<td><img src="image27.png" alt="Image" /></td>
</tr>
</tbody>
</table>
"What are the differences, in what concerns the discovery of emergent or accidental shapes, when hand-sketching and working in a CAD program?"

"In the CAD program shapes that would hardly be found on a hand drawing emerged."
Stella Tomyioshi, FAU-USP

"Relationships are more unexpected, more alternatives can arise."
Daniela Vaz, FAU-USP.

"There are more chances of different shape combinations,... of accidental forms, much more than in a manual process."
F. H. T., FAU-USP.

"There are more possibilities. I started doing something and I changed in the middle when I saw something cooler."
Rodrigo Azevedo, FAU-USP.

"Usually the difference is not very big in what concerns the discovery of shapes, but the program is more versatile, allowing to erase and restart the drawing as many times as necessary, or simply [focus on] the interesting parts, reproducing and editing them quickly and easily."
Renata Figueiredo, FAU-USP.

"When working by hand you can give intention to the tracing, forcing the emergence of the shapes you are interested in."
Juliana Beividas, FAU-USP.

"New shapes emerge naturally as we sketch."
Marcelo Nakazaki, FAU-USP.

"It's hard to think about the architectural program in the CAD software."
Laila Milek, FAU-USP.

VI.e.8. Confusion between concepts

In the first parts of the prototypical system described in this thesis, the introduction of concepts evolved from more specific or formal ones to the less specific and rigid ones. As explained before, compositions generated by older concepts could always be obtained by
new concepts, but were not the only types that could be created by them. For example, symmetric compositions can be obtained through the recursive application of "design rules", while the notions of "parametric variations of a shape" and of "rule-based designs" were inevitably present in the introduction of the "recursion" concept. Similarly, applying design rules recursively or parametrically were two of the possible ways of generating rule-based designs. In other words, the pedagogical approach was based on the introduction of progressively more complex methods that were nested within each other, allowing students to take advantage of what was learned from one topic to the other.

Nevertheless, in the experimental courses, students sometimes confused by the overlapping concepts, especially the first ones. Even "symmetry", the concept with which students were most familiar, was sometimes confused with recursion and parameterization. This confusion can be considered as a sign that students were able to establish connections between the different generative systems presented. Yet, a more formal introduction of the concepts after the initial informal analogies could have avoided this situation. An evidence of the confusion was given by students' comments in class or by e-mail, such as those on the next page. In order to help students clarify this confusion, specific questions about how the concepts related to each other were asked at FAU-USP. For example, in one specific assignment, students were asked to find out the compositional rules that generated the different kinds of symmetries studied.

A second evidence of the confusion between some concepts was the occurrence of many symmetrical compositions among designs presented for other assignments, especially at UNICAMP, where students had not been warned about it. Table 26 shows a comparison between this type of accidental result at UNICAMP (in recursive and ruled-based composition exercises) and FAU-USP's specific assignment described above.

Among the three types of symmetric compositions generated by rules, wallpaper symmetry seemed to be the hardest to achieve, with one single occurrence in each case study. Figure 98 shows an unsuccessful attempt to generate a wallpaper pattern. The reason for this is probably the fact that, in order to generate a wallpaper composition based on a single rule, it is necessary to overlap shapes, which is considered a typical bad practice or mistake in
VI. Data Analysis

Comments:

"For me there are many relationships between symmetry and recursion. The same composition can be symmetric and recursive and I often mix them up and cannot tell which is which."
FAU-USP student, after the session on recursion.

"For me, recursive, symmetric and parametric shapes are very alike and you often can see the three situations in the same composition."
FAU-USP student, after the session on parameterization.

"Parameterization is a type of recursion, but with different variables. In a certain way, it is possible to generate a symmetric building [plan] with parameterization, but not always the opposite."
FAU-USP student, answering a question about parameterization.

Table 26: Intentional and accidental symmetries generated by recursion.

<table>
<thead>
<tr>
<th>UNICAMP workshop</th>
<th>FAU-USP mini course</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frieze symmetry</td>
<td></td>
</tr>
<tr>
<td>Wall-paper symmetry</td>
<td></td>
</tr>
<tr>
<td>Circular symmetry</td>
<td></td>
</tr>
</tbody>
</table>
VI.Data Analysis

CAD drafting.

The last evidence of the confusion between the three concepts was in the architectural examples submitted as research homework, which is better explained in the next topic.

VI.e.8. Research

Table 27 and Table 28 show research questions and architectural examples presented by UNICAMP and FAU-USP students during the experimental courses.

The search for architectural examples of the concepts introduced was probably the most pedagogical exercise in the two experimental courses. Even when examples displayed characteristics of more than one concept, it was possible to explain, in the frequent homework reviews, in which features of the buildings each concept was present. At UNICAMP, some of the examples searched for were even recreated with Recursion Assistant or in programs developed by the students themselves. At FAU-USP, some examples of symmetry from the architectural repertoire were used as a motif for the corresponding architectural design assignment (Figure 100).

VI.e.9. Summary of understanding of concepts

In summary, students' answers leave no doubt about their understanding of the concepts introduced. On the other hand, some of the images presented as examples of the occurrence of symmetry, recursion, and parameterized shapes in architecture could be easily representative of more than one of these concepts. This fact made students confused sometimes, but was important to show them how all topics were inter-related. The research exercises allowed students to generalize them to other cases, training their eyes to see architecture from a computational point of view. Some students reported how the concepts became clearer to them as they were able to recognize them in architectural examples.

Figure 100: Symmetric design with motifs based on Aalto's and Niemeyer's buildings.
VI. Data Analysis

Table 27: Research exercises at UNICAMP.

**Symmetry**

Symmetry examples by Gabrielle Damaso; recursion examples by Gabrielle Damaso (top), Marina Otaviano (middle top), Giovana Bianchi (middle bottom) and Daniel Moreira (bottom left); parametric shapes example by Marina Otaviano.

**Recursion**

**Parameterized shapes**
### VI. Data Analysis

**Table 28:** Research exercises at FAU-USP.

<table>
<thead>
<tr>
<th>Symmetry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples suggested by Leandro Robles and Marcelo Nakazaki.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Recursion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples suggested by Marcelo Nakazaki and Daniela Costa.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameterized shapes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples suggested by Daniela Costa.</td>
</tr>
</tbody>
</table>
VI.f. Applying the concepts to architectural design

Architectural design exercises using the concepts of symmetry, recursion and parameterized shapes were proposed as homework at the FAU-USP mini-course. At UNICAMP, where part of the students had no architectural background, the direct application of the concepts to architectural design was only suggested as one of the options for the final project.

Table 29 shows exercise directions for each topic and some examples submitted by FAU-USP students. In the symmetry design exercise, only a minority of the students followed the directions to develop very conceptual, mass studies. Many students presented elaborated plans (some of which had already been developed for studios or other classes), showing wall widths and architectural details such as doors and windows (Figure 101). This fact was symptomatic of students' original attitude towards CAD as an overly precise tool. It was probably unacceptable to some of them to present preliminary, imprecise studies in a CAD program.

Marcos Donizete proposed different symmetrical compositions based on architectural unit designed by Alvar Aalto. Daniela Costa created her own basic unit and explored different arrangements. Marcelo Nakazaki used Recursion Assistant to design a spiraling museum with a central open space. The lighter areas represent circulation and the darker areas are different-sized exhibition spaces. Daniela Costa used the same application to develop three different plans for a movie theater with different-sized auditoriums. The lighter areas represent the entrance halls with emergent shapes.

In the parametric design assignment, Stella Tomyioshi created different-sized classrooms and corresponding hallway sections with a single AutoCAD block inserted at different X and Y scales. Daniela Costa used the same technique in 3D, proposing different building configurations based on parametric variations of a basic block.

