In case of architectural design
Critique and praise of Case-Based Design in architecture

Jury:
Voorzitter Prof. Joos Vandewalle, vice-decaan
Prof. Herman Neuckermans, promotor
Prof. André Loeckx
Prof. Jan Schreurs
Prof. Johan Wagemans
Prof. Albert Dupagne (Université de Liège)
Prof. Rivka Oxman (Technion Haifa)

Proefschrift voorgedragen tot het behalen van het doctoraat in de toegepaste wetenschappen door Ann Heylighen

UDC 72.011 Mei 2000
For my grandmother
Acknowledgements

Five years ago, I had to decide on a subject for my graduate’s thesis. Professor Herman Neuckermans made the suggestion to have a look at Case-Based Design, at that time a ‘hot topic’ in the field of CAAD. Computers were not exactly what I would call my cup of tea, but I nevertheless accepted his suggestion – not in the last place because it would enable me to spend one semester at the ETH in Zürich. Although the thesis became an unqualified success in terms of scores, it left me with an uneasy feeling. Judging from the literature, the applications of Case-Based Design in architecture were not (yet) convincing. Yet, the underlying ideas seemed so plausible, that I could not dismiss it as the umpteenth flash in the CAAD pan. Immediately after my graduation, I received a grant from the Fund for Scientific Research (FWO) Flanders, which enabled me to pursue the matter with a doctoral research.

During the course of this research, I’ve had a lot of help and support. It would be impossible to mention everyone I met and exchanged opinions with, who gave me suggestions and encouragement. But among them, I feel a special obligation to thank, above all, Professor Neuckermans, who generously invited me into his CAAD-lab, thus giving me access to his personal expertise, his research budget (especially the part meant for travelling), his library and – last but not least – the daily help and friendship of his other CAAD-assistants. Among them, a very special thanks goes to Benjamin Geebelen, Bart Geeraerts and Kris Nuyts for the 1013 ways in which they have contributed to – and once in a while kept me from – my work. In 1997/1998, the CAAD-lab was temporarily reinforced by Raf Segers, the first thesis student I’ve supervised and, more importantly, the living dynamo behind DYNAMO.

The data analysis for the evaluation of DYNAMO and the experimental study on the effects of examples was performed in close collaboration with Dr. Ilse Verstijnen from the department of Psychology. Collaborating with the “dépendance in de Tiensestraat” – Professor Johan Wagemans, Ilse and Pim – was not just instructive, I really enjoyed it.

The collaboration with Jan Bouwen from the department of Applied Economics was less structural, and above all less self-evident, but therefore not less fruitful. I am indebted to him for his exchanged opinions, suggestions and encouragement in the course of my research, and for his detailed constructive comments on earlier drafts of this text.

In addition to Professor Wagemans, I would like to thank Professor André Loeckx and Professor Jan Schreurs for the stern and critical, but productive way in which they accomplished their task as members of the supervising committee. Thanks also to Professor Albert Dupagne (Université de Liège) and Professor Rivka Oxman (Technion, Haifa), who have accepted to be on the jury.

I am, of course, extremely grateful to the architects I’ve interviewed – Jan Delrue, Mauro Poponcini, Paul Van Aerschot, Werner Van dermeersch, Paul Vermeulen and
Peggy Winkels. They have given their precious time very generously and have made the writing of chapter 4 a fascinating and thoroughly rewarding experience.

In addition to the CAAD-colleagues, I was given considerable help (and support) by many other staff members and students of our department. In particular, I would like to thank all those who have contributed to the experiments. They are (amongst others): the students and the titular of the 2nd year Basic Design course; the studio teachers Evi Corne, (once again) Mauro Poponcini, Ann Verdonck and Hans Verplancke; the external judges Guido Geenen and Ivo Vanhamme; the students and the titular of the 4th year design studio; the studio teachers Hans De Petter, Leo Van Broeck and (once again) Ivo Vanhamme; and the external judges Heike Löhmann and Piet Stevens.

Karen Depoortere and Paul McHale were continuously helpful during my work by keeping an eye on the grammar and spelling of the text.

For encouraging me all the while, not only as I worked on this thesis, but also during my study, I am very grateful to my parents and my sister and, for his daily online support, to Toon. They saw me through the good times and the bad, and were always convinced that I would succeed, especially when I was not.

Thanks are due to many friends who have supported me in multiple ways. However, particular thanks go to Liesbet, Veerle and Gunther for their extra encouragement during the last months, and to the members of the working group ‘Woman and University’, for holding up themselves as stimulating example.

I dedicate this work to my grandmother, who had to miss me on many Sunday afternoons. My talents are hers, the opportunities I was given are the ones she should have got.
In case of architectural design

Critique and praise of Case-Based Design in architecture

Architects are said to learn design by experience. Learning design by experience is the essence of Case-Based Design (CBD), a sub-domain of Artificial Intelligence. Part I critically explores the CBD approach from an architectural point of view, tracing its origins in the Theory of Dynamic Memory and highlighting its potential for architectural design. Seven CBD systems are analysed, experienced architects and design teachers are interviewed, and an experiment is carried out to examine how cases affect the design performance of architecture students. The results of this exploration show that despite its sound view on how architects acquire (design) knowledge, CBD is limited in important respects: it reduces architectural design to problem solving, is difficult to implement and has to contend with prejudices among the target group. With a view to stretching these limits, part II covers the design, implementation and evaluation of DYNAMO (Dynamic Architectural Memory Online). This Web-based design tool tailors the CBD approach to the complexity of architectural design by effecting three transformations: extending the concern with design products towards design processes, turning static case bases into dynamic memories and upgrading users from passive case consumers to active case-based designers.

Case-Based Design in architectuur

Een kritische studie

Architecten, zo beweert men, leren ontwerpen uit ervaring. Leren ontwerpen uit ervaring is de essentie van Case-Based Design (CBD), een sub-domein van de Artificiële Intelligentie. In deel I wordt de CBD-benadering onderworpen aan een architectuurkritische studie. CBD is gebaseerd op de Theorie van het Dynamische Geheugen, en biedt het architectuurontwerpen heel wat voordelen. Zeven CBD systemen worden doorgelicht, ervaren architecten en ontwerpbegeleiders worden geïnterviewd, en de impact van concrete ontwerpervaring (cases) op de ontwerpprestatie van architectuurstudenten wordt experimenteel onderzocht. Hieruit blijkt dat CBD bijdraagt tot een beter inzicht in hoe architecten leren ontwerpen, maar beperkt is qua visie op het architectuurontwerpen, implementatie en gebruikerscontext. Deel II beschrijft het ontwerp, de implementatie en de evaluatie van een Dynamisch Architectuur-Geheugen On-line (DYNAMO), een ontwerpinstrument dat de CBD-aanpak poogt af te stemmen op de complexiteit van het architectuurontwerpen. Het verschuift de aandacht van ontwerpproducten naar ontwerpprocessen, vervangt statische case bases door een dynamisch geheugen, en maakt van de gebruikers actieve case-producenten in plaats van passieve case-consumenten.
CONTENTS

Introduction .......................................................................................................................... 1

Chapter 1: Conceiving architecture

1.1 The complexity of architectural design ................................................................. 9
1.2 Knowledge engineering in architectural design .................................................. 11
   1.2.1 Things to know ................................................................................................. 11
   1.2.2 Ways of knowing ............................................................................................ 12
1.3 Knowledge channels for conceiving architecture ................................................. 14
   1.3.1 Heuristics ....................................................................................................... 16
   1.3.2 Analogy .......................................................................................................... 17
      1.3.2.1 Iconic analogy ......................................................................................... 19
      1.3.2.2 Canonic analogy ...................................................................................... 21
   1.3.3 From analogy to metaphor (and back to analogy) ......................................... 22
   1.3.4 Type in a nutshell ......................................................................................... 25
   1.3.5 Types and instances ...................................................................................... 30
   1.3.6 A grid across a complex field ...................................................................... 34
1.4 Prior notices ............................................................................................................. 36

Part I. A critical exploration of Case-Based Design in architecture

Chapter 2: What is Case-Based Design based on?
From Artificial Intelligence over Dynamic Memory to Case-Based Design

2.1 Artificial Intelligence .............................................................................................. 42
   2.1.1 What? ............................................................................................................. 42
   2.1.2 Why? ............................................................................................................. 43
   2.1.3 How? ............................................................................................................ 43
2.2 Case-Based Reasoning: theory, model and application ....................................... 44
   2.2.1 The Theory of Dynamic Memory .................................................................. 45
      2.2.1.1 Scripts .................................................................................................... 45
      2.2.1.2 Reminding ............................................................................................ 46
      2.2.1.3 Reminding and problem solving ......................................................... 46
   2.2.2 The cognitive model behind CBR .................................................................. 47
      2.2.2.1 Structure and organisation of knowledge ........................................... 47
      2.2.2.2 Reasoning processes .......................................................................... 48
      2.2.2.3 Learning ............................................................................................... 50
      2.2.2.4 Generalised knowledge ..................................................................... 50
   2.2.3 Applications .................................................................................................. 51
      2.2.3.1 Constructing Case-Based Reasoners .................................................. 51
      2.2.3.2 Investigating human intelligence ......................................................... 52
2.3. Case-Based Design: premise and promises ...................................................... 53
   2.3.1 Premise ......................................................................................................... 54
   2.3.2 Promises of CBD for architecture .............................................................. 55
2.4 Summary ................................................................................................................ 58
# Contents

3.6 SEED ........................................................................................................... 89
  3.6.1 Case and other bases ................................................................. 89
  3.6.2 Retrieval ................................................................................. 90
    3.6.2.1 Characterising the target ............................................ 90
    3.6.2.2 Recalling and selecting cases ................................... 90
  3.6.3 Manipulation ........................................................................ 91
  3.6.4 Discussion ............................................................................ 91
  3.6.5 References .......................................................................... 92

3.7 WEBPAD .......................................................................................... 93
  3.7.1 Case base ............................................................................. 93
    3.7.1.1 Design case ................................................................. 93
    3.7.1.2 Case representation .................................................. 93
  3.7.2 Retrieval ............................................................................... 93
  3.7.3 Manipulation ........................................................................ 94
  3.7.4 Discussion ............................................................................ 94
  3.7.5 References .......................................................................... 94

3.8 Synthesis ........................................................................................... 95

Chapter 4: Are architects natural Case-Based designers?

Architects speaking

4.1 Procedure ......................................................................................... 100
4.2 Biographies .................................................................................. 102
4.3 Cases in conceptual design ............................................................ 103
  4.3.1 Whether .............................................................................. 103
    4.3.1.1 Cases from colleagues ............................................ 106
    4.3.1.2 Personal projects ...................................................... 108
  4.3.2 Which .................................................................................. 113
  4.3.3 Why ..................................................................................... 116
4.4 Cases outside conceptual design ....................................................... 118
4.5 Cases in design education ................................................................. 119
4.6 Summary ......................................................................................... 123

Chapter 5: Do cases make a difference?

5.1 Introduction .................................................................................... 126
5.2 Related work ................................................................................ 127
5.3 Experiment ...................................................................................... 129
  5.3.1 Assignment ........................................................................ 129
  5.3.2 Subjects .............................................................................. 130
  5.3.3 Procedure ........................................................................... 130
  5.3.4 Judgement ........................................................................... 132
  5.3.5 Dependent and independent variables .................................. 132
5.4 Analysis ........................................................................................... 132
  5.4.1 Hypotheses ......................................................................... 132
  5.4.2 Statistical analyses .................................................................. 133
5.5 Discussion ....................................................................................... 137
  5.5.1 Effects on design quality/concept ....................................... 137
  5.5.2 Effects on design fixation/creativity .................................... 139
  5.5.3 Effects of teacher experience .............................................. 140
5.6 Summary and conclusions ............................................................... 141
Chapter 6: Case-Based Design in architecture
A critical review

6.1 The cognitive model behind CBD
6.1.1 Design experience ................................................................. 144
6.1.2 Retrieving relevant cases ......................................................... 146
6.1.3 View on architectural design ...................................................... 147
6.2 Implementing CBD tools ............................................................... 149
6.2.1 Representing design cases ......................................................... 150
6.2.2 Integration ................................................................................. 151
6.2.3 Dynamics ................................................................................. 152
6.3 CBD in education ........................................................................... 153
6.4 Taking stock ................................................................................... 155

Chapter 7: DYNAMO – A Dynamic Architectural Memory On-line
The idea(l)

7.1 Introduction .................................................................................. 162
7.2 The cognitive model of CBD .......................................................... 163
7.2.1 Concrete cases .......................................................................... 163
7.2.2 Dynamics .................................................................................. 164
7.3 DYNAMO: an (inter-)active workhouse ............................................ 164
7.3.1 Between design cases ................................................................. 165
7.3.2 Between human designer and computer ........................................ 167
7.3.3 Between (student-)designers ...................................................... 169
7.3.4 Between practice and education ................................................. 170
7.3.5 Action ....................................................................................... 171
7.4 The larger context .......................................................................... 171
7.4.1 A conceptual framework for CAAD .............................................. 171
7.4.2 IDEA+ ...................................................................................... 173
7.5 An alternative to (current) CBD systems? ........................................ 175
7.5.1 Cognitive model ......................................................................... 175
7.5.1.1 Design experience ................................................................. 175
7.5.1.2 Retrieving relevant cases ......................................................... 177
7.5.1.3 View of architectural design .................................................. 177
7.5.2 Implementation ......................................................................... 177
7.5.2.1 Case representation ............................................................... 177
7.5.2.2 Integration ............................................................................. 178
7.5.2.3 Dynamics .............................................................................. 179
7.5.3 Use ......................................................................................... 179
7.6 Summary ...................................................................................... 179

Chapter 8: A working prototype
Implementation

8.1 Introduction ................................................................................ 182
8.1.1 The Web ............................................................................... 182
8.1.2 Dynamic Web documents ......................................................... 184
8.1.3 User input ................................................................................. 186
8.2 A working prototype ................................................................. 186
8.2.1 Design cases ........................................................................... 186
8.2.2 Memory organisation ............................................................... 187
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.2.2.1 ID</td>
<td>188</td>
</tr>
<tr>
<td>8.2.2.2 Optional indices</td>
<td>188</td>
</tr>
<tr>
<td>8.2.3 Interface</td>
<td>191</td>
</tr>
<tr>
<td>8.3 DYNAMO in action</td>
<td>194</td>
</tr>
<tr>
<td>8.3.1 Searching the case base</td>
<td>195</td>
</tr>
<tr>
<td>8.3.2 Feeding the case base</td>
<td>197</td>
</tr>
<tr>
<td>8.4 Status and similar systems</td>
<td>202</td>
</tr>
<tr>
<td><strong>Chapter 9: DYNAMO in the studio</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Evaluation</strong></td>
<td></td>
</tr>
<tr>
<td>9.1. Why reuse?</td>
<td>208</td>
</tr>
<tr>
<td>9.1.1 CBR-appropriateness</td>
<td>208</td>
</tr>
<tr>
<td>9.1.2 Relevance for architecture</td>
<td>209</td>
</tr>
<tr>
<td>9.2 Setting and procedure</td>
<td>210</td>
</tr>
<tr>
<td>9.2.1 Subjects</td>
<td>210</td>
</tr>
<tr>
<td>9.2.2 Task/assignment</td>
<td>210</td>
</tr>
<tr>
<td>9.2.3 Procedure</td>
<td>210</td>
</tr>
<tr>
<td>9.2.4 Judgement</td>
<td>211</td>
</tr>
<tr>
<td>9.3 Student engagement</td>
<td>212</td>
</tr>
<tr>
<td>9.3.1 Informal appreciation</td>
<td>212</td>
</tr>
<tr>
<td>9.3.2 Formal appreciation</td>
<td>213</td>
</tr>
<tr>
<td>9.4 Stimulating and/or hampering factors</td>
<td>217</td>
</tr>
<tr>
<td>9.5 Impact on design quality</td>
<td>217</td>
</tr>
<tr>
<td>9.5.1 Dependent and independent variables</td>
<td>218</td>
</tr>
<tr>
<td>9.5.2 Hypotheses</td>
<td>218</td>
</tr>
<tr>
<td>9.5.3 Statistical analyses</td>
<td>219</td>
</tr>
<tr>
<td>9.5.4 Discussion</td>
<td>220</td>
</tr>
<tr>
<td>9.6 Summary and conclusion</td>
<td>222</td>
</tr>
<tr>
<td><strong>Epilogue</strong></td>
<td></td>
</tr>
<tr>
<td>Followed directions</td>
<td>226</td>
</tr>
<tr>
<td>Future directions</td>
<td>229</td>
</tr>
<tr>
<td>CBD’s model of design cognition</td>
<td>229</td>
</tr>
<tr>
<td>From prototype to fully-fledged tool</td>
<td>230</td>
</tr>
<tr>
<td>Integration with IDEA+</td>
<td>230</td>
</tr>
<tr>
<td>Content search, thesaurus and feedback</td>
<td>230</td>
</tr>
<tr>
<td>What you see is what you get</td>
<td>231</td>
</tr>
<tr>
<td>What is DYNAMO good for?</td>
<td></td>
</tr>
<tr>
<td><strong>Appendix A: Architects’ questionnaire</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Appendix B: Design assignment reuse</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Appendix C: Students’ questionnaire</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Appendix D: Statistical results of DYNAMO’s evaluation</strong></td>
<td></td>
</tr>
<tr>
<td><strong>References</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Samenvatting</strong></td>
<td></td>
</tr>
<tr>
<td><strong>List of publications</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Curriculum vitae</strong></td>
<td></td>
</tr>
</tbody>
</table>
“Every project is a step forward, and each time there are risks involved, on technical, constructive, material or whatever level. Each time you learn something, and one time you’ve maybe gone a little too far, or not far enough, so that the next time you know: I must absolutely not do this ever again, or here I still can go much further, and I should go much further.”

Werner Van dermeersch (1999)
It is a commonplace to assert that architects learn design by experience. To learn by experience is the essence of Case-Based Reasoning (CBR), a branch of Artificial Intelligence (AI) engrafted onto a memory-based view of cognition. Relatively young as this branch may be, a surprising number of tools that flow from CBR research are already commercially successful, daily demonstrating the power of (re-)using cases in widely divergent tasks, domains and disciplines. A domain that does not seem to share in this success until now is architectural design. Unlike the early expectations of CBR pioneers, a convincing breakthrough of CBR in this field has yet to come. Despite serious efforts to develop Case-Based Design (CBD) systems – CBR applications tailored to the task of designing – these systems have rarely become widely used tools in design offices or schools of architecture. In search of reasons for this so far limited success, the present thesis embarks on a critical exploration of the role of cases in architectural design. The main objective is to make a diagnosis for CBD’s disappointing contribution to the field of architecture and, if desired, to propose a strategy for improvement.

Our exploration starts from the premise that if cases play such a prominent part in architectural design as CBD researchers assert, then successful and proficient CBD tools are likely to require a deeper understanding of this role. For example, what makes architects rely – consciously or unconsciously – upon a specific case during design, and what are the effects on the quality of the resulting design product? Can we identify differences between case reuse by brilliant and poor designers, or by expert and novice designers, c.q. design students? If so, can we help move students more rapidly along the path to expertise? If we are to develop CBD tools that are really tailored to architects’ needs during design, we should get beyond the ‘hand waving’ that characterises most current discussions of CBD in architecture. That is, we must know more about cases in architectural design than that they play an important part. We need a more differentiated understanding of the broad range of phenomena that are grouped under the umbrella term Case-Based Design.

Exploring the role of cases in architectural design raises a host of difficulties, starting with the meaning of the word case itself. To give a precise definition of case is as difficult as coming up with a definition of experience, which is inseparably linked to the notion of case in CBR research. The term case, as used by CBR researchers, stands for the interpreted representation of a real experience, including all details that make the experience individual [Kolodner, 1997]. At first sight, this definition does not bring us much further, since the term experience can be used in many different meanings. Yet, despite the wide variety of possible interpretations, most CBD systems unanimously interpret cases as concrete design projects.

Throughout the literature on architectural design, architects, design researchers and psychologists have given expression to this notion of cases, without necessarily making explicit mention of the term. For example, several authors use the term episodic knowledge to convey the idea of ‘particular, experience-linked sources which are at the same abstraction level as the target problem (‘cases’), rather than general knowledge structures at a more abstract level’ [Visser, 1996 p.273], a meaning that comes close to the definition of cases adopted in CBR research. As
INTRODUCTION

variations on this theme we can cite the terms episodic data [ibid.] and episodic memory [Oxman, 1990]. The former actually extends the notion of episodic knowledge and embraces the fact that, next to knowledge (i.e. data from an internal source), designers may use information on particular experiences coming from external sources. Events in episodic memory can be regarded as design precedents, which for their part come close to the notion of cases as concrete design projects. The term design precedent was introduced by Rivka Oxman to denote the representation of a memorable design project, including the particular conceptual contribution that gives the project the status of a recognised, outstanding example of design [Oxman, 1990; 1994]. Since then, precedents have found their way into design research as knowledge chunks about past designs that are re-usable in new, but similar design situations [Flemming & Aygen, 1999; Pasman & Hennesey, 1999]. Similarly, studies on analogy and fixation in design have come up with a number of terms that convey the notion of cases as particular design projects. The term within-domain sources is a case in point [Casakin, 1997], the everyday expression examples as well [Christiaans & van Andel, 1993].

Apart from this Babel-like jargon, a critical exploration of case (re-)use in architectural design also raises difficulties of methodology. In general, if we want to develop a better understanding of design, there are several possible approaches to choose from [Lawson, 1994 p.2]: we can skim the literature for interesting theories or models of design; we can analyse the design task, propose processes that are likely to take place and simulate these processes in a computer memory; we can ask architects how they do it, observe them at work, or conduct controlled experiments on them. Throughout four decades of design research, each of these methods has definitely won its spurs. Yet, when it comes to choosing between them, all seem unsatisfactory in some respect. So, what should we do to investigate the role of cases in architectural design? According to Bryan Lawson, “Quite simply, we should do all we can. There is a role for philosophising about design, we can learn from observation, we might get insight from experiments, and we should be foolish indeed not to try to ask designers how they do it” [ibid. p.3; italics added].

Taking Lawson’s advice rather literally, we have opted to examine the role of cases in architectural design in different ways and from different perspectives. To be precise, the thesis brings together voices from CBD and other design researchers, practising architects, design teachers and architecture students on the subject of cases in architectural design. As the chapters will reflect, CBD researchers bring precision in how case (re-)use can be implemented in or supported by computer tools. Practitioners and design teachers, for their part, furnish a fresh and pragmatic perspective on the subject, as well as resistant preconceptions. Architecture students then, by participating in experimental studies, provide us with a body of evidence on the effects of calling in cases – whether or not provided by a CBD tool – on the quality of design projects. The hope, of course, is that by examining cases from multiple perspectives, we may begin to converge on a (more) complete and diversified picture of their role in architectural design than any one method alone could provide.
A consequence of our choice for various approaches is the perhaps unusual structure of the thesis. For that reason, figure 0.1 plots a sort of map of the thesis, which might be useful to guide the reader through this structure.

The thesis starts with a chapter that introduces the reader into the specific and highly complex nature of architectural design. We argue that this nature accords with the definition of complex systems given by [Cilliers, 1998], and briefly discuss a selection of strategies that architects call in to cope with this complexity.

The remainder of the thesis is devoted to one specific strategy – the (re-)use of concrete design cases – and is divided into two parts. Part I, consisting of chapters 2 to 6, critically investigates the CBD approach within the context of architecture. In order to situate the roots of this approach, chapter 2 takes an excursion into the domain of AI. The idea is to zoom in step-by-step on CBD and to provide a background for its further exploration in the following three chapters. A useful way to begin this exploration is by putting a selection of specific CBD systems under the microscope. For this purpose, chapter 3 is conceived as a case base of CBD tools for architectural design. The systems described are especially concerned with the field of architecture and, taken together, give a fairly good overview of the various directions in CBD
research. Switching from a theoretic to a more pragmatic perspective, chapter 4 hands over the floor to six architects who have ample experience in both design practice and education. They are asked to describe how – and even whether – they (re-)use cases to conceive architecture. In order to get a better insight into the impact of this (re-)use, chapter 5 turns down yet another road. It describes an experimental study that explores the positive and/or negative effects of exposure to cases during the design process on the quality of the resulting design product. By way of check-up, chapter 6 tries to bring some of the key issues raised by these different approaches into sharper focus, by confronting the views of the different stakeholders. The main objective is to make a diagnosis for CBD’s so far limited contribution to the field of architecture.

Part II, confined to the next three chapters, takes up the challenge to help bridging the gap between CBD research and architectural design, based on the findings of the first part. The bridge is called DYNAMO – Dynamic Architectural Memory On-line – and takes the form of an interactive digital design tool built around concrete cases. Whereas chapter 7 outlines the main ideas behind DYNAMO, chapter 8 describes the implementation of these theoretical ideas as a working prototype. The evaluation of this prototype by student-architects is the subject of chapter 9. Finally, an epilogue looks back upon our exploration of cases in architectural design, attempts to formulate conclusions and maps out possible routes for future excursions.
CONCEIVING ARCHITECTURE

“When I have to solve an architectural problem ... the ... demands ... are so numerous that they form a maze which cannot be worked out by rational methods. The ensuing complexity prevents the basic architectural idea from taking shape. (...) I start drawing, giving free rein to my intuition, and suddenly the basic idea is born, a starting point which links the numerous, often contradicting elements already mentioned, and bring them into harmony with each other...”

Alvar Aalto (1957)
Within the vast domain of Artificial Intelligence (AI), Case-Based Reasoning (CBR) is to be situated in the field of knowledge engineering, a sub-field concerned with the question of what knowledge is required to fulfil intelligent tasks. Chapter 1 will try to answer this question in the case of a specific task, the task of conceiving architecture. Its main objective is to identify what knowledge is needed for architectural design, and in particular for concept generation.

Using knowledge as a vehicle to point out the specific nature of architectural design, might look like dragging the Trojan horse within architecture’s walls. For inside these walls, knowledge is often viewed with disdain, as a hindrance to unfettered creativity [Press, 1998] or an encapsulation of “freeze-dried prejudices” [Rittel, 1985]. This disdain, however, does not make architectural design an exception to the ‘rule’ that every discipline has its own realisation of knowledge [Scarborough & Burrell, 1996]. In an article significantly titled ‘Designerly ways of knowing’, Nigel Cross contends that design has its own distinct “things to know, ways of knowing them and ways of finding out about them,” but that we need to be more articulate about them [Cross, 1982]. That is exactly what this chapter is aiming at.

As sections 1.1 and 1.2 will point out, architectural design is a highly complex, knowledge-rich activity. Especially the early, conceptual stage of the design process makes great demands on the architect, as it involves multiple bodies of knowledge as well as various ways of knowing. Section 1.3 will shift attention to the strategies architects use to cope with this high complexity. Most of these strategies have been described by many authors before. The majority of these descriptions, however, analyse and criticise the end products that result from using them during design. By contrast, we will consider these strategies from the point of view of the process itself. In other words, we will look at them in a ‘modus operandi’ as opposed to a ‘modus operatum’ [Bourdieu, 1977], that is how their use, as it unfolds over time, is perceived by someone working on a design, instead of how it looks with the hindsight of being finished. More precisely, we will try to point out how these strategies may function as knowledge channels during concept generation.

The focus on process in architectural design is a relatively recent phenomenon. For many years and centuries, designing was learned implicitly by watching and working with a master. It is only recently that researchers slowly have become interested in ‘how architects think in action,’ i.e. in the architectural design process [Akin, 1979; Hamel, 1990; Lawson, 1980; Rowe, 1987]. According to Charles Eastman, there are many reasons why the next decade will – and should – be marked by a radical focus on process in design [Eastman, 1999]. Nevertheless, there still is a great reluctance to narrow the limits of the ‘black box’ and make the design process explicit. To some such explicitness would remove the mysticism and art of design by reducing it to a

---

1 See chapter 2, p.44.
3 Eastman offers four reasons why a better insight in the design process will be demanded: to better manage complexity; to improve collaboration among the specialists in design teams; to improve design education; and – the main reason for our interest in the design process – to improve tool support for design [Eastman, 1999 pp.III.228-229].
standard procedure or method. Our focus on the architect’s design process, however, was not inspired by an interest in method or systematisation. Instead, it derives from the idea that a better understanding of the design process may help to improve the tool support of this process. If we are to develop performant computer tools – whether case-based or not – for use by architects, we might do well to develop a profound understanding of how architects work.

1.1 The complexity of architectural design

According to Paul Cilliers, complex systems can be characterised as consisting of large numbers of components, which in themselves can be simple and which interact dynamically by exchanging energy or information [Cilliers, 1998]. Even if one component interacts with only a few others, the effects of these interactions are propagated throughout the entire system.

When using ‘systems’ in a broader sense so as to include activities, tasks and processes, in short phenomena, then architectural design obviously matches Cilliers' definition of a complex system. Indeed, architectural design – like industrial, interior and urban design – is a highly complex activity involving a great variety of ‘components’ or ‘dimensions’. Back in 25 B.C., this multidimensionality of architectural design was already identified by Vitruvius, according to whom "the ideal architect would be a man of letters, a skilful draftsman, a mathematician, familiar with historical studies, a diligent student of philosophy, acquainted with music, not ignorant of medicine, learned in the responses of juris consults, familiar with astronomy and astronomical calculations" [Vitruvius/Morgan, 1960]. Vitruvius described architecture as a dynamic balance between three dimensions: Firmitas, Utilitas and Venustas, the corners of his legendary triangle. Today, however, a polygon with many more angles would be needed to map out the different fields architects must be knowledgeable about. Acoustics, chromatics, ecology, economy, ergonomics, materials science, soil and other mechanics, project management, domains of physics, psychology and sociology are all somehow relevant to architectural design. Of course, many of these issues can be seen as mere subdivisions of Vitruvius' three categories. Yet, recent developments within the profession force architects into new roles, which require if not expertise then at least appreciation of genuinely new bodies of knowledge.

On top of being numerous and diverse, the issues an architect must take into account are often contradictory and always highly interwoven. Every decision he takes is likely to have implications that cut across multiple aspects. Enlarging a window, for example, may result in more light and a better view, but at the same time cause more heat loss and greater problems of privacy [Lawson, 1990]. Rather than the diversity of the isolated issues themselves, it is this very interconnectedness which makes architectural design such a highly complex activity.

The architect's greatest impact comes during the early stages of the design process, when he must come up with one or a few ideas powerful enough to encompass the different aspects. These underlying ideas are known to architects by many names,
CHAPTER 1

ranging from 'image' [Alexander, 1979] over 'primary generator' [Darke, 1978] to 'organising principle' [Rowe, 1987], but most often are called the 'parti'4 [Leupen, Grafe, Körnig, Lampe & De Zeeuw, 1997] or 'concept' [Lawson, 1994]. Such concept does not necessarily require the addition of an extra ingredient. In fact, every aspect already present in the design situation, e.g. a special feature of the site or programme, or a curious trait of the client, may qualify for this focal role. Moreover, underlying ideas are rarely found in the singular. In the Institut du Monde Arabe in Paris, for example, Jean Nouvel combined the need for sun shading with a ‘Moucharabieh’ pattern5 and the idea of a light-controlling diaphragm in a camera lens (figure 1.1) [Sharp, 1990]. This resulted in a giant Islamic pierced screen, which makes this modern high-tech building a permanent reference to traditional Islamic architecture.

![Figure 1.1. Façade of Jean Nouvel’s Institut du Monde Arabe, Paris (France), 1981-1987.](image)

Whether played by an outsider or an insider, in the singular or the plural, the value of concepts in architectural design derives from their potential to manage complexity by offering a framework to integrate all aspects into one coherent and meaningful design.6 Hence Eberhardt Rechtin contends, the architect is not a "general engineer",

---

4 The term ‘parti’ is said to be derived from the French expression ‘prendre parti,’ i.e. to make a choice, and refers to the main shape or overall form of the scheme [Leupen e.a., 1997].
5 In Islamic or Islamic-influenced architecture, Moucharabieh is a projecting second-story window of latticework, made up of small wooden bobbins composed in intricate geometric patterns. It is a familiar feature of urban residences in North Africa and the Middle East, where it provides the interior with light and air while shading it from the hot sun.
6 Apparently, this potential is not always considered an advantage. Christopher Alexander, for instance, completely dismisses the use of an ‘image’ as it would interfere with his patterns: “Architects sometime say that in order to design a building, you must have ‘an image’ to start with, so as to give coherence and order to the whole. But you can never create natural a thing in this state of mind. If you have an idea — and try to add patterns to it, the idea controls, distorts, makes artificial, the work which the patterns themselves are trying to do in your mind. Instead you must start with nothing in your mind” [Alexander, 1979].
but a specialist in managing complexity, uncertainty and ambiguity by means of workable concepts [Rechtin, 1991 p.13].

1.2 Knowledge engineering in architectural design

1.2.1 THINGS TO KNOW

From the point of view of knowledge engineering, the high degree of complexity makes architectural design an intrinsically knowledge-rich activity [Lawson, 1994]. The many different 'components', c.q. issues an architect must take into account, correspond to as many bodies of (specialised) knowledge. Each of these bodies matches the definition of component knowledge [Henderson, 1996], provided that ‘component’ is understood as a specific aspect of a design (e.g. cost or colour) rather than as a physical part (e.g. a wall or window). Rebecca Henderson defines component knowledge as knowledge about the basic ingredients and the way in which they are implemented in a particular component [ibid.]. In general, this knowledge can be characterised as local, active, and focused, as it enables one to deal with a specific aspect without needing to understand the internal functioning (or other aspects) of the whole. Architects use it to analyse whether their design fulfils a specific requirement, such as in cost calculations, structural safety analyses or energy studies.

The merging of all these bodies of component knowledge produces, as one might expect, technical, judgmental and professional conflicts [Rechtin, 1991]. The resolution of these conflicts by means of one or a few concepts requires a second type of knowledge, namely knowledge about the way in which components are integrated and linked together into a coherent whole [Henderson & Clark, 1990]. Within the design and technical innovation literature, this knowledge type is coined architectural knowledge, which might lead to confusion when transposed to the domain of architecture. It seems reasonable, therefore, to replace the term 'architectural knowledge' in the context of this thesis by the name 'concept knowledge'. By consequence, when knowledge engineers ask what knowledge is needed for conceiving architecture, the answer should include at least two types of knowledge: concept knowledge and several bodies of component knowledge.

Both types of knowledge appear in two different guises, as they can take on either an implicit or an explicit form. Michael Polanyi distinguishes implicit, c.q. tacit from explicit knowledge based on the degree to which knowledge can exist independently of a specific context or knower [Polanyi, 1964; 1967]. Since explicit knowledge easily

---

7 Note that this view of an architect as a manager chimes with the very origin of the term itself, which derives from the Greek ἁρματικος. The prefix ἁρματικος significantly means ‘the one who is in charge of,’ the substantive ἁρματει signifies ‘carpenter’ or ‘building master’.

8 Although this may appear a purely theoretical distinction, the recognition of component and concept as different levels of knowledge is a useful instrument for design research. It has proven to be an enriching pair of glasses to observe, analyse and understand concept generation in architectural design [Heylighen, Bouwen & Neuckermans, 1999].
migrates from one knower to another, it can be acquired through explicit instruction. Tacit knowledge, on the other hand, is learned implicitly, i.e. by exposure to a rich and complex stimulus environment without overt, conscious strategies to learn [Reber, 1989]. Concept knowledge, for instance, is not only taught explicitly in theoretical courses, but also acquired implicitly in the design studio. During discussions with design students, studio teachers give mini master demonstrations on how to develop a meaningful concept. In this way, concept knowledge embedded in experienced designers (i.e. studio teachers) is implicitly passed on to student designers [Heylighen e.a., 1999].

1.2.2 WAYS OF KNOWING

If architectural design is a knowledge-rich activity, however, this richness does not only lie in the different types of knowledge required, but also in the different ways of knowing (them). During design, at least two ways of knowing are dealt with. The first mode is passive and scientific in that it considers knowledge as a matter of the observed object alone, abstract and academic [Schön, 1983]. Most of the time, the knower is a rather distant observer – a role that might fit scientists as a glove, yet feels uncomfortably tight to designers. One way in which design differs from science is namely that “the solution is not simply lying there among the data, like the dog among the spots in the well-known perceptual puzzle; it has to be actively constructed by the designer's own efforts” [Cross, 1982].

9 To say it with Herbert Simon, “The natural sciences are concerned with how things are. Design on the other hand is concerned with how things ought to be,” and one might well suppose the introduction of the verb 'ought' to involve a different way of knowing [Simon, 1969]. Knowing components and concepts on a passive, technical-rational level enables architects to analyse and evaluate existing designs. Yet, it is not constructive, that is, it does not point the way to the creation of a new design. Rather than yet another piece of knowledge to be known, such creation requires another way of knowing, one that is inherent to the very design process itself and thus presupposes involvement. This second, constructive mode of knowing is both embedded in and developed through the action of designing, thus being in fact continuously 'under construction'. Instead of a passive observer, it forces the knower into the role of an active, integrated actor [Schön, 1983].

9 To be precise, the difference referred to here is not that between design and science, but between, on the one hand, design and, on the other hand, the stereotypical view of scientific research as arriving at scientific theories through inductive reasoning. In the early 1960s, both Thomas Kuhn's *The structure of scientific revolutions* [Kuhn, 1962] and Karl Popper’s *Conjectures and refutations* [Popper, 1963] did away with this stereotype. Rather than straightforward induction, they pointed out, revolutionary scientific discoveries often involve intuitive ‘paradigm shifts’. Darwin’s Evolution Theory or Einstein’s Relativity Theory, for instance, resulted from the positing of a tentative explanation, followed by an exploration of its implications. These new insights into the nature of scientific activity showed up science as strongly related to design and, vice versa, revalued design as scientifically respectable [Archer, 1999]. In fact, as will be pointed out further in this text, many designers experience their activity as a form of research (see p.13 and chapter 4, pp.110 ff).
This active development of knowledge/knowing in a particular design context may explain why many architects experience their activity as a form of research. Santiago Calatrava, for example, believes that technical innovation is more likely to result from designing a specific building than from solving abstract problems [Lawson, 1994 p.30]. Yet, as mentioned above, technique is not the only issue of concern for an architect. Because of the multidimensionality of architecture, the development of knowledge through design occurs at several fronts at the same time. An architect does not have to conceive whatever design, but one that coherently integrates the “20 major issues a designer of buildings has to consider” [Richard Burton in Lawson, 1994 p.14]. Considerable psychological evidence has shown, however, that it is quite impossible to keep all these issues simultaneously in one’s mind [Wagemans, 1996]. When the number of issues exceeds ‘the magical number seven, plus or minus two’, they are simply not sustainable by short-term memory [Miller, 1956; referred to in Neuckermans, 1994 and in Wagemans, 1996]. Consequently, the architect cannot but switch attention permanently from one aspect to another. In this respect, Richard MacCormac and Michael Wilford compare designing to juggling: “a juggler who’s got 6 balls ... and an architect is similarly operating on at least 6 fronts simultaneously and if you take your eye off one of them and drop it, you're in trouble. There is a sequential development but it is on several fronts simultaneously” [Lawson, 1994 p.114].

These fronts, however, are not only to be situated at component level, but also at concept level. The term ‘concept generation’ for the early stage of the design process misleadingly suggests that the concept comes into being at once and from then on can be treated as a static, invariant feature of the design project. In reality, however, there is a continuous two-way interaction between the design and the concept behind it, each informing the other: the concept allows the architect to impose an order on the design while the design enables him to explore and develop the concept. Schön therefore describes concept generation as an experiment of which the results are only dimly apparent in the early stages of the project [Schön, 1985]. Indeed, often very little of the concept is understood until late in the design process [Lawson, 1994].

What makes the architect’s ‘juggling’ from concept to component and between different components so wicked, is that not all components are known in advance. Rather, they gradually pop up during the design process. Peculiar to design is namely that the definition of a design problem grows parallel to its solution [Archer, 1973; referred to in Neuckermans, 1994]. Many aspects of the problem are unknown before design starts and are only discovered while designing. As Gianfranco Carrara and Yehuda Kalay contend: “The search for a solution produces insights into the design problem, that are not apparent before the search begins, revealing compromises that

---

10 Even after the process, the idea behind a design project cannot be treated as a neutral and invariant feature. As will be pointed out in the discussion on metaphors, the concept the designer started from is not always the one critics or other architects see in it when the building is realised. Moreover, however static its physical form may be, what a building means changes dynamically over time due to changes in its use, new developments in its environment, other projects imitating or – on the contrary – opposing it, etc. (see also chapter 7 pp.168 ff).
must be made, opportunities to be had, and trade-offs that will improve the overall quality of the solution. These insights not only produce new design parameters and define new objectives to be achieved, but may also radically alter previously stated objectives” [Carrara & Kalay, 1994]. By consequence, the design process takes on a character of being ‘end-justified’, in the sense that the architect redefines the problem conditions so as to fit his proposed solution. Although such an approach might appear somewhat perverse, it is often quite plausible and even necessary given the under-constrained nature of most design problems [Rowe, 1982].

At this point, it is easy to understand why the view of architectural design as pure problem solving was doomed to failure. According to this view, which is still quite popular among AI and CBR researchers, an architect’s task comprises three distinct operations: 1. definition of the functional and behavioural objectives; 2. production of alternative design solutions; and finally 3. evaluation of the expected performances of the solutions and comparison with the predefined objectives [Carrara, Kalay & Novembri, 1994]. Yet, given the parallelism between problem and solution in design, it is obvious that this sequential model already starts from a false premise. Design problems cannot be comprehensively stated at the outset, but are gradually (re-) defined during design. By consequence, the division into and strict sequence of separate steps – problem definition, solution and evaluation – does not do full justice to the task of conceiving architecture.

That is not to say that evaluation does not show up in architects’ design process. On the contrary, the continuous oscillation between the different issues that must be taken into account is extremely intense and, just like juggling, difficult to maintain for a long time. Architects therefore seem to alternate moments of intense activity with more quiet periods. It is during these quiet periods, when the architect evaluates the current state of his design, that the technical-rational mode of knowing pops up [Schön, 1983].

Taking up the question of knowledge engineering again, it seems therefore reasonable to consider the task of conceiving architecture as walking on a thin line between two ways of knowing two types of knowledge, i.e. between passive knowledge and active knowing of both components and concepts.

1.3 Knowledge channels for conceiving architecture

In the previous sections, architectural design has been characterised as an extremely complex activity that makes high demands upon architects’ knowledge. Yet, if the process of design requires considerable amounts of knowledge beyond what is stated in the design problem, the question arises: where then do architects find all this knowledge?

As will be pointed out in the following chapter, cases, i.e. concrete projects from the past, have been recognised as an important source of knowledge during design.

11 The term ‘problem solving’ in this thesis refers to problem solving in the strict sense of the word, that is solving tame problems as opposed to wicked problems.
[Oxman, 1994]. Quite a few authors, especially in the field of Case-Based Design, claim that architects make extensive use of previous projects in the act of designing [Domeshek & Kolodner, 1992; Fang, 1993; Hua & Faltings, 1993; Pearce, Goel, Kolodner, Zimring, Sentosa & Billington, 1992; and Schmitt, 1993, to name but a few]. Several stages of the design process, ranging from the initial programming stage over conceptual design to the final development of detailed working drawings, are said to be supported and/or constrained by cases from the past. In the final stage, for example, design solutions from previous projects are often very useful. Think of the use in many design offices of standard details, which reduce the production of working drawings mainly to a copy-and-paste exercise. But let us return to the focus of this chapter, the early conceptual stage of architectural design. During this stage, architects are said to spend much time thinking about existing designs, reviewing literature, pouring over formal and informal documentation of earlier works. In their search for ideas and concepts, they visit buildings, browse magazines and pull old blueprints from their own files. The reason seems to be that previous design cases provide grist for a number of decisions to be made during concept generation [Domeshek & Kolodner, 1992].

This grist, however, is only one of the ingredients of the breeding ground that nourishes – and is nourished by – architectural design. Perhaps an appropriate name for this breeding ground is the term ‘culture medium’, both in the sense of cultural baggage, and in the sense of a seed-bed for growing micro-organisms. It embraces various substances, phenomena and traces – both from within and from outside the field of architecture – all of which may function as raw materials for concept generation. From time to time, the relationship between this seed-bed and architectural design is being denied – the tabula rasa idea of the Modern Movement is a case in point – yet in some sense even such denial may be considered a sort of relation. Within the context of this first chapter, we would like to situate the use of cases as postulated by Case-Based Design research within the broader context of architecture’s culture medium and focus on how this medium feeds concept generation.

Regarding the early stages of the design process, there are a number of mechanisms, devices, strategies that architects use to profit from the design knowledge embedded within architecture’s culture medium. As we will point out, these mechanisms help to extract from this medium smaller or larger bodies of knowledge, which are plugged into the design task at hand. In this respect, they can be seen as instrumental for managing the complexity of architectural design. Professional architects have become apt at using one or a combination of these strategies to transform the design situation from an ill-defined set of requirements to one or a few incipient ideas. In order to provide some background for the further study of the role of cases in conceiving architecture, this section will briefly discuss how architects draw upon this seed-bed in the act of designing.

Due to lack of time, we are forced to limit our discussion to a selection of mechanisms – heuristic search, analogy, metaphorisation, typology and case-based design (in the strict ‘CAD’ sense of the word). This selection implies that the coverage of the discussion may be neither representative nor complete. Moreover, when
considering each of these mechanisms as such, the discussion will probably appear disputable, rather unfounded and above all incomplete. Indeed, a profound investigation of say analogy in architectural design transcends the scope of a single thesis, let alone of this first chapter. Nevertheless, our discussion may open interesting perspectives on how these mechanisms function as knowledge channels in the act of designing. As will be pointed out, these channels do not operate independently, but are highly interwoven. Together, they form a sort of tissue, a filter between the breeding ground and the ‘growing’ design. Thus, although each mechanism has been awarded more or less a separate sub-section, we do not deny in any way their complex intertwining. By consequence, the fact that a specific design – say Le Corbusier’s Maison DOM-INO – is cited as illustration of one mechanism, does not exclude that it may illustrate other mechanisms as well. Moreover, if a study of case (re-)use in architectural design is hampered by a Babel-like jargon, this applies even more to the following discussion. What one calls an analogy is a metaphor for another, and while some use the term heuristic as a collective noun for all strategies architects use, others consider it a specific strategy. Compared to other mechanisms, however, heuristic search seems subject of relatively little debate, and is thus perhaps a good candidate to set the ball of our discussion rolling.

1.3.1 HEURISTICS

In general, heuristics can be thought of as devices for narrowing down the search space when faced with a problem. They do so by performing a pre-selection on the possible solutions. Chess players, for instance, tend to investigate only a small number of possibilities, whereby the choice of these possibilities and the order in which they are verified is determined by heuristics [De Mey, 1972 p.37; referred to in Schreurs, 1986 p.63].

With respect to architectural design, heuristic search refers to directed search for a solution based upon experience and insight in the problem situation. Heuristics supply ‘contextual knowledge’, i.e. knowledge of what is reasonable within a given context. This knowledge is derived by generalisation from specific examples of past solutions, and is subject of general consensus [Rechtin, 1991 pp.18-19]. Heuristics may address a variety of aspects, such as climate, human behaviour or building materials, and are known or presumed to lead to a possible, yet not necessarily optimal solution.

The usefulness of heuristics is many-fold [ibid. p.20]. They can help evaluate architectural choices, act as sanity checks or first-order assessments, and even function as teaching aids. Their principal potential, however, consists in adding structure to ill-defined design situations and narrowing down the scope of possible solutions. Since heuristics are accumulated through many years of education, experience and examples, only experienced architects can fully profit from their potential. Whereas the design process of novice architects typically proceeds through trial and error, their more experienced colleagues call in heuristics – consciously or unconsciously – to rapidly reduce the abundance of possibilities to a manageable handful. Although many

---

12 See introduction, p.2.
heuristics do not only concern conceptual design, they can serve architects particularly well when used collectively and early in the design process [ibid. p.20; italics added]. Commitments made in this stage have the largest effects and are the hardest to undo later.

In addition to this interpretation as generalisations of past solutions, however, the term heuristics may be interpreted in a wider sense as well. Peter Rowe, for instance, uses heuristics as an umbrella term to cover all aids, devices and mechanisms used by architects in the act of designing [Rowe, 1982]. According to this view, the knowledge channels that will be discussed in the following sub-sections are sub-classes of the great big family of heuristics. To some extent, this view seems to be shared by Jan Schreurs, who brings up metaphor and analogy as two different ‘heuristics’ [Schreurs, 1986 pp.64-69].

1.3.2 ANALOGY

In general, analogy can be defined as a likeness of relations, such as ‘A is related to B like C is related to D,’ or in short, ‘A:B :: C:D’ [Casakin, 1997 p.16]. Such likeness implies the existence of a higher-order abstraction that holds equally well for A:B and C:D. Especially in the early stages of the design process, analogy seems to be a very powerful tool, as it can bring forth valuable knowledge from a known situation (the source or base A:B) to the ill-defined design situation at hand (the target C:D). Architects try to reason and learn about their design by relating it to a more familiar situation in the culture medium that can be viewed as structurally parallel.

Of course, architects do not hold absolute sway over analogical reasoning. According to George Polya, analogy “pervades our thinking, our every day speech and our trivial conclusions as well as artistic ways of expression and the highest scientific achievements” [Polya, 1973 p.37]. Indeed, human use of analogy is apparent both in laboratory experiments and in naturalistic settings, including politics, psychotherapy and scientific research [Holyoak & Thagard, 1997 p.35]. There are a number of anecdotal examples of scientists having found major breakthroughs by analogical reasoning. A case in point is the discovery of the benzene structure by the chemist Friedrich von Kekulé [Verstijnen, 1997]. In those days, organic molecules were assumed to consist of strings of carbon atoms. After numerous fruitless attempts to establish the structure of the benzene molecule, von Kekulé was drowsing in front of the fireplace when suddenly the solution to his problem showed up. The flames, which took the form of snakes biting their own tails, suggested him to drop the string-assumption, and think of the benzene molecule as a ring structure.

---

13 To be precise, Rowe distinguishes between five classes of heuristics according to the subject matter involved: anthropometric analogies, literal analogies, environmental relations, typologies and formal ‘languages’. Literal analogies and typologies will be extensively discussed further in this text.

14 Apparently, analogy pervades our thinking already from our earliest youth, as Polya illustrates with the following joke. A small boy’s pet had to be taken to the veterinary and he inquired: “Who is the veterinary?” When he got the answer “The animal doctor”, he asked: “Which animal is the animal doctor?” [Polya, 1973].
In general, human use of analogy is guided by three broad classes of constraints: 1. to some extent by direct similarity of the elements involved; 2. by a pressure to identify an isomorphism – a set of consistent one-to-one correspondences or structural parallels – between the elements of the source and the target; and finally 3. by the reasoner’s goals or purpose – what the analogy is intended to achieve [Holyoak & Thagard, 1997 p.36]. According to the authors, “these three kinds of constraints – similarity, structure and purpose – do not operate like rigid rules dictating the interpretation of analogies. Instead they function more like the various pressures that guide an architect in creative design, with some forces in convergence, other in opposition, and their constant interplay pressing toward some satisfying compromise that is internally coherent” [ibid.].

In this respect, it is perhaps not entirely coincidental that analogical reasoning frequently pops up in architectural design. Although architects are often extremely reluctant to admit it, many of them have used, and are using analogy for concept generation. They borrow existing forms, or form-giving constructs, as a point of departure for their project. Rowe speaks in this respect of ‘literal’ analogies, because the subsequent architectural forms most often resemble the physical analogue very closely [Rowe, 1982]. In his book Design in Architecture, Geoffrey Broadbent considers analogic design as the most potent source of creative ideas in architecture [Broadbent, 1973 p.35]. In case of analogic design, he contends, designers draw visual analogies with existing buildings, with forms from nature, from painting and so on; structural analogies with the feeling of tension and compression in the designer’s own body; philosophical analogies with principles from physics, biology, and so on [Broadbent, 1980 p.311]. More in general, Broadbent distinguishes between two ways of using analogy as a creative technique for conceiving architecture. Depending on the source or basis of what architects borrow, he speaks of either iconic or canonic analogy [Broadbent, 1973 pp.338-339]. Of course, this subdivision may seem somewhat arbitrary, as a single analogy might possess both iconic and canonic qualities. Yet, given the purpose of this chapter, making this distinction is nonetheless useful to help us unwire the threads between architectural design and its breeding ground.

---

15 To be precise, Broadbent distinguishes between four types of design: pragmatic, iconic, analogic and canonic. Historically, analogic design is linked with the first use of drawings and models as analogues for the building whilst it was being designed [Broadbent, 1980 p.311].

16 This distinction is not to be confused with two other categorisations Broadbent describes in this book, although there is obviously some overlap between them. As already mentioned, Broadbent distinguishes between four ways of generating forms architects have used during the course of history: pragmatic, iconic, analogic, and canonic design, in chronological order of application [Broadbent, 1973 pp.25 ff]. Later on, he replaces iconic by typologic and canonic by geometric design [Broadbent, 1980]. Secondly, Broadbent makes a distinction – the one referred to here – between iconic and canonic analogy as two ways of using analogy in the creative sense [Broadbent, 1973 pp.338-339]. Finally, he distinguishes between three types of analogy generation, namely personal, direct and symbolic analogy [ibid. pp.350-353].
1.3.2.1 Iconic analogy

In case of iconic analogy, sources are often objects from the natural world, or from fields outside architecture as such. Other bases of iconic analogy are images, scenes, stories or paintings of real or imagined circumstances.

Famous examples of ‘natural’ or ‘organic’ analogies can be found in the work of Frank Lloyd Wright, Santiago Calatrava and Eero Saarinen, to name only a few (figures 1.2 to 1.5). The volume of Wright’s Unitarian Church in Madison, for instance, was inspired by his own hands clasped together in prayer, whereas the columns of his Johnson Wax Building were shaped by analogy to water lilies. According to his own account, Calatrava heavily relies on forms like the human skeleton or frames of other animals during design. He imitates these organic forms not so much for the sake of form, but primarily because of structural issues, such as dynamic balancing [Lawson, 1994]. This can be seen in the Campo Volantin footbridge in Bilbao, to name only one of his bridge designs, or the completion he proposed for St. John the Divine Cathedral in New York. Another famous example of ‘natural’ analogy we want to mention, is the TWA Terminal at Kennedy Airport in New York. Referring to the concept of flying, Saarinen gave it a wing-shaped vault analogous to the wings of a bird.

As illustrations of ‘unnatural’ analogies, we can cite Jørn Utzon’s Opera House in Sydney and Rem Koolhaas’ project for the Sea Trade Terminal in Zeebrugge (figures 1.6 and 1.7). Just as the opera house was strongly conditioned by the shape of sailing yachts entering Sydney harbour, Koolhaas started from the shape of a bollard to moor ships at the quay. A well-known example of visual analogy with a painting is the Schröder House in Utrecht, designed by Gerrit Rietveld in close collaboration with Mrs. Schröder (figure 1.8). The building strongly resembles a Mondriaan painting, in the formal delineation of its facade as well as in its interior plan. In doing so, it materializes the two-dimensional aesthetic qualities of De Stijl17 [Sharp, 1972 p.74].

Whether based on water lilies, yachts or a Mondriaan painting, the use of iconic analogy in architecture is typically appreciated because of the added value it gives to

---

17 De Stijl is a group of Dutch artists in Amsterdam in the early 1920s, including painters, architects and poets. Its most outstanding painter was undoubtedly Piet Mondriaan. As a movement, De Stijl influenced painting, decorative arts (including furniture design), typography and architecture.
the building, in other words because of the resulting design *product*. Rarely, however, its benefit has been considered from the perspective of the design *process*. When ‘juggling’ with the host of aspects a design must address, the introduction of an existing form provides architects with something to hold on to. It considerably reduces the abundance of possibilities for the component ‘form’, and provides a rough contour within which all of the design can be moulded. That is not to say that this form immediately solves all problems at once. On the contrary, as the story of the Sydney Opera House illustrates, accomplishing the full potential of an analogy and drawing its consequences throughout all aspects of the design, is far from trivial a task. Yet, given the unorganised way in which these aspects pop up and interact early in the design process, it seems plausible to consider iconic analogy as a powerful aid for managing the complexity of conceptual design.

1.3.2.2 Canonic analogy

Whereas iconic analogy derives from concrete objects, shapes or images, canonic analogy is based on abstract geometry. Sources include two- and three-dimensional grids as well as proportional systems. The use of grids is rather straightforward, yet perhaps the application of proportional systems might deserve some extra explanation.

A proportional system can be thought of as a series of rectangles all bearing the same ‘ideal’ proportions. A series that was rather popular in the Renaissance, is the one based on the proportion $1:\sqrt{2}$. The golden section, another well-known proportional system, starts from the proportion $1:\phi$, in which $\phi$ is the irrational ‘golden number’. A third proportional system we want to mention, is Le Modulor. Developed by Le Corbusier, Le Modulor combines two $\phi$-series derived from a standard human figure, the so-called ‘six-foot detective’. The Red series contains the detective’s length, i.e. 1830 mm; the Blue series is based on his length when raising his arm, i.e. 2260 mm. In order to subdivide the resulting rectangles, Le Corbusier made use of the so-called Tracés Régulateurs, three geometric construction lines within a rectangle that may help further organise the composition.18

Just like grids, proportional systems are used by architects as guidelines for bringing correspondence and coherence in their projects. When deciding on the dimensions of a design, be it the height of a building, the length of a room or the width of a window, preference is given to rectangles with particular proportions. Building designs based on Le Modulor and the Tracés Régulateurs are, for instance, Atelier Ozenfant and Villa La Roche-Jeanneret, both in Paris, Villa Stein in Garches, and the Unité d’Habitation in Marseille [Eveno, 1987] (figure 1.9). A less known, yet radically consequent application of canonic analogy, is the summer retreat for Krishnamurti in Ommen designed by Johannes Brinkman and Leendert van der Vlugt (figures 1.10 and 1.11). Both plans and elevations of this retreat are entirely geared to the standard dimensions of Tekton panels, the main building material used. The grid imposed by these prefabricated panels, however, has been put to an aesthetic use resulting in a flawless composition of volumes and balanced arrangement of planes [Molenaar, 1996].

---

18 For a more detailed description of proportional systems, we refer to [Knable, 1987].
As the latter example illustrates, the use of canonic analogy in architecture may be and is often explained by its power to provide buildings with a visually perceivable order. Geometric correspondence and coherence in, say, the façade of a building would be perceived as more balanced. According to Broadbent, another reason for using canonic analogy is that it gives buildings more authority. Architects who lack confidence in their personal judgements would call in the authority of a geometric system [Broadbent, 1973 p.35]. In our opinion, however, the use of canonic analogy during the design process may be justified in a more pragmatic way as well. Just like iconic analogy, it helps to reduce the number of possible compositions – both in 2D and 3D – and thus to manage the complexity of the design task at hand. To say it in the words of Mies van der Rohe: “I put a grid over the whole campus; that was more a mechanical help. After that we had not to speculate where we put our columns. We put our columns on the crossing points of the grid, all the way through…” [Mies van der Rohe, 1960; referred to in Broadbent, 1973].

1.3.3 FROM ANALOGY TO METAPHOR (AND BACK TO ANALOGY)

Analogy, we have pointed out, is a potent source of creative ideas in architecture. However, what one author calls analogy, is a metaphor for another. Indeed, if the knowledge channels between architectural design and its breeding ground are said to be highly intertwined, this definitely applies to analogy and metaphor.

None other than Aristoteles would have defined\textsuperscript{19} the process of metaphorisation as “giving the thing a name that belongs to something else; the transfer being either from genus to species, or from species to genus, or from species to species, or on grounds of

\textsuperscript{19} We deliberately use the conditional, as this is only one translation of Aristoteles’ definition and differences may exist with other translations.
analogy” [Aristoteles, 1980; referred to in Schreurs, 1986 p.147]. According to Schreurs, the first and most important element of a metaphor is the fact that a transfer takes place, in Aristoteles’ examples between different contexts. A second characteristic is that, in spite of the differences, a similarity exists between the transferred terms from these contexts.

In his dissertation Model and Metaphor, André Loeckx adopts all elements of Aristoteles’ definition, yet propounds the innovative character as a prerequisite for metaphorisation. He defines a metaphor as “an addition of a new aspect to a semantic field or as transplantation of one field to another on the basis of similarity, difference and effective transfer so that an event takes place that, implicitly or explicitly, is experienced as innovation” [Loeckx, 1982 p.II.306; italics added]. In short, the process of metaphorisation entails a transfer of meaning from one domain to another, thereby merging elements that are usually seen as unfamiliar. By consequence, a metaphor can be understood as an unconventional way of describing an object, event or situation as another object, event or situation [Indurkhya, 1992].

Just like analogies, the use of metaphors is far from unique to the field of architectural design. Most people probably associate metaphors with literature, as they frequently pop up in poems and novels. Poets, novelists and other artists use metaphorisation as a fertile basis for creativity: seeing something as something else is an essential ingredient of creative behaviour. The greater the distance between both, the more creative the result [Neuckermans, 1994 p.oproc-13]. Yet, metaphors are also used in less ‘artistic’ fields, like for instance quantum physics or cognitive science. They have proven particularly helpful to describe, explain or think of phenomena that are not yet completely understood. Examples of ‘scientific’ metaphors are building blocks for quarks [Hoorn, 1997], or the image of a personal notebook for human memory.²¹

²⁰ It should be noted, however, that a(n) – all too – great distance may result in a far-fetched metaphor as well.
²¹ See chapter 2, p.45.
CHAPTER 1

To illustrate the process of metaphorisation in architectural design, we can cite Aldo Rossi’s extension to the San Cataldo cemetery in Modena. In this design – the winning entry for a national design competition – the cemetery seems to be conceived as a city for the dead [Schreurs, 1986 pp.213 ff]. Viewing death as a reversal of life may have inspired Rossi to design the cemetery by inversing a city for the living – turning an apartment into structural work, cutting of the perspective of a path, etc. (figure 1.12). It should be noted, however, that metaphors in a building identified by critics, visitors or other people may differ from what the architect used (consciously) as basis for his design [Neuckermans, 1994]. Indeed, when conceiving architecture, architects do not necessarily follow the same path of reasoning as those who analyse their design afterwards. In case of San Cataldo, Rossi himself would have handed the key to identify the city metaphor in ‘L’azzuro del cielo’, the accompanying note in which he explains his design to the jury [Schreurs, 1986 p.231], yet such post-hoc declaration still does not prove he really worked that way during design. Nevertheless, one can easily imagine that if Rossi has consciously used this metaphor in the act of designing, it may have functioned as a framework to structure the design situation, and therefore facilitated conceptual design.

At first sight, the use of metaphor in architectural design seems largely analogous to that of analogy. Nevertheless, several authors agree that analogy and metaphor are two quite different phenomena. What they do not agree on, however, is what this difference consists of. According to Hernan Casakin, for instance, the main difference lies in the nature of the relationship between ‘source’ and ‘target’. Whereas an analogy depends largely on syntactic (structural) similarity between two objects, a metaphor is based on semantic relations [Casakin, 1997 p.16]. To illustrate this difference, we can refer once again to Le Corbusier’s Unité d’Habitation in Marseille (see figure 1.9). The relation between a wine bottle rack and the independent frame of the dwelling units is mainly structural, and therefore analogical. In contrast, Le Corbusier’s reference to an ocean liner for the design of this building rather functions as a machine metaphor [Goldschmidt, 1994].

For Schreurs, however, the wine bottle rack is a metaphor, just like the ship and the machine [Schreurs, 1986 pp.311-319]. Rather than completely distinct phenomena, he seems to consider analogy and metaphor as the extremes of a continuum [ibid. p.249]. To some extent, both strategies operate in the same way, but what differentiates one from the other is the degree of innovation, the metaphor being more novel. This seems to chime with the great emphasis Loeckx places on the innovative character as a prerequisite for metaphorisation [Loeckx, 1982 p.II.306].

Another attempt to point the distinction between metaphor and analogy in architectural design is made by Broadbent. His attempt takes up a quote from Charles Jencks, who cites the Casa Battló of Antonio Gaudí as an example of architecture with a rich variety of meanings (see figure 1.13) [Jencks, 1977; referred to in Broadbent, 1977 p.48]. According to Broadbent, Jencks misuses the term metaphor to refer to straightforward, simple, visual analogies in Gaudí’s design:

“The first two floors have a curious colonnade formed by visual analogy with human bones. The main façade, with its undulating forms in brown, green and
blue ceramics, obviously is an icon for the sea, whilst Jencks points out the boldly tiled roof actually ‘looks like’ a dragon. It is dominated by a pinnacle bearing a Christian cross. Bones, sea and dragon are all icons at the level of simple analogy, but as Jencks also points out the whole is an expression of Catalan nationalism in which the dragon of Castille has been slain by St George – the patron Saint of Barcelona. The bones of course represent those of the martyrs who have died in the cause. Now obviously this represents a ‘higher’ level of meaning – shading towards say illusionism – which certainly is not revealed by a direct reading of the simple, visual analogies. This is metaphor, and we should do well to reserve the word for such deep and subtle meanings, rather than applying it, indiscriminately, to simple, visual analogy” [Broadbent, 1977 p.480].

Thus, instead of attributing the distinction between analogy and metaphor to the source-target relationship or degree of innovation, Broadbent simply situates them on two different levels of meaning. In case of the Casa Battló, the metaphor seems to occupy the higher level, yet the roles may be reversed as well. A case in point is the museum in Gibellina Nuova designed by Francesco Venezia. After a forceful earthquake in 1968, only one wall in Gibellina remained standing: a fragment of the neo-classicistic façade of the Palazzo di Lorenzo. In 1982, Venezia was asked to incorporate this fragment in the design of a museum for the new city. Even before it was completely finished, Schreurs identified in this building no less than five metaphoric transfers: the old city is partially within the new city; the museum is dug up, the fragment dug in; the ruin is a new wall; the façade is a work of art; the museum is a small city [Schreurs, 1986 pp.179 ff]. When confronting these metaphors, Schreurs discovered a certain coherence between them, a higher-order abstraction that may be characterised as … analogy! “Four metaphors are kept together by a common motive, subtended by an *ana-logic*” [ibid. p.202; italics added]. For each of these four, the contexts between which a transfer takes place can be ordered along the semantic axe old-new or past-present. The analogy between these contexts makes the metaphors intensify each other’s meaning.

### 1.3.4 TYPE IN A NUTSHELL

Apart from metaphor, analogy is also closely related with the notion of type. Type has been used by architects throughout most of architectural history, and its significance for architecture has been described by many authors in many different ways. Usually, the first use of type in architectural treatises is ascribed to Quatremère de Quincy in the early 19th century. In Rossi’s opinion, this major theoretician of architecture gave a masterly definition of type, by opposing it to the notion of model:

“*The word ‘type’ represents not so much the image of a thing to be copied or perfectly imitated as the idea of an element that must itself serve as a rule for the model … The model, understood in terms of the practical execution of art, is an object that must be repeated such as it is; type on the contrary, is an object according to which one can conceive works that do not resemble one another at all. Everything is precise and given in the model; everything is more or less*”
CHAPTER 1

vague in the type” [Quatremère de Quincy, 1832; referred to in Rossi, 1982 p.40].

However, the idea of type, as opposed to the explicit use of the term by theorists, is a much older phenomenon, going back as far as Vitruvius [Madrazo, 1995]. To give a comprehensive overview of the evolution of this phenomenon would go beyond the scope of this thesis, let alone its first chapter. Instead, we shall try to clench this evolution in a nutshell, and quickly move on to how type functions as a device for concept generation.

Particularly helpful in this respect is the article ‘Typologies’ by Philippe Panerai, who places type and typology in architecture in a historical perspective [Panerai, 1981]. As Panerai points out, the first explicit manifestations of typology in architecture pop up in the scientific climate of the first industrial evolution. In the 18th century, scientists started using type as a concept expressing the essence of objects or persons. This concept enabled encyclopaedists in botany, zoology and mineralogy to describe a rather large number of related objects in an economic way. In the early 19th century then, the notion of type found its way from the natural science to the field of architecture.

Being a cultural production, however, architectural design shows considerable differences with the natural evolution of animal or botanical species. By consequence, type in architecture is rarely studied as an aim in itself, but rather from a practical/generative perspective, i.e. with an eye on the production of new buildings. Depending on the explicitness of this generative aspect, Panerai distinguishes between two kinds of types: standard types and type-layouts.

Standard types result from a tacit agreement on the correspondence between a set of spatial layouts and style elements on the one hand, and a building practice on the other hand. This correspondence is stable in the sense that it is generally accepted in a particular period and society. Examples of historically standardised types are the Roman villa, the Gothic cathedral, the Ottoman mosque, the mansion or the middle-class house. In their time, they served both as the program and as the spatial scheme of new cathedrals, mosques or mansions to be built. At least until the 19th century, these types were passed on from one (master) builder to another, not through catalogues, but through the master-apprenticeship relationship.

Whereas the generative aspect of standard types is rather implicit, typified elements or type-layouts are postulated explicitly as production instruments for the built environment. Particularly illustrative in this respect is Jean Nicolas Louis Durand’s idea of generative typology. Durand does not only propose a catalogue, le Parallèle, from which (student-)architects can select typified examples, i.e. examples stripped from all history and environment. He even provides them with a manual, le Précis, to

---

22 The distinction between type and model largely coincides with the one between type and prototype. Whereas a type covers many variants, a prototype is made for the production of a whole series of identical exemplars [Loeckx & Verpoest, 1993 p.144].

