

**THE IMPACT OF INTRODUCING ARTIFICIAL INTELLIGENCE
TECHNOLOGY TO ARCHITECTURE**

AND ITS LEVERAGE ON THE CONCEPT OF DESIGN AUTOMATION

by
YASSER M. EL-QUESSNY
B.Arch. Cairo University
1984

SUBMITTED TO THE DEPARTMENT OF ARCHITECTURE
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

**MASTER OF SCIENCE IN ARCHITECTURE STUDIES
AT THE MASSACHUSETTS INSTITUTE OF TECHNOLOGY**

June 1987

Copyright © 1987 Yasser M. El-Quessny

The author hereby grants to M.I.T. permission to reproduce
and to distribute publically copies of this thesis document in whole or in part.

Signature of Author _____
Yasser M. El-Quessny
Department of Architecture
May 6, 1987

Certified by _____
Eric Dluhosch
Associate Professor of Building Technology
Thesis Supervisor

Accepted by _____
Julian Beinart
Chairman
Departmental Committee for Graduate Students

MASSACHUSETTS INSTITUTE
OF TECHNOLOGY

JUN 08 1987

LIBRARIES

بسم الله الرحمن الرحيم

الم نهرح لك صدرك * ووضعنا عندك وؤرك * الذه انقض ظهوك
* ورفعنا لك ذكرك * فإن مع العسر يسرا * إن مع العسر يسرا
* فإذا فرغت فانصب * وإلى ربك فارغب *

صدق الله العظيم

**THE IMPACT OF INTRODUCING ARTIFICIAL INTELLIGENCE
TECHNOLOGY TO ARCHITECTURE
AND ITS LEVERAGE ON THE CONCEPT OF DESIGN AUTOMATION**

by
YASSER M. EL-QUESSNY

Submitted to the Department of Architecture on May 6th, 1987
in Partial Fulfillment of the Requirements for the Degree of
Master's of Science in Architecture Studies

ABSTRACT

The problem addressed in this research is understanding the nature evolving and new technologies into the domain of architectural design and building technology.

This thesis, essentially, is an exploration of the ways that "Artificial Intelligence" techniques may support systematic and rational architectural design and, by extension, the "Building Systems" process.

The motivation for working in this area of research stems from the serious need to develop a new methodological design approach for architects. Given the fact that computer science and *artificial intelligence (AI)* in particular, is an evolving domain for problem solving techniques, it is believed that it could be a comprehensive tool for achieving that goal.

Thus, this thesis describes a generic *design methodology* using intelligent computers, that may eventually help generate a new approach to architectural design, and assist in the development of new building technologies.

This research traces two discrete, but related, concerns: *the questioning of the nature of the architectural design paradigm*, and *the applications of artificial intelligence technology in architecture*, both discussed within the context of DESIGN AUTOMATION, where their common ground is establishing a thinking model of how to approach a problem.

Thesis advisor: **Prof. Eric Dluhosch**
Title: **Associate Professor of Architecture**

ACKNOWLEDGEMENTS

I dedicate this thesis to my parents *Mohamed El-Quessny, Soa'd Hegab*, and my sister *Iman*, who have given me love, support, and encouragement. The effort put in this thesis is but a small token of gratitude and appreciation for them.

I sincerely thank Prof. Eric Dluhosch, my thesis supervisor, for his guidance, encouragement, and open mindedness; for allowing me to follow my interests, and putting my feet onto the right track for an academic research methodology. It is his insightful vision, sharp questions, and constructive arguments, that helped direct my work towards an unorthodox methodology of research

I would like also to attribute special thanks to Prof. Aaron Fleisher, among others, for his continuous intellectual debates that greatly influenced my way of thinking.

Lastly, but not least, I would like to extend my gratitude to my friends who stood beside me and made my stay at MIT, and Boston, rather pleasant.

To *Amira*; my wife, friend, and guardian angel.
It is her understanding, love, care, support, and encouragement that kept me targeted to get a decent piece of research done. To her patience, and continuous sacrifices, I am much indebted.

To *Mariam*, my daughter.
She brought happiness, liveliness, and blessings with her coming into our lives.
... *Girl, I love you very much..*

Table of Contents

ABSTRACT	iii
PREFACE	viii
CHAPTER.1. INTRODUCTION	2
1. Historical Review	2
1.1. Prelude	2
1.2. The Machine Age and the Modern Movement	3
1.3. The Information Age and Automation	4
1.4. An Overview	5
2. Design and Industrialization	5
2.1. Design of Building Systems	6
2.2. The Building Team and Automation	7
2.3. Building Type Specialization	8
2.4. An Overview	9
3. Computers in Architectural Technology	10
4. Artificial Intelligence	12
4.1. The Basic Sciences of the Artificial	12
4. 1.1. Problem Solving	13
4. 1.2. Expert Systems	13
4.1.2.1. Building an Expert System	13
4.1.2.2. Expert Systems vs. AI	14
4.1.3. Machine Learning	14
4.1.3.1. Learning by Rote	14
4.1.3.2. Learning by Analogy	15
4.1.3.3. Learning by Induction	16
4.1.3.4. Explanation-based Generalization	16
4.1.3.5. Learning by Observing (Discovery)	17
4.1.4. Knowledge Representation	17
4.1.5. Vision	18
4.1.6. Robotics	18
5. Applications of AI in Architectural Technology	19
5.1. Computer-aided Design	19
5.1.1. Conceptual Design	19
5.1.2. Synthesis	19
5.1.3. Analogy	19
5.1.4. Semantic Mapping of Graphical Objects	20
5.1.5. The Electronic Sketch-book	20
5.2. Project Managenet	21
5.3. Construction	23
5.4. Maintenance	23
CHAPTER.2. DESIGN PARADIGMS	26
1.1. Well-defined Problems	26
1.2. Ill-defined Problems	26
1.3. The Designing Problem	27
2. Why Choose Artificial Intelligence?	28
3. Design Computational Models	29
4.1. Simulation Models	30
4.2. Generative Models	31

4.3. Optimization Models	32
4.4. The Choice of a Model	33
5. Optimization in Building Technology	34
5.1. AI Approach to Optimal Design	34
5.2. Artificial Design Systems	35
5.2.1. Computational Methods	36
5.2.2. The Search for Alternatives	36
5.2.3. Reasoning Strategies of the System	37
5.2.3.1. Intelligent Databases	37
5.2.3.2. Intelligent Interfaces	37
CHAPTER.3. C.A.I.R.O. SYSTEM	39
1. Objectives	39
2. Knowledge Representation for a Designing Problem	39
2.1 Space Planning	40
2.1.1. Description of the Problem	40
2.1.2. Background	40
2.2. Knowledge Representation for Proto-type Plans	41
2.2.1. Knowledge Representation for Input/Output Data	43
2.2.1.1. Qualitative Data	44
2.2.1.2. Quantitative Data	44
2.2.1.3. Using Quantitative and Qualitative Data	46
2.2.2. Knowledge Representation for the Generative Model	46
2.2.3. Knowledge Representation fro the Optimization Model	48
3. Approach	48
3.1. Analysis	48
3.2. Synthesis	49
3.2.1. Problem Reduction	49
3.2.2. Abstraction and Implementation	49
4. The Architecture of the System	50
4.1. The Framework	50
4.2. An Overview of the Components	50
4.2.1. Backbone Module	51
4.2.1.1. Input/Output Data	51
4.2.1.2. Monitor	52
4.2.1.3. A Knowledge Base Engineering System	53
4.2.2. Generative Module	55
4.2.3. Optimization Module	55
CHAPTER.4. EPILOGUE	57
1. Future Development of CAIRO System	57
2. Leverage of "Automation" on Architecture	59
2.1. CAD and the Future Pattern of Architectural Practice	59
2.2. Forecasting Change and Effect	59
2.2.1. Effect on Individual Designer	60
2.2.2. Effect on Designing Process	61
3. An Overview	64
GLOSSARY	66
BIBLIOGRAPHY	70

List of Figures

Index	Title/Description	Page
fig.1.	The innovation Cycle as interpreted by C.Jencks/1971.	2
fig.2.	A schematic illustration for the concept of designing an Intelligent Building.	5
fig.3.	The Designing Process According to the Systems Approach.	7
fig.4.	The Building-team in the 20th. century.	8
fig.5.	Vertical and Horizontal Structures in design organizations.	9
fig.6.	A schematic architecture of URBAN5 system.	11
fig.7.	Antecedents-Consequents Rules for Logical Deduction.	14
fig.8.	A Sequence of Examples for Near-misses for Learning about ARCHes (to be used to explain figures 11/12).	15
fig.9.	Example for Learning by Analogy.	15
fig.10.	Example for Explanation-based Generalization.	16
fig.11.	Trans-frame Representation with K-lines.	18
fig.12.	A Possible Scenario on the Electronic Sketch-book.	21
fig.13a.	A Simulation Model.	31
fig.13b.	A Generation Model.	32
fig.13c.	An Optimizaition Model.	33
fig.14.	The presumed Architecture of the "Artificial Design System".	36
fig.15.	Input/Output Data for a Proto-type Plan.	41
fig.16.	The Algorithm of Design.	42
fig.17.	Input/Output Data as Quantitative & Qualitative Knowledge.	44
fig.18.	An Example of Qualitative Input Data.	44
fig.19.	An Example of Quantitative Input Data.	45
fig.20.	Generation Module Output.	47
fig.21.	Optimization Module Output.	48
fig.22.	The Architecture of C.A.I.R.O. system.	51
fig.23.	The BACK-BONE Module.	52
fig.24.	Case-library Configuration.	52
fig.25.	The KBES Architecture.	53
fig.26.	Blackboard Model for KBES Framework.	54
fig.27.	The Prunning Process through the "Multi-stage Search Tree".	55
fig.28.	The Expected Invention of Pattern of Design-effort over total brief-design-build-use processes.	61
fig.29.	Record of the Design-effort over a Part of a Typical Project.	61
fig.30.	The Effect of Automation on the Management Pyramid.	63
fig.31.	Prepresentation of a Typical Model of an Information System for the Building Industry based on a Central Computer.	64

PREFACE

For the last two decades, much research has been directed towards studying different theories of *design and building systems* [Dietz & Cutler 1978], on the one hand; and *the leverage of automating the design process in the profession of architecture* [Cross 1977], on the other hand. Unfortunately, very few of those researchers have directed their work towards the development of new building system technologies by means of computer automation. Given the urgency of such an issue, I decided to pursue the research in this direction.

Architecture design, by nature, is a classical example of ill-defined problems. Still the design process is claimed as to be *computable* (to be explained in chapter 2). Hence, the primary concern of this thesis is to throw lights on the ways computer technology techniques, and Artificial Intelligence (AI) in particular, can aid architects in resolving their architectural problems.

"*Space-Allocation*" in architectural design, has been chosen as a case-study for this research. It is considered to be an excellent domain for establishing a thinking model about approaching the design process, applying AI techniques, and introducing them into the field of architecture.

Because of the nature of the space-allocation problem (explained in chapter 4) that requires a lot of incomplete and/or unreliable knowledge, various design parameters about the nature of the problem at hand, need to be defined. Their interactions need to be described as well as categorized in order to model the designing process and finally automate it with the aid of AI technology.

This research, from one point of view, emphasizes the use of AI techniques in approaching design problems. Historically, research has confirmed that using new methodologies as well as new techniques may generate critical points of change along the architectural innovation-cycle (fig.1). The proposed system is expected to reveal a hypothesis as to whether AI techniques will affect that cycle positively, or negatively.

From another point of view, it cannot be claimed that either the hypothesis or the methodology in this thesis is perfect or complete, because of the ill-defined/ill-structured nature of the architectural design problem (explained in chapter 2).

Chapter-1 reviews some of the historical architectural theories and methodologies that have helped shape the way contemporary architecture and building technology are operating today, and how far those methodologies and tools influenced and structured existing styles of architecture. A search for new techniques in the field of *design* will then be initiated in an attempt to see their influence on future changes in style and new technologies.

Also, this chapter will briefly describe what *artificial intelligence technology (AI)* is all about, including the computational techniques that support both design analysis and synthesis. Available state-of-the-art applications of AI techniques to various engineering problems will be demonstrated; every example will be supported by currently available technology that makes its implementation possible.

Chapter-2 discusses the nature of *design-paradigms* in general, and explores how far computational methods are able to resolve problems in these paradigms. Simons' methodology: "The Science of Design", will be considered, which proposes to treat design theory as a new discipline for optimizing an objective function over a region of alternatives bounded by given constraints.

Chapter-3 explains the nature of a chosen design paradigm; "*space planning/ allocation*", and how the proposed system, **C.A.I.R.O** (Creative Architecture by Intelligent Recursive Optimization) which is intended to aid the architectural designing process, can help resolve that problem. This chapter describes ways of representing knowledge about the problem of concern. Then a description of the architecture of CAIRO system is given in more detail.

Chapter-4 is an epilogue; it suggests eventual developments for the proposed system (CAIRO) and speculates on both future trends in the use of this computer technology by architects as well as on new trends in architecture as a result of its usage.

*"Everything should be made as simple as possible,
but not simpler".*

A. Einstein

1

INTRODUCTION

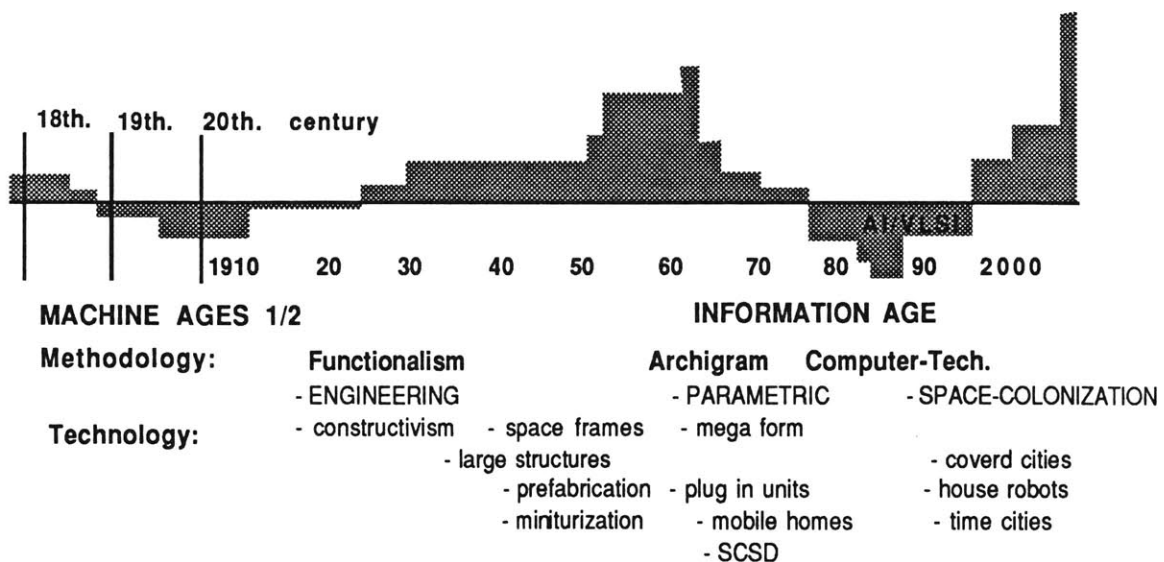
1. HISTORICAL REVIEW

The task of this historical review is to present contemporary events within the framework of their immediate precursors. A historical overlook must go far enough to render a comprehensive understanding of the present, as well as to render expectations for the future. Therefore, a historical summary of the circumstances and relationships which affect the architectural present may help in speculating about the architectural future trends, and in forecasting change in building technology as well.

1.1. Prelude

The behaviour of designers, patrons, and architects has varied according to time and place, as there were variations in problems posed to architects and in the "answers" provided.

Nevertheless, almost all the examples of these architectural "answers" have something in common, that is: they mark critical points of change on the architectural innovation-cycle throughout the modern movement, by using either a new technology, or following a new methodology (fig.1.).



[FIG.1.] The Innovation-cycle as interpreted by C. Jencks /1971 In "Architecture 2000"

Given the fact that architecture is constantly developing [Jencks 1971], this thesis seeks to explore new territories in engineering technology, and introduce them into the domain of architecture in order to take part in the architectural development process.

1.2. The Machine Age and the Modern Movement

Modern architecture was born of the technical, social, and cultural changes connected with the industrial revolution. In fact, it began with the effects of this revolution on building and town-planning, i.e. between the end of the eighteenth century and the beginning of the nineteenth century, and more particularly in the years immediately after Waterloo [1815].

The crucial point of the whole industrial development, which demanded the greatest effort in the establishment of the concept of industrialization, was to bridge the gap between theory and practice, and to undertake action in contact with reality. This was achieved immediately before and after W.W.1., and more precisely in 1919, when Gropius opened the Weimar school. Strictly speaking, this is the point at which one can begin to talk about the modern industrial movement in architecture, as a direct offspring of the Machine Age [Benevolo 1985]*.

The *first Machine Age*** had a long and productive existence in architecture. Its existence has influenced the design of some of the most significant buildings erected world wide from 1925 - 1960, marking this age, ambitiously, as *the style of our times* [Benevolo 1985]. This ambition manifested itself in a number of symbolic monuments all around the world, and the monuments of the 1950's are still standing as a proof.

What appeared to be the *second Machine Age*, as glorious as the first, started to develop in the early sixties. This age fulfilled what has been promised by the first machine age but was never properly delivered (e.g. miniaturization, transistorization, jet and rocket travel, etc.), leading us to the *Information Age*.

Indeed, the first and second machine ages, helped designers of the CENTRE POMPIDOU in Paris, SAINSBURY CENTER at the University of East Anglia, etc., to address other creative ideas in the later years, developing what is known as the *Machine Aesthetics*.

*However, some historians, as Anthony Bird, believes that Paxton's Crystal Palace [1851] is considered to be the first pilot modern building.

**The expression Machine Age has been often used by historians because it has assumed a reasonably precise meaning in modern phraseology. It has been used not only as a historical term, but as a living policy as well. Perhaps best defined by W. Morris in 1881: "The art we are striving for is a good thing we can all share, which will elevate all; in good sooth, if all people do not soon share it, there will be none to share".

Therefore, one can claim that both ages had introduced a new trend in the theories of architectural design and building technology, as well as new methods of thinking about them. In fact, these theories prompted architects to keep changing, where the keyword for changing is "*Automation*".

1.3. The Information Age and Automation

Our world today is undergoing a major change in applied technology as a result of introducing new information and processing technologies of communications.

The *information technology revolution* is not confined solely to the world of science, it is also bringing about dramatic changes in the way we live, work, and even think. Information technology, in its strictest sense, is the new science of collecting, storing, processing, and transmitting information. It is the life-blood of Industrialization.

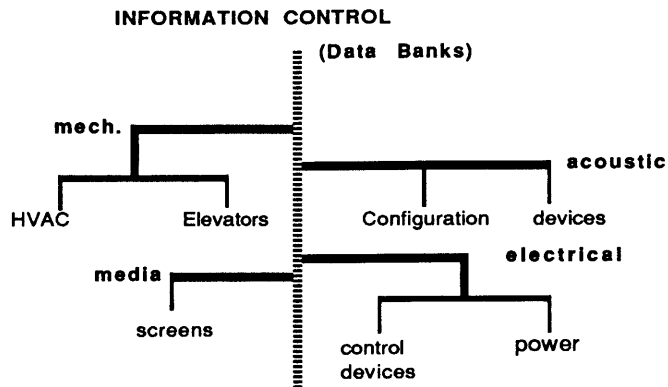
While mankind has developed myriad ways of applying and controlling power to dominate and shape our environment, mainly through the use of machines, the handling of information has lagged considerably, because of the volume and complexity of information being handled nowadays. This lag created problems in various fields of engineering development (e.g. project management). Hence, in response to the great demand of gaining access to accurate information, despite its complexity, and as a result of the availability of new evolving technologies capable of facilitating this job, scientists and engineers started considering this problem by trying to simulate human thinking by means of a parallel network of computers in order to handle information in a way similar to that of a human mind. Computers have become the corner-stone of the information age.