Although the architectural design projects presented were not geometrically complex and were based on simple computational techniques, they were valid as an opportunity for students to experiment with the new use of CAD in building design.
**VI. Data Analysis**

**Table 29: Design exercises at FAU-USP.**

**Symmetry**
"Develop a quick study for a residential project, using one of the symmetric arrangements presented in class (bilateral, frieze, wall-paper or circular) or their combinations. Use the viewports method or any other technique, including hand-sketching."

**Recursion**
"Develop a quick study for an institutional building, using one of the recursive techniques presented in class (concentric, subdivision, branching) or their combinations. Use the viewports method, Recursion Assistant or any other technique, including hand-sketching."

---

In these row houses, the lighter areas correspond to the bedrooms and living room, and the darker areas to the kitchen and bathrooms.

---

Symmetric compositions with a unit based on a building by Niemeyer.

---

Symmetric compositions for housing units based on a T-shaped unit.

---

In this museum project, the circulation happens in the lighter areas around the patio, and the darker areas are differently sized exhibition rooms.

---

In this school, the darker areas represent the corridors and the lighter areas are classrooms.

---

In this auditorium complex the darker areas represent the projection rooms, and the lighter areas are the halls. This student explored different recursive combinations based on similar basic shapes.
**VI. Data Analysis**

**Table 29: Design exercises at FAU-USP**  
(cont.)

| This student explored the use of a single basic unit (on the right) at different X, Y and Z scale parameters, to create buildings for different uses. 
| Daniela Costa |
| This student created a block formed by a classroom and hallway section to help her design a school plan. The insertion of the basic block at different scales resulted in hallways that are proportional to the classrooms in front of them. 
| Stella Tomyioshi |
| Two building designs by Rodrigo Azevedo and Laila Milek. |

**Parameters**

"Develop a quick study for a building with cultural purposes, using parameterized shapes (by hand, with AutoCAD blocks or with VBA programming)."

**Emergence**

"Develop a mass plan for any building type, starting from a rule-based 3D composition and using Boolean operations to add, subtract and intersect shapes."

Unfortunately, due to time constraints, most students developed their 3D emergence exercises only abstractly, without an architectural programme in mind. The compatibility of this technique with the meeting of architectural requirements demands both 3D modeling and architectural design experience."
VI. Data Analysis

* The translation of the expression "computational design" to Portuguese is a complex issue. While in English the word "computation" means "the act of computing or calculating", in Portuguese the corresponding word "computação" can mean "the act of counting" (not calculating), but is generally associated with "the study or the use of computers". For this reason, the word "computation" was rarely used in the experimental courses, and the term "projeto lógico" (logical design) was used instead, as a reference to Mitchell's The Logic of Architecture.

Table 30: Post-treatment and follow-up questions concerning attitude towards CAD in architecture at UNICAMP and FAU-USP (open questions).

In your opinion what are the reasons for the use of computation in design, and what are its most common applications?

Do you think the workshop has changed your attitude towards the role of CAD in design?

VI.g. Attitude-change towards CAD

The change in students' attitude towards the use of CAD in architecture, the most important issue in this thesis, could be measured in part through the comparison between pre- and post-treatment questionnaires. I acknowledge that a deeper understanding of the process should not only take into account students' declarations right after the courses, but also their actual actions in subsequent projects. Nevertheless, this type of follow-up assessment, which would require a much longer time-span, was not possible due to the study's constraints. On the other hand, a hint of this type of action or at least intention of action was suggested by UNICAMP students' post-treatment questionnaires three months after their exposure to the new instruction system (see next item, External reach).

Table 30 summarizes the answers to the questions asked in both case studies in order to assess attitude-change, allowing to compare students from the two institutions. UNICAMP students apparently showed a better understanding of the first question. It must be noted, though, that the word "computation" was translated to Portuguese in different manners in each case: in the first case as "lógica" (logic) and in the second as "lógica computacional" (computational logic). The introduction of the word "computacional" apparently made students link the question to the use of computers in general. The presence of attitude-change was better measured by the second question, to which the majority of students answered yes both at FAU-USP and UNICAMP.
As explained before, at the UNICAMP workshop few students filled out pre-assessment questionnaires. Nevertheless, most of them did mention their previous attitude towards CAD in the post-treatment questionnaires to explain how it had changed. Seven among the 8 students who returned this questionnaire acknowledged a change in attitude. The only student who did not, an engineering undergraduate, said he already had a knowledge of programming (Auto Lisp) that allowed him to use CAD beyond representation (although it is possible that he was referring to automation and analysis, rather than shape exploration).

"Do you believe the workshop has changed your attitude towards the role of CAD in architectural/civil engineering design?"

"Certainly, we usually see CAD only as a drafting tool and the workshop showed how it can also be a generative tool, with very creative results."

Giovana Bianchi, architecture undergraduate student, UNICAMP.

"Yes. I've learned to see other possibilities in the use of CAD. Still in a very immature way, but since then I started to pay attention to everything that I could automate in the creative process of design. I've also started to imagine new ways to reduce the old repetitive processes."

Marina Otaviano, architecture undergraduate student, UNICAMP.

"Yes, it has. Although I used to see CAD as a drafting tool, I started to see possibilities in the composition and organization of architectural elements. And that in different stages of the design process and in different scales. A mass study or a building detail can be interpreted in different manners when their compositional logic is understood. This thought already means a new way to face design and the [CAD] tool by which it is developed. What is most impressive in all this is to perceive a 'grammatical' property of the process, something I had never noticed before: as in [verbal] languages, where there are grammatical rules and a vocabulary, programmed CAD allows us to express ourselves in a personal manner through the same rules and elements. A drafting tool can be adapted to an individual and personal way of thinking, both for the conceptual phase and the development of the project."

Daniel Moreira, M.Sc. in architecture, UNICAMP.
The examples above show that these students acknowledged a creative use of CAD, despite their previous belief that CAD was simply a drafting tool. It is interesting to note, in Marina Otaviano’s answer, how she used the word "automation" when referring to uses of computation both in the creative and the routine parts of the design process. The use of the expression "to see possibilities" in two of the answers above is another clear indication of a change in attitude.

Daniel Moreira’s answer is a clear statement of how this student was able to assimilate all the concepts present in the workshop, and probably infer a few more. He suggested the use of computation in both analysis and synthesis, at different scales and in different phases of the design process. By mentioning the "grammatical property of the process", Daniel showed a clear understanding of the brief shape grammars contents presented in the introductory lecture. Finally, he proved seeing new possibilities in the use of CAD when stating that "a drafting tool can be adapted to an individual and personal way of thinking, both for the conceptual phase and the development of the project".

FAU-USP students’ answers to the same question right after the last session of the mini-course showed a similar change, yet without the same enthusiasm (see next page). Daniela Costa’s answer clearly states how she used to see CAD as a precision tool, and then started seeing it as a shape exploration aide for the initial phases of design. Fábio Bellini is an interesting case of a student who reported having deliberately searched for meaningful uses of the computer in design before. It was rewarding to hear from a student like this that he was finally able to see some of the "wider possibilities" (Mitchell, 1990b, 483).

On the other hand, Laila Milek’s answer is symptomatic of a typical attitude of students who learned to design without the use of computers - even if just for drafting. This is still the case for many students at FAU-USP, where CAD subjects are not mandatory and elective subjects are only offered to seniors. Although most students end up using computers by their own means (learning to use CAD software by themselves or in private courses), some still go through the entire architectural education drafting and designing in the traditional way. The lack of familiarity with the medium obviously makes the exploration of the new uses more difficult.
"Do you believe the workshop has changed your attitude towards the role of CAD in architecture?"

"I think the main thing I've learned is to change the idea that CAD is just a precision tool and a representation tool, therefore only useful in the last phases of a design process. I've learned that we can use CAD in the initial phases of a design process, in the exploration of shapes, in the formalization of ideas, in other words, it can help us to develop creative, different ideas."

Daniela Costa, FAU-USP.

"No wonder! Despite my [previous] inquiries, I had not found an answer about a really intelligent use of CAD; before, I used to understand customization as just a process to save time in repetitive tasks, but it can help humans in processes that are much more complex."