23 Note that the notion of standard in standard types is slightly different from that in the standard houses promoted by the Modern Movement. Whereas the former have gradually become the standard in the course of history, the latter are suddenly imposed as the norm.
interpret these examples in new design situations.\(^\text{24}\) Since this interpretation hardly takes into account local conditions or variety of cultures, Durand’s generative typology perfectly chimes with the idea of mass production, the production of whole series of identical exemplars. The more or less vague type becomes a precise model, the type becomes prototype. This idea of prototype would be taken up again in the post-war period by the Modern Movement, which advocated the industrial production of standard ‘Existenzminimum’ houses, and lives on today in the phenomenon of so-called catalogue houses [Wortmann, 1999].

In the 1950s, however, the ‘original’ notion of type suddenly seems to experience a revival, marked by Saverio Muratori’s book *Studi per un operante Storia urbana di Venezia* [Muratori, 1959; referred to in Panerai, 1981 p.16]. The book results from ten years of research, ten years of examining the Venetian urban tissue under a typological microscope. Carlo Aymonino, together with Rossi, would later attempt to systematise Muratori’s approach by testing it on other cities [Aymonino & Rossi, 1966; referred to in Panerai, 1981 p.18]. Aymonino considers typological research first and foremost as a knowledge instrument, a means to understand the city in its historical evolution and to grasp the logic behind current urban phenomena. Nevertheless, the revival is not a return to the scientific, independent notion of type – i.e. independent of production – but rather to the notion of standard type, be it with extension to less ‘famous’ types than the ones mentioned above. For Aymonino, as for Muratori, the knowledge resulting from typological research is meant to serve the design process: it is a prerequisite for a well-reasoned intervention.

Indeed, just like heuristics, the use of types enables architects to profit from knowledge about past solutions to related architectural problems [Rowe, 1982]. A major difference, however, is that this knowledge is not expressed in the form of rules, but takes the form of an architectural object that summarises the essential characteristics shared by a family of buildings [Loeckx & Verpoest, 1993 p.142]. This object embodies valid principles that, as judged from examples, lead to adequate architectural qualities. Although these principles can address any aspect of architectural design, most of them concern the spatial distribution of functions. Therefore, when starting from a specific type, an architect knows more or less what the spatial configuration of his building is going to look like. In this way, type can be considered as a vehicle to store solutions that have reliably shown to be performant, and to reuse these solutions in new design situations.

Yet, if this vehicle takes the form of an object instead of a rule, questions may arise as to how to ‘apply’ such an object in a new design situation. On this matter, the authority to consult is undoubtedly Rossi. In his theory as in his practice, the notion of type occupies a pre-eminent place, and is inseparable from the notion of analogy. Scheurs summarises this twofoldness in Rossi’s work as follows: “analogy forms the core of the subjective moment in the design process, a process that works on objective data obtained from a typological analysis of existing cities” [Scheurs, 1986 p.213]. In other words, typologic design may be thought of as a special form of analogic design

\(^{24}\) For the story of the development of *le Parallèle* and *le Précis*, see also p.31.
CHAPTER 1

whereby the role of source is played by a building type. The resulting design refers analogically to the type at issue. Since this type results from examining the urban tissue, the city constitutes in fact the material of architecture [Rossi, 1982 p.170].

Typologic design, however, already existed before typologic dissections of urban tissues came into vogue. Broadbent considers it one of the four basic types of design – more precisely, when plotting them chronologically, as the second oldest of the four. In typologic design, he contends, “the designer starts with some fixed ‘mental image’ of a familiar building form as the best possible solution to this problem of using the materials which happen to be available at a particular place, with a particular climate, to house an established ‘life-style’” [Broadbent, 1980 p.311]. Originally, Broadbent had labelled this type of design ‘iconic design’ [Broadbent, 1977]. Yet, when studying the overlap between on the one hand his four types, and on the other hand Charles Peirce’s three-part categorisation of the sign into icon, index and symbol [Peirce, 1897-1903; referred to in Broadbent, 1980], he noticed that his analogic and canonic design represented different aspects of iconicity, whilst his iconic design had little or nothing to do with Peirce’s original concept. That is why he changed its name into typologic design, thus lining up with the long tradition of design by typology [Broadbent, 1980 p.326].

Given this long tradition, examples of typologic design may be found throughout the entire architectural history. Almost immediately the villas designed by Andrea Palladio come to mind. It might be somewhat misleading though to cite them within the context of this chapter, for this would suggest that Palladio had first chosen a type (the Palladian villa) and subsequently created different variations of it (his various villa designs). However, only after Palladio had created a coherent collection of architectural works, it was possible to speak about the common type underlying the individual designs [Madrazo, 1995]. In this respect, a better illustration of type as generative design strategy may be the use of the Palladian villa by other architects. Even well into the twentieth century, Wright started from this type when designing his first prairie house, a building that later would develop into a type itself.

Closer to home, a typical example of typologic design is represented by the numerous ‘bel-étage’ houses that were built after the second World War. The bel-étage type refers to terrace houses shunted on six meter plots, with garage and entrance at ground floor level and a through living room spanning the entire width of the façade on the first floor [Loeckx & Verpoest, 1993].

More recently, the work of some present-day Flemish architects seems to point at another revival of typologic design, be it a more subtle and sophisticated offspring than in the post-war period. A case in point is the house of Marie-José Van Hee, a building that quietly combines the qualities of several basic building types [Jacobs, 1998 p.145]. Located in the centre of Ghent, Van Hee conceived her L-shaped house

25 Although Rossi himself pretends to join in with Quatremère’s notion of type as opposed to the notion of model (see also p.25), questions may arise as to the operationalisation of this notion in his own work. Despite the admiration for Quatremère’s definition, Francis Strauven points out, Rossi and the neo-rationalists in his wake tend to ply type as model in practice [Strauven, 1982].
CONCEIVING ARCHITECTURE

as a kind of farm yard. With the inner garden enclosed by the cloister and the tiled roof of the mezzanine above the kitchen, the ensemble is undeniably reminiscent of a monastery garden. In the living room, the high wooden ceiling, high-set vertical windows, wall recesses and block-shaped staircase, remind us of another type, a medieval knights’ hall (figures 1.14 and 1.15). By returning to the essentials of these types, Van Hee manages to maintain the quality and dignity of traditional buildings within a high-density urban context.

Figures 1.14 & 1.15.
Inner garden and living room of Marie-José Van Hee’s house in Ghent (Belgium), 1997.

A similar approach can be found in the work of Henk De Smet and Paul Vermeulen. They repeatedly started from basic types – sheds or terrace houses – and re-interpreted these in a surprisingly refreshing way. Think, for instance, of Martina’s house in Wetteren, which completely complies with the image of the traditional terrace house, and at the same time totally transforms this type [Paulissen, 1996a]. The same applies to the country house in Leopoldsburg designed by Sylvie Laenen and Dirk Janssen, who successfully adapted the original type of the Kempen farmhouse to contemporary building requirements [Paulissen, 1996b]. According to Strauven, type as used by this group of Flemish architects can be considered an architectural equivalent of the commonplace [Strauven, 1998]. The notion of commonplace stands for an ordinary statement whose original meaning, from frequent use, has faded or been forgotten [Bekaert, 1987]. In short, something common, yet referring to an origin. Weak as it may be, this reference is preserved by the commonplace. In a similar way, people live in ordinary building types, like traditional Flemish terrace
houses or small imitation farmhouses, without thinking about the structure or origin of these types. For the architects cited above, the return to the essentials of familiar types is a technique to include and preserve the advantages of traditional ways of dwelling in contemporary architecture.

1.3.5 TYPES AND INSTANCES

Depending on the degree to which these essentials are effectively readable in the resulting design, the latter may be considered an instance of the type at issue. As already mentioned, it refers analogically to the type itself, but also to other instances of this type. According to Schreurs, “The objects of a typological family can be said to refer to each other by their analogical characteristics. Each element acquires meaning only thanks to the analogical relation with a series of other elements, thanks to the correspondences and differences between the terms in a chain of analogical relations, which taken together makes up the type” [Schreurs, 1986 p.164]. The role played by the members of a typological family, c.q. instances of a type, should not be underestimated. They do not merely illustrate the type. Without them, there simply would be no type at all. One of the fundamental conclusions Muratori draws from his typological study of Venice, is namely that the type does not allow to be characterised outside of its concrete application [Panerai, 1981 p.16], i.e. outside of its concrete instances. Save as manifestations of a certain type, however, these concrete instances as such may serve concept generation as well, which brings us back to the starting point of our discussion: the (re-)use of concrete cases in architectural design. Because this (re-)use constitutes the subject of this work, it is introduced only briefly here.

Case-Based Design in the strict sense of the word can be viewed as a particular form of analogic design, one in which the analogy is based on a 'within-domain' source [Casakin, 1997]. Think, for instance, of Broadbent’s description of analogic design as drawing analogies with existing buildings, with forms from nature, from painting and so on [Broadbent, 1980]. We already brought up the common reluctance among architects to admit the use of analogy in their work. With regard to cases, this disinclination is even stronger. Few architects artlessly acknowledge to rely on previous projects during design. Instead, they tend to feel resentment when critics refer to other buildings in describing their work, apparently assuming that, somehow, this diminishes their ‘originality’ [Broadbent, 1973 p.38]. Nevertheless, as already mentioned, several authors claim that architects make extensive use of previous projects when conceiving architecture, as this would provide them with valuable knowledge for concept generation.

How and why architects use these cases from the past when conceiving architecture, is the subject of this thesis. Yet, within the context of this chapter, we would like to stress the value of case (re-)use as strategy for managing the complexity of conceptual design. In this respect, cases have some characteristics in common with heuristics and types. They encapsulate knowledge about previous design solutions that can be useful for the new design situation. As with types, this knowledge is expressed

26 See chapter 4, p.103.
in the form of an architectural object. A major difference, however, is that this object is not an abstract model that generalises among several examples, but a concrete project including all details that make the design unique. By consequence, cases potentially contain knowledge about every single aspect of architecture, as well as about the integration of these aspects into a coherent whole. In other words, cases can supply both component and concept knowledge.

The value of cases as vehicle to communicate design knowledge was already acknowledged in the early 19th century by Durand. We already cited his work in relation to typology, because of his generative mechanisms for designing buildings classified by their function. Yet, rather than catalogues of building types, the original versions of Durand’s *Recueil* and *Précis* may very well be considered as case bases avant la lettre. Le *Recueil* is a collection of ninety-two ‘planches comparatives’, which compare particular buildings with the same functional program by their plans, elevations, and cross sections [Durand, 1801]. The projects are said to have been built between 1780 and 1790, though Durand is not very strict when it comes to source quotation. Originally, the work had two pedagogical functions. It served as an encyclopaedia, in which architecture students could find examples for the courses on drawing and analysis. In addition, ‘les beaux plans’ and ‘les grandes compositions’ were also used by the students as source of inspiration for their own design projects, i.e. as support for conceptual design. Throughout the successive editions, however, le *Recueil* gradually transforms into a catalogue of stereotypical designs, in other words, the cases mutate into models. Durand makes ever more ‘corrections’ so as to make the originally historical examples correspond better to his principles of the ideal composition [Huet, 1984]. A similar evolution can be seen in le *Précis* [Durand, 1817]. In this book, Durand no longer speaks of ‘historical’ building types, but of ‘modern’ buildings that correspond to the needs of his time (hygiene and health, administration and justice, education and politics). Here too the level of abstraction increases throughout the successive editions, and reaches its very peak in la Partie Graphique of 1821, which is meant to replace the first volume of le *Précis*: buildings are no longer classified by functional program, but based on the number of ‘entr’axes’ [Szambien, 1984]. These later, adjusted editions of Durand’s work may explain why his name is almost automatically associated with the notion of type in architecture. Yet, the original versions of both books were obviously inspired by an intention to use concrete cases as sources of design knowledge.

Before proceeding with an example of case-based building design in the strict sense of the word, we would like to dwell on this knowledge-content of cases. Especially the encapsulation of concept knowledge seems to make case (re-)use a quite unique strategy for conceptual design. As described above, concept knowledge is concerned with integrating several aspects, issues and requirements of a building into a coherent whole.

---

27 A case base is a collection of indexed cases stored as complete patterns of experiences from the past (see chapter 2, p.51).

28 Note that the full name of this work is ‘Recueil et parallèle des édifices en tout genre, anciens et modernes, remarquables par leur beauté, par leur grandeur ou par leur singularité’ [Durand, 1801; italics added].
and meaningful design. Expressing how to achieve such integration in the form of a heuristic, type or model is extremely difficult, because few generalisations hold over all aspects of a design. Particularly significant in this respect is the observation already mentioned that heuristics are especially helpful when used collectively [Rechtin, 1991 p.20]. Since very few of them cover more than one issue, achieving integration between several aspects requires the combination of multiple heuristics.

An interesting attempt to formulate design knowledge in an integrated way, is Christopher Alexander’s *Notes on the synthesis of form* [Alexander, 1964]. The constructive diagrams he proposes in this book are meant as general principles of good design that consider several issues simultaneously. A constructive diagram can be thought of as a schema connecting the verbal requirements with their architectural answer, a sketch containing both problem and solution. It is the blending of a ‘requirement diagram’, which only gives information about the requirements, and a ‘form diagram’ bearing witness to the shape of the design. A striking fact about Alexander's principles, however, is that they have actually very little generality [Hua, Smith & Faltings, 1994]. Rather than general principles of good design, the constructive diagrams are prescriptions of specific design solutions in particular design situations. What Alexander’s attempt illustrates, is that integrating knowledge from many aspects almost automatically amounts to formulating particular cases of good design. Cases provide a natural solution to the integration problem in architectural design: being themselves integrated solutions to particular contexts, they can provide the glue that holds a design together [Kolodner, 1993 p.82]. Using a concrete design case enables architects to implicitly achieve trade-offs among the several issues, aspects and requirements a building must address. By consequence, case (re-)use seems to be a particularly powerful strategy early in the design process, when the architect’s task is to come up with a concept to integrate the different aspects of his design.

![Figure 1.16.](image)

Le Corbusier’s Maison DOM-INO, 1914.

The use of cases in architectural design can take on many different forms, some of which may be recognised more easily than others. For the time being, we will cite one example that cannot deny being based on a particular project from the past. The
example is once again taken from the repertoire of Le Corbusier, which illustrates that architects do not necessarily stick to a single strategy. In 1914, Le Corbusier began designing his famous Maison ‘DOM-INO’. Rather than a house designed for a specific site or context, DOM-INO stands for a housing system of two-storey units. In essence, each unit consists of three horizontal slabs, supported by six columns and connected by concrete stairs (see figure 1.16). Units can be filled in, linked or expanded in various ways. When designing this system, Le Corbusier seems to have been inspired by the Monastery of Ema, which he visited in 1907 on his trip in Northern Italy (see figure 1.17). Apparently the monastery had impressed the young traveller so greatly that he decided to visit it again three years later. Although it is difficult to know what exactly he found so impressive about the monastery, it seems reasonable to assume that he was greatly attracted by the simple, almost cubic forms of the individual cell units, as well as by the concept of ‘Living Apart Together’, that is the successful combination of privacy and community within a single building. Judging from Le Corbusier’s drawings, these features have seriously influenced his designs for the filling in of the DOM-INO units and for their arrangement into a larger whole (see figure 1.18).

29 It should be noted, however, that although the monastery of Ema is an important reference point in this design, it is definitely not the only one. Vice versa, the Maison DOM-INO is perhaps an obvious, but certainly not the only example of a design in which Le Corbusier draws on this monastery. The Unité d’Habitation in Marseille is – once again – a case in point.
In fact, these features would influence the architect’s concepts of housing and urban planning throughout a considerable part of his career [Serenyi, 1967]. Yet, we have chosen the Maison DOM-INO because Le Corbusier’s thinking on this system laid the foundation for his later architectural ideas, and because here the link with Ema almost immediately strikes the eye. It is, perhaps, not completely coincidental that we find such a striking example of ‘case-based design’ precisely in his repertoire: Le Corbusier was not the product of an established architecture school, but instead made the unusual choice of educating himself [Turner, 1977]. By consequence, we should not be surprised by the profound influence of buildings he was exposed to in his early years, as this exposure in effect constituted his ‘design education’.

### 1.3.6 A GRID ACROSS A COMPLEX FIELD

The task of conceiving architecture, we have pointed out, involves at least two types of knowledge, that is knowledge concerning the different aspects or components in architecture, and knowledge about how to integrate these components into a coherent and meaningful design. This section has reviewed a selection of mechanisms architects use during this task, to extract smaller or larger bodies of knowledge from architecture’s breeding ground. Table 1.1 gives an overview of these mechanisms and the types of knowledge they are likely to provide architects with.

<table>
<thead>
<tr>
<th>strategies</th>
<th>component</th>
<th>concept</th>
</tr>
</thead>
<tbody>
<tr>
<td>heuristic</td>
<td></td>
<td></td>
</tr>
<tr>
<td>analogy</td>
<td></td>
<td>light grey</td>
</tr>
<tr>
<td>metaphor</td>
<td></td>
<td></td>
</tr>
<tr>
<td>type</td>
<td></td>
<td></td>
</tr>
<tr>
<td>case</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TABLE 1.1.**

Strategies used by architects for conceptual design and the knowledge they are likely to supply. (We use grey tones as a linear scale of quantitative values with dark grey as major knowledge supply, white as no knowledge supply at all, and light grey as the middle value.)

When using *analogy*, architects call in an existing form (water lilies, yachts or a Mondriaan) or form-giving construct (grids or proportional systems) as a point of departure for their design. According to some, the relationship between ‘source’ and ‘target’ is usually one of structural similarity, which means that the form of the target is directly influenced by that of the source. By consequence, analogies first of all serve as channels of component knowledge, more precisely of knowledge concerning the component ‘form’. In second instance, however, analogies may function on the concept level as well. In architecture, the concept that integrates the different components into a meaningful design, is not necessarily an extra ingredient, it may very well be a component that is already present in the design situation. If the component ‘form’ is chosen to play this integrating role, analogy can be viewed, in some sense, as supplying concept knowledge too.
Following this line of thought, the use of *metaphors* in architectural design differs from the use of analogies in that the nature of the relationship between ‘source’ and ‘target’ is semantic rather than structural. Rossi’s idea to conceive a cemetery as a city for the dead does not have direct implications on the form of his design, nor on any other aspect, and yet influences all of them. The metaphor provides the architect with a framework to integrate all these aspects in a meaningful way. Consequently, it seems reasonable to consider metaphors primarily as channels of concept knowledge.

*Heuristics*, on the other hand, are obviously to be situated at the component level. Being derived by generalisation from past solutions, they usually encapsulate knowledge concerning one specific aspect architects must take into account, be it climate, human behaviour or building materials. Because few generalisations hold over all aspects of a design, heuristics rarely address more than one component at the same time.

In a similar way, *types* in architecture can be seen as generalisations of design solutions that have reliably shown to be satisfactory in the past. Instead of rules of thumb, however, these generalisations take the form of architectural objects that may serve as blueprints for future designs. Although these blueprints might embody virtually any aspect of architectural design, they usually focus on the spatial distribution of functions.\(^{30}\) This seems to suggest that starting from a specific type provides architects first and foremost with component knowledge, to be precise with knowledge concerning the component ‘function’. Yet, some architects – Van Hee, for instance, or Vermeulen and De Smet – seem to use the spatial distribution of a type in a more conceptual way, that is as a means to intensify the meaning of their design and, to some extent, of the type itself. In this respect, types may be seen as channels that supply concept knowledge as well.

*Cases*, like types, encapsulate knowledge about previous design solutions in the form of architectural objects. These objects, however, are concrete, full of detail, and thus implicitly containing knowledge about every single component of architecture, as well as about their integration into a coherent design. The particulars in a given project offer architects an integrated view of design issues that would be lost if these issues were taken up separately. By consequence, cases can be viewed as supplying both component and concept knowledge.

We should keep in mind, however, that these different mechanisms do not operate independently, but are highly interwoven. Moreover, the distinction between concept and component knowledge is only one pair of glasses to unravel the mechanisms architects use for concept generation. By consequence, table 1.1 does not claim to be the ultimate classification, but rather an attempt to map, to lay a grid across a highly complex, if not chaotic field. Or rather across a part of this field, for the strategies summed up constitute but a raw and incomplete image of the mimetic game between architectural design and its culture medium. We use the term *mimesis* here not in the strict – Aristotelian or Platonic – sense of the word, i.e. as imitation or faithful copy,\(^{30}\) Most probably, Rossi would not approve of our associating type with the functional aspect of architectural design. Yet, he admits that, for the most part, typological studies have failed to go beyond the aspect of function [Rossi, 1982 p.41].
but, following Hilde Heynen, as referring to more general figures of similarity and difference, to certain affinities or correspondences [Heynen, 1999 p.193]. It still has to do with copying, but with a specific form of copying which implies a critical moment [ibid. p.ix].

Heynen inherits this broader interpretation from Theodor Adorno, who is in turn indebted to Walter Benjamin’s mimetic theory of language. Although Adorno rarely describes mimesis in precise terms, the concept plays a crucial role in his Aesthetic Theory [Adorno, 1997; referred to in Heynen, 1999 p.183]. His interpretation definitely has a much broader connotation than the traditional notions of art as an imitation of nature. In his book, Adorno “refers to “mimesis” as meaning a kind of affinity between things and persons that is not based on rational knowledge and which goes beyond the mere antithesis between subject and object. The mimetic moment of cognition has to do with the possibility of approaching the world in a different way than by rational-instrumental thinking. For him mimesis is something else than simple visual similarity between works of art and what they represent. The affinity Adorno refers to lies much deeper” [Heynen, 1999 p.184].

With respect to architectural design, Heynen recognises such deeper affinity primarily in the process of metaphorisation, which is why she considers metaphor as operating in the realm of mimesis: “it [metaphor] exposes a hidden resemblance that can be observed between two entities which belong to different fields” [Heynen, 1999 p.195].31 When adopting this broader interpretation, however, there seems little reason for excluding other mechanisms – mechanisms discussed and not discussed – from the realm of mimesis, all the less given their complex interwovenness. In analogic, typologic and case-based design, too, buildings are conceived on the basis of processes of correspondence, similarity and difference. As has been illustrated, the reference points here are extremely varied in character: water lilies, a Mondriaan painting, a proportional series, a standard type, a concrete design case. All these ingredients of the culture medium lend themselves to being treated mimetically and thus to being translated into a new design. In this translation, analogy, metaphor, typology, etc. seem to act as wires, telephone wires passing design knowledge from the breeding ground to the new design situation. Once it is finished and can be built, inhabited, visited, studied and criticised, the new design – including the wires connecting it with the culture medium – becomes part of the medium itself, in turn lending itself to feeding future designs. In this way, the culture medium nourishes and is nourished by architectural design in an endless mimetic game.

1.4 Prior notices

The central thesis of this first chapter has been that conceiving architecture is an extremely complex task that makes high demands on the architect’s knowledge. It entails active knowing and passive knowledge both at concept and component level.

31 Apparently, the mimetic character of metaphorisation in architecture has struck other authors as well. See for instance [Schreurs, 1986 pp.366-367].
In order to comply with these demands, architects have an arsenal of strategies at their disposal, which supply one or more of the knowledge types needed for concept generation. We have attempted to describe these strategies from the point of view of an architect being involved in the design process. From this perspective, their major usefulness seems to be that they provide him with a stable starting point, with something to hold on to within the wicked web of aspects, issues and requirements. They allow for one or more ‘juggling balls’ of the design to be temporarily fixed, and thus to shift attention to the remaining issues, while giving orientation to their resolution.

The first part of this thesis is devoted to a critical exploration of one specific strategy, the (re-)use of concrete design cases. Yet, before embarking on this exploration, there are two prior notices to be made. First of all, we should be aware that there is no such thing as the use of cases in architectural design, just as little as one can speak of the design process in architecture in the sense of a fixed flow of work. Since a design problem is by definition unique – i.e. universally new or new to the designer – design processes are not replicable, and will always have something of a mystery tour. The fact that architects use a number of strategies that have proven helpful to support this process, does not change anything about that. Thus, regardless of which mechanism is used and when, conceiving architecture remains one of the “most magical and precious of all human activity – the creation of something new and original” [Lawson, 1994]. Secondly, a critical exploration of the role of cases in architectural design implicitly implies a value judgment of architecture. There are different viewpoints to assess the value of a piece of architecture, and along with the viewpoint, the judgment may vary. At one time in history formal considerations determined architectural quality, later functional ones. Today, we believe that what makes architecture valuable is its positive contribution to man and society as social, cultural and political phenomenon. There is nothing wrong with taking this view, provided we are clear about it being our view at this moment.
A critical exploration of Case-Based Design in architecture
WHAT IS CASE-BASED DESIGN BASED ON?

From Artificial Intelligence over Dynamic Memory to Case-Based Design

“Since people obviously learn from experience and reuse old solutions to solve new problems, why not have intelligent computers do the same thing? Rather than expect system builders, domain experts and knowledge engineers to figure out, code and enter into memory all the abstract rules that a system would need in order to function, why not give the machine a large library of experiences and some mechanism for applying those experiences to the task of solving new problems?”

Christopher Owens (1988)
CHAPTER 2

During the complex and knowledge-intensive task of conceiving architecture, cases, i.e. concrete projects from the past, are assumed to provide architects with a rich source of design knowledge. This very assumption has bred a relatively young branch of research known as Case-Based Design (CBD). The branch represents only one of the applications of Case-Based Reasoning (CBR), a problem solving theory and technology developed within the domain of Artificial Intelligence (AI).

In order to situate the roots of CBD, this chapter starts by taking a quick look into the vast domain of AI (section 2.1). Subsequently, section 2.2 discusses the theoretical foundations of CBR, the cognitive model built upon these foundations and the actual applications of this model. The idea is to zoom in step-by-step on CBD (section 2.3), and to provide some background for its further exploration in the field of architecture.

2.1 Artificial Intelligence

2.1.1 WHAT?

CBR has been developed as a theory and technology within the domain of AI. The term ‘Artificial Intelligence’ was introduced by John MacCarthy in 1965 at the Dartmouth Conference, a gathering of ten researchers from a variety of domains. Twenty years later, Marvin Minsky, one of these Dartmouth pioneers, defined AI as “the field of research concerned with making machines do things that people consider to require intelligence” [Minsky, 1985 p.326]. At first sight, this definition does not help us much further, as there does not seem to exist a general agreement on the meaning of the term ‘intelligence’, let alone on the tasks that require it.

Minsky himself, for example, links intelligence with the ability to solve problems. He describes it as the processes in our mind that enable us to solve problems we consider difficult [ibid. p.72]. Herman Neuckermans would rather reserve the term intelligence for human beings, who can think creatively and freely, i.e. not controlled by rules [Neuckermans, 1992 p.283]. According to Christopher Riesbeck and Roger Schank, on the other hand, “a machine is unlike to be missing some quality or other that would keep it from being intelligent” [Riesbeck & Schank, 1989]. They think of intelligence as a continuum rather than a ‘now you have it, now you don’t’ affair. The idea is that everybody has – and a machine can have – certain basic processes of intelligence, but what differentiates smart from stupid is the way in which these processes are run. In his textbook Essentials of Artificial Intelligence, Matt Ginsberg adopts a definition of intelligence which he owes to the British mathematician Alan Turing [Ginsberg, 1993 p.6]. In the 1940s, Turing proposed to judge a machine’s intelligence by the following criterion:

Imagine that you are typing into a computer terminal. At the other end of the line is either another person or an artificial system of some sort. You have thirty minutes to ask whatever you want; if, at the end of that time, you cannot reliable distinguish the human from the artificial respondent, the artificial system is deemed to be generally intelligent.
The advantage of the Turing test is that it gives a reasonable sharp description of intelligent behaviour. We can now refine Minsky’s definition by saying that AI is the field of research concerned with making a machine that can reliably pass the Turing test. The recipe of this machine mixes ingredients from domains like philosophy, linguistics, psychology, mathematics, physics, statistics, neuroscience and biophysics, spiced with a significant dose of computer science [Ginsberg, 1993 p.vii].

Before discussing the intentions and daily pursuits of AI researchers, we would like to mention an architectural variant of the Turing test developed by Piet Colruyt [1994, p.XI]:

Imagine that a client asks both a laurelled architect and a computer to design a building, say a house, a castle, an apartment or a school. If an expert in the field of architecture—a critic, a theorist or an historian—cannot distinguish which design is made by the architect and which by the computer, artificial architectural design is possible.

The concrete version of the ‘Colruyt test’ has even more appeal: if statistically anonymous computer entries win more than half of a number of important architecture competitions, one could speak of real Artificial Intelligence in architecture.

2.1.2 WHY?

Having discussed what the term ‘intelligence’ may stand for in the field of AI, the question arises why people want to create or simulate intelligent behaviour within a machine. In their attempt to make computers perform intelligent tasks, AI researchers can be roughly divided into two groups according to their intentions. For the members of the first group, the computer is first and foremost a powerful problem-solving tool. Their ultimate purpose is to write software that enables a machine to solve hard problems, preferably better than people currently can. A less ambitious objective is to develop machines that provide human problem solvers with intelligent support. By contrast, the intentions of the second group must be situated in the realm of cognitive psychology. According to these researchers, the simulation of intelligent behaviour produces a better understanding of the basic principles underlying (human) intelligence. It is obvious, however, that this division into ‘engineering’ and ‘psychological’ AI is as artificial as the intelligence in the field itself. Depending on the relative importance they attach to each of these goals, many researchers find themselves somewhere in between the two camps, using the computer both as a tool and as a method of inquiry.

2.1.3 HOW?

Minsky’s definition of AI delineates the long-range goals of the domain. Getting a machine to pass the Turing test, however, is far too difficult to have much impact on the daily activities of AI researchers. Most of them therefore confine their ambition to one of the sub-problems the construction of an intelligent computer involves.
CHAPTER 2

In order to identify AI’s sub-fields, the task of making a machine solve problems can be decomposed into several steps. First and foremost, one needs to identify which knowledge is needed to solve the problem. The two next steps consist in selecting an appropriate representation language and translating the knowledge into that language. Finally, the consequences of the knowledge must be drawn on to solve the problem.

The sub-field of knowledge engineering takes care of the first three steps. Researchers in this field study what problem solvers need to know and how this knowledge can be encoded so as to make machines reason with it. They have developed and explored several approaches to answer these questions, one of which is Case-based Reasoning.

A second sub-field concentrates on the final step, i.e. searching a solution to a problem. Within the AI community, search stands for solving a problem by examining a large number of possible solutions [Ginsberg, 1993 pp.10-11]. Playing chess and generating crossword puzzles, for instance, can be viewed as search problems. What makes them difficult is that the space being searched often grows exponentially with the size of the problem. Hence the need for techniques and algorithms that can solve search problems with the limited computational resources available in practice.

While the core of the AI army takes care of knowledge engineering and search algorithms, the remaining researchers have specialised in applications of their colleagues’ ideas. They work on topics as diverse as games, natural language understanding, vision, robotics and – AI’s commercial showpiece – expert systems. Whereas the overall objective of AI research is to equip a machine with general knowledge, developers of expert systems limit themselves to making computers knowledgeable about one particular domain or discipline. Some of the CBD systems that will be discussed in the following chapter can be considered experts in sub-fields of architectural design.

2.2 Case-Based Reasoning: theory, model and application

Although AI researchers do not always understand the term ‘intelligence’ in the same way, and although their intentions may differ, one of their few consensual beliefs is that knowledge constitutes an essential ingredient of intelligent behaviour [Kolodner & Riesbeck, 1986 p.1]. A central issue in AI is therefore to ‘extract’ knowledge from people and put it into a computer so that it can use this knowledge for problem solving.

Whether one envisages machine intelligence in general or in a specific domain of expertise, the basic issues on the agenda are the same: extracting knowledge from people (knowledge acquisition), representing this knowledge in a computer-understandable form (knowledge representation) and making the computer reason with this knowledge (reasoning schemes). This requires insights into how knowledge resides in the human mind and how people make use of it, in other words, it requires a plausible cognitive model. Based on different models of human reasoning, AI researchers have developed several problem solving strategies, of which CBR is a relatively recent one.
WHAT IS CASE-BASED DESIGN BASED ON?

In the traditional view of reasoning, both within AI and cognitive psychology, reasoning is largely a process of remembering abstract pieces of knowledge and linking them with each other [Kolodner, 1993]. This means that knowledge resides in human memory in the form of general, abstract principles such as rules or models, and that people reason by applying the principles that are appropriate for their problem at hand. Strategies based on this traditional view are Rule- and Model-Based Reasoning. The CBR approach, on the other hand, derives from an alternative view of human reasoning. Rather than linking abstract pieces of knowledge, reasoning is seen as a process of remembering one or a small set of concrete instances and basing decisions on comparisons between the new situation and the old instance [Kolodner, 1993]. This cognitive model was inspired by Schank’s Dynamic Memory Theory [Schank, 1982], and in turn inspired AI researchers to model people’s raw memory.1

2.2.1 THE THEORY OF DYNAMIC MEMORY

The Theory of Dynamic Memory derives from a view of everyday understanding as an explanation process: “a process which requires you to construct explanations for behaviour and events that have occurred” [Riesbeck & Schank, 1989 p.3]. At first glance, this would seem extremely difficult: creating an explanation for everything you see and hear cannot but require real intelligence of presumably very complex a nature. This expectation, however, is not confirmed by Schank’s observations of human behaviour: even some very unintelligent people manage to understand the world they live in fairly well, thus suggesting that everyday understanding cannot be that difficult.

2.2.1.1 Scripts
A first hypothesis assumed that people were carrying around in their heads a kind of general field guide, a set of scripts for acting in stereotypical situations, like eating in a restaurant, visiting a dentist, etc. [Riesbeck & Schank, 1989 p.4]. Later, the script idea was expanded to personal experiences: most scripts would be highly personal and not shared amongst individuals. In fact, the guide should be thought of more as a personal notebook that each of us writes and maintains in his own handwriting, with illegible scribbles, mistakes, corrections and redundancies [Kolodner & Riesbeck, 1986 p.1]. Every single situation we have experienced is assumed to be written as a script in our notebook. This extended script idea was then used to explain why people have little problems with understanding: if they have experienced events similar to the current one, the corresponding script helps them to interpret the new situation. If, on the other hand, there is no script that exactly matches the situation at hand, they search the notebook for a comparable event and modify its script into a more suited situation. Hence, the hypothesis about the nature of human thought is that thinking comes down to applying and modifying scripts.

1 By someone’s raw memory CBR researchers mean all the experiences one has had or has heard about. Unlike the general rules that can be derived from them, raw memory contains these experiences in their original form.
2.2.1.2 Reminding

The essence of the whole script story is that understanding does not need to be so complex if it is considered a process of adapting an explanation rather than creating one: explaining something new is possible by modifying an explanation that exists already. The claim is that, given the choice between thinking hard and adapting the explanation of an existing script, people will opt for the script solution every time. Finding in memory the right script at the right time – ‘reminding’ in every day usage – therefore becomes the key to successful understanding.

Yet, if our personal notebook contains a script for every single event we have encountered, the question arises: how do we find the right one? In order to answer this question, Schank’s team has been collecting and analysing remindings for several years. These analyses inspired the hypothesis that remembering, understanding, experiencing and learning cannot be separated from each other. We understand by trying to integrate new things we encounter with what we already know. Thus, understanding causes us to come across old experiences as we process new ones.

The reminding of these previous experiences is not random, however. We remind a particular experience, namely the one closest to the current situation. The reason is that the structures we are using to understand the new experience are the same structures used to organise memory. In other words, understanding can be seen as a process of storing pieces of the situation in relevant places in memory. The key question being ‘where should I store this experience?’, we cannot help but pass through similar old experiences while processing new input.

A side yet significant effect of this process of understanding is that memory never behaves exactly the same way twice, since it changes as a result of its own experiences. If memory has a set of structures, it seems obvious that they cannot stay immutable when new information comes in [Riesbeck & Schank, 1989 p.23]. As experiences and memory structures are recalled and used, memory gets an opportunity to try out the knowledge associated with them. This allows it to re-organise and re-define itself, in other words to learn from its experiences, which accounts for the term dynamic memory [Kolodner, 1993 p.105].

According to Riesbeck and Schank, this learning can take different forms [Riesbeck & Schank, 1989 p.23]. The easiest one is storing new episodes in terms of old expectations generated by previous experiences. Eventually expectations that used to work will have to be invalidated. Indices to unique experiences that were once useful will cease to do so because similar experiences have been encountered. New structures will have to be built.

2.2.1.3 Reminding and problem solving

Once reminding was made central to understanding, as the theory of Dynamic Memory did, researchers began to ask how it might be useful to other cognitive tasks. Observations of people suggested that remembering old situations is central to problem solving too. In fact, people rarely construct solutions to every day problems from scratch: ‘They do not reason about each problem they face as if they had never before faced a problem like it. Rather, they try to find the best plan they have heard or
WHAT IS CASE-BASED DESIGN BASED ON?

previously used that is closest to the problem at hand, and attempt to adapt that plan to
the current situation” [Riesbeck & Schank, 1989 p.5]. Based on these observations,
one could argue that the field guide metaphor is not entirely suited for memory. Such
guide only explains what something is, whereas memory also seems to tell what to do.
Therefore, we should rather think of our personal notebooks as “survival manuals with
a healthy dose of etiquette instructions” [Kolodner & Riesbeck, 1986 p.7].

While reminding previous experiences may play a key role in everyday life, it is
even more obvious in particular disciplines. The most obvious example – and hence
the most cited in the CBR literature – is undoubtedly law. Anglo-American law
systems and schools heavily rely upon previous cases. Students have to learn cases
from the past and to make new decisions by abstracting the essentials of an
appropriate case. Also in architecture, education makes – or used to make – extensive
use of historical and contemporary cases. Architectural history and case studies of
existing buildings traditionally filled a substantial part of the curriculum. Riesbeck and
Schank explain the case-based education in these domains by advocating they are not
based upon systematic principles, or at least do not pretend to be so. Other
professionals, doctors for instance, are not explicitly taught to use previous cases until
late in the curriculum, but often find it an efficient way to make accurate diagnoses
and prescribe effective treatments.2

The conclusion that can be drawn from these and many other observations, is that
when confronted with a problem, remembering previous experiences and using these
memories to solve the problem at hand, is a natural process for people. In this process,
previous cases can play different roles: either they suggest an almost-right solution to
the current problem, or they warn of the potential for failure [Kolodner, 1993 p.77].

2.2.2 THE COGNITIVE MODEL BEHIND CBR

Inspired by these observations of human behaviour, an attempt was made to develop a
plausible cognitive model that could explain this behaviour. There is neither need nor
space to describe this model in full detail here. Instead, we will focus on those aspects
of the model that are particularly relevant for the further discussion of CBR’s
application to architectural design. These aspects are compared with the corresponding
elements of the more traditional model underlying Model- and Rule-Based Reasoning.

2.2.2.1 Structure and organisation of knowledge

A cognitive model should make some predictions about the structure and organisation
of people’s knowledge. The model underlying CBR claims that knowledge resides in
memory in the form of specific events (cases) and generalisations of specific events
and that both are organised in the same way. What is new compared to the traditional
view of reasoning, is that people’s knowledge does not only consist of abstract,

2 It should be noted, however, that things are slowly changing in schools of medicine. At the
University of Maastricht (The Netherlands), for instance, medical students are confronted
with concrete cases from the very start of the curriculum. In fact, several different disciplines
at this university have recently adopted a problem-driven approach to education [Van
Houdenhove, 1998].

47
generally applicable principles, but also of concrete cases. Within the context of CBR, the term ‘case’ stands for the interpreted representation of a real experience, including all details that make the experience individual [Kolodner, 1997].

Although the term Case-Based Reasoning may suggest cases to be the only form of knowledge in memory, there is no such commitment in its underlying cognitive model. In essence, CBR is no more than reasoning from experience, and whether this experience is best called a specific case or a generalised episode is mainly a question of frequency. It is obvious that when an activity is repeated often enough, it loses its specificity and will reside in memory in a more generalised form. Writing each single tooth-brushing session in our notebook, for example, simply would take too much space and effort. Instead, we store a description of brushing our teeth in general that covers all individual experiences.

An interesting aspect of the model is that both specific cases and generalisations reside in the same memory structures. They are organised in the same way, retrieved by the same processes, and therefore equally accessible. Why is it, then, that CBR systems seem to favour cases to neglecting generalised episodes? According to Janet Kolodner, the reasons are more practical than theoretical. First of all, there has already been much research on the role of general knowledge in reasoning, whereas the use of specific experiences is largely an unexplored field. Hence, the CBR community has decided to spend most attention to the role of concrete cases. A second reason is that researchers prefer the systems they develop to make use of the most specific knowledge available, and this is most often provided by cases. Within the AI community, there is a general agreement that ‘strong’ knowledge, i.e. knowledge specific to a particular problem, is more efficient than ‘weak’, generally applicable knowledge. The former requires less inference to figure out how to apply it, which significantly speeds up the problem solving process [Kolodner, 1993 p.14].

It should be noted, however, that the role of generalisations recently gets more attention from CBR researchers. Unlike the earliest ‘pure’ CBR systems, the younger generation reflects the need to integrate both general and specific knowledge by comprising several ‘hybrid’ systems. Hybrid Case-Based Reasoners use more than one knowledge source and/or problem solving approach, for instance causal models in combination with specific cases [Maher, Balachandran & Zhang, 1995 p.36]. Several of the CBD tools discussed in the following chapter, e.g. FABEL and SEED, have opted for such hybrid approach.

2.2.2.2 Reasoning processes
Cognitive models do not only address the form(s) knowledge takes in memory, they also make claims about how people use this knowledge. The model upon which CBR is based, proposes two macro-processes for reasoning with cases: remembering and manipulating.

At first glance, the process of remembering is not particularly new. The traditional view of reasoning already used this process to retrieve general principles (rules or models) that apply to the problem at hand. The difference between the traditional and CBR’s model, however, lies in the fact that the former requires a perfect match
WHAT IS CASE-BASED DESIGN BASED ON?

between the current situation and the knowledge that is retrieved. A rule’s consequent, for instance, is only applied when the problem description exactly matches its antecedent. A case, on the other hand, does not require a perfect match to be a candidate for retrieval. Having something in common with the new situation – also called ‘partial matching’ – may already be sufficient.

CBR’s cognitive model explains this ‘new’ mode of remembering by assuming that the memory processes and structures that support understanding, also serve problem solving. According to the theory of Dynamic Memory, the process of understanding looks for a place in memory to store the current experience, and halts at those locations that seem most appropriate. The knowledge chunks residing in these memory places – cases or generalised episodes – are likely to address a situation that is similar, yet not necessarily identical to the current one, and are available for reasoning.3

A consequence of the partial matching phenomenon in CBR, is that retrieved cases cannot be applied as such. First, they need to be manipulated to make up for the differences with the current problem. Instead of directly applying a rule’s consequent or instantiating the values of a model’s attributes, CBR modifies existing solutions to fit the new situation. Adaptation processes restrict this modification to one single case. Merging, on the other hand, takes suggestions from several different experiences, and combines these into a new solution.

So far the CBR community has spent most attention to developing adaptation methods. Riesbeck and Schank divide them roughly into structural and derivational processes [Riesbeck & Schank, 1989 p.14]. Structural adaptation directly modifies the solution stored in the retrieved case. Derivational methods, on the other hand, re-run the reasoning chain that generated this solution, be it now in the context of the new problem. In other words, whereas the former reuses the end product of a previous solution, the latter recycles the process that generated it.

In many real-world domains, however, solutions are rarely based on a single case from the past. Designers, for example, are said to merge fragments of multiple previous designs, each of which may address another aspect of the design problem at hand. Here a distinction can be made between on the one hand crossing properties of different cases, and on the other hand composing pieces of structures contained by cases [Hua, Schmitt & Faltings, 1992].

---

3 This means that remembering occurs as a natural implication of the process of understanding, in other words, the reminding just described is accidental or unintentional. By contrast, CBR suggests a more intentional form of reminding: facing a problem to be solved, we try to remind ourselves intentionally of a similar situation. However, the CBR community assumes the process of traversing memory to be the same for both intentional and unintentional reminding, the only difference being the source that triggers and directs this traversing: for accidental reminding, the source is simply what one is experiencing; for intentional reminding, it is a kind of hypothetical description of what one is looking for, e.g. a description of the problem to be solved.
2.2.2.3 Learning
Being rooted in the Theory of Dynamic Memory, an essential ingredient of CBR’s cognitive model is the claim that memory is dynamically changing with each new experience.

Thinking of memory as a continually growing trace of experience seems more plausible than considering it a static knowledge base, as the more traditional model does. Both Rule- and Model-Based Reasoning reduce the human mind to a mere repository of general principles. When it uses these principles, this knowledge is not changed, which obviously does not correspond to how a real memory works. People learn from their experiences and using memory itself is such an experience. By consequence, learning is naturally integrated with other reasoning processes.

The model does not only claim that memory changes, but also specifies some supposable sorts of change. The most simple way in which memory improves its behaviour, is by acquiring new cases. Each event we experience is stored in memory, thus furnishing additional familiar contexts for solving problems. Sometimes, however, an experience is not immediately stored in the right place. This causes the case to be recalled while solving a problem for which it appears irrelevant or inappropriate. Hence, a second kind of learning is re-indexing cases already stored. Re-indexing allows memory to fine-tune its recall apparatus so that it remembers cases at more appropriate times in the future. Memory also changes by creating new generalisations. This way of learning derives from the fact that what might start as a new experience, different from the norm, eventually might become the norm. The first time you brush your teeth, for example, it is stored as a unique event. Yet, after say twenty-four teeth-brushings, memory will summarise them into one general description.

2.2.2.4 Generalised knowledge
Unlike what the term Case-Based Reasoning might suggest, the cognitive model behind it predicts several kinds of generalised knowledge that are likely to be found in memory.

First of all, there are general descriptions of particular kinds of situations, such as the summary of twenty-four teeth-brushings just mentioned. The model assumes these descriptions to be organised by the same structures and indexed in the same way as specific experiences.

In addition, the human mind needs knowledge about how to adapt old solutions to new situations. Since future problems cannot be anticipated, only some very limited adaptation knowledge can reside in the cases themselves. Hence the assumption that this knowledge must be of a more general nature [Rosenman, Gero & Oxman, 1992 p.296]. Janet Kolodner admits that, although CBR’s cognitive model predicts the existence of such knowledge, it has fewer claims to make about the structures it lives in [Kolodner, 1993]. More research is needed to understand where exactly adaptation knowledge fits within memory’s structures.
WHAT IS CASE-BASED DESIGN BASED ON?

In contrast to the traditional view of reasoning, CBR apparently requires multiple kinds of knowledge: specific cases, generalised episodes and adaptation knowledge. More types of knowledge, thus, but perhaps knowledge that is easier to acquire.4

2.2.3 APPLICATIONS

Conforming the two purposes of AI research, the ‘engineering’ and the ‘psychological’ one,5 the cognitive model presented here plays a double role. It provides a methodology for developing intelligent systems, and a basis to build upon a better understanding of how people use cases for intelligent tasks.

2.2.3.1 Constructing Case-Based Reasoners

First of all, CBR’s model of cognition serves as a blueprint for the development of artificial Case-Based Reasoners. In the early 1980s, computer scientists started to translate the reasoning processes and knowledge structures of Schank’s Dynamic Memory into computer algorithms and databases, the first steps towards complete CBR systems. If Schank was the (spiritual) father of these systems, Kolodner was their mother. In 1983, one year after the publication of Schank’s book, she developed CYRUS, the very first Case-Based Reasoner. CYRUS included processes for indexing, making generalisations, re-organising and searching memory [Kolodner, 1993 p.120].6 Since then, the CBR family has extended with systems of several kinds, for several purposes and in several domains. Regardless of the purpose or domain it is built for, the core of each system is the case base: a collection of specific cases stored as complete patterns of experiences, and labelled by a set of characteristic features serving as indices. Given a new problem, the most relevant cases in memory are retrieved, thereupon either the user or the system itself manipulates the corresponding solution(s) so as to fit the current situation.

Fully automated CBR systems solve problems completely by themselves. The human user only specifies the problem and, if possible, gives feedback on the solution the system produces. Retrieving relevant cases in memory and adapting them to the new situation, is taken care of by the system itself. In contrast, there are a number of systems that, based on a problem description given, merely provide the user with cases to consider. Often these systems do not have knowledge about cases beyond their indices. The user is responsible for selecting the case(s) to further work with, manipulating the selection and evaluating the result, in short, for decision making. In between these two extremes, a whole range of intermediate systems are possible, leaving more or less responsibility to the human user. Which blend of automation and user interaction is better, depends on the purpose the CBR system is built for. Some

4 In practice, however, the kinds of knowledge predicted by the cognitive model turn out to be insufficient to make a machine reason in a case-based way. For instance, a CBR system also needs a similarity metric, i.e. explicit knowledge about how to compare different cases and how to select the most relevant one.
5 See p.43.
6 She is also co-parent of some of the earliest CBD systems for architecture. Both Archie and Archie-II belong to the Kolodner family (see also chapter 3, pp.61 ff).
system developers want their product to solve problems, others prefer it to only suggest solutions or give general advice. Still others build a database rather than a Case-Based Reasoner, reducing the system’s task to storing and retrieving partially matching cases.

The disciplines for which CBR applications have been developed – and are under development – are as diverse as legal judgment, medical diagnosis, cooking, football and design. Although CBR is a relatively young member of the AI family, a surprising number of these applications are already commercially successful [Watson, 1997 p.61]. More than 130 companies are using CBR for a variety of tasks, daily demonstrating its power to resolve fundamental problems in AI research. It should be noted, however, that CBR itself is not a solution or an algorithm. As Christopher Owens contends, it is no more than “a methodological approach whose success depends upon the solution of several problems in the area of memory” [Owens, 1988 p.310]. Indeed, proposing memory structures, reasoning processes, kinds of learning and types of knowledge – like CBR’s cognitive model does – is one thing, consequently implementing them into a machine is another.

2.2.3.2 Investigating human intelligence
Pursuing the more psychological ambition of AI, some CBR systems are developed as a test bed to simulate and investigate how exactly people utilise cases when performing intelligent tasks. There are several reasons why such investigation of human intelligence may be useful.

First of all, a more profound understanding of the role cases play in intelligent tasks would allow us to improve human problem solving. When confronted with a problem, remembering previous experiences and using these in the situation at hand, seems to be a natural process for people. Natural though this process may be, however, it is not always plain sailing [Kolodner, 1993]. Sometimes people reuse a previous solution blindly, without checking its appropriateness for the current situation. Moreover, they are not always reminded of the most relevant case when facing a new problem. Especially novices suffer from lack of experience(s) to solve problems in a case-based way. A better insight into which parts of the reasoning process are difficult or problematic, may be useful to help people reason better.

The observation that people do not always recall the right case at the right time, has led to the development of interactive Case-Based Aids. Situated at the low end of the automation line, these systems pay tribute to the principle of man-machine complementarity. Following this principle, tasks are divided between computer and human user corresponding to their respective qualities, so that they can complete each other in an optimal way [Neuckermans, 1994 p.caad-5]. As to the processes of CBR, computers tend to be good at storing and retrieving cases, yet when it comes to

---

7 The focus of this thesis being the role of cases in conceiving architecture, some CBR systems for the field of architectural design will be discussed in more detail in chapter 3.
8 As chapter 6 will point out, this implementation causes serious problems when developing CBR systems for architectural design.
9 See also p.46.
WHAT IS CASE-BASED DESIGN BASED ON?

creative adaptation or judgment, they are usually outperformed by people. By consequence, developers of Case-Based Aids consider the computer as a mere storage and retrieval tool for augmenting people’s memory: the machine holds a repertory of cases and makes the right case(s) available at the right time. Making decisions based on these cases is completely left to the human user.\(^{10}\)

Secondly, a better insight into people’s CBR process would enable the development of teaching strategies and tools that employ cases more effectively. As Kolodner contends, “If people are comfortable using examples to solve problems and know how to do it well, then one of our responsibilities as teachers might be to teach them the right examples and effective ways to index them” [Kolodner, 1993 p.28].

At first sight, this educational exploit of CBR research does not necessarily require the deployment of computer technology. Schools of law and architecture adopted a case-based approach to teaching long before computers were even thought of.\(^{11}\) Also in other kinds of schools, teachers have always been using cases, usually to illustrate the subject matter and make facts more compelling and easier to remember. Within the field of CBR, however, some researchers got convinced that computers can help capitalize on students’ natural case-based learning, and started to develop Case-Based Teaching systems. These systems should supply students with appropriate cases representing situations they will encounter in professional life [Kolodner, 1993 p.40]. Examples of Case-Based Teaching systems are Dustin, a tool for language learning [Ohmaye, 1998], CreANIMate, which teaches children about biology [Edelson, 1998], SPIEL, a Story Producer for IntertactivE Learning [Burke, 1998], and PRECEDENTS,\(^{12}\) which teaches student-architects concepts of museum design [Oxman & Oxman, 1994a]. Rather than replacing human teachers, these systems aim at providing an educational environment that allows teachers to change their role from knowledge supplier to student monitor.

2.3 Case-Based Design: premise and promises

Starting from the vast domain of AI, we have first zoomed in on the sub-field of knowledge engineering (section 2.1), and secondly on one of the approaches developed within this field, i.e. Case-Based Reasoning (section 2.2). The third and final step is now to focus on the application of CBR to design, or in short, on Case-Based Design. In order to do so, this section starts with the basic premise underlying this application, and subsequently sketches the promises it holds for the field of architectural design. These promises have inspired the development of various CBD systems, a selection of which will be reviewed in the following chapter.

\(^{10}\) Examples of Case-Based Aids for architectural design are Archie-II and PRECEDENTS. They are developed to assist (student-)architects with designing public buildings like courthouses and museums, and will be discussed in more detail in chapter 3.

\(^{11}\) See also p.47.

\(^{12}\) See also chapter 3, pp.83 ff.
2.3.1 PREMISE

Case-Based Reasoning is only one of the strategies AI researchers use to make machines perform intelligent tasks.\footnote{For a comparison of the advantages and disadvantages of the different problem solving strategies used in AI, see [Heylighen, 1996 pp.5-12].} Which strategy is most suitable largely depends on the domain or task the system is built for. Some domains or tasks are so well understood and defined, that a small number of rules already suffice to reason about them. In that case, Rule-Based Reasoning seems to be the most appropriate approach to use. Other tasks, such as devising electrical circuits, are highly technical and require the kind of causal reasoning a Model-Based strategy supports. Still others are so poorly understood or defined that cases provide the only reliable source of inferences. For such ‘weak-theory’ domains CBR seems to be a more reliable approach than Rule- or Model-Based Reasoning [Kolodner, 1993 p.74].

According to the CBR community, architectural design unmistakably falls into the latter category. Since problems in this field are typically ill-defined and repeatedly re-defined during the design process,\footnote{See chapter 1, pp.13-14.} they cannot be solved by a fixed set of rules or causal model. Therefore, it seems plausible to assume that architects rely heavily on past design experience. Especially during the early, conceptual stage of design, previous design cases would provide grist for a number of decisions to be made [Domeshek and Kolodner, 1997]. In pursuit of a suitable concept, architects would search their memory for past design experiences that may be relevant, and use these to start in on the situation at hand. This very premise has laid the foundation for a substantial body of research, which fosters CBR systems specifically tailored for architectural design.

At this point, one may ask why AI researchers did become interested in architectural design at first. Ulrich Flemming sees two reasons for this interest. In general, AI is concerned with making machines do things that appear to involve some intelligence if done by humans, which is obviously the case for designing architecture. Moreover, unlike traditional areas of AI research like natural language understanding or vision, design tasks "are done consciously by designers, who may be queried about the structure they impose on a task and from whom specialised domain knowledge can be extracted" [Flemming, 1994; italics added]. If that was really the belief of AI researchers, they must be seriously disappointed by now. For conscious as architectural design may be, it is to some extent also intuitive, expressing impressions, ideas and thoughts which architects have unknowingly collected on a subconscious, unconscious or preconscious level [Papanek, 1984]. Whatever the reason may be, architectural design has definitely acquired a particular position among CBR’s fields of interest. It is believed to be one of the most interesting and challenging activities ever facing CBR and AI researchers.
WHAT IS CASE-BASED DESIGN BASED ON?

2.3.2 PROMISES OF CBD FOR ARCHITECTURE

Apparently, the field of architectural design furnishes CBR researchers with an appealing playground for experimentation. Yet, what do they have to offer in return? According to the CBR community, the application of CBR to architectural design holds great promises for both practice and education in this field.

1. CBD systems can bypass the knowledge acquisition bottleneck of Rule- and Model-Based approaches to equip CAAD tools with design knowledge.

We already mentioned how difficult it is to capture concept knowledge in a general format. Very few rules or models hold over all aspects of a design. Yet, even at component level, general knowledge is difficult to acquire. Given the numerous aspects a building must address, and the intuitive way in which many design decisions are taken, an architect can impossibly enumerate all component knowledge he makes use of during design. Encoding this knowledge in a general format would be impractical anyway, because parts of it are very subjective and only apply to one specific design. Yet, while architects may have trouble telling knowledge engineers which knowledge they rely on during design, they usually have no problem telling stories about previous design experiences, which can be represented as cases. Unlike rules, design cases capture an informal understanding of the domain at a concrete level, and thus can act as alternative encapsulations of design knowledge. Being equipped with a collection of such cases, CBD systems are able to produce complete and complex designs without requiring an integral domain model [Hua & Faltings, 1993 p.135].

2. Given the fact that design problems are difficult to decompose, “cases can provide the glue that holds a solution together” [Kolodner, 1993 p.82].

The second promise CBD systems seem to hold, is related to the fact that design problems in architecture are what Kolodner calls ‘barely-decomposable’. As already mentioned, buildings have to fulfil a whole series of different, sometimes even contradictory requirements. It is quite impossible, though, to design a building by decomposing it into sub-problems according to these different aspects, solving each one separately and recomposing the sub-solutions into a final design. For just like the components in complex systems, the requirements of a building interact very strongly, implying that the sub-solution for one aspect is likely to have implications for other aspects’ sub-solutions. Even if a requirement interacts with only a few others, the effects of the corresponding sub-solution are spread throughout the whole design. Therefore, all sub-solutions would necessarily be preliminary and subject to change as long as there are other un-addressed aspects whose solution may affect not only the local neighbourhood, but even principal design decisions made early in the design process [Bakhtari & Bartsch-Spoërl, 1994 p.761].

15 See chapter 1, p.32.
16 See also chapter 1, p.35.
17 See also chapter 1, p.9.
It is obvious, then, that every decomposition of a design problem is doomed to be an impoverished reproduction of the real problem. In fact, a design problem with its complex, mutually related aspects, functions and requirements, can be viewed as a Gestalt, i.e. “a whole in which the nature of the parts are fused and interdependent interacting in a specific structural manner” [Wertheimer, 1969; referred to in Neuckermans, 1994 p.dec-4]. Designing such a whole is extremely difficult, because it has to integrate satisfactory solutions to each of these requirements.

We already mentioned how difficult it is to achieve such integration by applying general rules or models. Since few generalisations address all aspects of architectural design, formulating design knowledge in an integrated way almost always comes down to citing concrete design cases.¹⁸ Being themselves integrated solutions to specific design problems, cases can provide the glue that holds the different parts of a design solution together. By starting a new design from one or more existing cases, CBD systems implicitly achieve trade-offs among the several issues a building must address.

3. **CBD systems allow to increase the efficiency of architects’ design process.**

   The value of rapid, effective and creative design is becoming ever more apparent in today’s global economy [Domeshek & Kolodner, 1992 p.497]. In the practice of architecture, however, many architects design every building as if it were the first in kind, coolly ignoring the design efforts of their experienced predecessors. According to Gerhard Schmitt, “without being based on an extensive body of case knowledge, architectural design will continue to be reinvented by the majority of architects who cannot build on their own experience. No other discipline can afford to operate this way with every artefact it designs” [Schmitt, 1993 p.19]. Indeed, imagine the car industry designing every new car model from scratch.

   Making use of previous design cases during the design process could help architectural design out of its current inefficiency: “Case-based design presents a shortcut for the generation of a new design: it starts from an entire previous design solution, saving the time necessary to create one from scratch, and adapts the pieces of the retrieved design that do not fit the new situation” [Maher, Balachandran & Zhang, 1995 p.30]. The fundamental assumption underlying this promise, is that such adaptation takes less effort than generating a design ‘ex nihilo’. According to some, this assumption only holds if the retrieved design case is sufficiently close to what is needed in the new situation [Hua e.a., 1992 p.574]. Only under this condition, the previous design efforts encapsulated by the case can be used to shortcut designing from scratch. By assuming that the old solution is almost correct for the new context, major revisions of the decisions are not made. The most enthusiastic CBR researchers, however, believe CBD systems to be efficient even when the old case is far from feasible a solution to the new problem. Given the strong interaction between the different ‘components’ of a building design, the large amount of adaptation knowledge needed in this case would still be weighed out by the effort to develop a completely new design.

---

¹⁸ See chapter 1, p.32.
WHAT IS CASE-BASED DESIGN BASED ON?

4. **CBD systems can extend the memory of architects – and especially of architecture students – by storing and retrieving information on design cases they may not have thought of themselves.**

Unlike in legal practice, where cases are recorded for institutional reasons, there is no systematic storage of design cases in architecture [Tzonis & White, 1994 p.27]. CBD systems could alter this situation. They may serve the field of architectural design by storing and giving access to information on large numbers of existing buildings, saving the design experience they represent from being forgotten. Both successes and failures are worth storing in such a thesaurus, the former as suggestions of ways to follow, the latter as anticipations to future problems.

Having at its disposal such a rich source of past design experiences, a CBD system might surprise the user by retrieving treasures he has not remembered on his own. There are several reasons why an architect may not have thought of a particular design solution. The current problem may lie somewhat outside his proper area of specialisation, he may just have a bad day, or perhaps he is a student and thus still novice in the field. Unlike expert designers, novices do not yet dispose of a reliable repertory of heuristics, since such repertory can only be accumulated by confronting many cases through many years of practical design experience. Awaiting these years, CBD systems can provide novices with a substitute for the experience they are lacking so far. In this respect, they are particularly promising for architectural education, as they can extend students’ memory with design experiences they may not have had themselves.

5. **CBD systems are able to learn from their own design experiences without major reprogramming.**

Machine learning occurs when a computer program improves its performance by learning from its own operations. Rule- and Model-based systems are quickly outdated because they have no means to update their knowledge by themselves. By contrast, a CBD system can increase its knowledge and improve its skills with every design problem it solves, simply by storing the design solution into the case base. If the user is actively involved in the adaptation of the old case and/or storage of the new one, as in Case-Based Aids, AI purists will argue that there is no machine learning in the strict sense of the word. Yet, actually, this may be rather beside the point here. The important thing is that CBD systems do not need major reprogramming to maintain a reasonable competency in architectural design. Simply storing newly generated cases will do.

Save acquiring new design cases, CBD systems can also perfect their performance by re-indexing cases already stored. Since generalising always implies the danger of losing some important information, rules, models or types are necessarily impoverished versions of the original design experience from which they have been

---

19 The term ‘thesaurus’ has itself an architectural origin, meaning a treasury or a storehouse [Steadman, 1994 p.301].
20 See also chapter 1, p.16.
21 See also p.50.
derived. Analyses are not guaranteed to be correct and the resulting generalisations never perfect. Once design knowledge is compiled into general principles, it remains compiled forever. In contrast, concrete cases, which contain experiences in their original, detailed form, can always be re-analysed, re-interpreted and re-understood. This allows CBD systems to fine-tune their recall apparatus, and thus to update their knowledge content continually.

2.4 Summary

By way of introduction to Case-Based Design, this chapter has taken a short excursion into the domain of AI, in which it is historically rooted. After zooming in on the sub-domain of knowledge engineering, we have taken a look at Case-Based Reasoning, a theory and technology developed within this sub-domain. CBR, as we have pointed out, derives from an alternative view of cognition that was inspired by the Theory of Dynamic Memory. Finally, we have focused on the application of Case-Based Reasoning to design, or in short, on Case-Based Design. We have discussed the basic premise underlying this application, and outlined the promises it holds for the field of architectural design. In essence, the advantage of using cases derives from the fact that they contain experiences in their original form, which makes them much richer and vivid than general rules or models. Yet, if concrete cases are so much richer than generalisations, it could be argued that a detailed description of a few concrete CBD systems would be much more useful than a general discussion of the promises they hold. It would be ironic indeed to discuss CBD on an abstract level only. Therefore, the following chapter will put some specific CBD projects under the microscope.
A CASE BASE OF CASE-BASED DESIGN TOOLS FOR ARCHITECTURE
CHAPTER 3

The previous chapter has given a general introduction to Case-Based Design, situating its roots in AI and CBR research and outlining the promises it holds for architectural design. Switching from a general to a more concrete perspective, this chapter has opted to put some specific CBD projects under the microscope. It is so to speak a case base of CBD systems for architectural design, and contains seven quite different cases of CBD research: Archie-II, CADRE, FABEL, IDIOM, PRECEDENTS, SEED and WEBPAD.¹ The systems described were selected because of their special concern with the field of architecture, and because taken together they give a fairly good overview of the various directions in CBD research. They represent research from teams of different sorts and sizes, in different contexts, and at different stages of development and implementation. This may explain why the level of detail and length of the case studies varies considerably from one system to another.

Each case study starts with a brief introduction into the main objectives and focus of the CBD system and subsequently runs through the different ingredients of the CBR recipe – case base (content, representation and memory organisation), retrieval and manipulation – describing how (and whether) these ingredients are implemented in the system at issue. It also provides a selective bibliography of references that have proven particularly helpful in understanding the system. A great deal of work in this area has been published in various journals and conference proceedings. To our knowledge, however, this work has hardly been subject to any evaluation, critique or discussion.² Nevertheless, a discussion on these systems is certainly worth developing, and is therefore initiated at the end of each case study. We will continue this discussion in more general terms in chapter 6.

Since there is no significance to the order in which the CBD systems are studied, the case base is organised in an alphabetical way. Readers may wish to read it from A to Z (or rather from A to W), follow a particular CBR ingredient through the different systems, or simply browse from one case study to another.

Our case base of CBD systems does not claim to be complete or definitive. It could not be so since the systems it contains may be further developed and new systems will most likely enter the CBD scene. Therefore, rather than a comprehensive catalogue, this chapter should be read as a snapshot of current CBD research.

¹ The case base builds further on the collection we set up in [Heylighen, 1996]. The Galathea project [Arlati, Bottelli & Fogh, 1996], which will come up in chapter 6 (see p.145 and p.150), was not included in the case base because we hardly found any information about its implementation.
² For instance, Mary Lou Maher and Andrés Gomez de Silva Garza give an overview of several CBD-systems, both within and outside the field of architecture, but hardly go beyond a neutral description of their features [Maher & Gomez de Silva Garza, 1997].
3.1 Archie-II
Archie-II is a Case-Based Design Aid for architects developed at the AI lab of Georgia Tech’s College of Computing in collaboration with members of Tech's College of Architecture. As its name suggests, the system descends from an earlier CBD Aid called Archie, which was one of the earliest CBR applications in the field of architectural design.

Archie-II supports architects during the early conceptual stage of building design, more specifically of public building design, by providing them with interesting design cases from the past. By allowing both successes and failures into the case base, cases in Archie-II may play several roles: some of them propose solutions, others warn of potential pitfalls, and still others suggest evaluation criteria. The system focuses on case representation and retrieval, leaving manipulation completely to the user.

The success of the Archie-II project has inspired the creation of DesignMuse, a general shell for constructing CBD Aids in other complex domains of conceptual design, such as initial design of aircraft subsystems.

3.1.1 CASE BASE
Archie-II contains the records of several courthouses and libraries. Existing descriptions of these buildings – blueprints and specifications – are augmented with evaluations of how the designs turned out. These evaluations were collected by spreading surveys across several stakeholders of the building. Both success stories and negative opinions are stored, respectively suggesting ways to follow or to avoid.

3.1.1.1 Design case
Design descriptions including the details of something as large as a courthouse or library are too extensive to directly assist architects’ design decisions. Adding evaluations swells cases even further, yet, at the same time, provides a key to break cases into more manageable chunks. Obviously, not every part of a public building is equally interesting. Useful evaluations focus on those features that are exceptionally good or particularly bad, and thus worth storing for the future. Describing these interesting parts by short pieces of text, Archie-II ends up in slicing its cases into stories. Three types of stories can be distinguished: point stories, interaction stories and cluster stories.

Point stories describe how certain features of a design (e.g. separated entrances) contribute towards, or undermine some particular goal (e.g. privacy). Some of them tell about features that did not work and what could be done to fix them, others about features that are particularly, and perhaps unexpectedly successful. Interaction stories discuss how features of a design case can be interpreted with respect to several design goals (e.g. privacy, security, circulation), perhaps advancing some while frustrating others. They provide access to underlying point stories, just as point stories provide access to covering interaction stories. The third class of stories, cluster stories, are mainly a browsing convenience. They serve as a table of contents by summarising several point stories that are located close to one another, for example, all stories about a particular room.
Apart from stories, Archie-II contains a second type of knowledge chunks, called guidelines. By generalising groups of particular experiences, these guidelines propose ways of thinking about a design, rather than provide absolute answers to design problems.

The version of Archie-II described in [Domeshek & Kolodner, 1992] contains several dozen stories and a similar number of guidelines. They are linked into a network of examples and counter-examples, generalisations and specialisations.

3.1.1.2 Case representation
For the discussion of how these knowledge chunks are represented, we refer to a screen shot of Archie-II’s interface (figure 3.1). The entire screen is divided into eight major frames.

The dark frame in the upper left corner is the mode control panel. From the system’s five possible modes – library, description, plan, notebook and history – the user has currently selected the plan mode.

The label and contents of the second control panel, the frame beneath the mode control, depend on the selected mode. In this screen shot, it provides the user with facilities to browse through building plans as in CAD packages (e.g. ‘pan’ or ‘zoom’).