Technologies provided by the information age revolution drew the attention of architects and engineers to the development of new concepts of building and automated building. The *automation concept* has become the password to future building development.

Intelligent buildings were the early offspring of *Automation* concepts in architecture and building technology [Irwin 1986]. These buildings are considered to have a "quasi-nervous system" which connects every point of the building to every other point in an integrated network of passing-information*. This network is automated by means of a central computer that resides in the building's "nervous-system" (fig.2.).

Intelligent buildings brought a whole new dimension of future building concepts to architects and engineers.

*The "*City Place*" in Hartford / Connecticut, has been considered the world's first intelligent building.



[FIG.2.] This is a schematic illustration of the concept of the intelligent/smart buildings, where there is a nervous system-like structure that passes information throughout the various sections of the building.

So great is the momentum generated by information technologies, that many architects and developers (as well as other members of the building team) view this area as a major source of new growth in productivity in almost every respect (i.e. use, operation, cost, etc.). Intelligent Control made it possible to mechanize mental work, to automate it, where automation is what this age is all about [Ackoff 1974].

1.4. An Overview

Modern architecture is considered to be the symbiotic relationship between new technologies and those architects who conceived them, as reflected in their work. It is questions like: "What's next?", that kept the theory and methods of architectural design, and building technology permanently, and irrecoverably changed and changing.

2. DESIGN AND INDUSTRIALIZATION

The hallmark of the design sequence in industrialized production systems from the nineteenth to the early twentieth century, was a linear fragmentation of tasks. Each specialized task is responsible for only one part of the final product. This calls for a formalized method which splits the whole product into components, and subsequently makes sure that these components can be recombined into the final product. This method is formally embodied in the design drawings which specify each component separately and accurately, and indicate its relationships with other components.

2.1. Systems Approach and Design of Building Systems

2.1.1. "*Building-Systems Design*", conventionally defined, is an orderly approach employed by an interdisciplinary team in order to analyze and remedy problems within a defined context, which leading to optimized results. It achieves this by conceptualizing a process which utilizes selected scientific and management techniques, such as: project management, system analysis, and operation research, to name but a few.

Strictly speaking, in all phases of building systems design, the proper technological procurement, transportation, assembly, and other applications of building systems, require serious consideration of the effects of interaction between components, and their synthesis in realization.

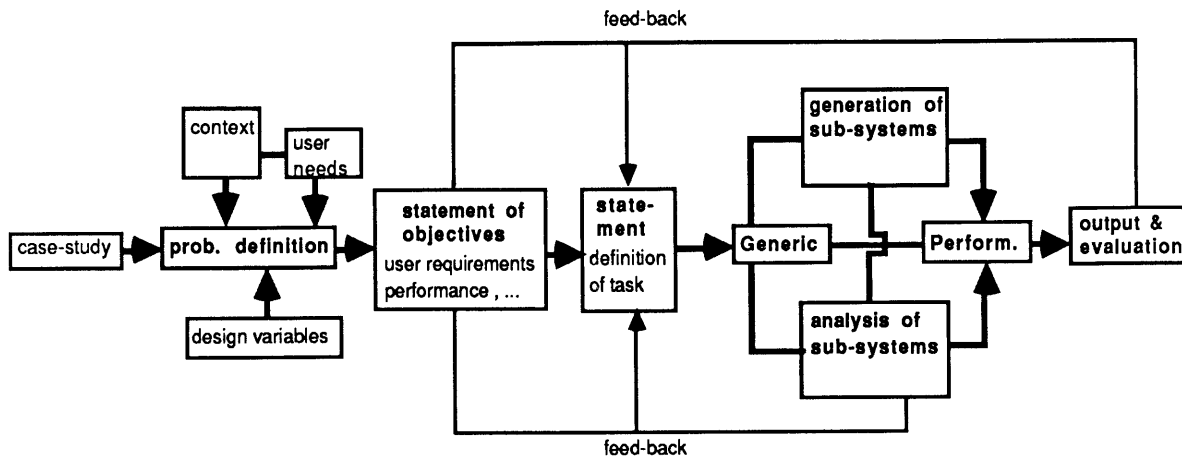
When designing a building system, all elements of that system must be considered as a whole. If not, seemingly unimportant details suddenly become significant in their ability to disrupt the solution. Therefore, a total system must integrate all the system's elements coherently.

2.1.2. A "*System approach*" is conventionally defined as the process of project development dealing with its planning, design, procurement, and erection in explicit procedural steps. Viewed as a building process, it should be able to respond to various context variables [Delgado 1987].

A "*System Approach*"* to building does not necessarily require the use of industrialized building systems or prefabrication. Work for a given project may be done in the factory. However, industrialization not only means shop-fabrication, but also the efficient organization of construction in that it combines shop-fabrication with orderly site assembly (fig.3).

The (professional) process of building systems design has two very strong features which make it an essential part of the overall pattern of industrialization: One separates designing from making. This separation by-passes the traditional craftsman's previous autonomy and authority in his work, and thus is an inherent aspect of the factory system and a subsequent development of the systems approach. The other provides, in its use of technical drawings, a formalized method for the abstract consideration of form. This method enables new forms to be devised and tested in drawn model-form before, and quite separately from, the process of both *production* and of *use*.

*While the building systems approach may provide the best way to obtain a desirable solution in terms of cost, time, or performance in a specific job, a system approach however, may readily be used for putting conventional products together in a traditional manner. Nevertheless, both notions are essential to be considered carefully for building rationally and systematically.



[FIG.3.] The Designing Process according to the "Systems Approach" [Delgado 1987].

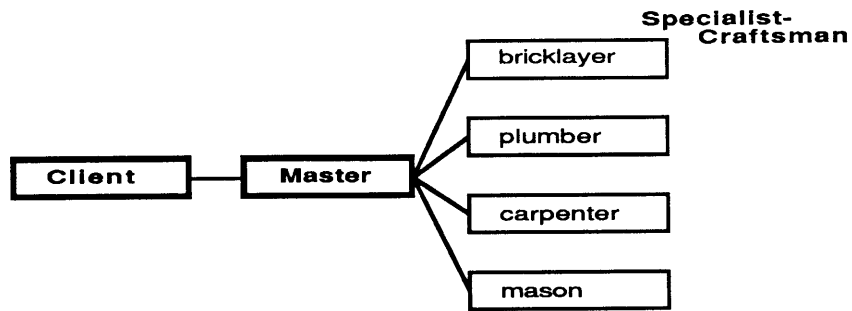
2.2. The Building Team and Automation

The design and the construction of a building typically involve a large number of different participants in the process. Those participants may be described as the "Building-Team".

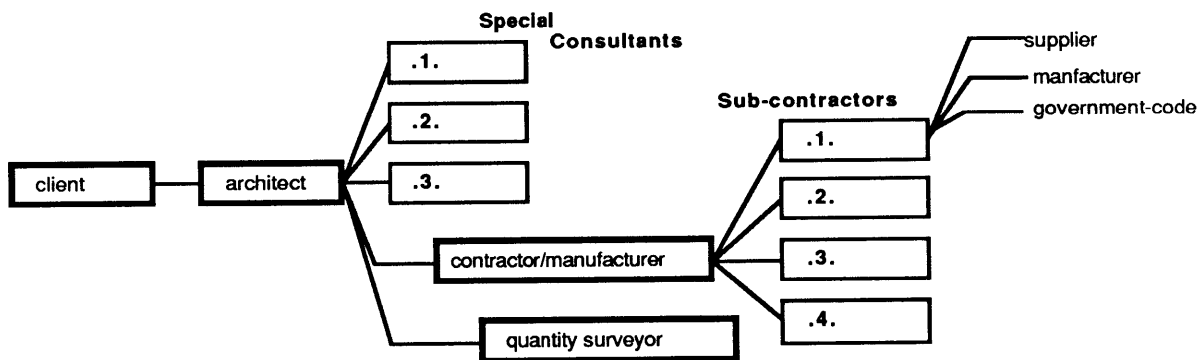
As a result of a lengthy evolutionary process (fig. 4a/4b), a fairly standard make-up and organizational structure for building teams has emerged in Britain, the U.S. and many other countries (1950-1960).

To fill the architectural design role within the larger building team, a wide range of different types of design organizations have emerged. The different tasks of a project are now being undertaken by appropriate specialists, and the result is the product not of a single mind, but of the contributions of many.

The limitations of the traditional professional process and the increasingly rapid rates of technological change, have led to the search for new design processes which use new methods and new design aids. Because of rapid technological change, many design professionals feel that traditional design methods (e.g. design-by-drawing) are no longer wholly adequate, and that a radical change is now required in the evolution of the design process in order to attempt overcoming its limitations.



[FIG.4a.] The Building-team in the medieval times.



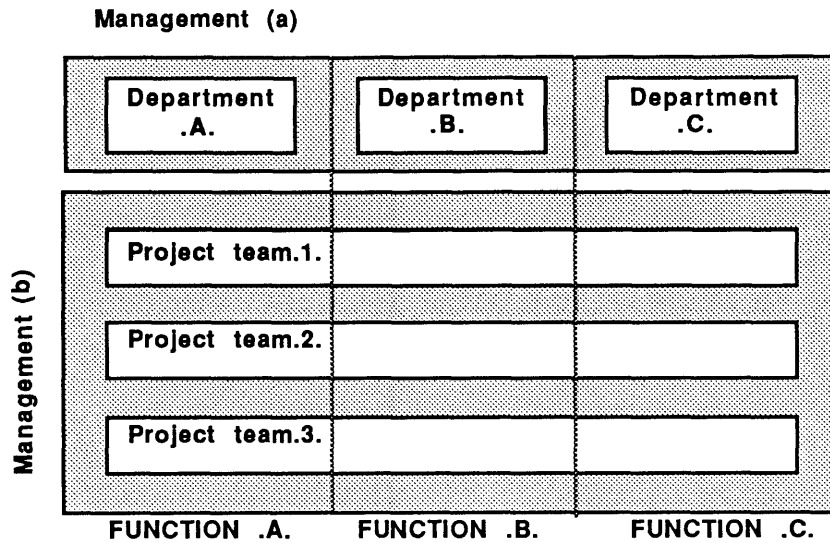
[FIG.4b.] The Building-team in the 20th century. (Notice that communication among participants has become very complicated, as well as interdependent [Mitchell 1979].)

2.3. Building Type Specialization

Since the 19th. century, building types have become increasingly diversified and specialized, as opposed to the typology of traditional buildings* (e.g. residential, educational, etc.). This diversity of building types is reflected in various design organizations, where members of these organizations tend to specialize in one or limited areas of building design** (fig.5). Generally, a typical architectural project can be broken into four standard and well-defined phases that can be summerzied as follows [Mitchell 1979]: Briefing phase; sketch design phase; construction document phase; construction supervision phase.

*In 1976, the historian Pevsner has identified about 20 basic modern building types.

**Many of the CAD systems that have been developed were produced for use by specialized organizations, and are quite specifically oriented towards a particular building. Typical examples are the "Harness" hospital design system.



[FIG.5.] Vertical and Horizontal structures in design organizations; (a) vertical structure, (b) horizontal structure, [Mitchell 1979].

2.4. An Overview

When architectural design is viewed in terms of the *systems approach* (i.e. its organizations, domain specific tasks, ..etc.), and considering how complicated it is to manage coordination among its organizations, it becomes clear that computer aided design (CAD) techniques can potentially be introduced at a variety of levels in order to resolve complexity of work among organizations (to be explained later in this chapter).

The most obvious and conservative approach is to accept the general framework of the conventional design and construction processes based on existing procedures and practices, and to develop various discrete computerized application programs and systems which merely replace certain manual design procedures in these processes.

A more ambitious approach is to replace the traditional *paper format* by a *computer based building description system* [Kalligas 1986], [Dluhosch & Vien 1987]. By integrating a CAD system with a wide variety of application programs, a *building description/assessment system* can be developed.

A still more ambitious approach is to reorganize the whole design and construction processes in order to take fullest advantage of the potentials of computer systems (CAD/CAM) with intelligent capabilities embedded within.

The application of CAD in architecture has lagged considerably behind applications in engineering. Hostility to the idea among architects, and ignorance of the potentials of computer technology, where computers have been used to address part of the problem in a fairly ad-hoc manner, contributed to this. But, the fundamental reason is the fact that architect's problems are less well-defined as engineering problems (to be explained in chapter 2). Consequently, the need for an "intelligent" machine becomes a matter of concern.

3. COMPUTERS IN ARCHITECTURAL TECHNOLOGY

Computers are considered to be the late offspring of the *Machine Age*, and at the same time, they are the "nervous-system" of the *Information age*.

The introduction of interactive-graphics terminal systems for architects posed the question of whether machines can perform some of the architectural skills, aside from being able to do representational 3-D viewing images. This question marked the first step towards thinking about using *intelligent machines* in architecture.

Why Machine Intelligence

The idea of intelligent CAD systems started with Sutherland (SKETCHPAD) and Negroponte & Groisser (URBAN5), among others [Jurgesen 1986]. It was those early idle speculations on intelligent environments by Negroponte and others, which paved the way for more research on the subject, and which tried to move architectural design to a new dimension of innovation.

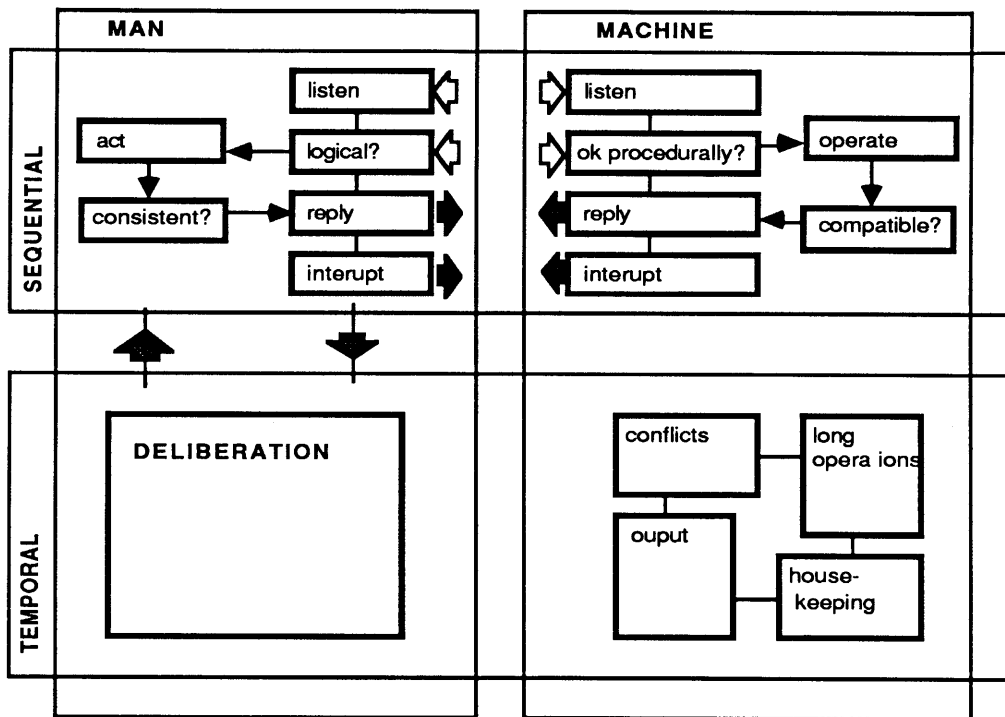
Negroponte believed that computers have the potential for assuming a responsiveness, individuality, and excitement in all aspects of living, to a hitherto, unrecognized degree [Negroponte 1969].

In his pilot experiment, THE ARCHITECTURE MACHINE, he examined the design process in terms of its being conducted (not necessarily by architects) in concert with computers that may exhibit, someday, signs of intelligent behaviour. Thus, giving the physical environment the ability to "design itself", to be knowledgeable, and to have a "self" existence [Negroponte & Groisser 1971].

Negroponte took the stand that computer-aided architecture without machine intelligence would be meaningless to architects, because the machine would not understand what it was aiding. This is a posture that results primarily from two anomalies, namely *context dependency*; and *missing information*. Subsequently, machines would have to evolve their

own mechanisms, so that each machine would differ from the other*, having its own identity, similar to human beings [Negroponte 1975].

He strived to develop techniques of thinking about "thinking", to understand the ingredients and motivations of intelligent behaviour. Also, to study the direct analysis of design activities, thus trying to effect a closer coupling between man and machine in order to achieve higher levels of replication of tasks.



[FIG.6.] The diagram illustrates the temporal & sequential organization of the architecture of the URBAN5 system. (note that the background activities are always temporal in their execution, but by definition, they surface as sequential events). [Negroponte 1971].

According to what Negroponte has revealed about his ideas [Negroponte & Groisser 1971], it could be concluded that a machine must have an *artificial intelligence*, simply because intelligent resolution of a problem requires the knowledge of its context.

*Negroponte claims that the architect is sometimes an unnecessary and cumbersome middle man, but still a human individual is needed in order to insert the intelligent capabilities into the machine until it can be totally independent.

4. ARTIFICIAL INTELLIGENCE (AI)*

As yet, there has no agreed upon definition of "Intelligence". However, the word has been reserved by authors on the subject for mental feats of unusual creativity or cleverness. Trying to impute the synonym of *artificial intelligence* with that of *human intelligence*, specialists have come up with several interpretations of what the AI technology is:

"Artificial Intelligence is the study of mental faculties through the use of computational models" [McDermott 1985].

"AI is the field of computer science that studies ideas that enable computers to do things that make people seem intelligent" [Winston 1984].

Artificial Intelligence has achieved considerable success in developing computer programs that perform the intelligent tasks of humans [Winston 1984]. It is now a well established field that includes many successful technologies, all oriented towards constructing programs that enable humans to solve problems [Minsky 1969]. It is considered to be a promising subject of research for the support of conscious and explicit assumption-making**.

4.1 The Basic Sciences of the Artificial***

The basic features of intelligence are certain *problem solving* capabilities used to communicate ideas with others. To solve a problem, it is necessary either to learn about it, or to reason about it. To communicate ideas with others in a comprehensible form, a commonly understood language, or vocabulary is needed, in order to represent the topics raised in an ordinary discourse**** [Simon 1969].

*This section discusses briefly the available state-of-the-art techniques of AI together with some available applications in architecture and engineering, confirming the feasibility of such a new technology. Each of the available AI techniques is discussed separately, with neither an antecedent nor a consequent, merely introducing those techniques first before mentioning their possible applications.