Fábio Bellini, FAU-USP.

"No. I still believe it is easier to think about shapes on top of hand drawings."

Laila Milek, FAU-USP.

VI.h. External reach

At UNICAMP, students were asked questions about the workshop’s external reach right after the final review and three months later. Among the 9 students who answered the first question (see next page), most mentioned their intention to use CAD for both applications, but there was a slight preference for automation, especially among engineering students, as shown in Chart 18.

**Chart 18**: Probable application of the techniques learned (semi-open question) suggested by 9 of the 12 UNICAMP students presenting final projects.
VI. Data Analysis

"Do you think from now on you will use more logical processes in your curricular projects? In case you do, would you use logic as a theme or as a way of automating repetitive processes?"

"From now on I will pay more attention to the use of logic in architecture, and will use it especially to automate repetitive processes."

Gabrielle Damaso, architecture undergraduate student, UNICAMP.

"I want to use both, both surprised me."

V. W., architect, graduate student, UNICAMP.

"I will use it more to explore interesting shapes and relationships."

Giovana Bianchi, architecture undergraduate student, UNICAMP.

In a post-assessment questionnaire sent by e-mail three months after the end of the workshop, the 12 students who had completed the workshop were asked the question on next page. All the 8 students who returned this post treatment questionnaire reported at least some intention to use the new concepts learned. Half of them mentioned the use of computational techniques in the automation of design processes, and the other half mentioned design exploration.

An important fact that could be observed in the answers was the fact that most students did not necessarily link the application of concepts to the development of a program. On the contrary, some reported an influence in the analysis of building design or when developing architectural projects by hand.

Similarly, FAU-USP students were asked if they thought they would use the concepts they had learned in their future design projects (see page 156). Among the 15 students who answered this question, nine mentioned an intention to use the techniques and concepts learned to generate shapes in the creative phases of the design process. Fewer students mentioned automation at FAU-USP than at UNICAMP. Apparently, the more technical character of UNICAMP and the presence of engineering students there coincided with a higher interest in automation than in creative design.
"Have you had the opportunity to apply what you have learned in the "Logic Design" workshop throughout the semester?"

"We tried to use recursion when designing a building, but it did not work out due to the limited dimension of the site and the architectural program."
Giovana Bianchi, architecture undergraduate student, UNICAMP.

"Many times, while developing a design, we say: 'imagine this shape used in a recursive way with that program! It would look very interesting...' But unfortunately we never get to test it, mainly for the lack of time."
Paula Roberta Baratella, architecture undergraduate student, UNICAMP.

"I'm always trying to find the rules that were used in designs I see in the magazines ... I think engineers use more parameterization, architects still show some resistance, that story about the hand trace, the 6B lead, nonsense! I love to hand sketch and I loved what you taught. I do not see any conflict."
Roberto Itapura, architect, UNICAMP.

"Although I haven't developed any program so far, I now see any work developed in plain CAD as an improvisation."
Daniel Moreira, M.Sc. in architecture, UNICAMP.

---

**Chart 19:** Possible application of the techniques learned (open question) suggested by FAU-USP students after the mini-course.
VI. Data Analysis

"Do you believe you will use what you have learned in this course in your design projects from now on? How would you do that?"

"The time you spend programming the computer can be used for other activities. Programming is only useful for architects who have many similar projects, like in the large offices, which is not my case. I believe I will still use CAD only as a graphic representation tool."
S. T. A., FAU-USP.

"Maybe I'll use these techniques in the initial phases of design, but it's hard to adopt right away a new way of designing, replacing that one that we've learned. But these techniques can help us a lot together with the process of designing that I have already incorporated."
Stella Tomiyoshi, FAU-USP.

"Yes, for example, I can use Recursion Assistant to sketch volume schemes in the initial phases of a design process, or use parameterization to study the work of architects and base my design on his proportions, which would lead to a better understanding of his work."
Marcelo Nakazaki, FAU-USP.

It is possible to conclude that most students in this study presented at least an intention to apply computational techniques to their design activities after being exposed to the prototypical system ideas. Nevertheless, a more significant reach would only be possible if other instructors could also encourage the same type of thinking.
VI. Data Analysis

Summary

The different types of data collected, consisting of assessment questionnaires, journals, images of architectural examples, graphics and programs, provided clues about different aspects of the proposed systems' instances-in-action. They allowed to assess students' performance and the efficacy of the courses' structure and materials. Students' performance, on its turn, provided clues about the efficacy of the theoretical proposal, and the impact on their attitude towards the use of CAD in architecture. They also provided elements that led to the drawing of unexpected conclusions about students' attitude towards programming and the applicability of the system to other fields, such as civil engineering.
VII. Conclusions

“It has been argued recently that the study of form itself should again become a central enterprise in architectural research and education… this view is justified only to the extent that knowledge about form can be obtained and taught in a scientific way.” (Stiny 1976, 187)

VII.a. Confirmation of original assumptions

Two out of the three assumptions made in the introduction of this thesis (Chapter I Introduction - Assumptions), before the development of the proposed prototypical system, were confirmed in the surveys carried out in the pilot work and case studies. At least among the students in this study, CAD was originally seen as an exclusively representational tool, and its inclusion in the architectural curriculum was justified mainly by job market demands. The few uses of CAD in the creative phases of design reported were usually limited to the three-dimensional manipulation of solids and the rapid representation of design alternatives where a few elements were changed "manually" to generate slightly different versions of the same design.

Another assumption that could be confirmed in pre-treatment questionnaires was the lack of knowledge about the history of CAD, which I consider a key factor in the understanding of CAD’s "wider possibilities" (Mitchell 1990b, 483).

The bias against programming among architecture students that I had originally assumed was not confirmed, and students acknowledged its importance in order to make a better use of CAD. Nevertheless, pre-treatment questionnaires revealed that most students did not have any programming experience at all, and during the experimental courses they often expressed their frustrations when learning the new techniques. This fact reinforces the need to introduce programming in a way that can provide a rapid return of the time invested (Negroponte, 1975a), so students will not feel discouraged by supposing there is a huge gap between what they know and what they need to know in order to produce useful programs. In the two case studies in this thesis, students with just a little programming experience were able to develop meaningful applications in a short period of time, by programming CAD software.

VII.b. Answering the original research questions

According to Nunan (1992), a research effort is not worthwhile if (1) the conclusions it leads to are impossible to apply in the real
world, and (2) its results are not diffused to the people who could make immediate use of it or extend the study to arrive at new findings.

In the present study, although not all research questions could be answered, at least the conclusions derived from the experimental courses have proved being applicable in the real world. With proper communication, the proposed system might be instantiated by other researchers in new contexts, providing data for more generalizable results, and helping solve the questions that were left unanswered.

The initial research questions, originally asked in Section II.d., are posed again:

1. Would it be possible to mix CAD practice and computational design theory successfully in the same instruction system?
2. How would technical and theoretical issues be balanced in this system?
3. Would students exposed to this system change their attitude towards the use of CAD in architecture?
4. Would students exposed to this system be able to apply the new concepts in other subjects and studios?
5. Would the teaching of CAD with a computational design approach give it a new relevance in the architectural curriculum, avoiding the skepticism that is starting to emerge?
6. Would the new approach have any positive influence in design?