The contents of the large central frame also change with the selected mode. In the plan mode, it displays the floor plan of a particular building with some annotated cluster stories. Should the user select the library mode, the central frame would show a customisable table summarising information about the buildings in the case library.

---

3 Note that this corresponds to the cognitive model underlying CBR, according to which human memory contains both specific and generalised knowledge (see chapter 2, pp.47-48).
The buttons in the dark upper right corner provide the user with some basic system options. Apart from the indispensable help and exit buttons, there is, for instance, a button to request a memory search. The ‘interests’ frame below allows users to specify search criteria. At this moment, the user is interested in stories about how the circulation system affects segregation of building zones. Based on this combination of interests, the system has selected the annotated stories in the main frame.

The story under consideration covers the dark region of the plan, and is displayed in the left frame at the bottom of the screen. In addition to the complete story text, the story frame contains the title and source of the story, as well as a set of links to related stories and guidelines, the latter being displayed in the right bottom frame.

Finally, stories and guidelines may be illustrated in more detail in the square section between story and guideline frame. In the screen shot shown, the section contains an abstract version of the courthouse floor plan that highlights the different zones and circulation paths of the story at issue.

3.1.1.3 Memory organisation
A useful scale version of Archie-II will contain a huge number of knowledge chunks – stories and guidelines – in which the user should find his way easily and quickly. Hence the need for an indexing system that allows retrieving the right case at the right time, or rather the right chunk of the right case at the right time. Archie-II distinguishes between two kinds of indices.

Descriptive indices identify design stories by means of a design issue plus at least one of the following four features: building space, functional component, stakeholder perspective and phase in the building’s life cycle. For example, a story may address safety in the courtroom (design issue and building space), efficiency of the air conditioning (design issue and functional component), or the effect of surveillance cameras on prisoners’ privacy in the holding cells (functional component, stakeholder perspective, design issue and building space).  

Relationship indices, on the other hand, function like pointers embedded by authors directly in their texts – like ‘See section 3.1.2 below for a more detailed discussion of retrieval’. In essence, a relationship index is a hard link between two presentations that directs the user’s attention to related material. In Archie-II, bi-directional links can be established between an interaction story and its underlying point stories, between a guideline and point or interaction stories that illustrate its application or failure, or between a cluster story and the point or interaction stories it summarises.

---

4 Since interaction stories discuss how several design issues are differently affected by some feature of the design, their index should contain more than one design issue. Archie-II solves this problem by descriptive indices with two parts, each mentioning a different issue and outcome.
CHAPTER 3

3.1.2 RETRIEVAL

Archie-II offers two possibilities for retrieving relevant cases. Descriptive indices support directed search, while relationship indices allow the user to browse through related stories and guidelines.5

3.1.3 MANIPULATION

At this point, the task of Archie-II is finished. The user bears the complete responsibility for understanding and applying (or ignoring) the information presented.

3.1.4 DISCUSSION

An interesting idea of Archie-II, in our opinion, is to use evaluations of existing designs as source of design knowledge. Currently, the usefulness of feedback for future projects is often neglected in architectural design. Making evaluative material available to an architect early in the design process can make him more aware of the downstream implications of his decisions. Design decisions have consequences for people who will carry out, use and maintain the building. By storing stories of all these people, Archie-II draws the architect’s attention to all life-cycle implications of his design early in the design process. Collecting such stories seems less intrusive than demanding architects to record the justifications of their decisions during design. When building users are proud of the way something works, they are happy to talk about it, and when something bothers them, they are quick to complain.

A second asset of the system is its attempt to fit an architect’s designerly way of thinking. Designers do not consider the different aspects of a design separately, but always in relation to other issues.6 When designing holding cells, for instance, an architect will not dwell on the problem of safety first and reflect on privacy afterwards. All these considerations are running through his head simultaneously. Archie-II’s descriptive indices attempt to support this kind of thinking, by allowing the user to search information on several aspects of a design at the same time.

Furthermore, the sophisticated indexing system enables Archie-II to follow an architect’s ‘juggling’ form one design aspect to another.7 For example, he may start to investigate acoustic issues in the courtroom, subsequently move to another type of room, then decide to concentrate on the heating and air-conditioning in that room, and so on. Thanks to the combination of descriptive and relationship indices, the system can manage such capers in the user’s mind fairly well.

Archie-II also shows some shortcomings. The complex indexing system, for instance, makes it difficult to add, remove or rearrange cases or stories in the case base. Unlike the

---

5 Browsing means looking for information without an a priori goal, whereas directed search assumes an explicit goal to be known beforehand.
6 See chapter 1, p.9.
7 See chapter 1, p.13.
expectations of the CBD-pioneers, the system cannot be updated by ‘simply’ storing new cases or reinterpreting cases already stored, as promised by the CBD-pioneers.  

A second drawback may be that Archie-II is too much tailored to public building design. The method of knowledge acquisition – post-occupancy evaluation – is rather common for some categories of institutional buildings, but may be less appropriate for other building types.

3.1.5 REFERENCES


---

8 See chapter 2, pp.57-58.
3.2 CADRE

CADRE – which, depending on the source, stands for CAse-based spatial Design REasoning [Schmitt, Faltings & Smith, 1994] or CAse-based building design through Dimensionality REduction [Hua, Smith & Faltings, 1994] – is a CBD system for preliminary building design. It is an interdisciplinary project involving researchers from Architektur und CAAD (ETH Zürich), Steel Structures and the Artificial Intelligence Laboratory (both at EPF Lausanne). The focus of this research has been on the adaptation of design cases to new environments and on the integration of interdisciplinary design criteria. This case study discusses a specific design problem with which the system has been tested, namely the adaptation of an existing residential apartment design to a slightly different site.

![Figure 3.2. Mario Campi’s Felder House before and after adaptation.](image)

3.2.1 CASE BASE

As to CADRE’s content, we should perhaps speak of a case rather than a case base. The system as described in [Hua & Faltings, 1993] contains all in all one design case, namely the Felder House in Lugano, a U-shaped residential apartment designed by Mario Campi (see figure 3.2).

According to the CADRE team, the main point in using cases is to avoid having to formulate knowledge explicitly. By consequence, they limit their research to cases that are as shallow as possible. Yet, although no information on the design process or outcome has been added, CADRE’s case has nevertheless a certain degree of depth. As the following section will point out, the description of the design solution has been augmented with domain-dependent knowledge for supporting the adaptation process.

---

9 Within the context of CBR, a distinction is made between shallow and deep cases. Shallow cases only contain a design solution, i.e. the description of a design product. Deep cases supplement this description with the design history, i.e. information regarding the design process such as decisions, their justifications or knowledge used in making them.
3.2.1.1 Design case
CADRE’s case is a complete building design. It is thought of as an integration of different functions according to three abstractions – spaces, circulation and structure. These abstractions are automatically derived from a geometric description of the building by the pre-processor Mod-4. First, the user describes the design by inserting walls, doors, windows, etc. together with room labels, material and construction information. The environment can be specified by contextual elements like parcel lines, neighbouring buildings, lakes and parks. Subsequently, Mod-4 converts the user input into an AutoCAD model and derives additional graph descriptions for the three abstractions mentioned. As to spaces, for instance, the system automatically identifies topological relations between rooms by analysing the model and its labels. Labels and positions of walls, doors and windows are used to deduce a graph encoding the required spatial adjacencies. In this way, Mod-4 derives the different abstractions of the building that are needed for later reasoning.

The case is further augmented with two sorts of adaptation knowledge, corresponding to the two types of adaptation CADRE supports. Dimensional knowledge takes the form of constraints on the dimensions of the design, for instance ‘width > 1.5m’. These constraints originate from criteria related to space, circulation or structure, and control the parameters of the AutoCAD model. In this way, the separate abstractions are mapped into a single model so as to maintain integration between them. Topological knowledge, on the other hand, exists in various forms. Examples are transformation rules that reduce the number of rooms if their dimensions fall below a prescribed minimum, or string grammars encoding desirable features of the design that must be preserved during adaptation.

3.2.1.2 Case representation
Users can choose between several representations of CADRE’s case: a geometry-based representation (the AutoCAD model), topology-based representations (e.g. a graph of adjacency relations between spaces) and grammar-based representations encoding the design’s desirable features. These representations may be stored in the case base or computed on demand.

In addition, users can choose between four levels of detail: ‘all details’ shows all information available about the design; ‘aggregate’ displays clusters of spaces (e.g. all spaces that are public, private or related to circulation); ‘centre lines’ represents the underlying geometry of the design without elements like walls and columns; and finally ‘bounding box’ gives the smallest bounding volume.

---

10 The term ‘function’ is here used to denote any feature of a building design. Structural stability, for instance, is also a function.
11 The U-shapedness of the Felder House, for instance, can be described as the string mnrmmrrmm, in which m means placing a square and then moving forward, and r stands for placing a square and then turning right.
3.2.1.3 Memory organisation
According to the CADRE team, indexing is not specific to architectural design. Therefore they have decided not to address this problem, and to concentrate completely on case adaptation.

3.2.2 RETRIEVAL
In order to adapt an existing building design to a new environment, the user first needs to specify the present problem context, which consists of a site description and/or some programmatic requirements.

The site description is processed in the same way as a design case. The user submits a geometric model of the site (including boundaries, location, orientation and other context specific information), which is subsequently transformed by the pre-processor Mod-4. If, in addition, the case must undergo programmatic changes (e.g. three bedrooms instead of two in the original design), the user has to specify desired sizes, areas, proportions and adjacencies of spaces, search depth (which spaces are to be allocated), and the like.

The resulting problem specification, however, does not serve as probe for case retrieval, but is only used during adaptation. Since CADRE focuses on adaptation, retrieval is left entirely to the user, who decides which case from the library will be adapted to the current design situation.

3.2.3 MANIPULATION
Adapting a design case to a new situation requires changing the structure, while preserving the integration of abstractions achieved in the case, as well as the quality for which it was selected. CADRE divides this adaptation process over two separate layers, a dimensional and a topological one.

3.2.3.1 Insertion
Adaptation starts with inserting the selected case into the new environment in such a way that a maximum coincidence with the original situation is achieved. To calculate this coincidence, CADRE makes use of positive and negative relations between spaces and contextual elements. For example, a living room facing south, a park or a yard is weighed positively. The case is checked against its original site first, and if positive links are found, their weight is doubled. The weight of negative links is considered less significant, and therefore divided by two. Subsequently, CADRE rotates and mirrors the design until the sum of all links between its spaces and the contextual elements of the new site has reached a maximum.

3.2.3.2 Dimensional adaptation
If, after insertion, dimensional discrepancies arise, CADRE activates the dimensional layer first. Dimensional adaptation changes values of numerical parameters that describe the geometry of the design, without removing or adding spaces or elements.

For this purpose, CADRE detects the discrepancies with the new situation, and converts them into a set of parameters. Discrepancies occur when differences between the
specifications of the case and of the new problem cause constraint violations. The building may, for instance, exceed the boundaries of the new site, or provide too little floor space. The parameters involved in the violated constraints, plus parameters related to them by other constraints, need modification in the new solution. Since their number can be extremely large, the parameter set is first reduced by dimensionality reduction.\textsuperscript{12} Though it may take hours of computation, the results of this process are quite impressive, as the thousands of building parameters shrink to a manageable handful. For this handful, a feasible value combination can be found either by optimisation methods or simply by user interaction.

In the event that all remaining constraints can be solved, one or more design solutions are obtained and displayed. If, by contrast, some constraints cannot be satisfied, or the user does not like the result, he can either select a new case for adaptation (provided the case base contains more than one building design) or trigger topological adaptation.

3.2.3.3 Topological adaptation

Topological adaptation alters the topology of a case by adding, suppressing or rearranging spaces or elements. For this kind of modification, CADRE’s developers have tested several approaches.

The first approach applies case-specific transformation rules to the topological graph of the case. Such rules try to satisfy the needs of the new problem by relaxing constraints on the (reduced) parameter set. A disadvantage of this approach, however, is that the number of rules grows rapidly and degrades the overall performance of the system.

A second approach uses string grammars to formulate knowledge about spatial adjacencies in certain building types. The idea behind grammars is to identify the most characteristic elements of a design case, and parse this information into similar, yet different combinations that preserve relations between elements as much as possible.\textsuperscript{13} The CADRE team had chosen string grammars for their computational clarity, yet soon discovered the difficulty to describe desirable features of spatial arrangements with them.

Preference is therefore given to the third approach, which starts by removing the spaces that are to be reallocated. Based on the alternatives this removal creates, spaces are inserted again one after the other. In case of multiple possible solutions, the user is prompted to select one configuration for further exploration.

If topological adaptation yields a feasible solution, it is followed by renewed parameterisation and dimensionality reduction, in order to ensure that the new topology meets dimensional requirements. If not, the selected case is declared unsuitable for the new situation, and the user is asked to pick another one from the case base.

\textsuperscript{12} Dimensionality reduction is a mathematical process that determines the variability of a dimensional constraint system. In general, a space of n parameters with m independent equality constraints reduces to (n-m) non-conflicting parameters by algebraically solving the constraint equations.

\textsuperscript{13} The idea is similar to, and probably inspired by writing software, whereby source code is compiled into object code, which in turn is assembled into machine code.
3.2.3.4 Evaluation and visual inspection

After adaptation, a geometric model of the resulting design is displayed and evaluated. When the user is not satisfied with the result, the adaptation process starts all over again.

3.2.4 DISCUSSION

CADRE is obviously a totally different kind of CBD system than Archie-II. Instead of providing designers with cases that may be relevant for new design situations, CADRE’s job only starts after a case has been selected.

When cases are to be manipulated by the system itself, they must be represented in a computer-readable form. An interesting aspect of CADRE is that this representation can be generated automatically by the pre-modeller Mod-4. The price of having to use computer-readable representations, however, is that essential characteristics of architecture are lost, simply because they refuse to be encoded in 0s and 1s. By representing design cases in a computer formalism, architecture is necessarily reduced from a meaningful form to an abstract, geometric shape.14

The emphasis on geometry also appears in the problem the developers chose to test their system, i.e. adapting a building that exceeds the dimensions of a site. Rather than by problems architects often face, the choice seems to be motivated by CADRE’s capabilities. Some hotel companies tend to use the same design for totally different sites, yet it is highly questionable whether this practice should be stimulated, let alone supported by a CBD system. Starting from a complete building makes it extremely difficult to respond to the specific context of the new design situation. Such response requires more than rotating and mirroring until the building fits within the perimeter or the living room faces south. In this regard, CADRE’s topological adaptation cannot but remind us of the space allocation techniques in the 1960s.

On the other hand, an interesting aspect of CADRE is the integration of CBD with a CAD package. This allows architects to consult the system during their normal design activities, without having to switch to another program. Such integration, however, might cause unmeant legal problems. Providing a complete CAD-model of an existing building may tempt an architect – in want of time, for instance – to copy the entire building without major changes. At best, Mario Campi might be flattered by one such copy, but will he still be after the tenth one?

3.2.5 REFERENCES


---

14 See also the discussion on case representation in chapter 6, pp.150-151.
A CASE-BASE OF CBD TOOLS FOR ARCHITECTURE


3.3 FABEL

FABEL is a joint research project conducted by a consortium of six partners led by the German National Research Centre for Information Technology (GMD). The main objective of the project is to support architects and civil engineers in planning buildings with complex installations. Although FABEL combines several AI approaches, and thus is actually a hybrid system, the central paradigm for its specification and implementation was Case-Based Reasoning.

FABEL is conceived as a collection of different tools and methods, called specialists, each of which addresses one step of the CBR cycle (for instance, retrieval or adaptation). They work together on a virtual construction site, where every architect or engineer can see which specialist operates in what area of the growing building design. Specialists are rather independent and have their own case base, case representation, and memory organisation.

3.3.1 CASE BASE(S)

3.3.1.1 Design case
FABEL’s primary data sources are CAD(-like) drawings of building projects. Such drawings consist of a huge set of (thousands of) design components, like walls, windows, doors or staircases. Since complete drawings are extremely difficult to manipulate, they are sliced into more manageable layout fragments, i.e. sub-sets of components, which constitute the basis of FABEL’s cases. In principle, this basis can be supplemented, or ‘deepened’, with comments, requirements or constraints. Yet, so far, FABEL only contains shallow cases, containing design components without any interpretation.

3.3.1.2 Case representation
With an eye on retrieval and adaptation, cases in FABEL are represented both in their original form, i.e. as a collection of design components, and as several specialist-specific interpretations.

Each design component is stored as a CAD-object with both geometric and semantic attributes. The former determine the size and position of the object, the latter include the subsystem the component belongs to (e.g. space, construction, floor, roof, ceiling, wall, façade, supply air, used air, hot water, cold water, sewage), its morphology or function (e.g. use, technical service, connection), precision or resolution (e.g. zone, bounding box, element) and scale (e.g. building site, entire building, floor, room, part of a room).

In addition to this ‘neutral’ case representation, FABEL is experimenting with seven different ‘coloured’ representations, i.e. case interpretations specifically tailored to FABEL’s specialists. They include two types of feature vectors, two types of gestalts, terms, graphs and bitmaps. There is neither space nor need to discuss all seven.

---

15 Terms encode structural knowledge about spatial arrangements and relative positions of objects in a machine usable form. The term (copy (X, 2, object)), for instance, means: take an object,
representations in detail here. Instead we have listed for each a concise description and representative example in table 3.1. For reasons of efficiency, the specialist-specific representations are constructed during pre-processing, and stored in the specialist’s case base some time (hours or even years) before being used for retrieval and/or adaptation. In fact, it are these derived representations (or interpretations) that constitute the indices of FABEL’s cases. Only they are inspected for retrieval.

3.3.1.3 Memory organisation
FABEL’s specialists do not only differ in how they interpret cases, they also organise their case base in different ways (see table 3.1). Bitmaps, for instance, are arranged in a hierarchy, the root of which has the coarsest resolution; graphs are partitioned into classes with a common maximal subgraph; and terms can be dynamically organised as a list or hierarchy, or can be partitioned.

3.3.2 RETRIEVAL
In FABEL, the purpose of retrieval is to find design cases similar to a given one (the query or problem). Just as in CADRE, users can manually select a case from the case base. In addition, however, FABEL offers various retrieval specialists to be used either separately, or in combination. The choice for multiple retrieval methods is motivated by the fact that two designs can be similar in various ways. The notion of similarity varies considerably with the situation, the intention of the user, or even personal preferences. For that reason, each retrieval specialist compares a different aspect or combination of aspects of the designs. In order to do so, the query is first dynamically interpreted and transformed into the specialist’s preferred representation, which is subsequently matched to the case representations in the corresponding case base. For this matching, each specialist is equipped with an appropriate comparison method (see table 3.1). When using feature vectors (1), for instance, a distance-based search algorithm compares the query fragment with a list of cases and returns a sorted list of the most similar ones. Optionally, the user can specify the number of retrieved cases and/or the degree of similarity, c.q. distance. In case of graphs, a graph algorithm matches the query graph with the representative common sub-graph of the classes in the graphical case base, and subsequently with the individual members of the best matching class. The output is a partially ordered set of structurally similar graphs.

In principle, all these retrieval methods can be combined at will, either sequentially or in parallel. In practice, however, FABEL furnishes four fixed ‘retrieval teams,’ which employ one or more of the retrieval specialists described. The first team, called ASM, compares cases by type and number of objects contained, combining feature vectors (2) with predefined gestalts. In the FAV-team, these gestalts are mingled with feature vectors (1), so as to compare cases by size, gestalts, number and types of objects. The third one,
CHAPTER 3

<table>
<thead>
<tr>
<th>approach</th>
<th>derived representation</th>
<th>example</th>
<th>memory organisation</th>
<th>comparison</th>
</tr>
</thead>
<tbody>
<tr>
<td>feature vector (1)</td>
<td>number of objects in layout fragment</td>
<td>no. of objects: 696 subsystems: (used air) functions: (equipment connection) resolutions: (zone bounding-box) scales (4 6 7) width: 5040 height: 5280 flip horizontal: no flip vertical: no</td>
<td>linear</td>
<td>distance-based user can specify: number of retrieved cases min. degree of similarity</td>
</tr>
<tr>
<td></td>
<td>different joints of semantical attributes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>width &amp; height of occupied area</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>feature vector (2)</td>
<td>(subsystem function resolution scale)</td>
<td>(10 used-air equipment zone 4)</td>
<td>associative</td>
<td>distance-based</td>
</tr>
<tr>
<td>bitmap</td>
<td>bitmaps of different grain sizes</td>
<td></td>
<td>hierarchical pixel-wise</td>
<td></td>
</tr>
<tr>
<td>predefined gestalts</td>
<td>predetermined set of 10 gestalts encoded as additional features (opaque to the system)</td>
<td>fishbone, row, …</td>
<td>associative</td>
<td>distance-based</td>
</tr>
<tr>
<td>term</td>
<td>machine usable encoding of objects’ arrangement &amp; position</td>
<td>copy (x, 2, object)</td>
<td>hierarchical modification</td>
<td></td>
</tr>
<tr>
<td>graph</td>
<td>nodes represent objects, edges connecting objects or neighbour relations</td>
<td>connecting object e.g. pipe</td>
<td>graph matching</td>
<td></td>
</tr>
<tr>
<td>dynamic gestalts</td>
<td>dynamically assembled gestalts</td>
<td></td>
<td>linear</td>
<td>structure mapping</td>
</tr>
</tbody>
</table>

TABLE 3.1. FABEL’s 7 approaches to case representation.

ODM, employs but one specialist, and retrieves cases based on object density maps only. Finally, the all-rounder ASPECT allows to combine and weigh several aspects of similarity, so that for every retrieval the most suitable blend of similarity measures can be defined.
3.3.3 MANIPULATION

As for retrieval, FABEL provides multiple methods to manipulate cases. AAAO and AgentEX are specialised in adapting construction and supply-air systems respectively, while TOPO is an all-rounder in topological adaptation. Unlike FABEL’s retrieval methods, however, all adaptation specialists share the same case representation: they view the design components and topology, and have only a limited field of vision. AAAO and AgentEX, for instance, can be used to satisfy specific constraints in the neighbourhood of certain design components, while maintaining a given topology.\(^\text{17}\) Note that the method that adapts a case may, does not need to coincide with the retrieval method that retrieved it in terms of how designs are viewed and compared.

AAAO was developed for a rather specific task, namely to position the columns of a MIDI construction\(^\text{18}\) in a building. In order to do so, AAAO is equipped with a special(ised) set of constraints on the positions of columns. The specialist either starts with a case, i.e. a distribution of columns in a similar building, or creates an initial distribution itself. The columns of the distribution begin to act in parallel as active autonomous objects: they evaluate their positions and strive to improve them by moving, generating new columns, or even destroying themselves. When the process stops, AAAO assesses the final distribution.

AgentEx supports the development of the installation system of a building. Like AAAO, it follows the paradigm of building elements being agents which autonomously develop solutions, yet the scope of tasks it supports is considerably greater.

TOPO, the least specialised of the manipulation specialists, takes care of the topology of a case. It is not task-specific in the sense that it can deal with any types of objects. In order to furnish a room, for instance, the user first selects objects of the type of furniture needed. After placing some of them, TOPO can be called in to complete the arrangement. Based on the most similar case in the case base, TOPO organises the rest of the requested furniture in the room. In case of suspicious topological relations, for instance a table against a door, TOPO warns the user and, after confirmation, adapts the topology. For this purpose cases are used that are more similar in the critical area.

3.3.4 FURTHER FABEL INGREDIENTS

As for retrieval and manipulation, FABEL also offers specialists to assess design solutions after adaptation, such as DOM or CheckUp. Moreover, being a hybrid system, FABEL contains some tools that generate solutions from scratch rather than based on existing cases. Examples of such tools are Roude, ANOPLA and Syn, and, to some extent, AAAO and DOM. Our main interest being CBR, however, we will not discuss them in detail here.

\(^{17}\) The approach is similar to CADRE’s dimensional adaptation (see case study 3.2, pp.68-69).

\(^{18}\) MIDI is a steel framework construction kit that was developed and applied by Fritz Haller [Haller, 1974]. It was designed for medium-sized buildings with high demands on technical services, like schools, laboratories or offices.
3.3.5 DISCUSSION

A very interesting aspect of FABEL is the idea to look at cases through various coloured filters, and to combine these filters into new colour combinations. Unfortunately, the colours chosen are rather biased, looking at cases from the point of view of either (2D) geometry or technology. In part, this is probably due to the specific task FABEL has chosen to support, that is designing buildings with complex installations. It could be argued, however, that also, and perhaps especially in such buildings, other aspects of architecture should not be overlooked.

A second asset of FABEL is its attempt to fit architects’ design process. An architect, we mentioned it before, cannot proceed in a straight line from (well-defined) problem to (optimal) solution, because both develop in parallel. In order to support this non-linear process, FABEL has opted for a fine-grained model of problem solving activities and a corresponding corps of specialised assistants. Rather than imposing on the user a fixed flow of procedures activated in sequential order, specialists can be used in a mutually interdependent way.

At this point one may ask why the grains of FABEL’s model, and thus the members of the specialist corps, correspond to sub-tasks of CBR, and not to sub-tasks of architectural design. As FABEL’s developers admit, “In design, elementary tasks seem to be interpretation, comparison, insertion, constraint checks, modification to comply constraints; which recombine in different ways to yield super-tasks like retrieval, elaboration, assessment” [Voss, 1997]. If the main objective is to support architectural design, why not start from the elementary tasks in design in the first place?

A more fundamental point of critique concerns the total lack of integration of FABEL’s specialists. Each specialist has its own case interpretation, case base, memory organisation and comparison method. Judging the independence of the methods and flexibility of the system more important, the FABEL team decided to leave the issue of integration for the time being. In our opinion, however, integration is a fundamental feature of CBR’s model of cognition. One of the reasons to call in cases for architectural design is precisely the fact that they are integrations of multiple aspects.

A final, more practical drawback of FABEL is that its users are assumed to work with a fully-developed CAAD-package, in other words, a CAD-package that allows to build up building models with domain-specific attributes. Although several research efforts have led to various conceptual models of such packages, so far few models have reached the stage of implementation, let alone of being commercially available. By consequence, hardly any architect currently corresponds to the clientele or end users FABEL has in mind. It could be argued, however, that this drawback only applies in the short term, as a convincing breakthrough of commercial CAAD-packages is probably (and hopefully) nothing but a matter of time.

---

19 See chapter 1, p.13.
20 See chapter 2, pp.55-56.
21 For an overview of the major initiatives in this field, we refer to [Hendricx, 2000].
3.3.6 REFERENCES


3.4 IDIOM

IDIOM – Interactive Design using Intelligent Objects and Models – is a CBD system for composing building layouts. The system was developed in order to investigate human-computer interaction, the use of preferences, and Model-Based adaptation and combination. Coming from the same stock as CADRE, IDIOM shows considerable improvements compared to its predecessor, the most important of which is undoubtedly its increased interaction.

3.4.1 CASE BASE

3.4.1.1 Design case
According to the IDIOM team, architects frequently reuse existing designs, yet rarely wish to adapt a whole building. Therefore, cases in IDIOM are no longer complete building designs, as was the case in CADRE, but separate spaces of constructed apartment buildings. Each case contains windows, furniture and doors, and has been carefully selected by an architect for its flexibility, compatibility and success as design. Apart from the location of the ‘mother building’ and the name of the architect, no background information is stored.

3.4.1.2 Case representation
As figure 3.3 shows, IDIOM represents its cases by means of abstract floor plans. In addition to walls, doors and windows, these plans also reveal the position of furniture in the original design. Pieces of furniture are reduced to grey rectangles, not only representing the size of the element, but also additional space needed for adequate use.

Additional information is given in a dialogue box, containing the current values of the case’s variables. These include the outer dimensions of the space as well as the positions.
of the elements (doors, windows and furniture) it contains. The sizes of the elements are fixed.

A third ingredient of IDIOM’s case representation is the display of arrows, indicating how far a given element can be moved. Arrows appear as soon as an element is clicked on with the mouse.

3.4.1.3 Memory organisation
IDIOM’s cases are labelled according to their function and grouped into five types: living rooms, kitchens, bedrooms, bathrooms and hallways. Since the system focuses on case adaptation and combination rather than on indexing and retrieval, memory organisation was not an item with a high priority on the IDIOM agenda.

3.4.2 RETRIEVAL

Retrieval in IDIOM is entirely left to the user. After having defined the dimensions of the site where the layout must be placed, the user selects a case from the case browser and inserts it into the site. As already mentioned, cases are organised into five groups according to their function.

3.4.3 MANIPULATION

Although IDIOM focuses on human-computer interaction, it is the system itself that adapts the selected case to the new environment. The interaction lies in the fact that the constraints on which this adaptation is based, are (partially) defined by the user.

IDIOM distinguishes between two types and three sources of constraints. The first distinction has to do with how imperative a constraint is. Fixed constraints must be fulfilled, whereas preferred constraints, in short preferences, may be de- and reactivated. De-activation occurs in case of conflicts with fixed constraints or preferences with a higher priority, re-activation when new opportunities arise. According to IDIOM’s developers, the concept of preferences was introduced to express those design requirements that cannot be modelled more precisely, such as social and political considerations.

Apart from types, IDIOM also distinguishes between different sources of constraints. When the user inserts a case into his design, all constraints associated with it in the case base are added to the current set of constraints.

The user can add further constraints to interpret the case in its new environment. He can specify the topology of the design by defining neighbourhood relations between two rooms. Such specification is possible for each pair of adjoining objects and uses a set of primitive topologies. After having consecutively selected the two rooms with the mouse, the user declares, changes or removes the relationship between them using the corresponding commands under the neighbourhood menu.

In addition to topology, the user can add constraints on sizes, distances and alignments between rooms and their elements. This is done by constraint posting, IDIOM’s most direct method of user interaction. Double clicking on an object causes to appear a dialogue box with the object’s current values, which users can manipulate directly.
Constraints that may be fixed include the object’s minimum width and length, absolute width and length and minimum area. The user can also post preferences and specify their priority, for example, stipulating that the minimum area of the dining room is more important than the preferred size of a single bedroom.

Before calculating a new solution for the layout, the system extends the current constraint set with constraints from active domain models. In IDIOM, models are causal mappings from structural parameters to behaviour related to individual objects and object groups. They provide the system with domain knowledge in the form of strict rules, guidelines, technological considerations and personal designer preferences. It is the user who chooses to activate a model by selecting it from a pull-down menu. The current version of IDIOM contains four models, reflecting the scope of domain knowledge the system has access to: subsidised housing, economical façades, adequate natural lighting and luxury construction.

All constraints, either fixed or preferred, imposed by user interpretation, domain models or the case itself, turn IDIOM’s cases into intelligent objects. Originally no more than a part of a successful design, a case becomes increasingly intelligent as it is interpreted by the designer within the context of a new design task. It is not the ‘rough’ case, but this intelligent object that is adapted to the new environment so as to fulfil all fixed constraints and as many preferences as possible. IDIOM calculates the feasible solution space through conflict resolution with preferences and dimensionality reduction,22 and selects a solution that involves minimal changes to the case and to the current design. The result is nothing more than a proposal, as the user still may change positions of walls, doors, windows and furniture. The calculated solution space is then used to display arrows indicating how far a given element can be moved.

The scenario just described can be repeated several times until a complete apartment layout is composed. When a case is added, the parameterisation as well as the set of constraints grows. Since dimensionality reduction is performed each time on the complete set, IDIOM finds a solution whilst preserving positions and sizes in the current design whenever possible. It maintains design requirements by solving constraints in stead of propagating them, thus identifying globally feasible solutions without running the risk of propagation cycles.

3.4.4 DISCUSSION

IDIOM, we already announced it, shows important improvements compared to its predecessor CADRE. The most fundamental change lies undoubtedly in its ambition to support architects as they compose designs themselves, rather than automate the design task.

This increased focus on interaction does not only apply to the use of the system, but also to its development. Architects are called in to select valuable designs for the case base and to test the system. This is, in our opinion, the most plausible way to arrive at a reliable CBD system that meets the requirements of today’s design practice.

22 See also p.69.
Another very interesting aspect of IDIOM is the incorporation of design knowledge in the form of models that can be switched on and off. There are indeed plenty of guidelines that architects must take into account, such as the specific requirements for subsidised housing. Including such guidelines in the constraint set during case adaptation guarantees that the final design solution will meet these requirements.

Of particular interest is the concept of objects becoming intelligent at run time, which reflects, in some sense, how a design develops throughout the design process. The lines and shapes architects draw early in the design process are actually little more than shapes and lines. It is only afterwards, as the design evolves, that these early sketches acquire meaning and start to represent brick walls, steel columns, wooden doors or other concrete objects. To some extent, IDIOM supports this gradual development by allowing the user to make an object as intelligent as he wants through constraint posting, model activation and preference specification.

According to the IDIOM team, introducing preferences should allow users to express design requirements that cannot be modelled more precisely, such as social and political considerations. In practice, however, both preferences and fixed constraints express dimensional requirements, the only difference being that the former are less imperative and can be given a certain priority. This means either that the social and political aspects of architectural design can be reduced to dimensional constraints, which – as far as we know – is not the case; or that IDIOM has not fulfilled the ambition to take them into account.

A second point of critique is IDIOM’s reduction of architecture to 2D layouts. The problem does not so much lie in the case representation – architects are after all used to working with floor plans – but rather in the fact that the third dimension, the height, is not even considered as a significant room parameter. Since architecture is essentially a spatial intervention, reducing rooms to 2D organisations speaks in our view of an impoverished view on architectural design.

Finally, there is the lack of context. According to the IDIOM team, each case has been carefully selected by an architect for its flexibility, compatibility and success as design. In our opinion, however, these qualities do not only depend on the space itself, but also on its context, that is its relation to neighbouring rooms, to the environment, etc. In IDIOM, the only background information stored is the name of the architect and the location of the mother building, which makes it difficult to interpret and situate the case in its original context. For such interpretation, at least an overview of the entire building design should be given.

3.4.5 REFERENCES


It should be noted, however, that this does not apply to all architects. See for instance the view expressed by Paul Van Aerschot, Eva Jiricna and Robert Venturi in chapter 4, p.104.
CHAPTER 3

3.5 PRECEDENTS

PRECEDENTS is a Case-Based Design Aid for architecture developed by Rivka and Robert Oxman. The system stores memorable design cases that have the status of precedents, i.e. recognised outstanding examples of a particular type or style of design, and makes these available to architecture students.

The design task PRECEDENTS concentrates on is the spatial organisation of museums in the early conceptual stage of the design process. In this stage, the role of cases is to support the human designer by providing him with meaningful concepts for museum design – unique ideas, explicit conceptual points and associated design principles.

The focus of this system has been on case representation, more specifically on representing the conceptual knowledge embedded within past designs, and making it fit to be searched and browsed within a computerised library of design precedents.

3.5.1 CASE BASE

PRECEDENTS stores significant designs from the past that provide relevant ideas for the spatial organisation in museum design. The library contains a number of famous museums, such as Frank Lloyd Wright’s Guggenheim museum in New York, James Stirling’s Neue Staatsgalerie in Stuttgart, or Le Corbusier’s Visual Arts Centre in Cambridge. In order to illustrate the possibility of cross-contextual search in a future larger precedent base, these museums are supplemented with some related building types, like for instance Hans Hollein’s Viennese jewellery store. All examples were selected as representative of current significant design, and for the range of issues and concepts they present.


24 Note that in some publications PRECEDENTS is referred to as MEMORABILIA.
3.5.1.1 Design case
Just as the courthouses and libraries in Archie-II, existing graphical descriptions of the selected precedents are obviously too large for teaching students about spatial-organisational concepts in museum design. Moreover, they do not explicitly express the concepts behind the design, which makes their interpretation entirely dependent on the viewer’s familiarity with the design or design field. Rivka and Robert Oxman therefore decided to decompose the selected designs into more manageable chunks by means of design stories.

Unlike Archie-II, PRECEDENTS only uses one type of stories. A design story is an annotated piece of text describing a conceptual point that characterises the uniqueness of a design. It extracts the meaning of a particular part of the building, by highlighting those aspects that are especially meaningful to a specific design task, in this case the early conceptual design of museums.

‘The urban site in Stuttgart which was given for Stirling to design the municipal museum consists of two main parallel roads that requires public access in order to provide for urban continuity. The museum, on the other hand, requires a private access system that does not accommodate public circulation. Stirling resolved this conflict in a unique and interesting way by cutting the building and creating a path through the building around an external court.’

(Neue Staatsgalerie, Stuttgart, James Stirling)

This example illustrates how a story links a particular design issue, e.g. urban continuity, with a specific design concept, e.g. path through, that addresses this issue. The public ramp passing through the central court is the form element that materialises this concept. These linkages between a design issue, a solution concept and a form element constitute in fact the knowledge content of precedents that the system tries to make explicit.

It is obvious that many different stories can be annotated to a single design. It is also obvious that issues or concepts in a story of one design may occur in stories of other designs as well. The issue of urban continuity, for instance, inspired both the central court of the Staatsgalerie and the walkway of the Visual Arts Centre (see figures 3.4 and 3.5).

PRECEDENTS’ stories were collected by analysing critical writings on the selected museum designs. It is around these design stories, rather than around design cases, that the system’s memory is organised.

3.5.1.2 Case representation
The representation of a design story comprises three components (see figure 3.6): a piece of text (the actual story), an illustration of the corresponding precedent, and a network of design issues and concepts connecting the story with stories from other precedents.

Precedents are usually illustrated in graphical form, which provides for a complete structural representation of the design as an entire solution. To this graphical illustration, several design stories are annotated in the form of short pieces of text, like the example cited above. This enables the graphical representation to play a double role: it illustrates

25 The concept of ‘path through’ describes the ability to pass through a building without entering it.
the annotated design stories and, at the same time, clusters them into precedent cases. The conceptual descriptions are derived from the concept vocabulary of the design task, i.e. all key concepts for the spatial configuration of museum design. This vocabulary, which contains issues like urban continuity, orientation and exposition, was identified by performing a content analysis of architectural literature.

3.5.1.3 Memory organisation
As already mentioned, memory is organised by design stories rather than by the design cases they describe. In essence, the memory structure is an associative network of stories related to high-level concepts of the task domain. The network is based on a tripartite representational schema, called the Issue-Concept-Form (ICF) formalism. In this formalism, the term issue refers to a point or objective that must be addressed in the final design (e.g. the issue of urban continuity in Stirling’s museum), the concept formulates a design idea in relation to that issue (e.g. path through), and the form represents the physical design artefact that materialises this solution principle (e.g. the ramp in the central court). The explicit mapping between these three results in a semantic network, which provides the basis for the indexing system. Instead of indexing the complete designs, every design story is indexed independently by the issues and concepts addressed in the text (the keywords in italics in the example above).26

This indexing by high-level domain concepts supports both directed search and browsing. The domain vocabulary of issues and concepts acts as the lexicon of

---

26 In general, each story addresses one issue.
PRECEDEENTS’ memory. From any node – issue or concept – in the network, related stories can be retrieved, which in turn call up the precedents they refer to. In addition, the indices connect stories from different precedents in which similar concepts occur, thus allowing the user to navigate from one design case to another.

### 3.5.2 RETRIEVAL

Like Archie-II, PRECEDEENTS offers two possibilities for retrieving relevant cases: directed search and browsing.

A first possibility is that the user specifies the design issues he is interested in by filling in a form. The system then searches relevant design stories by using these issues as entries in the lexicon of the semantic network. The retrieved stories automatically call up their corresponding precedents.

If the user wants to investigate other solution concepts, or wishes to explore various concepts in the same precedent, he can use the story’s indices to browse through the library. Doing so, the user exploits the indexing system in an associative way by activating the linkages between similar design concepts within the semantic net. This enables him to find alternative solutions for the same design issue in diverse stories and precedents.

Figure 3.7 illustrates stories, concept vocabularies and recall of related examples relative to the story cited above. After the user has specified the issue of urban continuity, PRECEDEENTS has selected the Staatsgalerie as relevant precedent. Design concepts

---

**Figure 3.7.**
Browsing in PRECEDEENTS.
underlying this museum are linked with related concepts of other precedents in the case base. The issue of orientation, for instance, connects the central court to a story about the ramp in the Guggenheim museum. Apart from museums, however, the Staatsgalerie shares concepts with other types of buildings too. For example, the decoration Stirling used to emphasise the entry of the museum cues the decorative entrance of Hollein’s jewellery shop.

During browsing, the user activates the links between the stories about the Staatsgalerie and those about other design cases. This automatically calls ups the corresponding graphical illustration, which may lead to the discovery of new, often unanticipated design concepts.

3.5.3 MANIPULATION

At this point, PRECEDENTS’ task is finished. The objective of the system is to help students access conceptual knowledge (concepts, ideas, principles) embedded in previous projects. The user is entirely free to decide whether and how to use this knowledge for his current design.

3.5.4 DISCUSSION

The major asset of PRECEDENTS is undoubtedly its attempt to elucidate the concept knowledge encapsulated by design cases. Currently, concept vocabularies in architectural design are poorly formulated. With PRECEDENTS, Rivka and Robert Oxman have tried to change this situation. Of course, their schematic vocabulary for museum design fills only a small corner of the huge domain of architecture. Yet the level of abstraction of the concepts and the method for deriving them, i.e. content analysis of architectural literature, seem widely applicable and thus easily extendible to other types of buildings.

A disadvantage of acquiring knowledge by analysing critical writings, is the enormous amount of work it requires. A full-scale version of PRECEDENTS will contain many more design cases than it currently does, and should be filled up and updated continuously. This raises the question who is going to perform all these analyses. Architecture students, i.e. the end users for which the system is meant, seem to be cut out for this task. They could analyse one or more precedents in preparation for, say, a design assignment or a seminar on architectural history. It is obvious that students will acquire more knowledge about a specific building by analysing it than PRECEDENTS can provide them. Yet, the system does not pretend to substitute such analyses. Moreover, a full-scale version could provide students with much more design precedents than they could ever analyse by themselves.

In some respect, the stories in PRECEDENTS can be considered less ‘objective’ than the stories Archie-II uses. Indeed, the former do not only depend on the author of the source text, but also on the person who performed the analysis. By contrast, the ‘pure’ records of people’s opinions in Archie-II may be less dependent on the person who collected them. There is a case for the neutrality of Archie-II, but also for the ‘personal

---

27 See case study 3.1 p.61.
CHAPTER 3

touch’ of PRECEDENTS, which, in some sense, can be considered an enrichment of the conceptual knowledge it provides.

Whether one prefers the neutrality of Archie-II or the personal touch of PRECEDENTS, the design knowledge both systems have to offer does not only reside in the design cases they contain, but also, to some extent, in the refined indexing system that organises these cases. The ICF-formalism in PRECEDENTS provides a sophisticated mechanism to elicit concept knowledge that is implicitly present in previous projects. Yet, every medal has two sides. Just as in Archie-II, the sophistication of the indexing system seriously complicates the addition, removal or rearrangement of cases and stories to update the case base.

3.5.5 REFERENCES

3.6 SEED

SEED – Software Environment to support Early building Design – is a hybrid design system that combines elements from both Case- and Model-Based approaches. During the early phases of building design, the system provides support for analysis, evaluation, but first and foremost rapid generation of design representations. In order to do so, SEED is conceived as a collection of modules, each addressing a specific task in the design process. So far, modules have been – or are being – developed for architectural programming, schematic layout design and 3D configuration generation. On a general level, all SEED modules model design problems and solutions in a similar way, yet they are free to choose internal representations or data models according to their individual computational needs.

Focus of this case study is SEED-Layout, a module for generating schematic layouts of the functional units of an architectural program. It is the most documented, and thus probably most developed module within the SEED-environment.

3.6.1 CASE AND OTHER BASES

Given the hybrid character of SEED, the case base is only one of the three ‘bases’ the system has at its disposal. The most robust of the three is the object database, which is based on an object-oriented paradigm, and whose underlying object model is shared by all modules. Objects in this database are instances of a certain class, and are characterised by attribute-value pairs. If one or more of these attributes are lists of objects of the same type, the object is called structured. Classes of objects that play a crucial role in SEED are Design Units (DU) and Functional Units (FU). Whereas the former are basic spatial entities that make up the representation of a design solution, the latter collect all requirements a DU must satisfy within a single design problem. In SEED-Layout, for instance, DUs are spatially continuous areas of a building with a specific geometry and location. FUs specify the minimum area requirements and spatial components these areas must allocate. Parallel to the decomposition of DUs (e.g. floors) into constituent units (e.g. zones, corridors and rooms on that floor), the corresponding design problems are decomposed into a hierarchy of sub-problems.

For modelling and technical reasons, the SEED team has chosen to equip its object model with single inheritance only. Nevertheless, objects can be multiply classified thanks to the development of a parallel knowledge base, which contains multiple classificatory taxonomies. These taxonomies are much less strict than the object model underlying the object database, and may vary from one module to another. SEED-Layout, for instance, may classify layouts by function or occupancy, e.g. as ‘residential floor’ or

---

28 Within an object-oriented approach, single inheritance means that an object class B inherits the structure or behaviour defined in another object class A. In case of multiple inheritance, an object class may inherit structure or behaviour defined in two or more distinct classes, which results in a class hierarchy that is a lattice instead of a tree [Booch, 1991 p.99].
‘kitchen’. In order to cleanly separate knowledge base from object database, classifications in the former are attached to objects in the latter via proxies.

The case base for its part is separated from both knowledge and object database. What constitutes a case in SEED first of all depends on the module being considered. Corresponding to its central object of interest, cases in SEED-Layout are individual layout designs. Secondly, it also depends on the level of decomposition in the object model. In SEED-Layout, for example, a case may be the layout of an entire building, a floor, a zone, a corridor or an individual room.

Regardless of the module or level in the hierarchy, each case is represented as a collection of proxies, which respectively represent a case description, a problem (a FU), a solution (a DU) and an outcome. Of these four proxies, the case description functions as index for retrieval. What this index exactly consists of varies once again from one module to another. In SEED-Layout, for instance, the case description contains the problem that is solved by the layout at stake. In other words, layouts are indexed (and thus retrieved) based on the FUs they allocate and the requirements they satisfy. Other modules may index (and retrieve) cases based on characteristics of the solution or some computed outcome. According to the SEED team, a major advantage of separating the case description from the usual problem-solution-outcome triad, is that it adds flexibility to case indexing and retrieval.

As to the creation of a case’s index, SEED has opted for a semi-automated approach. Suppose the user has created a successful layout with the support of SEED and wants to store it in the case base for future use. In order to do so, he can simply ask the system to generate an indexing scheme and store the case under that index. Optionally, the user may edit the index by adding extra classifications or annotations.

3.6.2 RETRIEVAL

3.6.2.1 Characterising the target
In order to retrieve relevant cases, the user enters an object configuration, which is to serve as target index at run-time. In SEED-Layout, the target index will be a problem specification, as layout cases are indexed by the problems they solve. In principle, users can specify whatever requirements they judge important. In practice, however, they are restricted to those requirements that can be taken into account by the module during automated form generation and case retrieval. In the course of the design process, the target specification can be changed dynamically, either by the user or by the module itself.

3.6.2.2 Recalling and selecting cases
As soon as the user has specified a target index, the case base can be searched for matching descriptions. Since this target index consists of a complex object configuration – i.e. a structured object composed of several attributes, some of which are in turn (structured) objects – comparison between target and case index proceeds object by object and attribute by attribute. SEED provides special match algorithms to this end, which rank the retrieved cases according to the closeness of fit with the target. To be precise, this measure of closeness to the target index is combined with values of object attributes as
defined in the object database, and classifications as described in the knowledge base. Preference is given to cases that have more structure, in other words, the more refined the hierarchical decomposition, the higher the ranking, everything else being equal. According to the SEED team, this would save time, both in terms of finding an initial solution, and in terms of re(de)fining the problem specification itself.

3.6.3 MANIPULATION

Just like case indexing, manipulation of cases in SEED is semi-automated. Important to notice is that this manipulation does not just affect the user’s design solution, but also his problem specification. Once a case, say a floor layout, has been selected, new design units of the case are inserted into the current solution. In a similar way, the associated functional units become part of the current problem specification, after which they can be inspected and interactively edited by the user.

3.6.4 DISCUSSION

Design problems, as we already mentioned, are not completely defined at the outset, but gradually develop during design. Accordingly, there is no assumption in SEED that the problem specification is complete at any time in the design process. The problem can be dynamically re(de)efined either by the user or by the system itself. Indeed, case adaptation may involve changes not only in the design solution, but also in the problem specification.

At odds with these dynamics within one module is the strict separation between and sequence of several modules. 2D layout and 3D composition are assumed to be clearly separate stages in architectural design, and are therefore handled by independent modules, each having their own problem specification, generation and evaluation element. Moreover, these modules are to be used in a fixed order, as the output of SEED layout constitutes the problem specification of the 3D composition module. In reality, however, architects do not necessarily start in 2D and, when this is finished, proceed with 3D. Some prefer to work exactly the other way around, while others – probably the majority – steadily switch back and forth between both.

At first sight, SEED’s separation of modules does not need to hamper such switching, as long as cases can be retrieved and reused across different modules. Being one of the long-term objectives of the system, this facility will probably be worked out as more modules are developed. In this respect, it is somewhat surprising though that SEED’s developers have chosen for module-specific case descriptions, i.e. indices that may vary from one module to another. As they contend, this undoubtedly adds flexibility to case indexing and retrieval, yet to us, such flexibility seems rather counter-productive when cases are to be exchanged across different modules.

A final assumption of SEED that makes our eyebrows raise, is that design problems can be decomposed into sub-problems, which in turn can be decomposed into smaller sub-problems, and so on. Because real-world design problems are barely decomposable,
every decomposition is necessarily an impoverished version of the original problem. In fact, this was one of the main reasons to apply CBD to design, as cases may provide the glue that holds the different components of a design together.

3.6.5 REFERENCES


---

30 See chapter 1, p.9 and chapter 2, pp.55-56.
3.7 WEBPAD

WEBPAD is a web-based design system that integrates CBD and CAD technology. Its main objective is to provide architects with relevant designs from a distributed resource of graphical CAD files on the web. WEBPAD comes from the same stock as PRECEDENTS, yet being much younger, the system is still largely under development.

3.7.1 CASE BASE

WEBPAD starts from the idea that each designer or design office develops his own case base. This case base can be either kept locally or shared with other WEBPAD users via the Internet.

3.7.1.1 Design case

Defining a case as an entire building design results in quite complex cases. On the other hand, considering building components (e.g. rooms) as cases makes it difficult to interpret and situate a case in its original context. Therefore, WEBPAD has chosen to provide both complete descriptions of the entire building, and descriptions of its separate components.

3.7.1.2 Case representation

One of the major ambitions of WEBPAD is to incorporate CAD drawings within a design case representation. In order to do so, the WEBPAD team has developed a multi-layered model for case representation comprising three different layers. The project layer provides a total representation and forms a cohesive framework for all documents related to a project. The drawing layer contains a graphical CAD representation of the entire building design. The component layer represents graphical and textural information of separate components of the drawing layer.

3.7.2 RETRIEVAL

Retrieval is taken care of by a Java applet running either within the user’s CAD program (e.g. Microstation) or in his Web browser (e.g. Netscape or Internet Explorer). When the user starts a search for a particular design, the applet first scans the local case base (if any), and then proceeds to other case bases accessible through the Internet.

Similarity measurements will take into account the similarity degree in each layer, and will make use of a semantic network based on the ICF-formalism that was developed in PRECEDENTS.

---

31 See for instance our critique on IDIOM p.81.
3.7.3 MANIPULATION

Just as in PRECEDENTS, manipulation is left entirely to the human user. He decides whether and how the retrieved case will be used in the current design situation.

3.7.4 DISCUSSION

Compared to other CBD systems described, WEBPAD is still in its infancy. We have nevertheless chosen to add it to the case base, because it represents some interesting ideas we did not find in other systems.

The first one is the idea to share cases, i.e. CAD drawings of building designs, across different designers and design companies. If CBD systems remain inside the walls of one design office, the efficiency within that office may very well improve. Yet, if CBD is to improve the efficiency of architectural design in general, systems should surpass the boundaries of individual design companies, which is exactly what WEBPAD wants to do.

A second asset of the system is that its case base automatically grows as a side-effect of the users’ ‘normal’ activities. Indeed, more and more architects make CAD drawings of the buildings they design. The only extra effort WEBPAD demands, is to store these drawings in a case base that is accessible through the Internet, and specify their access permissions if necessary. However, how and by whom these CAD drawings are to be converted into WEBPAD’s three-layered representation, is not entirely clear yet. This conversion seems far from trivial a task, the solution of which will determine whether or not the system can learn by ‘simply’ storing new cases.

A final advantage of WEBPAD, in our opinion, is the fact that its retrieval applet can run from within the user’s CAD program. In other words, architects who want to use the system do not have to learn (and buy) another CAD package, but can stick to the package they are used to working with. Moreover, during design, they do not have to leave the design environment they are working in to search for a relevant design case.

3.7.5 REFERENCES


---

32 See chapter 2, p.56.
33 See chapter 2, pp.57-58.
3.8 Synthesis

In this chapter, we have put a number of Case-Based Design systems for architecture under the microscope. Table 3.2 gives an overview of these systems and summarises which ingredients of CBD’s model of cognition they have implemented. Ingredients explicitly mentioned by the system developers as point of special interest are marked in darker grey.

As to the knowledge structures and organisation proposed by this cognitive model, all systems described contain specific events, c.q. cases. These cases are implemented as design products at various scales, ranging from entire building designs (Archie-II, CADRE, PRECEDENTS, WEBPAD) over layout designs (SEED-Layout) to spaces (IDIOM) or components (FABEL, WEBPAD). For reasons of manageability, cases may be further chopped up into stories (Archie-II, PRECEDENTS), specific representations (CADRE, FABEL) or components (SEED-Layout, WEBPAD).

In addition to specific events, the model predicts several kinds of generalised knowledge that are likely to reside in memory. Generalisations turn up in Archie-II (guidelines), PRECEDENTS (high-level concepts) and SEED (in the form of object model and classifications), whereas adaptation knowledge can be found in CADRE (constraints, rules and grammars) and IDIOM (constraints and domain models).

According to CBD’s model of cognition, specific and general knowledge are organised in the same way. Strictly speaking, only Archie-II does right to this aspect of the model, as its stories and guidelines are linked by a common set of indices. A slightly different approach to memory organisation is adopted in PRECEDENTS, where stories are organised by a sophisticated tripartite Issue-Concept-Form formalism. In this way, generalisations, c.q. high-level concepts, in fact serve to organise specific events. In SEED, on the other hand, the ties between specific and generalised knowledge are much weaker. The system comprises three parallel ‘bases’ that are clearly separated: an object database that stores the general object model, a knowledge base containing several classifications, c.q. generalisations, and finally a case base filled with specific cases.

When faced with a new design problem, the model provides several reasoning processes to use all this knowledge. First of all, memory remembers the knowledge that is most relevant for solving the problem at hand. In most CBD-systems, this remembering is implemented as retrieving those cases that have the most features in common with the new situation. To this extent, cases in Archie-II, for instance, are labelled by descriptive and relationship indices that can be compared with features of this situation. In FABEL, where users have the choice between four different ‘retrieval teams’, features range from functional components over occurrence of certain layout patterns to structure or topology. CADRE provides virtually no support for remembering, as it has chosen to concentrate completely on manipulation.

When it comes to manipulating the retrieved cases, the task of Archie-II, PRECEDENTS and WEBPAD is finished. All three systems have opted to leave this reasoning process entirely to the user. CADRE, FABEL, IDIOM and SEED-Layout, on the other hand, provide support for structural adaptation. They alter the dimensions and/or
topology of a given design solution so as to fit the new situation. SEED-Layout does not only adapt the solution, but also the problem specification. As to derivational adaptation and merging, there still seems a great deal of work to do. To our knowledge, very few of the systems described have implemented these kinds of manipulation.

The section of table 3.2 devoted to learning looks even more like virgin territory. A central ingredient of CBD’s model of cognition is the claim that memory dynamically improves its performance every time it is used – by storing new cases, re-indexing old ones or creating new generalisations. In the CBD systems studied, this dynamic learning is hard to find. Perhaps the tool most eager to learn is CADRE, whose pre-modeller Mod-4 aims at facilitating the acquisition of new cases. In WEBPAD, new cases would be acquired by simply making CAD-files accessible through the Internet, yet how these files are to be compiled into project, drawing and component layers, and indexed according to the ICF-formalism is not entirely clear yet.

In summary, ingredients of the cognitive model that are fairly well represented in the CBD systems studied are the use of specific events, case retrieval and structural adaptation. The combination with generalised knowledge and uniform organisation of both types of knowledge are less well represented, whereas derivational adaptation, merging and learning are quasi absent.
<table>
<thead>
<tr>
<th>KNOWLEDGE STRUCTURES</th>
<th>ORGANISATION</th>
<th>REASONING PROCESSES</th>
<th>LEARNING</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPECIFIC</td>
<td>GENERAL</td>
<td>ADAPT. KNOWL.</td>
<td>construction</td>
</tr>
<tr>
<td>Archie-II</td>
<td>STORIES</td>
<td>point interaction cluster</td>
<td>GUIDELINES</td>
</tr>
<tr>
<td>CADRE</td>
<td>BUILDING geometry/graph/grammar 4 levels of detail</td>
<td>dimensional CONSTRAINTS transformation RULES string GRAMMARS</td>
<td></td>
</tr>
<tr>
<td>FABEL</td>
<td>CAD-COMPONENTS (sub-sets) 7 interpretations</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IDIOM</td>
<td>SPACES plan dialogue box arrows</td>
<td>CONSTRAINTS domain MODELS</td>
<td>by function</td>
</tr>
<tr>
<td>PRECEDENTS</td>
<td>STORIES graphical illustration annotated text</td>
<td>high-level CONCEPTS</td>
<td>ICF-formalism</td>
</tr>
<tr>
<td>SEED-Layout</td>
<td>LAYOUT DESIGNS decomposition</td>
<td>object MODEL CLASSIFICATIONS</td>
<td></td>
</tr>
<tr>
<td>WEBPAD</td>
<td>BUILDINGS + COMPONENTS 3 layers</td>
<td></td>
<td>ICF-formalism</td>
</tr>
</tbody>
</table>

**TABLE 3.2.** Overview of the CBD systems studied along with the CBD ingredients implemented.
ARE ARCHITECTS NATURAL CASE-BASED DESIGNERS?

Architects speaking

“It’s almost like the tide: these references come and go, some hang on, others are dropped because they turn out to be not that relevant.”

Paul Vermeulen (2000)
CHAPTER 4

If we want to understand the role of cases in conceiving architecture, there are several possible techniques we can employ. One possibility is to analyse the task of conceptual design and propose roles that cases are likely to play in this task. The field of CBD research, which has been introduced in chapter 2, is largely characterised by this first approach. Many of the CBD systems presented in chapter 3 seem based on how architects are supposed to use cases rather than on any evidence that they actually do so. Although this technique can be easily criticised,1 one of its contributions is that it defines the subject with practical precision and draws attention to fundamental questions concerning concept generation [Lawson, 1994].

In an attempt to answer these questions, this chapter has opted for a different approach: we have asked architects to describe how – and even whether – they use previous cases to conceive architecture. This approach, however, has been subject to serious critique as well. Like any professional who has to sell his services in the marketplace, architects may not always describe their design processes honestly, afraid to give away ‘the secret of their success’ [Lawson, 1994; Eastman, 1999]. Moreover, given the intuitive and partially unconscious nature of architectural design, architects may not always be fully aware of ‘what they are doing when no one is looking.’2 And even if they were, few of them would be able to articulate this awareness with reasonable accuracy. Indeed, most designers are better at designing than at explaining [Lawson, 1994].

Despite these potential pitfalls, we have nevertheless opted for conducting in-depth interviews with a small sample of Flemish architects, be it with some special qualifications. The chapter begins with a concise description of how the interviews were conducted (section 4.1), and introduces each architect in a brief biography (section 4.2). The following sections discuss several aspects that, judging from our interviews, characterise case (re-)use in real-world architectural design: whether, which and why cases are used for concept generation (section 4.3), as well as in other stages of the design process (section 4.4), and how their use is viewed in architecture schools (section 4.5). Finally, section 4.6 ends with some concluding remarks about Case-Based Design in the practice and education of architectural design. Here and there, frames with mini case studies will turn up. They are not necessary to follow the thread of the story, but serve as a kind of intermezzo for the reader. The frames describe explicit cases of ‘Case-Based Design’ that were brought up in the course of the interviews.

4.1 Procedure

This chapter largely results from six in-depth interviews with the following architects: Jan Delrue, Mauro Poponcini, Paul Van Aerschot, Werner Van dermeersch, Paul Vermeulen and Peggy Winkels. The six were chosen because they represent different

---

1 See chapter 6.
approaches, practices and generations of architectural design. In addition, all interviewees have ample experience as design teachers, which makes them even more privileged sources for our investigation. Case-Based Design starts from the premise that architects learn design by experience. Precisely because of this learning aspect, it seems reasonable to pay special attention to how cases are used in design education. Being both practising architects and design teachers, our interviewees seem very well placed to shed more light on the role of cases not only in design, but also in learning (to) design.

The interviews were conducted at the architects’ offices [Delrue, 1998; Van Aerschot, 1998], at home [Van dermeersch, 1999] or at school [Poponcini, 1998; Vermeulen, 2000; Winkels, 1998]. In order to enable them to prepare the interview, the architects received a brief list of questions in advance. Although the interviews started from this formal questionnaire, the discussion often took place in a rather informal atmosphere. The informal quality of the interviews seems justified given their exploratory character and their aim, which was to examine whether architects and/or architecture students indeed (re-)use previous cases when conceiving architecture, and if so, to identify possible differences with the premise(s) of most CBD research. The interviews, each of which took approximately one hour, were tape-recorded and notes were made. Afterwards, we transcribed the tapes and identified common themes across the six transcripts.

Varied as these six interviews might be, it is obvious that the full range of approaches to architectural design is by no means covered by them. Therefore, we have supplemented the interviews with further material about architectural design, taken from various written sources, in order to collect sufficient evidence for later theoretical discussion. There are a variety of sources that can provide insights into how, whether and which cases serve in conceiving architecture. We have, for instance, personal accounts from famous architects that at least provide some outline of how previous projects may have influenced new designs. A source that has been particularly helpful in this respect is Bryan Lawson’s Design in Mind [Lawson, 1994]. In this book, major architects openly discuss their process of design, including sources of inspiration or other influences. There are also public presentations where architects have reported on the design thinking that went into their projects. At times these are more than rationalisations for the purpose of justifying a design and deal with what really happened. Finally, we have numerous attempts by historians and design critics to trace the roots and inspiration sources of major architectural projects.

---

3 See their biographies in the following section.
4 See chapter 2, p.54.
5 See appendix A. Due to lack of time, Paul Vermeulen did not receive the list in advance.
6 Since all interviews were conducted in Dutch, quotations have been translated as literally as possible to avoid any danger of our interpretation interfering with the meaning of what was said.
7 See chapter 6.
4.2 Biographies

Jan Delrue (°1939) studied architecture at the University of Leuven, where he met his future design partner Guido Konings. Immediately after his graduation as ‘Burgerlijk Ingenieur Architect’ in 1962, Delrue became assistant at the department of Architecture. One year later, he joined forces with Konings to found the architecture and engineering office Archiduk. One of their first realisations was an academic hospital in Pellenberg. Since then the office has established itself as one of the most consistent producers of hospitals, health centres and laboratories, in Belgium as well as abroad. In the meantime, Delrue studied urban design and construction at the Technical University of Delft (The Netherlands). He continued to pursue an academic career with research into rationalisation of the building industry, for which he was awarded his doctorate in 1969. Delrue is currently titular of several courses at the University of Leuven, including rationalisation, construction, deontology and architectural design. In addition, he is a member of various national and international organisations in the field of hospital design and health care.

One of Delrue’s former students is Mauro Poponcini (°1956). He graduated in 1981 as ‘Burgerlijk Ingenieur Architect’ and, one year later, as Master of Conservation of Historic Towns and Buildings. After four years of professional experience, Poponcini decided to combine practice with design teaching. He started as assistant at the department of Architecture in Leuven, where he became appointed professor in 1996, and held a visiting lectureship at the Academy for Architecture in Tilburg. In the meantime, he continued to work as practising architect, and in 1991 he eventually joined with his former fellow student Patrick Lootens to form Poponcini & Lootens BVBA. In addition to the two titular principles, there are currently ten permanent staff members and two free-lancers. The scale of their projects ranges from furniture, interior design, exhibitions and architecture up to urban studies. Several of their projects have won design competitions and/or attracted major design awards.

Paul Van Aerschot (°1938) studied architecture in Leuven as well. In 1961, as soon as he had graduated, he was accepted into the faculty of the department. At this moment, he is part-time professor, teaching design and a course on renovation. In his practice, too, renovation is well represented, as he has renovated several buildings of major historic importance: abbeys, colleges, an industrial complex, a medieval castle and (part of) a beguinage. Located in the centre of this beguinage, his office is – consistent with his teaching and practice – converted from an old house. Many of his realisations, renovation as well as new-built projects, were commissioned by the University of Leuven, where he has become known as home architect. However, Van Aerschot seems as much in his element designing a single-family house as he does when designing an auditorium or an academic library. In 1995, he joined with Hans Verplancke, one of his former students, to form Van Aerschot & Verplancke, which is currently four architects strong.

---

8 A five-year basic programme leads to the degree of ‘Burgerlijk Ingenieur’, equivalent to Master of Science in Engineering in the USA and Diplom-Ingenieur in Germany. This title is recognised and protected by Belgian Law for those who graduate from a Belgian University.
ARE ARCHITECTS NATURAL CASE-BASED DESIGNERS?

In the same year Van Aerschot finished his studies, Werner Van dermeersch was born. He studied architecture at the HAIR NHIBS school in Antwerp, where he graduated in 1984. After a practical training at the office of Jo Crepain, he started his own design office in 1989. His realisations range from the headquarters of a brewery, private residences and clothing shops, over furniture and design exhibitions, to music projects and multi-media installations. In addition, Van dermeersch has been teaching design both to architecture students and jewellery designers, first as assistant-teacher and later as appointed professor at various schools and institutes.

Paul Vermeulen (°1962) studied architecture in the same period as Van dermeersch, be it at a different school, the University of Leuven. Since his graduation in 1985, he has been working at the department of Architecture – first as a research assistant, subsequently as a studio teacher, and currently as a lecturer in architectural criticism. As a gifted critic and writer on architecture, he is one of the editors of the Dutch trend-setting architectural magazine Archis. However, Vermeulen is at least as well known for his designs as for his writings. Since 1989, he designs together with Henk De Smet, an intense collaboration that has produced an extensive portfolio of projects: private and social housing, office and university buildings, urban planning and renewal, not infrequently on thankless sites “in the pores of the Ugliest Country” [Loeckx, 1996 p.13]. In 1996, the work of Vermeulen and De Smet was awarded a major retrospective exhibition in the International Arts Centre deSingel in Antwerp.

Peggy Winkels (°1965) was educated at the Provincial Higher Architecture Institute in Diepenbeek, where she graduated in 1988. Immediately afterwards, she started her practical training at De Gregorio & Partners. Having become accepted in De Gregorio’s permanent team, Winkels is currently informed of every project in the office and mainly responsible for the design concepts. In the meantime, she has also cooperated with Groep Delta and, more recently, with the Libost-Groep for the ‘Green Boulevard’ project in Hasselt. Together with a landscape architect, urban designers and engineers, Winkels was closely involved in the concept and the spatial design of this prestigious urban project. Since 1993, she combines her work in practice with part-time design teaching, originally at the institute where she studied, and currently at the department of Architecture in Leuven.

4.3 Cases in conceptual design

4.3.1 WHETHER

The question is not new, but it is stirred up again by the recent application of CBR to architectural design: When conceiving architecture, where do architects begin? Do they draw inspiration from previous projects or not? The architects we interviewed answered this question almost unanimously in the negative. Yet, their answers were followed by an enormous ‘But.’

Peggy Winkels never looks directly at architecture. She admits that there are architects whose work is very inspiring, yet when starting a project, she always has her own outline, her own image first. It is only later in the design process, that she turns to
existing projects so as to refine her own concept. Mauro Poponcini and Werner Van
dermeersch fully agree with her: at their office, it rarely happens that previous projects
are explicitly consulted during conceptual design. Jan Delrue and Paul Van Aerschot
are more careful, realising that “Of course, you never know what you, almost
unconsciously, carry along in your head” [Delrue, 1998]. Yet, Van Aerschot is
nonetheless definite in his assertion that he never consciously looks for inspiration in
other projects. Paul Vermeulen, on the other hand, seems to know quite well what he
carries along in his head: “Other projects – own projects as well as projects you know
from the literature or from courses – are simply lodged in your memory, are part of
your overall culture, and you are influenced by them in a kind of all-penetrative way”
[Vermeulen, 2000]. He is the only interviewee who answers the question whether he
draws inspiration from previous projects with “Yes, of course” [ibid.].

Yet, if the others do not draw their inspiration from such projects, where, then, do
their concepts come from? According to our interviewees, concepts may have many
different origins. Some have to do with the programme, site or client, others with
music or making sushi, and some even with… existing design projects!

If any factor can be said to have more of a generative influence on concept
 generation, it is probably the functional programme, the purpose of the building. Without slavishly serving the functional principles of the Modern Movement, most
architects seem to start thinking about the programme: How does it work? How does it
function? What are the ingredients? [Poponcini, 1998; Van Aerschot, 1998; Van
dermeersch, 1999]. Van dermeersch accommodates these ingredients into separate
volumes, and spends a considerable amount of time arranging these into all possible
combinations.