**The constellation of what was to be known later as expert systems, drew the attention of academia and industry to the great potentials of AI techniques as problem solving tools. However, the wide applicability of production rules paradigm, used in expert systems, contributed to the current blurring misconception between expert systems and artificial intelligence, (to be explained in section 4.1.2.2)

***This term is used in a broader sense by H. Simon in his book "The Sciences of the Artificial" MIT Press 1969.

****There are two other very popular problem solving techniques [McDermott 1985]: The first is; **Generate and Test** ; it uses two basic modules. A **generator** that enumerates possible solutions, and a **tester** that evaluates each proposed solution either accepting it, or rejecting it. The second technique is; **Rule-based system**, (explained in section 4.1.2)

An equally intriguing question is how far AI can solve ill-structured problems in architectural design? To answer that question, current AI techniques (or basic sciences of the artificial, that are highly viable of adding the so-called *intelligence* capabilities into the machines) should be explored as follows:

4.1.1. Problem Solving

Much of AI is about problem solving. Consequently, competence in AI requires access to a multiplicity of problem-solving paradigms.

The most common paradigm of problem solving is *intelligent search**. The class of problems within which this intelligent search is to be done, is characterized by certain *states* (an initial state and a goal state) and certain *operations* on them. The nature of the states and operations may occasionally have more than one output or input. The basic character of *problem solving* is to find the right sequence of operations to lead from an initial state to a final state (see also fig.27).

The strength of the problem-solving technique (e.g. intelligent search) is that it does not require much understanding of the domain to which they are applied, similar to most of "logic" programs. Hence, it could be used as a generic tool for solving any domain-specific problem.

4.1.2. Expert Systems

Expert systems is an area of AI characterized by an intense focus upon knowledge. The stores of knowledge in expert systems must be large, because experience shows that a lot of knowledge is needed in order to solve interesting problems [Winston 1984]. The knowledge must be task-specific, because experience also shows that we need to know, first, specific things about particular problems in order to solve them.

In other words, expert systems are an attempt to identify, formalize, encode, and use the knowledge of human experts as the basis for a high performance program.

4.1.2.1. Building an Expert System. Building an expert system is a form of intellectual cloning. Expert-system builders, or *knowledge engineers*, find out from experts what they know and how they use their knowledge to solve problems. Once this debriefing is done, the expert-system builders incorporate knowledge and expertise in a computer program, making the knowledge easily replicable, readily distributed, and essentially "immortal".

The one limitation that makes most of the available off-shelf expert systems incapable of performing as a general problem-solver tool is that each system is built according to specific

*The language should suit the pragmatics of the society of users, so that the semantics should attract the conversant. To learn the syntax of the language, as English, or a computer language for the development of graphics software.

assumptions. These assumptions are based on the logic in the area of expertise to which that technique is to be applied. Therefore, these systems are only of value to someone with a candidate problem that fits a specific domain of knowledge.

Rule-based systems for Synthesis:

Rn : if : condition-1
: condition-2

Then : action-1
: action-2

Rules of Inference:

R1: if animals have feathers
then it is a bird.

R2: if the animal flies; and it lays eggs
then it is a bird.

[FIG.7.] **Antecedents - Consequent rules are just a special case of using logic for inferring deductions according to given rules.**

4.1.2.2. Expert Systems vs. AI. The difference between expert systems and AI, in the opinion of the author, is the misconception that arises when considering their synonyms. Expert systems are logic programs that are used in coding verbal rules (*if-then rules*). Therefore, they are useful in applications where the judgement is mostly qualitative and where the current level of knowledge of the experts is satisfactory. But, since an essential feature of intelligence is *learning* [Michalski 1981], one might accuse expert systems of not possessing this *intelligence* feature, since they perpetuate current levels of knowledge without improving it*.

No expert can live up to his claim of expertise without a continuous process of *learning*. Therefore, LEARNING is seen to be the central role in Artificial Intelligence [Minsky 1975].

4.1.3. Machine Learning

Machine Learning is considered to be the most fundamental feature of intelligence. To *learn* is to extract deeper insights into a repeated situation. No matter how often the problem solver encounters a problem, he will always try exactly the same series of moves [McDermott 1985], perhaps with some variations in strategy, even if these moves end in failure.

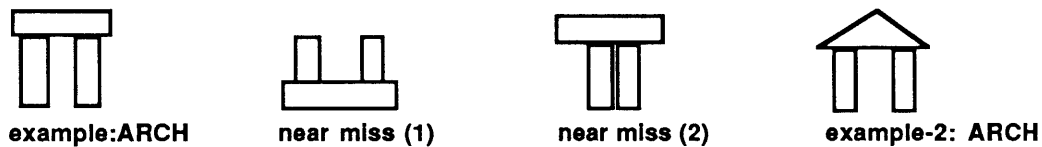
The following is an overview of the categories of Machine Learning:

4.1.3.1. Learning by Rote: This is the way we learn how to utter our first words and how children do multiplications, i.e. by memorizing the multiplication tables. A possible application of learning by rote is encoding some fixed rules in the form of "*if ...then*". This is similar to the low-end of the range of current expert systems.

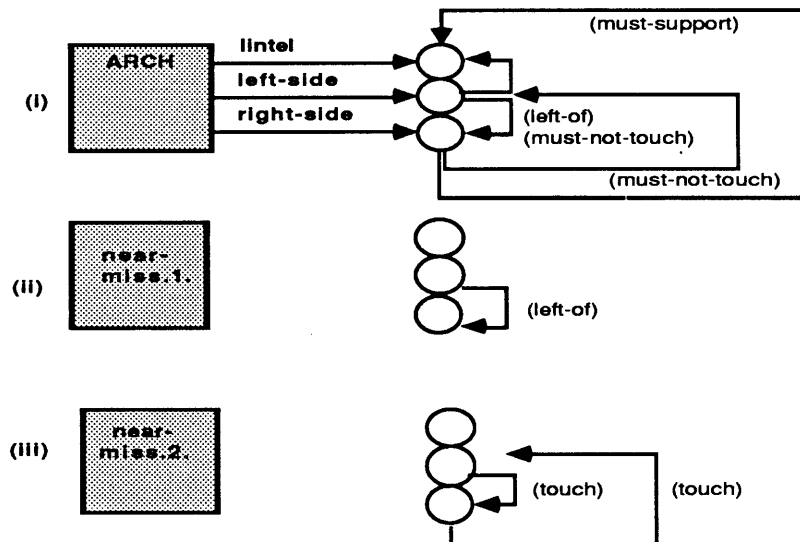
*The programs BACON [Simon & Langley]; AM [Lenat]; are attempts to feature learning capabilities in an expert system.

4.1.3.2. Learning by Analogy. Learning by analogy enables us to develop programs that learn from cases encountered in the past. Hence, reasoning about new cases proceeds according to knowledge acquired through previously encountered cases.

In this category, learning takes place by *positive examples* and by *near misses*, which would have been *positive examples* except for their violation of one or a few aspects of reasoning about the given examples [Winston 1980]. Learning by Analogy is very helpful in domains where the knowledge sought is yet unrecorded. By feeding both positive and negative examples into the program, a structure of the knowledge starts to be formed.



[FIG.8.] A sequence of examples and near-misses for learning about ARCHes (to be used in explaining examples of fig.9. & fig.10.) [Winston 1984].

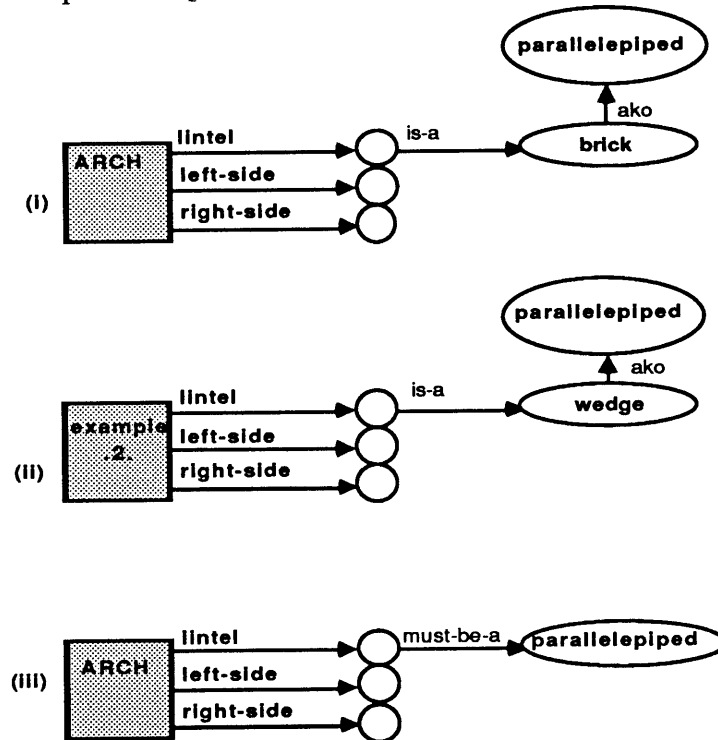


[FIG.9.] The near-miss-1 frame lacks support-links. Therefore, the conclusion is that support-links are essential. The support-links in the ARCH frame are altered, indicating that they are required in all the arches as in (i). The near-miss-2, compared with the ARCH frame in (i), adds touch-links. The conclusion is that the touch-links must not be present. Touch-links are added to the ARCH frame in (i), altered to indicate that they are forbidden in all arches. (Notice that "left-of"; "must-support"; etc. are qualitative attributes attached to the links).

4.1.3.3. Learning by Induction. *Learning by Induction* is a superset of *Learning by Analogy*. Therefore, the formation of knowledge by means of encountering both positive and negative examples is a typical application of the approach. It can be viewed as a *heuristic search*, through a space of symbolic descriptions, generated by an application of various inference rules of generalization, and specialization [Michalski 1984].

In engineering, we find several evolving domains where experts can classify the instances as positive or negative examples, but where they are unable to express sophisticated knowledge as a systematic hierarchical structure. *Learning by Induction* promises to structure the transformation of knowledge in these engineering domains (e.g. structural behaviour design process).

4.1.3.4. Explanation-Based Generalization. *Explanation-Based Generalization* is based upon having a half-ordered theory*, according to which the system tries to explain every case encountered. If the theory explains a case, then the system, inductively, generalizes the theory used in its explanation [Mitchell 1985].



[FIG.10.] The Intel in the ARCH frame in (I), is a brick, while the corresponding object in the example frame in (II), is a wedge. Evidently, it does not matter. The IS-A link in the ARCH frame is changed to MUST-BE-A link, and directed from brick to parallelepiped, as shown in (III), which is the most specific common generalization of "brick" and "wedge", [Winston 1984].

*A theory that partially covers an evolving domain.

4.1.3.5. Learning by Observation (Discovery). *Learning by Observation* is used in situations where an appropriate condition is not applicable. The learner must depend on himself by extracting knowledge from random examples to which he may be exposed [Lenat 1983].

A current experimental application is in the domain of failure mechanisms in pavements [elShafei 1985]:

- An Autonomous Land Vehicle (ALV) gathers information about cracks in pavements through a vision system -*Observation*.
- The ALV system/program builds a case-library about the information gathered, using machine learning techniques -*Learning*.
- Subsequently, the system gives a diagnosis of the "failure" based on the primary information about the domain it has knowledge about (theory), in addition to the learnt information from recent diagnostic experience.

4.1.4. Knowledge Representation

This is probably the part of AI where the largest number of paradigms have been developed. Knowledge Representation is very vital in any mental process, and drastically affects its efficiency. Similarly, it is important for simulating a real problem in a computer system as it is in our minds. This offers the researcher a real chance to inspect his own knowledge about a specific domain once he lays down this knowledge as *code**

Structuring and studying the representation in itself offers several insights into the domain, which may spur useful modifications not perceived before.

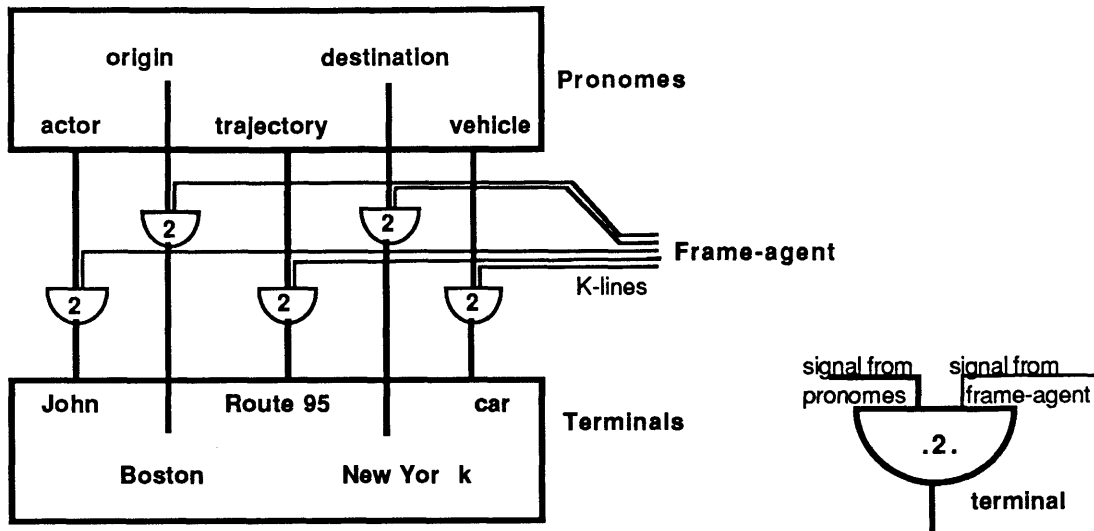
Thus, the crucial question becomes; "How to represent/structure knowledge for a system?".

One of the major approaches for representing knowledge is introduced by M. Minsky as "Frames" [Minsky 1975]:

Frames

A frame is a mental structure for representing data about a common situation. It contains data about an event, space, or object, along with information about how to use the frame's expectations and what to do if the expectations are not met [Minsky 1975]. Such information is always in the form of loosely attached default values. This is parallel with human perception in routine situations, where observation is accompanied, and often directed, by pre-existing mental states called "beliefs".

*Some of the major approaches to the representation of knowledge:are: *predicate calculus; procedural embedding; semantic nets; semantic primitives.*



[FIG.11] Trans-frame representation with k-lines [Minsky 1987].
 "John drove from Boston to New York on the turnpike"

4.1.5. Vision

Vision is the information-processing task of understanding a scene from its projected images. An image is a two dimensional function $[f(x, y)]$ obtained through a sensing device that calculates the value of an image feature at all the points $(x, y)^*$. The task of a machine-vision system is to understand the scene represented by an array of pixels** [Bar, Cohen & Feigenbaum 1982].

4.1.6. Robotics

There is a dual ancestry to robotics emerging from industrial engineering: *automation technology*, and *computer science and AI*.

Industrial robots are a fact in the shop floors in many factories and this is a proof of the usefulness of hybridizing AI techniques with other engineering technologies (e.g. building technologies) tackling bigger problems in a better way***.

*These values are binary for black and white images, gray level for half-tone images, or vectors of color measures for colored images.

** The value of projected points being pixels

***Those bigger and harder problems will be discussed in section 4.2, through the manner of AI applications in engineering showing how better AI techniques will handle those problems.

5. APPLICATIONS OF AI IN ARCHITECTURAL TECHNOLOGY

In engineering, we can find a number of paradigms of problem domains for which AI could be the best way for analyzing and solving those problems [Winston 1986]. The following paragraphs are a review of some possible AI applications in architectural technology and engineering that have been either implemented, or are implementable

5.1. Computer-aided Design

CADs, as discussed previously, have become an important feature in many architectural and engineering offices. Adding intelligent capabilities to a CAD system represented an essential step to be taken towards *automation* [Kroll 1983].

The following is an overview of how AI techniques can be used for aiding architects/engineers in their work.

5.1.1. Conceptual Design

Conceptual design is the most important part of the design process, as it is the foundation that supports all proceeding steps.

Using AI techniques, one can develop a "feature-driven CAD": The user defines the features expected from the product (e.g. "closure"), and the system starts gradually building up a description of the product. The system depends upon its own expertise (from learning), i.e. primary knowledge, plus experience gained from previous sessions, but it needs to switch into an interactive mode whenever several alternatives meet the requirements equally.

5.1.2. Synthesis

Similar to what a student-architect is taught at school, the proposed system should know:

- Elementary design blocks in the domain under consideration and their functions.
- The basic concepts of combining several elementary design blocks together into a system capable of performing multiple and more complex tasks.
- How to synthesize several systems into a bigger system that is capable of performing multiple and/or more complex tasks.

5.1.3. Analogy.

The technical edge that an experienced architect has over a novice is that he (the experienced architect/engineer) still uses the synthesis/analysis paradigm in an *analogical approach* aside from the *decompositional approach* used by the novice engineer [elShafei 1986]. Once an architect captures the conceptual idea of a specific design, he keeps the characteristics

of his design in his mind through *memory* organization; such as the k-lines proposed in Minsky [1987], so that he can retrieve these characteristics meeting similar cases.

By keeping a sort of *case-library* in mind or in a file, the experienced engineer can design new systems in shorter time. Any knowledge-based-system should take advantage of this to deserve being called an *expert system*..

5.1.4. Semantic Mapping of Graphical Objects.

Understanding the semantic mapping of graphical objects is an essential criterion for differentiating between an *intelligent* CAD and a conventional drafting system.

Through semantic mapping, we build a *model* of the real-life physical object, which captures more than its geometry, i.e. frame representation, as for example: considering orientation in the design of bedrooms.

Only an *object-oriented* CAD system can symbolically manipulate drawn geometrical models according to real-life processes (eg. rotate pieces of furniture with respect to the room) and automatically propagate the required changes in all the views and at any level of detail that may get affected by that primary manipulation.

The building up of a model of a physical object enables the system to understand how the different views describe different aspects of the same model in the real world. Based upon this understanding, the system gains the following capabilities [El-Quessny 1987]:

- 2D/3D conversions.
- Computer-aided dimensioning.
- Sketchy modeling.
- Design guidance.

5.1.5. The Electronic Sketch-book.

Since the design process is not linear in nature, architects often want to sketch an idea other than the one they are currently working on. A system can be created for the user to call up an electronic sketch-book capable of capturing a flashing idea and recapture ad-hoc ideas. A mechanism must be developed for cross referencing, so that if the user wishes to design a component for which he had made previous notes/sketches, the system will provide him with these notes/sketches to read, and/or allow him to input them into his drawings.

Using Minsky's theory of "frames" [Minsky 1975] for knowledge representation about various objects in the world of the electronic sketch-book, will help in describing a possible implementation (see 4.1.4).

Once the complete design of the product is passed to that system, and using a system with learning capabilities and domain knowledge, the user can ask for one of the following (fig.12.):

- Generate shop-drawings and details, from drawings.
- Design a mold for the product under consideration.
- Detailing of falsework.

object frame

object-name:	window-1
object -type:	corner-window
a-kind-of:	opening
a-part-of:	exterior-panel
object-function:	daylighting-and-ventilation
keywords:	'(corner-window opening)
material:	aluminum
consists-of:	'((section#32) (section#34) (frame#4))
points-of-contact:	direct
peculiar-properties:	(center-distance-cms. 90) (angle-of-rotation upwards-180) (glass-fill-cms 1.2))
physical-file:	"x1>cad.window-1"
remarks:	"just a cloudy sketch"

[FIG.12.] This figure describes a possible scenario in the "electronic sketch-book".

5.2. Project Management

Nowadays, a typical construction project uses a tight mesh of interconnecting and well coordinated disciplines. A project manager has the job of organizing the work on site, providing coordination among job participants, and keeping track of every minor detail of all work being done on site.