Given the study constraints, the only questions that could be fully answered were items 1 and 2, with the prototypical system developed, explained in detail in Chapter IV. As shown in Section V.c, all students returning questionnaires at UNICAMP were satisfied with the workshop. When asked if they liked better the theoretical or practical parts of the course, students were in favor of both, with a slight preference for the practical exercises. Similarly, at FAU-USP, students were in general very satisfied with the system. Some answers to a question in the post-treatment questionnaire pointed to the importance of the practical aspects in the learning of concepts, adding evidence to what could be observed in class (see next page). Even the readings, a part of practical courses that is usually disregarded, were acknowledged as interesting and important to the understanding of concepts in both case studies by students and participant observers.
VII. Conclusions

<table>
<thead>
<tr>
<th>Which were the best aspects of the course?</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;The course proposal is very good (…) It was in between a practical and a theoretical course.&quot;</td>
</tr>
<tr>
<td><strong>Fabio Bellini, FAU-USP</strong></td>
</tr>
<tr>
<td>&quot;The method used was very interesting, divided in a sequence of learning stages, not forgetting to emphasize the practical side, which was very important to fix what was learned with the readings and lectures.&quot;</td>
</tr>
<tr>
<td><strong>Daniela Costa, FAU-USP</strong></td>
</tr>
<tr>
<td>&quot;The examples were visual, not theoretical ones, which helped in the assimilation of the concepts.&quot;</td>
</tr>
<tr>
<td><strong>Flávio Tanabe, FAU-USP</strong></td>
</tr>
<tr>
<td>&quot;There was a clear distinction between theory and practice in each session.&quot;</td>
</tr>
<tr>
<td><strong>Marcelo Nakazaki, FAU-USP</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Comments on the readings selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Very interesting reading selection.&quot;</td>
</tr>
<tr>
<td><strong>Marcos Donizete, FAU-USP</strong></td>
</tr>
<tr>
<td>&quot;The reading of scientific papers along with creative and practical activities resulted in an environment of great interest in the classroom, encouraging students to do research on their own.&quot;</td>
</tr>
<tr>
<td><strong>Prof. Doris Kowaltowski, UNICAMP</strong></td>
</tr>
</tbody>
</table>

Most important of all, in Sections VI.d and VI.e (Understanding the design environments and Understanding the concepts), it was possible to demonstrate students' understanding of the design environments and tools used and the concepts introduced through the exercises presented by them. Those exercises allowed students to put into practice the computational design theories learned, using techniques and resources at their reach. The two case studies proved also that the system is flexible enough to be applied in different formats and for different background students, even when they are mixed in the same group, which was the case at UNICAMP.

Questions 3 and 4 were directly related to the efficacy of the application of the prototypical system. The results of the two instances
of the system, explained in detail in sub-sections VI.g. and VI.h (Attitude change and External reach), show how the first expectation - the change in students' attitude towards CAD - was met in both cases, and there was at least an intention by students to fulfill the second one. External influences of the proposed system were only evaluated at UNICAMP, but not at FAU-USP, due to time constraints. It is true that most students were not able to effectively apply in other subjects what they had learned in the experimental workshop. This was due to different problems, such as the unavailability of computers in studios, the lack of time, and difficulties in making the new design techniques compatible with architectural program requirements. Yet, students reported a strong conceptual influence and an intention to do it. As suggested by Prof. Kowaltowski, the support of other instructors in the school would make the continuation of students' experience much more effective.

The applicability of the results obtained cannot be over-generalized, since the number of students surveyed was not significantly representative of the population of architecture students, even if we take into account Brazilian students only. Nevertheless, they provide a preliminary conclusion of the type of response that can be expected from students in similar contexts.

Finally, it has not been possible to answer questions 5 and 6, which would have demanded a longer time span and the collaboration of other faculty members, including studio instructors, as described above. On the other hand, other type of observations could be made and non-expected consequences of the new approach ("side-effects"), which were not foreseen by the research questions, have also emerged.

VII.c. Unexpected observations

A few unexpected observations could be made during the case studies in this research. For example, a comparison between the level of achievement of students with different technical skills in the proposed system, which was not among the objectives of this thesis, was possible because of the different emphasis given to CAD at UNICAMP and FAU-USP. Besides, some more experienced, graduate students happened to be present in the first case study. FAU-USP students were less skilled than UNICAMP students in the use of CAD software, and were at the same time much more skeptical, originally, about the usefulness of computers in the creative process.
of design, which could be observed in class discussions. Apparently, the less familiar students were with CAD software, the harder it was for them to start seeing new possibilities in the use of computers in design. One of the graduate students at UNICAMP, Daniel Moreira, who had a background in architecture and some programming knowledge, was able to develop an interesting application from both the conceptual and technical points of view, while the programs developed by less skilled students were limited to more straightforward problems, such as creating parametric variations of shapes. Daniel’s experience in programming probably saved him from being anxious about the technique, leaving more time for a reflection on the computational nature of F. L. Wright’s designs. An undergraduate architecture student at UNICAMP with no previous programming experience, instead, reported that she had to give up developing a program to generate an Aztec pyramid, due to a difficulty in solving a programming issue, and had to start a new, simpler final project from scratch.

Another unexpected comparison that could be made was among the two different formats of the case-study courses. The summer course’s compact format was probably more effective in terms of the assimilation of the technical contents, especially the programming language. In the more extensive format tried at FAU-USP, students did not necessarily have more time for completing homework, and seemed to forget the technical details learned from one week to the other. On the other hand, in the second case, students probably had a better opportunity to reflect on what was being presented at the conceptual level. This could be noticed in classroom comments and discussions, where arguments were in general more elaborated than at UNICAMP.

Finding out the level of interest in the proposed system by students in other fields was not among the purposes of this study as well. Nevertheless, the presence of civil engineering students at the UNICAMP workshop allowed observing their response to the contents introduced. One of the most interesting aspects in the workshop was that, while architects were fascinated by the new possibilities opened to them by programming, engineers seemed to develop a new sensibility towards architecture, through the use of computers and computation in design. They learned to appreciate buildings from a computational point of view, as stated in the comment below, sent by e-mail three months after the workshop:
VII. Conclusions

"I believe that now it is hard for me to develop any project without trying to apply what was seen in the workshop, as well as it is impossible to analyze a structure without trying to find the basic shape and how it was repeated to generate the building."

Fernando Basilio, civil engineering undergraduate student, UNICAMP.

Even the mix between students with different backgrounds was important. In the post-treatment questionnaire, the same student stated:

"The mix between architecture and civil engineering students was perfect. I could see the difference in how architecture [and civil engineering] students see architecture, and this will be important in my future projects."

Fernando Basilio, civil engineering undergraduate student, UNICAMP.

VII.d. Critique

My personnal criticism to this study is mainly concerned with assessment methodology issues, but also with the way the fundamental concepts were introduced.

Due to my lack of experience in course assessment and to the experimental character of the courses, I ended tried different questionnaires in each case study, which made the comparing the results of the two case studies more difficult and less effective. Because it was hard to decide on the assessment materials' most appropriate media, I experimented with different ones. It was also difficult to decide on the length of questionnaires, to choose between multiple-choice or open questions, and to evaluate the need for identification in the feedback material. For example, at the UNICAMP workshop students were asked to turn in survey forms anonymously, to encourage their sincerity, but this eliminated the possibility to link the data to the specific characteristics of the subjects (which, in that case, varied a lot), or to distinguish listeners from registered students. For that reason, most of the data collected with questionnaires at UNICAMP could not be used as evidence of architecture students' understanding of the concepts. It was also impossible to make the turning in of questionnaires mandatory and the return rate was low. On the other hand, if linking the return of questionnaires
to the final grade effectively increased the return rate at FAU-USP, some of the forms were clearly answered in a rush and open questions were frequently left blank.

Moreover, it became clear that any kind of assessment involving a large number of students should take advantage of as many automation resources as possible. In the course of this study, I often felt overwhelmed with the amount of information collected and the need to organize it in a meaningful way. The lack of automation in the handling of forms and exercises (partly because of the lack of the appropriate hardware and software resources) contradicted what I was teaching in my own courses. This issue would have been critical if the research had aimed to study a larger group.

It has to be acknowledged that the prototypical system could have included some other important computational design concepts, such as vocabularies of shapes and hierarchical structures. Yet, although there was no explicit reference to them, they were implicitly included in other topics that were considered more relevant in this study, such as in rule-based designs.