Rather than abstract volumes, Van Aerschot immediately associates the programme
with what he calls an “almost constructive form.” His inspiration is nearly always
constructive, or constructional: “When I conceive a space, it is already almost
immediately enclosed by walls, there is a roof on it, there is a foundation underneath”
[Van Aerschot, 1998]. According to Lawson, this urge for concreteness early in the
design process represents a view strongly held by several of the architects he has
interviewed, in particular by Santiago Calatrava, Eva Jiricna, Robert Venturi and Ken
Yeang [Lawson, 1994]. Jiricna, for instance, contends to have little time for abstract
ideas: “It’s not an abstract process. I think that if you are a painter or a sculptor, then
it’s all very abstract, but architecture is a very concrete job.” A similar dislike of
designing in the abstract is expressed by Venturi, who prefers to confront what he calls
‘reality’ early in the design process. What all these architects are reminding us of here,
is that the reality of design, certainly in architecture, is bound by an intrinsic
materiality. More importantly, they confirm observations by other researchers that
many architects imagine new designs in considerable concrete detail, often much more
detail than needed for conceptual design [Eckert & Stacey, 1999]. According to Eckert
and Stacey, this untimely high level of detail indicates that designers draw design

9 Also in circles of well-known architects, the programme appears to be a significant source
of inspiration. For Michael Wilford, for instance, the early generators of design ideas have
very much to do with the function of the building [Lawson, 1994].
ideas from concrete examples, so that what they imagine are variations on previous projects rather than original creations.

Apart from the programme, other factors that frequently feed concept generation are the client and the site [Delrue, 1998; Poponcini, 1998; Van Aerschot, 1998; Van dermeersch, 1999]. Van dermeersch stresses the prominent part played by the client during conceptual design: he acts as sender, as nuisance, as (gold) mine of ideas. Hence the need to spend enough time in getting to know the client, his limitations, expectations and dreams. This view of the client as source of inspiration is very similar to that expressed by Robert Venturi and Denise Scott Brown: “We get some of our best ideas from clients, we love collaborating with them” [Lawson, 1994].

The building site, for its part, may inspire by hosting an impressive tree, offering interesting views, or simply by its geometry. Even an at first sight uninteresting, banal site may figure as primary generator in the act of designing. As already announced in his biography, Vermeulen and his design partner do not recoil from projects on downright thankless sites. Their repertoire comprises a few minuscule alterations of extremely narrow terrace houses. According to Vermeulen, many contemporary architects – suppose they accept the job in the first place – only aim at completely erasing the original situation, to substitute it by something that is architecturally correct, by analogy with politically correct. In the designs of Vermeulen and De Smet, however, this ‘banal’ situation often acts as the very starting point. This situation does not only comprise the site, but also the type of a traditional terrace house: a front door, two rooms after one another, a corridor with the stairs and – last but not least – a typically Belgian ‘extension’. Their trademark seems to lie in respecting this typological pattern and yet altering the situation completely, so that the resulting design nevertheless feels entirely new and contemporary.10 In their opinion, traditional types possess perhaps small, but nevertheless existing qualities, and it would be a pity to throw out the baby with the bath water [Vermeulen, 2000].

Rather than typology, Van Aerschot considers basic geometry an excellent starting point, especially for our Western way of building.11 In essence, we build by assembling orthogonal elements – columns, beams, rectangular plates, etc. – and “if you follow geometrical principles, it is easier to hold that under control. Whereas if you work in a very organic way, then you still must keep it upright, then you need an engineer to tell: ‘How do we keep it in the air?’ When my design is finished, the stability is already solved too” [Van Aerschot, 1998].

Delrue, for his part, refers to many different sources of inspiration, beginning with small-scale urban fabric – the density of an Italian city, for instance – through a deep fascination with climatology – “Why is it coolish here inside? Why is it chilly out on the street?” – and the fantastic possibilities of atria [Delrue, 1998]. In fact, judging from the stories of our interviewees, virtually everything can inspire concept generation: dance, music, film, photography, ... “Yes, everything, I mean everything:

---

10 The high typological sensitivity in the work of Vermeulen and De Smet was already brought up in chapter 1, p.29.
11 A similar view seems to be held by Richard MacCormac. If any source of inspiration can be seen throughout MacCormac’s work, it is basic geometry [Lawson, 1994].
MTV, … well, everything, you mention it. And I think that it is actually very important to be busy with everything indeed, or be busy with everything as much as possible” [Van dermeersch, 1999]. The Flemish ‘master builder’ Bob Van Reeth fully agrees with him: “It is being incredibly influenced by everything I see. And also: wanting to be influenced. It is stealing and nicking everywhere. It is: giving chances to life” [Luyten, 2000].

Yet, if everything can be a source of inspiration during conceptual design, does that not include concrete cases too? Although most architects doggedly denied relying on previous projects during conceptual design at first, many of them were forced to reconsider their initial reply in the course of the discussion. A distinction can be made here between cases designed by other architects and projects from their own repertory.

4.3.1.1 Cases from colleagues

Asked where his concepts come from, Poponcini lists an almost infinite array of inspiration sources, at the end of which it slips out: “and sometimes it might even happen that you are inspired by other designs, for instance, by Diener & Diener or Herzog & de Meuron” [Poponcini, 1998]. Van dermeersch admits to be inspired by the work of Diener & Diener too, in particular by their experiments with structures, transparency or materials. As if by way of excuse, however, he immediately adds that this probably happens to every architect, including Diener & Diener themselves. What they design comes from somewhere too.

Apparently, Swiss architecture meets with general approval these days, for in the discussion with Winkels it pops up as well. She acknowledges that there are projects by Herzog & de Meuron, Alvaro Siza or Renzo Piano which she could fall back on for almost every design. In the course of the discussion, however, she calls herself “an enormous scatterbrain in these things” [Winkels, 1998]. What she seems to mean by this, is that although she greatly admires the work of the Swiss, Siza and Piano, it is in fact projects by other architects from which she learns most – architects that are less in vogue or with whom she would not like her work to be associated at all. Aldo Rossi’s work, for instance, is absolutely not what she would design herself, but she has to admit that she learns a lot from it.

This somewhat paradoxical stance reminds us, to some extent, of a striking account by Santiago Calatrava. In discussing his major influences and interests, the architect

---

12 It would be misleading, however, to suggest that all interviewees share this view. Van Aerschot, for instance, sees some parallels between music and architecture, yet does not consider music a source of inspiration: “There are parallels, you recognise it, and you can follow it from an architectural outlook, but it is not that you can use these things. Especially not music: it’s volatile … that doesn't work.”

13 According to Poponcini, this ‘somewhere’ is often simply ‘in the wind.’ Even if two projects look alike, that does not necessarily mean that the one is inspired by, based upon or copied from the other. Society changes and it is quite normal that a number of architects react to these changes in a similar way. Architecture is always a child of its time, if only by technology. Van Reeth speaks in this respect of “projects (…) that are just in time,” projects that propagate something of the aspirations of today, rather than repeating once more those of yesterday [Luyten, 2000].
ARE ARCHITECTS NATURAL CASE-BASED DESIGNERS?

explicitly refers to the work of Louis Kahn, with whom his work is not normally associated by the critics. As Lawson contends, “It is probably an interest in his concern with the nature and inherent characteristics of building materials which he shares with Kahn rather than the issue of form” [Lawson, 1994].

Winkels’ hidden admiration for Rossi is not inspired by formal issues either. Instead, she lays great emphasis on his ability to solve “small human things” in architectural design. She describes how, when she was a student, backdoors were totally taboo. A backside had to look equally public as a façade, and a door was preferably disguised as a sliding surface. Today she realises that it is actually very nice to have a backdoor, a place where you can hang your coat and pull off your dirty shoes. The rough outline of a design is conceived by Winkels herself, but these small human things derive from the work of Rossi and are translated into another language. The form does not play a part. It is the idea, the concept that counts [Winkels, 1998].

A strikingly similar view is expressed by Delrue. At the time of the interview, he was involved in a competition for the conversion of a hospital. Pointing at the plan of another hospital building, lying on his desk, he contends:

“Why does this lie here? Well, that is for this concrete thing, because here I know that one develops a certain kind of semi-intensive-care units. (…) And I’m going to apply that in the project I’m currently working on. It will be a totally different shape (…) but I mean, of course I’m going to draw from the very extensive information I have. You look for analogies with, but it’s not in the sense that I’ve seen something nice, I’m going to copy it, not that” [Delrue, 1998].

It seems to be a fundamentally important point that is being made here, a point that might serve to refine the model of Case-Based Design. Although they consciously draw inspiration from existing designs, all three are suggesting that they do not take over the formal characteristics of those designs. To say it in Delrue’s words,

“I don’t like that shape, it should be different, but the idea behind it, the underlying idea, that does inspire, yes” [Delrue, 1998].

What we see here therefore is that cases may fundamentally affect the concept behind a design, yet without recognisable implications on what the resulting building looks like. Calling in cases does not, by definition, mean copying.14

Moreover, cases that are called in at first might very well be dropped later on in the design process. As Vermeulen contends: “In fact, it goes on a very high tempo. Throughout the entire design process, you are constantly drawing comparisons with examples you know (…) which you suddenly think relevant for what you’re doing yourself. But it’s almost like the tide: these references come and go, some hang on, others are dropped because they turn out to be not that relevant” [Vermeulen, 2000].

---

14 Of course we have three fairly experienced architects here, and it may be that designers with less experience pick up the more superficial aspects of existing projects. Van Aerschot, for instance, complains about students blindly copying shapes from other designs. See also the discussion on cases in design education (pp.119 ff).
CHAPTER 4

The projects he refers to are not necessarily examples of Architecture with capital A, like the ‘well-known Swiss’ mentioned above, it might be anonymous things as well.

Whether anonymous or not, in part the frequent use of references by Vermeulen may be due to the fact that he does not design alone, but always in close collaboration with De Smet. During design, they call in concrete projects as a means of communication, of conveying things to each other: “For instance, while you’re working you could say: ‘I find this or that idea inappropriate, it’s too much project x, and what I like better is project y,’ while these projects x and y, well-known projects, might be named never again afterwards. Once that slope or that obstacle has been taken, you simply may not need them anymore (…) It’s indeed a sort of exact change to clarify in a circumscribing way which aspect you mean exactly without having to tell a very long tale” [ibid.].

Another factor of influence may be the course on architectural criticism Vermeulen teaches. The preparation of this teaching perhaps makes him read more than the average architect, yet he doubts whether this affects directly what he designs himself. Sometimes, things he can explain very well to students, are highly unlikely to have a direct application in his work, for various reasons. He realises, however, not to master how his own frame of reference evolves: “I think that we don’t do ‘no matter what’, and that some things will never come in, but even then it can happen that although things are faraway from yourself as a whole, you still appreciate aspects of it” [ibid.].

While Vermeulen acknowledges cases to play a prominent part in the design process – “During the way you cover, it’s as if you run into old pals” – it rarely happens, perhaps not counting exceptions, that an existing project in its entirety figures as starting point of a design. Do such exceptions occur in his own repertoire? “Well, … yes. If I say now ‘No’, you could say of course: ‘And over there then?’” [ibid.]. As it turns out in the course of the interview, ‘over there’ refers to two ’youthful lapses’: an office design for Apple in Brussel, and the Vanderbeck-Arnold house in Sart-Messire-Guillaume.15

4.3.1.2 Personal projects

A second way in which architects seem to use previous cases, is by embroidering on personal projects, i.e. projects they have designed themselves. Van Aerschot, for instance, acknowledges to be often inspired by older work. In his opinion, each project in part relies on the previous one, and thus is actually wrapped up in an evolution. Van dermeersch describes this evolution as follows:

“Every project is a step forward, and each time there are risks involved, on technical, constructive, material or whatever level. Each time you learn something, and one time you’ve maybe gone a little too far, or not far enough, so that the next time you know: I must absolutely not do this ever again, or here I can still go much further, and I should go much further” [Van dermeersch, 1999].

15 See frames on p.109 and p.120 respectively.
A villa Savoye for Apple

Once Vermeulen and De Smet designed an office building for Apple that had an explicit relationship with Le Corbusier’s villa Savoye. How did it come to this relationship? “In fact, we were making that design, and at a given moment (...) there are a sort of general ideas of how it should become more or less, conceptual ideas, and you try to sketch it a bit, and you start noticing – because you never sketch that very well, of course – this looks like the villa Savoye…” Although it was somewhat funny, initially the architects offered resistance, “because you think: ‘we simply can’t do that.’” Yet, when thinking it over more deeply, they finally started realising that the resemblance was more than coincidental and thus a reference to sustain: “On a conceptual level, we were actually making a great big villa Savoye: a ground floor tailored to the car, then a spacious residential platform – a sort of show-box towards outside and apart from that very strongly introverted – and further free volumes on the roof for other parts of the programme that were not repetitive.”

At the moment they recognised their design as being the villa Savoye, the reference became in fact productive.

Unfortunately, the Apple office only reached the stage of preliminary design, and it would be interesting to know the sequel of the story if they had been allowed to build it: “because I can very well believe that this reference to the villa Savoye – which was very helpful at that specific moment, because it actually gave us insight in what we were doing – that it would gradually fall off and become less obvious.” The colour scheme, for instance, was quite Le Corbusian, but it is highly likely that it would change completely, or even disappear.

Eventually, the design has only been polemic and the architects really liked to turn the office very clearly into a villa Savoye. Doing so, Vermeulen felt himself a bit like Inigo Jones, the great expert on and admirer of Palladio, who deployed a neo-Palladian style for designs that solved totally different problems. “At that time,” Vermeulen remembers, “I had the feeling that what we were doing actually looked somewhat like that.”

‘That time’ refers to the late 1980s, a period when – especially in Flanders – a neo-modernist idiom had not very well crystallised yet. A few years later, however, this idiom was already so widely accepted, that if the architects had to build the design, they probably would have dropped it. Vermeulen: “You should be extremely careful, for something that seems a fresh reference at the moment you think it up, may already be so chewed-up and worn-out at the moment you must build it, that you better distance yourself from it.”
Further evidence for this ‘stepping-stone theory’ is found in Winkels’ way of working:

“For instance, I tend to place windows much deeper into the wall. I usually take thicker walls because I find shadowing very important. And I always take this one step further in order to search how I can solve it in a technically sound and yet not too expensive way” [Winkels, 1998].

In fact, projects that embroider upon a theme started in a previous project turn up in the repertory of many architects. Take for instance Alvaro Siza’s recent Museum in Serralves. In this project, diverse aspects of museum designs from the past can be found. One aspect is the ‘turned table’ solution Siza already introduced in the Centro Galiziano in Santiago. With an eye on an optimal diffusion of day- and artificial light, the centre of the ceiling in the exhibition rooms is lowered, which gives the impression of tables that are turned upside down. In this way, Siza exploits previous projects, not as sources of citations, but as raw material for new designs [Dubois, 1999].

At the time of the interview, Vermeulen and De Smet were struggling with light as well, be it in an auditorium instead of a museum. The auditorium is to become part of an office building in Aalst, and is largely comparable in terms of size and situation to the one they recently designed for the agriculture institute in Leuven. According to Vermeulen, this previous auditorium may serve as a sort of leg up in two different ways. Either you say: yes, we can draw inspiration from it, it may be of help. Or you can say: I found this or that specific aspect not very successful after all, I’d like to do it differently. For instance, both auditoriums are supplied with natural light that can be darkened. The one in Leuven has a glass ceiling, above which both a roof light and artificial light are situated. Yet, due to the glass, Vermeulen has noted, an essential aspect of the space is lost: people usually do not experience that there is natural light at all. “This observation will probably stop us from applying a glass ceiling in that auditorium in Aalst, so that (...) you do realise that it [the skylight] is there” [Vermeulen, 2000].

Willem-Jan Neutelings calls such new designs, which build further on their predecessors, ‘spin-off’ projects [Neutelings, 1998]. Within the context of a competition design, he got the idea to conceive the headquarters of a bank as a glass bell. By putting a glass skin over the entire building, he could create a Mediterranean climate between building and bell. Although the bank project itself did not win the competition, and thus was not built, the underlying concept was further developed in several ‘spin-offs,’ ranging from an apartment building over a printing office to an office building at Schiphol airport.

A similar step-wise evolution, be it a much longer one, characterises the repertory of Delrue. The clarity with which he sketches this evolution is impressive and the precision with which he traces back his design decisions, shows the application of enormous experience. Delrue’s schema of the relation between his hospital designs clearly illustrates this point (see figure 4.1). Pointing at this schema, he contends:

“Indeed, this project is not thinkable without this (...) so as I proceed, each following project is tributary to the previous, for there is a logical evolution.
Figure 4.1. Relationship between Jan Delrue’s hospital designs.
Possibly there are others, but one can find a logical thread in this sequence of projects” [Delrue, 1998].

Being both a theoretician and practitioner in the domain of hospital design, Delrue suggests that his practice should not be seen as resulting from his theory, but rather the other way around. The thread that apparently runs through his repertory does not result from a preconceived theoretical agenda, it originates in and is directed by the concrete projects themselves. It is only afterwards, when revisiting his buildings and reproducing them, that he was able to detect a supposable story line. What Delrue seems to be telling us here is that, even in a highly specialised field like hospital design, the evolution of knowledge is driven not so much by theoretical reflections, but rather by concrete design projects.16

According to Van Aerschot, however, it is not so much the project, i.e. the design product, but rather the approach, i.e. the design process behind it, that drives this evolution:

“It’s not that you say: ‘I’m going to use that form again, for it found favour. I’m going to build something like it again.’ I simply can’t. But the approach, that you say: ‘Look, with this approach I can …’, which you try out with other programs” [Van Aerschot, 1998].

He illustrates this point with a series of single-family houses he designed some years ago (see also p.114). At a given moment, he had adopted a particular approach, which was based upon constructive and geometric considerations. In the next house, he decided to continue with the same approach in order to verify its usefulness within the context of another site, brief and client. “So, in that sense, you draw your inspiration from your older work, but usually as a continuation of a process that actually transcends one single project, yet over a longer period.”17

---

16 A similar view is expressed by Venturi and Brown when talking about their famous expedition to Las Vegas. What they learned from this trip, first showed up in their design projects. It was only afterwards, when analysing their own work, that they were able to formulate the lessons about symbolism that eventually resulted in the book Learning from Las Vegas [Venturi, Scott Brown & Izenour, 1972]. This story seems to chime with studies in psychology about the nature of implicit learning (see chapter 1, pp.11-12). Some characterise this phenomenon as an indirect way of learning: “People possess representations of the events they have experienced, and those representations drive future performance. However, people do not have direct, conscious access to those representations. Instead, when a stimulus [e.g. a new design task, AH] is encountered, they can only observe their performance and attempt to judge the likely source of their behaviour [e.g. an expedition to Las Vegas, AH]” [Whittlesea & Dorken, 1997 p.64]. Moreover, the view of a design as a way of developing instead of demonstrating theory echoes the contention that design knowledge/knowing is actively developed in a particular design context (see chapter 1, pp.12 ff).

17 Van Aerschot describes here a sort of ‘super’ design process that transcends the immediate urgencies of a single project, and that is governed by his personal agenda. Ken Yeang makes a similar distinction between, on the one hand, the project design task itself and, on the other hand, a more general design process that continues to be informed by each project [Lawson, 1994]. This general process may pursue quite specific objectives, as is illustrated by the work of Calatrava. His repertory, which contains projects of various scales and for many purposes,
ARE ARCHITECTS NATURAL CASE-BASED DESIGNERS?

Whereas CBD researchers would consider Winkels’ and Vermeulen’s approach examples of structural adaptation, what Van Aerschot is describing here sounds more like derivational adaptation.¹⁸ Unlike structural adaptation, which recycles the end product of a previous solution, derivational adaptation re-runs the process that generated this product. Judging from Van Aerschot’s description, the latter type of adaptation does occur in the design process of real-world architects, be it for a slightly different reason than CBD researchers assume. As Van Aerschot contends, he re-uses the approach of a previous project not so much because it went down particularly well, but to fine-tune the approach and explore its applicability for other projects. The CBD view on derivational adaptation is further nuanced by Delrue, according to whom a prerequisite for such derivation is to have gone through the original process yourself:

“If we had left it at a visit to another hospital and if we had seen that, then we hadn’t understood it (…) You must have struggled with it yourself. You must know afterwards what doesn’t work” [Delrue, 1998].

4.3.2 WHICH

In general, CBD researchers rarely distinguish between ‘second-hand’ and personal cases, i.e. between projects designed by others and self-made designs. The architects we interviewed, however, did make an explicit distinction between both. Which one prevails, is not so much a matter of preference, but seems to evolve over time. According to Van dermeersch, this evolution inevitably starts from colleagues’ cases and ends with personal projects:

“As an architecture student, first you’re going to look at masses of things, ranging from, yes, Plato’s cave to the latest building by Rem Koolhaas or something like that, and you take a bit of everything … And at a given moment, you start filtering, in fact, you say: ‘Yes, this and this and this I find interesting,’ and then you see in fact, you get a sort of pyramid of interests that becomes narrower and narrower and narrower … until, at a given moment, you stand in the book shop and it turns out that there’s actually nothing left that (…) still can fascinate you. And then it’s like: yes, now you can only do it by yourself, and nobody else can serve as an example. That’s an evolution I have gone through” [Van dermeersch, 1999].

This evolution might explain why Vermeulen designed the Apple-villa and the Vanderbeck House very early in his career: “In my opinion, it actually had to do with the fact that we didn’t have a repertory ourselves. For as you start doing more, the previous projects you’ve made are going to weigh more heavily, and simply all the experiences they entailed become more important as well, [so] we didn’t feel that great need anymore to tie in so literally with an existing building” [Vermeulen, 2000].

¹⁸ See chapter 2, p.49.
Four houses and an auditorium

In the early 1970s, Van Aerschot was asked to design a single-family house for the family Dupré. He started from a square floor plan – “the building site (…) was more or less a square, then you don’t have to seek something else” – and put a pyramid-shaped roof on top of it. By placing the pyramid diagonally and cutting of the corners, the surface could be halved on the first floor within the same structure.

Later on, he would start from an analogous structure in three other houses – house Schreus (1977), Sabbe (1980) and Carmeliet (1989) – yet giving it each time a different interpretation: one house was flat, the other built like a spiral, and so on.

In Van Aerschot’s opinion, elementary geometric figures and volumes – square, pyramid, circle – “are essential, timeless components, inherent to the physics of building, the instrumentarium ‘par excellence’ for the organisation of space, architectural composition and constructive detailing.” At first sight, this attitude might seem somewhat conservative, opposed to, or at least sceptical about every innovation or artistic inspiration. Yet, “the building tradition is not a historic relic, but a living “organism” that needs a basic knowledge and insight, or to say it with a great word, moral to evaluate its development and innovation.”

In fact, Van Aerschot gradually developed this view and rational approach to architecture not only in these four houses, but throughout several larger private and public buildings as well. Eventually, it would culminate in the Pieter De Somer auditorium, an exceptional occasion to put forward ‘ex cathedra’ his view on the essence of building. In this respect, the four houses on the left can be considered preliminary studies of the basic concept of the auditorium below.
ARE ARCHITECTS NATURAL CASE-BASED DESIGNERS?

Unlike Van dermeersch, however, Vermeulen clearly experiences this evolution as endless, in the sense that personal projects never take over completely from others’ work: “They are not closed worlds (...) there is still the huge field of architectural culture on which you draw, but closer to you a few objects have been added, which you know better simply because you’ve made them yourself. Yet, your own architecture culture evolves too: you see new things, or others you used to be enthusiastic about, are not that helpful anymore now. So your own culture evolves along as well” [ibid.]. Vermeulen is not only aware of the fact that his culture evolves, but also how: there are new things he suddenly refers to; or other things he used to refer to and doesn’t find interesting anymore; or things he already knows for a long time, but never found worthy of note or didn’t dare to do, and which suddenly become interesting. What Vermeulen is describing here cannot but remind us of the dynamic way in which human memory changes, as claimed by CBD’s cognitive model. More precisely, it shows striking similarities with what CBD researchers call case acquisition and re-indexing.

Whether endless or not, Van dermeersch and Vermeulen both have gone through an evolution from calling in others’ designs to relying on personal projects. Apparently, however, other architects have not undergone this evolution, or at least not the entire one. Van Aerschot, for instance, seems to have skipped the first phase, since he explicitly denies the influence of other architects in his student projects:

“Even as a student already, I have never really wanted to emulate, or follow architects. And it went even so far that, if I found something and saw it afterwards in a magazine in another architect’s work, I erased it automatically. It didn’t matter to me anymore” [Van Aerschot, 1998].

Yet, when discussing his experience as design teacher, and trying to remember: ‘How did I design when I was a student?’, some sources of inspiration nevertheless turn up.

---

19 Can these closer, self-made objects be viewed as a sort of selection or synthesis of that larger field? Precisely because of the evolution of this field, Vermeulen seriously doubts whether that is correct: “Well, I myself would not be able to do that, I think. But assume you’re up against an extremely sharp critic who really has studied your work very well, even then I ask myself whether it would be possible to derive simply from our projects what our frame of reference is, because there are always break-ins”[ibid.]. Deleu’s critique on the Vanderbeck house (see frame p.120) illustrates indeed how difficult it is to detect (all) references in others’ designs.

20 See chapter 2, p.50.

21 After more than 20 years of experience as practising architect, Pascal Cottenier reacted in exactly the same way when designing his own house [O’Seery, 2000 p.74]. Originally, Cottenier wanted to give his house – an alteration of an existing terrace house – a quite eye-catching façade. He describes it as “a façade with a zip fastener on the front.” “Ground and first floor remained unaltered, and from the following floors on, the façade opened like a zip.” When submitting his request to build it, however, the Province made a problem of this façade. After 15 long months of waiting, Cottenier suddenly abandoned the design: “I’d just obtained the permission to build it, when I discovered an identical idea in an architecture book about Berlin: a house with a zip on the façade. At that stage it didn’t matter to me anymore, I’ve made a new drawing.”
CHAPTER 4

Whereas his co-student 22 was a severe Mies-adept, Van Aerschot himself preferred the more sculptural work of Le Corbusier. Afterwards, he got interested in projects by Scandinavian architects like Aalto and Jakobsen, which were also more sculptural, but simpler. “And you draw your inspiration from that, not to imitate, but you want to make a building that fits within that spectrum.”

4.3.3 WHY

So far we have tried to find out whether architects use design cases during concept generation and, if so, which cases they use – cases from others or cases designed by themselves. A third question we have tried to answer is why architects decide to draw their inspiration from a specific project, in other words, why they select a particular project. In CBD jargon, the conventional shorthand for this is case retrieval and selection. CBD researchers assume that architects select a particular project because of its relevance for the current design.23 In CBD systems, being relevant is translated into showing striking similarities with the current design situation,24 and if possible, having lessons to learn for the design at hand.25 The architects we interviewed, however, sketch a rather more subtle and realistic picture of case selection than can be found in most CBD literature.

Asked why they choose to rely on a particular project, the shortest, and perhaps most honest reply came from Poponcini: “Often simply because I feel like designing something like it” [Poponcini, 1998]. Confronted with her colleague’s confession, Winkels admits that she knows the feeling:

“Yes, that’s true too. That you feel like: ‘Now, I want to design something like this. Can’t I use this in my project?’ It works that way too” [Winkels, 1998].

A clear example of the way it works, is given by Delrue. In discussion, he reveals a great admiration for a new hospital in The Netherlands, which he calls ‘the cathedral’. The building is so impressive, he contends, that he can hardly wait for an opportunity to draw his inspiration from it:

“Oh, it’s a feeling I have. The next time a new hospital is needed in Flanders – perhaps I won’t live to see it – then I’ll definitely remember it. (…) I think I’ll immediately drag the client to this hospital” [Delrue, 1998].

As far as Poponcini is concerned, the selection of a specific case is not directly programme-bound. At his office, it might happen that a cultural centre inspires the design of a house or vice versa. The greater the difference between source and target project, the greater the challenge. Take for example Stellwerk 4 Auf dem Wolf, a signal tower of the Swiss Bundesbahn in Basel (see figure 4.2). Given a rather simple programme, Herzog & de Meuron conceived the building as a copper cabin, a clean

---

22 At Van Aerschot’s student days, only two students of his year studied architecture: Van Aerschot himself and his co-student. Officially, they were three, but the third one never showed up.
23 See chapter 2, p.54.
24 See for instance case retrieval in FABEL (chapter 3, pp.73-74).
25 See for instance Archie-II (chapter 3, pp.61 ff)
ARE ARCHITECTS NATURAL CASE-BASED DESIGNERS?

cube with a façade of metal sheets. To Poponcini the challenge is now to try out this clean concept with a more complex programme, for instance the design of a house. He firmly believes architectural design benefits from this kind of cross-fertilisation between dissimilar projects, and therefore, finds it very important that a design office can cope with various programmes.

Figure 4.2. Stellwerk 4 Auf dem Wolf in Basel (Switzerland) by Herzog & de Meuron, 1992-1995.

Delrue, on the other hand, has opted for a totally different strategy. His office almost exclusively deals with the design, extension and renovation of hospitals. Yet perhaps this should not surprise in a highly specialised and rapidly developing domain like hospital design. Precisely because of this rapid development, Delrue tries to keep track of hospital buildings all over the world, and thus draws his inspiration only, or at least primarily from cases with the same programme.

Vermeulen and Van Aerschot find themselves somewhere in between. While the Vanderbeck house and auditorium in Aalst follow Delrue’s strategy, the Apple-villa is more in line with Poponcini’s preference. Van Aerschot agrees with Poponcini that source and target project do not need to have the same function, yet adds in a commonsensical way: “When the programme is the same, it’s easier of course.” In general, however, his selection does not seem to be inspired by programmatoric considerations. As already mentioned, Van Aerschot frequently draws his inspiration from older work, for instance, to investigate whether the approach adopted in one

26 Non-architects might perhaps be surprised by this example, yet most architects will agree with Poponcini that a house is one of the most complex programmes to design: “You should be able to ventilate and to look out of a window in your office, and in the bedroom too, and to swing the lettuce near the kitchen, etc. So (...) it’s a very complex programme, a house.”
project also applies to other designs. Often, however, it is not so much because this project worked particularly well, but rather the opposite, because he is not completely satisfied with the result:

“Each building should be made twice, or three times. So when it is finished, you say: ‘Look, if I had to start again, then I would do it like this, or like that, or that seems superfluous to me. I’d rather continue working in that direction’” [Van Aerschot, 1998].

Yet, what does all this teach us about case selection in real-world design? Judging from our interviews, case selection is a conscious course of action, but one that is already in progress prior to any particular design task. The decision to rely, fall back or embroider on a specific case seems to be taken when confronted with the case itself rather than with a new design problem.27

4.4 Cases outside conceptual design

With respect to other stages in the design process, other than concept generation, architects seem less reluctant to admit the support of previous projects.

Winkels, as we mentioned, first conceives her own idea, and it is only when this concept has taken shape, that she calls in cases to refine it.28 She seems to invest a substantial amount of time and effort in this refinement, in clarifying what the design is going to look like, and often seems to achieve this through a collage. By collating images from various sources, she tries to visualise the atmosphere she wishes to create.29 Although these images may come from diverse directions, it frequently happens that they depict existing buildings – an old gallery or a monastery, for instance. Everyone recognises the atmosphere in this kind of buildings, and she likes to incorporate that in her projects.

Vermeulen, for his part, seems to call in existing projects throughout the entire design process. Very often, when he asks himself ‘What would this bring?’, he simply

27 At first sight, this does not seem to apply to the Apple office by Vermeulen and De Smet. The architects did not decide to draw on the villa Savoye after all, it simply showed up in the sketches they were drawing. Fortunately, the reference turned out to be not completely coincidental, in Vermeulen’s opinion because the office had much in common with Le Corbusier’s famous work. Yet, perhaps the reference was not completely coincidental for another reason as well. In their search for a neo-modernist idiom, which at that time had not crystallized yet, the architects may have wanted to make something that would fit within the modernist gamut. Doing so, they might have been drawing almost automatically, c.q. unconsciously things that looked like the Villa Savoye, as it represents one of the milestones in this gamut.

28 A similar distinction between invention and refinement is made by Robert Venturi: “For Venturi ‘invention’ involves originality and the ‘what’, whereas ‘refinement’ involves development and the ‘how’. He considers both of equal importance in the creative process (…) and feels that recently invention has been encouraged at the expense of refinement, resulting in a consequent loss of quality in design” [Lawson, 1994].

29 It seems to be a technique to better understand an emerging idea or design concept. Once visualised in this way, she can draw all implications inherent in this idea and work them out. It also serves to communicate her design to others, for instance the client.
ARE ARCHITECTS NATURAL CASE-BASED DESIGNERS?

goes looking for it. For instance, during the design of his own house, an alteration of an existing building, the question arose whether a glass front door would fit the façade of an old house. In order to answer this question, Vermeulen simply went biking in Ghent, until he had found various glass front doors. “Of course it’s never exactly the same, but you can ask yourself: ‘Does it fit in our house or not?’” [Vermeulen, 2000].

Others find cases useful sources of layout solutions for specific programmes. At Poponcini’s office it can happen that, when designing say a home for the elderly for the first time, they study a number of examples, both good and bad ones, in order to learn the ingredients of the programme as well as useful layout solutions.

Also in relation to technical aspects, cases seem to be consulted from time to time [Poponcini, 1998; Van Aerschot, 1998; Winkels, 1998].

“Well, not often really, but it does happen that (...) you browse through Detail, because you want to make a glass façade, and that you have a look at how Jean Nouvel does it” [Poponcini, 1998].

Van Aerschot uses previous projects as a reference for materials, colours or details, but actually finds these details only secondary considerations. According to Delrue, cases play a part throughout the entire design process. In every single stadium, there are projects from the past that he consults, must look up, or wishes to see again. For this reason, his office has an impressive ‘case base’ filled with information on hospital designs. Part of this information has already been assimilated, yet an enormous lot is still raw material, and is only processed in response to a concrete design task [Delrue, 1998].

4.5 Cases in design education

Case-Based Design, as mentioned before, assumes a cognitive model that intimately integrates reasoning, remembering and learning. Of particular relevance to an investigation of real-world CBD is therefore the use of cases in design education. In general, expert teachers draw on a considerable fund of knowledge about teaching, in addition to knowledge of the subject matter [Shulman, 1990]. Given their ample experience in design teaching, our interviewees are thus particularly well placed to shed more light on the role of cases in learning (to) design.

30 Once again referring to the hospital plan on his desk, he contends: “It’s the first time that I colour this, because now I want to know: ‘but, where are their utilities here?’ While in the past I’ve looked at that nursing unit a few times, like…and that’s the case for an enormous lot of things.”

31 See chapter 2, p.50.
An early Mies in Sart-Messire-Guillaume

In 1990, Vermeulen and De Smet were asked to design a house for the family Vanderbeck-Arnold. They had just designed the Apple-villa, and were entirely locked up in a neo-modernist idiom. If there was something clear about this project, however, it was that a flat roof was completely out of the question: “In fact, all cinq points of Le Corbusier’s architecture were strictly forbidden, by town planning as well as by the client himself, because he showed up with a totally different frame of reference.” The architects had due difficulty to get started – “a sort of writer’s block, to name it like that” – since everything they drew was too tributary to that high-modernist architecture.

And then, by browsing and talking it became clear that both of them knew the Riehl house in Pottsdam, an early design by Mies van der Rohe. Both had always been fascinated by one specific image: the façade looking over the lake. When studying that house in more detail, however, it turned out not to be applicable to their design at all, if only because of the geography of the situation. Moreover, “it wasn’t such a good house either, because that one strong image and the rest of the house seemed to clash, but… that one image, of the façade looking over the lake, over there in Pottsdam, that kept haunting us.” Eventually, that image released them from their writer’s block, filling up the black hole in their idiom. From then on, everything went much easier: totally different things came to the surface, references to Adolf Loos or villa’s in Knokke-le-Zoute, “and everything became relevant simply because we had found the key to enter, and the key was indeed that house by Mies.”

In some sense, Vermeulen admits, this reference cost them a bit courage, as the architects felt to take a road not everyone would appreciate.1 Nevertheless, they didn’t take trouble anymore to filter it out, or erase it again, also a bit as a statement: “Actually we thought that, if the Barcelona-pavilion may be imitated, why would that former Mies not qualify for it? It was also true, it really had been inspiring for us (…) to work backwards like that, to see (…) in reverse order where he [Mies] comes from and in this way actually arrive at a point that we still had in common with him, but that was now more relevant to us than what he had done afterwards.”

At that time, Luc Deleu – a colleague architect and critic – reproached them that it was after all ridiculous to fall back on “Mies before he was Mies.” Vermeulen still seems to chuckle at the very thought that Deleu did recognise that one point, but not the entire re-editing, let alone the many other references in the building: “He was only able to place the most recognisable image but none of the other. That did reassure us, that this house indeed could pass for original and not for a decoction of that coincidental house.”

1 Eventually, their presentment proved correct: “No one in Flanders knew what to say about the intriguing villa (…). A finger exercise? A misunderstanding?” [Loeckx, 1996 p.11; italics added].
ARE ARCHITECTS NATURAL CASE-BASED DESIGNERS?

Asked whether students draw on existing projects for concept generation, the general tenor of the replies can be summarised as: “Of course.” Even Van Aerschot, who denies to be influenced by others in his own student time,32 finds it quite normal that students look at examples. They must draw their inspiration from somewhere, and when they start, they have not yet built up a personal baggage.

Poponcini likens design students to children seeing their first film or reading their first comic or book. “If they make a drawing afterwards, it is always about that because they want to assimilate it. They are thinking about it as if it were unique. They want to express and use everything they know while actually they still know very little.” Similarly, if students have seen an architectural project for the first time, the odds are that it will leave tracks in their own design.

This might seem typical of weaker students, yet all interviewees are explicit in their assertion that it happens to the best of design students. The only difference Winkels has noted, is that the better students come up with better examples. According to Van Aerschot, they are also better at interpreting. “For the others, it’s often copying, and in a volatile way.” Many students tend to adopt forms, shapes or materials from other projects, and there is nothing wrong with that, provided they know why they choose a specific example and draw the full consequences of their choice. What is very difficult to students, he assumes, is to separate the sheep from the goats: “I don’t blame them for adopting things, but for adopting them without question, without reflecting.”33

This superficial form of CBD might explain why few interviewees are sympathetic to stimulating case use by design students. Whereas all six find it normal that novice designers look at examples, only three of them believe that they play a crucial role in design education. Referring to some of his students who say: “I don’t want examples, I don’t want to be influenced,”34 Van dermeersch contends:

“On the one hand, that’s a good starting point, but on the other hand, I find it a wrong starting point, because you must know where you stand, and also because all these designs do have an idea, and a philosophy, and it is interesting to study them, and to know them, and you must be able to make references, even if it was only to position yourself, or for my part to evolve faster to something” [Van dermeersch, 1999].

Precisely because of these references, Van dermeersch finds teaching jewellery designers much more difficult than architecture students. As a design teacher, he

---

32 See also p.115.
33 During the interview, Van Aerschot repeatedly underscores the crucial importance of criticism. He finds that, in general, students are too uncritical. Rather than at the students themselves, however, he points the finger at contemporary magazines on architecture. What seems to preoccupy all magazines, without exception, are exceptional cases, yet Van Aerschot has serious doubts about the practical consequences of this preoccupation.
34 Apparently, such students already existed at the time Van Aerschot studied architecture (see p.115). A possible antidote against such headstrong attitude, Vermeulen suggests, is to draw students’ attention to the fact that most of the great – or less great – architects they know, did not design in a vacuum either, but are linked by multiple threads to what preceded. It might be very useful that students learn to see evolutions in history or connections between designers, and not interpret these as a devaluation of their heroes.
employs cases as a frame of reference, an alphabet, a dictionary to refer to, and being an architect himself, he is far more fluent in the architectural alphabet than in that of jewellery design.

“To architecture students you can say: ‘Look at Frank Lloyd Wright, or look at Mies van der Rohe, or look at…’ and you can in fact project a complete conceptual framework at once to the other… and hope that he has understood it. It’s relative, but in principle, you can make an example of something that is considered to be known” [ibid.].

Vermeulen, too, finds it sometimes useful, and certainly permissible to refer to examples during design teaching, provided they are introduced in not too instrumental a way. Students should not get the message that the solution to their problem lies in Wright’s project x, but be stimulated to look at the problem from this angle, which may act as an eye-opener or spark off a sort of Aha-Erlebnis. Of course, as Van dermeersch already insinuated, you are never sure what the student will do with project x. Yet, in Vermeulen’s opinion, actually that is just as well: “If the student is creative – what he should be after all – he is likely to associate it with other things he thinks of, maybe in another way than you would do it yourself” [Vermeulen, 2000].

Van dermeersch attaches great interest to teaching students to control and direct this kind of associations, instead of passively wait until they are struck by a bright idea. To support such ‘inspiration on demand,’ he has introduced the use of a memory book. In this book, students continually collect collages, images and preferences, which can serve as base material during concept generation.

Also Winkels is quite convinced that cases can and should play a prominent role in design education. She would find it very useful if students attempted to really fathom another architect, try out his approach, and see whether it is their cup of tea or not.

“In fact, I can only advise them to design once in the style of Le Corbusier, and once in the style of Rem Koolhaas (…) and by doing so they learn to look better as well. For many make a design, for instance, and (…) you recognise it as being in the style of Rem Koolhaas, but when it is developed on a more detailed scale, it is actually still very traditional, because they do not have the thinking process yet. And the more they look, the more they are going to look at the detail, for instance in Koolhaas’ work” [Winkels, 1998].

Particularly interesting in this respect is a 1st year design assignment given at the architecture school of Newcastle upon Tyne [Leitch, 1995]. The project, which was significantly called ‘Little Brother’, would undoubtedly meet with Winkels’ approval. At the start of the assignment, each student was introduced to a local architect and his architecture. After an analytical phase in which students tried to uncover and understand the architect’s design philosophy, they were asked to design a ‘little brother’, a small, single cell, studio, to be built within the garden of a house from his repertoire. Students had access to the building itself, to copies of the original working drawings and to the architect, who could be inquired about his intentions when he designed the house. The surprisingly high quality of the student designs, which inspired the organisation of an exhibition, clearly indicates the value of concrete cases in architectural design education.
ARE ARCHITECTS NATURAL CASE-BASED DESIGNERS?

Why, then, are the other interviewees so reluctant to use cases explicitly when teaching design? Apparently, their reluctance is inspired by a form of fear, fear that such use would stimulate students to blindly copy existing designs. Asked whether he would find it useful to provide student designers with concrete examples, Delrue replies: “I myself will not do that, stimulate to copy something that already (exists).” In general, this standpoint seems to be the rule rather than the exception among design teachers. As Venturi and Brown point out, in recent years design students have been taught not to copy. It is nevertheless their firm opinion, that students should be prepared to learn more by copying from the masters:

“You have to have something basic that you can either build on or evolve from or revolt against. You have to have something there in the first place and the only way to get it is to copy, in a good sense of the word” [Venturi in Lawson, 1994].

4.6 Summary

In an attempt to examine the role cases play in the practice and education of architectural design, we have interviewed a small number of architects, c.q. design teachers. This section reviews some features that have emerged during these interviews as characteristic of how cases are used – and how this use is regarded – within architecture’s walls.

Although the interviewees represent different generations, approaches and views of architectural design, they are fairly consistent in their replies regarding practice. Particularly striking is the undeniable and nearly unanimous denial that cases would be used for concept generation. Although the issue was brought up in a neutral tone, five of the six interviewees automatically seemed to make an association with malpractice. It simply did not occur to them that cases might well be viewed as something to learn from, and that there is nothing wrong with using them for conceptual design. On the contrary, their replies reflect a general resentment of references to previous projects, especially projects designed by other architects.

In the course of the discussion, the interviewees were willing to reconsider their first denial, yet most confessions of case use were glossed over with excuses like “Everybody does it” or “Well, not often really” – as if learning from previous projects were something to be ashamed of. Amidst these excuses, however, several ingredients of the CBD recipe turned up (e.g. case selection, structural and derivational adaptation, case acquisition and re-indexing), be it in a more subtle and realistic form.

35 Paradoxically, Denise Scott Brown points out that students copy ideologies [Lawson, 1994].
36 In part this resentment might be due the fact that referring or borrowing is confused with copying. If a design wants to pass for original, copying is obviously out of the question. Ironically, the very denial to rely on previous projects, c.q. the claim to be original is far from original itself. Think for instance of the tabula rasa idea of the Modern Movement (see also chapter 1, p.15).
than CBD researchers usually assume. In stages of the design process other than concept generation, consulting cases seemed less of a taboo.

As to the role of cases in design education, opinions were more divided. All interviewees found it evident that design students fall back on existing projects, yet when it came to stimulating or supporting this phenomenon, they suddenly split into two camps. For design teachers of the first camp, case (re-)use by students may perhaps be pardonable, it is absolutely not to be applauded, let alone encouraged. The opposite camp, on the contrary, is convinced that design students would benefit from such encouragement. What both camps have in common, however, is that their conviction seems to be built on presuppositions and intuitions as much as on clear evidence about the impact of cases on student designers. This suggests that, although CBD claims to derive from how designers learn (to) design, in architecture the role of cases in this learning is far from clear yet. In order to gain a better insight in this role, we have conducted an experimental study that explores the positive and/or negative effects of cases on students’ designs, which is the subject of the following chapter.
DO CASES MAKE A DIFFERENCE?

*Cases open the door, but you must enter by yourself.*

after a Chinese Proverb

---

1 ‘Teachers’ has been replaced by ‘Cases’.
This chapter reports on an experimental study about the effects of using cases, i.e. concrete examples from the past, in architectural design education. The study was actuated by the radically opposing views on, or rather assumptions about these effects, as emerged in the previous chapters. While CBD researchers generally agree that cases (can) play a crucial part in learning design, there is no such agreement among design teachers. For three of the six teachers we interviewed, exposing students to examples is to be avoided, as it would hinder creativity and increase the danger of design fixation. The other three interviewees view cases as a vital basis for students’ design and learning process. This raises the question whether the role of cases in learning design is sufficiently understood to develop effective CBD systems for architecture.

The primary goal of our study was therefore to explore the effects of using cases in architectural design education. Given the fear for design fixation, a more specific goal was to investigate whether cases either limit or improve creative design solutions. In the study, 2nd year architecture students were asked to design an entrance hall for an apartment building. Half of them had access to entrance hall projects of previous years, the other half did not. Analysis of the students’ designs revealed some interesting effects. In particular, the probability of getting a higher score for the concept, the choice of materials and colours, and the creative character of their project, was positively influenced by exposure to examples. Yet, this positive influence especially occurred for the more skilled and motivated students.

5.1 Introduction

Among CBD researchers, it is generally assumed that design cases constitute an essential source of inspiration and information during the design process. As encapsulations of previous design experience, they are thought to contain valuable knowledge that may serve new design situations. Especially during conceptual design, cases would provide grist for a number of decisions to be made. They also seem very useful for designers who are novice in the field, and thus have no or little design experience of their own to rely on. For this reason, CBD systems are believed to be particularly promising for design education, as they can provide students with a substitute for the design experience they are lacking so far. This belief has inspired the development of systems like PRECEDENTS, CBD systems that are explicitly education-oriented. Their main objective is to support the design and learning process of student-architects by means of concrete design cases.

Whereas CBD researchers amply agree on the role of cases in design education, among design teachers opinions are much more divided. While it is generally

---

2 This chapter is a short version of a more elaborate article ‘Exposure to Examples’, to be published in Artificial Intelligence in Design 2000 edited by John Gero [Heylighen & Verstijnen, 2000].
3 See chapter 2, p.55.
4 See chapter 2, p.57.
5 See chapter 3, pp.83 ff.
6 See chapter 4, pp.119 ff.
assumed that architects learn design by experience, quite a few teachers are rather reluctant to use cases explicitly in design education.\textsuperscript{7} Apparently, their reluctance is inspired by fear, fear that such use would stimulate students to blindly copy (parts of) existing projects, and prevent them from coming up with creative design solutions. In design research, this phenomenon is known as ‘design fixation’ [Christiaans & van Andel, 1993; Purcell & Gero, 1996]. Other design teachers are convinced that students may and even should be exposed to design cases, yet according to Venturi and Brown, these teachers form a minority within schools of architecture today [Lawson, 1994].\textsuperscript{8}

Confronting the strong and unanimous belief of CBD researchers with the divided opinions among design teachers, it seems that the influence of cases on student-architects is not entirely clear. This raises the question whether the role of cases in learning architectural design is sufficiently understood to develop effective CBD tools for architecture.\textsuperscript{9} In order to gain a better insight into this role, we have conducted an experimental study which explores the positive and/or negative effects of cases on students’ design projects.

After the theoretical reflections in chapter 2, and the in-depth interviews in chapter 4, this chapter represents yet a different approach to investigate the role of cases in conceiving architecture. It describes an attempt to conduct an experiment on a larger sample of student architects involved in a specific design task. We deliberately use the term \textit{attempt}, for as other design researchers will confirm, such experiments are notoriously difficult to control without resorting to highly artificial laboratory situations [Lawson, 1994].

To provide some background to our experimental study, section 5.2 reviews the results of a series of related experiments that recently have been performed. The procedure of our study is outlined in section 5.3, followed in section 5.4 by the hypotheses that were tested and the results of the analyses. Finally, the chapter closes with a discussion of the outcome (section 5.5), a summary and conclusions (section 5.6).

\textbf{5.2 Related work}

While it is generally accepted that architects learn (to) design by experience, there is no established theory or model that describes or explains this process. As long as we do not know how – and even whether – architects are influenced by cases during design, we cannot expect computer scientists to program successful CBD tools for architecture. In order to develop such systems, we first need a better insight into the

\textsuperscript{7} Symptomatic of this reluctance is the assertion by one of the interviewees that “I myself will not do that, stimulate to copy something that already (exists)” (see chapter 4, p.122).

\textsuperscript{8} See also chapter 4, p.123.

\textsuperscript{9} This question is not just important for the development of CBD tools that are explicitly education-oriented, but for CBD tools in general. In fact, it touches on the very essence of the entire CBD enterprise, which starts from the premise that architects \textit{learn} design by experience. Moreover, within the context of life-long learning, it might be argued that even CBD tools for practice may or should be education-oriented to some extent.
effects of cases on learning and knowledge development in general and on architectural education in particular.

In the recent past, several experiments have been performed that address questions related to this issue. Terry Purcell and John Gero have investigated whether and under what conditions exposure to an example during design leads to design fixation [Purcell & Gero, 1996]. The fixation effect appears to depend on the sort of example used. Mechanical engineers, for instance, become fixated when shown a typical example, but not when exposed to an innovative one. Industrial designers, however, appear to show no evidence of fixation under either of both conditions. This suggests that the role of concrete examples may be field-dependent, and thus caution is called for when extrapolating these results to the field of architecture. Moreover, Purcell and Gero have focused on the negative effects of using examples during design, without examining whether this use might have any potential benefits.

Within the field of architecture, Hernan Casakin has studied the effects of exposure to visual displays during design [Casakin, 1997]. Students and professional architects were asked to solve a series of well-defined and ill-defined design problems, with and without exposure to visual displays, and with and without explicit requirements to use analogy. There is neither space nor need to describe the outcome of this experiment in detail here. Instead, we will focus on the results that are most relevant with regard to our study. Casakin's findings suggest that students benefit from exposure to visual displays during design, even when not explicitly required to use them. The displays turned out to be particularly useful when solving ill-defined problems – which is generally the case during conceptual design – rather than when trying to find unique solutions to well-defined problems. The displays Casakin used, however, contained a mixture of images from the field of architecture (within-domain sources), as well as from other domains (between-domain sources) like art, engineering and science. Strictly speaking, only the use of within-domain sources can be considered CBD, and since both kinds of images were randomly organised during the experiment, the question arises whether the same results would have been found if only within-domain displays had been used.

A study that did focus on the influence of cases, be it under the label of 'precedents', is that by Gert Pasman and Jim Hennessey [Pasman & Hennessey, 1999]. They compared the effects of exposing design students to a set of design cases organised in a random versus a typological way. The results suggest that a typological organisation of cases considerably facilitates the transfer of design knowledge to the new situation. Yet, since all subjects were from the field of industrial design, again caution is in line when drawing conclusions for the field of architecture.

Although the experiments cited above provide CBD researchers – and design teachers – with interesting insights, some fundamental questions remain unanswered. Would student-architects working on a design task produce better design when provided with cases, i.e. concrete examples of existing designs from the field of architecture? Or would these examples prevent students from being creative, resulting in less innovative designs? In order to answer these questions, we have conducted an
DO CASES MAKE A DIFFERENCE?

experiment that studied the influence of exposing students to examples within the field of architecture.

5.3 Experiment

5.3.1. ASSIGNMENT

The experiment used one of the assignments of a course called ‘Basic design’. Although it is not the main design course, Basic Design deals with various topics that are serviceable to architectural design. Topics include drawing skills, graphical expression, theory of colour, photography, etc. The students who attended this course were the subjects in our experiment. They were asked to design an entrance hall for an apartment building, focusing on the choice of materials and colours. In this building, the hall occupies one bay of 3 by 5 by 3 meters (see figure 5.1). The façade does not necessarily adjoin public space, but can be situated at the side of the building block. The two columns at the far end of the entrance are structurally necessary and may not be removed. The task was to choose materials and colours for floor, sidewalls, ceiling, joinery, façade, columns, letterboxes and door-bells, and to decide on the kind of lighting (direct, indirect, general, punctual or accentuated).

Figure 5.1.
Floor plan and section of the entrance hall.
5.3.2 SUBJECTS

All 58 subjects in the experiment (28 female and 30 male) were architecture students of the 2nd year Architecture, Urban Design and Planning at the Faculty of Engineering at the University of Leuven. Having had one year of design education, they participated in the experiment as it is an obligatory project in the curriculum. Students were randomly assigned to an experimental group of 31 subjects, i.e. a group exposed to design examples, and a control group of 27 subjects. Both groups turned out to be more or less equally balanced with respect to gender and grades obtained for Design and Basic Design in the 1st year. These grades henceforth will be referred to as design1 and basic1.

The studio was led by Professor Herman Neuckermans, the titular of the Basic Design course, assisted by four studio teachers. Between these five teachers, there are considerable differences in terms of professional expertise with architectural design in general and familiarity with the project in particular. Two of them had twenty or more years of experience in the field, the others thirteen years or less. This difference in experience will be taken into account as the variable teacher-experience.

5.3.3 PROCEDURE

The assignment was explained during an introductory seminar. To be precise, two separate seminars were organised: one for the control group and one for the experimental group. Save the assignment itself, the latter group was also given a verbal explanation of highly qualitative entrance hall projects from previous years. The examples were selected from the archives of the department by the titular of the course.

The project took four hours a week for eight weeks. During this period, samples of materials and colour cards were open to consultation in the studio. In addition, students of the experimental group had access to the selected design examples from previous years.

From the start to the end of the project all students worked individually. Five weeks after the introductory seminar, they discussed a first proposal with one of the five studio teachers involved in the project. In the following weeks, students were given the opportunity, but were not obliged to engage into additional discussions. The total number of discussions is reflected by the variable #discussions. Upon completion of the project, students handed in a dossier containing colour-pencil drawings of floor plan, ceiling, walls, a presentation of the lighting and at least one perspective view, all drawn on A4 or A3 format. Figure 5.2 shows an extract from such a dossier.

5.3.4 JUDGEMENT

In order to investigate the effects of exposure to examples, the final projects of all students were judged both by the tutors involved with the assignment and by two

---

10 Henceforth, variables will be typeset in italics.
external design teachers. The latter were completely unaware of the experimental conditions we introduced, and were simply asked to assess various aspects of the resulting projects. Apart from a global score reflecting an overall impression of the project, the judges had to rate each project on seven five-point scales each measuring a specific feature of the project. These features were selected because of their particular relevance for the experiment, the assignment or the design field, i.e. architecture. They included the quality of the concept, the choice of materials and colours (the focus of the assignment), the formal articulation, functional organisation and constructive solution of the design (the ‘corners’ of architecture’s triangle), the creative character of the design, and the quality of the presentation.

Figure 5.2. Floor plan and section of the project by Kenny Cupers

A remark should be made concerning the evaluation of these effects. In most conventional approaches to design evaluation, only the final product is assessed and not the cognitive learning increment of the student [Oxman, 1999]. We share Oxman’s reservations concerning the measurement of students’ design learning by judging their project only, yet few convincing alternatives have been developed so far. Therefore, the effects of exposure to examples were evaluated by assessing students’ design products only. See chapter 1, p.9.
5.3.5 DEPENDENT AND INDEPENDENT VARIABLES

Whereas the external judges filled in all scores, the tutors involved left the construction-scale blank as they found the feature irrelevant for this assignment. Because one of the judges turned out to give systematically lower scores than his colleagues, the scores of all judges were standardised in order to be comparable. Subsequently, we calculated for each variable the mean score for each subject. Thus, the total set of fifteen dependent variables consisted for the internal judges of: global-score, concept-score, material&colour-score, form-score, function-score, creativity-score, and presentation-score, and for the external judges of: global-score, concept-score, material&colour-score, form-score, function-score, construction-score, creativity-score, and presentation-score.

In addition to the major variable of interest examples, indicating whether or not students had access to example designs, the independent variables gender, basic1, design1, teacher-experience and #discussions were selected for analysis.

5.4 Analysis

5.4.1 HYPOTHESES

The experiment attempted to test whether exposing student-architects to design cases during the design process affects the quality and creativity of their design products. Given on the one hand the CBD paradigm and on the other hand design teachers’ fear for fixation, the following hypotheses can be constructed:

Hypothesis 1: Based on CBR's cognitive model and the assumption of CBD researchers, we expected the experimental group to come up with higher-quality design projects than the control group. Because cases are said to be particularly useful in the conceptual stage of the design process, the experimental group was expected to perform better especially with respect to the concept of the projects.

Hypothesis 2: Based on the fixation ideas of many design teachers, on the other hand, we expected the experimental group to become fixated by the examples, and thus to be outperformed by the control group with regard to the creativity of their design projects.

In terms of the variables of the experiment, these hypotheses can be rephrased as follows:

Hypothesis 1: The independent variable examples was expected to contribute positively to the dependent variables reflecting the quality of the design project, especially to concept-score, and concept-score. The group that was allowed to inspect examples would, in other words, show a higher mean concept-score, and concept-score than the control group.
Hypothesis 2: We expected the independent variable examples to contribute negatively to the dependent variables reflecting the creativity of the design solution, i.e. creativity-score, and creativity-score_e. The experimental group would, in other words, show a lower mean creativity-score_i and creativity-score_e than the control group.

5.4.2 STATISTICAL ANALYSES

In order to get an overview on our large data set, we analysed the data by a so-called stepwise-forward regression. Regression analysis in general is a way to establish the influence of the independent variables on the probability of the dependent variable. To be precise, the stepwise-forward variant considers each independent variable for its contribution to the dependent variable. It starts out with the independent variable that contributes most, subsequently includes the second most important contributor, and so on, until a sufficient list of contributors is created. In the end, the dependent variable can be rewritten as a function of the selected independent variables in the regression equation Y=a+b_1X_1+b_2X_2+... . In this equation Y is the dependent variable, the X’s are the independent variables, a is the intercept and the b’s are the coefficients of the independent variables (i.e. the X’s). The results of our regression analyses can be found in table 5.1. The first column of the table delineates each dependent variable, the second through the seventh column shows the coefficients of each independent variable. Finally, the last column shows the statistics for the equation. The intercept is not included in the table because it provides no relevant information about the relationship between the dependent and independent variables. Hence, to compose the equation for the first dependent variable, table 5.1 should be read as follows:

\[
global-score_i = a + .374 \text{ basic1} + .319 \text{ #discussions} + .186 \text{ examples} + -.143 \text{ teacher-experience}
\]

In this case, only four of the six independent variables are selected to constitute the equation, of which the first two (basic1 and #discussions) are considered significant contributors as is indicated by an asterisk. The last two are needed to create a sufficient equation, but are no major contributors. The final column shows the relevant statistics for the above equation; the asterisk indicates that a sufficient equation is reached.

Inspection of table 5.1 reveals a frequent significant contribution of the variable basic1. No less than nine of the fifteen scores turn out to be significantly influenced by this variable: global-score_i, concept-score_i, form-score_i, material&colour-score_i, creativity-score_i, presentation-score_i, function-score_e, form-score_e, and construction-score_e. Interestingly, this influence does not coincide with the contribution of design1, the score for design in the 1st year.

Further inspection of table 5.1 shows a relatively frequent significant contribution of the variable #discussions. Remember that this variable reflects the number of meetings a student had with the design teachers. Whereas the first meeting was obligatory for all students, returning for additional meetings occurred on students’
own initiative. In this respect, the total number of discussions may reflect the motivation of the student. Indeed, it seems plausible to assume that more motivated students are more eager to discuss their project with design teachers.\textsuperscript{13}

<table>
<thead>
<tr>
<th>dependent variable</th>
<th>basic1</th>
<th>design</th>
<th>#discussions</th>
<th>examples</th>
<th>teacher-experience</th>
<th>gender</th>
</tr>
</thead>
<tbody>
<tr>
<td>global-score\textsubscript{i}</td>
<td>.374*</td>
<td>.319*</td>
<td>.186</td>
<td>-.143</td>
<td>Adj. $R^2=.246$, F(4,53)=5.660*</td>
<td></td>
</tr>
<tr>
<td>global-score\textsubscript{e}</td>
<td>.190</td>
<td>.251</td>
<td>.139</td>
<td></td>
<td>Adj. $R^2=.123$, F(3,54)=3.660*</td>
<td></td>
</tr>
<tr>
<td>concept-score\textsubscript{i}</td>
<td>.294*</td>
<td>.428*</td>
<td>.140</td>
<td>-.258*</td>
<td>-.123</td>
<td>Adj. $R^2=.267$, F(5,52)=5.160*</td>
</tr>
<tr>
<td>concept-score\textsubscript{e}</td>
<td>.190</td>
<td>.231</td>
<td>.168</td>
<td></td>
<td>Adj. $R^2=.072$, F(3,54)=2.477</td>
<td></td>
</tr>
<tr>
<td>form-score\textsubscript{i}</td>
<td>.357*</td>
<td>.385*</td>
<td>.192</td>
<td>-.219*</td>
<td></td>
<td>Adj. $R^2=.301$, F(4,54)=7.237*</td>
</tr>
<tr>
<td>form-score\textsubscript{e}</td>
<td>.339*</td>
<td>.214</td>
<td>.170</td>
<td></td>
<td>Adj. $R^2=.163$, F(3,55)=4.756*</td>
<td></td>
</tr>
<tr>
<td>function-score\textsubscript{i}</td>
<td>.333*</td>
<td>.219</td>
<td>.140</td>
<td>-.220</td>
<td>Adj. $R^2=.154$, F(4,53)=3.600*</td>
<td></td>
</tr>
<tr>
<td>function-score\textsubscript{e}</td>
<td>.290*</td>
<td>.159</td>
<td></td>
<td></td>
<td>Adj. $R^2=.084$, F(2,55)=3.624*</td>
<td></td>
</tr>
<tr>
<td>material&amp;color-score\textsubscript{i}</td>
<td>.228</td>
<td>.357*</td>
<td>.182</td>
<td></td>
<td>Adj. $R^2=.194$, F(3,54)=5.588*</td>
<td></td>
</tr>
<tr>
<td>material&amp;color-score\textsubscript{e}</td>
<td>.346*</td>
<td>.253*</td>
<td>-.140</td>
<td>-.174</td>
<td>Adj. $R^2=.172$, F(4,53)=3.962*</td>
<td></td>
</tr>
<tr>
<td>creativity-score\textsubscript{i}</td>
<td>.253*</td>
<td>.347*</td>
<td>.146</td>
<td>-.218</td>
<td>Adj. $R^2=.184$, F(4,53)=4.214*</td>
<td></td>
</tr>
<tr>
<td>creativity-score\textsubscript{e}</td>
<td>.262*</td>
<td>.210</td>
<td></td>
<td></td>
<td>Adj. $R^2=.076$, F(2,55)=3.338*</td>
<td></td>
</tr>
<tr>
<td>presentation-score\textsubscript{i}</td>
<td>.387*</td>
<td>.275*</td>
<td>-.234</td>
<td>-.229</td>
<td>Adj. $R^2=.235$, F(4,53)=5.387*</td>
<td></td>
</tr>
<tr>
<td>presentation-score\textsubscript{e}</td>
<td>.251</td>
<td>.181</td>
<td></td>
<td></td>
<td>Adj. $R^2=.059$, F(2,55)=2.797</td>
<td></td>
</tr>
<tr>
<td>construction-score\textsubscript{e}</td>
<td>.259*</td>
<td></td>
<td></td>
<td></td>
<td>Adj. $R^2=.083$, F(2,55)=3.577*</td>
<td></td>
</tr>
</tbody>
</table>

TABLE 5.1. Results from the regression analyses of the data.

Although the variable examples occurs with a high frequency in the analyses, its contribution reaches significance only in case of the material&colour-score\textsubscript{e}. We hypothesised that the variables basic\textsubscript{1} and #discussions might play such an overwhelming role, that the effects of the variable examples may be obscured. Hence, the question arose whether the effect of exposing students to examples varies with students’ skill as measured by basic\textsubscript{1} and/or with their motivation as reflected by #discussions.

\textsuperscript{13} Experienced design teachers might object that students may have had other reasons to return for additional discussions: perhaps they were lazy and wanted the teacher to make a design for them, or perhaps they hoped to get in the teacher’s good (score) books. Although design teachers may not like such reasons, they can still be considered a form of motivation.
In order to answer this question, we split the subjects into two groups: students with low scores on basic1 (i.e. lower than the mean score of 13.08; the scores ranged from 10 to 16) and students with high basic1 scores. In addition, subjects were divided by the #discussions variable: students who had just one (obligatory) discussion and students who had two or more meetings.

In an ANCOVA analysis, all variables are allowed to act on the same level. Therefore, we applied ANCOVA analyses with the three variables of interest (#discussions, basic1 and examples) as independent variables, while allowing the remaining independent variables (teacher-experience, design1 and gender) to assert their influence as co-variants. In these ANCOVA analyses, examples now turns up as a major determinant for the dependent variables concept-score_e, material&colour-score_e, and creativity-score_e, (see respectively figure 5.3, figure 5.4 and figure 5.5).

The leftmost graph of all three figures shows the mean score given by the external judges for students scoring low on basic design in the 1st year, the rightmost graph displays the mean score for those scoring high on basic1. In general, a positive effect of examples could be established for all three dependent variables – concept-score_e, material&colour-score_e and creativity-score_e. Throughout figures 5.3, 5.4 and 5.5, the groups with access to examples are generally found to score higher than the no examples groups. The ANCOVA analysis indeed confirms this observation. The statistical values of this main effect of examples are displayed above each figure.

Further inspection of figures 5.3 and 5.4 suggests that this effect may be largely attributable to the high mean scores in the ‘high on basic1 + 2 or more discussions’ group, as this group scores exceptionally high compared to the other three groups when exposed to examples (the black square in the upper rightmost corner of the right-hand graphs). These other groups show only a small improvement as an effect of examples, which (due to a large variance) may not be significant when put to a separate statistical test. Consequently, although the small increase in mean scores for all three groups may contribute to the overall positive effect of examples, when separated out this small increase turns out not to be statistically significant for each group as such. In other words, apart from a general positive effect of examples, on the basis of statistical analyses only the group scoring high on basic1 in combination with 2 or more discussions can be said to profit considerably from exposure to examples.

---

14 Remember that the judges’ scores were standardised in order to make them comparable. Standardisation reforms the scores so that the mean scores are zero. After standardisation scoring worse than the mean is indicated by a negative sign, scoring better than the mean by a positive one.

15 A post-hoc statistical analysis (Tukey’s HSD) indeed confirms this hypothesis for the variables concept-score_e, and material&colour-score_e.
Figure 5.3. Influence of basic1 (student skill), #discussions (student motivation), and examples on the appreciation of external judges for students’ design concept (concept-score_e).

Figure 5.4. Influence of basic1 (student skill), #discussions (student motivation), and examples on the appreciation of external judges for students’ choice of materials and colours (material&colour-score_e).
This pattern, however, only shows up in figures 5.3 and 5.4, which display the means for the concept-score and material&colour-score. With respect to the creativity-score, the picture is somewhat different, in the sense that the ‘high on basic 1 + 2 or more discussions + examples’ group cannot be said to tower above all other. As can be seen in both graphs of figure 5.5, exposure to examples offers a considerable surplus for students with 2 or more discussions (the black square in the upper rightmost corner of both graphs), irrespective of their scores on basic 1.

In summary, on the basis of our initial regression analyses, we reckoned examples to be overshadowed by basic 1 and #discussions. As can be seen in these figures, it was indeed worthwhile to separate out the latter two from the main variable of interest. Nevertheless, we should keep in mind that the influence of examples only showed up after the influence of the variables basic 1 and #discussions was taken into account.

5.5 Discussion

5.5.1 EFFECTS ON DESIGN QUALITY/CONCEPT

Does exposure to examples positively affect the quality of students’ design projects, and more specifically the concept behind these projects, as predicted by Hypothesis 1?

The results of our analysis suggest that the answer to this question first of all depends upon how well the student performed on the Basic Design course in the 1st year. Only for better performers, both the concept-score and the material&colour-
One possible explanation for the role of design skill is given by design teachers. According to them, the most skilled design students are often students who already have a considerable background before entering school – because their parents hold an extensive library, are subscribed to interesting magazines, drag their children to exhibitions, or are fond of travelling [Poponcini, 1998; Winkels, 1998]. Such a background may help students perform better on Basic Design in the 1st year, and thus might contribute to a higher basic1 score. Following this line of thought, it can be argued that when these students are exposed to examples in the 2nd year, this background may not only be further amplified by the projects provided. It also may help students to make sense of these examples, and thus to call them in for their own project in a productive way.