Project management has developed into a vital engineering domain, but obviously in all projects, and in mega projects in particular, it would be hard for any human individual to perform a perfect job of managing a project, given the many variables that correspond to the tasks being executed by the different job participants on site. Therefore, a computer is of considerable utility for monitoring a job in all its complexity and detail.

The following section is a discussion of the potentials that AI holds for the project management domain, both in *general*, and in *mega* projects.

Project Control Planning

This has been a focus of intensive research during the past 50 years leading to various techniques that range in sophistication from "Gantt charts" to CPM and PERT methods.

Despite the numerical precision gained by the computers, project management planning packages (e.g. Primavera) still suffer from the following serious limitations [elShafei 1986]:

- **Delay-fragility:** CPM systems, as a market standard, remain vulnerable to numerous *project planning* violation. PERT systems try to avoid that caveat by assigning a discrete probability to the "task" duration instead of a fixed value as in CPM, but the estimation of the probability distribution in itself is prone to error.
- **Modification inflexibility:** Almost all projects experience some kind of modification at various stages of work. Accordingly, the work schedules have to be adjusted and readjusted every time a change is made. Current computer packages for project management [Primavera 1981] enable users to make the needed modifications through successive manual interference, but not automatically.
- **Insufficient details:** Almost all planning packages do not go all the way in breaking down the job "tasks" into subtasks because of the exponential complexity associated with such breakdowns. Hence, project managers settle for some level of detail to stop at and start feeding in their *lump-sum* expectations for the "tasks" at that level. The bigger the project, the shallower the level of detail at which the project manager stops the automation [Anderson 1978].

Therefore, using AI current techniques, an *intelligent planner* with the following features should be built to contain:

- Gradual understanding of the surrounding world.
- Hierarchical Planning.
- Central Resource allocation.
- Delay Tolerance.
- Adaptive Planning.
- Monitoring & Evaluation.

Mega Projects

The *intelligent planner*, as proposed previously, answers more of the problems for the mega projects than it does for small projects.

Mega projects are often constructed by companies foreign to the client country. Accordingly, the lack of mutual understanding between the contractor and the local authorities is not rare. The development of an *atlas knowledge package* (one for every country) should

help to bridge such a gap. Every package should describe, on-line: the country, its economy, its laws,..etc.

5.3. Construction

The field of construction is comprised of various subfields, most of which have the potential to be automated. It is expected that robots will invade construction sites and take care of certain jobs that are currently being handled by humans (for example, robots can take care of hazardous jobs. Besides, they also have the edge in terms of speed, accuracy, access, etc.) [Kroll 1983].

Robots in Construction

The following are some of the jobs that could be assigned to a robot in a construction site:

- Site preparation, which contains a considerable amount of cut and fill.
- Placement of precast concrete.
- Post tensioning.
- Erection of steel frames.
- Cladding.
- Handling of hazardous materials.
- Maintenance [HOTEP; elShafei & Moavenzadeh 1986].

5.4. Maintenance

Maintenance now makes up a considerable share of the construction industry market. This portion varies from country to country and from state to state. Maintenance governs about 29% of the US Construction market [National Bureau of Census 1981].

Opportunities to enhance maintenance efficiently can be found in developing technological improvements in equipment, materials, and processes used for maintenance. AI technology offers a wide range of technological improvements as, for example, in equipment. Following is a discussion of the ways AI can help assess a more efficient maintenance market.

Condition Assessment

In maintenance circles, the term "*condition assessment*" is a synonym for the term "*monitoring*" in project control, therefore the previous arguments (discussion in section 4.2.2) are also valid here.

Every engineering domain has its own maintenance requirements, which in turn are controlled by condition assessment. This takes place through the detection of known, or unknown, patterns of behaviour in the system under consideration. Such detection, or *pattern*

recognition -in several applications, is not a straightforward *matching* exercise, but rather needs some mental faculties to infer these patterns. In Civil Engineering, early applications can be found in the work of Haas et al [1985], and elShafei & Moevenzadeh [1986].

Robotics is an obvious example of AI's continuing potentials in engineering technology. As mentioned before, the "Road Machine" is a maintenance vehicle for improving road conditions* [elShafei & Moevenzadeh 1986] (see also 4.1.3.5).

This road machine has been under development for the past three years at the AI-lab (MIT). It is a maintenance station pulled by a car over a highway, and it performs the following tasks:

- Non-destructive condition assessment, through a vision system, of pavement crack samples
- Diagnosing the causes of detected distresses in failure mechanisms.
- A robotic arm is attached for obtaining required additional information (physical samples).
- A feedback system for evaluating previously made decisions.
- A learning facility is attached to take advantage of the cases and feedback in improving the system's knowledge.

*More information about that machine is found in the previous references.

"How many times in the course of my life had I been disappointed by reality because, at the time I was observing it, my imagination, the only organ with which I could enjoy beauty, was not able to function, by virtue of the inexorable law which decrees that only that which is absent can be imagined".

Marcel Broust

.2

DESIGN PARADIGMS

In the work on design paradigms today, the design process is understood as a process of defining, incrementally, an initially ill-defined problem (section 1.1/1.3) and concurrently proposing and testing possible answers [Eastman 1975]. It is also characterized as a series of decomposition and recomposition cycles [Kroll 1983].

As new knowledge becomes available, either through the gathering of data, or the knowledge generating process itself, portions of the design problem are isolated into well-structured subproblems that can be solved by analytical or *heuristic* *techniques. i.e. that is finding *a solution* to the problem, and not finding *the solution*.

In general, problems can be categorized as: *well-defined* problems and *ill-defined* problems.

1.1. Well-defined Problems

A well-defined problem exists when the knowledge about the domain of the problem has a high degree of coherence, and lends itself well to technological solutions. Solving the problem becomes simply a matter of identifying the problem and then applying known solution procedures to the relevant facts. That is to say that, if the problem goal-set (G) contains numerous acceptable solutions, there is room for variation, and different solution generation procedures may tend to generate characteristically different subsets of (G), or a single solution if optimization is feasible.

Well-defined problems can be solved by means of the use of *algorithmic* strategies. There are several ways of solving a well-defined problem algorithmically. The simplest is the method of *plug-and-chug*, where appropriate values are substituted in an equation. Complex problems may be separated by *decomposition*, then solved.

1.2. Ill-defined Problems

Ill-defined problems are defined as such for two reasons: (1) The solution procedure may not be known, or (2) all the information required for the solution is not available. Hence it is often difficult to articulate exactly what the problem is, and it may even be more difficult to determine whether or not a design solution is really "the solution" to a problem.

Although the concept of an ill-defined design problem is not in itself very well-defined, it is possible to identify some practical characteristics:

**These are means of dealing with difficult formalized problems. They are the distillation of knowledge into a highly usable form. They are broad strokes that lop off whole sets of possibilities, effectively reducing the variety of options and relieving the decision-maker from reviewing all alternatives.*

- No complete or definitive formulation is "given" at the outset.
- No single applicable model is inevitably "right".
- The formulation which is achieved is not rigorous in the scientific sense.
- The problem formulation and the solution do not remain stable at over time.
- The formulation may embody conflicts and inconsistencies which must be resolved during the course of the design process, thus changing original assumptions.
- The problem solver does not have all pertinent information available for consideration at any one time, nor does he know whether the information he has is sufficient or insufficient for a particular solution.

1.3. The Design Problem

There are different schools of thought about designing in architecture, with different interpretations of what design aspects are [Jurgensen 1986].

- One school believes that it is a *problem-seeking* process, confirming the idea of creation (tautology), i.e. the process is the application of creativity. Here, the designer must find the problem to be solved before he can solve it (For example: "A house that affirms the concept of enclosure").
- Others believe that it is a *problem-solving* process, where there is an identifiable problem as well as an identifiable criterion for achieving a design goal that is not trivially attainable otherwise. Accepting the latter interpretation, the design process can be understood to consist of successive stages of refinement, where refinement proceeds in two alternating steps:
 1. Describing constraints.
 2. Exploring alternatives/variants.

The vital function of design thus becomes matching the design vocabulary with context requirements and constraints, where the hierarchy of preferences guides decision-making about the solution*.

However, the boundaries between well-defined and ill-defined problems in architectural design change as people learn from experience and incorporate, into models and systems, the intelligence gathered in the process of past experiences. In fact, not only do the boundaries change, given the general context of the original problem, but the context itself changes along with the nature and scope of decisions as knowledge is acquired. Thus, we can say that designers become pattern recognizers who attempt to manage, intelligently, the complexity manifested by the design problem.

* *Although there is a hint of perfection in the model of evaluation as a phase of matching a design against its requirements, such a view can create problems, where requirements are ill-defined. It also leaves no room for fulfilling requirements that were not articulated, or for innovation in general.*

Knowledge about the Problem-Solving Process

Acquiring new knowledge about the design delivery process is as important for the automation of the design process as it is for improving the practice of architecture. In fact the two goals are closely related. As we are able to better *understand* the paradigms that lead to design decisions, we find ourselves able to respond to a larger number of design constraints and achieve higher degrees of success in reaching a good design decision about the solution. Similarly, this *understanding* of the paradigm gives rise to thinking about computational tools which may assist in solving specific design problems more accurately, as well as make the design delivery process more efficient.

Computational tools used in representing knowledge about a design problem work within a general framework which includes;

- Powerful man-machine interfaces,
- High level heuristics for integration of powerful solutions.
- Expert systems for the management of the design delivery process.

If one assumes that the design process is characterized by a set of coherent decisions about given constraints which have an observed effect on the final artifact, then it may be possible to work backwards from a set of existing designs and extract knowledge about the decision making process in terms of quantitative and qualitative types of knowledge which were required for representing the original design problem.

2. WHY CHOOSE ARTIFICIAL INTELLIGENCE (AI) ?

"The primary goal of AI is to make machines smarter". [Winston 1984].

It may be ascertained that an intelligent approach to any problem is essential. In response to this, we find that in the recent past AI techniques have come to be a valuable tool for successfully systematizing and utilizing large bodies of informal, empirical, contextual and judgemental knowledge. The problem of dealing with multiple objectives is also handled well by present AI tools. An inherent feature of these tools is that they facilitate the organization of existing knowledge that was hitherto available only in disconnected "chunks".

AI, with its present techniques, is expanding the scope of understanding problems, and is enriching our capabilities to deliver viable solutions to otherwise difficult and resistant problems (eg. man-machine interaction).

Here are some of the reasons why AI techniques have been chosen to work with. These can be summarized as follows:

- Unlike other computer techniques, various AI techniques, when confronted with ill-structured problems, such as architectural design, use a heuristic approach.
- Existing methods of symbolic and mathematical reasoning have limited capabilities for representing ill-defined problems.
- Pragmatic reasoning would suggest that if a human expert achieves outstanding performance because he is knowledgeable, then computer programs that embody and use that same knowledge would be equally effective.

From another perspective, the very idea of AI research can also be viewed as a very large design project, performed with extensive computer aid. Therefore, simulation models are built in order to gain additional insight into the use of particular methods or alternative methods for handling a certain problem (to be discussed in chapter 3).

3. COMPUTATIONAL MODELS OF DESIGN

Design computational models are integrated abstractions which in turn, form systems of thought that lead to models of design.

The contingency of artificial phenomena [Simon 1969] often creates doubts as to whether these models fall properly under science. Sometimes, these doubts are directed at their teleological character and the consequent difficulty of disentangling *prescriptions* from *descriptions*.

As models are either used for prediction or simulation, they are defined in Operation Research literature as *descriptive* or *normative*, the former being either *static* or *dynamic* and the later being of the *analog type*. However, this classification is unnecessarily restrictive as *iconic* and *symbolic* models are often used, especially by architects, and are often considered from the point of view of manipulation rather than pure representation**.

Computational Models in Architecture

Computer modeling systems in architecture provide designers with a tool that offers some distinct advantages over traditional media used in design.

*The need for the designers to know how things 'ought-to-be' in order to achieve goals and functions, introduces a dichotomy between the normative and descriptive.

**A different computational model of expertise is the use of production-rules, i.e. expert-systems. Design expertise involves informed preference among alternatives as well as logical deduction.

Traditionally, designers' solutions are executed by recording their ideas on some external medium, such as a drawing or a physical model*. The purpose of modeling is to take the geometry of an object and to create a (computer) model which can then be inspected and tested, and subsequently modified according to specific criteria.

More important than the ability of the new medium to store geometric objects is its ability to automate methods of design, (manipulation of forms limited by human individual skill). Thus, the purpose of building a computational model is to test the problem's constraints as a representation for design expertise, through a simulation program, and also to provide designers with a higher-level platform for programming than conventional language.

Present CAD models, with few exceptions, fall into one of three categories; namely, *Simulation, Generation, Optimization*.

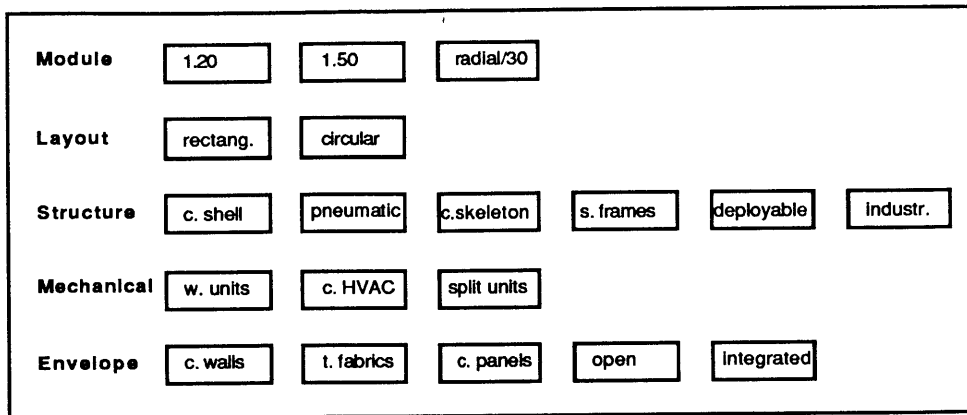
3.1. Simulation Models

Simulation is the imitation of a system and its testing under a variety of simulated, or limited environments, to gain a better understanding of the system.

The great strength of simulation is that by fixing the design variables one can examine their consequences on as many different aspects of the problem as there are prediction methods available (e.g. structural simulation).

However, the disadvantage of simulation is that it requires the user to operate by *trial and error*. To get any quantitative information, he must first have a solution, and the design process therefore involves a cyclical procedure of evaluation where different possibilities of design options are examined. Moreover, design options and sensitivity to changing assumptions can only be investigated by repeating the simulation many times with different sets of decisions. It also tells the designer nothing about the chosen solution compared with other feasible solutions, unless the analysis is repeated with different designs in a process of informal optimization.

In *simulation models*, the computer is used to predict the consequences of a set of design decisions by manipulating a constraint model which describes the design. All decision making is external to the model.



[FIG.13a.] A morphological logic chart for the design of a Building System is considered to be a classical example of a SIMULATION MODEL.

3.2. Generative Models

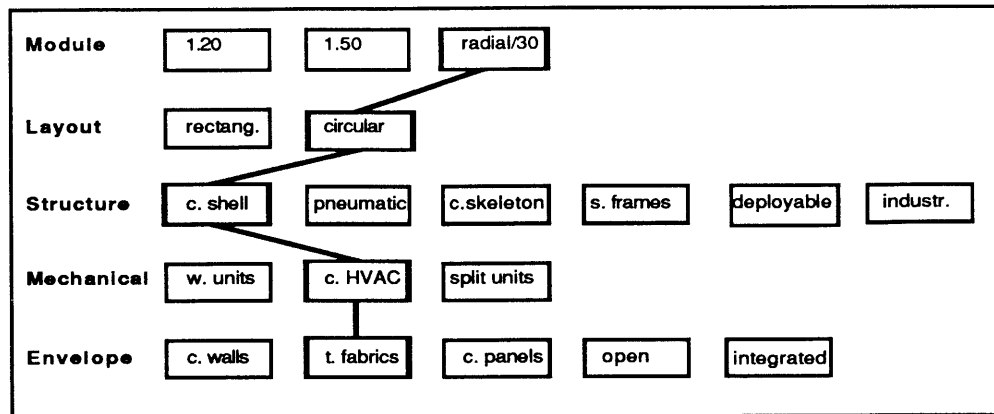
If suitable decision rules can be formulated, generation models will produce an unranked catalogue of alternative solutions from which the designer can choose* (e.g. the study of rules and priorities that make an architect's work distinctive).

If the goal of a problem-solver is to obtain some pre-existing tangible object (e.g. a pen), then the problem-solving involves: First, designing an appropriate candidate object. Second, verifying whether it matches the given goal description. Finally, the pre-existing elements are re-assembled into a new configuration.

But if the goal is to design something that is as yet non-existent (a proof of a theorem, or the design of a building), then the question of how a potential solution may be produced arises. The answer may be found in constructing a Generative System which can be used to produce a variety of potential solutions.

In *generation models*, the computer is used to explore the consequences of a *recursive* application of an ordered set of decision rules that confirm some meaningful value to the architect and/or meet the requirements stated as constraints.

*The application of used by Palladio to design his villas, [Mitchell 1979]. By investigating the style of architecture by means of a generative system, it proved possible to generate a catalogue of floor plans that Palladio might have used.



(FIG.13b.) The same morphological chart can be used as a GENERATION MODEL, by going through its possible permutations, one attribute of a row at a time. A total of 540 different designs could be generated.

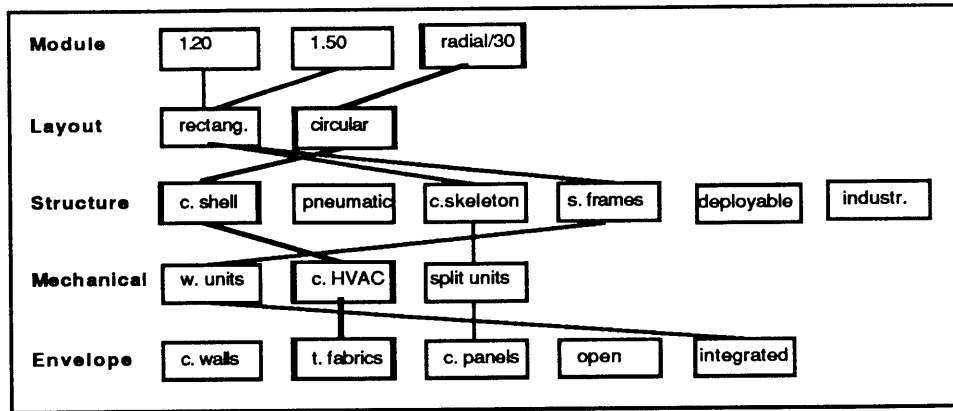
Having the ability to thoroughly arrive at a solution would allow the architect/machine to intelligently challenge and reform the assumptions about the generated alternatives*. As a result those models would allow him/it to explore a broader range of possibilities for any particular design. This is particularly important in complex projects where the architect's primary role may be to orchestrate various aspects of the design job among participants.

3.3. Optimization Models

Optimization models search the whole field of feasible solutions to identify those best suited to the stated goals. Thus, optimization directly approaches an answer to a designer's fundamental question; "what is the best solution?".

These models define "approaches" to an answer rather than an "answer", because most architectural design problems are characterized by disparate and conflicting goals. Thus the definition of the "best solution" depends on the designer's judgement concerning the relative importance of the different objectives, (example: layout-planning, building services, space allocation, ..etc.). However, optimization models tell the designer nothing about the performance of the objectives, unless he further investigates the solutions produced in a process of simulation.