The confusion between some of the concepts by students in the case studies, particularly between recursive and rule-based design, was probably due in part to the lack of more powerful computer applications to demonstrate the use of multiple design rules applied in free sequences. Nevertheless, it may also have been the result of the very strategy used to secure their understanding. After being introduced through very informal categories and analogies to the real world, the concepts could have been reintroduced in a more formal, scholarly way.

VII. Conclusions

VII.e. Communication and future work

Research is not simply searching for information, but also making good use of what is found, so it can lead to discovery (Orna & Stevens, 1995). It is important to communicate research results to others, so the search can continue.

As Prof. Kowaltowski stated in an interview after the UNICAMP workshop, one of the greatest impediments to the use of computer-supported creative techniques is the lack of a formal link between CAD subjects and studios. In fact, even at UNICAMP, where students have access to CAD subjects and better computer resources
than at FAU-USP, a student complained about the impossibility to apply the concepts learned in design because there were no computers in her studio. It is true that she could probably still try to apply the concepts learned in a "by-hand" manner (which another student actually did), but in that case the effects would be much slower and limited in terms of complexity if compared to the results obtained in the experimental workshop.

For the reasons above, it is my intention to communicate the results of this research by publishing articles and presenting papers in CAD and computational design conferences. Besides, it is also my intention to publish a handbook on the topic, which could be used by CAD instructors willing to add a new approach to their courses or by students who want to try out the creative side of CAD. The publishing of the results of the present work is expected to encourage other instructors to try a new approach to CAD education and to help revert the general skepticism about the use of computers as a true architectural design aide.

Finally, in my future work as an architectural design and CAD instructor, I hope to be able to apply my prototypical system in new situations. Some of the improvements that I am already planning to introduce in the system are:

1. After initially introducing the concepts in an informal way, stepping back to reintroduce them with more formal descriptions.

2. Stating the relationships between the concepts and the possible generalizations more explicitly, in order to clarify important, underlying commonalities between them.

3. Suggesting a larger number of exercises for the effective application of the concepts in real-world architectural design situations.

4. Developing strategies to deliberately generate interesting, architecturally relevant emergence that is not based on obvious compositional principles, and at the same time is not simply accidental.
VII. Conclusions

Summary

In this concluding Section, the main achievements of this thesis have been presented, taking into account the goals initially established. In summary, it is possible to say that:

- The prototypical instruction system initially proposed was developed and implemented at an experimental level, in two case studies.
- In both case studies, a change in students' attitude towards the use of CAD in architecture could be observed.
- In both case studies it was also possible to observe at least an intention to apply the new concepts and techniques to external situations.
- In one of the case studies students have reported a conceptual influence in other subjects three months after the course.

Besides, some unexpected observations could also be made, such as:

- The lower the technical knowledge in CAD software and programming, the higher the students’ resistance to see CAD as a design partner;
- The lack of experience in programming was probably a constraint to the development of programs that dealt with more profound problems. Less experienced students limited their programs to simple parametric variations of shapes, while more technically skilled students were able to add more variables and procedures to their programs.
- The proposed system proved to be also beneficial to civil engineering students.

At the same time, unanswered research questions have been acknowledged and weak points of this work were identified. Those included:

- The difficulties in handling data and assessing results.
- The lack of appropriate applications to illustrate some of the chosen concepts.

Finally, strategies for communicating the results and giving continuation to the research, as well as improvements to the proposed system in future applications have been suggested.
VIII. Bibliography


VIII. Bibliography


VIII. Bibliography


Rogers, E. N. (1979) De "Esperienza dell'architettura." In E.


VIII. Bibliography


### IX.Appendix A: Students' names and background

**Note:** Students whose full names, not just initials appear in the lists below have agreed in having their real names cited.

#### IX.a. UNICAMP

Workshop observer: Prof. Doris Kowaltowski (Architect)

On the table below, the information about students' previous experience with CAD and programming was not provided by them, but inferred from the contents of previous subjects they had coursed or classroom comments.

<table>
<thead>
<tr>
<th>Name</th>
<th>Background</th>
<th>AutoCAD experience</th>
<th>Programming experience</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. C. A.</td>
<td>Architect, graduate student</td>
<td>N/A</td>
<td>No</td>
</tr>
<tr>
<td>Amanda P. Petter</td>
<td>Civil engineering student, 4th year</td>
<td>2D</td>
<td>Pascal</td>
</tr>
<tr>
<td>Ana Lúcia Harris</td>
<td>Faculty, architecture school</td>
<td>3D</td>
<td>Yes</td>
</tr>
<tr>
<td>Ana M. Monteiro</td>
<td>Faculty, architecture school</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>C. M. S.</td>
<td>Architecture student, 4th year</td>
<td>3D</td>
<td>No</td>
</tr>
<tr>
<td>Christian T. Dittz</td>
<td>Architect, graduate student</td>
<td>N/A</td>
<td>No</td>
</tr>
<tr>
<td>Daniel da Rocha</td>
<td>Architecture student, 4th year</td>
<td>3D</td>
<td>Yes</td>
</tr>
<tr>
<td>Daniel Moreira</td>
<td>Architect, Ms.C. in multi media</td>
<td>3D</td>
<td>Yes</td>
</tr>
<tr>
<td>Fernando M. Basilio</td>
<td>Civil engineering student, 5th year</td>
<td>2D</td>
<td>Pascal</td>
</tr>
<tr>
<td>Fernando S. Ribeiro</td>
<td>Civil engineering student, 4th year</td>
<td>2D</td>
<td>Pascal</td>
</tr>
<tr>
<td>Gabrielle Damaso</td>
<td>Civil engineering student, 4th year</td>
<td>2D</td>
<td>Pascal</td>
</tr>
<tr>
<td>Giovana Bianchi</td>
<td>Civil engineering student, 4th year</td>
<td>2D</td>
<td>Pascal</td>
</tr>
<tr>
<td>Lauro Luiz Fco Fo</td>
<td>Faculty, architecture school</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>Livia C. Berriel</td>
<td>Architecture student, 4th year</td>
<td>3D</td>
<td>No</td>
</tr>
<tr>
<td>M. C.</td>
<td>Civil engineering student, 4th year</td>
<td>2D</td>
<td>Pascal</td>
</tr>
<tr>
<td>Marina Otaviano</td>
<td>Architecture student, 4th year</td>
<td>3D</td>
<td>No</td>
</tr>
<tr>
<td>N. B.</td>
<td>Faculty, architecture school</td>
<td>N/A</td>
<td>No</td>
</tr>
<tr>
<td>Patrícia Dias Falcão</td>
<td>Civil engineering student, 4th year</td>
<td>2D</td>
<td>Pascal</td>
</tr>
<tr>
<td>Paula R. Baratella</td>
<td>Architecture student, 4th year</td>
<td>3D</td>
<td>No</td>
</tr>
<tr>
<td>R. C. B.</td>
<td>Architect</td>
<td>N/A</td>
<td>No</td>
</tr>
<tr>
<td>Roberto Itapura</td>
<td>Architect</td>
<td>N/A</td>
<td>No</td>
</tr>
<tr>
<td>Sérgio Montagner</td>
<td>Civil engineering student, 5th year</td>
<td>2D</td>
<td>Pascal</td>
</tr>
<tr>
<td>V. W.</td>
<td>Architect, graduate student</td>
<td>3D</td>
<td>Auto Lisp</td>
</tr>
</tbody>
</table>
## IX. Appendix A

### IX.b. FAU-USP

Course observer: Prof. Marcelo Giacaglia (Civil Engineer)