Secondly, the answer also appears to depend upon the degree of student motivation, as reflected by the number of discussions subjects sought with their design teachers. Only for the more motivated students both the concept-score, and the material&colour-score, are positively influenced by exposure to examples. Eager as they are to design a good project, these students may call in every design support they can find, be it discussions with teachers, examples or whatever other source of design knowledge. Whereas less motivated students, i.e. those who did not ask for extra discussions, may have gazed rather casually at the examples provided, their more motivated colleagues probably have concentrated more intently, since the design knowledge encapsulated by these examples could be useful to improve their own project. In fact, the largest improvement results from the combination of motivation with skill. For those students who are more motivated and more skilled, both the concept-score, and the material&colour-score, show a large positive effect of examples. Clearly students who are not only motivated to look for additional information, but also skilled enough to handle this information can profit from exposure to examples.

In summary, our results show that the effect of exposure to examples on the quality of students' design projects, and in particular on the quality of their design concept, is influenced on the one hand by the subjects' skill (as reflected by the variable basic1) and on the other hand by their motivation (as reflected by the variable #discussions). At this point in our research, it seems therefore plausible to suggest that, in order to profit from previous cases during conceptual design, a combination is needed of at least two elements: on the one hand enough motivation and interest to go (deeply) into others' design projects, and on the other hand sufficient background knowledge and skill to understand and interpret the concept behind these designs.

16 Remember that concept generation in architecture does not necessarily involve the addition of an extra ingredient to the design situation. Every aspect already present in the situation may qualify for this focal role (see chapter 1, p.4). Given the focus of the assignment on the choice of materials and colours, it seems plausible that this choice has served as primary generator of several student projects. This might explain why examples contributes not only to the concept-score, but also to the material&colour-score, as the latter is likely to reflect, to some extent, the quality of students’ design concepts as well.
This effect, however, only occurred in the judgments given by the external teachers, and not in the scores from the teachers involved with the assignment. For none of the aspects of the project, the chance of getting a higher score from the latter teachers is significantly influenced by the variable examples. A possible explanation might be that, although the teachers involved did not know explicitly to which group students belonged, they probably have seen students of the experimental group consulting projects from previous years. Unconsciously, they might have taken this into account when assessing the final projects, thus neutralising the positive effect of exposure to examples. Another explanation may be found in the fact that these scores are influenced by the teachers' familiarity with the examples. Because they remember projects from previous years (especially the good ones), it seems plausible that they implicitly compared students' projects with their predecessors rather than assessing them 'as such'.

On the basis of these results we reckon the assumption of CBD researchers that concrete design cases provide student architects with a significant source of concept knowledge to be correct, but mostly so when the explorers of the cases are skilled and motivated.

5.5.2 EFFECTS ON DESIGN FIXATION/CREATIVITY

Does exposure to examples prevent students from designing creative and innovative projects, as expected in Hypothesis 2? At this point in our research the answer is no. Neither the chance of getting a lower creativity-score, from the external judges, nor the chance of getting a lower creativity-score, from the teachers involved is significantly influenced by the variable examples. Rather the reverse was found in our experiment: the creativity-score, is positively influenced by exposure to examples when taking students' skill and motivation into account. This means that the external judges evaluated the creativity of the design more positively when skilled and motivated students were allowed to explore examples. From the results of our experiment, the prejudice of many design teachers that exposing students to cases would increase the danger of design fixation, thus turns out to be undeserved.

To researchers who have studied creative behaviour, this result will hardly come as a surprise. In the hypothesis that examples would have a limiting effect on creative solutions, creativity seems to be viewed as 'creatio ex nihilo'. Yet, as Arthur Koestler contends, "The creative act is not an action of creation in the sense of the Old Testament. It does not create something out of nothing; it uncovers, selects, reshuffles, combines, synthesizes already existing facts, ideas, faculties, skills" [Koestler, 1989; referred to in Neuckermans, 1994]. Koestler is not the only one who believes creative behaviour to start from what already exists. Others describe creativity as "the easy recombination of ideas in the preconscious" [Barron, 1969; referred to in Neuckermans, 1994], or "putting the elements of one's experiences into new combinations" [MacKinnon, 1970; referred to in Neuckermans, 1994]. In other words, rather than inhibiting creativity, examples may function as a fruitful foundation on which creative solutions can blossom.
5.5.3 EFFECTS OF TEACHER EXPERIENCE

A side, yet remarkable result of our analysis is the influence of the teacher’s experience on the quality of students’ design project. Although it was not anticipated by our hypotheses, it is nevertheless worth discussing this influence here. According to table 5.1, two scores, concept-score and form-score, are significantly influenced by the variable teacher-experience. What makes this result so remarkable, is that the influence turns out to be negative, in other words, the more experienced the design teacher with whom students discussed their design, the lower the chance of scoring well on the concept and formal aspect of their project.

At first sight, these findings seem rather surprising, as more experienced teachers might be expected to have acquired more design knowledge to help students design a better project. However, the view of learning implicitly assumed in this expectation, namely as a one-way knowledge traffic from teacher to student, completely misrepresents how design is learned. Crucial to the development of design knowledge, as we have pointed out, is the interaction between student and teacher [Heylighen e.a., 1999]. Just as designing can be viewed as walking on a thin line, learning design can be compared to learning to rope dance: students act out the role of designer while teachers are around to hold on to in case they should lose balance. Embroidering on this idea of ‘rope dancing’, one possible explanation for the effects of teacher-experience might be a lack of interaction between students and more experienced teachers. When discussing a student’s project, expert teachers rarely get on the rope together with the student.17 Apparently, they like either to do their solo or to stay safely on the ground, i.e. to comment on the student’s work. Novice design teachers, on the other hand, apparently prefer joining the student in his design process, to help him develop his design knowledge, yet perhaps partly because it permits them to hold on to the student should they lose balance themselves.

The difference between novice and expert teachers we are trying to explain here is similar – and thus probably related – to the difference between the language of design and the language about design. The former is in fact the language of doing design, and involves both sketching and talking. The latter is a kind of meta-language that describes and reflects on the design process. It requires the speaker not only to master design, but also to be able to explicitly formulate assumptions and strategies that otherwise remain tacit. Consequently, expert design teachers are likely to speak the language about design more fluently, and thus probably more frequently in the studio than their less experienced colleagues. Students for their part are even less experienced and thus familiar with this language, so that the conversation with an experienced design teacher might change into a speech instead of a dialogue.

17 In [Heylighen e.a., 1999], a significantly negative correlation was found between on the one hand the teacher’s design experience and on the other hand the degree of interaction between student and teacher during the discussion. In other words, the more experienced the teacher, the less interaction took place.
5.6 Summary and conclusions

This chapter has reported on an experiment about the effects of using cases, i.e. concrete examples from the past, in architectural education. The underlying assumption of CBD research is that cases provide architects with a significant source of inspiration and information, especially for conceptual design. Exposing students to cases during the design process would therefore lead to higher-quality design products. Many design teachers, on the other hand, expect such exposure to have a limiting effect on creative design solutions, as it would increase the danger of design fixation. The goal of our experiment was therefore to investigate whether cases either limit or improve the quality and creativity of students’ design projects.

The results of our research to date may be summarised in the following way. Exposure to cases seems to have a positive effect on the quality of students’ projects as appreciated by independent judges. When shown highly qualitative examples from previous years, the chance of getting higher scores for the concept and for the focus of the assignment is significantly positively influenced. Furthermore, these examples do seem to have an improving effect on the creative character of students' designs. All three influences, however, turn out to depend upon students’ skill and motivation.

These results may have important implications for the developers and users of CBD tools in architectural design education. First of all, it is highly improbable that students using such systems risk to be hindered in their creativity, as we found no signs for the design fixation feared by some. On the contrary, the examples in the case base have the potential to help students improve the quality and innovative character of their own design projects.

This potential, however, seems to be higher for more skilled and motivated students. Thus, if CBD researchers are to develop more effective CBD tools, i.e. effective for all student-designers, simply providing a case base filled with examples will not do. Rather, these examples should be organised and presented in such a way so as to make up for the background and/or motivation weaker students seem to lack. The challenge is to engage students into in-depth explorations of others' design projects. Only then one could speak of a CBD tool that really supports student-architects during design.

If these results can be confirmed by future experiments, they may also provide lessons for research on CBR and AI at large. According to design teachers, students who perform well in the 1st year, are often students who have already built up a considerable background. Although this background transcends the field of architecture, it is likely to include at least some cases of architectural design. If basic1 can be taken to reflect this background, the influence of this variable seems to suggest that, before the CBR cycle can get going, a sort of germinal case base is needed. This raises questions about the content of this case base, in other words, about the number, variety and quality of cases needed for CBR to flourish.

It is, however, too early to proclaim such speculations based on one experiment only. Further evidence is needed that skill and motivation are indeed prerequisites for successful Case-Based Design. In the study presented, the subjects represented a
continuum from good over intermediate to weak performance on Basic Design in the 1st year. A better strategy for determining the influence of student skill, therefore, might be to repeat the study with a mixture of excellent and weak students only. Furthermore, it would be interesting to investigate whether expert architects profit even more from examples than students, highly skilled and motivated as they are.

Awaiting the results of this investigation, design teachers can surely start off by preparing themselves for a profound change in mentality. According to our study, their present emphasis on originality and fear for design fixation wrongfully excludes examples from supporting design in the studio. To acknowledge this fact is but a first step, yet it is not an unimportant one.
CASE-BASED DESIGN IN ARCHITECTURE

A critical review
In the previous chapters, we have studied the role of cases in architectural design in various ways and from multiple perspectives. Chapter 2 has taken an excursion into the vast domain of AI, where the study of this phenomenon represents a substantial body of research. It has traced back its origins and pointed out the premise and promises of this research, followed by the description of a small yet varied set of CBD systems in chapter 3. Switching from a theoretical to a more pragmatic perspective, chapter 4 has handed the floor over to six architects who have ample experience in both design practice and education. Their opposite views on the use of cases by design students has actuated the experiment described in chapter 5. Chapter 6 tries to bring some of the key issues raised by these previous chapters into sharper focus, by dealing with them directly and confronting the views of the different stakeholders.

The discussion starts from the observation that the expectations of the CBD pioneers are far from being accomplished today. Despite serious attempts to develop reliable CBD tools, a convincing breakthrough of CBD in architecture has yet to emerge. Of course, one may argue that CBD research is still in its infancy, yet the extensive and successful use of CBR applications in other domains and disciplines can be taken to refute this argument. Rather, there might be more fundamental reasons for the so far limited success of CBD in architecture. Based on the issues brought up in the previous chapters, reasons can be found at three different levels. Level one pertains to the model of cognition underlying CBD, which raises some fundamental questions in the field of architecture. Secondly, at the level of implementation, few tools manage to exploit the full consequences of this cognitive model, often leading to an oversimplification of CBD and/or architectural design. Level three adopts a broader perspective and takes a look at the context in which CBD tools are to be used. Although this context may not relate directly to these tools themselves, its impact on their (in-)effectiveness should not be underestimated.

6.1 The cognitive model behind CBD

The first level concerns the model of cognition in which CBD is historically rooted. Although this model may provide a plausible explanation of how designers acquire and apply design knowledge, within the field of architecture it raises some specific difficulties.

6.1.1 DESIGN EXPERIENCE

First of all, there is some confusion about the leading actor in CBD’s cognitive model: design experience. In general, design experience can be interpreted in multiple ways.

---

1 This chapter is a more elaborate version of the article ‘(Learning from experience)²’, to be published in a special issue of Automation in Construction edited by Andre Brown [Heylighen & Neuckermans, 2000a].
2 See chapter 2, p.52.
3 Remember that the term ‘case’ in Case-Based Design stands for the interpreted representation of a real experience, including all details that make the experience individual (see chapter 2, p.48).
CBD IN ARCHITECTURE

[Heylighen, 1998]. Strictly speaking, having experience in design means having designed yourself. In a wider sense, observing an architect’s design process can be seen as a source of experience too. Finally, architects acquire experience by exposure to designs made by others, either in real life or, more often these days, through pictures in magazines, books, lectures or exhibitions.

CBD researchers tend to focus exclusively on this latter interpretation of design experience, the Galathea team being the proverbial exception to the rule [Arlati e.a., 1996]. Design cases, that is to say interpreted representations of design experience, are usually taken to mean (fragments of) design products, generally, but not necessarily, at the final stage of the design process. For the architects we interviewed, however, to learn from design experience clearly means to learn from the design process at least as much as from the designed product. Two interviewees were quite explicit in their contention that what they reuse is not just what has been designed, but rather the way it has come about.

A first element of critique is therefore the biased portrayal of design experience in CBD research. Although design products definitely contribute to the acquisition of design experience, they cannot reveal the constantly changing conditions that actually structure the process of designing [Brown & Duguid, 1996]. Indeed, "conflicting demands from within the client organisation, the remoteness of the user, difficulties with the bearing capacity of the soil, an unsympathetic planning authority, changing circumstances during the design period, restricted or inflexible methods of financing scheme… and many more difficulties remain inscrutable to all but the most perceptible and insightful of architectural critics" [Lawson, 1990]. Dealing with such changing and conflicting conditions requires a form of knowledge, c.q. knowing that is embedded within the very act of designing and thus escapes the static form of a design product [Schön, 1985]. It is obvious then that, if architects learn design by experience, this learning does not only involve exposure to designs, but also active engagement in designing. As one of the interviewees puts it, “You must have struggled with it yourself”.

It would be misleading, however, to depict CBD researchers as completely unaware of the distinction between product and process. Evidence of such awareness can be found in the idea of ‘deep’ cases: design products augmented with traces of the design and building process, such as options, decisions, justifications, assessments and revisions [Hua, Faltings & Smith, 1996]. In PRECEDENTS, for instance, cases

---

4 Think, for instance, of the traditional master-apprenticeship approach to architectural design education, which, to some extent, lives on today in the practical training of graduated architects.

5 See chapter 4, pp.112-113.

6 As pointed out in chapter 1, the knowledge-richness of design does not only lie in the different types of knowledge needed, but also in the different ways of knowing (see pp.12 ff).

7 See chapter 4, p.113. Note that this view shares some territory with Dewey’s idea that thinking and doing are inseparable: “Only by wrestling with the conditions of the problem at first hand, seeking and finding his own way out, does he think” [Dewey, 1916; referred to in Coyne, 1995 p.40].

8 See chapter 3, p.66.
contain descriptions of conceptual points, highlighting those aspects which were particularly decisive in the early stage of the design [Oxman & Oxman, 1994a; 1994b]. Archie-II supplements cases with stories from different stakeholders – people who carried out, use or maintain the building – telling how the design turned out [Domeshek & Kolodner, 1993]. Nevertheless, many CBD researchers share CADRE’s preference for cases that are as shallow as possible for the obvious reason that deep cases are not there for the taking [Hua e.a., 1996]. A case in point is the FABEL team [Voss, 1997 pp.328-329]. After passionate discussions about what should be represented in a case, they decided on a totally liberal case format. So far, however, FABEL only contains cases with a design solution, and possibly a reference to the building project it was taken from. Its developers seem rather pessimistic about getting richer cases in the future, and therefore advise other CBD teams to identify cases they might find without too much effort.

6.1.2 RETRIEVING RELEVANT CASES

Another element of CBD’s cognitive model that does not perfectly match real-world design, is the retrieval of a relevant case. Faced with a new design problem, the model assumes designers to retrieve the most relevant case in memory for solving the problem at hand.9 Within the field of architecture, however, what makes a previous design experience relevant to a new design situation is far from clear. Underlying most CBD research is the assumption that relevance equals similarity, in other words, that the most relevant case is the one having the most features in common with the new design. In FABEL, these features range from functional components and visual appearance over occurrence of certain layout patterns to structure or topology [Voss, 1997]. Although this interpretation of relevance might apply to other design disciplines, its applicability to architectural design is highly questionable. In the series of interviews we conducted, few architects did confirm the assumption of CBD researchers. Neither did they manage to formulate exactly why they call on a particular case during design. Probably the most honest answer we got was: "Often simply because I feel like designing something like it" [Poponcini, 1998]. This feeling is not necessarily prompted by striking similarities between previous and current design situation.10 nor by excellent qualities of the case in question.11 In fact,

---

9 See chapter 2, pp.48-49 and p.54.
10 On the contrary, the more significant the difference, the greater the challenge, according to Poponcini. Note that this corresponds to findings about the reuse of episodic knowledge in design. Although in such reuse the relation between previous (source) and current (target) design situation is usually one of similarity, designers may exploit other relationships as well, especially opposition [Visser, 1996]. If we consider the use of cases, c.q. episodic knowledge as a special form of analogy, this seems to confirm that the three constraints that guide analogical reasoning – similarity, structure and purpose – do not operate like rigid rules, but rather like various, sometimes opposing forces pressing toward some satisfying compromise [Holyoak & Thagard, 1997] (see also chapter 1, p.18). As far as Poponcini concerns, similarity does not seem to have an important part in this compromise.
11 Van Aerschot for instance likes to embroider on previous projects he is not completely satisfied with.
judging from our interviewees, there is no system in this feeling at all, except perhaps for the moment when it shows up. In discussion, several architects suggested that they already had selected one or more ‘relevant’ cases, yet were still waiting for a suitable design project to call them in [Delrue, 1998; Poponcini, 1998; Winkels, 1998]. In other words, the decision to draw on a particular case seems to be taken when confronted with the case itself rather than when facing a new design problem.

This is probably much more important than it may sound, since one of the chief challenges in CBD is to make systems retrieve ‘the right case at the right time.’ Much of the research on CBD, as well as on CBR in general, deals with finding reliable (and preferably rapid) algorithms for case retrieval. Given the description of a new design situation, these algorithms start roaming memory in search of the most similar case available. To this extent, each case is labelled with one or a small set of indices that can be compared with features of the probe. This means that the relevance of a case is commonly defined in terms of the design task at hand, and thus may be thought of as an arrow pointing from current to previous project. In real-world design, however, the selection of a relevant project appears to be made in retrospect of that project itself rather than at the outset of a new design. Cases are already caught up in a selection process before a new design starts: confronted with a project they would like to rely, fall back or embroider on, architects wait for an opportunity, i.e. a design task that provides room for doing so. In our opinion, this looks more like an arrow pointing from previous to current project rather than the other way around.

6.1.3 VIEW ON ARCHITECTURAL DESIGN

A more fundamental point of critique concerns the view of design that is implicated in the whole CBD enterprise. The very idea to apply CBR to design starts from the premise that design can be considered a form of problem solving, at best a particularly fascinating one. In case of architectural design, the least one can say is that this premise is a serious over-simplification of what goes on in reality. Of course, the view of design as a form of problem solving was not born within the CBD community, yet considering its popularity amongst CBD researchers, it is worth discussing this view in some detail here.

The idea to consider design as a problem solving activity dates from the early 1960s, a period dominated by an utilitarian, functionalist view of economic, social and cultural phenomena. One of the books giving evidence of this idea is Community and privacy by Serge Chermayeff and Christopher Alexander [Chermayeff & Alexander, 1963]. According to the authors, any design – regardless to the particularities of the

---

12 In every day life, we use a similar technique to look up something in a book. Yet, whereas we call ‘an index’ the entire list of keywords in a book, CBR researchers use the term for each single keyword. In order to eliminate potential confusion, a distinction is made between probe and label indices: the former refers to the description an algorithm uses to find something in memory – the reader’s guess at a good keyword; the latter to the description that has been assigned to the contents of memory – the alphabetised list of keywords in a book [Domeshek, 1994].

13 See chapter 1, p.14.
domain – is a problem that must be solved. So far, tradition and intuition had been successful in design because problems had been simple, but now – that is anno 1963 – design problems had become so complex, that they required new, rational methods to be solved. Despite this increased complexity, however, design was regarded as solving *tame* problems, i.e. a well-defined problems with a controllable solution that can be found by following a well-described sequence of steps: problem definition, generation of alternatives, simulation and prediction, evaluation and feedback [Smothers & Drumond, 1975; referred to in Neuckermans, 1994 p.omod-3].

The impact of this new view of design as a general-purpose solving activity can hardly be underestimated [Tzonis & White, 1994]. First of all, it gave birth to a whole generation of so-called Space Allocation methods – techniques for automating the optimal spatial arrangement of activities in buildings. Moreover, it seriously influenced the infant discipline of Design Methodology,\(^\text{14}\) which attempted to overcome the unscientific character of design by developing rational design methods. In his book significantly entitled *Design methods - seeds of human futures*, John Christopher Jones publishes no less than 35 methods, which about embody the problem solving approach to design [Jones, 1970]. Characteristic of these early methods are the systematic exploration of possibilities, the emphasis on optimisation and the frequent reduction of new problems to an already known problem. To reduce a new problem to one that is already known is the very essence of CBR.

The idea to apply CBR to design dates from the early 1990s. Exactly three decades after the publication of *Community and privacy*, Kolodner publishes her book *Case-Based Reasoning*, which, in some sense, would become the bible of CBR research [Kolodner, 1993]. According to the author, problems in design are defined as a set of constraints and the task of a problem solver is to provide a concrete artefact that solves the constraint problem [ibid. p.80]. Such artefact can be found by following the steps of the CBR-cycle as prescribed by the model of cognition: problem definition, case retrieval, modification (either adaptation or merging), assessment and storage.

After a short period of boundless optimism, however, the view of design as pure problem solving was called into increasing question, in architecture as in other design disciplines [Cross, 1977; Neuckermans, 1975]. Instead, for a variety of reasons, architectural design is being viewed, studied and developed as a highly complex activity with its own unique characteristics. One problem with the problem solving view is that it denies all meanings of architectural form that have no functional justification. As Leandro Madrazo argues, “Even though the design of certain objects can be seen as a search for the form that better satisfies the function, or functions, it would still be difficult to contend that the forms of a Greek temple and a Gothic cathedral, or of the villa Rotonda and the villa Savoye have been created to give response to specific functional demands, unless the cultural and symbolic values of art form are included among them” [Madrazo, 1995 pp.345-346]. Moreover, to researchers in other fields, what architects do appears so peculiar, that it starts to acquire the status of a model for complex activities in other disciplines. If previously architectural design was considered a form of problem solving, today activities with a

\(^{14}\) The first Design Methods Conference was held in London in 1962.
high level of complexity are viewed as a form of architecting [Rechtin, 1991]. However, this view is only just beginning to be sympathetically received in universities, where architecture traditionally has a low research profile [Heylighen & Bouwen, 1998]. In this respect, it is perhaps understandable, yet nevertheless regrettable that CBD helps keeping the problem solving view of architectural design alive.

It would be misleading, however, to depict CBD’s view of design as if it were still the paradigm of the 1960s. Several systems described in chapter 3 show some significant improvements compared to the original Chermayeff-Alexander approach. Archie-II, for instance, allows to take several design issues simultaneously in consideration, and to jump from one issue to another, as architects tend to do. In SEED, then, problem specifications are not assumed to be complete at the outset, but can be dynamically re(de)fined at any time in the design process. This redefinition is motivated by the fact that many aspects of a design problem are unknown before design starts and are only discovered while designing. Yet, dynamic as the problem specification in SEED might be, the system cannot deny viewing design as a problem to be solved and, what is worse, as a problem that can be neatly decomposed into separate sub-problems. This view of design as problem solving is even more apparent in CADRE and IDIOM, whose adaptation algorithms look suspiciously like the Space Allocation techniques in the 1960s.

Typically, design research is motivated by a desire to change designing for the better, and one might expect that this applies to CBD research as well. Yet, as Daniel Schodek argues, “good design will not result simply because we have a structured framework for the design process in a computer-based model that sounds reasonable, and a database that supports this process” [Schodek, 1994 p.453]. It is too easy to escape into the realm of a cognitive model and forget about what is involved in conceiving architecture, as CBD researchers seem to do. As Lawson contends, “Those who write about the design process as if it were merely problem solving do the field a disservice. A large part of the business of designing involves finding problems, understanding and clarifying objectives and attempting to balance criteria for success” [Lawson, 1994 p.5]. Part of the business of designing undoubtedly involves problem solving too, yet to reduce architectural design to that part is just that – a reduction.

### 6.2 Implementing CBD tools

A second level that poses serious problems is that of implementation. Even if the model of cognition behind CBD could perfectly match real-world architectural design, few tools manage to draw the full consequences of this model, which often leads to an oversimplification of CBD and/or architectural design.

---

15 See the discussion of Archie-II (chapter 3, pp.64-65).
16 See chapter 1, pp.13-14.
17 See also case study 3.6 (chapter 3, pp.89 ff).
18 See also case study 3.2 and 3.4 (chapter 3, pp.66 ff and pp.78 ff).
6.2.1 REPRESENTING DESIGN CASES

The problem already starts with the representation of cases, i.e. records of design experience in a computer memory. A major advantage of using design cases (as opposed to abstract rules or models) is that cases are concrete, full of detail, vivid and open to interpretation. However, the price of having to represent cases may be that this very richness is lost, simply because it refuses to be encoded in a digital format. Indeed, "representing design cases requires an abstraction of the experience into a symbolic form that the reasoner (either computer or human) can manipulate" [Maher & Gomez de Silva Garza, 1997; italics added].

No wonder CBD researchers prefer interpreting design cases as products rather than processes. A major implementation question Galathea faces is finding an appropriate format to describe and compare different states during design [Arlati e.a., 1996]. To represent a design process in a computer memory would require, in some sense, to repeat that process.

Yet, even in case of design products, i.e. building designs, digital representation can be characterised as a movement toward abstraction [Matthews & Temple, 1999]. Architecture has an intrinsic materiality, with physical objects to be touched and handled and that occupy space. As Van Aerschot contends, “designing and building is not so much the graphical expression of an abstract concept, the fossilisation of a drawing, but a brusque confrontation with a tough, unruly and prosaic materiality, and (that) a profound insight in this materiality is the first condition for a value measure of this concept” [Van Aerschot, 1991]. In case of architectural design, CBD researchers are thus confronted with a dilemma: between the intrinsic concreteness of architecture and the abstraction typifying digital representation, there lies a fundamental contradiction.

As long as this digital representation is used by human designers, there may be little cause for alarm. Indeed, what an architect derives from what is shown on the screen is much richer than what the computer only knows as 0s and 1s [Maher, 1998]. Through years of experience, architects learn to develop a mapping between a design's representation and reality, allowing them to form an idea of what the real building looks, feels and even sounds like. Yet, when cases are manipulated by the computer, as happens in CADRE, FABEL, IDIOM and SEED, the problem becomes more serious. To our knowledge, computers are not aware of a reality other than the digital representation of a design. For them, this representation is the design. By

---

19 See chapter 2, pp.57-58.
20 Note that this does not apply to all design disciplines. In the field of software design, for instance, design products are intrinsically immaterial.
21 Some may argue that a solution to this dilemma is (or will be) offered by Virtual Reality technology. Yet, "paradoxically, even virtual-reality systems deny the importance of engaging the senses in the physical world. One of the more extreme aims of virtual reality is to present sense data ‘directly to the brain,’ circumventing the body’s normal engagement in the physical world" [Coyne, 1995 p.29].
CBD IN ARCHITECTURE

consequence, the digital representation of design cases in these systems reduces architecture from a meaningful form to a set of abstract, geometric shapes.22

That is not to say that CBD researchers are completely indifferent to aspects of architecture other than geometry. The IDIOM team, for instance, introduced the idea of preferences precisely to take into consideration those design issues that cannot be modelled more exactly, such as social and political considerations.23 In practice, however, there is little evidence to support the assumption that these good intentions have survived the stage of implementation, for IDIOM’s preferences turn out to express dimensional requirements only.

In summary, though based on a cognitive model asserting the prevalence of concreteness, CBD systems paradoxically tend to trade in abstractions. For reasons of implementation, they rely on transplanting a rich and intrinsically physical phenomenon like architecture into a rarefied context of constants and variables. Design cases are extracted from context and reduced to abstract computer-readable formalisms. Irrespective of one’s belief in the possibility of AI, it is obvious that these formalisms so far fail to represent cases as records of concrete design experiences.24

6.2.2 INTEGRATION

Inseparably linked to the previous topic is the coherence cases can provide, the glue that holds the different aspects of a design together. Just like a case’s concreteness, this integration is seriously challenged by implementation and often completely lost in CBD systems.

Some systems concentrate completely on one single aspect of architectural design. CADRE and IDIOM, as mentioned, have focused on geometry or, more precisely, on the geometrical constraints on floor layouts.25 In this case, integration is not a question, as other issues are simply not taken into consideration. This focus on one single aspect is rather surprising for systems presenting themselves as CBD tools. It

22 The difference between shape and form referred to here, comes from Leandro Madrazo [Madrazo, 1995 p.358]. A shape, e.g. a geometric figure, is amenable to systematisation and, as such, can exist independently of the mind that thinks or conceives the shape. A form, on the other hand, exists only in so far as it is a ‘form in somebody’s mind’ and, as such, cannot be detached from the designer who conceives the form. Note that this distinction differs from that drawn by Charles Moore, who considers form as a kind of constant, and shape as an actualisation of that form [Moore & Allen, 1976]. For instance, all spoons have the same form – a recipient to hold liquid with a handle – but their specific shape may vary.

23 See case study 3.4 (chapter 3, p.79).

24 Architectural design, we have pointed out, shows striking similarities with complex systems (see chapter 1, p.9). To represent such system accurately, is to represent each and every interaction in the system as well as each and every interaction with the environment, which is of course complex too. In short, it is to represent life, the universe and everything [Cilliers, 1998 p.4]. At this stage, it may appear that we are arguing for representing design cases in this way, yet it is obvious that there is no practical way of doing this. A digital representation, like every model, has to be limited. The problem with CBD systems so far is not that their case representation is limited, but rather where the limits are drawn, i.e. what aspects are considered essential in architectural design and thus worth representing.

25 See case studies 3.2 and 3.4 (chapter 3, pp.66 ff and pp.78 ff).
CHAPTER 6

raises the question what advantages of using cases are left, if they are reduced to one feature anyway.

A totally different approach, adopted in FABEL and SEED, is to develop separate modules according to different aspects of architectural design (such as construction, topology or functional subsystem). Precisely because integration of such diverse aspects proved to be a non-trivial task, FABEL’s modules – significantly called ‘specialists’ – are rather independent and each equipped with a separate case base and adaptation method [Web_Fabel, 1999]. As long as this independence exists, perhaps such a tool should be called a collection of Rule- or Model-Based systems rather than a Case-Based one.

6.2.3 DYNAMICS

If there are characteristics of cases that raise serious problems of implementation, the structure of the case base is what we call a real problem. The model of cognition behind CBD was inspired by the Theory of Dynamic Memory. According to this theory, human memory never behaves exactly the same way twice, since it changes as a result of its own experiences. Each time memory structures are used to recall previous experiences, memory re-organises and re-defines itself, be it by acquiring new experiences, re-interpreting old ones or creating new generalisations.

Being rooted in this Dynamic Memory Theory, a major promise of CBD systems is their ability to maintain and enhance their competency without major reprogramming. Unlike Rule- or Model-Based systems, a CBD tool can increase its knowledge and improve its skills each time it is used, simply by storing the design result into the case base. Moreover, by re-indexing cases already stored, the system can fine-tune its recall apparatus and update its knowledge content continually.

In the CBD systems studied, however, this promise is far from being accomplished today. Unlike the expectations of the CBD pioneers, few tools can be updated by ‘simply’ storing new cases or re-indexing cases already stored. If any trace of dynamics can be found, it is perhaps in CADRE, not coincidentally a system with shallow cases. Its pre-modeller Mod-4 was developed precisely to facilitate the acquisition of new cases. Users can add a case by simply inserting walls, doors and windows, and specifying their function, material and construction. From this user input, Mod-4 automatically derives an AutoCAD model plus additional graph descriptions needed for adaptation.

By contrast, systems like Archie-II and PRECEDENTS, which have opted for deep cases, can hardly be called dynamic at all. A major shortcoming of both systems, we have pointed out, is their highly complex memory organisation. Although the sophisticated indexing system in Archie-II and PRECEDENTS definitely contributes

---

26 See case studies 3.3 and 3.6 (chapter 3, pp.72 and pp.89 ff).
27 See chapter 2, p.46.
28 See case study 3.2 (chapter 3, p.67).
29 See case studies 3.1 and 3.5 (chapter 3, pp.64-65 and p.88).
to the richness of the content, it seriously complicates the addition, removal or rearrangement of cases and stories in the case base.

Consequently, there appears to be a strong tension, if not a contradiction between the dynamics of a CBD system on the one hand, and the richness of its case representation and memory organisation on the other hand. If a dynamic memory, c.q. case base is vital to successful CBD, this casts doubt on the need for richer cases advocated above. It is obvious that a CBD system cannot easily change when its case representation and, a fortiori, its indexing system is too complex. This does not imply that there should be no indexing system at all, but rather that indices should be flexible.

6.3 CBD in design education

Unlike the previous levels, where CBD’s (in-)effectiveness has been discussed in relative isolation from the wider context in which it is used, level three tries to adopt a broader perspective. This perspective includes how CBD is integrated in design education and how its use is presented by teachers, viewed by students and supported by schools. All these factors can considerably influence the effectiveness of learning outcomes, even though they may not relate to CBD itself.

CBD systems, as mentioned before, would be particularly promising for design education, as they can provide students with a substitute for the experience they are lacking so far.30 However, there is little evidence that such systems, whether explicitly education-oriented or not, have become widely used tools in schools of architecture. Their use currently seems to be thwarted by several difficulties, often associated with prevailing prejudices.

In general, design teachers are rarely burning with enthusiasm to introduce computer tools in design teaching [Neuckermans & Geebelen, 1999]. In case of CBD systems, this enthusiasm may be even harder to find, as these tools might exceedingly increase the danger of students blindly copying (parts of) previous projects. Because of this supposed risk for design fixation, many design teachers are already reluctant to provide students with concrete cases as such [Lawson, 1994].31 The fact that CBD systems make cases available by medium of a computer, is most likely to do this prejudice more harm than good. To design students, on the other hand, CBD’s medium is probably less of an obstacle. Yet, regarding the explicit use of cases, students seem to share their teachers’ concern that being creative becomes more difficult. Some of them are downright afraid for design fixation, as can be inferred from the contention: “I don’t want examples, I don’t want to be influenced”.32

In chapter 5, we have dismissed the prejudice against cases as an ill-informed view of creativity. Yet, instead of ill-informedness, perhaps the term confusion, or even better, over-emphasis is more in line here. All too often design teachers, and people in

30 See chapter 2, p.57.
31 See also chapter 4, pp.122-123.
32 An interviewee citing some of his students (chapter 4, p.121). See also another interviewee’s recollection of his own student time (chapter 4, p.115).
In general, seem to confuse creativity with originality, a word of similar, but not identical meaning. In general, “The original mind slipping free of the conventional and the commonplace, comes up with things no one else has thought of – the new idea, the different approach. (...) The creative mind goes further, combining the fruits of experience and imagination in an original way to recreate reality in a new form” [Hayakawa & Ehrlich, 1994 p.114; italics added]. In case of architectural design, the difference between creativity and originality is aptly put by Herman Hertzberger. Describing his design of the entrance stairs to a school building, he contends [Lawson, 1994]:

“The problem is that you have certain moments when many children have to (pass through), the problem also is that sometimes you have a small number of people waiting, the problem is that sometimes it rains and then it is not very nice to sit there, the problem is … So you get this whole list of things that altogether represent the problem. And then you say, well given all these things, the stair should not be too small, should not be too large, it should be covered over, it should not be … and so on. There are always these contradictions. This for me is creativity, you know, finding solutions to all these things that are contrary, and the wrong type of creativity is that you just forget about the fact that sometimes it rains, you forget that sometimes there are many people, and you just make beautiful stairs from the one idea you have in your head. This is not real creativity, it is fake creativity!”

In order to obtain the label ‘original’, Hertzberger points out, an architectural design only needs to be different – different from the ordinary, from the commonplace, from what other architects have designed. The label ‘creative’, however, in addition requires that this difference makes a difference, in other words, that it entails a substantial improvement on several if not all fronts of the design. Thus, while creativity implies architectural quality, there is no such implication in case of originality. Our built environment indeed provides little evidence that the most original design is necessarily the best. Nevertheless, we seem to be living in an era in which originality is valued for its own sake – by architectural magazines, by juries of design competitions, and apparently by design teachers as well. All too often, teachers wishing to applaud original and imaginative work, forgive functional inadequacies of students’ designs [Lawson, 1994]. Also their concern with respect to design fixation seems to relate specifically to the issue of originality.

33 Neuckermans, for instance, describes creativity as the ability to create, to bring about something original [Neuckermans, 1994 p.121]. This (over-)emphasis on originality has led Bazon Brock to the statement: “Only hairdressers are creative,” meaning that anything different from the norm or the arbitrary is labelled as creative nowadays [Schmitt, 1993 p.13].

34 To say it in the words of Robert Venturi, “It’s better to be good than to be original” [Lawson, 1994]. Note that the difference between good and original is playing tricks on other fields than architecture as well. In the 5th edition of Leuven Kort, Flanders’ prominent short film festival, it was precisely this difference that made it so hard for the jury to reach an agreement: “We found the entries both good and original. Only, what was good, was not original, and what was original was not good,” according to the jury report [Tuts, 1999].

35 See for instance the complaint by one of the interviewees about the preference of most magazines for exceptional projects (see chapter 4, p.121).
At this point, it may be useful to ask where this emphasis on originality originates from. In the history of architecture, it turns out to be a relatively recent phenomenon, which can be traced back to the second half of the 19th century. At that time, the concept of composition, prevalent in painting since the Renaissance, entered architectural discourse at the Ecole des Beaux Arts [Leupen e.a., 1997 p.47]. The main shape of a building was no longer achieved by distribution and disposition, i.e. by dividing and subdividing a single entity, but by combining components into a larger composition.36 This conceptual shift from distribution to composition did not only change the way the plan was organised, it also introduced the notion of originality in architecture. If until then designs were ordered to reflect the divine and only true model,37 from now on the personality and individuality of the architect entered the design, and would gain ever more importance in the centuries that followed.

The current fear that cases would increase the danger of design fixation, is to be understood within the context of this evolution. On the face of it, this mediation on divine and personal portions in architecture seems far removed from the discussion on cases in design education. Yet, if design teachers are unsympathetic to providing students with cases today, it is largely because the latter portion is considered so important. Of course, we would not go as far as to argue that architecture should reflect the divine model again, but the present emphasis on the individual portion of the architect wrongfully excludes existing projects from nourishing new designs. The observation that students effectively benefit from concrete cases, prompts us to question the validity of this emphasis, especially in the context of design education.

6.4 Taking stock

The application of CBR to design seemed to hold great promises for the domain of architecture in general, and for architectural design education in particular. Yet, unlike in other areas of application, CBD systems have rarely become widely used tools in architectural design. In an attempt to explain this so far limited success, we have put the views of the previous chapters against one another. The outcome of this confrontation can be summarised as follows.

Case-Based Design has a lot to offer. It gives one plausible explanation of how designers in architecture and other design disciplines acquire and use (design) knowledge. Unfortunately, it is limited in significant respects. In particular, it is based on a model of cognition that views architectural design as a form of problem solving. As such, tools that flow from CBD research are largely inconsistent with the reality of architectural design. Furthermore, when it comes to implementation, features at the heart of the cognitive model – concreteness, integration, dynamics – are lost in the

36 In the jargon of the Design Methods Movement, distribution and composition would perhaps be called a top-down, respectively bottom-up approach to design [Neuckermans, 1994 p.oproc-16].

37 The method of distribution derived from Vitruvius, brought order in a design by means of geometric principles and proportional systems. These were regarded as an objective medium between the divine model to be emulated and human creation [Leupen e.a., 1997].
transfer from human to computer memory. Last but not least, a considerable part of the target group appears to be loaded with prejudices, which may seriously hamper a convincing breakthrough of CBD in the field of architecture.

In fact, a major obstacle for this breakthrough turns out to be confusion: confusion on the side of CBD researchers, over the meaning of design experience, the relevance of design cases, and the level of complexity in architectural design; confusion on the side of architects, design teachers and students, over the impact of cases, and over what creativity is, and is not. This confusion cannot be sensibly dismissed as a peripheral problem, but should rather be taken as a serious criticism of how CBD research and, by extension, CAAD research in general is currently conducted. As long as researchers keep gathering at conferences to discuss the newest retrieval and adaptation algorithms, (student-)architects will probably quietly ignore their work. We must take this ignorance very seriously, and perhaps admit that it may indicate a gap between real-world and artificial architectural design. Of course it is tempting to conclude that, in order to bridge this gap, further research is needed. Yet, perhaps something else is needed here. If CBD tools are to really support architectural design, what may be required is a better communication and co-operation between, on the one hand, the practice and education of architectural design and, on the other hand, the world of research.

At this point, one may ask why we want to bridge this gap at all, in other words, why we attempt to apply CBD to architecture in the first place. The answer is that, although current CBD tools do not meet the requirements from real-world architecture, there are strong notices to the contrary, notices that encourage us not to brush the entire CBD enterprise aside. One notice is the significant success of the ‘Little Brother’ project [Leitch, 1995]. In this project, 1st year architecture students were asked to design a single cell studio in the garden of a house designed by a local architect. In order to do so, they were provided with a unique combination of different sources of design experience: students had access to a ‘living’ architect, who could be inquired about his intentions during the design process; they had access to 'Big Brother', a concrete design product devised by this process; and they had access to the original working drawings of this product. Strictly speaking, Little Brother has nothing to do with CBD, yet it clearly demonstrates that a rich combination of design experiences can contribute to design education, be it without the support of computer technology.

Yet, even in the form of design products, i.e. projects, design experience appears to be a valuable instrument for architectural design. The results of our experiment seem to indicate that design students do benefit from exposure to concrete projects. Although this benefit may be higher for skilled and motivated students, it strongly suggests that CBD indeed has an important part to play in schools of architecture.

Taking stock, it seems plausible to assume that CBD can help to deal with the complexity of architectural design, if further refined, properly implemented and impartially used. Cases are valuable knowledge channels for (student-)architects, and it is quite possible that today’s information technology may further improve their

38 See also chapter 4, p.122.
functioning. Before it can do so, however, there are obstacles to be taken on (at least) three different levels: the model of cognition needs refining, there are implementational issues to be addressed, and the target group must be submitted to a thoroughgoing change in mentality. To recognise these obstacles is one thing, to take them on is yet something different.

The second part of this thesis will make an attempt to bridge the gap between CBD and architectural design. The ideal bridge would eliminate all obstacles at all three levels simultaneously. Yet, since the construction of such a bridge is beyond the scope of a single thesis, we have decided to tackle a manageable subset of obstacles. First of all, we take the challenge to develop a tool that engages users into in-depth explorations of cases, which, as the results of our experiment suggest, is a prerequisite to benefit from the model of CBD. At the level of implementation, we have chosen to address (the lack of) dynamics, as they are at the heart of the whole CBR enterprise. Finally, an attempt will be made to actively involve users in the development of the tool, which seems an obvious remedy to anticipate possible prejudices.

Our bridging operation focuses primarily on architectural design education, but the concepts described are also applicable to the practice of architectural design and, by extension, even to other complex design domains.
II

Bridging the gap between Case-Based Design and architecture
DYNA MO – A DYNAMIC ARCHITECTURAL MEMORY ON-LINE

The idea(l)

“DYNA MO – 1: normally called a generator. A machine that converts mechanical energy into electrical energy by electromagnetic induction; 2: In precise terminology, a generator of direct current – as opposed to an alternator, which generates alternating current.”

(Modern Dictionary of Electronics)
CHAPTER 7

Part I of this thesis has studied the role of cases in architectural design from several different perspectives: CBD-researchers (chapter 3), practising architects and design teachers (chapter 4) and student-architects (chapter 5). By comparing and confronting these perspectives, chapter 6 has attempted to make a diagnosis for CBD’s so far limited contribution to the field of architecture. Due to an inaccurate view of the use of cases in architectural design, combined with problems of implementation, CBD systems still provide limited support for student- and professional architects. Prejudices and fear for design fixation further aggravate this problem by discouraging (student-)architects from using cases during design.

Part II of this thesis is devoted to the development of a possible strategy to help bridge the gap between CBD and architecture. The strategy is called DYNAMO – Dynamic Architectural Memory On-line – and takes the form of an interactive digital design tool built around concrete cases. Whereas this chapter outlines the main ideas behind DYNAMO, chapter 8 describes the implementation of these theoretical ideas as a working prototype. The evaluation of this prototype by student-architects in a 4th year design studio is the subject of chapter 9.

7.1 Introduction

Based on our research about the role of cases in architectural design, we started to develop DYNAMO, an interactive design assistant for student- and professional designers in the field of architecture. DYNAMO can be considered a CBD tool in so far that it was inspired by the cognitive view underlying Case-Based Design, and Case-Based Reasoning in general. CBR, we have pointed out, is a theory and technology within the field of Artificial Intelligence, based on a memory-centred model of cognition.1 Instead of simulating this model within a computer program by using AI techniques, the objective of our tool is to facilitate and nourish this way of reasoning within the human designer's mind. In addition, DYNAMO attempts to enrich the CBR model by situating it in a larger context. The idea is to provide a platform for interaction and knowledge exchange between designs and (student-) designers in various contexts and at different levels of experience.

After a quick review of the cognitive model underlying CBD (section 7.2), section 7.3 will outline the main ideas behind DYNAMO, thereby pointing out how the tool incorporates this model, and at the same time extrapolates it beyond the individual. This extrapolation comes down to stimulating and intensifying interaction, not only between a single user and a computer system, but also between design cases, among (student-)designers, and between architectural practice and education. This should result in a design tool that both feels cognitively comfortable to (student-)architects, and sets up conditions that, judging from our findings so far, are crucial for fruitful case (re-)use in architectural design. Section 7.4 sketches the larger context by describing DYNAMO’s role within the integrated design environment it is part of. This environment, called IDEA+, is based on a conceptual framework of CAAD and

---

1 For a discussion of CBR within the context of AI, see chapter 2.
aims at supporting architects from the very start of the design process. Finally, section 7.5 points out how DYNAMO may contribute to bridging the gap between CBD research and architectural design, by indicating how it addresses the problems current CBD tools are facing.

### 7.2 The cognitive model of CBD

Implicitly, or otherwise, all CAAD tools embody a particular view of design and designers [Tweed, 1998]. Central to the view embodied by DYNAMO is the model of cognition underlying CBD, and CBR in general. There is neither space nor need to repeat all details of this model in depth here. Instead, we will focus on two aspects that are crucial to understand the main idea behind DYNAMO, and point out their relevance for the field of architectural design. ²

#### 7.2.1 CONCRETE CASES

One aspect in which CBR differs from other views of reasoning is that people’s knowledge does not only consist of abstract, generally applicable principles, but also of concrete, specific experiences (cases) [Riesbeck & Schank, 1989; Kolodner, 1993]. Within the domain of architecture, we have pointed out, the term ‘experience’ can be interpreted in several ways. Architects acquire design experience by designing themselves as well as by observing the design process and/or design products of other designers [Heylighen, 1998].³ What all these interpretations have in common, is that there is a concrete design project at stake. Concrete projects indeed play a key role in the field of architectural design, in practice as well as in education.

Practising architects are typically concerned with the actual concreteness of the specific objects they are designing rather than with generalisations [Liddament, 1999]. Moreover, although they are often extremely reluctant to admit it, architects call in projects from the past during several stages of the design process including concept generation.⁴

Also in design education concrete cases (can) play a prominent part. The particulars in a given project offer student-architects an integrated view of design issues that would be lost if these issues were taken up separately. By consequence, exposure to cases during the design process is likely to have a positive effect on the quality of students’ design project, in particular on the quality of their design concept. This positive influence, however, turns out to occur mainly for the more skilled and motivated students.⁵ In order to profit from previous cases during conceptual design, students need, on the one hand, enough motivation and interest to go (deeply) into others' design projects and, on the other hand, sufficient background knowledge and skill to understand and interpret the underlying design concepts.

---

² For a description of this model of cognition, see chapter 2, pp.47-51.
³ See also chapter 6, pp.144-145.
⁴ See the interviews with practising architects in chapter 4.
⁵ See the experiment with student-architects in chapter 5.
7.2.2 DYNAMICS

A second aspect of CBD’s cognitive model we would like to dwell on, is the dynamic behaviour of human memory. Being based on the Theory of Dynamic Memory, the model claims that memory is dynamically changing with each new experience [Schank, 1982]. Thinking of memory as a constantly growing trace of experience seems more plausible than considering it a static knowledge base, as more traditional cognitive models do. Both Rule-Based and Model-Based Reasoning reduce the human mind to a mere repository of general principles. When it uses its knowledge, that knowledge is not changed, which obviously does not correspond to the way in which human memory works. Designers and people in general learn from their experiences and using memory itself is such an experience.

The model does not only claim that memory changes, but also proposes some specific kinds of changes. The simplest way in which memory improves its behaviour is obviously by acquiring new cases. Each experience is stored in memory, providing it with additional familiar contexts for new situations. As already mentioned, cases in architectural design can be acquired by actively designing, but also by visiting a building or browsing architectural magazines.⁶

Sometimes, however, a case is not immediately stored in the right way. This causes the case to be recalled in a situation for which it turns out to be irrelevant or inappropriate. Hence, a second manner in which memory changes is by re-indexing already stored cases. It allows memory to fine-tune its recall apparatus so that it remembers cases at more appropriate times. As will be illustrated in the following section, in architecture it is even highly questionable whether there exists such a thing as the ‘right way’ to store a design case in memory.

A third kind of memory change is the creation of new generalisations. This way of learning derives from the fact that what starts as a new experience, different from the norm, eventually might become the norm. What is considered normal in architecture today was not normal ten years ago [Tweed, 1998]. And what goes for architecture in general, equally applies to individual architects, design offices or architecture schools.

7.3 DYNAMO: an (inter-)active workhouse

Convinced of the importance of concrete cases in architectural design, and inspired by the Theory of Dynamic Memory, we decided to develop DYNAMO. At first stage, the tool is developed for design education, more specifically for the design studio.⁷ However, as will be pointed out further in this chapter, its full potential will only be

---

⁶ See chapter 6, p.145.
⁷ The design studio is a quite particular format of education that is regarded as the cornerstone of design education in many schools of architecture. In the studio, students work on small, yet realistic design projects tutored by more experienced architects. The studio offers student-designers an ‘espace transitoir’ on their way to real practice [Winnicot, 1971]. For a more detailed description of the design studio, see [Schön, 1985].
DYNAMO – THE IDEA(L)

exploited when used in professional design offices as well.\textsuperscript{8} DYNAMO, which is intended to assist (student-)architects during conceptual design, tries to kill two birds with one stone. At short notice, it provides (student-)designers with a rich source of inspiration, ideas and design knowledge for their present design task, as it is filled with a permanently growing collection of design cases. Its long-term objective is to initiate and nourish the life-long process of learning from design experience as suggested by CBD's model of cognition.

CBD provides us with a plausible explanation of how people, and more specifically designers, acquire (design) knowledge. Yet, like all design theories, it privileges one particular view of design, thereby obscuring other equally valuable aspects. In case of CBD, design is considered predominantly a cognitive activity, whereas the social aspects of acquiring design knowledge are largely underplayed. In the field of architecture, however, concepts and insights are developed and renewed as much through interaction as by individuals in isolation.\textsuperscript{9} DYNAMO can become an important theatre for this interactive process of concept generation, by offering a platform where different forms of interaction can take place, and at the same time acting as a repository to nurture these interactions and store their results. Therefore, the tool is conceived as an (inter-)active workhouse rather than a passive warehouse [Schank & Cleary, 1995]: it is interactively developed by and actively develops its users’ knowledge.

7.3.1 BETWEEN DESIGN CASES

A first form of interaction DYNAMO supports is the confrontation between different design projects. Consider, for instance, the recent extension of the Musée des Beaux Arts in Lille designed by Ibos & Vitart and Rossi’s Bonnefantenmuseum in Maastricht. Both projects share the same programme, the period in which they were built, and their location nearby an existing building – the old Musée des Beaux Arts and the Wiebenga Hall respectively. Yet, the way in which the new building stands towards the old is quite different. In Maastricht the approach can be coined ‘juxtaposition’, whereas in Lille the relation between both buildings is one of ‘reflection’, as the new museum is literally a mirror for the old (see figures 7.1 to 7.3).

Vice versa, a similar concept may underlie quite different designs. Take for instance two projects by Robbrecht and Daem, the Aue pavillions for Documenta XI in Kassel (see figure 7.4) and the office buildings for the Katoen Natie in Antwerp (see figures 7.5 and 7.6). Although these designs differ considerably in programme,

\textsuperscript{8} Henceforth DYNAMO users will therefore be referred to as (student-)architects or (student-)designers, meaning students participating in a design studio and/or professional architects working in a design office. Both kinds of users are actively involved in designing.

\textsuperscript{9} The view of architectural design as a sequence of cognitive operations conducted entirely within one brain is one of the ideas found wanting in Lawson’s book Design in Mind [Lawson, 1994]. Moreover, as pointed out in [Heylighen e.a., 1999], conversation between student-designer and design teacher plays a key role in the development of meaningful design concepts.
site and context, one of the major ideas underlying both is that of two trains driving at a different speed passing each other.

**Figures 7.1, 7.2 & 7.3.**
Extension of the Musée des Beaux Arts in Lille (France) by Jean-Marc Ibos and Myrto Vitart, 1997; front and side view of Aldo Rossi’s Bonnefantenmuseum in Maastricht (The Netherlands), 1993.

This kind of confrontation between dissimilar projects based on similar concepts – and vice versa – can provide crucial insights into a specific project, as well as into architectural design in general. In order to support and stimulate such confrontations, DYNAMO has at its core a web of indices that allows retrieving and browsing between design cases in multiple ways. Every project is labelled with several features, the so-called indices, and linked to projects with common characteristics. If we consider design cases as encapsulations of design knowledge, this web of indices
DYMONO – THE IDEA(L)

further enhances each case’s value. It allows users to approach a design from different perspectives and to situate it in relation to other designs. The knowledge content of DYNAMO therefore does not only reside in the cases it contains, but also in the web of indices between them, and thus changes dynamically when new cases are added or old cases get new indices.

Figures 7.4, 7.5 & 7.6.
Aue Pavillons in Kassel (Germany), 1992;
Offices for the Katoen Natie in Antwerp (Belgium), 1993,
both by Paul Robbrecht & Hilde Daem.

7.3.2 BETWEEN HUMAN DESIGNER AND COMPUTER

In fact, a similar indexing system can also be found in many other multimedia tools, digital libraries and CBD systems. Most of these tools, however, do not fully exploit CBD’s cognitive model in that they are static in nature: just like a warehouse, they inventory information that can be consulted, yet these consultations leave the information unchanged. DYNAMO, on the other hand, is conceived as an active workhouse rather than a passive warehouse, meaning that it is interactively developed by its users’ knowledge.

10 Think for instance of Archie-II and PRECEDENTS, two CBD tools examined in chapter 3 (pp.61 ff and pp.83 ff), or of digital libraries like archINFORM or LORA, which will be discussed in the following chapter (p.203).
A second form of interaction DYNAMO supports is namely the interaction between human user and computer system. The tool is interactive in the sense that users cannot only *use* the web of indices to consult and navigate between cases in memory, they can also *change* and improve memory as suggested by CBD’s cognitive model, i.e. by adding new projects, making new links between them or creating extra indices.

In a field like architectural design, the choice for an open-ended approach to case collection almost goes without saying. Every single day buildings are (being) designed, built, extended or converted, which makes DYNAMO already outdated at the very moment it is implemented. Therefore, we have opted to actively involve the user in case collection by enabling him to add design projects in a user-friendly way – projects designed by himself or buildings he has visited and/or finds particularly interesting.

Plausible as this option might be for case collection, extending this approach to (re-)indexing probably needs some further explanation.\(^{11}\) As mentioned already, it is highly questionable whether there exists a ‘right way’ to store a design project in memory, in other words, whether it is possible to label a project once and for all by a set of indices [Heylighen, 1997]. One reason is that the meaning and interpretation of a project – and thus its position among other projects – changes over time, due to changes in its use, other projects imitating or opposing it, or new developments in its environment. Take for example Van den Broek & Bakema's Auditorium of the Technical University in Delft (The Netherlands). Until recently, this colossal concrete building looked like a spaceship dropped on the campus by accident. With the arrival of Mecanoo's new University Library, however, the Auditorium has gained a totally new context. The spaceship, rather than having fallen from the sky, has landed at the foot of an inviting green hillock, as if the place was always meant as a landing pad for spacecraft [Wortmann, 1998].

A second reason for involving the user in the process of (re-)indexing is that an architect will not interpret the same project in the same way all the time, since his present design task may influence his interpretation considerably. Being involved with a specific design task makes him look differently at previous projects and come to new insights, leading to new or additional indices. When struggling with daylight in a deep terrace house, for example, he may discover that the Katoen Natie is also a good example of a building successfully illuminated with skylight (see figure 7.6), and may then add this feature to the case’s indices. This phenomenon of ‘just-in-time’ interpretation is clearly expressed by one of the interviewees when talking about his (paper) case base of hospital projects. Part of the information on these hospitals has already been assimilated, yet an enormous lot is still raw material, and is only

\(^{11}\) Re-indexing can take two forms: 1. adding another value to an existing (kind of) index – for instance, if the index ‘concept’ of say Centre Pompidou contains the value ‘machine’, users may want to add the value ‘inside out building’ to the ‘concept’ index of this case; 2. adding another type of index – for instance, in addition to ‘concept’, users may want to index cases by ‘colour’. Both forms of re-indexing are supported by DYNAMO (see also chapter 8, pp.197-201).
processed in response to a concrete design task.\textsuperscript{12} In a similar way, another interviewee describes how he had always been fascinated by an image of Mies’ Riehl house. However, it was not until he was struggling with a specific design, that he studied the house in more detail (and discovered that, in fact, it was not such a good house at all).\textsuperscript{13} Because we cannot anticipate all possible design tasks the user will be involved in, let alone all possible situations under which a particular project may be of interest, we cannot index cases in advance once and for all. Therefore, we have opted for an open, interactive indexing process.\textsuperscript{14}

In summary, calling in the user, i.e. a (student-)architect actively involved in designing, in case collection and (re-)indexing has the advantage that DYNAMO can draw on his knowledge to permanently nourish the content of its case base and refine its indices, so as to stay in tune with the evolving needs, preferences and insights of the user.

7.3.3 BETWEEN (STUDENT-)DESIGNERS

This interactive approach to case collection and indexing, however, is only feasible when the tool is used collectively by multiple (student-)architects. By inherently compensating for the weaknesses of individual users, this collective use creates a whole potentially stronger than any individual designer [Press, 1998]. DYNAMO therefore can be called interactive, not only in the traditional sense of human-computer interaction, but also because it supports interaction among students, design teachers and practising architects in multiple ways.

First of all, it gives student-architects access to the work – and thus the design knowledge – of architects outside the design studio. In most theoretical courses like architectural theory and history, students are confronted with a fairly narrow canon of projects by exemplary architects, usually through a static slide show, or through books and magazines in the library. DYNAMO provides them with a much broader collection of cases, and this in a way that better fits their designerly way of thinking. Catalogues of slide collections and libraries typically serve historians and theoreticians better than designers. Whereas the former regard an existing design project as a subject to study, the latter perceive it as a source of images and concepts that can drive their own projects. Unlike library catalogues, DYNAMO makes cases accessible through issues (i.e. indices) that are at stake during design – issues at both concept and component level.\textsuperscript{15} Moreover, being part of an integrated CAAD environment,

\textsuperscript{12} See chapter 4, p.119.
\textsuperscript{13} See chapter 4, p.120.
\textsuperscript{14} See also the discussion on case representation (p.178).
\textsuperscript{15} Unlike Yoshitaka Mishima and Peter Szalapaj, we do not contend that the use of conventional media, such as slides, books and journals, is insufficient for students to understand buildings entirely [Mishima & Szalapaj, 1999]. What we do contend, is that information technology allows to organise and structure the information on existing buildings in a way that better fits the needs of students (and professional architects) during design.
DYNO allows the user to consult other architects’ work during design without having to walk to the library or even switch to another computer programme.\footnote{For a more detailed description of this CAAD environment, see section 7.4.}

A second way in which DYNAMO supports interaction between (student-) designers, is by serving as a collective external memory for the students and design teachers in the studio.\footnote{The notion of collective external memory is taken from [Wegner, 1987].} The tool enables them to store and share knowledge, expertise and insights accrued in the course of design assignments. At the start of a new assignment, design teachers can load DYNAMO with projects they find particularly relevant, and index them in such a way that this relevance becomes clear. In the course of the assignment, students may add other interesting projects to this repository, which acts to inform the assignment and generate well-founded design solutions.

Finally, DYNAMO has the potential to increase the frequency and quality of the dialogue among design students as well as between students and design teachers. The importance of this dialogue in design education should not be underestimated. In fact, the entire teaching process in the studio builds upon a conversation between students and professional architects, c.q. design teachers. Rather than from some mysterious process in an individual designer's mind, it is often through such conversations that new concepts in architecture emerge. The more frequent and richer this dialogue, we have pointed out, the better students are able to develop a meaningful concept for their design [Heylighen e.a., 1999]. DYNAMO has the potential to increase this frequency and intensity in several ways. Newly added cases or indices, for example, may act as points of departure for discussion by highlighting specific aspects of the assignment not initially shared. Moreover, by enabling DYNAMO users to comment on cases, the tool can serve as an on-line discussion forum where students and design teachers can share their views on existing projects.

7.3.4 BETWEEN PRACTICE AND EDUCATION

A fourth level of interaction DYNAMO aims to support, is between on the one hand design education, and on the other hand the world of practice for which student-architects are being prepared. In most architecture schools today the relation with professional practice is limited to hiring practising architects for studio teaching. Being accessible to both students in architecture schools and professional architects in design offices, DYNAMO provides an opportunity to expand and at the same time intensify this relationship. Architects working in practice may draw fresh insights from students' work. Vice versa, input from practice – design projects as well as indices – keeps schools permanently updated about the constantly changing problems and processes within the profession and the society at large, which enables them to formulate design assignments of topical interest. This does not mean that design education should submit its agenda unconditionally to the hot topics in practice. Instead, it should take those topics further and, not hindered by 'practical' constraints, play the role of an experimental laboratory, a think tank for architectural design. In this way, DYNAMO offers the prospect of building a bridge – made up of cases –
between education and practice, by facilitating productive exchange of design efforts between both parties.\textsuperscript{18}

7.3.5 ACTION

Although DYNAMO itself does not perform any form of (case-based) reasoning, it nevertheless can be ascribed a certain form of action, in the sense that it actively develops users’ design knowledge by stimulating them to learn from previous projects. When adding a new case to DYNAMO, for example, the user is responsible for representing and labelling the project in such a way that important aspects are legible and easily accessible [Akin, Cumming, Shealey & Tuncer, 1997]. This forces him to view the project from different perspectives and to answer questions like ‘what does this project tell me?’, ‘in what circumstances might it be a relevant example?’ and ‘where does it fit in relation to the other projects?’ – questions that enhance the user’s ability to understand the project and remember it later on [Schank & Cleary, 1995]. In this way, the tool promotes the kind of thinking that helps to learn in a better way from existing designs.

Using DYNAMO in the design studio, i.e. during design, in addition ensures that design knowledge is gained in the same context as it will be used. In general, students very rarely bring knowledge acquired in theoretical courses to the studio [Marda, 1997]. The reason is that they have not mentally encoded the material in terms of issues they face when designing. Because they have not thought about how it helps to solve problems at stake during design, the knowledge is difficult to access when they do face such a problem [Schank & Cleary, 1995]. In the studio, on the other hand, the project in which students are involved provides them with a framework to organise newly acquired knowledge. Thus, their project does not just serve as a motivator – it stimulates students to learn because they are eager to design a good project – but also as a guiding context to integrate what they learn.

7.4 The larger context

7.4.1 A CONCEPTUAL FRAMEWORK FOR CAAD

Research in the field of CAAD has gone through many generations and philosophical perspectives. In practice, however, the bulk of today’s design software can hardly be called a spectacular \textit{aid} in the design process, if an aid at all. One of the reasons, in our opinion, is that architects are still rather dancing to the tune of the software industry than vice versa [Neuckermans & Heylighen, 1999]. In order to cover the

\textsuperscript{18} Perhaps, our view is rather naïve and the future may force us to reconsider it. Practice and education each have their own logic and are driven by quite different forces. Nevertheless, our idea perfectly chimes with the current tendency towards life-long learning and post-academic education. Therefore, until further notice, we stick to our naivety. If our university advises its graduates to ‘Take a maintenance contract with your diploma’, why not encourage recently qualified architects to subscribe to DYNAMO?
broadest market with the least amount of products, many software companies offer CAD packages with the most general applicability. Yet, while designing, architects do not think in terms of polylines, 3D faces or offset. They work with concepts like types, spaces, walls, columns or staircases. The only way to really integrate the computer in the design process is therefore to develop software that is in tune with this designerly way of thinking.

Moreover, a real CAAD system should offer support from the very start of the design process. Today most architects do not switch on the computer until the design is more or less ‘designed’ and only needs to be visualised. Commercial CAD packages have proven to be a quick alternative for the traditional drawing board, but actually provide very little support in the early conceptual stage of design where creative and evaluative tasks are on the agenda [Carrara, Kalay & Novembri, 1991]. As first step in the design process, the conceptual phase is in some sense the most influential. Commitments made here have the largest effects and are the hardest to undo later. Thus, if we want to improve design, then conceptual design is certainly an area with the potential for a high pay-off. Building an architectural model right from the start on computer allows using the machine to compute tests early in the design process. Moreover, it avoids the unproductive translation of a hand-made design into a digital version later on [Neuckermans, 1992].
Given this need for more designerly appropriate software on the one hand, and for earlier use of computers in the design process on the other hand, a general conceptual framework for Computer-Aided Architectural Design has been worked out by Neuckermans [ibid.]. In this framework, illustrated in figure 7.7, the building program is subdivided in three levels of detail spanning the normal scope of architectural design: master plan, building block and room or space. Extension both upwards to the urban design scale and downwards to building elements is still possible. Each level can function as entry point for the design process, allowing the architect to start at the bottom, at the top or in the middle, depending on his personal preference and/or the scale of the design problem at hand. Whichever level he chooses, the architect can start building his CAAD model with design entities he is familiar with. Building types, spaces and elements like walls, columns and doors take the place of polylines, ruled surfaces and other strange CAD objects. Optionally, these entities can be positioned on grids with densities varying from one level of detail to another. The glue that holds the design entities together is a central data model, which defines the relationships between entities on the same level as well as across different levels [Hendricx, 1997; 2000]. Apart from levels, entities and grids, a fourth ingredient of the framework is a set of tests that allow checking the design in every stage of the design process. Possible tests concern cost/m² or cost/m³, compactness, energy requirements, traffic, morphology, views and sights, shade and shadowing, temperature fluctuations, level of insulation, comfort prediction, daylighting or elementary stability. The tests differ from level to level and from design phase to design phase, depending on the detail of the design at that moment.

Instead of imposing a fixed way of designing (for instance, starting with spaces and deducing the necessary building elements next), the framework tries to support the architect’s personal design approach: starting at any level in the hierarchy, with any design entity, either top-down or bottom-up. It remains primarily graphical, open, not deterministic but interactive. It aims at serving the designer, not at steering his design process.

7.4.2 IDEA+

Based on the conceptual framework outlined above, an Integrated Design Environment for Architecture called IDEA+ is being developed [Hendricx, Geebelen, Geeraerts & Neuckermans, 1998]. Figure 7.8 sets out the implementation scheme of this environment, which consists of several modules. The core module is an object-oriented data model that specifies the behaviour of and relationships between design entities. This data model serves as common basis for in-house developed modules, each of which implements a specific ingredient of the conceptual framework. At this moment, a 3D-modeller and a daylighting test IDEA-I are under development [Geebelen, 2000]. In addition, the environment allows plugging in external modules, for instance graphical packages like AutoCAD or Microstation. Unlike native IDEA+ modules, these external applications communicate with the central data model through special interfaces.
All these tests check purely computable, i.e. quantifiable performances. In architecture, however, a design must score and thus be evaluated on many other aspects as well.\textsuperscript{19} In spite of the tendency to undervalue what cannot be expressed in objective, quantifiable data, the performance of a building is by no means simply a reflection of the physical environment itself [Jockusch, 1991]. Aspects like quality of space, form and light, contribution to social coherence, historic significance, or impact on the environment are equally, if not more important. Since these \textit{qualities} cannot be computed by means of rules, formulas or exact computer programs, architects entirely depend on their own intuition and experience in predicting them. Hence the high degree of uncertainty, which is so characteristic of design. To some extent designing is always taking a plunge into the future, making a ‘guess’ at the nature of the world that the designed object will encounter [Jachna, 1995]. Indeed, architects and other designers permanently take decisions and options with an eye to situations that can only become reality after the designed object has been realised [Neuckermans, 1994]. Yet, difficult as they might be to predict, the less ‘computable’ qualities of a design cannot escape existing design projects. Hence the idea to mobilize information on already realised designs for the evaluation of new ones. DYNAMO may help the user predict how his ‘primitive’ design performs on a certain aspect by drawing analogous assumptions from performances of similar projects in the case base. In this way, IDEA+ does not only allow architects to test quantifiable aspects of a design early in the design process, it also draws their attention to \textit{all} aspects of the options they take, thus stimulating the awareness of the downstream implications of their decisions.

\textsuperscript{19} Remember the various ‘components’ of architectural design discussed in chapter 1, p.9.
7.5 An alternative to (current) CBD systems?

As pointed out in part I, the CBD systems we examined fail to satisfy the contemporary request to support architects’ design process in several important respects. Table 7.1 summarises the main reasons for this failure. It lists the factors that have been identified as necessary to the current gap between CBD and architectural design. At this point, one may ask whether DYNAMO will help to bridge this gap, in other words, how DYNAMO tackles the problems current CBD systems are struggling with. Each of these problems will be considered briefly here to conclude our theoretical description of DYNAMO.