*Unlike the production rules used in inferring solutions, generative systems transform sets of procedures, plans, for constructing layouts, rather than operations on the spaces themselves.



[FIG.13c.] Of all the possible design alternatives generated for a given Building System, only one is chosen according to pre-defined criteria. OPTIMIZATION models determine the optimum solution to be chosen.

The disadvantage of optimization is the difficulty of formulating meaningful and quantifiable objectives in a discipline characterized by multiple and ill-defined objectives. However, by using *object-oriented programming* at a high level of *abstraction*, it is possible to resolve the conflicts between competing constraints.

In *optimization models*, the computer is used to prescribe and document a set of decisions in order to achieve a specified goal. Some decision making is internal to the model and is purposeful. Decisions are chosen according to their ranking on an explicit scale of effectiveness.

3.4. The Choice of a Model

The question as to which model approach is appropriate, depends on the kind of information to be extracted.

To facilitate the extraction of design decision knowledge, it is necessary to structure the data about given constraints in a way that matches not only the objects that are to be discovered, but also the process by means of which such objects are generated. One structuring method that satisfies these needs is *multi-criteria optimization*. This method identifies a set of design solutions among which a best solution for any group of goals to be evaluated must lie [Gero & A.D. Radford 1980].

Therefore, after discussing the differences between the various approaches, and mentioning the advantages as well as the drawbacks of each approach, a *"hybridized"* system is believed to emulate and complement each of the models through the manner of its application.

4. OPTIMIZATION IN BUILDING TECHNOLOGY

Initial optimization research has been focused on seeking improved optima (in *non-convex quadratic programming problems* [Casalina 1975]), such as the layout design of systems and facilities including: hospitals, commercial buildings, industrial and urban layout problems.

But when talking about building systems technology, optimization of design often takes many dimensions with respect to finding an optimum solution, i.e. there is an optimization problem for every single step taken in the process of designing a system*.

Optimization also provides a means of predicting the most likely demand, or outcome, (e.g. the best allocation of resources, or the best description or classification of a system, etc.). In building systems design, decisions concern the physical form of *building* (e.g. placement of windows, and other design elements and systems together), and it is the arrangement of these physical elements which are of primary concern to the architect.

Optimization Techniques

Aside from the various optimization techniques mentioned earlier, Linear Programming offers itself as a simple but powerful tool for executing the optimization processes.

Linear Programming is an efficient and well-developed numerical method, which has been used to determine distributions of, for example, apartment types and land use, and is also used to minimize development costs, or to decide the distribution of services within a building, as well as other problems which can be modelled by a linear relationship between variables.

However, many of the architectural problems, as mentioned earlier, belong to a class *NP-complete*** problems, which are difficult to optimize, with the result that describing and expressing a global optimum goal is difficult even to approximate.

4.1. AI Approach to Optimal Design

There are two paradigms from AI technology that can be used to help solving ill-structured problems such as design in Architecture. These are called: *Search and Inference*.

The processes of *search* and *inference* in design are not algorithmic in nature, but rather of a fluid *holistic* nature, where all major processes have to be manipulated simultaneously at every stage in design development.

***Optimization provides the mechanism for maximizing, or minimizing, effectiveness in the use of plan layouts, as well as space allocation. An optimization technique provides an engine for design, and also provides a means of simulating the human-decision making process, where utility maximization is the basis for decisions, (see chapter 3)*

*** A non-polynomial programming function, that has no solution.*

4.1.1. The process of *Search* can be described as a process of searching through alternative states of the representation in order to find a state that satisfies predefined criteria. Search strategies can be divided into three main groups [Winston 1984]:

- The first search-technique finds some path or sequence, but not always the best.
- The second group of search-technique finds the optimal path.
- The third group is used when there is an adversary relationship.

The best search method to be used depends on the nature of the alternative/solution state, and how well it has been defined. It also depends on the ability to make judgements along the path of the *search* process (fig.27).

4.1.2. Inference can be used when there is a clear way to deduce new facts from the known facts. This strategy is effective when knowledge can be expressed as axioms.

There are two types of inference, as identified by Charniack 1985: Deduction, Abduction. In Deduction, particulars are derived from general principles, as long as the axioms are true. In Abduction, knowing the percentage of situations with some attributes pertaining to a situation, and knowing the likelihood of both the attributes and the situation, it is possible to figure out the conditional probability of the situation given the attribute. If there are many attributes, then the number of percentages needed to be known becomes astronomical. Expert Systems are logically correct inferences, based on either deduction or abduction theories.

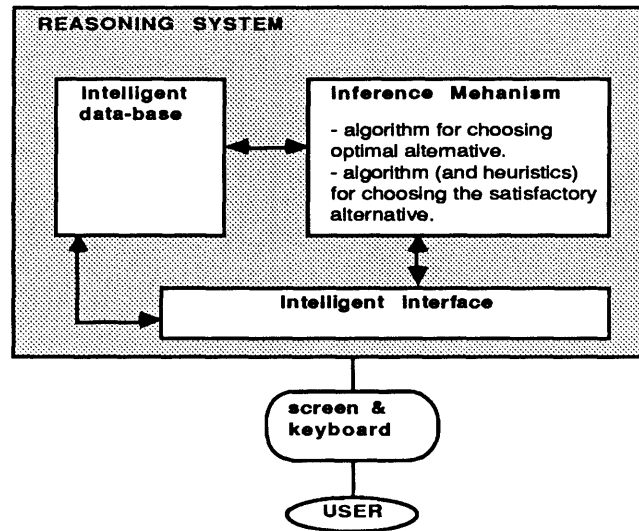
The requirements of a chosen technique, either *search* or *inference*, for use in this context relate to the characteristics of the information required, and the types of the problems and the objectives encountered.

4.2. Artificial Design Systems

These are presumed to be intelligent vehicles for designing purposes, as proposed by H. Simon [1971] and in light of the conceptual ideas taken from the *General Problem Solver* -GPS- [Simon, Newell & Shaw 1957]. These systems are intended to embody intelligent capabilities, where a programmer feeds the system with the needed information, and the system gives results/solutions accordingly.

An Artificial Design System (ADS) is intended to be a "design unit" as such by itself, i.e. it is intended to handle the design process from start to the end. Its *architecture* is based on the concept of modules (to be explained in chapter 3). Through the integration of simulation, generation, and optimization modules, a designer will be able to arrive at what could be called an *Optimized Building* [Simon 1971].

The basic concepts of building an ADS are as follows* (fig.8):



[FIG.14.] The suggested architecture of the "Artificial Design System".

4.2.1. Computational Methods

- Algorithms: For choosing *optimal* alternatives.
- Algorithms and heuristics: For choosing *satisfactory* alternatives.

4.2.2. The Search for Alternatives

This is often a *heuristic search*. Intelligent problem-solving systems in the real world do not merely assemble problem solutions from original problem components, but must search for appropriate assemblies. Therefore, an intelligent system should have the following capabilities:

- General applicability of the proposed iterative, hierarchical, recursive and associative models and protocols, thus allowing application at any level within the design process, and expansion to other sub-systems concentrically, to encompass total building integrity.
- Systematic evaluation of conflicting objectives.
- Modular acquisition and utilization of knowledge from different domains.
- Invoking different reasoning strategies.
- The effective use of qualitative and quantitative knowledge.

* In chapter 3, an implementable version of an artificial design system [CAIRO -Creative Architecture by Intelligent Recursive Optimization], currently being worked on, will be described. This system is intended to help solve the problem of space-allocation, as a given design problem. The CAIRO system, uses some of the AI techniques presented in chapter 1, in order to try representing, on a large scale, the unique attempt of designing an optimized building.

4.2.3. Reasoning Strategies of the System

The prototype intelligent system should have a rich repertoire of reasoning strategies, driven by the designer (and by accumulated experience/knowledge). Thus, the system will be capable, during the solution of a specific problem, of performing the following tasks automatically:

- Test hypotheses.
- Define conjectures.
- Assert, on-line defined, intermediate goals.
- Create alternatives.
- Simultaneously take several alternatives and evaluate them.

Accordingly, this artificial design system should be provided with:

4.2.3.1. Intelligent Databases. The effective use of an intelligent system depends heavily on the availability of an intelligent database, which allows innate reasoning during the research of the database and conversational interaction with the user. This interaction is attained by answering user-directed queries or accepting new elements for the tables of data, and identification of patterns among database elements. The basic form used to represent knowledge and data is that of a "frame" (see chapter 1).

4.2.3.1. Intelligent Interfaces. The database system, described above, should be supported by a simple and *transparent interface* between the designer and the computer. Thus, the system should include:

- Graphic interface with easy manipulation of graphic objects.
- Automatic generation of data-models, describing the graphic objects (icons), and containing all available knowledge regarding the graphic object.
- This *built-in* understanding of the problem characteristics by the graphic objects, to allow the graphic interface to draw conclusions and provide explanations, while the designer concentrates on other creative work.

" 'Elsewhere' is another view -possibly from philosophy- or other 'elsewheres' as well, since the views of man are multiple. Each view has its own questions. Separate views speak mostly past each other. Occasionally, of course, they speak to the same issue and then comparison is possible, but not often and not on demand".

CREATIVE ARCHITECTURE BY INTELLIGENT RECURSIVE OPTIMIZATION (C.A.I.R.O. system)

This chapter is intended to be a description of a computer system under development, where a given design problem, typically described as being ill-defined, is resolved using the intelligent capabilities embedded within the system.

1. OBJECTIVES

The major goal to be pursued is to develop a prototype of an automated design system capable of performing human-design tasks by its ability to :

- Systematically generate a list of alternatives, distinguished by particular trade-offs.
- At the same time, take into account a broad and diverse spectrum of design concerns, design criteria, as well as guidelines of good designs .
- Choose an optimum solution from the alternatives generated.

The assumption underlying these goals, is that the human cognitive apparatus is not particularly well suited to perform all of the previously listed tasks efficiently, except for the choice of optimal solutions* [Flemming 1986]. Therefore, the role envisioned for the proposed system (CAIRO) is that of a "design assistant" who will be able to generate alternative layouts, taking a broad range of constraints and criteria into account. The system is also intended to be capable of adding other criteria (heuristically) in order to finally provide a choice of an *optimum* solution through a chain of substantial recursive optimization processes (see fig.20/21).

The following is a description of ways to represent knowledge about a chosen problem.

2. KNOWLEDGE REPRESENTATION FOR A DESIGNING PARADIGM (SPACE PLANNING)

The chosen design problem, *space-planning*, may be considered a good area for applying AI techniques, since its domain is small enough to be precise about, but also broad enough to discuss different elements of a building from a more general point of view.

*Even though, optimization can, to some extent, be also automated.

Space planning is a good example of an ill-defined design problem. Therefore, knowledge representation for such a problem becomes an essential component for resolving it.

In the following section, knowledge representation for *space-planning* will be discussed in more detail, thus revealing the capabilities of certain AI techniques that may help define and structure *space-planning* in a way simple enough to resolve it.

2.1. Space Planning*

The proposed design problem of space planning/allocation is relatively intractable because of the fact that knowledge about the *design process*, and by extension the *building systems design process*, is incomplete and/or unreliable [Flemming 1978]. The design problem about to be tackled in this chapter concerns the understanding of the nature of knowledge in the domain of "Space Planning" within the context of other evolving domains in science and engineering (e.g. computer sciences).

Understanding the nature of knowledge requires recognizing the problem-domain's significant parameters; how they interact, finding the relationships that describe such interaction, and above all, describing how to represent such knowledge in order to be able to model the actual behaviour of that paradigm.

2.1.1. Description of the problem

"Space Planning" is that aspect of architectural design which is concerned with the physical arrangement of spaces within a building, or a site, to fulfill the requirements of diverse human activities.

The spatial relations between the objects to be allocated serve as basic design variables which define differences between layouts. They are usually represented in an orthogonal structure, which allows to represent the generation of layouts in an abstract manner, followed by the choice of an *optimum* designing model [Flemming 1986].

2.1.2. Background

The first "Space Planning" programs, used the "Operation Research" formulation represented by *quadratic assignment problems* [Gero & Radford 1962], were concerned primarily with the cost of circulation, i.e. traffic efficiently within the plan, as well as other quantifiable constraints for synthesis.

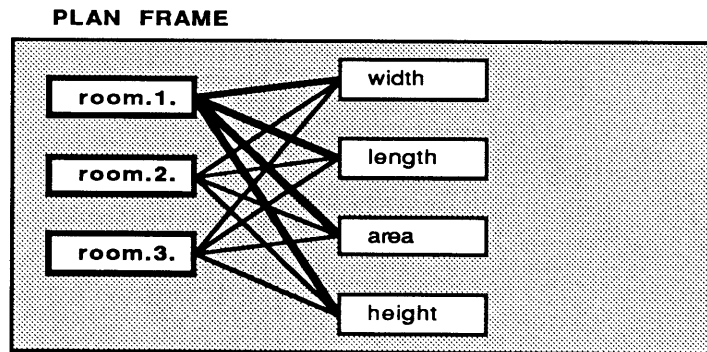
**There are further reasons for choosing this paradigm in particular, as it is generally better understood than some other aspect of building design (despite the fact that knowledge about the design problems is incomplete and/or unreliable, as mentioned before). This paradigm permeates building design at most of its stages, and thus provides a rich context for the definition and investigation of problem domains with various degrees of complexity. Most important, it may establish connections with other disciplines which also deal with Building Systems Design aspects.*

Garson & Steadman [1970], followed by Mitchell et al.[1976] and Flemming [1978] continued the work on finding an optimum *plan layout*, (through the exhaustive enumeration of solutions to layout problems) using various optimization techniques. This led, eventually, to the development of *automated space-planning systems*.

Automatic space planning systems have provided a *test-bed* for the exploration of the interactions between human and artificial problem-solvers. It was perhaps the first practical attempt to apply explicit techniques of AI technology* (eg. GPS).

2.2. Knowledge Representation for Proto-type Plans

A prototypical floor-plan is a convenient way of representing much of what is known of the general *plan* and its entities. Borrowing Minsky's "frame" theory (see chapter 1), a *plan* can be seen as a complex of relatively fixed relations between variable elements (fig.15.). A proto-type plan will have a slot in the frame for various elements and attributes, and a range of acceptable values for those slots. This process, according to the "frame" theory, parallels the expectations that people bring to most of the familiar concepts of design.



[FIG.15.] Input/Output data

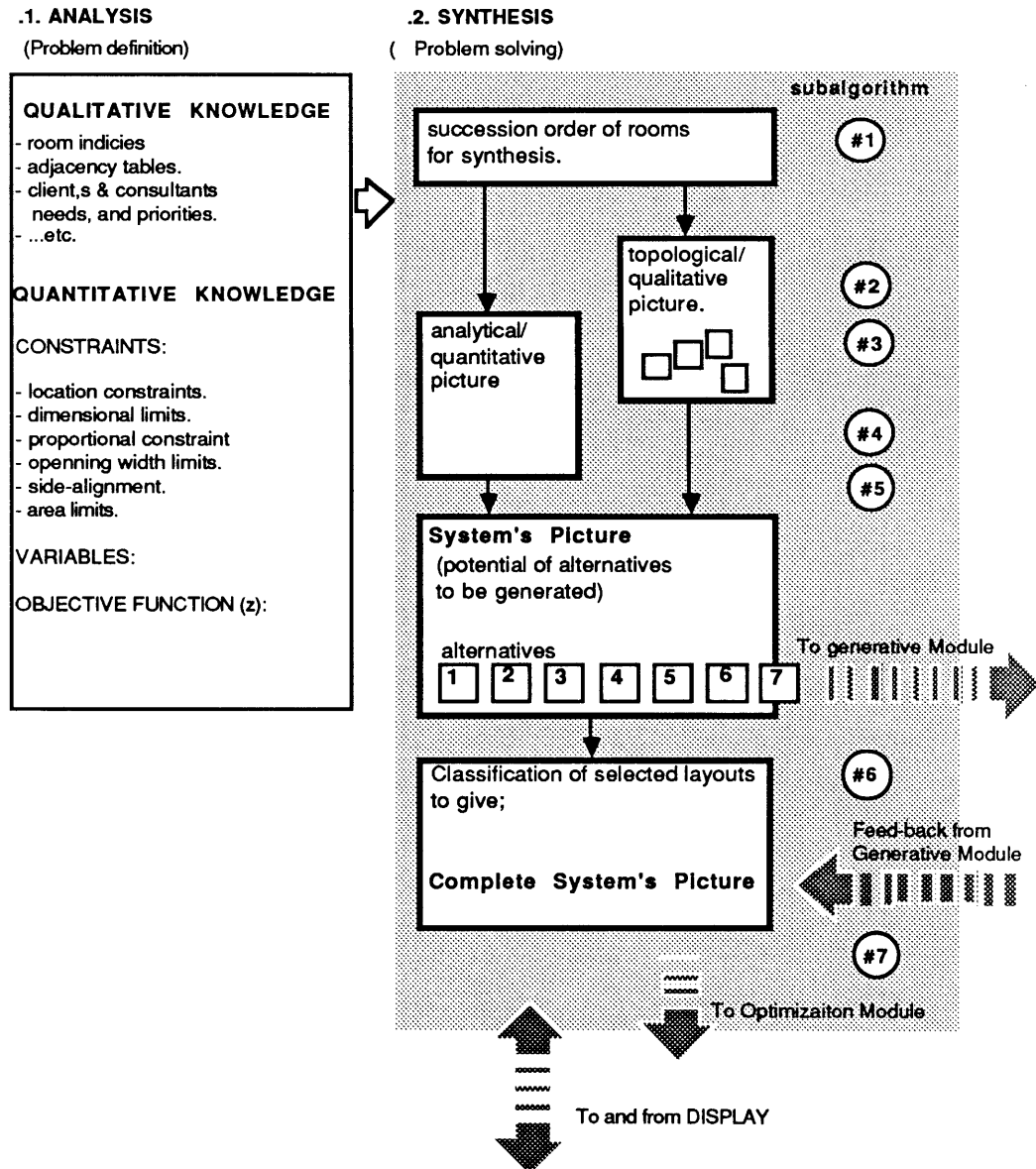
In this context, an attempt to define the design process is carried out in the form of a procedure** consisting of a set of unambiguous rules which specify a sequence of operations that provide a solution to the design problem (e.g. allocation). The design rules and constraints are subject to modification and change at any stage. This process of definition on rule setting is

*H. Simon described this paradigm to be solved by a GPS that decomposes the ill-structured problem into several abstracted well-defined problems.

**A procedure is a format in which certain steps for solving a problem are expressed. An algorithm is a procedure, and so are the rule-based inferences and heuristic search.

2.2.1. Knowledge Representation for Input/Output Data

Knowledge Representation achieves great conceptual clarity by drawing a clear distinction between, on the one hand, quantitative and continuous properties of a solution, such as the dimensions of the spaces to be allocated, and on the other hand, some of its qualitative or discrete properties (particularly the spatial relations among the allocated spaces). **fig.17**



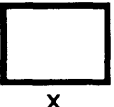
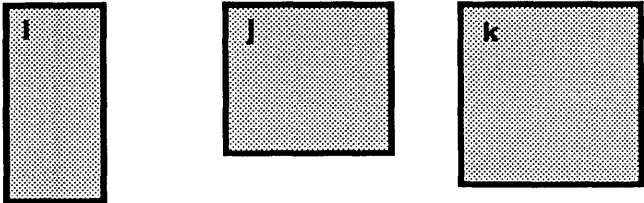
[FIG.17.] The Input & Output data represented as qualitative and quantitative knowledge (for the Synthesis and the Analysis processes). [El-Quessny 1987].