<table>
<thead>
<tr>
<th>Name</th>
<th>Background</th>
<th>AutoCAD experience</th>
<th>Programming experience</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ana C. Salomao</td>
<td>Architecture student, last year</td>
<td>3D</td>
<td>No</td>
</tr>
<tr>
<td>Chen Chih Cheng</td>
<td>Architecture student, last year</td>
<td>3D</td>
<td>Yes</td>
</tr>
<tr>
<td>Daniela M. Costa</td>
<td>Architecture student, last year</td>
<td>2D</td>
<td>No</td>
</tr>
<tr>
<td>Daniela Vaz</td>
<td>Architecture student, last year</td>
<td>2D</td>
<td>No</td>
</tr>
<tr>
<td>Fabio A. Bellini</td>
<td>Architecture student, last year</td>
<td>3D</td>
<td>Yes</td>
</tr>
<tr>
<td>Fernando Mello V. F. H. T.</td>
<td>Architecture student, last year</td>
<td>2D</td>
<td>Yes</td>
</tr>
<tr>
<td>Juliana M. Beividas</td>
<td>Architecture student, last year</td>
<td>2D</td>
<td>No</td>
</tr>
<tr>
<td>Laila Milek A.</td>
<td>Architecture student, last year</td>
<td>2D</td>
<td>No</td>
</tr>
<tr>
<td>Leandro M. Robles</td>
<td>Architecture student, last year</td>
<td>3D</td>
<td>No</td>
</tr>
<tr>
<td>Marcelo Nakazaki</td>
<td>Architecture student, last year</td>
<td>2D</td>
<td>No</td>
</tr>
<tr>
<td>Marcos D. Machado</td>
<td>Architecture student, last year</td>
<td>2D</td>
<td>No</td>
</tr>
<tr>
<td>Min Kyung Lee</td>
<td>Architecture student, last year</td>
<td>2D</td>
<td>No</td>
</tr>
<tr>
<td>Renata Figueiredo</td>
<td>Architecture student, last year</td>
<td>2D</td>
<td>No</td>
</tr>
<tr>
<td>Rodrigo Azevedo</td>
<td>Architecture student, last year</td>
<td>3D</td>
<td>No</td>
</tr>
<tr>
<td>S. C. T.</td>
<td>Architecture student, last year</td>
<td>2D</td>
<td>No</td>
</tr>
<tr>
<td>S. T. A.</td>
<td>Architecture student, last year</td>
<td>2D</td>
<td>No</td>
</tr>
<tr>
<td>Stella L. Tomiyoshi</td>
<td>Architecture student, last year</td>
<td>2D</td>
<td>No</td>
</tr>
<tr>
<td>Thiago B. R. Lessa</td>
<td>Architecture student, last year</td>
<td>2D</td>
<td>No</td>
</tr>
</tbody>
</table>
X. Appendix B: Description of exercises

X.a. Pilot work

Ex 1.1: Developing a design environment with wall-paper symmetry.
Ex 1.2: Developing a design environment with wall-paper symmetry including rotation.
Ex 1.3: Developing a design environment with bilateral symmetry in two axes.
Ex 1.4: Developing a personal symmetric design environment.
Ex 1.5: Designing with circular symmetry in a pre-set environment (Symmetry Assistant).

X.b. UNICAMP

Ex 1.1: Developing a design environment with wall-paper symmetry.
Ex 1.2: Developing a design environment with wall-paper symmetry including rotation.
Ex 1.3: Developing a design environment with bilateral symmetry in two axes.
Ex 1.4: Developing a personal symmetric design environment.
Ex 1.5: Designing with circular symmetry in a pre-set environment (Symmetry Assistant).
Ex 2.1: Designing with recursion and nested details with AutoCAD blocks.
Ex 2.2: Generating the Koch curve with Recursion Assistant.
Ex 2.3: Designing with Recursion Assistant.
Ex 3.1: Instantiating AutoCAD blocks at different X and Y scale factors, trying to represent Niemeyer's Mondadori building.
Ex 3.2: Instantiating parameterized building profiles with a ready-made application.
Ex 3.3: Modifying the ready-made application.
Ex 4.1: Using a ready-made application for generating multiple design alternatives.
Ex 4.2: Modifying the ready-made application.
Ex 5.1: Using a ready-made application for generating bathroom layouts.
Ex 5.2: Modifying the ready-made application, adding more variables to it.
Ex 6.1: Finding novel shapes in a rule-based composition with AutoCAD's BHATCH command.
Ex7-2D: Creating 2D rule-based designs.
Ex7-3D: Creating 3D rule-based designs.

**Final project:**

Practical (applications or designs using the concepts learned) or theoretical (analysis of a building design); examples: parameterizing an architect's work, developing of a design algorithm (projects may be implemented in VBA or may be developed conceptually).

**X.c. FAU-USP**

S1: Bilateral composition in a viewport environment.
S2: Frieze composition in a viewport environment.

S3: Wall-paper composition in a viewport environment.
S4: Circular composition in a viewport environment.
S5: Research of architectural examples, and symmetric design of a horizontal or vertical residential complex.
R1a: Concentric composition in a viewport environment.
R1b: Spiraling composition in a viewport environment.
R1c: Recursive linear composition in a viewport environment.
R2: Koch curve with Recursion Assistant.
R3: Free composition with Recursion Assistant.
R4: Research of architectural examples, and recursive design of an institutional building.
P1: Parametric arches with AutoCAD blocks.
P2: Parametric arches with ready-made VBA application.
P3: First program: parametric line or profile.
P4: Research of architectural examples, and recursive design of a cultural building.
G1: Adding a slider bar to the first program.
G2: Adding a button that automatically generates alternatives to the first program.
G3: Describe situations in which it is necessary to enumerate design alternatives.
A1: Adding variables to bathroom layout ready-made application
A2: Developing conceptual program for a design process.
E1: 2D emergent design from a rule-based composition generated in Rules Assistant 2D and picked up by filling up closed areas.
E2: 3D emergent design from a rule-based composition generated in Rules Assistant 3D and edited with Boolean solid operations.
XI. Appendix C: Excerpts from classroom journals

XI.a. UNICAMP

Friday 1/02

Because I had the impression that not all students had completely understood the program the day before, I started this class with a new exercise, in which I conducted students step-by-step in the creation of a simple user form (Figure 102). We then proceeded to add a simple code to this form. Contrary to what I had thought, explaining ready-made programs is not enough; students need to build their own applications from scratch to understand the basic principles of programming.

Tuesday 5/02

Today we started with the text presentations, which allowed for interesting discussions. The "Hello World"-style looping-through-code exercise helped students understand yesterday's program. I started by showing a simple technique for creating image files that can be used as background or as image panels in the user form: the image of a drawing object displayed in AutoCAD was saved in .BMP format with the use of the Tools/Display Image/Save command. The image was then retouched and cropped with MS Paint. Most students were not familiar to Corel Draw and Photoshop, but that simple application was enough to allow them creating more sophisticated interfaces (Figure 103). Although not the focus of this course, designing user form-based interfaces was something that students were in general very interested in. It seems to be a good idea to start a programming class for architects with this approach, although it is always dangerous to deviate the focus of the class.

V. W. showed me a program she developed to model F. L. Wright's grandma's house columns. I asked her to present her program to the class, as I did with Daniel, in order to encourage other students to create their own programs. She parameterized the basic block's width, length and height, and I suggested that she could parameterize also the number of rows and the scale increment of each row in relation to the previous.

Thursday 07/02

I asked the students if the impact of using 2d and 3d Rules Assistant was different now from when they had used recursion assistant last
week, before having any idea of how to create an application in VBA. They answered that now they were able to figure out what was going on behind the screen. Some of them even opened the code to see how the program worked.

XI.b. FAU-USP

Monday 04/22

When I read Mitchell's quote "...wider possibilities were largely ignored", a student immediately asked me "Which possibilities? We have always been told in this school that computers are only good for representation, so are you going to show us which possibilities are those?" I replied, saying that that was exactly the purpose of the experimental course.

(...)

Students started doing the exercises on their own. Apparently, the bigger difficulty was with the DVIEW command, which we used to rotate the views inside the viewports. Most students had never used this command before. Explaining that the command was intended to be used for setting conic perspective views seemed to confuse them even more.

(...)

I offered to stay in the afternoon for helping students with their homework, but only 2 people came, not for asking technical question, but rather to ask about the purpose of the exercises. Apparently, it was not clear to them why we were using those commands in a new, different way.