<table>
<thead>
<tr>
<th>level</th>
<th>issue</th>
<th>problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. model of cognition</td>
<td>design experience</td>
<td>(over-)emphasis on design product</td>
</tr>
<tr>
<td></td>
<td>case retrieval</td>
<td>reduction of relevance to similarity</td>
</tr>
<tr>
<td></td>
<td>view of architectural design</td>
<td>problem solving</td>
</tr>
<tr>
<td>2. implementation</td>
<td>case representation</td>
<td>reduction of architecture to geometry</td>
</tr>
<tr>
<td></td>
<td>integration</td>
<td>lost</td>
</tr>
<tr>
<td></td>
<td>dynamics</td>
<td>lost</td>
</tr>
<tr>
<td>3. context of use</td>
<td>design teachers &amp; student-architects</td>
<td>prejudices and fear for design fixation</td>
</tr>
</tbody>
</table>

TABLE 7.1. Possible reasons for the so far limited success of CBD tools in architecture.

7.5.1 COGNITIVE MODEL

7.5.1.1 Design experience

To learn from design experience is the essence of CBD. The entire approach builds upon cases, being interpreted representations of real design experiences. In most CBD systems, cases are design products, usually but not necessarily at the final stage of the design process. According to architects, however, it is not so much the design product, but rather the design process that matters. A first element of critique on current CBD systems is therefore their biased portrayal of design experience in architecture. Although design products definitely contribute to the acquisition of design experience, they do not do full justice to what learning from experience means in architectural design.

Some may argue that our blanket critique on CBD’s view of design experience is betrayed by our enthusiastic use of design products in DYNAMO. Indeed, cases in DYNAMO represent design projects and not design processes. However, by actively involving the user in case collection and indexing, these projects stimulate reflection on the process that conceived them in several ways.

---

20 See chapter 4, p.112.
First of all, by enabling users to add projects from their personal repertoire, DYNAMO benefits from ‘authentic’ design cases, i.e. projects described by the architect who designed them. Such cases are authentic in the sense that they come directly from an architect’s design experience, and as such have properties that make them particularly appropriate to learn from. It is one thing to read what a critic thinks about a piece of architecture, but quite another to see it described by the person who conceived it. When architects describe a building they have designed, their description tends to echo the design process behind it. Because authentic cases include not just a description of the project, but also the designer’s perspective on this project, they are likely to reflect how it was thought about and perceived during design.

A considerable portion of DYNAMO’s cases, however, will be ‘second hand’, i.e. described by people other than the architect in question. In order to provide at least some information on how these projects were conceived, DYNAMO will include interviews with and/or written comments by the architects as much as possible.

Yet, whether authentic or second hand, supplemented with interviews or other process-related comments, simply exposing users to design cases is insufficient to stimulate and support learning from design experience. The level of motivation of the user as well as his background knowledge determine how much he will learn, and how useful the cases will be to address his needs during design. Motivation makes one willing to step into a project and reflect on how it has been conceived, whereas background knowledge helps to understand this concept. DYNAMO cannot directly control either of these. However, it can define the role of the user in ways that force him into in-depth explorations of cases and, on the other hand, provide (extra) background knowledge to make sense of them.

One way to trigger in-depth explorations of a design case is by asking users, i.e. student- or professional architects in the midst of a design process, to write a comment on the case and share their view with other DYNAMO users. It is our hope that this on-line discussion will invite the user to go deeply into the case and, being involved in designing himself, to reflect on how it has been conceived. A second way to stimulate such reflection is by enabling users to add new projects they have visited or find particularly interesting. Adding a project forces users to consider the project from different perspectives and to compare and contrast it with other projects in the case base. These other projects provide the user with (extra) background knowledge, which may be helpful to interpret the case at stake.

In summary, instead of purely transmitting information on design products to the user, DYNAMO invites him to explore how they were conceived, by submitting new cases and by commenting on cases already available. In this way, projects act as an alibi, a trigger to reflect on the design process behind it, while other projects they are linked to provide the user with background knowledge for interpretation.

---

21 In this respect, authentic cases are comparable to what Burke calls first-person stories, stories about events the storyteller has experienced himself [Burke, 1998 pp.176-177].

7.5.1.2 Retrieving relevant cases
A second element of CBD’s cognitive model that has been found wanting in interviews with architects, is the retrieval of relevant cases. The model assumes that one or more cases are retrieved when confronting a design problem. According to our interviewees, on the other hand, the decision to draw on a particular case is taken when confronted with the case itself, which is often before a new design problem prompts for support. DYNAMO allows for such timely case retrieval by means of its interactive approach to case indexing. When coming across an interesting case in DYNAMO, users can immediately mark it as relevant for future (re-)use.

7.5.1.3 View of architectural design
Our most fundamental point of critique on CBD research is its view of architectural design as a form of problem solving. This view speaks from the ambition to automate (parts of) the design process, as for instance in CADRE, FABEL or IDIOM. Because an architect’s task is much more complex than solving tame problems, it is highly questionable whether the way to high quality architecture leads via automation [Heylighen, 1996]. Consequently, DYNAMO only provides cases in a format that fits architects’ designerly way of thinking, yet this thinking itself is left entirely to the human user. The tool does not offer any support for case manipulation, as this task, in our opinion, has to remain with the architect himself. The only support DYNAMO offers is the storage of the user’s design results in the case base and their potential reuse for future situations.

Moreover, DYNAMO is part of an integrated design environment that aims at supporting the human designer, not steering his design process. The IDEA+ user may start designing at whatever level he prefers, put his design to the tests he considers necessary, and consult cases whenever he feels like it. Neither DYNAMO, nor IDEA+ in general force architects to follow a fixed sequence of steps, be it problem specification-solution-evaluation, or retrieval-manipulation-storage. Instead, they enable them to proceed in the way they are used to or feel comfortable with.

7.5.2 IMPLEMENTATION

7.5.2.1 Case representation
When it comes to implementing CBD systems, a major advantage of using cases is often lost. The richness of a concrete design is difficult, if not impossible to represent in a computer memory, especially when this representation must follow a computer-readable format. In CBD systems, design cases are therefore reduced from rich sources of design knowledge including all aspects that architectural design involves, to 2D layouts or feature vectors.

In order to fully appreciate the richness of a design case, the ideal would be to visit the site and the building itself, talk with the building users, query the architect about his intentions during the design process, and inspect his design sketches and working

---

23 See chapter 3 pp.66 ff, pp.72 ff and pp.78 ff.
24 See the case studies 3.4 and 3.3 respectively (chapter 3, p.78 and p.72).
Because this is not always possible – interesting buildings do not always stand around the corner, and if they do, they are not necessarily open to the public – the next best alternative is to provide the user with a description of the case that is as rich and varied as possible.

Since DYNAMO itself does not perform any kind of manipulation, cases can contain information in whatever format: photos, drawings (sketches, plans, sections, elevations), 3D models, videos (of a visit to the building or an interview with the architect), textual descriptions (by the architect, critics, theoreticians or historians), and even (for public buildings) links to a Web camera on the site.26

In addition, the varied and interactive indexing system of DYNAMO further enhances the richness of its cases, not only because of the different types of indices – these are important as such – but also because of the different interpretations, i.e. values users ascribe to these indices. It is, for instance, completely wrong to label the villa Savoye just as a single-family house, for this building is many things at the same time. It is not only a built manifesto of Le Corbusier’s ‘five points of new architecture.’27 It is not only an exploration of the possibilities of the – at that time – modern construction material reinforced concrete. It is all that, but it is at the same time a lot more. Every DYNAMO user makes his own image of the villa Savoye, for every (student-)architect tends to project his actual concerns onto it, being the concerns of his design task at hand. The image thus shifts with every interpreter, yet at the same time becomes richer, whereby the final conclusion could be: that villa Savoye is actually fairly complex. Maybe architecture is indeed somewhat more complex than it is often presented. By providing an open indexing system, DYNAMO ensures that the interpretation of the complex richness of a design case is never closed.

7.5.2.2 Integration
A second advantage of using cases that poses serious problems of implementation is the integration of different aspects within a single design case. Just like a case’s richness, we have pointed out, this integration is often lost in current CBD systems. Since DYNAMO does not perform any manipulation of cases itself, this loss is not an issue for the tool as such. Yet, it is one for the CAAD environment it is part of. In fact, it is precisely this need for (maintaining) integration that motivated the development of an object-oriented data model, which forms the core of the environment and provides the glue between all (in-house developed) IDEA+ modules [Hendricx, 1997; 2000].

25 It could be argued, indeed, that the best representation of a design case is the building itself. This may explain the overwhelming success of the Little Brother project, where students had access to a concrete building, its architect and his original working drawings [Leitch, 1995]. See also chapter 4, p.122.
26 Moreover, an additional advantage of using authentic cases in DYNAMO is their tendency to be vivid and full of detail [Burke, 1998 p.177].
27 These five points are: 1. columns (pilotis), 2. the roof garden (toit-jardin), 3. the free floor plan (plan libre), 4. the strip window (fenêtre en longueur), and 5. the free façade (façade libre).
7.5.2.3 Dynamics
Being rooted in the Dynamic Memory Theory, a major promise of CBD research was that it would produce design tools that learn, i.e. develop and improve their performance dynamically each time they are used. However, few of the CBD systems studied live up to these expectations. In Archie-II and PRECEDENTS, for instance, the sophisticated case representation and indexing scheme seriously hamper dynamic change.

In our opinion, this dynamic change is at the very heart of the whole CBD enterprise, and therefore a vital ingredient of every CBD system. DYNAMO, we have pointed out, changes dynamically as it is used in several ways: users continuously feed the system with new cases and add information to, comment on or re-index cases already stored.

7.5.3 USE

If DYNAMO manages to live up to all the expectations described above, it will have addressed an important portion of the problems CBD research is currently facing. But, as important as it might be, it is not enough to guarantee a convincing breakthrough of CBD in architecture. Judging from interviews with architects, a major obstacle for this success may very well be the prejudice among the target group. Some design teachers and student-architects are quite explicit in their fear for design fixation. Due to an (over-)emphasis on originality, cases are banned from informing design in architectural education.

The only solution to this problem seems to be a serious change in mentality. Such changes, however, typically take a long time, which transcends the scope of a single thesis. Nevertheless, DYNAMO somehow addresses this problem by trying to anticipate possible prejudices against itself. The obvious way to do this is by allowing users to have a say in its development as much and as early as possible. Design teachers actively participate in DYNAMO’s development in the sense that they can ‘load’ or ‘seed’ the tool with projects they find relevant. Students are involved by submitting additional cases, writing on-line comments, etc. This strategy of user participation carries through into the very process of implementing DYNAMO. As will be described in chapter 9, design teachers and students are engaged in testing a prototypical system at an early stage of implementation.

7.6 Summary

Inspired by the view of cognition underlying CBD, we have sketched DYNAMO, a(n) (ideal) tool to support (student-)architects during conceptual design by means of concrete design cases. At first stage, the tool is intended for the design studio, yet with the potential of expansion into the office setting. In this chapter, we have outlined the theoretical ideas behind the tool, thereby pointing out how it incorporates CBD's view of cognition while at the same time extending it beyond the individual. In addition, we have situated the tool within the context of IDEA+, an integrated design environment.
CHAPTER 7

that supports architects from the very start of the design process. Finally, we have pointed out how DYNAMO may (help) bridge the gap between CBD research and architectural design by indicating how it addresses the problems identified in the previous chapter.

‘Hard-core’ CBD researchers will probably judge DYNAMO unworthy of the label ‘CBD’, because the tool itself does not perform any (case-based) reasoning. Indeed, since cases are opaque to the system – i.e. it does not have any knowledge of the cases beyond their indices – the only thing DYNAMO can do is retrieve them, based on the values of their indices. Whether and how these cases are (re-)used for new design tasks is left completely to the user. Yet, one cannot deny the concept of the tool being firmly rooted within the Theory of Dynamic Memory. Moreover, its interactive approach to (re-)indexing does, if not supply, then at least stimulate the kinds of awareness needed to effectively call in previous cases during design, i.e. a productive 'balance' between relevance, innovation, difference, etc. [Liddament, 1998].

Nevertheless, the objective of our research is obviously not to obtain the CBD – or whatever other – label, but to clarify and support the use of cases in architectural education and practice. The utility of CBD is evidenced in a wide range of disciplines. Yet, as we also have noted, it upholds a particular view on design which cannot be forced to fit our growing understanding of architectural design as a complex, socially mediated activity, and of the role of interaction in the development and renewal of design knowledge. Therefore, we have opted for an extrapolation of CBD so as to embrace this social aspect of architectural design. This might help to generate a design tool that is cognitively comfortable to designers (as it is based on their cognitive behaviour) and at the same time engages them into interaction with other designers in different contexts and at different levels of experience.

Chapter 8 will now discuss the implementation of these theoretical ideas in a working prototype, followed by an evaluation of this prototype by 4th year design students in chapter 9.
A WORKING PROTOTYPE

Implementation
CHAPTER 8

This chapter turns from the theoretical issues of DYNA, which have been outlined in the previous chapter, to its implementation as a working tool. A first prototype has been built within the scope of a graduate’s thesis [Segers, 1998] and has subsequently been further developed, extended and refined [Heylighen, Segers & Neuckermans, 1998a, 1998b; Heylighen & Neuckermans, 2000b]. Because we have opted for a Web-based approach for this implementation, section 8.1 starts with a general introduction into the key elements and terminology of the World Wide Web, thereby pointing out how Web technology allows us to translate our theoretical ideas into a working prototype. Readers who are already familiar with this technology may want to skip the first section, and immediately go for the second one, which describes the prototype itself. To be precise, it outlines the three major components the tool consists of: a collection of cases, a database that structures this collection, and a user interface to consult and modify the collection. After simulating a possible scenario of using the prototype (section 8.3), the chapter closes with situating the tool in relation to other comparable systems, both inside and outside the field of Case-Based Design (section 8.4).

8.1 Introduction

The previous chapter has given a description of (the ideal) DYNA, a Dynamic Architectural Memory Online that supports (student-)architects during conceptual design by means of concrete design cases. Translating these theoretical ideas into specifications for a concrete tool, at least three preconditions must be fulfilled by the prototype:

1. Cases must be represented with as much and as varied information as possible. Furthermore, they should be easily connected to other cases, as these are to act as background knowledge for their interpretation.1

2. The case base should be simultaneously accessible to users at different locations – student designers in architecture schools and professional architects in design offices.2

3. The content and structure of the case base must be able to change dynamically as it is used.3

With these three demands in mind, we have opted for a Web-based approach to build our prototype. The objective of this section is therefore to give a brief introduction into Web technology, and to point out how this technology enables us to implement these specifications in a working tool.

8.1.1 THE WEB

Many people use or have heard of the World Wide Web (WWW), but few can precisely describe what it is. The Web is a global collection of documents that can be accessed via the Internet. Although some tend to use both terms almost as synonyms,

---

1 See chapter 7, pp.177-178.
2 See chapter 7, p.170.
3 See chapter 7, p.179.
in fact the WWW represents only a part of the Internet. The Internet consists of a worldwide network of mutually connected computers that supports communication across the globe. Each of these computers has a unique Internet Protocol (IP) address by which it can be identified in an unambiguous way. Like other computer networks, the Internet has various software programmes running on it, such as e-mail, newsgroups and – the focus of this section – the WWW.

In order to understand the exchange of Web documents across the Internet, a distinction should be made between servers and clients. As suggested by their names, the former handle requests from the latter. Typically the client software requests documents from the server software, which sends the requested files back to the client. Examples of Web server software are Microsoft Internet Information Server (IIS), iPlanet Web Server or Apache. Widely used Web clients, better known as Web browsers, are Netscape Communicator or Microsoft Internet Explorer.

In order to enable Web browsers to access documents on the WWW, each Web document has a unique address, the Uniform Resource Locator (URL). This address, for instance http://www.someserver.someplace/somedirectory/somefile.ext, consists of three components: the protocol, the name of the server machine and the name of the document itself.

The most common protocol used for exchanging Web documents over the Internet is the HyperText Transfer Protocol (HTTP), which allows displaying HyperText and images in a graphical environment. Hyper is used here as the opposite of linear. In general, text documents are (to be) read in a linear fashion: this sentence before that, this paragraph before that, this page before that, and so on. HyperText does not hold to this pattern and enables the person viewing a Web page to go anywhere, anytime he wants. Documents in HyperText are namely equipped with links to (parts of) other documents, so that readers can easily switch – ‘browse’ or ‘surf’ – from one document to another [Web_HTML, 2000].

After the protocol, the URL specifies the name of the server machine that stores the document at stake. This name can be either the machine’s IP address or its domain name, www.someserver.someplace in the example above. A domain name is a meaningful and easy-to-remember ‘handle’ for an Internet address. Internet domain names are located and translated into IP addresses by the domain name system (DNS). Because maintaining a central list of domain name/IP address mappings on one server would be impractical, sub-lists are distributed across multiple DNS servers throughout the Internet in a hierarchy of authority [Web_Whatis, 2000].

The third part of the URL contains the name of the document being requested – somefile.ext – often preceded by one or more subdirectories on the server machine. In the example above, the document somefile.ext is stored in the subdirectory somewhersubdirectory on the server www.someserver.someplace. The extension .ext indicates the type or format of the document.

4 Usually the server runs on a different machine than the client, although this is not always the case.
5 Correspondingly, the URL above starts with http://. Other protocols supported by Web browsers are, for instance, ftp:// (file transfer protocol), gopher:// or news://.
What types of documents does the Web support? Most of the documents on the Web are text files published in HyperText Markup Language (HTML). HTML is actually a subset of the Standard Generalised Markup Language (SGML), the standard for how to create a document structure. HTML files contain codes, so-called tags, which lay down the layout of the document and specify how the file should look when displayed by a Web browser. The tag `<P>`, for example, warns the browser to start a new paragraph. Because HTML pages can incorporate text, but also graphic images, sound and video, they may offer Web users a rich multimedia experience. Apart from text files, whether or not written in HTML, most standard Web browsers are able to display GIF and JPEG images. Other kinds of files – executable programmes, word processor documents, spreadsheets or CAD models, for instance – may be exchanged as well, but are rarely supported by Web browsers. When receiving such file, clients can either use a plug-in or helper application to display the file – a word processor or a CAD package, for example – or simply save the file to their hard drive.

Using Web technology to build a working prototype of DYNAMO has several advantages [Segers, 1998 p.23]. DYNAMO, as we have pointed out, should be able to provide users with a case representation that is as varied and as rich as possible. A first advantage of Web technology is that it supports virtually any format of (digital) information, ranging from text and images over video and sound to CAD and even VR models. Web browsers have evolved in such a way that they can display most of these formats, and if not, the Web still allows to exchange the information. Furthermore, by using HyperText, information about one case can easily be linked to information about other cases.

Secondly, the Web makes DYNAMO accessible to various users spread around the world. If we store the case base on a server machine, any architect or architectural student who has access to the Internet and disposes of a standard Web browser, can consult the tool. Moreover, although DYNAMO provides users with a large amount of information, this information does not occupy any disk space on their personal computer, since it is stored once on a central server machine.

An additional advantage of adopting a Web-based approach is that today many people, and especially students, have grown accustomed to using the Web. This means that they are already familiar with the interface of the programme, and thus will probably learn to use the tool without much effort.

8.1.2 DYNAMIC WEB DOCUMENTS

Most documents on the Web, as we have mentioned, are written in HTML, a language for specifying the layout of Web pages. Yet, if we want to build a convincing prototype of DYNAMO, simply filling its case base with HTML files will not do. Indeed, in order for the tool to be dynamic, users must be able not only to consult the case base, but also to manipulate its content and structure. HTML pages, however, are

---

6 Another subset of SGML is Extensible Markup Language (XML). Whereas HTML describes the content of a Web page in terms of how it must be displayed and interacted with, XML describes this content in terms of what data are being described. It is expected that HTML and XML will be used together in most Web applications in the future.
A WORKING PROTOTYPE

static in nature: they do not change unless someone manually edits the file and alters the content. In other words, each time the content or structure of the case base changes, every single HTML file containing information that is affected by this change needs editing.

A way out for this problem is offered by the Common Gateway Interface (CGI), a gateway that connects Web servers to so-called CGI scripts [Craig, 1996; Herrmann, 1996; referred to in Segers, 1998]. In essence, a CGI script is a software programme that communicates with Web documents. Via the gateway, such scripts can receive input from and send output to Web Browsers, which allows to create much more dynamic and interactive Web pages than HTML as such.

Just like HTML pages, CGI scripts are called up by means of an URL, for instance http://www.someserver.someplace/scripts/somescript?param1=value1&param2=value2. The last part of this URL, ?param1=value1&param2=value2, allows to send user input, in the form of one or more parameters, from the client to the server. Parameters are separated by a & symbol. The server transmits the request to the CGI script, which processes the user input and sends the result back to the client.7

CGI scripts can be written in almost any programming language. Languages that are often used for scripting are C, C++, Perl or VisualBasic, to name only a few.8 For our prototype, we have chosen to write scripts in PHP, a server-side, cross-platform, HTML-embedded scripting language [Web_PHP, 2000].9 Our choice was motivated by the fact that, unlike most other scripting languages, PHP was developed specifically for the task of programming for the Web, which allows to create much more powerful Web applications. Much of its syntax is borrowed from C, Java and Perl, supplemented with a couple of unique PHP-specific features.

A very interesting feature of PHP scripts is their fast access to commercial databases – databases created in Oracle® or Microsoft® Access, for example. In DYNAMO, we have opted to structure and organise memory by means of an Access database.10 Access is a fully-fledged, comprehensive 32-bits Relational Database Management System (DBMS) to make database applications in a Windows environment [Jennings, 1997; referred to in Segers, 1998]. A DBMS is a collection of programmes that enables users to create and use databases. By means of an Open Database Connectivity (ODBC) driver, Access databases can communicate with PHP scripts. ODBC provides a collection of functions that allow scripts to query and modify databases as requested by clients without having to start up Access.

In summary, PHP scripts seem to be cut out to put dynamics and user interaction into our DYNAMO prototype, as they can (a) receive input from DYNAMO users via CGI, (b) based on this input, either look up information in or manipulate DYNAMO’s

7 Usually, the result is sent first to the server, which in turn transmits it to the client. Some CGI scripts, however, send the result directly to the client.
8 Perl stands for Practical Extraction and Reporting Language. VisualBasic is the stepchild of BASIC, which stands for Beginners All-purpose Symbolic Instruction Code.
9 According to [Web_PHP, 2000], the official meaning of the acronym PHP is HyperText Preprocessor. Being HTML-embedded means that PHP scripts can be written simply between the tags <<?php and ?> in the HTML code.
10 See also section 8.2.2 pp.187 ff.
database via ODBC, and finally (c) send the result of this operation via the server back to the user.

8.1.3 USER INPUT

In addition to PHP, a second scripting language we have used to build our prototype is JavaScript. JavaScript is a simplified version of the more general and more powerful object-oriented programming language Java. Compared to this language, JavaScript definitely offers fewer opportunities. Yet, unlike Java, it has the advantage of being designed specifically to integrate software programmes within Web pages. These programmes can be written simply between the tags <script> and </script> in the HTML code.

Although, at first sight, JavaScript shows striking similarities with PHP, it should be noted that there is an important difference between both. PHP code is read and executed at the server side and remains completely opaque to clients. JavaScript code, on the other hand, can be read and executed by a Web browser, i.e. at the client side [Kentie, 1996 p.275; referred to in Segers, 1998].

Yet, why should DYNAMO want to execute programmes at the client side? PHP scripts, as we have pointed out, are able to receive input from DYNAMO users, and use this input to query or manipulate the database. User input can be obtained in several different ways, for instance by asking the person who views a Web page to press a button, tick an option in a check box or fill in a form. Equipping DYNAMO’s Web pages with JavaScript allows us to check this user input before it is sent to the server and, in case of incorrect or incomplete information, immediately prompt the user to adjust it. Obviously, it is much more efficient and timesaving to perform this check at the client side, as it saves users from having to wait until the connection with the server is established. Only when the user input is complete, the information is transmitted to the server side and processed by the PHP script. Taken together, PHP and JavaScript thus enable us to meet the demand for dynamics in our tool at both server and client side.

8.2 A working prototype

Based on the theoretical ideas outlined in the previous chapter, a first prototype of DYNAMO has been built [Segers, 1998], and has subsequently been further developed, extended and refined [Heylighen e.a., 1998a, 1998b; Heylighen & Neuckermans, 2000b]. In essence, the prototype consists of three major ingredients: 1. a collection of design cases – the actual memory content of DYNAMO; 2. an underlying database that organises and structures this memory; and 3. a user interface allowing to both consult and modify memory.

8.2.1 DESIGN CASES

Cases in DYNAMO are entire building designs, both built and unrealised projects. With an eye to a wide applicability, extension to include other levels of the conceptual
A WORKING PROTOTYPE

framework for CAAD – both upwards to urban design and downwards to building elements – are within the bounds of possibilities.\textsuperscript{11} For each project, all information available is collected in a folder bearing the name of the case. Case folders are in turn collected in a directory named after the architect (see figure 8.1). When the user selects a case, all related information available in the case base is displayed on a dynamically constructed Web page with links to the corresponding files. Using Web technology enables us to combine traditional media, such as text, photos (scanned or, more often these days, taken with a digital camera) and graphics (sketches, plans, sections, elevations), with 3D models, computer animation, video (e.g. a visit to the building) and sound (e.g. an interview with the architect).

The advantages of this combination are manifold. Compared to written data, the visual and spatial representation of architecture better fits the architect’s way of thinking. Essentially, a design comes into being through manipulation of non-verbal information: the designer knows, thinks and works in a visual way [Cross, 1982]. Furthermore, mixing multiple modes of representation provides DYNAMO users with a richer learning experience [Vora & Helander, 1997], which results in a better understanding of the design. Using HyperText, however, does not only allow for a rich combination of media, it has the advantage of giving users control over navigation and the ability to choose information interactively [Mishima & Szalapaj, 1999]. In addition, it allows linking the tool to external information resources on the WWW – an architect’s homepage, for instance, or a Web cam on the site – which considerably enlarges the scope of its content.

8.2.2 MEMORY ORGANISATION

The underlying database that structures and manages all information about DYNAMO’s cases was created with Microsoft\textregistered Access (see figure 8.2). The core of the database is the cases table. In this table, each case is characterised by a set of indices, i.e. features that can serve as filter criteria during retrieval and/or as links to

\textsuperscript{11} See also chapter 7, pp.171 ff.
other cases. Three of these features are obligatory and must be specified for each project in the system – not so much because they are extremely important characteristics, but because their combination serves as an ID that identifies each case in an objective, unambiguous way. The remaining indices are optional and only filled in when the feature is considered relevant for the case in question. In addition to the cases table, the database contains a (growing) number of architect tables. As suggested by the name, their task is to group cases by architect and inventory for each project what files are currently available in or linked to the system. In this way, the structure of the database reflects the file hierarchy on the server.

8.2.2.1 ID
One of the key objectives of DYNAMO is to develop a case base that changes dynamically as it is used. The need for dynamics, as we have pointed out, does not imply that there should be no indexing system at all, but rather that indices should be flexible and amenable to change. It is obvious though that if all indices could be changed at will, the case base would soon become a chaos and eventually unusable. Hence the idea to provide each case with an ID, a combination of three indices that remain unchanged and thus can serve as a backbone for the case base [Segers, 1998]. Unlike other indices, the ID triad must be specified for every single case. Hence the importance of choosing features that are obvious and unambiguous for all projects. Furthermore, features should be selected so that each project has a different ID. Based on these criteria the following indices have been selected: the architect, the name of the building or its owner, and the person who entered the project into the case base. In fact, for case identification combining the first two would already do. Yet, the latter has been added because it indicates whether or not cases are authentic and, to some extent, gives users an idea of the ‘authority’ of the information provided.

8.2.2.2 Optional indices
As already mentioned, the remaining indices are optional, which means that they are not necessarily specified for every project in the case base. In the first version of the prototype, optional indices included concept, period, location, surface and gradient of the site [Segers, 1998]. Throughout the successive versions, however, the indexing system has undergone considerable changes and extensions. Table 8.1 lists the set of indices paired with example values that are currently available in the database (do not try to read it; it is shown only to exemplify the web of indices and index values by which cases in DYNAMO are labelled and linked). In order to help users find their way in this (growing) web, we have opted to cluster optional indices in five categories: form & space, function, construction (the ‘corners’ of architecture’s triangle), context and concept.

---

12 See chapter 6, p.153.
13 The first version of the prototype was illustrated with private residences on a sloping site, which accounts for the index gradient [Segers, 1998].
14 See also chapter 1, p.9.
Given DYNAMO’s intention to support (student-)architects during conceptual design, some might argue that labelling cases by concepts would already do. In our opinion, however, it seemed advisable to index cases with other features as well. Concepts in architectural design do not appear out of the blue, but often have to do with a particular position with regard to the programme or context (the site, for instance, or the client).\textsuperscript{15} Moreover, these extra indices allow selecting cases by other than purely conceptual criteria. The period in which a project was built, for instance, may be indicative of the style or of the building technique used. All together, this web of provisional indices allows approaching cases from different perspectives and selecting them by various criteria.

\textsuperscript{15} See also chapter 4, pp.104 ff.
<table>
<thead>
<tr>
<th>category</th>
<th>index</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>concept</td>
<td>metaphor / analogy</td>
<td>bollard, citadel, collage, crab shell, cross, flash of lightning, glass bulb, glass house, glass plate, mirror, ocean liner, park, ramp, shed around construction site, ship, spiral, strange attractor, trains, tree house, umbrella, waterfall</td>
</tr>
<tr>
<td>relation old-new</td>
<td>(in case of extension or conversion)</td>
<td>continuation, contrast, dialogue, encapsulation, historical transparency, juxtaposition, reflection, respect, structural alteration</td>
</tr>
<tr>
<td>form&amp;space</td>
<td>formal articulation</td>
<td>axial symmetry, colour, cone, cube, curved, parabolic, pyramid, rhythm, symmetry</td>
</tr>
<tr>
<td></td>
<td>spatial configuration</td>
<td>grid, inner courtyard, mountain, patio, roof garden, spiral, underground</td>
</tr>
<tr>
<td>function</td>
<td>programme</td>
<td>airport, art gallery, auditoria, cafeteria, church, concert hall, conference centre, exhibition pavilion, footbridge, housing, library, museum, observatory, offices, private residence, public records office, school, sea terminal, shop, theatre, university restaurant</td>
</tr>
<tr>
<td>history</td>
<td></td>
<td>competition design, conversion, extension, new built, renovation, restoration, temporary</td>
</tr>
<tr>
<td>original programme</td>
<td>(in case of reuse)</td>
<td>art gallery, boxing ring, castle, church, cinema complex, citadel, college, confection atelier, dairy, factory, hangar, leisure centre, monastery, museum, palazzo, printing office, private residence, railway station, school, skating rink, warehouse, woollen mill, …</td>
</tr>
<tr>
<td>construction</td>
<td>construction type</td>
<td>cantilever, massive, skeleton, slab</td>
</tr>
<tr>
<td></td>
<td>construction material</td>
<td>brick, concrete, glass, steel, wood</td>
</tr>
<tr>
<td></td>
<td>façade</td>
<td>brick, concrete, copper, glass, granite, plaster, terracotta, timber, titanium</td>
</tr>
<tr>
<td>context</td>
<td>period</td>
<td>1915 … 1999</td>
</tr>
<tr>
<td></td>
<td>site</td>
<td>campus, hillside, industrial, island, lakeside, park, riverside, rural, seaside, sloping, suburban, urban, water</td>
</tr>
<tr>
<td></td>
<td>climate</td>
<td>temperate, Mediterranean, continental, …</td>
</tr>
<tr>
<td></td>
<td>location</td>
<td>Austria, Belgium, Denmark, Finland, France, Germany, Italy, Japan, Mexico, Portugal, Spain, The Netherlands, USA</td>
</tr>
<tr>
<td>ID</td>
<td>architect</td>
<td>Alvar Aalto, Luis Barragán, Christin Conix, Hilde Daem, Sverre Fehn, Frank Gehry, Steven Holl, Toyo Ito, Louis Kahn, Le Corbusier, Mecanoo, Willem-Jan Neutelings, Stefaan Onraet, Dominique Perrault, Gerrit Rietveld, Carlo Scarpa, Bernard Tschumi, Jørn Utzon, Paul Van Aerschot, Frank Lloyd Wright, Peter Zumthor, …</td>
</tr>
<tr>
<td></td>
<td>user</td>
<td>(name of the person who entered the case)</td>
</tr>
</tbody>
</table>

**TABLE 8.1.** Overview of the indices that are currently available in DYNAMO. Indices are grouped per category and paired with a (non-exhaustive) list of example values.

We deliberately use the term provisional because as soon as the tool is in use, users may start indexing cases by other features than initially available in the prototype. So far, the categories and indices provided roughly cover the issues that, in our opinion, are usually at stake when conceiving architecture. Yet, despite many shared interests among architects in general, there is considerable variation across individuals in how they go about designing [Tweed, 1998]. By actively involving users, i.e. (student-) architects in the midst of conceptual design, in the indexing process, the tool can be made more in tune with their specific interests and needs. For example, one could
imagine an architect who loves music and would like to categorise designs according to famous composers or different music styles.\textsuperscript{16}

8.2.3 INTERFACE

Having discussed what DYNAMO’s cases consist of, and how the underlying database labels and links them by various indices, this section turns to the interface required to access these cases. As illustrated in figure 8.3, this interface is composed of different modules. The user interacts with the tool through a standard Web browser, for instance Netscape\textsuperscript{\textregistered} Communicator or Microsoft\textsuperscript{\textregistered} Internet Explorer.\textsuperscript{17} Cases can be consulted by specifying selection criteria on a search page. The query may be very simple, for example, a specific building programme. Yet, it is also possible to enter more complex queries to find cases that match two or more criteria, say conversion of warehouses in a suburban context during the 1990s. Upon submission, the query is handled by a PHP script at the server side, which bridges the gap between the browser and DYNAMO’s database. By means of ODBC, the script can search the cases table of the database for cases that match the query without having to start up Access. When the search is completed, the script transmits the catch to the browser, which displays the result as a list of cases.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{module_diagram}
\caption{Suite of modules to access DYNAMO.}
\end{figure}

\textsuperscript{16} See also chapter 4, p.105.
\textsuperscript{17} Ultimately, the tool is to become accessible from within a 3D modeller, which is currently being developed within the context of the IDEA+ project (see chapter 7 p.173).
Figure 8.4. Search result (right) and summary of the information available on Tadao Ando’s Koshino House in Hyogo (left).

In figure 8.4, for example, the rightmost frame of the browser shows the result of a search for buildings on a sloping site. When clicking on a specific project in the list, a (dynamically constructed) summary of all information about this project appears in the bigger left-hand frame. In this screen shot, it features the summary of the Koshino House designed by Tadao Ando. At present, the information on this project available in the tool comprises a 3D CAD-model, drawings (an axonometric view, several plans and a cross section), photographs, a text excerpt from a book, links to related Web sources and finally a short list of bibliographic references, which can serve as a kick-start for a more detailed study of the case. Indeed, DYNAMO does not pretend to provide a substitute for existing information sources on design projects. A system like DYNAMO should not, and pragmatically cannot, encompass all information on a particular project. Information exists in a continuum of sources and media, and it would be counter-productive to extend any one source or medium to take the place of all the others. Moreover, it is not possible to foresee all types of information that may contribute to a better understanding of a design, as this varies from one project to another. Rather than absorbing all existing information on design projects, DYNAMO wants to make these projects accessible in a – for designers – cognitively comfortable way.
As will be illustrated in the following section, the user interface allows to access information about projects, but also to feed the case base – with new cases and/or indices – via the browser without having to switch to another programme. One important aspect of the system yet to be discussed is therefore the accuracy of the information provided. Despite DYNAMO’s measures to avoid incorrect user input like spelling errors or indistinct index names,\(^{18}\) it is obvious that the system is not fully foolproof. The nature of its cases and task precludes the system from being able to check newly entered information itself. To understand design cases, it would need natural language understanding, image recognition as well as notions of architectural history, construction and even geography. Ideally, it would have studied architecture and, if possible, some experience in architectural practice. Since DYNAMO does not dispose of these qualifications, at least some facility for manually maintaining the case base should be provided.

![Check-up cases](image)

*Figure 8.5. List of DYNAMO’s latest acquisitions viewed by an administrator.*

With an eye on this maintenance, the prototype is equipped with a second interface for users with special access permission, system administrators if you will. These permissions may be given to the titular of a design course, for instance, or to the head of a design office. Just like DYNAMO users, an administrator interacts with the case base through a simple interface that allows expressing himself through a combination of mouse clicks and typing. Figure 8.5 shows the screen he is first presented with after logging on. It lists the latest acquisitions of the case base, more specifically the cases that have recently been submitted or supplemented and have not yet been checked for

---

\(^{18}\) See the user scenario in section 8.3.
CHAPTER 8

correctness. 19 Clicking on the information button causes to appear a more detailed report of the files and index values submitted, which the administrator may alter in case of inaccuracies. This should enable him to maintain DYNAMO’s case base without any knowledge of HTML, PHP or even Access.

Figure 8.6.

8.3 DYNAMO in action

In the previous section, we have described the different ingredients of DYNAMO’s prototype – the (growing) collection of cases, the underlying database and the interface. The best way to introduce the prototype, however, is to show it in operation. Since DYNAMO is in a constant state of flux, it is highly recommended to investigate the tool on-line. By way of foretaste, this section meanwhile presents an annotated scenario of the system in use.

The scenario describes how a DYNAMO user perceives and interacts with the tool. To place yourself in the proper context, imagine yourself in a design office or studio, as a (student-)architect involved in the task of designing say a ceramic museum on an urban site. You have just read the brief and you are looking for ideas to get started.

19 The control buttons along the bottom of the screen allow to display a similar list of newly created indices.
A WORKING PROTOTYPE

You start up DYNAMO by typing its URL in a standard Web browser. Figure 8.6 shows the first screen you see after logging on. Along the bottom edge of the screen, user control buttons allow to exert control over your interactions at any moment, by switching from the search to the feed mode (or vice versa), asking for help, quitting the programme or starting over with a new search.

![Figure 8.7.](image)

8.3.1 SEARCHING THE CASE BASE

When opting for the search mode, as you apparently did in figure 8.6, you are presented with six clusters of search criteria: five categories of optional indices plus the ID. Clicking on the name of a category causes to appear the criteria it comprises (figure 8.7). You may request cases that show one specific feature, or cases that meet a combination of criteria from one or more categories. In order to do so, you simply fill out the index field(s) of interest. Alternatively, you may press the arrow next to the field(s), in order to get a list of index values that are currently available in the case base. In the screen shot presented, you have selected the category function, pressed the arrow of the index current programme, and selected the index value museum from the list.

---

20 In the current version of DYNAMO, access is controlled by a system of user names and passwords.
Figures 8.8 & 8.9.
Upon submission, DYNAMO displays a list of cases that meet the criteria you specified. The rightmost frame in figure 8.8 gives an overview of all projects in DYNAMO with the functional programme museum.\(^{21}\) Projects marked with an asterisk (*) are newly entered, which means that they have not yet been examined for correctness and completeness. In order to have a closer look at a particular case, you can click on the corresponding name or thumbnail picture, after which more detailed information on the project appears in the frames on the left.

Judging from the dark grey top frame, you want more information on the extension of the Musée des Beaux Arts in Lille designed by Ibos & Vitart. In addition to the name of the building and architect(s), this frame displays the location of the project – both city and country – as well as the period in which it was designed and/or built.

The white frame in the middle displays a photo of the selected case together with an overview of what information is currently available on the project. At present, the case content of the Musée des Beaux Arts comprises several drawings (plans and sections), photographs (both interior and exterior views), a bibliography and two articles from architectural magazines. Each piece of information can be viewed by simply clicking on the corresponding link. When selecting the link plans, for instance, you first get a preview of the different plans available – a plan of the site, existing situation, ground floor and basement (figure 8.9). One additional mouse click allows to view the plan of your choice in more detail.

Returning to the screen of figure 8.8, the frame on the left lists the characteristics of the project at stake. Additional characteristics can be specified by pressing the add info button and completing one or more fields in the dialogue box that pops up (figure 8.10). If, after having viewed the information available, you want to share your view on the project with other DYNAMO users, the discussion button leads to a dialogue box for writing and submitting an on-line comment.

Once you are done with the Musée des Beaux Arts, you have the opportunity to browse to other cases that have features in common with the project at issue. When you click on one of the characteristics in the left-hand frame – for instance, the spatial configuration underground – the rightmost frame displays a new list of projects that share the selected characteristic (figure 8.11). Alternatively, you may start a new search or switch to the feed mode by using the control buttons at the bottom of the screen.

### 8.3.2 FEEDING THE CASE BASE

Apart from specifying extra characteristics and writing on-line comments, there are two ways in which you can feed the case base: either by submitting information on a case or by creating a new (kind of) index (figure 8.12).

\(^{21}\) If you had specified multiple criteria, the system would have looked up first cases that show all features specified, and subsequently cases showing at least one feature of interest. With only a few cases available in early versions of the system, the latter was done to save DYNAMO from coming empty-handed all too often. As the case base starts to become more populated, however, the selection mechanism could be restricted so as to retrieve only cases that meet all criteria specified.
CHAPTER 8

Figures 8.10 & 8.11.
Imagine you got quite curious about Ibos & Vitart’s museum, even to the extent that you decide to pay it a live visit. Once returned from your trip to Lille, you may want to share the photos you have taken with other DYNAMO users. When selecting the option case submission, a dialogue box requests you to fill in the case’s ID – architect, building and your name (figure 8.13). Pressing the arrows next to the former two fields calls up a list of architect and building names that are currently available in the system. This allows you to check whether the case is already present in the case base and helps avoiding spelling mistakes. Since the Musée des Beaux Arts is indeed mentioned in the list, you can complete the case’s ID by a simple mouse click. In order to add your photos to this case, the lower half of the screen asks you to select the file you want to submit, specify the type of the file (e.g. 3D model, text, drawing or, in this case, photo), give a concise description of the information it contains (view of the entrance hall, for instance) and identify the source of the information (see figure 8.14).\footnote{If this source is a document on the Web, you do not have to select the file, but instead fill in the \textit{URL} so that the system can create a link to the file.} As soon as these data are submitted, a new dialogue box pops up with the request to specify as many optional indices as possible (figure 8.15).
Figure 8.15.

Upon pressing the submit button, the project’s indices are updated in the database, more specifically in the cases table, while the photo is transmitted to the subdirectory ‘Musée_des_Beaux_Arts’ in the directory ‘Ibos_Jean-Marc’ on the server. The procedure to submit information on a new case is completely analogous, the only difference being that you have to type the ID features instead of selecting them from the list.

Apart from adding information on cases, however, you can also feed the case base by creating a new index. If you like colourful architecture, for instance, you may want to label and link projects by colour. In order to do so, DYNAMO asks you to specify the index name (in this case colour), type of index (verbal, numeric or multiple), and give a concise description of what the index name denotes (for instance, ‘main colour of the façade’) (figure 8.16). Because it is very important to choose a unique name for each index, the arrow next to this field once again allows to display a list of all index names that are already available in the system. As soon as you submit the required information, a new column is created in the cases table of the database, which allows, from now on, to label and select cases by their colour.

23 A multi-index is an index that consists of multiple components. The index location, for instance, comprises two components: country and city.
At present, DYNAMO is a working prototype that contains more than 80 indexed cases from 53 different architects. Cases are labelled and linked by 17 (kinds of) indices, which are clustered into six categories (concept, form&space, function, construction, context and ID). The relative simplicity of DYNAMO’s case representation, memory structure and interface was an intentional choice motivated by the priority of producing a functional system in a short period of time. In order to test this functional system, we have conducted a formal evaluation in a 4th year design studio. As will be pointed out in the following chapter, the main objective of this evaluation was to find out whether DYNAMO manages to engage design students into in-depth explorations of concrete cases. Yet, since this evaluation represented the first time the prototype received extensive use, it was also an occasion for bug identification and repair. Consequently, the system was modified considerably following this evaluation.

Whereas access to the first version was mainly a privilege of Netscape® users in Windows environments, the prototype has meanwhile made substantial progress in terms of compatibility. The most recent version has been tested successfully on several browsers, machines and platforms, including UNIX (SGI O2) and Apple (Apple PowerMac 7100 and 7200). Other major improvements have to do with the user
interface. The results of the evaluation, as will be pointed out, suggest that although students found the tool engaging, they were not engaged in ways that met with our original expectations. Therefore, we have tried to tailor the interface so as to make opportunities for active participation – writing on-line comments, adding extra features, new cases or additional indices – immediately strike the eye.

Despite these improvements, however, DYNAMO remains a prototype system. Due to the relative simplicity of its memory organisation, the programme speed may start to suffer as the case base becomes more populated. Furthermore, inaccuracies remain in the interface for maintaining the case base. Because this side of DYNAMO has received virtually no use from the design teachers involved in the evaluation, improvement of the prototype following this evaluation has mainly concentrated on the user side. The interface to check and correct information is therefore not yet fully developed. However, with additional effort, it could be polished to account for issues it is currently unable to deal with effectively.

Awaiting these future improvements, let us close this chapter by situating our prototype with regard to other digital archives, educational design tools and CBD systems. LORA, the Library Of Recommended Architecture, is an on-line library for exemplary 20th century architecture, developed at the University of Art and Industrial Design in Linz (Austria) [Web_LORA, 2000]. Architects, students and journalists are polled for their favourite projects, which are listed either according to architects or to the persons who recommended them. In addition, projects can be searched by type, year of construction, city and country. The German, more elaborate counterpart of LORA is archINFORM, a digital archive including over 8000 built and unrealised projects from various architects and planners [Web_archINFORM, 2000]. Once again, the main focus is on architecture of the 20th century. Projects can be selected by architect, town or keyword. At first sight, both systems show significant similarities with DYNAMO: they are Web-based and on-line accessible, and they collect information on entire building designs. Like many catalogues, however, their indexing system seems to serve historians and critics better than architects. Whereas the former regard an existing design as a subject to study, the latter perceive it as a source of images and concepts that can drive their own design process. By actively involving the users, i.e. (student-)architects in the indexing process, the collection could be made more in tune with their specific needs during design. Unfortunately, however, there is no possibility to index projects interactively, or – in case of archINFORM – even to submit new cases. Furthermore, unlike DYNAMO, neither of both systems intends to become part of an integrated design environment.

24 See chapter 9, p.213.
26 According to the archINFORM website, a special form for registering new projects is still under construction. Users who want to add a project are asked to send an e-mail with attached image-files or use snail-mail. A prerequisite for registering new projects is that they have been published in an architectural magazine or an architectural book [Web_archINFORM, 2000].
Among the group of design tools specifically intended for education, a very close relative of DY\textsuperscript{N}AMO is EDAT, an Electronic Design Assistant Tool developed at Carnegie Mellon University in Pittsburgh (PA) [Akin e.a., 1997]. Similarities are striking and include the tool's objectives and underlying pedagogical ideas as well as some aspects of its implementation. EDAT is conceived as a centralised store for information gathered by student-designers in the early stages of the design process. Just like DY\textsuperscript{N}AMO, the tool forces students to represent and index design information in a clear and comprehensible way. The work on EDAT puts us in a hopeful mood about the value of our approach. Since the project is terminated, its developers were in fact quite pleased to see their approach carried on in our research.

Among the members of the CBD family discussed in chapter 3, perhaps the most distant relative of DY\textsuperscript{N}AMO is CADRE.\textsuperscript{27} As we have pointed out in the case study, CADRE aims at (partially) replacing human architects, whereas DY\textsuperscript{N}AMO wants to support them during design. Another difference is the view on architecture underlying both systems. Unlike CADRE’s reduction of architecture to pure geometry, DY\textsuperscript{N}AMO makes an attempt to grasp the full complexity of architectural design.

At first sight, the same applies to IDIOM, as this system is rooted in the CADRE project.\textsuperscript{28} However, in spite of its focus on geometric and topologic aspects of design, IDIOM considerably differs from its predecessor. Rather than automate the configuration task, the system supports designers as they compose designs themselves. In addition, social and political aspects of architectural design are taken into account by means of preferences – at least, that is the intention of the system’s developers. Since the exact determination of geometric parameters is not immediately at stake during conceptual design, we view IDIOM and DY\textsuperscript{N}AMO as complementary.

CBD systems that do share DY\textsuperscript{N}AMO’s concern with supporting architects during the conceptual stage of the design process are Archie-II and PRECEDENTS.\textsuperscript{29} The former stores floor plans of public buildings, supplemented with stories from different stakeholders about how the design turned out. These stories should draw the architect’s attention to all life cycle implications of his decisions early in the design process. Organising several stories around a case, however, implies a highly complex indexing system, which makes dynamic case collection or (re-)indexing extremely difficult, if not impossible. To allow for such dynamics is one of the major ambitions of our tool.

Also in PRECEDENTS, the sophisticated memory organisation seriously complicates the further development of the case base in a dynamic way. PRECEDENTS, like DY\textsuperscript{N}AMO, attempts to elicit concept knowledge that is implicitly embedded in design projects. In particular, it provides student-architects with meaningful concepts from memorable museum designs. Following the Issue-Concept-Form (ICF) formalism, cases are organised in a tripartite network, providing for a rich case representation that serves designers rather than theoreticians. Compared to this formalism, DY\textsuperscript{N}AMO’s memory organisation is downright simple, yet it has

\textsuperscript{27} See case study 3.2 (chapter 3, pp.66 ff).

\textsuperscript{28} See case study 3.4 (chapter 3, pp.78 ff).

\textsuperscript{29} See case studies 3.1 and 3.5 respectively (chapter 3, pp.61 ff and pp.83 ff).
the advantage of being much more flexible and amenable to dynamic change. A second difference between both systems lies in the origin of the concepts they provide. In PRECEDENTS, the system developers have identified a concept vocabulary by analysing architectural literature. In DYNAMO, on the other hand, concepts are interactively identified and refined by the users, i.e. by (student-)architects actively involved in designing.

A final system we would like to mention is WEBPAD, a relatively young member of the CBD family. So far, at least two parallels with DYNAMO can be drawn: both systems share the ambition to exchange design cases across architects in different contexts, and to make these cases accessible from within a CAD environment. Given the early stage of WEBPAD’s development, however, it is difficult to pronounce upon how exactly both systems may relate to each other.

Considering the list of parallels and contrasts with other CBD tools presented here, one may suggest that DYNAMO itself can be considered a product of Case-Based Design. Indeed, to some extent our prototype results from trying to merge the strengths and avoid the weaknesses of the cases studied in chapter 3. Whether the resulting case is worth storing in the case base for future use is the subject of the following chapter.
DYNA MO IN THE STUDIO

Evaluation

“It seems a very interesting tool, but we simply don’t have time to use it.”

anonymous architecture student
So far, the second part of this thesis has outlined the theoretical ideas behind DYNAMO and discussed their implementation as a working prototype. This chapter will describe how this prototype was used and appreciated by 4th year architecture students. In Winter/Spring 1999, DYNAMO received its baptism of fire in a traditional design studio at our department. The theme of the studio being reuse, students were asked to design either a public library within a former industrial hall or an extension to their school, which is located in a 16th century castle. During the entire project, all students had access to DYNAMO’s case base, which was for the occasion supplemented with extra reuse projects.

In order to provide some background to the evaluation of DYNAMO, section 9.1 starts with explaining why we opted for a reuse project to test our prototype. After sketching the setting and procedure of our study (section 9.2), the analyses to examine DYNAMO’s performance are reported on in sections 9.3, 9.4 and 9.5. Finally, the chapter ends by summarising the outcome of the evaluation and formulating conclusions.

9.1 Why reuse?

In order to test our prototype, we selected a design studio with the theme reuse. The choice was made on the basis of a number of criteria arising from the objectives of our research. With the development of DYNAMO we try to kill two birds with one stone: on the one hand we aim at bridging the gap between CBD and architectural design; on the other hand it is our ambition to contribute to valuable computer support for the practice and education of architectural design.

The first objective requires the design task to be CBR-appropriate, i.e. it has to 1. be a recurring task; 2. have a critical mass of similar cases; and 3. be a non-trivial task, meaning that the user feels the need for and values support.1

As to the second objective, it is important that the design task is relevant to architecture’s practice and education today. Developing a tool that supports the design of medieval fortresses might be quite exciting, but would perhaps not so much contribute to a more intelligent use of computers by contemporary (student-)architects.

9.1.1 CBR-APPROPRIATENESS

The reuse of existing buildings for new purposes seems to fulfil these requirements quite well. First of all, renovation and reuse projects represent a substantial and still growing part of today’s architectural practice. Both are increasingly viewed as a valuable alternative for new building, not only for economic reasons or to save space, but also because of the potential to preserve buildings with a certain architectural, historic, didactic or other value. The fact that Belgium (and Europe) is already more than built up, causes us to believe that the future will even reinforce this trend.

---

1 Characteristics of a CBR-appropriate task according to Betsy Cordingley, British Telecom Laboratories, at EWCBR’96 [Cordingley, 1996].
Also on the second criterion – the number of existing cases – reuse scores fairly well. The treasure of converted warehouses found in three Belgian cities (Antwerp, Brussels and Ghent) would already fill a rich case base [Mertens & Patteeuw, 1997], without even speaking of other types of converted buildings, or reuse projects in other cities and countries.

Thirdly, if architectural design is characterised by a high level of uncertainty, this certainly applies to designing a reuse project. A first source of uncertainty is the (physical) condition of the building, which often only shows its true colours during – or in the worst case, after – the execution of the project. In addition, nothing guarantees that the client’s requirements are compatible with the building he has set his mind on. Apart from an extra dose of uncertainty, reuse projects also labour under some practical difficulties, simply because the design must be executed inside an existing building. Last but not least, there is a whole cluster of architectural, urban, historic, political and social meanings that the building has (had) and which the architect must draw somehow into conversation. By consequence, there is every reason to consider the reuse of an existing building – be it a warehouse, a factory hall or another type of building – as a non-trivial design problem for which (student-) architects are likely to value support.

9.1.2 RELEVANCE FOR ARCHITECTURE

But what about the relevance to the practice and education of architectural design? In the 21st century, problem number one on the agenda of European architects will undoubtedly be the city. No longer fulfilling the requirements of today’s sophisticated industry, former industrial quarters are more and more abandoned, dooming substantial parts of the city to death. One remedy to cure the city from this cancer is to bestow 19th century industrial architecture with new functions, thus pumping new life into these quarters.

In this respect, DYNAMO may have an additional application in the making of feasibility studies. Many valuable pieces of industrial architecture are currently demolished, due to lack of time and information to examine what they have in store for future use. In order to facilitate such examinations, a tool that gives access to already realised reuse projects of similar buildings might be a very useful instrument.

It should be noted, however, that the reuse project reported on in this chapter is nothing more than a test case to evaluate DYNAMO’s prototype. With an eye on a wide applicability, both in practice and in education, the tool should be easily extendible to other kinds of design projects, and thus not tailored specifically for reuse projects.

---

2 See also chapter 7, p.174.
9.2 Setting and procedure

9.2.1 SUBJECTS

All 48 subjects in the study (12 female and 36 male) were student-architects of the 4th year Architecture, Urban Design and Planning at the Faculty of Engineering at the University of Leuven. Having had three years of design education, they participated in the reuse studio as it is an obligatory subject in the curriculum. Ten of them chose to work independently, the remaining 38 split off into two-person teams. The studio was led by Professor Paul Van Aerschot, the titular of the theoretical course on reuse, assisted by three studio teachers.

9.2.2 TASK/ASSIGNMENT

The students who participated in this design studio could choose one out of two design assignments: either the conversion of a former fabric hall into a public library, or a reorganisation of and extension to their school, which is located in the 16th century castle of Arenberg. Both assignments focus on dealing with an existing building, be it in the form of a reuse or extension project. Students are required to consider the building from multiple perspectives, such as formal articulation, spatial configuration, construction and materials used.

The site of the first assignment consists of the former ateliers of the Belgian Railway Company near Leuven. On this site, three factory halls are to be preserved and converted into a market hall, a public library and an indoor sports hall. Students were asked to design the library, and to investigate possible connections between the library and market hall.

The second assignment is to be situated in the Arenberg park, where one campus of our university is located. The castle, which acts as landmark in this park, houses the administration of the Faculty of Engineering as well as the Department of Architecture. The task was to reorganise and optimise the West wing of the castle (design studios, lecture rooms, secretary, photocopy room) and expand it with a reception hall, material museum and exhibition room. 3

Five solo designers and 13 two-person teams chose the library project, the others – 5 singles and 6 pairs – opted for the reorganisation of and extension to the castle.

9.2.3 PROCEDURE

Before the start of the assignment, DYNAMO was filled with information (primarily plans, sections and photographs) on several significant reuse projects and libraries, some of which were suggested by the studio teachers. Originally, we hoped that studio teachers would actively participate in loading the case base with relevant cases, yet simply getting them to make a list of projects already required quite an effort. Instead of the teachers, we collected information on these projects and added this information

3 A detailed description of both assignments can be found in appendix B.
DYNA MO IN THE STUDIO

to DYNA MO’s case base. At the start of the assignment, students were given a demonstration of how the prototype worked. All students participating in the studio were allowed to use the tool for as long and as often as they liked. By assigning each user – student or studio teacher – a user name and password, we could keep track of who logged on to the system and when. The system could be accessed not only from the computer class at school, but from every PC connected to the Internet. Students were able to consult the projects provided, write on-line comments, as well as add other cases they found relevant to the assignment. A DYNA MO help desk was set up to help students with using the tool, answer questions and respond to technical problems.

Aside from having access to DYNA MO, the design project was conducted as usual. Students met in the studio roughly twenty hours per week, spending much of this time either working alone or in pairs, or discussing their project with the studio teachers.

After the conceptual phase of the assignment, students were given a questionnaire in order to examine whether DYNA MO succeeded in engaging students.4 With an eye to future improvements, the questionnaire also asked how students liked several aspects of the tool, such as the interface, choice of cases or selection criteria. Some questions dealt with elements that may influence student engagement, such as whether or not students have a PC at home, like surfing on the Internet, or used CAD software to model their project. From the 48 students who participated in the studio, only 19 filled in the questionnaire. For this reason, the results in sections 9.3 and 9.4 are based on a sample of 19 students only.

9.2.4 JUDGMENT

Upon completion of the assignment, students presented their project to a jury consisting of all design teachers involved in the studio. The jury gave each project a global score as customary in other design assignments. Afterwards, two critics – architects with professional expertise related to reuse projects – were invited to evaluate the students’ design projects. Although both architects have ample experience with design teaching, they were not otherwise involved in this studio, and were simply asked to assess the resulting projects. Apart from a global score reflecting their overall appreciation of the project, they had to rate each project on six five-point scales each measuring a specific feature of the design. These features were selected because of their particular relevance for the experiment, the assignment or the design field, i.e. architecture. They included the quality of the concept (because of the assumption that cases contain concept knowledge and DYNA MO’s intention to support conceptual design),5 the design’s attitude towards the existing building (the focus of the assignment), the formal articulation, functional organisation and constructive solution

---

4 See appendix C.
5 See chapter 1, p.15 and chapter 7, p.165.
CHAPTER 9

of the design (the ‘corners’ of architecture’s triangle), and the creative character of the design (because of possible effects of design fixation).

Based on the students’ replies to the questionnaires, the evaluation of their projects by the jury and the scores from the external critics, the following sections will attempt to answer three questions:

1. Did DYNAMO succeed in engaging students to explore the design cases provided?
2. What factors stimulated or hampered this engagement?
3. Did the use of DYNAMO have any impact on the quality of students' design projects?

9.3 Student engagement

From the 48 students who participated in the studio, 35 (73%) effectively made use of the system. Because the study described here represented DYNAMO’s baptism of fire, we consider this percentage a reasonable level of success, all the more since using the tool was far from being encouraged by the studio teachers. Unfortunately, the questionnaire, which was designed to examine whether students found the tool engaging, had much less success. As already mentioned, only 19 (40%) students filled in the questionnaire, which still represents a reasonable response given the strict time schedule of the assignment.

9.3.1 INFORMAL APPRECIATION

The question whether they wanted to use DYNAMO again in future design projects, was answered positively by all respondents except for two. According to one of both, “it seems a very interesting tool, but we simply don’t have time to use it.” Rather than from lack of time, her colleague seemed to suffer from fear for design fixation, as he comments: “The chance of being original diminishes as information increases. EVERYTHING HAS ALREADY BEEN DONE. There are just some things that I'm not yet aware of.”

When asked how they liked DYNAMO, it was evident from the answers that most respondents found the system engaging. The responses indicated that students were first of all attracted by the quality of the visual material and, to our surprise, by the tool’s inherently ‘ecological’ nature. According to one third of the respondents, DYNAMO would finally offer a solution to the waste of paper caused by making copies in the library. Other strong points frequently popping up in the responses are the colour interface, the variety and scope of information and, compared to the library,

---

6 See chapter 1, p.9.
7 See also the experiment in chapter 5.
8 To the question whether they felt encouraged by studio teachers to use DYNAMO, 85% of the respondents answered in the negative. Only 11% had the impression that teachers were informed about the content of the case base. Judging from the registered logons, two of the four teachers had a look at the tool, be it only once or twice.
the quick and easy way to find projects of interest. One, presumably extremely disorderly student especially appreciated the fact that, unlike paper copies, DYNAMO can never be lost.

The responses also indicated that, although students found the tool engaging, they were not necessarily engaged in ways that met with our expectations. In principle, we expected that students would consult cases in DYNAMO, but also actively participate in adding and commenting on projects. Yet, none of the students exploited the opportunities for active participation. One student seemed to find the idea of ‘information from students for students’ quite compelling, even to the extent that he devoted a great deal of attention to praising it in the questionnaire. However, this same student then showed little or no interest in putting this idea into practice. This goal was probably too ambitious given the very limited time schedule of the assignment.

 Asked about the drawbacks of DYNAMO, no less than 13 of the 19 respondents mentioned the obvious fact that you need a computer to consult the system. Despite serious efforts of our university, students do not always have easy access to computers, let alone to the Internet, and this may have prevented DYNAMO from functioning the way it was originally intended to. As one student put it: “If, during design, you just want to check something about a façade, a joining detail, a certain layout, then it’s much simpler to consult the paper copies lying next to you on your table, than to wait till the next day to bike through the rain to the computer class.” Yet, even for well equipped students, the medium by which DYNAMO makes cases available seemed to function as a brake: “It’s difficult to go through during spare moments, say a train journey.” However, the very fact that this student thinks of consulting the tool on the train suggests that he appreciates rather than questions its usefulness. Apart from the medium, respondents apparently had little to complain about. One student seemed disappointed by the number of cases in the system: “Once there are more projects, I’ll come and look again,” another complained about the fact that “you always have to press the submit button.”

Finally, it should be noted that, to our surprise, students’ responses were hardly affected by the prototype nature of the software. Most errors, such as JavaScript error messages, did not appear to disrupt students. However, they were occasionally frustrated by the slowness of the computers in the classroom, puzzled by platform-incompatibilities or confused by browser-dependencies. All in all, the DYNAMO helpdesk received five e-mails and one phone call.

9.3.2 FORMAL APPRECIATION

In addition to an informal appreciation, students were asked how they liked specific aspects of DYNAMO by rating them on a five-point scale. Students could choose from very poor, poor, neutral, good or very good. Their responses were coded from 1 to 5, with very poor corresponding to 1 and very good to 5. The frequency and mean scores of their responses are displayed in figures 9.1 to 9.3. Mean scores that differ significantly from a neutral 3 score in a paired $t$ test are indicated by an asterisk (*). 9

---

9 For a more detailed report of the pared $t$ tests, see tables 1 to 3 in appendix D.
Given DYNAMO’s ambition to assist conceptual design, the questionnaire asked to what extent students felt supported by the system in this stage of the design process. On average, student opinions about the support for exploration, concept generation and concept development fluctuate around a neutral 3 score: 3.17, 2.94 and 2.88 respectively (figure 9.1). None of the means are significantly different from 3 in a paired t test.

Obviously, the extent to which DYNAMO can support designers is closely related to the quality of the case base. As for any CBD tool, its effective performance largely depends on the richness, diversity and number of cases [Oxman & Shabo, 1999]. Asked about DYNAMO’s case base, students seemed to be enthusiastic about the choice of the cases and selection criteria, yet remained neutral as to the case content (figure 9.2). As indicated by the asterisk (*), the means for case choice and selection criteria – 3.56 and 3.61 respectively – are significantly higher than 3 (p< .05), whereas for case content – 3.28 – no significant difference is found. The positive rating for selection criteria chimes with the fact that students did not exploit the opportunity to create new indices. When asked why they did not, the majority of the respondents replied that DYNAMO’s criteria were OK. Only one respondent suggested to add a criterion, or rather to do away with multiple criteria. He would prefer to simply select cases by typing in a keyword – whether this word pertains to a project’s concept, context or whatever – and let the system search across different indices to find a matching case.

Figure 9.1. Frequency of student responses to the question whether they felt supported by DYNAMO during conceptual design. Mean scores are mentioned between brackets.
Finally, the questionnaire tried to find out whether the prototype was perceived as user-friendly. User-friendliness is a characteristic of every commercial software application, at least if we may believe their ads and salesmen. Although this characteristic is increasingly recognised as decisive for the success of an application, few people can describe what it actually consists of. Whether or not users perceive a programme as user-friendly turns out to depend on a cocktail of (sometimes subjective) factors [Froyen, 1999; Monden, 1999]. Therefore, instead of asking for a general appreciation of DYNAMO’s user-friendliness, the questionnaire solicited how students liked several ingredients of this cocktail: the interface, language used, programme speed, finding speed, ease of learning, ease of use and help contents. The interface of the prototype takes the form of a simple Web page, allowing users to interact by mouse clicks or typing.10 With an eye to using DYNAMO across architecture schools (and design offices) in different countries, the interface is written in English. The speed of the programme refers to how fast the system reacts to and processes user input, the finding speed to whether users can easily find the information they are looking for. An essential ingredient of user-friendliness is whether or not the programme is easy to learn, without intensive study or months of unproductive bungling [Richens, 1989]. Once the user has grown accustomed to the programme, it

---

10 See chapter 8, pp.191 ff.
CHAPTER 9

implementation

should be easy to use, without too much errors, serious (mental) effort or regular resort to the help button [Monden, 1999].

On average, students seemed either neutral or positive about DYNAMO’s user-friendliness (see figure 9.3). They liked the look and language of the interface, found the programme easy to use and were satisfied with the help contents. As indicated by the asterisk, the mean scores for these four aspects – 3.31, 3.65, 3.50 and 3.47 respectively – are significantly higher than 3 (p< .05). Programme speed, finding speed and ease of learning were judged neutral by the students, as no significant difference between the mean scores – 3.12, 2.83 and 3.38 – and the neutral 3 score was found.

In summary, there is significant evidence in students’ responses that they found DYNAMO engaging. Three quarters of the participants effectively made use of the tool, and nearly all respondents would like to use it again in the future. Serious questions arise, however, about the nature of this engagement, since none of the students has taken up the offer to go deeply into cases. When asked how they liked specific aspects of the prototype, students were either neutral or positive. In particular, the case content, ease of learning, programme and finding speed of DYNAMO received a neutral 3. Positive scores were given to the system’s choice of cases and selection criteria, look and language of the interface, ease of use and help contents.

Figure 9.3. Frequency of student responses to the question how well they liked specific aspects of DYNAMO’s user-friendliness. Mean scores are mentioned between brackets and marked with an asterisk (*) when significantly different from the neutral 3 score in a paired $t$ test (p< .05).
9.4 Stimulating and/or hampering factors

Although DYNAMO managed to engage 35 participants in the studio, there remains a group of 13 students who could not be made to use the tool. Moreover, between the 35 DYNAMO users, there were considerable differences in terms of frequency of use. Judging from the logons registered by the system, some students had a look at the tool only once, others were logged on repeatedly. Therefore, the second objective of our evaluation was to find out what factors stimulated or hampered student engagement. The hypothesis being explored was that students who had access to the tool at home, are more computer-literate or used CAD software to model their design, would deploy DYNAMO more easily than others.

In order to examine this hypothesis, the questionnaire asked about students’ personal computer equipment, how often they surf on the Internet, and whether they used a CAD package for this specific design project. Furthermore, we collected the scores students received for the CAAD course, which were used as a measure for their computer literacy. Subsequently, we calculated the correlations between, on the one hand, how often students made use of the system as recorded by the system and, on the other hand, the factors reflecting their computer equipment, familiarity with the Internet, use of CAD software and level of computer literacy.

The analyses of these correlations, however, yielded no results of statistical significance. The extent to which students made use of the prototype does not correlate significantly with any of the factors analysed. This seems to suggest that DYNAMO is fairly ‘democratic’, in the sense that it does not seem to privilege students with private access to or prior knowledge of computer technology. For the time being, the question why the tool did not manage to engage 13 of the 48 participants thus remains unanswered.

9.5 Impact on design quality

The third, most ambitious goal of our evaluation was to investigate whether the use of DYNAMO had any impact on students’ design performance.11 As pointed out in chapter 5, the extent to which students benefit from cases during design heavily depends, on the one hand, on their design skill and, on the other hand, on their motivation to design a high-quality project.12 One of the main objectives of DYNAMO was therefore to support all (student-)architects during design, in other words, to present cases in such a way so as to make up for differences in background and motivation. To get an idea of whether our prototype lived up to this ambition, the analyses to examine the third question, on students’ design performance, followed the lines of the first study as much as possible.