2.2.1.1. Qualitative Data

- Consist of elements and relationships that have absolute definitive value (on both the ordinal and the nominal scales), as well as assumed values.
- From the various combinations of the qualitative data, only one solution/model is selected, by imposing certain selection criteria. These combinations embodied in a "system's picture"*, satisfies the selection criteria conditions to give a *qualitative/topological answer*.

Examples of Qualitative Input Data (fig.18.):

- The table of rooms (i j & k) and the relationship between them described as a *semantic network*.
- Qualitative preference of certain values (e.g. subjectively, giving scalar measure to the preference of "adjacency" among rooms).

room number & room name	degree of adjacency between rooms		
	I	J	K
I bed room		9	8
j living room	4		6
k dining room	7	6	
room number 			
I 2.70 x 4.50			
J 3.60 x 3.60			
K 3.60 x 4.00			

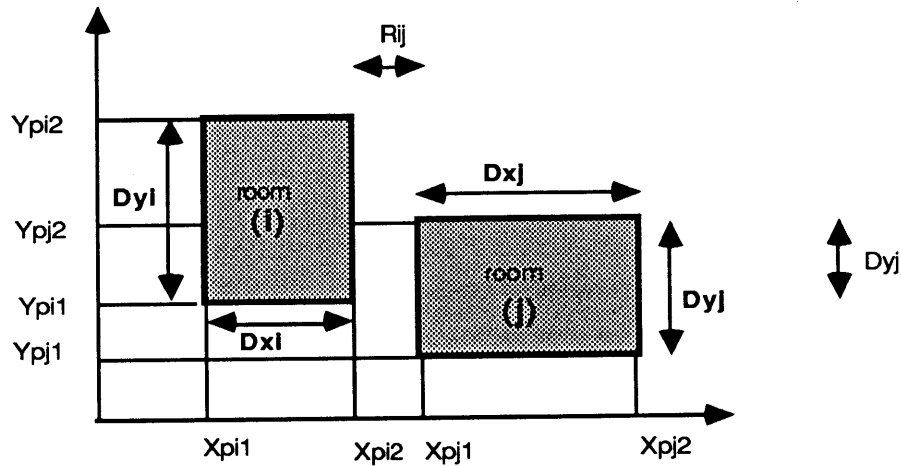
[FIG.18.] Qualitative data representation

2.2.1.2. Quantitative Data

- Consist of elements and relationships that are measurable and can be expressed by a symbol variable. A value to a given variable determines it as a physical entity. Mathematical analysis can also be applied to the variables.

*A system's picture is a term taken from Ellin [1979], for representing a layout as a geometrical presentation. It could also be called the *geometric picture* of a layout. This term is used when we do not use the terms of the system's approach .

- From the various sets of values and variables that satisfy the quantitative data, only one solution/model is selected by imposing certain analytical selection criteria (e.g. minimizing a given objective function). This set of values is incorporated in the "system's picture", and it is called *quantitative/analytical answer*. (fig.19)



- k** total numbers of rooms.
- L_{ij}** sharing length of walls (room(i) - room (j)).
- R_{ij}** distance between the two lines on which two walls are allocated.
- l_{aij}** index expressing the evaluation of item (a) on the relationship between the two walls of room(i) and room (j).
- $P(r)$** total number of columns when the number of rooms are R.
- D_{xi}** x-direction (E/W) width of room.
- D_{yi}** y-direction (N/S) width of room.
- $X_{pi}(1)$** x-coordinate of western wall of room.
- $X_{pi}(2)$** x-coordinate of eastern wall of room.
- $Y_{pi}(1)$** y-coordinate of southern wall of room.
- $Y_{pi}(2)$** y-coordinate of northern wall of room.

[FIG.19.] Quantitative data representation.

Examples of Quantitative Input Data:

- The set of **VARIABLES**:

- Drawing dimensions,
- Rooms dimensions,
- Location coordinates of rooms,
- Opening width in wall,
- Distances (gaps) between rooms.

- **CONSTRAINTS:**

Location constraints of the rooms,
Dimensional limits of the rooms,
Proportional constraints of the rooms,
Area limits,
Others.. (opening width limit; non-overlapping of rooms; side alignment).

- **OBJECTIVE FUNCTION(s):**

Minimizing or maximizing; building costs, built area, number of rooms in a given area, amount of energy consumed in a building, etc.

2.2.1.3. Using Qualitative and Quantitative Knowledge; Using both qualitative and quantitative data in the process of knowledge representation may further complicates the problem since, on the one hand, existing quantitative knowledge is inadequate to address the process as a qualitative whole, and on the other hand, existing qualitative information may ignore a valuable source of insight when considering the problem. Therefore, a *Hybrid approach* is considered if one is to achieve any degree of practical success.

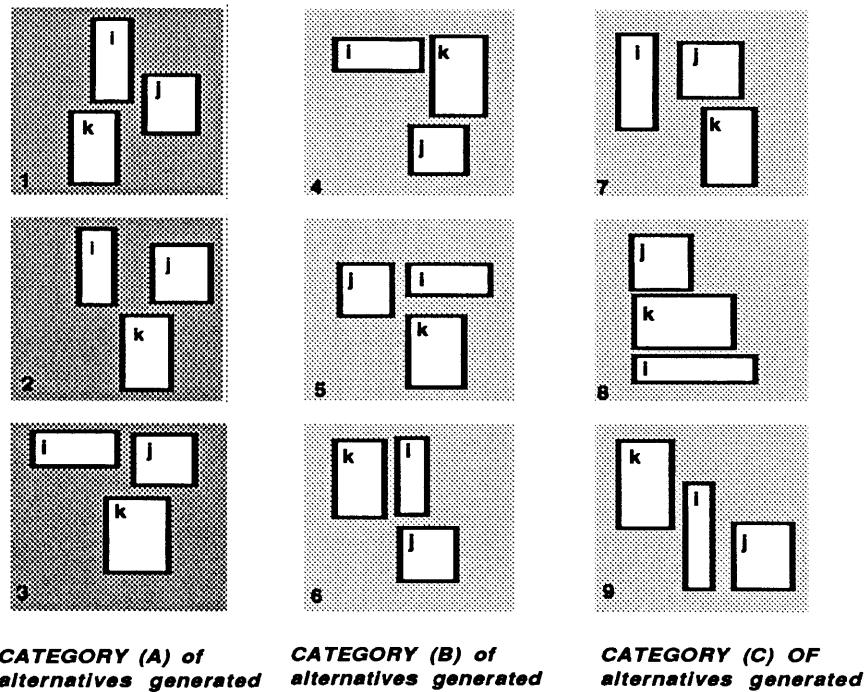
2.2.2. Knowledge Representation for the Generative Model

The problem of space planning stresses the importance of using a formalized representation for properties of spatial relations among the spaces to be allocated, and calls for explicit specifications of the necessary and sufficient conditions under which such representations are to be considered *well-formed*, or *syntactically correct*. That is, every object that satisfies these conditions represents a solution. In the enumeration of the solution sets, these representations play a crucial role in two ways (fig.20):

2.2.2.1. Each paradigm is an **abstraction**, since it supresses by definition certain properties of the solution it describes. Different solutions can, therefore, have a similar representation, and each representation consequently describes not merely a single solution, but an entire class or subset of solutions.

In a suitably selected representation, the set of solutions is divided into a finite set of subsets which can be enumerated by generating a *well-formed* representation as *objects*. The relations recorded by that representation are the crucial design variables that define differences between the subsets, and account for the variations generated during enumeration.

The generation of solutions/alternatives is itself based on a set of construction rules, and explicit proofs are required to assure that the set of *well-formed* representations is both *complete* and *closed*, under the application of these rules. i.e. every sequence of the rule applications creates a *well-formed* representation, and every inductive proof of these results is straight forward if the rules are formulated as *recursive re-write rules* [Flemming 1986].



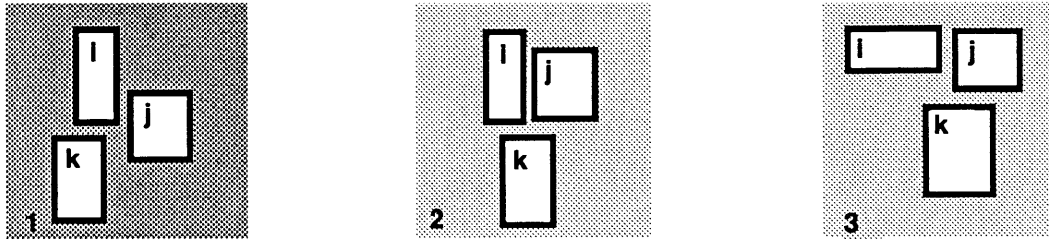
[FIG.20.] By applying first round Optimization to the alternatives generated, they are categorized according to their meaning as well as their potential for satisfying program requirements.

2.2.2.2. Each representation must record the spatial relations characterizing the solutions it describes accurately enough to allow for an explicit formulation of the *dependent* as well as *independent* constraints which govern the dimensions of the allocated spaces, and change as the spatial relations between spaces change [Yamagata & Aoki 1986].

After a generic representation has been generated, a particular member of the subset of solutions, described by these representations, can be found. That subset of solutions is generated by formulating all constraints imposed on the dimensions of the allocated spaces, and by computing a set of dimensions which simultaneously satisfy these constraints.

2.2.3. Knowledge Representation for the Optimization Model

In this section, H. Simon's, "science of design" approach will be considered, which proposes to treat design theory as a new discipline of optimizing an objective function over a region of alternatives bounded by constraints. Here, the solution process is represented by a sequence of tasks which are also defined in terms of its sub-goals (fig.21).



[FIG.21.] Alternative (1) is chosen among all the other alternatives of category(A), of fig/20, to be an optimum solution by applying further substantial recursive optimization procedures after the process of alternatives-generation, in fig.20.

3. APPROACH

This section addresses the problem of how to approach previously represented knowledge in order to be able to encode it in the system. Generally, this task may be divided into the following steps:

3.1. Analysis

Analysis is that process by which a problem is defined in a way suitable for the application of a solution which corresponds to the problem definition, i.e. the development of a definition for the problem. Analysis moves the information up in the plan by determining aspects of the behaviour of the composite systems from appropriate aspects of the behaviour of its elements and their interconnection. Thus, up until here, the paradigm follows, recursively, a decompositional approach in tackling big problems, until it reaches a level defined well enough to start design, i.e. start generating solutions.

Analysis proceeds as follows:

- Identification of all sub-problems.
- Listing of all required and available inputs.
- Listing of all desired outputs.

Subsequently, the given layout is transformed into an equivalent symbolic form, which is called the *system's picture* of the layout. For example, the rules of transformation in the CAIRO system are:

- Every wall of the layout shall be represented together with its centre-line.
- All rooms shall be enumerated, and each room will be identified by a number index.
- Each layout is provided with a set of coordinates (x,y), so that the relative location of the layout can be given in relation to the general layout.

3.2. Synthesis

In the synthesis of a problem, successive steps should correspond to the problem solution. Synthesis moves information in the plan, as the descriptions of the parts of the allocated spaces mechanism are refined.

In the CAIRO system, the resolution of the problem at hand proceeds along two major paths, namely *reduction* and *abstraction*:

3.2.1. Problem Reduction

Problem Reduction is a strategy which reduces the overall problem to a set of sub-problems, and identifies a goal for each sub-problem (subalgorithms of fig.17).

Control is passed from one sub-problem to another, depending upon what goals remain to be achieved and the relative priority of the sub-problems. This strategy is essentially similar to that followed in many engineering problems. It is particularly appropriate when the solution process is reasonably well understood, so that realistic goals can be established.

3.2.2. Abstraction and Reimplementation

Abstraction and Reimplementation is the approach used in order to analyze the source program first, and in order to obtain an abstract high level *understanding* of the relationships between entities of a plan [Waters 1985].

The program is then reimplemented in the programming language used (LISP), based on this *understanding*.

4. THE ARCHITECTURE OF THE SYSTEM

The C.A.I.R.O. system is designed to permit the user to build up a particular architectural/engineering problem-solving capability by combining *modules* in a flexible manner.

In conventional programs, knowledge is implicit, but in this system, the knowledge sources are explicit to the task, while the modularity of the components of the program allow for some

flexibility. It also minimizes the effort required to incorporate new knowledge by using an object-oriented-programming environment.

The basic architecture of the system is based on a highly distributed and loosely coupled set of programs, conceived as modules of implementation (fig.22.). The key feature of the system is an object network language that simplifies the construction of the object and the constraints, functional libraries, and user interface packages.

4.1. The Framework

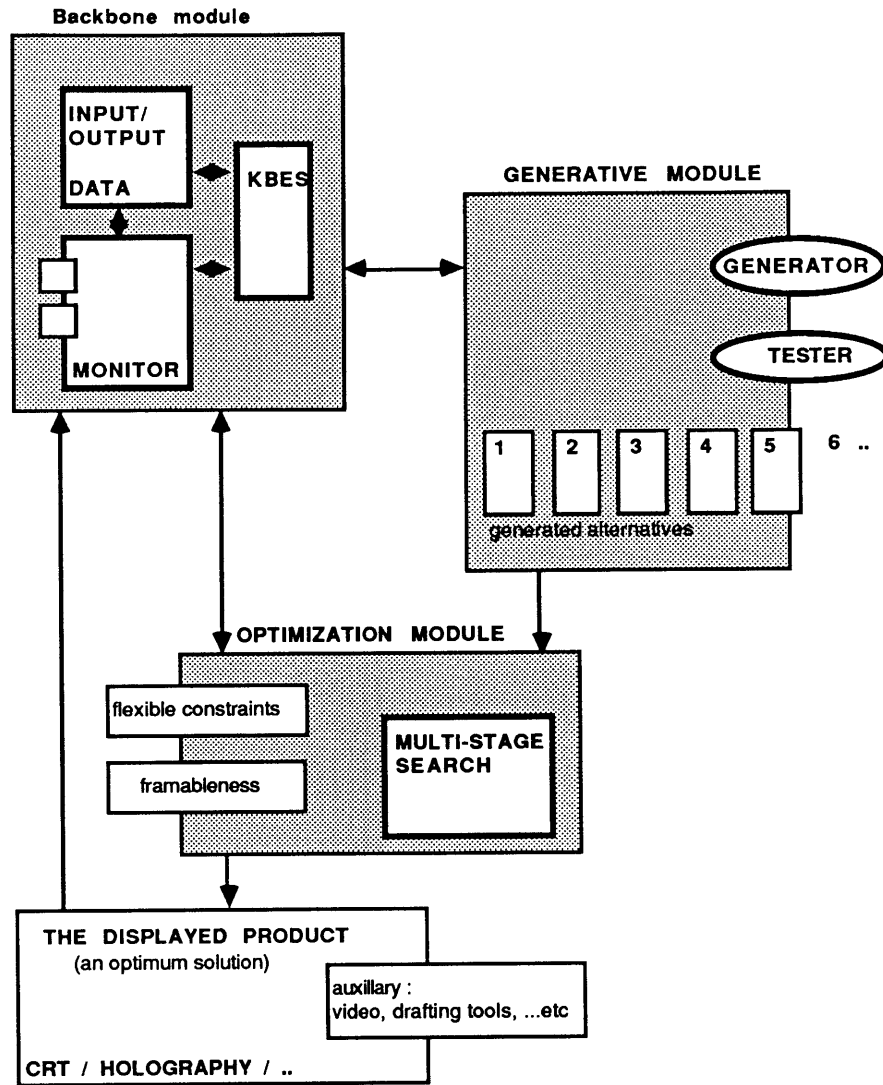
The current choice for the system development framework is a *blackboard* model framework (Nii 1982). This framework provides an excellent paradigm for knowledge intensive problem solving, requiring multiple cooperation, communication, and intelligent problem solveing. As such, it simulates well the nature of the architectural design problem-solving process (explained in chapter 2).

Problem solving behaviour is based on the coordination of individual processors denoted as *knowledge modules (KM)*. Each of these is used to develop a different hypothetical solution to the problem at a different level of problem abstraction (fig.25.). Communication is required to generate, combine and evaluate the various hypothetical solutions (fig.26.).

The central data structure, "*the context*", is used to support this communication, and is operating within the same framework.

4.2. An Overview of the System's Components

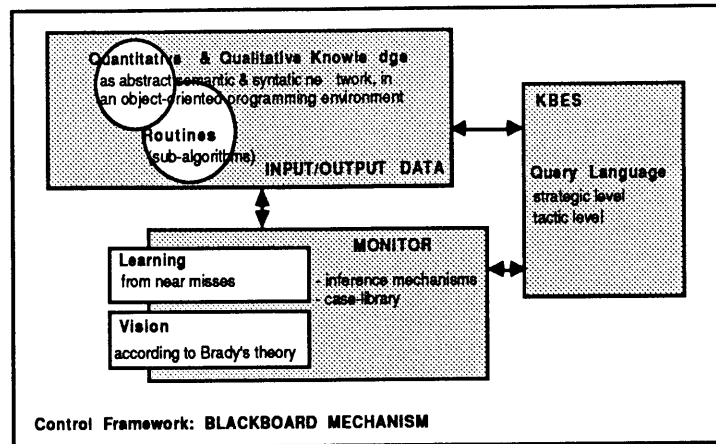
This section is an overview of the different modules of the proposed system, and is better described by illustrations to facilitate easier understanding. Wherever the illustrations fail to give a fully comprehensible picture of the inner components of the different modules, the text should help to explain such deficiencies.



[FIG.22.] Creative Architecture by Intelligent Recursive Optimization (the architecture of C.A.I.R.O. system).

4.2.1. Back-Bone Module (Base-Module fig.23.)

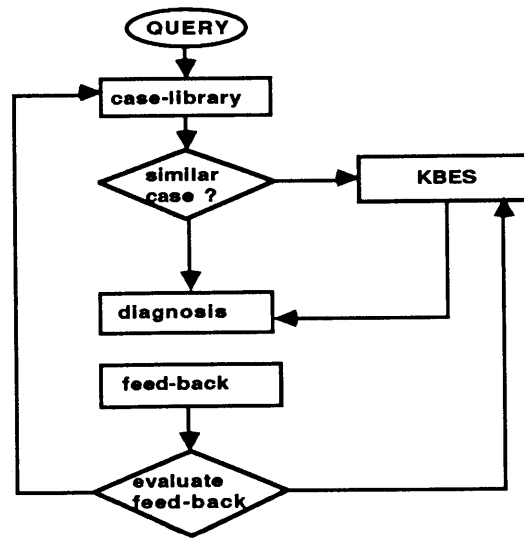
4.2.1.1. *Input/Output Data:* Functional and physical descriptions of the spaces to be allocated are expressed as a *semantic* and *syntactic* network of relationships in an object-oriented-programming environment. For example, the model employs the degree of adjacency as an index of the positional relationships between spaces (rooms/plans), so that the higher the degree of adjacency between two spaces, the closer they may be allocated together. In addition, spaces are also described in terms of their relations to each other (eg. space-1 is east of space-2, and space-3 is south of both of them).



[FIG.23.] The BACK-BONE Module, with the Input/output data cell shown related to other cells.