Monday 04/29

Many students had difficulty understanding the overlapping of viewports in the circular symmetry exercises. Therefore, for the second session's exercises on recursive design with viewports I have created pre-set files with the viewports already configured, so students could download them and start drawing right away, without having to spend too much time. Yet, some students did want to follow the on-line instructions to create their own environments, so they could understand better what was going on.
When I asked students to give me examples of recursion in architecture, nobody risked answering. But when I showed them examples of recursive compositions, comparing them to examples from nature, they seemed very convinced and interested.

After the lecture, Chen told me how he had learned Basic and Pascal programming in the past, but he had completely dropped it because he saw no use for it in architecture. He said it seemed like he would need to learn a lot of programming in order to be able to do anything useful in architecture.

Monday 05/13

I started a discussion about the division of work between human designers and machines, based on Mitchell's text, The theoretical foundation of computer aided design. Looking at my table summarizing Mitchell’s points, a student expressed his concern with the blank cells in the "Human functions" column, in Table 11: Division of tasks between human designers and machines, after Mitchell (1975). His attitude reminded me Negroponte's (1970) affirmation that most people fear giving intelligence to a machine. I explained that computers do not produce programs on their own, these are always made by human programmers. If an architect programs a computer to create designs according to his criteria, he is indirectly creating designs himself, through the generalization of his own design concepts, which can be then instantiated with different parameters according to each particular context requirements and constraints.

After class, another student told me Mitchell's text had made him "very nervous", and asked me its relevance in the history of CAD. I answered the text was important, but of course it had been written many years ago, and even Mitchell himself might have changed his mind since then. Nevertheless, I explained how the CAD package we were using (AutoCAD), like other off the shelf programs, did not incorporate most of the ideas that were present in the text, and
the public never asked for that. Therefore, it is important to know the original purposes of CAD software development in order to understand some of the possible ways in which CAD can help us today.

Monday 05/20

Students felt nervous with the need to develop a program. Some mentioned things like "I can't do this" or "this is too hard for me". Their attitude contrasted with the answers to the pre-treatment test, when most said they believed programming was important to them. Some students did not even come back after recess.
XII. Appendix D: Assessment questionnaires

XII.a. Pilot work

Pre-assessment questionnaire

1. Number:
2. Age:
3. Course:
4. Year:
5. Are you experienced in any programming language?
   a. Yes.
   b. No.
   c. Which one?
6. What is CAD for you?
7. What do you know about the history of CAD?
   a. Nothing. I was never interested in the history of CAD.
   b. Nothing. I have never found any material about the history of CAD.
   c. I know the history of CAD since its origins.
8. In case you know the history of CAD, cite some important events in it (such as when and where it was originally developed, when it became popular, etc.)
9. Do you think it is important to learn CAD in an architecture school? Justify.
10. Do you think CAD could be more used by other subjects? Which ones?
11. CAD can help you …
   a. … being more efficient.
   b. … being more creative.
   c. … finding novel shapes.
12. What do you think about the generation of shapes by the computer?
   a. I've never heard about that.
   b. I don't think it can be considered an artistic use.
   c. I think it is interesting, but I would not use it.
   d. I think it is interesting, but it must require advanced programming techniques.
   e. I think it is interesting and I would like to try it, even if I needed to learn computer programming.

Post-assessment questionnaire

1. The lecture was…
   a. Clear.
b. Confusing.
c. Well-illustrated.
d. I'd like to see more illustrations.
e. Easy to follow.
f. Hard to follow.
g. Comments:
2. The contents about the history of CAD presented…
a. … have changed my opinion about the meaning of the word CAD.
b. … have changed my opinion about the use of CAD in architecture.
c. … (I already knew what was presented).
3. Do you think CAD could be used in other subjects? Which ones?
4. Would you be interested in a course about the use of CAD in the generation of architectural shapes? Why?
5. In the case you have answered yes to the previous question, would you be interested in learning simple programming techniques with the purpose of generating architectural shapes?
a. Yes.
b. No.
6. … and what about with the purpose of making drafting easier?
a. Yes.
b. No.
7. Do you think the exercises developed in class today …
a. … have nothing to do with the contents introduced.
b. … do not teach AutoCAD commands properly.
c. … allow learning AutoCAD commands in a better way.
d. … allow seeing more possibilities in the applications of CAD in architecture.
e. … made the contents introduced easier to understand.
8. Did you have any difficulty doing the class exercises? Justify.

XII.b. UNICAMP

Questionnaire 0a - Before the introductory lecture

1. Age: ___ Course: __________ Year: _____
2. Which subjects related to computers have you taken so far?
3. English language knowledge:
a. No.
b. Fluent.
c. Just read.
d. Other.
4. What do you expect to learn in this course?
5. What is CAD for you?
6. Do you have experience with any CAD program? Which one(s)?
7. What is your level of experience with AutoCAD?
   a. 2D.
   b. 2D and 3D.
   c. Too little.
   d. Nothing.
8. Are you experienced with any programming language? Which one(s)?
9. What is your knowledge about CAD history?
   a. I don't know anything because I was never interested in it.
   b. I don't know anything because I could never find material on it.
   c. I know the history of CAD developments.
10. In case you know the history of CAD, cite some important facts in its development (when it was initially created, when it became popular, etc.)
11. Do you think it is important to learn CAD in an architecture school?
    a. No. It's a waste of time, I can learn it on my own.
    b. No. I don't plan on using CAD to develop my design projects. In the future, I'll have drafters who will do that for me.
    c. Yes. CAD is a pre-requisite for internships.
    d. Yes. CAD can help me be more productive and efficient.
    e. Yes. CAD can make my designs better.
    f. Other.
12. Do you think CAD could be more used in other subjects in your course? Which ones and how?
13. What do you think about the generation of shapes by the computer?
    a. I've never heard about that.
    b. I think it's interesting, but it must require advanced programming skills.
    c. I think it's interesting, and I'd like to try it, even though I need to learn some programming for that.
    d. Other.

Questionnaire 0b - After the introductory lecture

1. The lecture was…
   a. Clear.
   b. Well illustrated
2. Comments:
3. The contents about the history of CAD...
   a. Have changed my opinion about the history of CAD.
   b. Have changed my opinion about the meaning of the word "CAD".
   c. Have changed my opinion about the role of CAD in architecture.
   d. Haven't added much to what I already knew.
4. What is your major interest in what concerns CAD?
   a. Drafting (representation).
   b. Realistic visualization and virtual reality (representation).
   c. Shape generation (synthesis).
   d. Performance evaluation (analysis).
   e. Other.
5. What kind of final project would you like to develop for this subject?
   a. A research paper about the use of computation in architecture.
   b. A program for automating drafting tasks.
   c. A program for generating shapes.
   d. A program for qualitative analysis of designs.
   e. Other.

**Questionnaire 1 - Symmetry**

1. Have you ever noticed the existence of different types of symmetry in nature, the decorative arts and architecture? Give some examples.
2. For you, being creative is necessarily breaking up symmetry or you think it is possible to be creative with symmetry?
3. Do you think the different types of symmetry shown in this lecture can help you in the creative exploration of architectural shapes? How?

**Questionnaire 2 - Recursion**

1. Have you ever observed the presence of recursion in nature, the decorative arts and architecture? Give some examples.
2. For you, which are the advantages of using recursion in archi-
3. Do you believe the techniques shown in this lecture can help you in the creative exploration of architectural shapes? How?

**Questionnaire 3 - Parametric shapes**

1. Have you ever observed the presence of parametric variations of shapes in nature, the decorative arts and architecture? Give some examples.
2. Do you think the parametric study of buildings by Reidy, Artigas and Niemeyer can lead to a better understanding of the formal exploration in the work of these architects?
3. Do you think the techniques taught in this session can help you in the creative exploration of architectural shapes? How?