---

11 As in chapter 5, this performance was evaluated by assessing the quality of students’ design projects. For a critical comment on this way of evaluation, see chapter 5, p.131.
12 See chapter 5, pp.137 ff.
9.5.1 DEPENDENT AND INDEPENDENT VARIABLES

As already mentioned, the overall quality of students’ design projects was evaluated both by a jury of design teachers involved in the assignment and by two external critics. In addition, the latter rated several aspects of the design (concept, form, construction, etc.) on a five-point scale. In order to make evaluations comparable across the different judges, all scores were standardised. For each external critic, we also calculated the correlations among his/her aspect scores. Judging from his high correlations, one of both critics did hardly distinguish between the different aspects of students’ designs. A possible explanation may be that this critic is used to teaching design in the 2nd year, and therefore did not know exactly what to expect from 4th year students. In fact, he seemed quite disappointed about the (low) quality of their projects, which may have biased his evaluation. For this reason, the scores of this critic were not taken into account in the analyses.

In result, the total set of eight dependent variables\(^{13}\) consisted of global-score\(_i\) given by the jury, and of global-score\(_e\), concept-score\(_e\), reuse-score\(_e\), form-score\(_e\), function-score\(_e\), construction-score\(_e\) and creativity-score\(_e\) given by one external critic. For each variable, scores were split into two groups according to the different options of the assignment – on the one hand the scores for the library designs, on the other hand the scores for the school projects.

As to the independent variables, the major variable of interest was the extent to which students made use of DYNAMO. Unfortunately, we had little control over this variable since the system did not record how much time students spent using the tool. Therefore, we took the frequency of use, i.e. how often students logged on to the system, as a measure for the influence of using DYNAMO instead. By analogy with the analyses in chapter 5, two additional independent variables were selected for analysis. The students’ score for design in the 3rd year, referred to as design\(_3\), was taken to reflect their background and skill, while the fact whether students completed the questionnaire served as a measure for their motivation.

9.5.2 HYPOTHESES

The analysis attempted to test whether the use of DYNAMO during conceptual design affects the quality of students’ design products. Given the fact that cases may act as channels supplying concept knowledge on the one hand, and DYNAMO’s intention to support all (student-)designers, irrespective of their motivation or background on the other hand, the following hypotheses can be constructed:

Hypothesis 1: Based on CBR’s cognitive model, we expected the use of DYNAMO to have a positive effect on the quality of students’ design projects. Because cases are said to be particularly useful in the conceptual stage of the design process, students who made extensive use of the tool were expected to perform better than their colleagues, especially with respect to the concept of the projects.

\(^{13}\) Henceforth, variables will be typeset in italics.
Hypothesis 2: Based on DYNAMO’s efforts to make up for differences in students’ background and motivation, we expected that more skilled and motivated students would not be privileged to benefit from using the tool.

In terms of the variables of the experiment, these hypotheses can be rephrased as follows:

Hypothesis 1: The independent variable frequency of use was expected to contribute positively to the dependent variables reflecting the quality of students’ design project, especially the concept-score. Students who made extensive use of DYNAMO would, in other words, show higher scores and, in particular, a higher concept-score than those who used the tool only occasionally or not at all.

Hypothesis 2: We expected the independent variables design3 and questionnaire not to contribute to the dependent variables reflecting the quality of the design projects. Students with lower scores on design3 and/or not willing to complete the questionnaire would, in other words, not come up with lower scores on their project than other students.

9.5.3 STATISTICAL ANALYSES

In order to examine the hypotheses formulated above, we analysed our data by a standard regression analysis. The standard variant differs from the stepwise-forward variant used in chapter 5 in that it considers all independent variables at once for their contribution to the dependent variable. The stepwise-forward variant is more appropriate for explorative studies in which a model is being built up, as was the case in our first experiment. In this study, however, we wanted to verify the model, whence the choice for a standard regression analysis.

The results of our analyses are summarised in table 9.1. As the first column indicates, the analyses were conducted separately for the two options (library and architecture school). The second column of the table delineates the dependent variables for which results of statistical significance were found. The third through the fifth column shows the coefficient of each independent variable. Finally, the last column shows the statistics for the equation.

The intercept is not included in the table, as it provides little information about the relationship between the dependent and independent variables. Hence, to compose the equation for the first dependent variable, table 9.1 should be read as follows:

\[
global-score = a + 0.946 \times \text{frequency of use} + 1.038 \times \text{design3} + 0.687 \times \text{questionnaire}
\]

In this case, only one of the three variables, frequency of use, is considered a significant contributor as indicated by the asterisk (p< .05). The last two variables – design3 and questionnaire – are needed to create a sufficient equation, but are no major contributors. The final column shows the relevant statistics for the above equation; the asterisk indicates that a sufficient equation is reached.

---

14 For a more detailed report of the regression analyses for these three variables, see tables 4 to 10 in appendix D.
CHAPTER 9

<table>
<thead>
<tr>
<th>Option</th>
<th>dependent variable</th>
<th>frequency of use</th>
<th>design3</th>
<th>questionnaire</th>
<th>equation statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>library</td>
<td>global-score&lt;sub&gt;i&lt;/sub&gt;</td>
<td>.946*</td>
<td>1.038</td>
<td>.687</td>
<td>Adj. R&lt;sup&gt;2&lt;/sup&gt; = .337, F(3,13) = 3.7096 *</td>
</tr>
<tr>
<td></td>
<td>concept-score&lt;sub&gt;e&lt;/sub&gt;</td>
<td>.395*</td>
<td>.285</td>
<td>-.314</td>
<td>Adj. R&lt;sup&gt;2&lt;/sup&gt; = .161, F(3,13) = 2.0214</td>
</tr>
<tr>
<td></td>
<td>reuse-score&lt;sub&gt;e&lt;/sub&gt;</td>
<td>.477*</td>
<td>.196</td>
<td>-.228</td>
<td>Adj. R&lt;sup&gt;2&lt;/sup&gt; = .208, F(3,13) = 2.4013</td>
</tr>
<tr>
<td></td>
<td>form-score&lt;sub&gt;e&lt;/sub&gt;</td>
<td>.392*</td>
<td>.388</td>
<td>-.222</td>
<td>Adj. R&lt;sup&gt;2&lt;/sup&gt; = .253, F(3,13) = 2.8032</td>
</tr>
<tr>
<td></td>
<td>function-score&lt;sub&gt;e&lt;/sub&gt;</td>
<td>.451*</td>
<td>.324</td>
<td>-.389</td>
<td>Adj. R&lt;sup&gt;2&lt;/sup&gt; = .320, F(3,13) = 3.5068 *</td>
</tr>
<tr>
<td></td>
<td>creativity-score&lt;sub&gt;e&lt;/sub&gt;</td>
<td>.432</td>
<td>.350</td>
<td>-.071</td>
<td>Adj. R&lt;sup&gt;2&lt;/sup&gt; = .148, F(3,13) = 1.9267</td>
</tr>
<tr>
<td>architecture school</td>
<td>reuse-score&lt;sub&gt;e&lt;/sub&gt;</td>
<td>-.621*</td>
<td>.671</td>
<td>.277</td>
<td>Adj. R&lt;sup&gt;2&lt;/sup&gt; = .591, F(3,6) = 5.3432 *</td>
</tr>
</tbody>
</table>

TABLE 9.1. Results from the regression analyses of the data. Only independent variables for which results of statistical significance were found are listed.

Inspection of table 9.1 reveals a frequent significant contribution of the variable frequency of use. No less than six scores turn out to be significantly influenced by this variable: global-score<sub>i</sub>, concept-score<sub>e</sub>, reuse-score<sub>e</sub>, form-score<sub>e</sub>, function-score<sub>e</sub> for the library designs, and reuse-score<sub>e</sub> for the school projects. In addition, the contribution of frequency of use to the creativity-score<sub>e</sub> in the library group might be considered a trend (p< .06). Further inspection of the coefficients (the b-values in table 9.1) reveals that the influence on all library scores is positive, whereas the school projects are affected in a negative way.

The other two independent variables – design3 and questionnaire – did not turn up as significant contributors to the seven scores listed, nor to the other dependent variables. On average, the coefficients for the variable questionnaire fluctuate around 0, yet for design3 these values are clearly positive. We should, however, keep in mind that only for three variables – global-score<sub>i</sub> and function-score<sub>e</sub> for the libraries and reuse-score<sub>e</sub> for the schools – a sufficient equation is reached.

9.5.4 DISCUSSION

Does using DYNAMO positively affect the quality of students’ design projects, and more specifically the concept behind these projects, as predicted by Hypothesis 1?

The results of our analyses may be taken to suggest that the answer to this question is yes. The dependent variables global-score<sub>i</sub>, concept-score<sub>e</sub>, reuse-score<sub>e</sub>, form-score<sub>e</sub> and function-score<sub>e</sub> are positively influenced by the variable frequency of use. Moreover, its influence on the variable creativity-score<sub>e</sub> clearly points in the right direction. The latter is probably much more important than it may appear, since it implies that no signs for design fixation were found. It is, by consequence, highly
improbable that DYNAMO users risk to be hindered in their creativity. On the contrary, the positive coefficient of frequency of use in the regression equation for creativity-score, \( (p < .06) \) might be taken to reflect DYNAMO’s potential to improve the creative character of students’ design.

The effect of using DYNAMO, however, heavily depends on the option students had chosen. For the library designers, effects of statistical significance were found for the overall quality of the design as well as for four specific aspects, including the quality of the concept. As far as the school designers are concerned, only the design’s attitude towards the existing building (the focus of the assignment) was significantly influenced, be it in a negative way. The most plausible explanation for this contrast between both groups seems to lie in the content of the case base. At the time of the assignment, DYNAMO contained eight libraries compared to only one school, a school of contemporary arts. Moreover, of the twenty-one conversion and/or extension projects, nine were situated in a factory hall or other industrial building, whereas no more than two dealt with a building comparable to the castle of Arenberg. In this respect, DYNAMO’s different impact on library and school designers clearly illustrates how the effective performance of a CBD tool is determined by the richness, diversity and size of its case base [Oxman & Shabo, 1999].

Although students’ library designs seem positively influenced by the variable frequency of use, it should be noted that only for the overall quality and functional organisation of their projects a sufficient regression equation was reached. By consequence, further evidence is needed before we can confirm that using DYNAMO indeed has a positive impact on the quality of students’ design concept. In the study presented, the extent to which students made use of the tool was taken to be reflected by the number of times they logged on. A better strategy for determining the impact of using DYNAMO, therefore, might be to record how much time students spend in using the tool and, more importantly, what they do during this time. Permanently keeping track of the path they follow through the case base should allow us to establish a more accurate measure for the influence of using DYNAMO.

In addition, further examination of this hypothesis requires a better evaluation of the quality of students’ projects to see whether these initial findings apply across judgments by different critics. The analyses to examine the influence on the quality of the concept, as well as on other aspects of the design, are based on scores from one critic only. Her colleague, as we have pointed out, did not seem to distinguish between these different aspects, which is why his judgments were not taken into account. The evidence from these analyses – that using DYNAMO positively influences the quality of students’ designs as appreciated by one critic, but that this appreciation seriously differs from that by another one – raises the question of how differences between judges impact learning outcomes.

Does DYNAMO live up to its ambition not to privilege more skilled and more motivated students, as predicted by Hypothesis 2?

As far as student motivation is concerned, the results of our analysis to date provide reason for encouragement. No statistical evidence was found that students’ level of motivation had any impact on the quality of their design projects. For none of the
scores, the variable questionnaire turned out to be a significant contributor. On average, the coefficients for this variable fluctuate around 0, which might be taken to suggest that it has little or no influence. However, once again it should be noted that for several scores no sufficient regression equation was reached. Moreover, the measure for student motivation considerably differed from the one used in the first experiment. In that experiment, the number of discussions between student and studio teacher were taken to reflect how motivated the student was to design a good project. Unfortunately, the teachers involved in the reuse studio did not keep track of which students discussed their project with them and when, so that we had to look for another motivation measure instead. It is obvious, however, that students’ motivation to complete a questionnaire does not necessarily equal their motivation to design a high-quality project. Therefore, further research is needed to examine whether these initial, promising findings also hold when using the same measure of motivation as in the first experiment.

To some extent, the same story goes for the influence of students’ design skill. For none of the scores, the variable student3 appears to be a significant contributor. An important difference with questionnaire, however, is that the coefficients for student3 are obviously positive. Although, strictly speaking, no statistical evidence was found that the variable student3 positively influences students’ design results, its pattern of \( b \)-values seems to point in that direction. In fact, this does not come as a surprise, as it confirms the findings of our first experiment. According to design teachers, as we have mentioned, the most skilled design students are often students with a considerable background, built up by travelling, reading, visiting museums, etc. When exposed to cases during design, this background may help students to make sense of these cases, and thus to (re-)use them for their project in a productive way. The fact that, in this experiment, cases were provided by medium of a computer is unlikely to have altered that situation.

In summary, our results to date put us in a hopeful mood about the impact of using DYNAMO during design. The results of the regression analyses for the library projects might be taken to suggest that the quality of students’ design projects – the overall quality as well as the quality of specific aspects – is positively influenced by using DYNAMO. For some aspects, however, no sufficient equation was found, so that further evidence is needed to confirm our hypotheses.

9.6 Summary and conclusion

This chapter has given an account of DYNAMO’s baptism of fire. It has described how the system was used and appreciated in a 4th year design studio devoted to reuse. Based on students’ responses to a questionnaire and the quality of their design results, an attempt was made to answer three central questions: 1. Did DYNAMO manage to engage students into in-depth explorations of design cases? 2. What factors stimulated or hampered this engagement? 3. Did the use of DYNAMO have any impact on students’ design performance?
Despite the prototype nature of the system, the answers to the questionnaire seem to suggest that students indeed found DYNAMO engaging. Except for two, all respondents would like to use the tool again for future assignments. However, their responses raise questions about the nature of their engagement. None of the students who used the tool exploited the opportunity to feed the case base, be it by writing an on-line comment, adding a project or indexing cases in a new way. The fact that students were not engaged as originally expected, might explain why they remained neutral about the tool’s support for conceptual design. Currently, DYNAMO’s ability to engage students comes from the quality and colours of its visual material and the paper it saves, rather than from the opportunities it offers for in-depth explorations of cases. In part, the problem might be due to the strict time schedule of the assignment. Other possible explanations may be the lack of enthusiasm on the side of the studio teachers, or problems of computer equipment. At the time of the assignment, the design studio was not yet equipped with computers to consult DYNAMO, let alone with input devices (scanners or digital cameras, for instance) to add information. As long as students must leave their drawing board – or even the studio – to access the tool, the majority simply will not use it. Although these obstacles – lack of time, of encouragement by the teachers, and of equipment in the studio – may not relate directly to DYNAMO itself, they might have prevented the tool from functioning the way it was originally meant to. Actively encouraging students to consult and feed the tool and providing hardware devices to do so, could greatly enhance the system’s ability to engage students in reasoning more deeply with cases.

Other obstacles, however, do relate to DYNAMO itself and thus are to be charged on our own bill. Although students’ comments were enthusiastic, they have also pointed out some aspects of DYNAMO that leave considerable room for improvement. Awaiting extra encouragement and computer equipment, the prototype was therefore significantly modified following this evaluation. First of all, we did away with the platform-incompatibilities and browser-dependencies of the programme. Furthermore, the user interface was given a profound face-lift to help users find their way more easily in the case base, and to make opportunities for in-depth exploration – feeding the case base, adding information, writing on-line comments – immediately attract the eye.\footnote{See also chapter 8, pp.202-203.}

The analyses to examine the second question, on stimulating and/or hampering factors, yielded no results of statistical significance. The extent to which students made use of the prototype does not seem to correlate significantly with personal computer equipment, familiarity with the Internet, use of CAD software or level of computer literacy. The question remains why the tool did not succeed in engaging all students who participated in the studio.

With regard to the third question, the results of our analyses provide every reason for optimism. As far as the library project is concerned, they strongly suggest that using DYNAMO positively influences students’ design performance. Furthermore, no statistical evidence was found that this use would increase the danger of design
fixation. On the contrary, if DYNAMO can be said to have any impact on the creativity of students’ designs, it is a positive one. Strictly speaking, we cannot yet pronounce upon whether the tool’s influence varies with students’ skill or motivation. As far as the latter is concerned, however, our results so far clearly point in the right direction, causing us to believe that further investigation will confirm our hypotheses. After all, the experiment described here was rather limited, and as with the one in chapter 5, it was very difficult to control. Studies of this nature cannot produce results that may be generalised. They describe what has happened, and to whom, and suggest but do not define reasons for the results observed. They are nevertheless important as preliminary work in the development of CBD and other design tools that take advantage of the demands of (student-)architects during design.

However, there are pitfalls in trying to assess today’s CBD systems, DYNAMO included. The most important feature of DYNAMO is not merely that it presents cases, but that those cases trigger in-depth explorations, stimulate reflection and prime discussions between students and design teachers. The latter must help students understand the relevance of retrieved cases through appropriate indexing and helpful on-line comments. Despite our firm intentions to actively involve studio teachers in the development of our prototype, we could hardly get them to have a look at the tool. As far as this involvement is concerned, we do not hesitate to say that DYNAMO’s performances are more than below par. It is, by consequence, extremely difficult to evaluate the impact of our prototype without taking into account our failure to ‘upgrade’ users – students and studio teachers – from passive information consumers to active participants in constructive debates.

Therefore, we would like to end this chapter by stressing the need to adopt a broader perspective when evaluating CBD tools. The effective performance of any such tool in architectural education does not only depend upon the richness, diversity and number of cases, or on the power of the representational system [Oxman & Shabo, 1999], but also upon how it is viewed by teachers, perceived by students and supported by the school. If we are to develop CBD tools that effectively support (student-)architects during design, the influence of the context in which they are used should not be underestimated. Judging from our first experience with DYNAMO, this context is much more difficult to ‘dynamise’ than the content and structure of a case base.
EPILOGUE

“Scientific tradition derives its capacity for self-renewal from its belief in the presence of a hidden reality, of which current science is one aspect, while other aspects of it are to be revealed by future discoveries. Any tradition fostering the progress of thought must have this intention: to teach its current ideas as stages leading to unknown truths which, when discovered, might dissent from the very teachings which engendered them.”

Michael Polanyi (1966)
“A small country was ruled from a strong fortress by a dictator. The fortress was situated in the middle of the country, surrounded by farms and villages. Many roads led to the fortress. The general knew that an attack by her entire army would capture the fortress. She gathered her army at the head of one of the roads, ready to launch a full-scale direct attack. However, the general then learned that the dictator had planted mines on each of the roads. The mines were set so that small bodies of men could pass over them safely, since the dictator needed to move his troops and workers to and from the fortress. However, any large force would detonate the mines. Not only would this blow up the road, but it would also destroy neighbouring villages. It therefore seemed impossible to capture the fortress.

However, the general devised a simple plan. She divided her army into small groups and dispatched each group to the head of one road. When all was ready she gave the signal and each group marched down a different road to the fortress so that the entire army arrived together at the fortress at the same time. In this way, the general captured the fortress and overthrew the dictator.”

The story of the general is, to some extent, the story of this thesis. The fortress stands for the role of cases in conceiving architecture. It is situated in the realm of architectural design, surrounded by analogies, metaphors, heuristics, types and proportional systems. Many ways lead to a better understanding of this role – theoretical reflection, literature study, interviewing architects, (more or less) controlled experiments, building a CBD tool – yet, each of these ways is severely restricted in its view on the phenomenon, as it tends to approach it from one side only. Indeed, when dealing with complex phenomena like architectural design, no single research method is likely to yield the whole truth [Cilliers, 1998]. The last thing this could mean is that research is unnecessary. To the contrary, we should do all we possibly can to investigate how designers (re-)use cases in conceiving architecture. Therefore, we have replaced ‘the research method’ by something more sensitive to the contingencies and complexity of our subject of study. Just like the general, we have opted to take different directions on our way towards a better understanding of Case-Based Design in architecture. Instead of following one approach, several methods have been used, allowing for different avenues of advance, different viewpoints and, perhaps, a better understanding of our subject. Rather than a system of certainties, we have called in these methods as lists to outwit the role of cases in architectural design [Neutjens, 1984].

Followed directions

Turning down the road of literature study, chapter 1 has pointed out the highly complex nature of conceiving architecture. The task makes high demands on architects’ knowledge, as it entails active knowing and passive knowledge at both concept and component level. In order to comply with these demands, architects have

1 Freely after ‘the General’s Problem’, taken from [Verstijnen, Wagemans, Heylighen & Neuckermans, 1999].
EPILOGUE

an arsenal of strategies at their disposal, which supply one or more types of knowledge needed for concept generation. Chapter 1 has attempted to describe these strategies – analogy, metaphor, heuristics, typology and case (re-)use – from the point of view of an architect being involved in the design process. From this perspective, their major usefulness seems to be that they provide the designer with a stable starting point, with something to hold on to within the wicked web of aspects, issues and requirements he must take into account.

The remainder of the thesis has focused on one specific strategy, the (re-)use of concrete design cases. In part I, we have critically explored the Case-Based Design approach within the context of architectural design. By way of introduction to this approach, chapter 2 has taken a short excursion into the domain of AI, in which CBD is historically rooted. After zooming in on the sub-domain of knowledge engineering, we have taken a look at Case-Based Reasoning, and subsequently at the application of CBR to design. We have discussed the basic premise underlying this application, and outlined the promises it holds for the field of architecture. Since architectural design may be considered a ‘weak-theory’ domain, CBR researchers assume that architects rely heavily on past design experience, especially during the early, conceptual stage of design. This very premise has laid the foundation for various CBD systems, design tools built around concrete cases. In general, the advantage of using cases seems to derive from the fact that they contain experiences in their original form, which makes them much richer and vivid than abstract rules or models.

Switching from a general to a more concrete perspective, chapter 3 has turned down the case study road. By putting some specific CBD projects under the microscope, it has become a so-called case base of CBD systems for architectural design. The seven quite different cases it contains – Archie-II, CADRE, FABEL, IDiom, PRECEDENTS, SEED and WEBPAD – were selected because of their special concern with the field of architecture, and because taken together they give a fairly good overview of the various directions in CBD research. Since this research is in a constant state of flux, however, the chapter could only provide a snapshot of current initiatives in this field.

Instead of adopting a CBD research perspective, chapter 4 found it high time to consider case (re-)use from the point of view of an architect/designer. In order to do so, we have handed over the floor to six architects, who were asked to describe whether, how and why they (re-)use cases during design. Although the interviewees considerably differed in terms of generation, approach and view on architectural design, they were fairly consistent in their replies regarding design practice. Particularly striking was their nearly unanimous denial that cases were used for concept generation. Yet, when reconsidering this denial afterwards, several ingredients of the CBD recipe turned up, be it in a more subtle form than assumed by CBD researchers. As to the role of cases in design education, however, opinions were more divided, suggesting that, although CBD claims to derive from how designers learn (to) design, in architecture the role of cases in this learning process is far from clear yet.

In order to shed more light onto this role, chapter 5 has chosen to turn down yet another road. An attempt was made to conduct an experimental study about the
positive and/or negative effects of cases on students’ designs. The results of this study strongly suggest that exposure to cases during design has a positive effect on the quality and creativity of students’ projects. Unfortunately, these effects seem to be a privilege of more motivated and skilled design students only. Despite this privileged nature, however, the results of the study point out the need for a radical change of mentality in design education, where the current (over-)emphasis on originality and fear for design fixation wrongfully exclude cases from supporting design in the studio.

By way of stop over on our exploration, chapter 6 has tried to bring some of the key issues raised in the previous chapters into sharper focus, by comparing and confronting the views of different stakeholders – CBD researchers, practising architects, design teachers and student designers. Judging from this confrontation, the CBD approach gives one plausible explanation of how architects and other designers acquire and use (design) knowledge, yet fails to satisfy the contemporary request to support architects during design in several important respects. First of all, it derives from a model of cognition that views architectural design as a form of problem solving. As such, tools that flow from CBD research are largely inconsistent with the complex reality of architectural design. Secondly, when it comes to implementing CBD tools, features at the core of the cognitive model (concreteness, integration and dynamics) are lost in the transfer from human to computer memory. Last but not least, the group aimed at by these tools appears to be loaded with prejudices and fear for design fixation, which is likely to be implicated in the current gap between CBD research and architectural design.

In an attempt to (help) bridge this gap, we decided to enter into the role of CBD system developer. Part II has given an account of the ideas and efforts that went into the design, implementation and evaluation of a Dynamic Architectural Memory On-line. With DYNAMO we have tried to reconcile both parties – CBD researchers and architects/designers – by focusing on aspects other than those emphasised in CBD tools so far. The hope, of course, was that such a shift of attention would allow fine-tuning the CBD approach so that it would speak to (student-)architects. In essence, this shift came down to three transformations vis-à-vis the CBD systems studied. The first was to extend their primary concern with design products towards design processes. The second was to tackle their static nature so as to do justice to Schank’s Dynamic Memory Theory. The third was to upgrade users from passive case consumers to active case providers. As such, each of these transformations may represent but a subtle shift in emphasis, yet taken together they seemed quite promising to finally give architecture a part in CBR’s success story.

As to the fulfilling of this promise, DYNAMO’s baptism of fire in a design studio put us in a hopeful mood. Despite the prototype nature of the system, the students who participated in the studio found DYNAMO engaging, even to the extent that they would like to use the tool again in the future. Furthermore, the results of our analyses strongly suggest that using DYNAMO positively influences students’ design performance and is unlikely to increase the danger of design fixation. On the contrary, if DYNAMO can be said to have any impact on the creativity of students’ designs, it is a positive one. Strictly speaking, we cannot yet pronounce upon whether the tool’s
influence varies with students’ skill or motivation, yet, as far as the latter is concerned, our results to date at least point in the right direction.

Although DYNAMO managed to engage students, they were not always engaged in ways that met with our original expectations. The most important feature of DYNAMO is not merely that it presents cases, but that those cases trigger in-depth explorations, stimulate reflection and prime discussions between students and design teachers. However, neither students nor teachers exploited the possibilities for (inter-)action provided by the tool. Given the strict time schedule of the assignment, a major obstacle for active participation seems to have been lack of time. In this respect, our first experience with DYNAMO has alerted us to the need for a broader perspective when evaluating CBD tools, and CAAD tools in general. The influence of the context in which these tools are used should not be underestimated. To put dynamics into DYNAMO’s case base is one thing, to dynamise the context it is meant for is yet something different.

At this point, it is obvious that the story of the thesis takes a different turn than the general’s story. Whereas the general succeeded in her ambition to capture the fortress, we cannot claim to have provided the final word on the role of cases in conceiving architecture. On the contrary, the initial success of DYNAMO only emphasises the importance of further research about case (re-)use in architectural design. At least three directions seem worth pursuing in the future. Firstly, additional work is necessary to fine-tune CBD’s model of cognition to the complex reality of architectural design. Secondly, DYNAMO should be developed from a prototype into a fully-fledged design tool. Finally, additional experience and evaluation is needed to explore the usability of DYNAMO’s approach as a strategy for quality improvement in architecture, as well as in other design disciplines. Each of these directions will be considered briefly here to close our story.

**Future directions**

**CBD’S MODEL OF DESIGN COGNITION**

The results of our research to date strongly suggest that using cases during design positively influences the quality of the resulting design product. A question that remains unanswered, however, is how architects/designers deploy cases to conceive high quality designs. Additional research is needed to develop a more profound understanding of how cases in general, and DYNAMO in particular, affect the design and learning process of architects and architecture students. Such understanding would allow us to tailor CBD’s model further to the reality and complexity of architectural design. This would, in turn, enable DYNAMO (as well as other CAAD tools) to take advantage of architects’ needs during design, and to help them improve the quality of their design projects.

In order to investigate the question how cases are deployed to conceive high-quality designs, it seems plausible to follow the road of experimental study. One possibility is
to ask professional and student architects to use DYNAMO for a specific design task. By permanently monitoring the route users follow through the case base, several user profiles may be identified, which can be linked to the quality of the resulting design products. A comparison between profiles of professional and student architects, and/or between profiles that correspond to high versus low quality designs, might lead to a better insight into how cases in general, and DYNAMO in particular, can be deployed productively to conceive high quality designs. Such comparison, however, requires a better evaluation of the quality of the designs than simply rating some aspects on a five-point scale. If we want to understand how cases affect the quality of design projects, a profound content analysis of these projects is needed.

FROM PROTOTYPE TO FULLY-FLEDGED TOOL

Although students’ comments on DYNAMO were enthusiastic, they have also pointed out some aspects of the tool that leave considerable room for improvement. Therefore, DYNAMO should be further developed from a prototype into a fully-fledged design tool, so as to account for user needs it is currently unable to deal with effectively.

Integration within IDEA+

This development first of all requires its integration within the IDEA+ environment that aims at supporting architects from the very start of the design process. The role of DYNAMO in this environment is to provide IDEA+ users with a rich source of inspiration and information in the form of design cases. In order to fully integrate the process of consulting cases with the design process, users must be able to swiftly go back and forth between the case base on the one hand and the IDEA+ modeller, i.e. the module they are using to develop their design, on the other hand. In this respect, the ideal would be if DYNAMO could be accessed from within the modeller itself instead of through a Web browser, as is currently the case.

Content search, thesaurus and feedback

In addition, DYNAMO could be improved by extending the retrieval mechanism of the case base with a content search facility. The idea is to compare the search criteria specified by the user not only with the cases’ indices in the database, but also with the text files (interviews, articles, comments) they contain. By allowing to find documents which contain a given word, such content search would considerably increase the power of the tool.

Furthermore, although DYNAMO’s facilities for interactive case submission and (re-)indexing have not yet received extensive use, we are already aware of some important problems such interaction would bring about, especially with regard to (re-) indexing. It is far from likely that (student-)architects in different contexts and at different levels of expertise will automatically agree upon a common indexing vocabulary. A possible solution may therefore be to link DYNAMO’s indexing system to a recognized thesaurus, such as the on-line Art and Architecture Thesaurus of the

2 See chapter 7, p.173.
EPILOGUE

Getty Research Institute [Web_ATT, 2000]. In this way, users could be encouraged to select indices that are available in the thesaurus.

It is obvious, however, that a thesaurus by no means can replace a (human) administrator. With a view to quality control, it is not only important that new entries are checked, but also that users get individual feedback concerning the information they have entered. Feedback facilities could be provided via DYNAMO’s user interface or by using e-mail.

What you see is what you get

Another possible extension is to supplement DYNAMO’s indexing system with visual keys. In the current version of DYNAMO, cases are labelled (and thus accessible) by verbal indices, c.q. keywords. Essentially, however, a design comes into being through manipulation of non-verbal information: the visual is the way in which the designer knows, thinks and works [Cross, 1982]. Indeed, concepts in architectural design are often more easily and completely expressed by an image or a picture than in words. Thus, if DYNAMO is to really support architects during conceptual design, cases should not only be represented, but also accessed in a visual way. Hence the idea to expand the tool by allowing to label and link design cases by images as well. Visual keys may be diagrams of a project, but also any photograph, picture or icon that can illustrate its design concept.

At first stage, we envisage the use of these visual keys not so much for directed search, but rather for browsing. In other words, images, photographs, sketches, etc. can be used as labels, but not (yet) as probes. This puts the task of retrieving relevant cases more at the side of the users themselves. It profits from their cognitive, perceptual and intuitive skills to find cases that may be of interest, but could be missed by search algorithms, for instance because they are not directly relevant to the query [Vande Moere, 1998]. The more visual labels DYNAMO will contain, however, the more difficulties users will have to find the right one at the right time. One possible solution might be to equip our tool with image database retrieval techniques, which allow querying databases by image content.

WHAT IS DYNAMO GOOD FOR?

DYNAMO was designed to address a specific problem: to make cases available in a way that fits architects’ designerly way of thinking. The essence of the approach adopted in DYNAMO is to support architects by exchanging design experience in the form of concrete projects. The relevance of this approach, however, transcends the domain of architecture as such. More and more, architectural design is being viewed as a model for complex design tasks in general [Heylighen & Bouwen, 1998; Rechtin, 1991]. A third direction for future research is therefore to explore and investigate the potential of DYNAMO’s approach as a strategy for quality improvement in various design disciplines.

---

3 See chapter 6, p.147.
DYNAKO may support other levels of the conceptual framework for CAAD than building design.\(^4\) Nothing in it makes it applicable to contain only entire buildings. Given its flexible memory organisation, it would be perfectly possible to develop a version for the level of rooms or building elements, or a version for urban design. Besides having to change the content of the case base, we would only have to adapt the indices to accommodate different scales.

DYNAKO may support software design as well. Imagine a software engineer involved in the design of a graphical user interface for a specific software application. Using DYNAKO in this example would require only simple changes. Instead of filling the case base with building designs, for instance, we would have to fill it with software applications and label these by their purpose, type of users, etc. Such case base could draw the engineer’s attention to the non-technical aspects of a user interface, like context, human factors or software psychology, factors that are usually underexposed in the literature on software engineering [Reymen, 1998]. Similarly, the approach adopted in DYNAKO may prove useful for supporting designers and improving design quality in say mechanical engineering or industrial design.

Awaiting these future excursions, we hope that this thesis at least has lifted a corner of the veil about the role of cases in conceiving architecture. At this point in our exploration, perhaps the best way to characterise this role is by comparing it not to a fortress, but rather to a puff pastry with countless layers and stuffings, each of which may trigger reflection. Reflection on how an existing design has come about can in turn contribute to a better understanding of how a new design situation may be addressed. In this respect, architecture from the past can be regarded as a mine of raw material – in the sense of design knowledge – to conceive architecture for the future. Architecture historians might have the impression that we think too utilitarian, because we focus on the practical value of our built heritage. In our opinion, however, to exploit this heritage as raw material for future designs is no disgrace. If we are to support designers in conceiving high quality architecture, it is simply a must.

\(^4\) See chapter 7, pp.171 ff.
A

ARCHITECTS’ QUESTIONNAIRE

This appendix contains the questions that were used for the interviews discussed in chapter 4. The questionnaire was sent to the interviewees in advance, so that they had time to prepare their answers. During the interview itself, the questions were used as a guideline. However, since the interviews often took place in a rather informal atmosphere, the topics were not always discussed in this order.

1. Where do you draw inspiration from during the early phase of design, more specifically during concept generation?

   Does it happen that you draw inspiration from existing projects – whether or not designed by yourself?

   If yes, do you look for a specific project?

   For instance:
   – projects with the same functional programme
   – projects on a similar site
   – projects by a well-determined architect, …

   In addition to existing architectural projects, are there other sources of inspiration you draw on during concept generation?

   For instance:
   – textbook architectural history or theory from the past
   – things that do not directly relate to architecture, such as drawings by (grand-)children, (M)TV, dance, theatre, museums, …

2. Are there projects (designed by other architects or by yourself) that have played – consciously or unconsciously – a role in one of your own projects?

   If yes, how would you describe this role?

3. Are there projects by other architects that you clearly recognise or assume to be inspired or influenced by another existing project?

4. Are there, except for concept generation, other phases of the design process where you call in existing projects, whether or not designed by yourself?

   If yes, which phases?

   Why?

5. Does it happen that you notice that students draw inspiration from existing projects during concept generation?

   If yes, does it happen often?
ARCHITECTS’ QUESTIONNAIRE

Which students?
   – Predominantly good design students, or rather weaker students?
   – In the lower or the higher years?

In general, do you applaud it or rather disapprove of it?
Does it have a positive or rather negative effect on the final design result?
DESIGN ASSIGNMENT REUSE

Concept and objectives

Reuse, where new purposes are accommodated in existing (old) buildings, may be appropriate for various reasons: architectural (aesthetic), historic, documentary, building economic. It is a practice that becomes more and more actual, not only for ordinary buildings without special qualities, but also for the important historic and architectural patrimony, as well as for the still frequently vacant industrial patrimony, which has unsuspected spatial and architectural possibilities to offer. This is pointed out in many recent realisations.

A reuse project exercise does not only refer to an important actual architecture issue, but is also extremely important from a didactic point of view. A new development project exercise never or rarely attains the problems of execution (and if it does, only in a very limited way), where the materiality of the building is investigated. That is why new development designs often remain abstract and noncommittal.

A reuse exercise starts from the materiality of the building. Starting from the analysis of the architecture, formal (style, form, spatiality, ...) and constructive aspects, a frame of thought is formed that is more than the material boundary conditions within which a design must be worked out. The building is more than a material boundary, a necessary evil, a hindrance, a stand-in-the-way, an unruly piece of inconvenience, by which every flight of creativity is discouraged or stifled. It is neither a willing victim, in which one can cut, demolish or renovate as one sees fit from a pre-set architectural vision.

Good reuse is not a mutilation but an upgrading of the existing architecture. The building enters into dialogue, has something valuable to offer that must be accepted with gratitude. It is the task of the architect to recognise this gift and to acknowledge it as an essential contribution to his architectural concept, which in turn provides the building with a new reason for existence, a new youth. Thus, dialogue does not mean drowning the conversation partner with modish commonplaces nor subservience or slavish imitation of forms or constructions. New and old neither stand next to nor opposite one another, but complement each other to form a whole that is more than the sum of both parts. Reuse is an excellent antidote against superficial, modish design and architectural prejudice.

---

1 In the context of this exercise, the term reuse is used to translate the Dutch ‘vernieuwbouw’. Note that in Dutch, ‘vernieuwbouw’ and ‘renovatie’ are related but different terms: the former accommodates a new purpose in an existing building; the latter preserves the building’s original purpose.
Option 1: Conversion of an industrial building into a public library

1.1 SITE

Workplaces of the Belgian Railway in Kessel-Lo. Only halls 4, 5 and (9) qualify for reuse. The intermediate smaller buildings between hall 4 and 5 can possibly be demolished or replaced. The remaining buildings are demolished and replaced by a housing estate.

1.2 PROGRAMME

The halls to be preserved are (were) predetermined as market hall, public library and indoor sports hall (basket-ball) respectively. Because of soil pollution, the change of use at short notice has become problematic. Other similar uses are to be considered.

1.2.1 Public library

See existing sub-library: Leopold III laan 4 (sports complex) in Kessel-Lo.

Opening hours:
- Monday 15.30 - 19.30
- Wednesday 14.00 - 17.00
- Thursday 15.30 - 19.30
- Saturday 14.00 - 17.00

The total area of the existing library takes up more or less 800 m². Extension is nonetheless desirable. Some spaces are not available, others are too cramped.

Concise programme description
- Entrance, entrance hall with enclosed porch, sanitary facilities and cloakroom (coat hooks and lockers (limited)).
- Lending counter (2 persons) with preferably separate entry and exit. Security installation. Storage possibility for returned books. Catalogue space is no longer needed. Titles are looked up via terminals, placed here and there in the library room (necessary supplies for data lines)
- Youth library with children’s section, storytelling corner or room
- Adult library with preferably closed reading room (with reference works not for lending)
- Toy library with open and closed (for specific target groups) sections
- Reading places scattered between the shelves
- Section for journals and magazines
- A multimedia section (video, slides, film) is not foreseen in a public sub-library.
- An Internet room (+/- 10 PC’s) is to be considered (lockable)
- Meeting room +/- 30 people
small lecture room (possibly)  
- Office space  
  Room for 8 people (of which 2 at the lending counter). Open-plan office. Visual contact with book spaces is desirable (supervision).  
- Lending room toys  
- Break room with kitchenette  
- Closed book repository  
- Garage for library bus with adjacent book repository  
- Reading café, cafeteria (possibly in combination with adjacent market function)  
- Technical room (heating, ventilation), storage space, …

1.2.2 Market hall  
Few additional specific supplies  
- Storage space, sanitary facilities  
- Limited space for administration  
- Fireplace (for possible other purposes, fairs, concerts, happenings, …) (high-voltage cabin)  

The market hall is not to be designed. Yet, possible connections between library and market hall are to be investigated. The location of specific purposes may be important in this respect.

Option 2: Alteration of the West wing of the Arenbergcastle for the Department of Architecture

2.1 SITUATION  
West wing including both staircases. The 18th century stairs as well as the spaces surrounding the West tower should remain undisturbed. The two levels plus the attic are available.

2.2 PROGRAMME  
The existing purposes are optimised and extended.  
- Secretary (not freely accessible to students) with 3 to 4 work corners: 120 m².  
- Storage space (cupboards or storage room)  
- Photocopy room (for students)  
- Lockable information desk  
- Reception hall with multifunctional character (vertical circulation, waiting room, exhibition room, sitting area(s), cafeteria (vending machines), …) (150 to 200 m²)  
- Lockable exhibition room, also suitable for meetings, seminars, workshops, … 150m²  
- Material museum 150m²  
- Archive 150 m²
DESIGN ASSIGNMENT REUSE

– The existing lecture room (00.29) and design studio (PGC) move to another location.

**Work distribution and time schedule**

The exercise is worked out per 2 students.

- **Friday 29 01 99** assignment, introductory seminar, site visit
- **Monday 01 02 99** decide on option
  additional explanation
- **Friday 12 02 99** submission sketch design (site plan 1/500, plans, elevations, sections 1/200)
- **Monday 15 Tuesday 16 02 99**
  individual evaluation
- **Thursday 18 Friday 19 03 99**
  intermediate individual evaluation preliminary design
- **Monday 29 03 99** submission preliminary design: site plan (roof plan, surroundings) 1/200, plans, elevations, sections 1/100
- **Monday 19 Tuesday 20 04 99**
  final evaluation with presentation (scale-) model (1/200) and/or perspective drawings

Appendices

1: Description library programme
   (excerpt Syllabus 1988, … Ministry of the Flemish Community, Department public libraries)
2: Historic note Workplaces (excerpt M&L September-October 1995)
3: Historic note Arenbergcastle
4: Master plan, measurement plans halls IV and V (student measurements), measurement plans castle wing are available in the design studio.

Paul Van Aerschot, Hans De Petter, Ivo Van Hamme, Leo Van Broeck
STUDENTS’ QUESTIONNAIRE

This appendix contains the questionnaire that was used to evaluate the use of DYNAMO in a 4th year design studio (see chapter 9). The questionnaire was given to all students who participated in the studio after the conceptual stage of the assignment. Only 19 of the 48 participants were willing to complete the questionnaire.

Note: for questions marked with an asterisk (*), more than one option may be ticked.

Part A

1. Where did you seek/find inspiration/information during the 1st phase of your design? *
   - library books
   - library magazines
   - personal books
   - personal magazines
   - DYNAMO
   - textbooks, namely:……………………………………………………………….
   - other, namely:…………………………………………………………………….

2. Was the use of DYNAMO stimulated by the design teachers? Yes/No
3. Did you have the impression that the design teachers were well-informed about the content of DYNAMO? Yes/No

Part B

1. How often did you consult DYNAMO?
   - never
   - once or twice
   - several times
   - regularly
   - often
   If never, skip questions B.2 up to and including C, and immediately proceed with part D.

2. Are there particular projects in DYNAMO that have been helpful to your design? Yes/No
   If yes, which project(s) and why? ………………………………………………………
   ………………………………………………………………………………….………….
   ………………………………………………………………………………….………….
3. Did you look for more information on projects you found in DYNAMO? Yes/No
   a) If no, why not?
      o I had no time
      o I did not feel like it
      o DYNAMO provided sufficient information
      o other, namely:__________________________________________________________
   b) If yes, Where?
      o library books
      o library magazines
      o personal books
      o personal magazines
      o Internet
      o other, namely:__________________________________________________________

Did you add the information to DYNAMO as well? Yes/No
   If you did not add it, why not?
      o I had no time
      o I did not feel like it
      o I did not understand how to do it
      o other, namely:__________________________________________________________

4. Did you add projects to DYNAMO yourself? Yes/No
   a) If no, why not?
      o I had no time
      o I did not feel like
      o I did not understand how to do it
      o DYNAMO contained sufficient projects
      o other, namely:__________________________________________________________
   b) If yes, how many projects?______________________________________________
      Where did you get these projects? *
      o library books
      o library magazines
      o personal books
      o personal magazines
      o Internet
      o other, namely:__________________________________________________________
5. Did you comment on the projects in DYNAMO?  
Yes/No
If no, why not?
  o I had no time
  o I did not feel like it
  o I did not understand how to do it
  o other, namely: .................................................................

6. Which selection criteria (indices) did you use most during design?
  1. ..............................................................................................
  2. ..............................................................................................
  3. ..............................................................................................

7. Which selection criteria did you find absolutely unnecessary?
  1. ..............................................................................................
  2. ..............................................................................................
  3. ..............................................................................................

8. Which selection criteria are missing in your opinion and why? ..................
..............................................................................................

9. Did you add selection criteria yourself?  
Yes/No
If yes, which?.................................................................
If no, why not?
  o I had no time
  o I did not feel like it
  o I did not understand how to do it
  o criteria in DYNAMO were OK
  o other, namely: .................................................................

10. Did you use the browse facility?
  o never
  o once or twice
  o several times
  o regularly
  o often

11. If you consulted DYNAMO, was it
  o alone?
  o together with your design-partner?
  o along with several students?

12. Where did you consult DYNAMO? *
  o in the CAAD-classroom
  o in the design studio
  o in a PC-class of the K.U.Leuven
  o at your student room
  o at home
  o at another student’s home/room
  o other, namely: ......................................................................

If you used DYNAMO at more than one place, which one did you prefer? ....
Why? ....................................................................................
STUDENTS’ QUESTIONNAIRE

Part C

Give the following aspects of DYAMO a score between 1 and 5 (1=very poor, 2=poor, 3=neutral, 4=good, 5=very good)

<table>
<thead>
<tr>
<th>Aspect</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>support in exploring the design assignment</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>support in finding a concept</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>support in developing the concept</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>selection of projects</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>content of the projects</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>choice of selection criteria (indices)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>looking up projects</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>browsing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>adding projects</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>adding information to projects</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>commenting on projects</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>adding indices</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>interface</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>language used</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>programme speed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>speed of finding something</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ease of learning</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ease of use</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>programme help (manual)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Part D

1. During design assignments in previous years, did you make use of the documentation bundle (if provided)?
   - never
   - once only
   - several times
   - regularly
   - often
   Why (not)?

2. What advantages of DYAMO do you see compared to the traditional documentation?

3. What disadvantages of DYAMO do you see compared to the traditional documentation?

4. Would you find it useful to repeat the deployment of DYAMO in future design assignments? Yes/No
   Why (not)?
APPENDIX C

5. Would you find it positive if the tool were more integrated within a CAD-environment (for instance, directly accessible from within AutoCAD)? Yes/No

6. What did annoy you when you used DYNAMO? 

7. What would you like to see changed about the system? 

8. What did you find positive about the system? 

9. Other suggestions/remarcs/comments? 

Part E

1. Do you have use of a PC? *
   o at home
   o at your student’s room
   If yes, which one? 

2. Do you have an Internet-connection? *
   o at home
   o at your student’s room
   If yes, via Kotnet? Yes/No

3. Do you surf on the Internet? 
   o never
   o once or twice
   o several times
   o regularly
   o often

4. Do you use a CAD-package for your design project? Yes/No
   If yes, which one? 

STATISTICAL RESULTS OF DYNAMO’S EVALUATION

1. Results of the paired \( t \) tests

Students were asked how they felt about different elements of the system, and could choose from *very poor*, *poor*, *neutral*, *good* or *very good*. Their responses were coded from 1 to 5, with *very poor* corresponding to 1 and *very good* to 5. Tables 1 to 3 give an overview of the mean scores. The \( p \) values result from a comparison with the neutral 3 score. Significant differences are marked by an asterisk (*)..

**TABLE 1.** Average student opinions on design support offered by DYNAMO.

<table>
<thead>
<tr>
<th>exploration</th>
<th>concept generation</th>
<th>concept development</th>
</tr>
</thead>
<tbody>
<tr>
<td>mean</td>
<td>3.17</td>
<td>2.94</td>
</tr>
<tr>
<td>(st.dev.)</td>
<td>(.786)</td>
<td>(.827)</td>
</tr>
<tr>
<td>( p )</td>
<td>n.s.</td>
<td>n.s.</td>
</tr>
<tr>
<td>( n )</td>
<td>18</td>
<td>17</td>
</tr>
</tbody>
</table>

**TABLE 2.** Average student opinions on DYNAMO’s case base. \( p \) scores result from a comparison with the neutral 3 score.

<table>
<thead>
<tr>
<th>case choice</th>
<th>case content</th>
<th>selection criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>mean</td>
<td>3.56*</td>
<td>3.28</td>
</tr>
<tr>
<td>(st.dev.)</td>
<td>(.616)</td>
<td>(.752)</td>
</tr>
<tr>
<td>(st.dev.)</td>
<td>(.778)</td>
<td></td>
</tr>
<tr>
<td>( p )</td>
<td>&lt; .001</td>
<td>n.s.</td>
</tr>
<tr>
<td>( n )</td>
<td>18</td>
<td>18</td>
</tr>
</tbody>
</table>

**TABLE 3.** Average student opinions on aspects of DYNAMO’s user-friendliness.

<table>
<thead>
<tr>
<th>interface</th>
<th>language used</th>
<th>program speed</th>
<th>finding speed</th>
<th>ease of learning</th>
<th>ease of use</th>
<th>help contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>mean</td>
<td>3.31*</td>
<td>3.65*</td>
<td>3.12</td>
<td>2.83</td>
<td>3.38</td>
<td>3.50*</td>
</tr>
<tr>
<td>(st.dev.)</td>
<td>(.704)</td>
<td>(.606)</td>
<td>(1.166)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(st.dev.)</td>
<td>(.985)</td>
<td>(.957)</td>
<td>(.1.098)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( p )</td>
<td>&lt; .05</td>
<td>&lt; .001</td>
<td>n.s.</td>
<td>n.s.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( n )</td>
<td>16</td>
<td>17</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>18</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>18</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2. Results of the standard regression analyses

**TABLE 4. Regression summary for dependent variable global-score**
R = .67913382, R² = .46122274, Adjusted R² = .33688953
F(3,13) = 3.7096, p < .03977, Std. Error of estimate: 2.0141

<table>
<thead>
<tr>
<th>library</th>
<th>N=17</th>
<th>BETA</th>
<th>St. Err. of BETA</th>
<th>B</th>
<th>St. Err. of B</th>
<th>t(13)</th>
<th>p-level</th>
</tr>
</thead>
<tbody>
<tr>
<td>intercept</td>
<td></td>
<td>3.345627</td>
<td>3.988731</td>
<td>.838770</td>
<td>.416755</td>
<td></td>
<td></td>
</tr>
<tr>
<td>design</td>
<td>.403085</td>
<td>.224538</td>
<td>1.037853</td>
<td>.578135</td>
<td>.095898</td>
<td></td>
<td></td>
</tr>
<tr>
<td>frequency of use</td>
<td>.577734</td>
<td>.249596</td>
<td>.946474</td>
<td>.408901</td>
<td>.037619</td>
<td></td>
<td></td>
</tr>
<tr>
<td>questionnaire</td>
<td>.142949</td>
<td>.239163</td>
<td>.687212</td>
<td>1.149750</td>
<td>.560305</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TABLE 5. Regression summary for dependent variable concept-score**
R = .56399585, R² = .31809132, Adjusted R² = .16072778,
F(3,13) = 2.0214, p < .16068, Std. Error of estimate: .82671

<table>
<thead>
<tr>
<th>library</th>
<th>N=17</th>
<th>BETA</th>
<th>St. Err. of BETA</th>
<th>B</th>
<th>St. Err. of B</th>
<th>t(13)</th>
<th>p-level</th>
</tr>
</thead>
<tbody>
<tr>
<td>intercept</td>
<td></td>
<td>1.263485</td>
<td>1.637206</td>
<td>.771732</td>
<td>.45407</td>
<td></td>
<td></td>
</tr>
<tr>
<td>design</td>
<td>.303304</td>
<td>.252609</td>
<td>.284924</td>
<td>.237300</td>
<td>.251294</td>
<td></td>
<td></td>
</tr>
<tr>
<td>frequency of use</td>
<td>.660910</td>
<td>.280799</td>
<td>.395033</td>
<td>.167837</td>
<td>.034986</td>
<td></td>
<td></td>
</tr>
<tr>
<td>questionnaire</td>
<td>-.178856</td>
<td>.269062</td>
<td>-.313707</td>
<td>.471924</td>
<td>.517836</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TABLE 6. Regression summary for dependent variable reuse-score**
R = .59712976, R² = .35656395, Adjusted R² = .20807871,
F(3,13) = 2.4013, p < .11468, Std. Error of estimate: .94542

<table>
<thead>
<tr>
<th>library</th>
<th>N=17</th>
<th>BETA</th>
<th>St. Err. of BETA</th>
<th>B</th>
<th>St. Err. of B</th>
<th>t(13)</th>
<th>p-level</th>
</tr>
</thead>
<tbody>
<tr>
<td>intercept</td>
<td></td>
<td>1.080480</td>
<td>1.872299</td>
<td>.577087</td>
<td>.573739</td>
<td></td>
<td></td>
</tr>
<tr>
<td>design</td>
<td>.177671</td>
<td>.245379</td>
<td>.196494</td>
<td>.271375</td>
<td>.481852</td>
<td></td>
<td></td>
</tr>
<tr>
<td>frequency of use</td>
<td>.677439</td>
<td>.272763</td>
<td>.476698</td>
<td>.191937</td>
<td>.027423</td>
<td></td>
<td></td>
</tr>
<tr>
<td>questionnaire</td>
<td>-.110512</td>
<td>.261361</td>
<td>-.228197</td>
<td>.539690</td>
<td>.679329</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TABLE 7. Regression summary for dependent variable form-score**
R = .62673625, R² = .39279832, Adjusted R² = .25267486,
F(3,13) = 2.8032, p < .08148, Std. Error of estimate: .75049

<table>
<thead>
<tr>
<th>library</th>
<th>N=17</th>
<th>BETA</th>
<th>St. Err. of BETA</th>
<th>B</th>
<th>St. Err. of B</th>
<th>t(13)</th>
<th>p-level</th>
</tr>
</thead>
<tbody>
<tr>
<td>intercept</td>
<td></td>
<td>.007079</td>
<td>1.486267</td>
<td>.004763</td>
<td>.996272</td>
<td></td>
<td></td>
</tr>
<tr>
<td>design</td>
<td>.429736</td>
<td>.238370</td>
<td>.388367</td>
<td>.215423</td>
<td>.094637</td>
<td></td>
<td></td>
</tr>
<tr>
<td>frequency of use</td>
<td>.682410</td>
<td>.264972</td>
<td>.392398</td>
<td>.152363</td>
<td>.023056</td>
<td></td>
<td></td>
</tr>
<tr>
<td>questionnaire</td>
<td>-.131629</td>
<td>.253895</td>
<td>-.222106</td>
<td>.428416</td>
<td>.612864</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
RESULTS OF DYNAKO'S EVALUATION

**TABLE 8. Regression summary for dependent variable: function-score**
\[ R = .66879607, \ R^2 = .44729919, \ \text{Adjusted } \ R^2 = .31973931 \]
\[ F(3,13)=3.5068, \ p<.04641, \ \text{Std.Error of estimate: .71077} \]

<table>
<thead>
<tr>
<th>library</th>
<th>BETA</th>
<th>St. Err. of BETA</th>
<th>B</th>
<th>St. Err. of B</th>
<th>t(13)</th>
<th>p-level</th>
</tr>
</thead>
<tbody>
<tr>
<td>intercept</td>
<td>.483512</td>
<td>.343501</td>
<td>1.407597</td>
<td>.71077</td>
<td>.736713</td>
<td></td>
</tr>
<tr>
<td>design3</td>
<td>.361830</td>
<td>.227423</td>
<td>.324596</td>
<td>.204020</td>
<td>.135624</td>
<td></td>
</tr>
<tr>
<td>frequency of use</td>
<td>.789467</td>
<td>.252803</td>
<td>.450623</td>
<td>.144299</td>
<td>.008083</td>
<td></td>
</tr>
<tr>
<td>questionnaire</td>
<td>-.232198</td>
<td>.242236</td>
<td>-.388927</td>
<td>-.958564</td>
<td>.355268</td>
<td></td>
</tr>
</tbody>
</table>

**TABLE 9. Regression summary for dependent variable: creativity-score**
\[ R = .55477678, \ R^2 = .30777727, \ \text{Adjusted } \ R^2 = .14803356 \]
\[ F(3,13)=1.9267, \ p<.17515, \ \text{Std.Error of estimate: 1.0259} \]

<table>
<thead>
<tr>
<th>library</th>
<th>BETA</th>
<th>St. Err. of BETA</th>
<th>B</th>
<th>St. Err. of B</th>
<th>t(13)</th>
<th>p-level</th>
</tr>
</thead>
<tbody>
<tr>
<td>intercept</td>
<td>.241318</td>
<td>.118780</td>
<td>2.031636</td>
<td>.907265</td>
<td>.255737</td>
<td></td>
</tr>
<tr>
<td>design3</td>
<td>.302594</td>
<td>.254512</td>
<td>.350100</td>
<td>1.188917</td>
<td>.058501</td>
<td></td>
</tr>
<tr>
<td>frequency of use</td>
<td>.586775</td>
<td>.282915</td>
<td>.431962</td>
<td>2.074035</td>
<td>.058501</td>
<td></td>
</tr>
<tr>
<td>questionnaire</td>
<td>-.033034</td>
<td>.071361</td>
<td>-.01258</td>
<td>.990370</td>
<td>.058501</td>
<td></td>
</tr>
</tbody>
</table>

**TABLE 10. Regression summary for dependent variable: reuse-score**
\[ R = .85301847, \ R^2 = .72764051, \ \text{Adjusted } \ R^2 = .59146076 \]
\[ F(3,6)=5.3432, \ p<.03940, \ \text{Std.Error of estimate: .62933} \]

<table>
<thead>
<tr>
<th>school</th>
<th>BETA</th>
<th>St. Err. of BETA</th>
<th>B</th>
<th>St. Err. of B</th>
<th>t(6)</th>
<th>p-level</th>
</tr>
</thead>
<tbody>
<tr>
<td>intercept</td>
<td>-.027429</td>
<td>.01258</td>
<td>2.180274</td>
<td>.990370</td>
<td>.114983</td>
<td></td>
</tr>
<tr>
<td>design3</td>
<td>.466828</td>
<td>.364103</td>
<td>.64103</td>
<td>1.84247</td>
<td>.047180</td>
<td></td>
</tr>
<tr>
<td>frequency of use</td>
<td>-.755228</td>
<td>.249469</td>
<td>-.621106</td>
<td>-.2.48971</td>
<td>.677670</td>
<td></td>
</tr>
<tr>
<td>questionnaire</td>
<td>.145077</td>
<td>.633558</td>
<td>.276616</td>
<td>.43661</td>
<td>.677670</td>
<td></td>
</tr>
</tbody>
</table>
REFERENCES

Bibliography


Börner, Katy (1993a) ‘Structural similarity as guidance in case-based design’, in Richter, M.M., Wess, S., Althoff, K.-D. and Maurer, F. (eds.) *1st European Workshop on Case-
Based Reasoning (EWCBR’93), Posters and Presentations, 1, pp.14-19 (referred to in Voss e.a., 1994).


REFERENCES


Durand, Jean Nicolas Louis (1801?) Recueil et parallèle des édifices en tout genre, anciens et modernes, remarquables par leur beauté, par leur grandeur ou par leur singularité, Impr. de Gillé, fils, Paris (France).

Durand, Jean Nicolas Louis (1817) Précis des leçons d'architecture données a l'école royale polytechnique, Paris (France).


Hendrix, Ann (2000) A core object model for architectural design, PhD. Dissertation, KULeuven, FTW, Dept. ASRO.


REFERENCES


REFERENCES

Luyten, Anna (2000) ‘We moeten intelligente ruïnes maken (De 10 gebeurtenissen die architect Bob Van Reeth hebben veranderd)’, in De Standaard Magazine, April 7, 2000, pp.8-10.


REFERENCES


Quatremère de Quincy, Antoine C. (1832) Dictionnaire historique d’architecture comprenant dans son plans les notions historiques, descriptives, archaeologiques, biographiques, theoriques, didactiques et pratiques de cet art, 2 vols., Paris (France) (referred to in Rossi, 1982).


REFERENCES


Interviews/Lectures/Statements

Cordingley, Betsy (1996) Statement at the 3rd European Workshop on Case-Based Reasoning – EWCBR’96, Lausanne (Switzerland), November 14-16, 1996.
Winkels, Peggy (1998), Personal communication, Leuven (Belgium), December 12, 1998.

Webliography

REFERENCES

List of illustrations

CHAPTER 1


CHAPTER 3

Figure 3.1. [Domeshek & Kolodner, 1993 p.93]; 3.2. [Hua & Faltings, 1993 p.136]; 3.3. [Smith, Stalker & Lottaz, 1996 p.111]; 3.4. photo: David Heald in [Arnell & Bickford, 1984 p.259]; 3.5. photo: North Carolina State University in [Web_archiNFORM, 2000]; 3.6 & 3.7. [Oxman & Oxman, 1994a p.60/63].

CHAPTER 4

Figure 4.1. Jan Delrue; 4.2. [Web_Stellwerk, 2000]. p.109. [Henk De Smet & Paul Vermeulen, p.131]; p.114. [Van Aerschot, 1991]; p.120. photo: Carine Demeter/Archief Henk De Smet & Paul Vermeulen in [De Smet e.a., 1996 p.137].

CHAPTER 5

Figure 5.1. Herman Neuckermans; 5.2. Kenny Cupers.

CHAPTER 7

SAMENVATTING

CASE-BASED DESIGN IN ARCHITECTUUR

*Een kritische studie*
SAMENVATTING

Inleiding

Architecten, zo beweert men, leren ontwerpen uit ervaring. Leren uit ervaring is de essentie van Case-Based Reasoning (CBR), een sub-domein van de Kunstmatige Intelligentie gebaseerd op een cognitief model waarin het geheugen centraal staat. Hoewel het gaat om een relatief jong onderzoekstak, zijn reeds een verrassend groot aantal CBR-toepassingen commercieel succesvol in diverse domeinen en disciplines. Het domein van het architectuurontwerpen lijkt (voorlopig) niet te delen in dit succes. In tegenstelling tot wat oorspronkelijk werd verwacht, laat een overtuigende doorbraak van CBR in dit domein nog steeds op zich wachten. Ondanks ernstige inspanningen om zogenaamde Case-Based Design (CBD) systemen te ontwikkelen, i.e. CBR-toepassingen specifiek afgestemd op de taak van het ontwerpen, hebben deze systemen amper de weg naar architectenbureaus of architectuurscholen gevonden. Op zoek naar een verklaring voor dit tot nu toe beperkte succes, waagt deze thesis zich aan een kritisch onderzoek naar de rol van concrete ontwerpervaringen, zgn. casussen in het architectuurontwerpen.

Het onderzoek gaat uit van de veronderstelling dat als cases inderdaad zo'n belangrijke rol vervullen bij het ontwerpen als CBD-onderzoekers veronderstellen, de ontwikkeling van succesvolle en performante CBD-tools eerst en vooral een grondig inzicht in deze rol vereist. Een onderzoek naar het (her-)gebruik van cases in het architectuurontwerpen stelt echter heel wat problemen, te beginnen met de betekenis van het woord ‘case’ zelf. Een sluitende definitie van dit woord is al even moeilijk te vinden als een nauwkeurige omschrijving van ‘ervaring’, een begrip dat onlosmakelijk verbonden is met de term case in Case-Based Reasoning. Deze term verwijst immers naar de geïnterpreteerde representatie van een echte ervaring, inclusief alle details die de ervaring individueel maken [Kolodner, 1997]. Hoewel ontwerpervaring op verschillende manieren kan worden geïnterpreteerd, blijken de meeste CBD-systemen een case op te vatten als een concreet ontwerpproject. Doorheen de literatuur over het architectuurontwerpen hebben architecten, ontwerpmethodologen en psychologen uitdrukking gegeven aan deze notie van case, zonder de term evenwel expliciet te vermelden.

Naast een verwarrend jargon heeft een kritisch onderzoek naar de rol van cases in het architectuurontwerpen eveneens te kampen met problemen van methodologische aard. Wil men, in het algemeen, een beter inzicht verwerven in het ontwerpen, dan zijn verschillende benaderingen mogelijk: literatuurstudie rond ontwerptheorieën en -modellen; het bedenken en op computer simuleren van plausibele ontwerpprocessen; interviews met en observaties van ontwerpers; of het opzetten van gecontroleerde experimenten [Lawson, 1994]. Hoewel, doorheen 40 jaar ontwerponderzoek, elk van deze methoden haar sporen verdiend heeft, lijkt elke methode op zich in één of meer

---

1 Case-Based Reasoning betekent letterlijk ‘redeneren op basis van casussen’.
2 In deze tekst zal hiervoor de Engelse afkorting AI (Artificial Intelligence) worden gebruikt.
3 Case-Based Design betekent letterlijk ‘ontwerpen op basis van casussen’.
4 In deze tekst zal voor casus de Engelse term case worden gebruikt.
opzichten ontoereikend. Daarom werd bewust geopteerd om de rol van cases in het architectuurstuderen op verschillende manieren en vanuit verschillende invalshoeken. Concreet bundelt de thesis stemmen van CBD- en andere ontwerponderzoekers, professionele architecten, ontwerpbegeleiders en architectuurstudenten, wat een completer en meer gediversifieerd beeld oplevert dan elke methode afzonderlijk.

Als gevolg van de optie voor meerdere benaderingen heeft de thesis een ietwat ongewone structuur. Hoofdstuk 1 introduceert de lezer in het specifieke en zeer complexe karakter van het architectuurstuderen. Deel I, bestaande uit hoofdstukken 2 tot 6, bevat een kritische evaluatie van de CBD-benadering binnen de context van de architectuur. Om de wortels van deze benadering te situeren, geeft hoofdstuk 2 een korte rondleiding in het domein van de AI, waarna hoofdstuk 3 een selectie van concrete CBD-systemen voor het architectuurstuderen onder de loep neemt. In hoofdstuk 4 beschrijven zes architecten, elk met aanzienlijke ervaring in de architectuurpraktijk en het ontwerponderwijs, hoe – en zelfs of – cases volgens hen worden (her-) gebruikt bij het (leren) concipiëren van architectuur. Om een beter inzicht te krijgen in dit leerproces, beschrijft hoofdstuk 5 een experimenteel onderzoek naar de effecten van het gebruik van cases op de ontwerpprestaties van architectuurstudenten. Hoofdstuk 6 confronteert de visies en resultaten uit de voorgaande hoofdstukken, om van daaruit de huidige kloof tussen het CBD-onderzoek en het architectuurstuderen te verklaren.

Deel II, bestaande uit de volgende drie hoofdstukken, gaat de uitdaging aan om deze kloof te (helpen) overbruggen. De brug heet DYNAMO, wat staat voor Dynamisch Architectuur-Memorie On-line, en bestaat uit een interactief, digitaal ontwerpinstrument opgebouwd rond concrete cases. Hoofdstuk 7 schetst de theoretische achtergrond van DYNAMO, waarna hoofdstuk 8 de implementatie van een werkend prototype beschrijft. De evaluatie van dit prototype door architectuurstudenten is het onderwerp van hoofdstuk 9. Een epiloog blikt tenslotte terug op ons onderzoek naar de rol van cases in het architectuurstuderen, formuleert conclusies, en zet mogelijke routes uit voor verder onderzoek.

**Hoofdstuk 1: Het concipiëren van architectuur**

Case-Based Reasoning behoort tot het domein van de kennistechnologie, een subdomein binnen het onderzoek naar Kunstmatige Intelligentie dat zich toelegt op de vraag: welke kennis is vereist voor het uitvoeren van intelligente taken? Dit hoofdstuk onderzoekt deze vraag voor één welbepaalde taak: het concipiëren van architectuur.

Eerst en vooral wordt aangetoond dat het architectuurstuderen kan worden beschouwd als een uiterst complexe, kennisintensieve activiteit. Paul Cilliers definieert complexe systemen als bestaande uit een groot aantal componenten, die op zich eenvoudig kunnen zijn, en dynamisch op elkaar inwerken door uitwisseling van informatie of energie [Cilliers, 1998]. Breidt men de term ‘systeem’ uit met activiteiten, taken, processen, kortom fenomenen, dan beantwoordt het architectuurstuderen duidelijk aan Cilliers’ definitie. Net als industrieel, interieur- en
SAMENVATTING

stedenbouwkundig ontwerpen, is het een complexe activiteit met zeer uiteenlopende ‘componenten’: akoestiek, ecologie, economie, ergonomie, grond- en andere mechanica, kleurenleer, materiaalkunde, projectbeheer, elementen uit de natuurkunde, psychologie en sociologie zijn allemaal op een of andere manier relevant voor het architectuuronterpen. Behalve talrijk en divers zijn de aspecten waarmee architecten rekening dienen te houden vaak tegenstrijdig en altijd nauw met elkaar verweven. Vandaar het belang van één of enkele krachtige ideeën, beelden of concepten, waarrond de diverse aspecten kunnen worden georganiseerd. Dergelijke ideeën helpen de complexiteit tijdens de conceptuele fase te beheersen doordat ze een kader aanreiken om alle aspecten te integreren in een coherente en zinvol ontwerp.


Vervolgens wordt nader ingegaan op de strategieën die architecten aanwenden om met deze complexiteit om te gaan: analogie, metafoor, heuristiek, typologie en (her-) gebruik van cases. Meer bepaald wordt aangetoond hoe deze strategieën ontwerp- en aanbrengen tijdens de fase van de conceptvorming. Tabel 1.1 geeft een overzicht van deze strategieën en de types van kennis die ze architecten kunnen bieden. Eerder dan als strikte classificatie is de tabel bedoeld als een poging om een rooster te leggen over een complex veld.