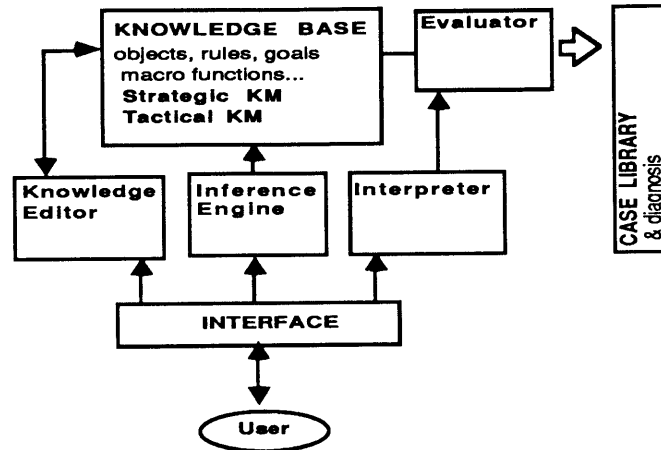
4.2.1.2. *Monitor*: The relationships established by a semantic network between entities of a proto-type *plan* are represented to the system as quantitative and qualitative data (as explained previously), and are stored in a case-library (fig.24.), so that whenever variations of similar cases are desired to be optimized, these relationships needed not to be called up again.

Machine learning modules (from near misses) and *vision* capabilities may be added in order to monitor the input/output data process.



[FIG.24.] Case-library configuration

4.2.1.3. A Knowledge Base Engineering System (KBES): The basic architecture of the KBES is depicted in figure.25. Knowledge is aggregated into sets of independent knowledge modules (KMs), each of which addresses one sub-problem:



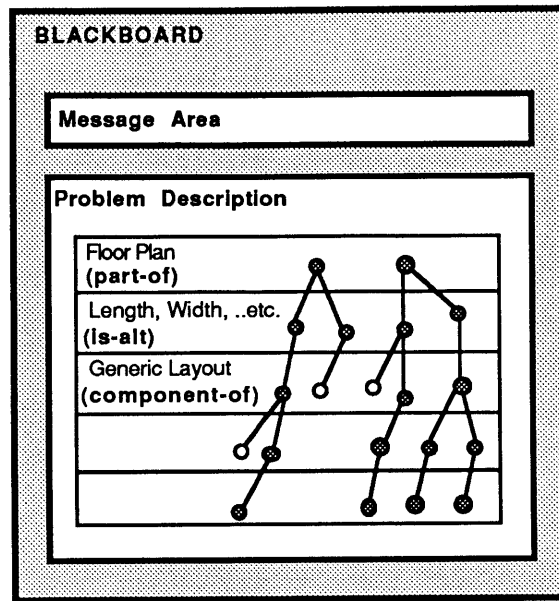
[FIG.25.] The KBES architecture

- Strategic "Knowledge-Modules" are used to control the problem solving process (qualitative and quantitative).
- Tactical "Knowledge Modules" are used to carry out the actions planned by the strategic KMs. The tactical KMs perform one problem-solution at a time. Each module may use a combination of heuristic and causal knowledge and may interface with algorithmic processors, or other design tools.

The *context* of the system is also implemented as a *blackboard* which contains a message posting area, information about the state of the problem solving process, and a description of the current design*. The message area provides a mechanism for the various KMs to communicate with each other, as well as place explicit requests for information and actions.

The *blackboard* has a static organization, containing data slots for all KMs, and is organized into a hierarchy which represent different levels of problem description abstraction. These slots contain descriptions of the design hypothesis as well as design alternatives currently under consideration. As the solution proceeds, the various KMs generate, update, and evaluate independently the solution hypothesis.

*The overall organization of the blackboard system does not use any requirements on the development framework used for the KMs.



[FIG.26.] Blackboard model for KBES framework.

In that *context*, the solution is comprised of nodes that are connected by means of relational links. Through these links, at a certain level in the Blackboard, the nodes can inherit information from other nodes, normally from the higher level, connected by these links. The relationships between nodes can be described as [Howard 1986]:

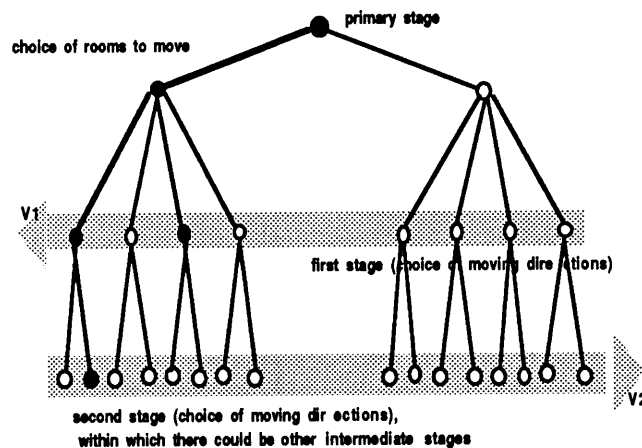
- **Generalization** involves the grouping of a set of generic nodes into a "super" node (denoted as *is-a*). The inverse process is **specialization**.
- **Classification** involves the grouping of a set of individual nodes into a generic node (denoted as *instance-of*). The inverse process is **instantiation**.
- **Aggregation** involves the construction of an object node from its constituent parts. In the context of building design two types of links are used to denote aggregation (i.e. *part-of*, aggregate systems into sub-systems of other systems. Combining surface elements in sub-systems is denoted as *component-of*).
- **Alternation** involves the definition of the alternative nodes for a generic type. (denoted as *is-alt*).

Once the input data has been formulated and fed into the system by means of substantial abstractions, the architect will be able to converse with the terminal to check the solutions generated by means of the user-interface mode.

4.2.2. Generative Module

The general approach on which this module is the *generate-and-test paradigm* which separates the *generation* of a solution from its *evaluation*. One advantage of this approach is its generality and conceptual clarity. The strict separation between *the generator* and *the tester* allows us to design a domain-dependent *generator*, defined in purely syntactic terms. With this separation, the system will be capable of handling the many relations that exist normally between "design" and "performance" variables in building design (eg. the performance variable *daylight-coefficient* in a room, is dependent on the following variables: room dimensions, materials, etc.). Conversely, each of these design variables influences other performance variables as well.

The *generator* can be complemented by different *testers* that use the domain-specific knowledge about the quality of the layouts to evaluate those layouts. The main function of the *tester* is to prune the search tree for computational efficiency and eliminate less favourable solutions, so that it passes only a manageable and limited number of alternatives.



[FIG.27.] The Pruning process through the "multi-stage search tree".

4.2.3. Optimization Module

In this module, two techniques are used to develop a more robust choice of a solution from the set of solutions generated by the *Generative Module*. These two techniques are:

- **Framableness** [Yamagata 1986]: To develop a better solution from the architectural point of view in terms of view, adjacency, etc. The spaces to be allocated are defined by the geometrical relationships of beams, columns, and walls.
- **Flexible Constraints** [Aoki 1986]: This is a method by which initial constraints can be constantly modified and re-evaluated throughout the design process.

"This experiment has provided all the information it can. True, it must be finished, but the planning has been done, and the circumstances anyway are so special that the actual execution of the work would not particularly relevant to other problems of cooperative building".

H. Fathy, Architecture for the Poor **4**

EPILOGUE

The most reliable way to anticipate the future is by monitoring contemporary events that, collectively, dictate future trends. This epilogue is intended to be an introduction to what, eventually needs to be done in terms of future *additions* and *alterations* to the system, rather than being offered as a conclusion. It also speculates on the impact of "*Automation*" by means of *intelligent systems* on the field of Architectural Technology.

Finally, the aim of this research is not dogmatic *persuasion*, but an effort to highlight certain feasibility in order to obtain new insights into the application of AI techniques in architecture and building design, and to help in the assessment of a new design hypothesis that may change the practice of architecture.

1. FUTURE DEVELOPMENT OF THE C.A.I.R.O. SYSTEM

The proposed system is intended to be used as a tool that will permit the designer free reign of his intuitive and creative powers (subconscious as well as conscious), yet provide him with an immediate evaluation of the effects of his tentative design proposals. The proposed scenario for the *CAIRO system* is as follows:

- The designer wants to design a building.
- The system asks about the different functions in the building and the design criteria.
- The system tells the designer about the various options by which he has to meet the stated functions, with the pros and cons for each option.
- The system offers to display previous examples of similar buildings stating what is negative or positive about them. The user can then modify the criteria accordingly.
- The system offers design alternatives, and the user picks one or more suitable ones.
- The system starts to detail the chosen alternative(s). Whenever the system encounters a decision choice against which it is indifferent, it asks the user, while at the same time demonstrating the consequences of each.
- The system produces a drawing of each alternative on a "mouse-sensitive" screen, in order for the user to modify any part of the proposed solution. The modifications are propagated automatically throughout the system, and the drawings are updated accordingly.
- The user may mention an additional feature (function), and the whole design is changed accordingly to match the new feature.

The CAIRO system will also be able to perform as a generic architectural tool for drafting, for the generation of working drawings, and production management. It should be able to perform the following tasks:

- Assist the architect's process of visualization: Scale and render images with various sources of light and shade. It should also recognize sketch drawings.
- Quantify and qualify for architect's evaluation measurements and calculations: Technical, relational, and geometrical aspects of a building, in an explicit and documented manner.

Eventually, this system will be using the full scope combination capabilities of multi-media* such as: sketching, natural language description, and gestures. It should also be supplemented by various modules of intelligence:

- First, a *recognition system*, which automatically identifies occurrences of stereotyped computational fragments and data structures in programs. This recognition system should also be able to identify the above mentioned fragments and structures, even though they may be expressed in a wide range of syntactic forms. It does so systematically by using a parsing technique.

- Second, the system should have the ability of *learning*, so as to confirm the concept of *intelligence* (discussed in the preceding section).

Machine learning to date has been limited to learning new expressions in some more or less a well defined language. However, the proposed system should be able to learn concepts using *positive examples*, i.e learning from "near misses" [Winston 1984].

- Third, the system should be able to build semantic network descriptions of shapes for a Vision Capability [Brady & Asada 1984]. This semantic network is transformed into a set of associative triples [Doyle & Katz 85], and inputs the learning component of the program. The learning program is a substantially modified version of Winston's ANALOGY program [Winston 1984], which can learn shape models. The output of the program should be a structured production rule that constitutes a procedure, recognizing subsequent instances of a learned concept.

Accordingly, the system is expected to maintain several different representational hierarchies such as: number-symbol, kind-of, and structural approximation. However, the need to learn shapes at every point in the hierarchy imposes certain *learnability* demands on the

*Early attempts at sketch recognition and graphical inference, informed with contextual knowledge, met with limited success [Negroponte, 1970]. The Sketchpad program [Sutherland 1963], and Sketchpad.3 [Johnson and Weinzapfel, 1971], enable the user to construct diagrams of constraint networks, using a light pen

vision system that generates the representation. Therefore, the hierarchical nature of the representation is maintained by using *structure mapping* as a generic process for learning and recognition. This could be realized by means of the following:

- A language-independent graphical representation for programs and programming structures which pertains to many syntactic features of programs [Zelinka, 1985].
- An efficient graph *parsing* algorithm.

2. LEVERAGE OF "AUTOMATION" IN ARCHITECTURE

This section is intended to be a discussion of the leverage of automation in the architectural profession regarding two major aspects: Changes that take place in an architectural office (e.g. tools, organization, ...etc.), and changes that take place among individual architects themselves regarding behavioural and professional role aspects.

2.1. CAD and Future Patterns of Architectural Practice

The continued development of cheap and powerful interactive computer graphic systems will soon make it possible for almost any architect to work with sophisticated integrated CAD systems. The effect of this development on the pattern of architectural practice is expected to be revolutionary. Graphics terminals will replace drawing boards. Video-discs, or similar media, will replace drawings. It will be possible to undertake projects within a much more compressed time frame. The ability to integrate sophisticated analysis and optimization techniques into the design process will make it possible to maintain complex as well as subtle design constraints.

Many CAD systems have evolved from new sophisticated programs as vertically integrated sets of software, combining both analysis and graphical capabilities for a single engineering problem solving task. In a move to integrate computer use throughout the design process, a horizontal integration of applications across domains is also underway.

This integration addresses the problem of unimpeded and rapid information flow as well as communications problems which are a major aspect of an interdisciplinary design process. Centralized database management provides the basis for linking various design applications into an integrated system.

2.2. Forecasting Change and Effect

There are two main problems which make investigation of the effect of new intelligent CAD systems difficult. First, there is the probability that intelligent computers will so alter the

normal design process that any speculation, based entirely on conventional practice, will cease to be reliable. Second, there is the prohibitive cost of setting up working experimental systems to provide the necessary hands on experience and feedback for testing new systems.

Nevertheless, this should not stop us from anticipating the future, especially in the light of past simulation experiments carried out by other researchers under the same conceptual assumptions, and the promise shown by these experiments.

2.2.1. Effect on Individual Designer

2.2.1.1. Stress: This is the most important effect which has emerged from simulation studies [Cross 1977]. All designers using intelligent CAD systems will find work in this area very stressful, due to the speed and accuracy with which the designer will be expected to work, and has to respond to the computer's precision. Also designers will have to maintain the integrity of, mainly, computer-aided decision making (precision, error, corrections, etc.) on a continuous basis, i.e. effort to upgrade.

2.2.1.2. Intensification of Work Rate: In this context, reference is made to a study by the Department of Labor in the USA [1984], which found that a designer, doing a job in the aerospace industry, spent 95% of his time performing reference work, and only 5% on actual design decision making. Therefore, the introduction of intelligent CAD systems should eliminate routine reference work, and actually intensify the decision-making rate of the actual design effort by up to 20% [Winston 1986].

2.2.1.3. Reduction of Staff: This is, apparently, the only quantifiable *benefit* that was found in a *cost-benefit analysis* for installing an intelligent CAD system at an architectural office*.

2.2.1.4. New Tasks: In contrast to all the previously mentioned potential *threats*, there are also many potential *promises* arising from the use of intelligent systems as efficient operators of automated design processes which will optimize rational aspects of design, whilst liberating designers from many mundane activities, and allowing them to pursue creative and intuitive aspects instead.

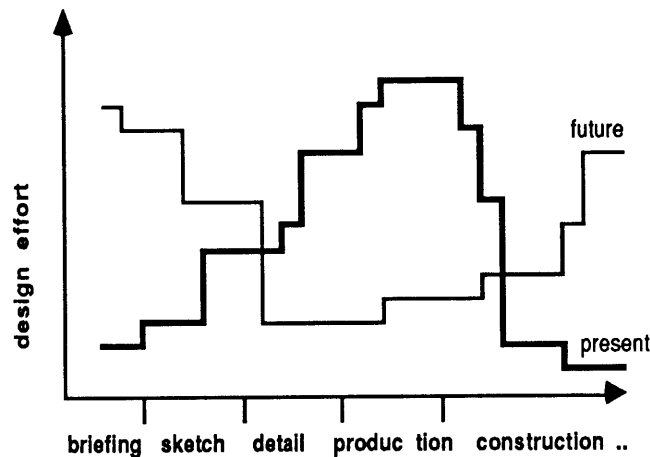
Hence, an important new professional role will emerge: that of the computer-systems' designer. People filling this position will require computer skills as well as domain-specific knowledge about the project at hand, in addition to their conventional architectural skills.

* Based on a similar experiment in the implementation of CEDAR system in the Department of Environment -design offices [Chalmers 1972], the increase in productivity is necessary to justify the per capita investment, and running costs of such an intelligent system was found to be 1.5 to 2 times more than that performed by the traditional method.

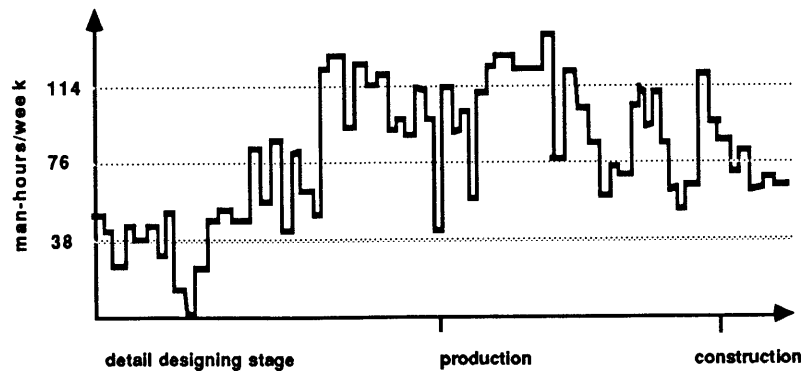
2.2.2. Effects on the Design Process

According to Esher & Llewellyn [1978], there are three main areas of fore-seeable changes in the building industry:

- **Building Management:** Changes from current patterns of traditional "task" operation towards operations offered by package-deal builders and industrialized building-systems manufacturers.
- **Building Construction:** Improvement of on-site working conditions as a result of the introduction of *robots* and continued evolution/change from *labor-intensive* to *capital-intensive* production.
- **Built Form:** Tending towards the incorporation of greater adaptability and flexibility in both exterior and interior building elements.



[FIG.28.] This illustration shows the expected inversion of pattern of the design-effort over the total brief-design-build-use processes [Cross 1977].



[FIG.29.] A record of the pattern of the design-effort of a typical project. (from a study of the "building time-table" by the Building Economics Research Unit -1978).

2.2.2.1. Briefing: Intelligent CAD systems may help to provide a *systematic approach* to design problems by enabling a much *closer fit* to be achieved between schedule of accommodation and the actual activity pattern to be accommodated, thus resulting in significant space savings (as in the case of space-allocation discussed in chapter 3).

2.2.2.2. Production Information: Automation of production information procedures, such as automated selection and combination of standard construction details may help reduce total design time of, for example, the structural frame of a building, possibly by 10 to 20 times [Thomson & Hughes 1984]. New *Patterns of Organization* of the design process must inevitably arise from such drastic changes [Cooley 1972; 1973].

2.2.2.3. The Management Pyramid: Intelligent systems would clearly bring about major changes in the method of working and in the composition of a typical design/management organization (e.g. an architectural office), confirming the concept of *systems building* approach (fig.31).

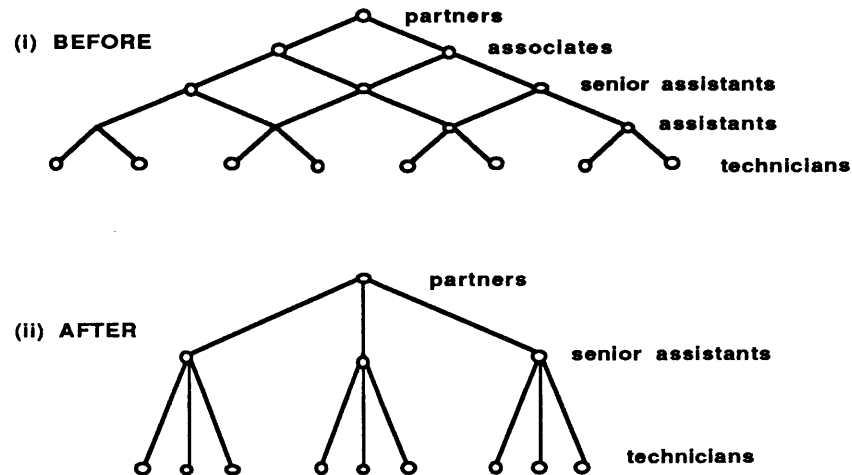
Automation will also introduce a group of new roles (e.g. programmers, consultants, etc.), that have as yet not been completely identified in architecture. This group will establish its presence at research conferences and meetings, where architecture is rarely -if ever- represented.