**Questionnaire 4 - Generation**

1. In your opinion, which of the methods in the block composition exercise (direct modeling in AutoCAD of a number of alternatives, sliding bars to change blocks' dimensions in real time, automatic generation all the possible alternatives, automatic generation all the possible alternatives within the desired criteria) would be the most useful in architectural design? Justify.

**Questionnaire 5 - Algorithmization**

1. How did you like the lecture on the algorithmization of design processes? Choose one or more:
   a. It did not add anything to what I already knew.
   b. I had already thought about it in an informal way. The lecture helped me formalize my thoughts.
   c. I had never thought about designing in that way. The lecture changed my way of thinking about design, especially in what concerns repetitive tasks.
   d. I think that kind of approach is extremely dangerous to the architect's education, reducing his/her creativity.
   e. Other.
2. What is your opinion about the design algorithmization exercise?
   a. Interesting, but not useful for the architect's profession. He/she can hire a programmer to develop this type of applications.
   b. Interesting, and useful for the architect's profession. He/she can develop his/her own small applications as needed.
c. It's an useless approach to architectural design. An architect would never think in that way.
d. That kind of approach is useful only in specific cases, so the learning of the programming technique cannot be justified.
e. It was interesting to learn the programming technique, because in the future I can ask a programmer to develop similar applications and I'll know exactly how they work and what is their reach.
f. Other.

3. Comments:

**Questionnaire 6 - Course evaluation**

1. In your opinion, what is the reason why architects use computation when designing, and what are the most common uses of computation in architecture?
2. Do you believe from now on you will use computation in your curricular design projects? In case your answer is "yes", will you do it as a theme for your compositions or to automate repetitive tasks?
3. What is your final evaluation of this course?
   a. It has reached my expectations.
   b. It hasn't reached my expectations.
   c. It was superior to my expectations.
4. What have you liked better (+) and less (-) in this course?
   a. Lectures.
   b. Exercises.
   c. Final project.
   d. Other.

**Post-treatment questionnaire (three months after the workshop)**

1. Do you think the workshop had changed your attitude towards the role of CAD in architecture/civil engineering?
2. Do you think the workshop has somehow changed the way you see architecture/civil engineering?
3. Have you had any opportunity to apply what you have learned in the workshop, since then? (if yes, explain if it was simply a conceptual influence or if you have developed programs and applied the concepts learned to design, and where it happened.)
4. Would you like to give continuity to what you learned in the workshop, taking part in other similar courses?
XII.c. FAU-USP

Questionnaire 1 - Introductory lecture and symmetry

1. Your name:
2. Architecture school year:
3. Level of fluency in English:
   a. I can't read
   b. I can read with some difficulty
   c. I can read without difficulty
   d. I can read, write and speak
4. Have you taken any other CAD course before? Where?
5. Which CAD packages have you already used?
   a. AutoCAD
   b. MiniCAD/VectorWorks
   c. ArquiCAD
   d. Microstation
   e. Other
6. Outside the school, you have access to...
   a. Personal computer
   b. AutoCAD
   c. Internet
7. With AutoCAD, you are experienced in...
   a. 2D drafting
   b. 3D modeling
   c. Both
8. Do you think it is important to learn CAD in an architecture school? Why?
9. Which of the following curricular subjects could make use of computer aided design?
   a. History of architecture
   b. Architectural design
   c. Building performance
   d. Structural systems
   e. Graphic design
   f. Industrial design
   g. Urban planning
10. How important is computer programming in the architect's profession?
    a. Very important
    b. Relatively important
    c. Not important
    d. Harmful
11. What is your opinion about computer programming?
   a. In what concerns level of difficulty:
      i. Too hard for me
      ii. Just hard
      iii. I can learn
      iv. Easy
   b. In what concerns your interest:
      i. I'm very interested
      ii. I'm interested
      iii. I have no interest

12. Have you ever learned any programming language? Which one(s)?

13. Did you have any knowledge about the history of CAD?
   a. Yes
   b. No

14. Do you think computers can help architects beyond realistic visualization and drafting? How?

15. What do you expect to learn in this course?

16. How would you evaluate the lecture on symmetry?
   a. Excellent
   b. Good
   c. Regular
   d. Bad
   e. Very bad

17. What is your level of interest in this lecture's topic?
   a. I'm very interested
   b. I'm somehow interested
   c. I'm indifferent
   d. I'm not interested

**Questionnaires 2 - 5**

1. Your name:

2. How would you evaluate the lecture on symmetry?
   a. Excellent
   b. Good
   c. Regular
   d. Bad
   e. Very bad

3. What is your level of interest in this lecture's topic?
   a. I'm very interested
   b. I'm somehow interested
   c. I'm indifferent
d. I'm not interested

4. Comments:

**Questionnaire 6 - Emergence and course evaluation**

**Emergence**

1. Your name:
2. In what concerns the topic of this Session…
   a. I already used this concept, informally
   b. I had already read or wondered about emergence
   c. I had never reflected about that concept
3. What are the differences in what concerns the finding of emergent and accidental shapes when hand drafting and using a CAD program?

**Course Evaluation:**

1. Summarize what you have learned in this course.
2. Do you think this course has changed the opinion you had about the role of CAD in architecture? Why?
3. In your opinion, which are the reasons that make architects use computation in design, and which are the most common forms of the use of computation in architectural design?
4. Do you think, from now on, you will use somehow what you have learned in your design projects? How?
5. In your opinion, which were the most important aspects of this course?
   a. Teaching new AutoCAD commands
   b. Teaching VBA programming
   c. Showing new ways to use AutoCAD
   d. Showing new ways to use computers in architecture
   e. Demonstrating that computers can help in the creative process of design
   f. Showing a new way to see architecture
   g. Showing how to deal with aesthetical questions in a less subjective way.
6. Do you think you will be a better architect with what you have learned in this course?
   a. Yes
   b. No
   c. Don't know
7. In your opinion, which is the relevance of this course's contents
to the architectural education?
   a. Fundamental
   b. Very important
   c. Just interesting
   d. Not necessary
8. Do you think a subject with this contents should be...
   a. Mandatory in the first semester
   b. Mandatory after an introduction to CAD
   c. Mandatory in the last year
   d. Elective
   e. Should not exist
9. In what concerns the methods used in this course...
   a. There were too many things
   b. It was dynamic and interesting
   c. It was tiring, repetitive
   d. It was limiting
   e. Other
10. In comparison to traditional materials, the course materials in this course were...
    a. More efficient
    b. Less efficient
    c. There is no difference
11. In what concerns the readings suggested in this course, they were...
    a. Easy/interesting
    b. Hard/interesting
    c. Easy/uninteresting
    d. Hard/uninteresting
12. Choose as many as you find applicable:
    a. The lectures were clear and well-organized
    b. The time available to develop home assignments was enough
    c. The time available to develop classroom assignments was enough
    d. Students received the necessary feedback about their progress
    e. In the discussions, students' opinions were respected
    f. The assignments were clearly explained
    g. The different background of students was respected
    h. The course has contributed to my overall education as an architect
13. For you, which were the best aspects of the course?
14. For you, which were the worst aspects of the course?
15. Would you have any suggestion to improve this course?
XII.d. Observers

Post-treatment questionnaire

1. What were the reasons for your accepting of and taking responsibility on this experimental course at FAU-USP/UNICAMP?
2. How did you like the course? Did it meet your expectations? Which were its most interesting aspects?
3. Do you have any critique to the course? Would you suggest any modification or the inclusion other topics in the course?
The accompanying CR-ROM contains the two experimental courses' web sites, with most of the course materials and students' exercises. Copyrighted material, such as the course readings, have not been included, but the suggested bibliography is available in both web sites.

To view the web sites, simply double-click a file called index.htm inside each folder (UNICAMP and FAU-USP). The application required for both is Internet Explorer 4.0 or higher, on both PCs and Apples. The password required to unlock downloadable self-extracting files is respectively “unicampmit” and “faumit”.