<table>
<thead>
<tr>
<th>Strategieën</th>
<th>Component</th>
<th>Concept</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. analogie</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. metafoor</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. heuristiek</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. type</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5. case</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TABEL 1.1. Ontwerpstrategieën voor de conceptuele fase en types kennis die ze aanbrengen. (We gebruiken grijstopen als een lineaire schaal van kwantitatieve waarden met donkergrijs als hoge kennisintensiteit, wit als geen kennisintensiteit en lichtgrijs als tussenwaarde.)

Wanneer architecten gebruik maken van analogie, nemen ze een bestaande vorm of vormgevend principe (een raster of proportiesysteem) als vertrekpunt voor hun ontwerp. Volgens sommige auteurs, wordt het gebruik van analogie gekenmerkt door een structurele (of syntactische) gelijkenis tussen ‘bron’ en ‘doel’, wat betekent dat de vorm van het doel rechtstreeks wordt beïnvloed door die van de bron [Casakin, 1997; Goldschmitt, 1994]. Bijgevolg leveren analogieën eerst en vooral componentkennis, meer bepaald kennis aangaande de component ‘vorm’. In tweede instantie kunnen
analogieën ook functioneren op conceptueel niveau. In architectuur is het concept dat de verschillende componenten tot een zinvol geheel integreren, immers niet noodzakelijk een extra ingrediënt. Het kan evengoed gaan om een component van de ontwerpsituatie zelf. Wordt deze integrerende rol gespeeld door de component ‘vorm’, dan kunnen analogieën tot op zekere hoogte ook worden beschouwd als leveranciers van conceptkennis.

Volgens deze redenering verschilt het gebruik van metaforen van dat van analogieën door de aard van de relatie tussen ‘bron’ en ‘doel’, die eerder semantisch dan syntactisch is. Aldo Rossi’s idee om een kerkhof te concipiëren als een stad voor de doden [Schreurs, 1986], bijvoorbeeld, heeft geen directe implicaties voor de vorm, noch voor de andere aspecten van het ontwerp, en toch worden alle aspecten erdoor beïnvloed. De metafoor biedt de architect een kader waarin alle aspecten op een zinvolle manier kunnen worden geïntegreerd. Bijgevolg lijkt het aannemelijk om metaforen te beschouwen als leveranciers van conceptkennis.

Heuristieken zijn daarentegen duidelijk actief op componentniveau. Ze veralgemenen voorgaande oplossingen, en leveren doorgaans kennis aangaande één specifiek aspect van het architectuurontwerpen (bv. klimaat of bouwmaterialen). Vermits veralgemeningen zelden gelden voor alle aspecten van een ontwerp, bestaan er bijzonder weinig heuristieken die meer dan één component tegelijk behandelen.

Ook types kunnen worden beschouwd als veralgemeningen van succesvolle ontwerpoplossingen uit het verleden. In plaats van vuistregels verschijnen deze veralgemeningen echter als architectuurobjecten, die kunnen fungeren als blauwdrukken voor toekomstige ontwerpen. Hoewel deze blauwdrukken zowel elk aspect van de architectuur kunnen incorporeren, gaat hun aandacht doorgaans uit naar de ruimtelijke distributie van functies. Met andere woorden, types voorzien de architect eerst en vooral van componentkennis, meer bepaald van kennis aangaande de component ‘functie’. Sommige architecten wenden de ruimtelijke distributie van een type echter aan op conceptueel niveau, i.e. als middel om een diepere betekenis te geven aan hun ontwerp. In dit opzicht kunnen types ook gezien worden als leveranciers van conceptkennis.

Net als types bestaan cases – in de strikte CBD-betekenis van het woord – uit architectuurobjecten die kennis bevatten over voorgaande ontwerpoplossingen. Deze objecten zijn echter concreet en gedetailleerd, en kunnen bijgevolg kennis leveren over elke component van de architectuur, alsook over de integratie van deze componenten tot een coherent ontwerp. Een concreet project in al zijn bijzonderheden biedt architecten een geïntegreerd beeld van ontwerpaspecten dat zou verloren gaan indien deze aspecten elk afzonderlijk werden aangesneden. Bijgevolg kan men cases beschouwen als leveranciers van zowel component- als conceptkennis.

Samenvattend kan men het concipiëren van architectuur omschrijven als een uiterst complexe taak, die hoge eisen stelt aan het kennen en kunnen van de architect. Het vereist actief kennen en passieve kennis op zowel concept- als componentniveau. Om aan deze eisen te voldoen beschikken architecten over een aantal strategieën, die een of meerdere van deze kennistypes aanreiken. Vanuit het standpunt van een architect die aan het ontwerpen is, blijkt hun voornaamste verdienste erin te bestaan dat ze een
stabil vertrekpunt bieden, een houvast te midden van het complexe web van aspecten, (programma-)onderdelen en eisen. Ze laten toe een of meer ‘componenten’ van het ontwerp tijdelijk te bevriezen en de aandacht te verschuiven naar de resterende topics, terwijl ze oriëntatie geven aan de oplossing ervan. De rest van de thesis gaat dieper in op één bepaalde strategie, het (her-)gebruiken van concrete cases uit het verleden.

Deel I: Een kritische studie van Case-Based Design in de architectuur

Hoofdstuk 2: Waarop is Case-Based Design gebaseerd?

Cases, i.e. concrete ontwerpervaringen uit het verleden, worden verondersteld architecten tijdens de complexe, kennisintensieve taak van de conceptvorming een rijke bron van ontwerpknowhow te bieden. Deze veronderstelling ligt aan de basis van een relatief jonge onderzoekstak, die bekend staat als Case-Based Design. Om de wortels van CBD te situeren, gaat dit hoofdstuk van start met een beknopt overzicht van het domein van de Kunstmatige Intelligentie. Vervolgens worden de theoretische grondslagen van Case-Based Reasoning besproken, alsook het cognitieve model gebouwd op deze grondslagen en de concrete toepassingen van dit model. Op die manier wordt stap voor stap ingezoomd op CBD en de nodige achtergrond opgebouwd voor de verdere studie ervan in het domein van de architectuur.

CBR werd ontwikkeld als theorie en technologie binnen het domein van de AI. Marvin Minsky omschrijft AI als het onderzoeksdomein dat betrokken is bij de ontwikkeling van computers voor taken waarvan men veronderstelt dat ze intelligentie vereisen [Minsky, 1985]. Grofweg kunnen AI-onderzoekers worden opgedeeld in twee groepen: in de ene groep beschouwt men dergelijke computers eerst en vooral als krachtige probleemoplossers, in de andere als een middel om meer inzicht te krijgen in de grondbegrinsen van (menselijke) intelligentie.

In beide groepen beperken onderzoekers hun ambitie veelal tot het oplossen van één van de deelproblemen die de ontwikkeling van artificieel intelligent gedrag met zich meebrengt. Het sub-domein van de kennis- en technologie legt zich bijvoorbeeld toe op de vraag welke kennis de uitvoering van een intelligente taak vereist, en hoe deze kennis kan worden vertaald naar computers. Diverse benaderingen werden hiertoe ontwikkeld en onderzocht, waarvan Case-Based Reasoning er één is. In tegenstelling tot Rule- en Model-Based Reasoning5 vertrekt CBR van een alternatieve kijk op hoe de mens redeneert. Eerder dan het linken van abstracte stukjes kennis, wordt redeneren beschouwd als een proces waarbij men zich één of een beperkt aantal concrete ervaringen herinnert, en beslissingen neemt op basis van vergelijkingen tussen de huidige en voorgaande situaties [Kolodner, 1993].


5 Letterlijk vertaald betekent Rule-Based Reasoning en Model-Based Reasoning respectievelijk ‘redeneren op basis van regels’ en ‘redeneren op basis van modellen’.
ruw geheugen’ slaat op alle ervaringen die men heeft meegemaakt of horen vertellen. In tegenstelling tot de algemene regels die eruit kunnen worden afgeleid, bevat het ruwe geheugen deze ervaringen in hun oorspronkelijke staat. Telkens wanneer ervaringen en kennisstructuren worden opgeroepen en gebruikt, krijgt het geheugen de gelegenheid de bijhorende kennis uit te testen. Dit laat het geheugen toe zichzelf te herorganiseren en -definieren, met andere woorden, te leren uit zijn ervaringen, vanwaar dan ook de term dynamisch geheugen [Kolodner, 1993].

Uitgaande van observaties van het menselijk geheugen werd een cognitief model ontwikkeld om deze observaties te verklaren. Volgens dit model bestaat kennis in het geheugen zowel uit specifieke ervaringen (cases) als uit veralgemeningen van specifieke ervaringen, en zijn beide vormen van kennis op dezelfde manier georganiseerd. Voor het denken op basis van cases worden twee macro-processen voorgesteld: herinnering en manipulatie. Herinnering staat voor het opzoeken van een ervaring uit het verleden die iets gemeenschappelijk heeft met de huidige situatie. Manipulatie kan bestaan in het aanpassen van één enkele case aan die huidige situatie of het versmelten van meerdere cases. Een essentieel kenmerk van het cognitief model is ten slotte dat leren op natuurlijke wijze geïntegreerd is met andere denkprocessen. De meest eenvoudige manier waarop het geheugen leert is door nieuwe ervaringen te vergaren. Een tweede manier is het herinterpreteren, c.q. herindexeren van voorgaande ervaringen. Tenslotte verandert het geheugen eveneens door het creëren van nieuwe veralgemeningen. Overeenkomstig de twee groepen van AI-onderzoekers – de ‘ingenieurs’ en de ‘psychologen’ – vervult het cognitief model van CBR een dubbele rol. Het fungeert als methodologie voor het bouwen van intelligente systemen en vormt de basis voor het ontwikkelen van een beter inzicht in hoe mensen cases aanwenden bij de uitvoering van intelligente taken.

De CBR-aanpak lijkt vooral geschikt voor zogenaamde ‘zwakke-theoriedomeinen’ [Kolodner, 1993], een label dat volgens CBR-onderzoekers duidelijk van toepassing is op het architectuurontwerpen. Aangezien problemen in dit domein vaag omschreven zijn en tijdens het ontwerproces herhaaldelijk opnieuw geformuleerd worden, zijn een vast stel regels of een causaal model ongeschikt voor het oplossen ervan. Vandaar de veronderstelling dat architecten bij het ontwerpen in belangrijke mate steunen op ontwerpervaring uit het verleden. Vooral tijdens de conceptvorming zouden voorgaande cases de stof leveren voor een aantal cruciale beslissingen [Domeshek & Kolodner, 1997]. Op zoek naar een geschikt concept, zo wordt verondersteld, tasten architecten hun geheugen af naar relevante ontwerpervaringen uit het verleden, en wenden deze aan in de huidige ontwerpsituatie.

Deze vooronderstelling ligt aan de basis van diverse CBD-systemen, i.e. CBR-toepassingen voor het architectuurontwerpen, die zowel voor de praktijk als het onderwijs in dit domein heel wat voordelen bieden: 1) In tegenstelling tot regels en modellen liggen concrete cases als het ware voor het rapen. 2) Aangezien ontwerpproblemen moeilijk te ontdelen zijn, kan een case de ‘lijm’ leveren die een ontwerpoplossing samenhoudt [Kolodner, 1993]. 3) CBD-systemen laten toe de efficiëntie van het ontwerproces te verhogen. 4) Bovendien vullen ze het geheugen van de architect aan door cases te bewaren en te herinneren die hij zelf misschien niet
Hoofdstuk 3: Een case base\textsuperscript{6} van Case-Based Design tools voor architectuur

In dit hoofdstuk worden zeven concrete CBD-projecten kritisch onder de loep genomen, met name Archie-II, CADRE, FABEL, IDIOM, PRECEDENTS, SEED en WEBPAD. De bestudeerde systemen zijn specifiek ontwikkeld voor het domein van de architectuur en geven samen een behoorlijk overzicht van de diversiteit binnen het huidige CBD-onderzoek. Ze vertegenwoordigen onderzoek door teams van verschillende aard en omvang, in verschillende contexten, en in verschillende fasen van ontwikkeling en implementatie.

Elke case-study start met een beknopte omschrijving van de hoofddoelstellingen en focus van het CBD-project, en beschrijft vervolgens hoe (en of) het systeem de verschillende ingrediënten van het CBR-recept implementeert. Aan het einde van elke case-study wordt een aanzet gegeven tot discussie, die in meer algemene termen wordt voortgezet in hoofdstuk 6.

De ‘case base’ van CBD-systemen is niet compleet of definitief, aangezien de systemen die ze bevatten verder ontwikkeld (kunnen) worden en er nieuwe systemen op het CBD-toneel zullen verschijnen. Eerder dan een uitvoerige catalogus biedt dit hoofdstuk een momentopname van het huidige onderzoek naar CBD in de architectuur.

Archie-II is een Case-Based Design Aid (CBDA)\textsuperscript{7} ontwikkeld aan het AI lab van Georgia Tech’s College of Computing in samenwerking met Tech’s College of Architecture. Het systeem bouwt voort op een voorgaande CBDA genaamd Archie, één van de allereerste CBR-toepassingen voor het architectuurontwerpen. Archie-II ondersteunt architecten tijdens de vroege, conceptuele fase van het ontwerpen, meer bepaald bij het ontwerpen van publieke gebouwen, door hen te voorzien van interessante cases uit het verleden. Aangezien de case base open staat voor zowel successen als mislukkingen, vervullen cases in Archie-II diverse rollen: sommige suggereren oplossingen, andere waarschuwen voor potentiële valkuilen, nog andere suggereren evaluatiecriteria. Het systeem legt zich toe op de representatie en het opzoeken van cases en laat manipulatie volledig over aan de gebruiker. Het succes van Archie-II inspireerde de ontwikkeling van DesignMuse, een platform voor het bouwen van CBDAs voor andere complexe ontwerpdisciplines.

CADRE – wat, naargelang de bron, staat voor Case-based spatial Design REasoning of Case-based building design through Dimensionality REduction – is een CBD-systeem voor de schetsontwerpfase. Het gaat om een interdisciplinair project

\textsuperscript{6} Letterlijk ‘een basis van casussen’.
\textsuperscript{7} Letterlijk ‘een ontwerphulp op basis van gevallen’.
met onderzoekers van Architektur und CAAD (ETH Zürich), Steel Structures en het AI lab (beide aan de EPF Lausanne). De klemtoon van dit onderzoek ligt op de aanpassing van cases aan nieuwe omgevingen en op de integratie van interdisciplinaire ontwerpcriteria. De casestudy bespreekt één specifiek ontwerpprobleem waarmee het systeem getest werd, namelijk het aanpassen van een bestaand woningontwerp aan een licht afwijkende site.

**FABEL** is een gemeenschappelijk project uitgevoerd door een consortium van zes partners onder leiding van het German National Research Centre for Information Technology (GMD). Hoofddoel is het ondersteunen van architecten en bouwkundig ingenieurs bij het plannen van gebouwen met complexe installaties. Hoewel FABEL diverse AI-benaderingen combineert, en dus eerder een hybride systeem is, was CBR het centrale paradigma voor de specificatie en implementatie van het systeem. FABEL is opgevat als een verzameling van diverse tools en methodes, zogenaamde specialisten, die elk een bepaalde stap van de CBR-cyclus voor hun rekening nemen. Ze werken samen op een virtuele bouwsite zodat elke architect of ingenieur kan zien welke specialist op welke plaats van het gebouwontwerp actief is. Specialisten zijn relatief onafhankelijk en hebben elk een eigen case base, case representatie en geheugenorganisatie.

**IDIOM** – Interactive Design using Intelligent Objects and Models – is een CBD-systeem voor het ontwerpen van gebouwenlay-outs. Het systeem werd ontwikkeld om de interactie tussen mens en computer nader te onderzoeken, alsook om het gebruik van voorkeuren en modellen bij de aanpassing en combinatie van cases uit te testen. IDIOM is de opvolger van CADRE, en heeft aanzienlijke vooruitgang geboekt ten opzichte van zijn voorganger. De belangrijkste verbetering is wellicht de toegenomen graad van interactie.

**PRECEDENTS** is een CBDA voor het architectuurontwerpen ontwikkeld door Rivka en Robert Oxman. Het systeem stockeert memorabele ontwerpen die beschouwd worden als precedents, i.e. erkende schoolvoorbeelden van een bepaalde ontwerpstijl of type, en maakt deze toegankelijk voor architectuurstudenten. PRECEDENTS concentreert zich op de ruimtelijke organisatie van musea in de vroege, conceptuele fase van het ontwerpproces. In deze fase bestaat de rol van cases erin de (menselijke) ontwerper zinvolle concepten aan te reiken voor het ontwerp van een museum. De klemtoon van het systeem ligt op de representatie van cases, meer bepaald op het representeren van de conceptkennis die ingebed zit in voorgaande ontwerpen, en op het toegankelijk maken van deze kennis in een gecomputeriseerde bibliotheek van precedents.

**SEED** – Software Environment to support Early building Design – is een hybride ontwerpmogeving die elementen uit Case- en Model-Based Design combineert. Tijdens de vroege fasen van het ontwerpen biedt het systeem ondersteuning voor de analyse, evaluatie, maar vooral de snelle representatie van een gebouwontwerp. Hiertoe is SEED opgebouwd uit een verzameling modules, die elk een specifieke taak in het ontwerpproces op zich nemen. Tot nu toe zijn – of worden – enkel modules ontwikkeld voor het opstellen van programma’s, het ontwerpen van schematische lay-outs en het genereren van 3D configuraties. Alle SEED modules modelleren
ontwerpproblemen en -oplossingen op een gelijkaardige manier, maar kunnen interne representaties of datamodels kiezen naar gelang hun individuele behoeften. De focus van deze casestudy is SEED-Layout, een module voor het genereren van schematische lay-outs voor de functionele eenheden van een programma.

WEBPAD is een ontwerpinstrument dat CBD-, CAD- en Web-technologie integreert. Hoofddoel is architecten te voorzien van relevante ontwerpen uit een gedistribueerde bron van grafische CAD-files op het Web. WEBPAD wordt ontwikkeld door dezelfde onderzoeksgroep als PRECEDENTS en is nog volop in ontwikkeling.

Welke ingrediënten van het cognitief model van CBD vindt men in de bestudeerde systemen terug? Wat betreft de kennisstructuren en -organisatie van het model, bevatten alle systemen specifieke ervaringen, c.q. cases. Deze cases zijn geïmplementeerd als ontwerpproducten op diverse schalen, gaande van volledige gebouwen over lay-outs tot ruimtes of componenten. Om deze cases beter te kunnen hanteren, worden ze verder opgedeeld in verhalen (stories), specifieke representaties of componenten.

Naast specifieke ervaringen, veronderstelt het model dat het geheugen diverse soorten algemene kennis bevat. Veralgemeningen vindt men terug in de vorm van richtlijnen, ‘high-level’ concepten, objectmodellen of classificaties, adaptatiekennis in de vorm van beperkingen (constraints), regels, grammatica’s of domeinmodellen.

Volgens het cognitief model zijn specifieke en algemene kennis op de zelfde manier georganiseerd. Strikt genomen vindt men dergelijke organisatie alleen terug in Archie-II, waar verhalen en richtlijnen verbonden zijn door een gemeenschappelijk set indices. Een lichtje verschillende aanpak is die van PRECEDENTS, waar verhalen geordend zijn rond een gesofisticeerd Issue-Concept-Formalisme (ICF) formalisme. Op die manier dienen veralgemeningen, i.e. ‘high-level’ concepten, eigenlijk om specifieke ervaringen te organiseren. In SEED, daarentegen, is de band tussen specifieke en algemene kennis heel wat zwakker. Het systeem bevat drie parallelle ‘bases’ die duidelijk gescheiden zijn: een object database bewaart het algemene objectmodel, een kennisbank bevat diverse classificaties, c.q. veralgemeningen, terwijl de case base gevuld is met specifieke cases.

Wanneer zich een nieuw ontwerpprobleem voordoet, zijn er volgens het model verschillende redeneerprocessen om deze kennis te gebruiken. Eerst en vooral herinnert het geheugen zich de meest relevante kennis om het probleem op te lossen. In de meeste CBD-systemen is dit herinneren geïmplementeerd als het opzoeken van cases die één of meer kenmerken gemeenschappelijk hebben met de nieuwe situatie. Hiertoe zijn cases in Archie-II, bijvoorbeeld, gelabeld met descriptieve en relationele indices, waarmee kenmerken van de situatie kunnen worden vergeleken. In FABEL, waar gebruikers de keuze hebben tussen vier verschillende ‘opzoekteams’, variëren de kenmerken van functionele componenten over bepaalde lay-out patronen, tot structuur en topologie. CADRE biedt nauwelijks ondersteuning bij het herinneren en concentreert zich volledig op manipulatie.

Wanneer het erop aankomt de ‘herinnerde’ cases te manipuleren, zit de taak van Archie-II, PRECEDENTS en WEBPAD erop. Deze drie systemen hebben ervoor
geopteerd dit proces volledig aan de gebruiker over te laten. CADRE, FABEL, IDIOM en SEED-Layout, daarentegen, bieden ondersteuning bij structurele aanpassing. Ze wijzigen de dimensies en/of topologie van een gegeven ontwerp oplossing totdat deze past in de nieuwe situatie. SEED-Layout past niet enkel de oplossing aan, maar ook de probleemstelling. Wat betreft aanpassing door afleiding (derivation adaptation) en versmelting blijft er nog heel wat werk te doen, aangezien weinig systemen deze vormen van manipulatie hebben geïmplementeerd.

Het *leren* biedt nog meer onontgonnen terrein. Centraal in CBD’s cognitief model staat de bewering dat het geheugen zijn prestatie dynamisch verbetert telkens het gebruikt wordt – door het opslaan van nieuwe cases, herindexeren of creëren van nieuwe veralgemeningen. In de bestudeerde CBD-systemen komt dit dynamisch leren nauwelijks aan bod. Het meest leergerige systeem is misschien nog CADRE, aangezien het beschikt over een pre-modeller Mod-4 om het verwerven van nieuwe cases te vergemakkelijken. WEBPAD zou nieuwe cases verwerven door CAD-bestanden toegankelijk te maken over het Internet. Hoe deze bestanden gecompileerd worden in een project-, tekening- en componentlaag, en hoe ze geïndexeerd worden volgens het ICF-formalisme, is echter nog niet duidelijk.

Samenvattend kan men stellen dat drie ingrediënten van het cognitief model goed vertegenwoordigd zijn in de bestudeerde CBD-systemen zijn, met name het gebruik van specifieke ervaringen, het opzoeken van cases en structurele aanpassing. De combinatie van specifieke met algemene kennis en de uniforme organisatie van beide soorten kennis zijn minder goed vertegenwoordigd, terwijl aanpassen door afleiding, verkrijgen en leren quasi afwezig zijn.

**Hoofdstuk 4: Zijn architecten van nature ‘Case-Based Designers’?**

*Architecten aan het woord*

Wil men een beter inzicht krijgen in de rol van cases bij het concipiëren van architectuur, dan heeft men de keuze uit verschillende methodes. Eén mogelijkheid bestaat erin de taak van de conceptvorming te analyseren en rollen te bedenken die cases wellicht bij deze taak vervullen. Het grof van het CBD-onderzoek dat werd voorgesteld in hoofdstuk 2, wordt in belangrijke mate gekenmerkt door deze eerste benadering. Heel wat CBD-systemen die aan bod kwamen in hoofdstuk 3, lijken gebaseerd op hoe architecten verondersteld worden cases te gebruiken, eerder dan op enige evidentie dat ze het daadwerkelijk op die manier doen. Hoewel deze methode vaak voorwerp is van kritiek, is één van haar verdiensten dat ze het onderwerp nauwkeurig definiëert en de aandacht vestigt op fundamentele vragen omtrent conceptvorming [Lawson, 1994].

In een poging om deze vragen te beantwoorden opteert hoofdstuk 4 voor een andere aanpak: architecten worden gevraagd om te beschrijven hoe – en zelfs of – ze gebruik maken van voorgaande cases bij het concipiëren van architectuur. Ook deze aanpak is

---

*Letterlijk: ‘ontwerpers die vertrekken van casussen’.*
echter voorwerp van kritiek. Zoals elke professional die zijn diensten op de markt moet verkopen, geven architecten wellicht niet altijd een eerlijke beschrijving van hun ontwerpproces, bang als ze zijn om ‘het geheim van hun succes’ prijs te geven [Lawson, 1994; Eastman, 1999]. Bovendien heeft het architectuurontwerpen een deels intuitief karakter, waardoor architecten zich niet altijd ten volle bewust zijn van hoe ze precies te werk gaan. En zelfs als ze dat waren, dan nog zouden weinigen in staat zijn dit bewustzijn nauwgezet te verwoorden [Lawson, 1994].

Ondanks deze potentiële valkuilen werd toch geopteerd voor diepte-interviews met zes architecten, die heel wat ervaring hebben met de praktijk alsook met het onderwijs van het architectuurontwerpen. Het hoofdstuk begint met een korte beschrijving van hoe de interviews werden afgenomen, en introduceert elke architect aan de hand van een beknopte biografie. Vervolgens worden verschillende aspecten behandeld die het (her-)gebruik van cases in het échte architectuurontwerpen kenmerken.

Hoewel de geïnterviewden verschillende generaties, benaderingen en visies op het architectuurontwerpen vertegenwoordigen, bleken ze behoorlijk consistent in hun antwoorden met betrekking tot de praktijk. Bijzonder opvallend was hun ontegensprekelijke en nagenoeg unanieme ontkenning dat cases een rol zouden spelen bij de conceptvorming. Hoewel het onderwerp op een neutrale toon werd aangebracht, associeerden de meeste geïnterviewden dergelijke rol quasi automatisch met wanpraktijken. Dat men van cases kan leren, en dat er niks mis is met het gebruik ervan tijdens de conceptuele fase van het ontwerpen, leek niet bij hen op te komen. Integendeel, uit hun antwoorden viel een algemene afkeuring af te leiden van referenties naar voorgaande projecten, vooral van referenties naar projecten ontworpen door andere architecten.

In de loop van de discussie zagen de geïnterviewden zich genoodzaakt hun eerste ontkenning te herzien, hoewel de meeste ‘bekentenissen’ van (her-)gebruik van cases meteen goedgepraat werden met excuses als “iedereen doet het” of “Niet vaak, hoor” – alsof men zich zou moeten schamen voor het leren uit voorgaande projecten. Tussen deze excuses door kwamen echter verschillende ingrediënten van het CBD-recept ter sprake (bv. het opzoeken van cases, structurele aanpassing, aanpassing door afleiding, het verwerven van nieuwe cases of het herindexeren van reeds verworven cases), zij het in een meer subtiele en realistische versie dan CBD-onderzoekers doorgaans veronderstellen. In andere fasen van het ontwerpproces dan de conceptvorming bleek het raadplegen van cases minder taboe.

Wat betreft de rol van cases in het ontwerponderwijs, waren de meningen meer verdeeld. Alle geïnterviewden vonden het evident dat architectuurstudenten te rade gaan bij bestaande projecten, maar wanneer het erop aankwam dit te stimuleren of ondersteunen, vielen ze plots in twee kampen uiteen. Voor ontwerpbegeleiders uit het ene kamp mocht het gebruik van cases door studenten dan al vergeeflijk zijn, het viel absoluut niet toe te juichen, laat staan aan te moedigen. De tegenpartij, daarentegen, was ervan overtuigd dat architectuurstudenten baat zouden vinden bij dergelijke aanmoediging. Wat beide kampen echter gemeenschappelijk hadden, was het feit dat hun overtuiging gebaseerd leek op vooroordelen en intuïtie, eerder dan op enige evidentie over de impact van cases op architectuurstudenten. Hoewel CBD beweert te
vertrekken van hoe ontwerpers (leren) ontwerpen, blijkt de rol van cases in dit leerproces – althans wat het domein van de architectuur betreft – verre van duidelijk. Daarom werd in het volgende hoofdstuk een experiment opgezet rond de effecten van cases op de ontwerpprestaties van studenten.

**Hoofdstuk 5: Doen cases ertoe?**

 Dit hoofdstuk brengt verslag uit van een experimenteel onderzoek naar het gebruik van cases, i.e. concrete voorbeelden uit het verleden, in het architectuuronderwijs. Aanleiding voor dit onderzoek zijn de radicaal tegengestelde visies van CBD-onderzoekers enerzijds en een deel van de onderwijsstaf anderzijds. Het CBD-onderzoek vertrekt van de veronderstelling dat cases voor architecten een significante bron van inspiratie en informatie vormen, vooral bij de conceptvorming. Studenten blootstellen aan cases tijdens het ontwerpproces zou daarom leiden tot ontwerpproducten van hogere kwaliteit. Heel wat ontwerpbegeleiders gaan er echter van uit dat dergelijke blootstelling creatieve ontwerpoplossingen in de weg staat, aangezien het gevaar voor ontwerpfixatie hierdoor zou toenemen. Doel van het experiment was daarom te onderzoeken of cases de kwaliteit en creativiteit van de studenten-projecten beperkt dan wel verhoogt. Hiertoe werden architectuurstudenten van het 2de jaar gevraagd een inkomhal voor een appartementsgebouw te ontwerpen. Tijdens het ontwerpproces kon de helft van de studenten kwalitatief hoogstaande inkomhalprojecten uit vorige jaren raadplegen, de andere helft niet.

Analyse van de studenten-projecten bracht enkele interessante effecten aan het licht. Blootstelling aan cases blijkt een positief effect te hebben op de scores die de studenten-projecten kregen van onafhankelijke beoordelaars. Wanneer studenten kwalitatief hoogstaande voorbeelden uit voorgaande jaren te zien krijgen, wordt de kans op hogere scores voor het concept en de focus van de opgave (in dit geval de kleur- en materiaalkiezen) positief beïnvloed. Bovendien blijken deze voorbeelden een positief effect te hebben op de creativiteit van de studenten-ontwerpen. De drie effecten zijn echter in sterke mate afhankelijk van de bekwaamheid en motivatie van de student.

Deze resultaten hebben wellicht belangrijke implicaties voor de ontwerpers en gebruikers van CBD-tools in het architectuuronderwijs. Eerst en vooral is het hoogst onwaarschijnlijk dat studenten die dergelijke tools gebruiken, het risico lopen beperkt te worden in hun creativiteit, vermits geen sporen van ontwerpfixatie werden gevonden. Integendeel, de voorbeelden in de case base kunnen de studenten ondersteunen bij het verbeteren van de kwaliteit en het innovatieve karakter van hun ontwerp.

Deze ondersteuning blijkt echter effectiever bij meer bekwame en gemotiveerde studenten. Wil men meer effectieve CBD-tools ontwikkelen, i.e. effectief voor *alle* student-ontwerpers, dan zal het bijgevolg niet volstaan om studenten eenvoudigweg een case base gevuld met voorbeelden aan te bieden. Deze voorbeelden dienen op zo’n manier georganiseerd en gerepresenteerd dat ze het gebrek aan achtergrond en motivatie bij zwakker studenten helpen opvangen. De grootste uitdaging bestaat erin
SAMENVATTING

om studenten te motiveren om projecten ontworpen door anderen grondig te bestuderen. Pas dan zal men kunnen spreken van een CBD-tool die architectuurstudenten daadwerkelijk ondersteunt tijdens het ontwerpen.

Indien deze resultaten kunnen worden bevestigd door bijkomende experimenten, kan men hieruit bovendien lessen trekken voor het CBR- en AI-onderzoek in het algemeen. Volgens ontwerpegeleiders zijn bekwame studenten, i.e. studenten die goed scoren in het eerste jaar, vaak studenten die reeds een zekere achtergrond hebben opgebouwd. Hoewel deze achtergrond het domein van de architectuur overstijgt, bevat hij allicht ook cases uit het architectuurontwerpen. Indien de bekwaamheid van studenten kan worden beschouwd als een maat voor deze achtergrond, dan lijkt de invloed van deze variabele te wijzen op de noodzaak van een embryonale case base om de CBR-cyclus op gang te brengen. Hierbij kan men zich vragen stellen omtrent de inhoud van deze case base, met andere woorden omtrent het aantal, de variëteit en de kwaliteit van de cases die noodzakelijk zijn vooraleer een Case-Based Reasoner aan de slag kan.

Het is echter nog te vroeg voor dergelijke speculaties op basis van één enkel experiment. Bijkomend bewijsmateriaal is nodig dat bekwaamheid/achtergrond en motivatie inderdaad noodzakelijke voorwaarden zijn voor succesvol CBD. In het experiment vormden de studenten een continuïum van uitstekende over middelmatige tot zwakke ontwerpprestaties in het eerste jaar. Een betere strategie om de invloed van de bekwaamheid/achtergrond van de student te bepalen is misschien om het experiment te herhalen met een mix van uitsluitend uitstekende en zwakke studenten. Bovendien zou het interessant zijn om na te gaan of expert architecten, gezien hun bekwaamheid en achtergrond, nog meer baat vinden bij voorbeelden dan studenten.

In afwachting van de resultaten van bijkomend onderzoek kunnen ontwerpegeleiders zich alvast beginnen voorbereiden op een grondige mentaliteitswijziging. Volgens onze studie zorgt hun nadruk op originaliteit en angst voor ontwerpfixatie er momenteel voor dat voorbeelden onterecht uit het ontwerponderwijs worden geweerd.

Hoofdstuk 6: Case-Based Design in architectuur

Een kritische reflectie

In de voorgaande hoofdstukken werd de rol van cases bestudeerd op verschillende manieren en vanuit diverse invalshoeken. Dit hoofdstuk tracht enkele sleutelelementen die hierbij aan bod kwamen scherper in beeld te brengen door de verschillende standpunten te confronteren. Deze confrontatie vertrekt van de vaststelling dat de verwachtingen van de CBD-pioniers vandaag verre van ingelost zijn. Ondanks ernstige pogingen om betrouwbare CBD-tools te ontwikkelen, laat een overtuigende doorbraak van CBD in het domein van de architectuur nog steeds op zich wachten. Men zou kunnen aanvoeren dat het CBD-onderzoek nog in de kinderschoenen staat, maar het uitgebreide en succesvolle gebruik van CBR-toepassingen in andere domeinen en disciplines lijkt dit argument te weerleggen. Meer fundamentele
oorzaken blijken aan de basis te liggen van het tot nu mislukte huwelijk tussen CBD en architectuur. Op basis van de elementen aangevoerd in de voorgaande hoofdstukken zijn er oorzaken te vinden op minstens drie niveaus.

Het eerste niveau heeft betrekking op het achterliggend *cognitief model* van CBD, waarvan de toepassing op architectuur fundamentele problemen stelt. Eerst en vooral bestaat er verwarring rond de hoofdrolspeler in dit model: ontwerpervaring. Hoewel dit begrip op diverse wijzen kan worden geïnterpreteerd, gebruiken de meeste CBD-onderzoekers deze term uitsluitend voor een ontwerpproduct. Een eerste element van kritiek is daarom het misleidende beeld van ontwerpervaring dat door het CBD-onderzoek opgehangen wordt. Het weze duidelijk dat ontwerpproducten bijdragen tot het verwerven van ontwerpervaring. Indien architecten leren ontwerpen door ervaring, dan houdt dit leren echter niet alleen blootstelling aan ontwerpproducten in, maar ook actieve deelname aan ontwerpprocessen.

Een tweede element van het cognitief model dat niet perfect overeenstemt met de werkelijkheid van het architectuurontwerpen, is het opzoeken van een relevante case. CBD-onderzoekers stellen relevantie gemakshalve gelijk aan gelijkenis, met andere woorden, de meest relevante case is degene die de meeste eigenschappen gemeen heeft met de nieuwe ontwerpsituatie. Tijdens de interviews in hoofdstuk 4 waren echter weinig architecten in staat deze gelijkstelling te bevestigen. Niet alleen de motivatie om een bepaalde case te selecteren, maar ook het moment waarop de selectie gebeurt, blijkt grondig te verschillen van wat het CBD-model veronderstelt.

Een meer fundamenteel punt van kritiek heeft te maken met de visie op het ontwerpen die de CBD-benadering impliceert. Het idee zelf om CBR toe te passen op het ontwerpen vertrekt van de premisse dat ontwerpen neerkomt op probleem-oplossend denken. Hoewel deze visie 40 jaar geleden behoorlijk populair was, werd ze na een korte periode van grenzeloos enthousiasme ernstig in vraag gesteld, zowel in de architectuur als in andere ontwerpdisciplines [Cross, 1977; Neuckermans, 1975]. Om diverse redenen wordt het ontwerpen gezien, bestudeerd en ontwikkeld als een uiterst complexe activiteit met eigen, unieke karakteristieken. In dit opzicht is het bijzonder jammer dat CBD de ‘probleematische’ visie op het ontwerpen nieuw leven inblaast.

Ten tweede, op het niveau van de *implementatie* slagen weinig tools erin om de consequenties van CBD’s cognitief model ten volle waar te maken, wat vaak leidt tot een ernstige reductie van de CBD-aanpak en/of van het architectuurontwerpen. Hoewel het cognitief model het belang van concreetheid verdedigt, blijken CBD-systemen paradoxaal genoeg een voorkeur te hebben voor abstractie. Om redenen van implementatie worden concrete ontwerpen uit hun context geplukt en gereduceerd tot abstracte, voor computers verstaanbare formalismen. Ook de integratie van verschillende aspecten – de ‘lijm’ – gaat vaak verloren bij het implementeren van CBD-systemen. Tenslotte blijken deze systemen zelden recht te doen aan Schank’s Theorie van het Dynamische Geheugen. In tegenstelling tot de oorspronkelijke verwachtingen, kunnen ze zelden worden geüpdate door nieuwe cases toe te voegen of reeds aanwezige cases te herindexeren.

Niveau drie bekijkt CBD vanuit een breder perspectief en werpt een blik op de context waarin CBD-tools gebruikt zullen worden, meer bepaald de context van het
ontwerponderwijs. Hoewel deze context niet onmiddellijk met de tools zelf te maken heeft, mag zijn impact op hun (in-)efficiëntie niet worden onderschat. Ontwerpbegeleiders staan zelden te trappelen om computers te introduceren in het ontwerponderwijs [Neuckermans & Geebelen, 1999]. In het geval van CBD-systemen blijken sommigen onder hen nog minder enthousiast, aangezien dergelijke tools studenten zouden stimuleren om voorgaande projecten te kopiëren. Studenten zelf storen zich allicht minder aan het gebruik van computers op zich, maar blijken de bezorgdheid om originaliteit met hun begeleiders te delen. De vaststelling dat studenten effectief baat vinden bij het aanwenden van concrete cases noopt ons ertoe deze nadruk op originaliteit in vraag te stellen.

Samenvattend kan men stellen dat Case-Based Design bijdraagt tot een beter inzicht in de manier waarop architecten ontwerpkennis verwerven en aanwenden, maar beperkt is in belangrijke opzichten. Meer bepaald is het gebaseerd op een cognitief model dat het concipiëren van architectuur gelijkschakelt met het oplossen van problemen. Tools die resulteren uit CBD-onderzoek zijn als zodanig inconsistent met de realiteit van het architectuurontwerpen. Bovendien, wanneer het erop aankomt deze tools te implementeren, gaan fundamentele aspecten van dit cognitief model – concreetheid, integratie en dynamiek – verloren in de transfer van menselijk naar computer-geheugen. Tenslotte blijkt een aanzienlijk deel van de doelgroep last te hebben van vooroordelen, wat een overtuigende doorbraak van CBD in het domein van de architectuur allicht ernstig bemoeilijkt.

Een belangrijk obstakel voor een dergelijke doorbraak is blijkbaar verwarring: verwarring aan de zijde van CBD-onderzoekers over de betekenis van ontwerpervaring, de relevantie van cases en de graad van complexiteit van het architectuurontwerpen; verwarring aan de zijde van architecten, ontwerpbegeleiders en studenten over de impact van cases, en over wat creativiteit is en niet is. Eerder dan een randprobleem lijkt deze verwarring een ernstige kritiek op het CBD-onderzoek en, bij uitbreiding, het CAAD-onderzoek in het algemeen. Zolang onderzoekers blijven samenkomen op conferenties om de nieuwste algoritmen voor het opzoeken en aanpassen van cases uit te wisselen, zullen architecten en architectuurstudenten hun werk wellicht blijven negeren. Deze negatie lijkt te wijzen op een kloof tussen het échte en het artificiële architectuurontwerpen. Uiteraard kan men besluiten dat om deze kloof te overbruggen, bijkomend onderzoek noodzakelijk is. Maar misschien is er deze keer iets anders nodig. Willen CBD-tools het architectuurontwerpen daadwerkelijk ondersteunen, dan is er eerst en vooral behoefte aan een betere communicatie en samenwerking tussen de praktijk en het onderwijs.

Op dit punt kan men zich afvragen waarom de kloof tussen CBD en architectuur überhaupt moet worden overbrugd. Het antwoord luidt dat, hoewel de huidige CBD-tools niet beantwoorden aan de behoeften van de echte architectuur, er sterke aanwijzingen zijn om de CBD-aanpak niet in z’n geheel van tafel te vegen. Eén aanwijzing is het opmerkelijke succes van het ‘Little Brother’ (Kleine Broer) project [Leitch, 1995]. In dit project werden architectuurstudenten van het eerste jaar
gevraagd een studio te ontwerpen in de tuin van een woning ontworpen door een lokale architect. Hierbij konden ze beroep doen op een unieke combinatie van diverse bronnen van ontwerpervaring: (1) een ‘levende’ architect, die kon worden bevraagd over zijn bedoelingen tijdens het ontwerpproces; (2) ‘Grote Broer’, een concreet product ontworpen tijdens dit proces; en (3) de originele plannen en tekeningen van dit product. Strikt genomen heeft Little Brother niets te maken met CBD, maar het toont duidelijk aan dat een rijke combinatie van ontwerpervaringen kan bijdragen tot het ontwerponderwijs, zij het zonder de ondersteuning van computertechnologie.

Maar zelfs in de vorm van enkel ontwerpproducten, i.e. projecten, blijkt ontwerpervaring een waardevol instrument bij het architectuurontwerpen. De resultaten van ons experiment lijken erop te wijzen dat student-ontwerpers inderdaad voordeel halen uit de blootstelling aan concrete projecten. Dit voordeel mag dan al groter zijn voor bekwame en gemotiveerde studenten, het suggereert in elk geval dat er voor CBD een belangrijke rol in architectuurscholen is weggelegd.

Maken we de balans op, dan lijkt CBD geschikt om met de complexiteit van het architectuurontwerpen om te gaan, indien de benadering verder verfijnd, correct geïmplementeerd en onbevooroordeeld aangewend wordt. Cases zijn waardevolle instrumenten voor (student-)architecten, en het is zeer waarschijnlijk dat de huidige informatietechnologie dit instrumentarium nog kan verbeteren.

In het tweede deel van de thesis wordt gepoogd om de kloof tussen CBD en het architectuurontwerpen te overbruggen. De ideale brug zou alle hindernissen op de drie niveaus tegelijk elimineren. Vermits de constructie van dergelijke brug het bestek van één thesis overstijgt, werd ervoor geopteerd een selectie van hindernissen aan te pakken. Eerst en vooral wordt de uitdaging aangegaan om een tool te ontwikkelen die de gebruiker motiveert om zich grondig in cases te verdiepen, wat – volgens de resultaten van ons experiment – een noodzakelijke voorwaarde blijkt om voordeel te halen uit de CBD-aanpak. Op het niveau van implementatie werd ervoor gekozen om het gebrek aan dynamiek aan te pakken, aangezien het de kern vormt van de hele CBD-benadering. Om te anticiperen op mogelijke vooroordelen, wordt tenslotte een poging ondernomen om de gebruikers actief te betrekken bij de ontwikkeling van de tool. Hoewel de brug in eerste instantie bedoeld is voor het architectuuronderwijs, zijn de beschreven concepten eveneens bruikbaar voor de architectuurpraktijk en, bij uitbreiding, zelfs voor andere complexe ontwerp domeinen.

Deel II: Een brug tussen Case-Based Design en architectuur

Hoofdstuk 7: DYNAMO – een Dynamisch Architectuurgeheugen On-line

In deel I van deze thesis werd de rol van cases in het architectuurontwerpen belicht vanuit verschillende invalshoeken, en werd door deze invalshoeken te confronteren naar een verklaring gezocht voor het tot nu toe beperkte succes van CBD in het domein van de architectuur. Door een onnauwkeurig beeld van het gebruik van cases bij het architectuurontwerpen, in combinatie met problemen van implementatie, bieden CBD-systemen momenteel Weinig ondersteuning aan student- en professionele
architecten. Vooroordelen en angst voor ontwerpfixatie vergroten dit probleem vermits ze (student-)architecten ontmoedigen om cases aan te wenden tijdens het ontwerpen.

Deel II van de thesis is gewijd aan de ontwikkeling van een mogelijke strategie om de CBD-ondersteuning voor het architectuurontwerpen te verbeteren. De strategie luistert naar de naam DYNAMO – Dynamic Architectural Memory On-line – en bestaat uit een interactief, digitaal ontwerpinstrument waarin concrete cases centraal staan. Het instrument, dat deel uitmaakt van een geïntegreerde ontwerpomgeving, kan worden beschouwd als een CBD-systeem in zoverre dat het geïnspireerd is door CBD’s achterliggend cognitief model. In plaats van dit model zonder meer over te nemen, wordt echter gepoogd het aan te passen aan de complexe realiteit van het architectuurontwerpen.

Dit hoofdstuk schetst de ideeën die aan de basis liggen van DYNAMO, een instrument om (student-)architecten te ondersteunen bij de conceptvorming aan de hand van concrete cases. In eerste instantie is het instrument bedoeld voor het ontwerponderwijs, maar het kan eventueel worden uitgebreid naar de ontwerppraktijk. Eerst en vooral wordt de theoretische achtergrond van dit instrument geschetst, en wordt aangetoond hoe het cognitief model van CBD incorporeert en tezelfdertijd extrapoleert, door interactie te stimuleren en intensifiëren tussen gebruiker en computer, maar ook tussen ontwerpen onderling, tussen (student-)ontwerpers, en tussen de architectuurpraktijk en het ontwerponderwijs. Dit moet resulteren in een ontwerpinstrument dat cognitief comfortabel aanvoelt voor (student-)ontwerpers en tezelfdertijd voorwaarden schept voor vruchtbaar (her-)gebruik van cases in het architectuurontwerpen. Bovendien wordt het instrument gesitueerd in de context van IDEA+, een geïntegreerde ontwerpomgeving die architecten ondersteunt van bij het begin van het ontwerpproces. Tenslotte wordt aangetoond hoe DYNAMO bijdraagt tot het overbruggen van de kloof tussen het CBD-onderzoek en het architectuurontwerpen, met andere woorden, hoe het de problemen aanpakt die in het voorgaande hoofdstuk werden geïdentificeerd.

Onderzoekers uit de harde CBD-kern zullen DYNAMO wellicht niet beschouwen als een CBD-systeem, omdat het instrument zelf niet ‘redeneert’ op basis van cases. Vermits cases opaak blijven voor het systeem – DYNAMO’s kennis van de cases reikt niet verder dan hun indices – is het opzoeken van cases op basis van de waarden van hun indices het enige waartoe het in staat is. Of en hoe deze cases (her-)gebruikt worden voor nieuwe ontwerpopdrachten wordt volledig aan de gebruiker overgelaten. Toch valt niet te ontkennen dat het concept van het instrument geworteld is in de Theorie van het Dynamische Geheugen. Bovendien stimuleert het interactief (her-)indexeren het soort bewustzijn dat nodig is om voorgaande cases effectief in te schakelen tijdens het ontwerpen, namelijk een productieve mix van relevantie, vernieuwing en verschil [Liddament, 1998].

Niettemin is het doel van het onderzoek uiteraard niet om een CBD- of eender welk ander label te verdienen, maar om de rol van cases in de praktijk en het onderwijs van het architectuurontwerpen te verhelderen en te ondersteunen. De visie op het ontwerpen die CBD erop nahoudt, strookt niet met het toenemend bewustzijn dat het
architectuurontwerpen een complexe activiteit is en dat (inter-)actie een cruciale rol speelt bij de ontwikkeling en vernieuwing van ontwerpkennis. Om dit sociale aspect van het (leren) architectuurontwerpen te omvatten, werd geopteerd voor een extrapolatie van de CBD-aanpak naar meer dan één ontwerper, door interactie te stimuleren tussen architecten in verschillende contexten en op verschillende expertise-niveaus.

Hoofdstuk 8: Een werkend prototype

Implementatie


Vervolgens wordt het prototype zelf beschreven, dat bestaat uit drie grote componenten: een groeiende verzameling cases, een gegevensbank die deze verzameling structureert, en een gebruikersinterface om de verzameling te raadplegen en te wijzigen. Op dit moment is DYNAAMO een werkbaar prototype met meer dan 80 geïndexeerde cases van 53 verschillende architecten. Cases zijn gelabeld aan de hand van 17 soorten indices (vormelijke articulatie, ruimtelijke configuratie, huidig programma, constructiotype, etc.), die gegroepeerd zijn in 6 categorieën (concept, vorm & ruimte, functie, constructie, context en ID). De relatieve eenvoud van DYNAAMO’s caseregistratie, geheugenstructuur en interface was een bewuste keuze, gezien de prioriteit om in een korte tijdsspanne een functioneel systeem te produceren.

Na de simulatie van een mogelijk gebruiksscenario, wordt het prototype tenslotte gesitueerd ten opzichte van andere, vergelijkbare systemen, zowel binnen als buiten het domein van Case-Based Design. Beschouwt men de gelijkenissen en verschillen met de CBD-tools uit hoofdstuk 3, dan zou men DYNAAMO zelf kunnen beschouwen als een product van Case-Based Design. Tot op zekere hoogte is het prototype het resultaat van een poging om de sterkte punten van de bestudeerde systemen te combineren en hun zwakke punten te vermijden.

Om na te gaan of de resulterende case het bewaren waard is, werd het prototype geëvalueerd door architectuurstudenten. De hoofdbedoeling van deze evaluatie, die het onderwerp vormt van het volgende hoofdstuk, was te onderzoeken of DYNAAMO student-ontwerpers kan moitveren om zich te verdiepen in concrete cases. Aangezien het prototype bij deze evaluatie voor het eerst op grote schaal gebruikt werd, was het eveneens een gelegenheid om bugs op te sporen en te herstellen. Het systeem heeft dan ook grondige wijzigingen ondergaan naar aanleiding van deze evaluatie. Was
toegang tot de eerste versie hoofdzakelijk een voorrecht van Netscape-gebruikers in Windows-omgevingen, dan heeft het prototype sindsdien aanzienlijke vooruitgang geboekt wat betreft compatibiliteit. De meest recente versie is met succes uitgetest op verschillende browsers, machines en platforms, inclusief UNIX (SGI O2) en Apple (Apple PowerMac 7100 en 7200). Bovendien werd de geellersersinterface bijgeschaafd zodat mogelijkheden tot (inter-)actie – on-line discussiëren of extra informatie (cases, indices) toevoegen – meteen in het oog springen. Ondanks deze verbeteringen is en blijft DYNAMO een prototype. Door de relatieve eenvoud van de geheugenorganisatie, zal de snelheid van het programma allicht daalden naarmate de case base meer gevuld raakt. Bovendien is de interface om informatie te controleren en corrigeren nog niet ten volle ontwikkeld.

Hoofdstuk 9: DYNAMO in het atelier

Evaluatie

Dit hoofdstuk beschrijft hoe het prototype werd gebruikt en geëvalueerd door studenten van het vierde jaar Burgerlijk Ingenieur Architect. In de winter en lente van 1999 onderging DYNAMO zijn vuurdoop in een traditioneel ontwerpatelier rond het thema vernieuwbouw. Studenten kregen de opdracht ofwel een openbare bibliotheek onder te brengen in een voormalige industriehal, ofwel een uitbreiding te ontwerpen voor hun school, die zich bevindt in een zestiende-eeuws kasteel. Gedurende het gans project hadden alle deelnemende studenten toegang tot het DYNAMO-prototype, dat voor de gelegenheid aangevuld was met extra vernieuwbouwprojecten. De evaluatie beschreven in dit hoofdstuk tracht drie centrale vragen te beantwoorden: (1) Slaagt DYNAMO erin om studenten te overhalen om cases grondig te bestuderen? (2) Welke factoren werken hierbij stimulerend/belemmerend? (3) Heeft het gebruik van DYNAMO enige impact op de ontwerpprestatie van de studenten? Bij wijze van achtergrond voor deze evaluatie, wordt eerst de keuze voor een vernieuwbouwproject gemotiveerd. Na een beschrijving van de setting en de procedure van de evaluatie, wordt verslag uitgebracht van de analyses in verband met de drie centrale vragen.

Na de eerste, conceptuele fase van de ontwerpoevening, werd de studenten gevraagd een evaluatieformulier omtrent DYNAMO in te vullen. Ondanks het prototype- karakter van het systeem, blijkt uit hun antwoorden dat DYNAMO enthousiast onthaald werd. Van de 48 deelnemende studenten maakten 35 (73 %) effectief gebruik van het systeem. Op twee na wensen alle respondenten DYNAMO opnieuw te gebruiken bij ontwerpopdrachten in de toekomst. Er rijzen echter vragen over de manier waarop het systeem gebruikt werd. Geen enkele student die de het prototype raadpleegde, heeft de mogelijkheid benut om de case base te voeden, hetzij via on-line discussie, door een project toe te voegen, of door cases te her-indexeren. Voorlopig spreekt DYNAMO studenten aan omwille van de kwaliteit en kleuren van het beeldmateriaal en het feit dat het papier bespaart, eerder dan omwille van de mogelijkheden die het biedt om cases grondig te bestuderen. Wellicht kan dit probleem ten dele worden toegeschreven aan het strakke tijdsschema van de opgave.
Andere mogelijke verklaringen zijn het gebrek aan enthousiasme bij de ontwerpbegeleiders en aan computeruitrusting in het ontwerpatelier. Op het moment van de evaluatie was het atelier nog niet uitgerust met computers om DYNAMO te raadplegen, laat staan met scanners of digitale camera’s om informatie toe te voegen. Zolang studenten hun tekentafel – of zelfs het atelier – moeten verlaten om cases te raadplegen, zal de meerderheid het systeem niet gebruiken. Hoewel deze obstakels – gebrek aan tijd, aanmoediging vanwege de ontwerpbegeleiders en uitrusting in het atelier – niet rechtstreeks met DYNAMO zelf te maken hebben, hebben ze het systeem mogelijk verhinderd te functioneren zoals oorspronkelijk bedoeld. Studenten actief aanmoedigen om de case base te consulteren en voeden, en de nodige uitrusting hiertoe voorzien, zou DYNAMO’s vermogen om studenten te overhalen grondiger met cases te redeneren aanzienlijk verbeteren.


De analyses in verband met de tweede vraag, rond stimulerende en remmende factoren, leverden geen statistisch significante resultaten op. De mate waarin studenten gebruik maakten van het prototype blijkt niet te correleren met persoonlijke computeruitrusting, vertrouwdheid met het Internet, gebruik van CAD-software of niveau van computeralfabetisme. De vraag waarom de tool er niet in geslaagd is alle deelnemende studenten aan te spreken, blijft voorlopig onbeantwoord.

In verband met de derde vraag is er duidelijk reden tot optimisme. Wat het bibliothekproject betreft, suggereren de resultaten van onze analyses dat het gebruik van DYNAMO de ontwerpprestatie van studenten positief beïnvloedt. Bovendien werd geen statistische evidentie gevonden dat dergelijk gebruik het gevaar voor ontwerpfixatie zou verhogen. Integendeel, als DYNAMO enige impact op de creativiteit van de studenten kan worden toegeschreven, dan is het een positieve. Strikt genomen kunnen we nog geen uitspraken doen over de relatie tussen deze impact enerzijds en de bekwaamheid en motivatie van de studenten anderzijds. Wat het laatste betreft wijzen onze resultaten echter duidelijk in de goede richting, wat ons doet geloven dat verder onderzoek onze hypotheses zal bevestigen.

Er zijn echter valstrikken wanneer men CBD-systemen, inclusief DYNAMO, wil evalueren. Het belangrijkste aspect van DYNAMO is niet dat het louter cases presenteert, maar dat deze cases aanleiding geven tot grondige casestudies, reflectie stimuleren en discussies op gang brengen tussen studenten en ontwerpbegeleiders. Begeleiders moeten studenten helpen de relevantie van een bepaalde case te begrijpen door gepaste indices en verhelderende (on-line) toelichting. Ondanks het voornemen ontwerpbegeleiders actief te betrekken bij de ontwikkeling van het prototype, konden
SAMENVATTING

we hen nauwelijks overhalen om de tool te bekijken. Wat deze betrokkenheid betreft, zijn DYNAMO’s prestaties dan ook duidelijk ondermaats. Het is, bijgevolg, bijzonder moeilijk om de impact van het prototype juist te evalueren zolang we er niet in slagen om de gebruikers – studenten en ontwerpbegeleiders – op te waarderen van passieve informatie-consumenten naar actieve case-producenten.

Daarom besluit het hoofdstuk met het benadrukken van de nood aan een ruimer perspectief wanneer men CBD-tools evalueren. De effectieve performantie van een dergelijke tool hangt niet alleen af van de rijkdom, diversiteit en het aantal cases, of van de kracht van het representatiesysteem [Oxman & Shabo, 1999], maar eveneens van de manier waarop de tool wordt aangebracht door ontwerpbegeleiders, ervaren door studenten en ondersteund door de school. Wil men CBD-systemen ontwikkelen die (student-)architecten effectief ondersteunen bij het ontwerpen, dan mag men de invloed van de context waarin ze worden gebruikt niet onderschatten. Afgaande op de eerste ervaring met DYNAMO is deze context heel wat moeilijker te ‘dynamiseren’ dan de inhoud en structuur van een case base.

Epiloog


De generaal bedacht echter een eenvoudig plan. Ze verdeelde haar leger in kleine groepen en stuurde elke groep naar het begin van één weg. Zodra alles klaar was, gaf ze het signaal en marcheerde elke groep langs een andere weg naar de versterkte stad, zodat het ganse leger op hetzelfde moment bij de stad aankwam. Zo veroverde de generaal de versterkte stad en bracht ze de dictator ten val.”

Het verhaal van de generaal is, tot op zekere hoogte, het verhaal van deze thesis. De versterkte stad staat voor de rol van cases bij het concipiëren van architectuur. Die bevindt zich in het rijk van het architectuurontwerpen, omringd door analogieën, metaforen, heuristieken, types en proportiesystemen. Vele wegen leiden naar een beter inzicht in deze rol – theoretische reflectie, literatuurstudie, interviews met architecten, (min of meer) gecontroleerde experimenten, de ontwikkeling van een CBD-tool – maar elk van deze wegen benadert het fenomeen slechts van één kant en geeft dus slechts een beperkt beeld. Wanneer men te maken heeft met een complex fenomeen zoals het architectuurontwerpen, dan is het hoogst onwaarschijnlijk dat één enkele onderzoeksmethode het fenomeen volledige in beeld kan brengen [Cilliers, 1998].
Daarom werd in deze thesis ‘de onderzoeksmethode’ vervangen door iets wat beter afgestemd is op de complexiteit van het studieobject. Net als de generaal hebben we ervoor gekozen om verschillende invalswegen te nemen naar een beter inzicht in dit studieobject. In plaats van één enkele benadering werden diverse methoden gebruikt, wat toeliet op meerdere fronten vooruitgang te boeken, verschillende invalshoeken uit te testen en, misschien, het onderwerp beter te begrijpen. Eerder dan een systeem van zekerheden verschaffen deze methoden een list om de rol van cases in het architectuurontwerpen te verschalken [Neutjens, 1984].

Om het uiterst complexe karakter van het architectuurontwerpen te belichten, sloeg hoofdstuk 1 de weg in van de literatuurstudie. Hieruit bleek dat architectuur concipiëren hoge eisen stelt aan de kennis van de architect. Het vereist actief kennen en passieve kennis op zowel component- als conceptrivier niveau. Om aan deze eisen te voldoen, beschikken architecten over een reeks strategieën – analogie, metafoor, heuristieken, typologie en CBD – die één of meer kennistypes noodzakelijk voor conceptvorming aanreiken. Vanuit het perspectief van een ontwerpend architect, blijkt de belangrijkste verdienste van deze strategieën erin te bestaan dat ze de ontwerper een houvast bieden om met het complexe web van aspecten, topics en eisen om te gaan.

De rest van de thesis was gewijd aan één bepaalde strategie, het (her-)gebruiken van concrete ontwerpcases. In deel I werd de CBD-benadering kritisch geëvalueerd in de context van het architectuurontwerpen. Om deze benadering te introduceren, maakte hoofdstuk 2 een korte excursie doorheen het domein van de AI, en ging het op zoek naar de wortels van de CBD-benadering. Na ingezoomd te hebben op het subdomein van de kennistechnologie, werd nader ingegaan op Case-Based Reasoning, en vervolgens op de toepassing van CBR op het ontwerpen. De aandacht ging hierbij vooral naar veel van de strategieën die aan de basis ligt van deze toepassing, en naar de voordelen die ze architecten te bieden heeft. Aangezien het architectuurontwerpen als een ‘zwakke-theoriedomein’ kan worden beschouwd, gaan CBR-ondervinken ervan uit dat architecten vaak beroep doen op voorgaande ontwerperervaring, in het bijzonder tijdens de vroege, conceptuele fase van het ontwerpproces. Het is precies deze veronderstelling die aan de basis ligt van diverse CBD-systemen, ontwerpinstrumenten opgebouwd rond concrete cases. In essentie blijkt het grote voordeel van cases erin te bestaan dat ze (ontwerp-)ervaringen bevatten in hun originele vorm, wat hen veel rijker en levendiger maakt dan abstracte regels of modellen.

Hoofdstuk 3 sloeg de weg in van de casestudies en bestudeerde zeven vrij verschillende CBD-systemen. Archie-II, CADRE, FABEL, IDIOM, PRECEDENTS, SEED en WEBPAD werden geselecteerd omdat ze speciaal ontwikkeld zijn voor het domein van de architectuur en samen een behoorlijk beeld opleveren van de diverse sporen binnen het CBD-onderzoek. Aangezien dit onderzoek nog in volle ontwikkeling is, kon enkel een momentopname worden gemaakt van de huidige initiatieven binnen dit domein.

Na het perspectief van de CBD-onderzoeker, vond hoofdstuk 4 de tijd aangebroken om de rol van cases te bekijken vanuit het standpunt van de architect/ontwerper.
SAMENVATTING

Daarom werd zes architecten gevraagd te beschrijven of, hoe en waarom ze cases aanwenden bij het ontwerpen. Hoewel de geïnterviewden aanzienlijk verschillen qua generatie, benadering en visie op het architectuurontwerpen, waren ze verrassend consistent in hun antwoorden aangaande de ontwerppraktijk. Bijzonder opvallend was hun quasi unanieme ontkenning dat cases een rol spelen bij de conceptvorming. Wanneer ze nadien deze ontkenning herzagen, kwamen echter verschillende ingrediënten van het CBD-recept naar voren, zij het in een subtielere vorm dan doorgaans verondersteld wordt door CBD-onderzoekers. Wat betreft de rol van cases in het ontwerponderwijs waren de meningen meer verdeeld. Dit lijkt erop te wijzen dat, hoewel CBD beweert gebaseerd te zijn op hoe ontwerpers leren ontwerpen, de rol van cases in dit leerproces althans binnen architectuur verre van duidelijk is.

Om meer klarheid te brengen in deze rol, sloegen we in hoofdstuk 5 nog een andere weg in en zetten we een experiment op rond de effecten van cases op de ontwerpprestatie van studenten. Volgens de resultaten van dit experiment blijkt blootstelling aan cases een positief effect te hebben op de kwaliteit en creativiteit van studentenprojecten. Jammer genoeg traden deze effecten vooral op bij meer gemotiveerde en bekwame student-ontwerpers. Niettemin wijzen de resultaten van onze studie duidelijk op de nood aan een radicale mentaliteitswijziging in het ontwerponderwijs, waar de huidige klemtoon op originaliteit en angst voor ontwerpfixatie cases onterecht weert uit het atelier.


In een poging om de kloof te helpen overbruggen, werd besloten om ook de weg van de systeemontwikkeling in te slaan. Deel II beschrijft het ontwerp, de implementatie en evaluatie van een Dynamisch Architectuur-Geheugen On-line. DYNAMO tracht beide partijen – CBD-onderzoekers en architecten/ontwerpers – te verzoenen door andere aspecten van de CBD-aanpak te beklemtonen dan de bestudeerde systemen tot nu toe hebben gedaan. Een dergelijke accentverschuiving zou CBD aantrekkelijker moeten maken voor architecten, ontwerpbegeleiders en architectuurstudenten. In essentie kwam deze verschuiving neer op drie transformaties
ten opzichte van de bestudeerde CBD-systemen. De eerste bestond in de uitbreiding van hun voornaamste bekommernis met ontwerp-"producten" naar ontwerp-"processen". De tweede hield in dat hun statische karacter werd aangepakt om recht te doen aan Schank’s Theorie van het Dynamische Geheugen. De derde transformatie bestond in de bevordering van de gebruikers van passieve case-consumenten tot actieve case-producenten. Wellicht is elk van deze transformaties op zich slechts een subtiele accentverschuiving, maar samen leken ze voldoende ambitieus om architectuur eindelijk een rol te geven in het succesverhaal van CBR.

Wat het waarmaken van deze ambities betreft, stemt de vuurdoop van DYNAMO in een ontwerpatelier ons bepaald hoopvol. Ondanks het prototype-karakter van het systeem, waren de deelnemende studenten enthousiast over DYNAMO, in die mate dat ze het systeem in de toekomst willen blijven gebruiken. Bovendien lijken de resultaten van de evaluatie erop te wijzen dat het gebruik van DYNAMO de ontwerpprestaties van studenten positief beïnvloedt, zonder dat het gevaar voor ontwerpfixatie te verhogen. Integendeel, als DYNAMO enig effect op de creativiteit van de studentenontwerpen sorteert, dan is het een positief. Strikt genomen kunnen geen uitspraken worden gedaan over het feit of dit effect varieert met de bekwaamheid of motivatie van de studenten, maar wat het laatste betreft, wijzen de resultaten alvast in de goede richting.

Hoewel de studenten enthousiast bleken over DYNAMO, werd het systeem niet gebruikt zoals oorspronkelijk verwacht. Het belangrijkste aspect van DYNAMO is niet dat het cases aanbiedt, maar dat deze cases diepgaande casestudies uitlokken, reflectie stimuleren en discussies tussen studenten en ontwerpbegeleiders op gang brengen. Noch de studenten noch de ontwerpbegeleiders maakten echter gebruik van de mogelijkheden tot (inter-)actie aangeboden door de tool. Gezien het strakke tijdsschema van de opgave, bleek tijdsgebrek een belangrijk obstakel voor actieve participatie. In dit opzicht heeft de eerste ervaring met DYNAMO ons gewezen op het beland van een ruimer perspectief bij het evalueren van CBD-tools, en CAAD-tools in het algemeen. De invloed van de context waarin ze gebruikt worden mag niet worden onderschat.

Op dit punt neemt het verhaal van de thesis duidelijk een andere wending dan het verhaal van de generaal. Terwijl de generaal slaagde in haar ambitie om de versterkte stad in te nemen, is het laatste woord over de rol van cases in het architectuurontwerpen nog lang niet gezegd. Het prille succes van DYNAMO benadrukt enkel het belang van verder onderzoek naar het (her-)gebruiken van cases bij het concipiëren van architectuur. Minstens drie sporen lijken het verkennen waard. Eerst en vooral is bijkomend onderzoek nodig om CBD’s cognitief model nauwkeurig af te stemmen op de complexe realiteit van het architectuurontwerpen. Ten tweede dient DYNAMO uitgebouwd te worden tot een volwaardig ontwerpinstrument. Tenslotte is bijkomende ervaring en evaluatie nodig om de bruikbaarheid van DYNAMO’s aanpak te onderzoeken als strategie voor kwaliteitsverbetering in architectuur, alsook in andere ontwerpdisciplines.

In afwachting van deze toekomstige excursies kunnen we alleen maar hopen dat deze thesis alvast een tipje van de sluier heeft opgelicht over de rol van cases bij het
SAMENVATTING

architecturontwerpen. Op dit punt in ons onderzoek kunnen we deze rol misschien nog het best karakteriseren door hem te vergelijken, niet met een versterkte stad, maar met een bladerdeeggebakje bestaande uit talloze laagjes en vullingen, die elk aanleiding kunnen geven tot reflectie. Reflectie over hoe een bestaand ontwerp tot stand is gekomen, kan op zijn beurt bijdragen tot een beter inzicht in een nieuwe ontwerpsituatie. In die zin kan architectuur uit het verleden worden beschouwd als een schat aan ontwerpkennis voor het concipiëren van architectuur voor de toekomst. De exploitatie van deze kennis voor toekomstige ontwerpen is, naar onze mening, niet alleen geen schande; willen we ontwerpers ondersteunen bij het concipiëren van kwalitatief hoogstaande architectuur, dan is het gewoon een must.
LIST OF PUBLICATIONS

Articles in international journals with reviewing process


Papers in international conference proceedings


Reports, unpublished conference papers and thesis


Ann Godelieve Heylighen was born in Leuven (Belgium) on August 29, 1973. After her secondary education (Latin-Mathematics) at the Heilig-Hart Instituut in Heverlee, she studied architecture at the University of Leuven, and spent one semester as guest student at the ETH in Zürich. In 1996, she graduated at the K.U.Leuven as Burgerlijk Ingenieur Architect (summa cum laude), with the thesis ‘Case-Based Reasoning in Architectural Design’ under supervision of Professor Herman Neuckermans. The same year, she was admitted into his research group ‘CAAD and Design Methodology’ as Research Assistant of the Fund for Scientific Research (FWO) Flanders. Besides her doctoral research, her task mainly consists of teaching CAAD, supervising final year theses and looking after students as ombudsman of the Architecture department. In addition, she is webmaster and active member of the working group ‘Woman and University’ of the K.U.Leuven.