2.2.2.4. Roles and Relationships in the Building-team*: It is not only the architect whose job will be fundamentally affected by intelligent systems, but roles and relationships of the different participants of the building-team. At the moment, the architect tends to play a central role in coordinating the activities and design inputs of other consultants. Automation will allow continuous, rather than intermittent collaboration among participants with intelligent systems being the medium which will make this collaboration more effective.

Even without the introduction of large-scale comprehensive intelligent computer systems in the building industry, new relationships and patterns of communication among participants in the design process are expected to accompany the growing use of commercial time-sharing computer systems and data-base technology. However, the current professional barriers which tend to restrict the potential for interprofessional collaboration will limit further developments of this kind;

**Schon [1969], argues that the transition to automation in the post industrial society, will inevitably bring about a total revolution in "Design". The most important feature of that revolution is that design will shift towards industrialization, i.e. from product to process, and from component to system. Hence, systems design will become " a central corporate function", and will be extended to total systems design approach*

Manufacturers of building components, for instance, will use a central building-industry intelligent system to gather data on trends in component selection in order to keep abreast of these trends and to modify their component ranges accordingly.



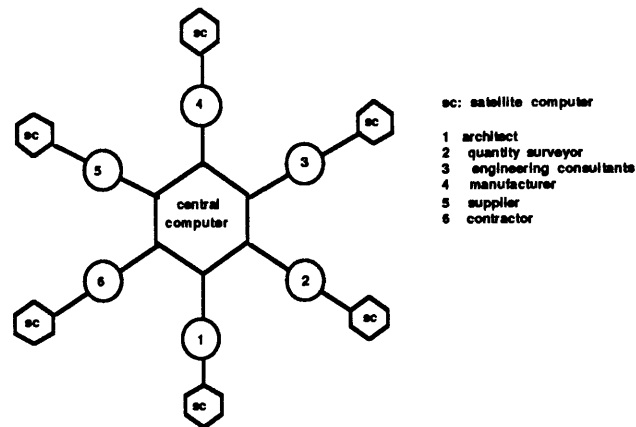
[FIG.30.] The effect of automation on the management pyramid [Mumford & Banks 1977].

2.2.2.5. User Participation*: By making the design process more open and explicit through the introduction of intelligent interactive systems, the way will be open for a wider range of participants to contribute to that process for users of buildings in particular, who traditionally have not been allowed little, if any, participation in that design process (see also fig.4).

As architectural technology and intelligent computer applications evolve towards each other, as has been suggested by Negroponte [1975], among others, there is a possibility of further development for a "*responsive architecture*", in which any adaptable environment will be enhanced with machine intelligence, i.e. the building might get to *know* its occupants, and change its facilities to meet their requirements and thus, adapt its behaviour to suit theirs**. Living in a building and designing that building would, therefore, become one process, and design, as a separate process-activity-job, would cease to exist.

*Mitchell [1979], argues that potentially, some intelligent computer systems will result in an opening-up of architectural and urban design processes to wider, and truer, participation, by making it possible for non-specialists to comprehend and directly manipulate quite powerful models of the environment.

**The "Architecture Machine" group [Negroponte 1971] have claimed that they could take a step towards allowing the urban dweller to participate in the design of his own environment by multiplying the availability of design services. They also proposed to increase the number of people to whom design services can be made available. (This may seem to be a logical implication of their work towards a "robot architect".



[FIG.31.] Representation of a typical model of an information system for the building industry, based on a central computer. Aside from having access to the National Central Intelligent Computer, the participants in each project are often assumed to share common files accessed through, and linked by satellite computers -sc-. (Note that the computer occupies the central role that architects have traditionally assumed to be their own role as the "leaders of the building-team".

3. AN OVERVIEW

In this research, a *design hypothesis* has been proposed and, derived from it, a description of the uses and organization of computing in design. The computer is not a consequence of the theory, but is rather an application of that theory. The *CAIRO System* is considered as an interesting ongoing experiment, which may expect us to believe that designing with intelligent computers is more robust than present manifestations of that theory in existing computer software.

It may appear that this thesis contraverses the interest of architects. The author does not believe this to be true. On the contrary, it is asserted that the tools and techniques that can be offered to the building-team by intelligent computers can only increase the level of professional abilities, the predictability of designs decision-making, and respect of the public at large. The new methodologies which will flourish as a result of introducing AI technology, and the concept of *design automation* in general, will deeply change the style of our times and may help to establish a new *Architecture* of the future.

I think that the concepts presented in this research constitute an appropriate environment for applications in architectural technology. I hope that this modest effort will serve as starting point for relevant future research in this promising area, and that my efforts will have contributed to some small degree to the implementation of the many pormising initiatives presented in this thesis.

"I am not yet lost in lexicography, as to forget that words are the sons of earth, and that things are the daughters of heaven. Language is the only instrument of science, and words are but the signs of ideas: I wish, however, that the instrument might be permanent, like the things which they denote".

S. Johnson

GLOSSARY

Note that italicized and bold-face words/expressions of the main text, are emphasized upon in this glossary.

- Abstraction** A conceptual tools of understanding. The more we are willing to abstract from the detail of a set of phenomena, the easier it becomes to simulate the phenomena, for we have to know only those parts of it that are crucial to the abstraction. An ill-defined problem can be abstracted, i.e. simplified as much as possible, in order to make it easier to solve.
- Agent** Any part or process of the cognitive mind that, by itself, is simple to understand -even though the the interactions among groups of such agents may produce phenomena that are much harder to understand.
- Algorithm** A prescribed set of roles, or instructions for a solution of a problem in a finite number of steps. An effective algorithm is the one that is computable.
- Architecture (of a system)** It is the specification of a (digital) computer system at a somewhat general level. It includes descriptions from the programmer's (user's) point of view of instruction, and user interface (I/O operation and control, etc.).
- Back-bone module** A common expression in the computer-science community, signifying underlying nodes of a multi-level distributed network providing communication services for the rest of the modules.
- Blackboard Mechanism** Often, we face a problem where the solution requires more than one point of view. For example: to design a building we need an architect, a structural engineer, etc. For every design problem in that building, every panelist will offer his own opinion and constraints. The compliance to all the imposed constraints is the synthetical process of design. Therefore, when the problem is broken down into its basic elements as loosely divided parts of a metaphorical blackboard, every panelist contributes to the blackboard, and the problem becomes less indeterminate, if not fully determinate [Nii 1986]. Sometimes, contradictions arise among panelists, therefore a blackboard moderator is essential in order to judge these contradictions according to certain priorities. The moderator is responsible for preparing an agenda of the topics to be discussed on the blackboard, [Winston 1984]. i.e. the blackboard mechanism is used to record the initial hypothesis as well as intermediate results.
- Block-world** A scenario adapted from Winston's doctoral thesis, "Learning Structural Descriptions by Examples" about a study of the world of children's building-blocks. The idea may at first seem childishly simple, but it has been one of the most productive ideas about AI, child psychology, and modern robotic engineering.

Frame	A representation based on a set of computer terminals to which other programming structures can be attached. Normally, each terminal is connected to a "default assumption", which is easily displaced by more specific information. Other ideas about frames are found in [Minsky 1975].
Generate-and-test	Solving a problem by trial and error, i.e. by proposing solutions recklessly, then rejecting those that do not work.
Heuristics	There is no agreement on what the definition of heuristics is. However, it is agreed that heuristics is an emerging concept bearing on logic. It is an important and effective tool which allows designers to reduce the number of possibilities, and make decisions. There are two major categories of heuristics; <i>procedural</i> and <i>substantive</i> . In a view of design as the movement between an existing state and a desired state. Procedural and substantive heuristics correspond respectively to the states and movement between them.
Holistic	Derived from the terminology of Gestalt theory. If the unexpected emergence from a complex system of a phenomenon that seemingly is not inherent in that system's separate parts. Such "emergent" phenomena show that a <i>whole</i> is more than the <i>some</i> of its parts.
Inference	A formal method of reasoning that underlies logical deduction. Rules of inference provide the means whereby a logician may use obvious results to derive new facts (see also predicate calculus).
Intelligence	A term frequently used to express the assumption that some single entity is responsible for the quality of a person's ability to reason. Minsky prefers to use the word as representing not any particular power or phenomenon, but simply all the mental skills that at any particular moment, we admire but do not yet understand.
Iteration	The repetition of a numerical or non-numerical process where the results from one or more stages are used to form the input for the next. It is a typical feature of LISP programming.
K-Line	This term is taken from Minsky [1980]. The theory that certain kinds of memories are based on sets of agents that reactivate one's previous partial mental states.
LISP	Acronym for "list processing". A high level programming language designed for the manipulation of non-numeric data.
Logical Structure	The popular but theory that much of human reasoning proceeds in accord with clear-cut rules that lead to foolproof conclusions. As used here, logical reasoning is applied in special forms of adult thought, which are used mainly to summarize what has already been discovered. Most of our ordinary mental work, common sense reasoning, is based more on thinking by analogy, applying to our present circumstances our representations of seemingly similar previous experience.

Memory	An omnibus term for a great many structures and processes that have ill-defined boundaries in both every day and technical psychology; these include what is called "re-remembering, "re-minding".
Message Passing	Message passing is a very powerful mechanism for simulating the hierarchy of constraints of a real-world problem, as it is always found that an organizational hierarchy devised from problem solving. This mechanism is based upon having actors, each of whom simulates a real-life actor from the organizational point of view, and decisions are made by passing messages among different actors [Hewitt 1977].
Model	Any structure that a person can use to simulate or anticipate the behaviour of something else.
Non-convex	This is a non-continuous function, hard to optimize.
Object-Oriented-Programming	It is a way for code-suppliers to encapsulate functionality in order to deliver it to the users (see chapter 3). It places increased emphasis on the relationship between the user and the designer of the code that distinguishes <i>object-oriented</i> from <i>conventional</i> programming. Object-oriented-programming can be added to almost any conventional programming language by grafting a small number of new syntactic features alongside the existing capabilities of the language. Its primary new features can be summarized in two key words; encapsulation and inheritance [Cox 1986]. The real value of an object-oriented language is in the libraries that become feasible once the language makes reusability possible. Through the explicit simulation of behaviour of the various objects, the consequences of an action become more traceable, so we may change some data and watch the consequences propagating automatically all over the system.
Parsing	The process of whether a string of input-symbols is a sentence of a given language. If so determining the syntactic structure of the string.
Predicate Calculus	A system of inference that is a generalization of the quantified or extended propositional calculus, obtained by introducing generalized functions and predicates.
Procedural Embeddings	This is a class of knowledge representation (imperative rules), where the user can state explicitly the desired results to be produced, but does not explicitly define what properties the result is expected to exhibit (e.g. rule-based expert systems).
Production Rules	A coherent set of rules, each of which identifies a condition of applicability and an action to be taken where the rule is applicable (e.g. expert systems).
Pronome	A type of agent associated with a particular role or aspect of representation -corresponding to an actor, trajectory, or cause of some action. Pronome agents frequently control the attachments of "terminals" of frames. To do this, a pronome must possess some temporary memory.
Quadratic	A function of a second degree, where its surface is a curve whose

problem	cartesian coordinates are algebraic of the second degree (e.g. paraboloids, ellipsoids, etc.).
Real-time-world	The environment in which a system's output is significant, i.e. since the input corresponds to some movement in the physical world. An output has to relate to the same movement. The lag from input time to output time must be sufficiently small for acceptable timeliness. This is typically important in project management.
Recursion	This term is taken from Minsky [1975]; the idea is that no society, however large, can overcome every limitation to solve its problems unless it has some way to re-use the same agents, over and over again, for different purposes.
Representation	A structure that can be used for something else, for a certain purpose, as one can use a map as a substitute for an actual city.
Search	The process of locating information in a table or a file by reference to a special field of each record.
Semantic	(Nets): A means of representing knowledge as a labeled directed graph. Each vertex of the graph represents a concept, and each label represents a relationship between concepts. (Primitives): Networks that are not capable of being broken down into simpler form, such as integers, Boolean expressions, etc.
Syntax	The rules defining the legal sequences of elements in a programming language. For example, in LISP, the syntax rules define the form of the various constructs in the language and give clues about the means of these constructs (see also parsing).
Test-bed	A system whose primary purpose is to provide a framework within which other systems can be tested. Test-beds are usually tailored to a specific programming language.
Trajectory	Literally, the path or route of an action or activity. However, this word is used not only for a path in space but, by analogy, can also be used for other realms of thought.
Trans-Frame	A particular type of frame that is based on the concept of the "trajectory" between two situations, one for "before" and the other for "after".

BIBLIOGRAPHY

Abelson, H. & G. Sussman "*Structure and Interpretation of Computer Programs*". Cambridge, MA ; MIT Press [1985].

Ackoff, R. L. "*Redesigning the Future*" New York; Wiley-Interscience, [1974].

Al Banna, S. et al. "*SOMI - An Interactive Graphics Space Allocation System*" Technical Report PB-214-007 Columbia University [1972].

Alexander, C. "*Notes on the Synthesis of Form*". Cambridge, MA. Harvard University Press [1964].

Anderson, J. "*Organizing for Large project Management - The client's needs*" Project Management Institute. Los Angeles, CA. [1978].

Banham, R. "*Theory and Design in the First Machine Age*" MIT Press [1980].

Bar, Avron, Cohen, & Feigenbaum "*The Handbook of Artificial Intelligence- 3 volumes*" Los Altos, CA. [1982].

Barber, Gerald R. "*Office Semantics*" Ph.D. Thesis, Dpartment of EEC, MIT [1982].

Benevolo, L. "*History of Modern Architecture: " 2 volumes, volume/1 "The Tradition of Modern Architecture"* MIT Press, Cambridge [1985].

Carbonell, J. ; R. Michalsk ; T. Mitchell "*An Overview of Machine Learning*" : In Michalski (ed.) Machine Learning. Palo Alto CA; Tioga Pub. [1984].

Casalina, V. "*Morphologies of Floor Plans*" Technical Report, UCLA [1975].

Chalmers, J. "*The Development of CEDAR*" In proceedings of the "International Conference on Computers in Architecture", York (British Computer Society, London); pp 126 - 140 [1972].

Charniack, Eugene, McDermott "*Introduction to Artificial Intelligence*" MA: Addison Wesley Co. [1985].

Charniak, Eugene, Reisbeck & McDermott "*Artificial Intelligence Programming*". Hillsdale NJ: Lawrence Erlbaum asscosiation [1980].

Chomski, Noam. "*A Language of Mind*". London UK: Harcourt, Bruce, Jovanovich [1968].

Cooley, M. "*Computer Aided Design: its nature and implications*" Amalgamated Union of Engineering Workers, Technical and Supervisory Section; Richmond, Surrey [1972].

Cox, Brad J. "*Object Oriented Programming -an evolutionary approach*" Addison-Wesley [1986].

Cross, N. *"The Automated Architect"* Pion Pub. London [1977].

Cross, N. *"Simulation of Computer Aided Design"* unpublished MSc. dissertation, University of Manchester Institute of Science and Technology; Manchester, England [1967].

Cross et al. *"Computer Aided Information Retrieval: a pilot study of some possibilities"* Building 11 September pp 127 - 131 [1980].

Cross, T.B. & Gouin, M.D. *"Intelligent Buildings - Strategies for Technology and Architecture"* Dow Jones-Irwin; Homewood IL. [1986].

Delgado, J. *"Decision Support System for Higher Educational Facilities"* Masters Thesis, MIT [1987].

Dietz, A. *"Building Technology- Potentials and Problems"* In Dietz & Cutler (ed.) *"Industrialized Building Systems in Housing"* MIT Press [1971].

Dluhosch, E. & Vien, T. *"Masonry Compute Software"* -Integrated teaching tool, combining a video-disc module, a designing module, and a glossary of masonry terms; for the International Masonary Institute- Washington D.C. [1987].

Doyle, Richard. *"Hypothesizing and Refining Causal Models"* Technical (Report AIM 811) Cambridge MIT [1983].

Eastman, C. M. *"Spatial Synthesis in Building Design"* New York: Willey & Sons (ed.) [1975].

Ellin, A. R. *"An Algorithm of Design for the Optimization of Building Layouts with Rectangular Rooms"* Vancouver B.C. Canada [1979].

El-quessny, Yasser & Dluhosch, Eric *"Classification"* Technical Report [1986].

El-Quessny, Yasser *"Introducing Artificial Intelligence Sciences to the Domain of Architectural Design & Building Technology"* paper presented at AREC-DAO 87; Barcelona [1987].

el-Shafei, Nayel S. *"Utilizing Qualitative Discovery in Quantitative Reasoning"* Master's Thesis MIT [1986].

el-Shafei, N. *"Artificial Intelligence in Engineering"* draft; MA [1986].

elShafei, N. *"HOTEP system"* A system under development at the Civil Engineering Dept. MIT; Cambridge [1986].

Esher, L. & Liewllyn-Davis, L. *"The Architect in 1998"* RIBA Jornal pp. 448 - 455 October [1978].

Evans, B. *"Intelligent Computer-Aided Design Systems"* unpublished MSc. dissertation, University of Manchester Institute of Science and technology; Manchester, England [1969].

Feigenbaum, E.A., Bauchanan, B.G., Lederberg, J. *"On Generality and Problem Solving -a case study using the DENDRAL program"* In Meltzer, Be., MITchie, D. (ed.) *Machine Intelligence*; Edinburgh University press [1971].

Flemming, U. *"A Generative Expert System for the Design of Buildings Layouts"* Technical Report, Carnegie Mellon University [1986].

Flemming, U. *"Wall Representations of Rectangular Dissections, and Their Use in Automated Space Allocation"* Environment and Planning B 5:215-232 [1978].

Garson, J. & Staedman, P. *"The Automatic Generation of Minimum House Standard Plans"* paper delivered on the Second Annual Conference of the Environmental Design Research Assoc. [1970].

Gero, J. & Radford, A. *"On Optimization in Computer Aided Architectural Design"* Computer Report CR34, Department of Architecture Science, University of Sydney [1980].

Goldberg, Adele *"Smalltalk-80: The interactive Programming Environment"* Reading MA; Addison-Wesley [1984].

Gross, Mark *"Design as Exploring Constraints"* Ph.D. Thesis: Dept. of Architecture MIT [1986].

Haas, Carl, Helen Shen, & Ralph Haas *"ADA System - Automated Data Acquisition and Evaluation System"* Technical Report Project 21156 University of Waterloo [1985].

Hewitt, Carl *"Viewing Control Structures as Patterns of Passing Messages"* Artificial Intelligence, 8(3), 1-10 [1977].

Hillis, William D. *"The Connection Machine"* Ph.D. Thesis MIT [1986].

Howard, H.C. & Sriram, D. *"Architecture of an Integrated Knowledge Based Environment for Structural Engineering Applications"* Technical Report 12-47-86 Department of Civil Engineering MIT [1986].

Jencks, C. *"Architecture 2000"*; Praeger Press [1971].

Jurgensen, Peter *"The Conceptual Design of Architectural Form; a performance spec for a computer system"*, Master's Thesis MIT [1986].

Kalligas, S. *"A Computing Environment for the Application of System Building in Architecture"* Master's thesis MIT [1986].

Kroll, L. *"An Architecture of Complexity"* MIT Press [1987].

Langley, P. & Simon, H. *"The Search for Regularity: Four Aspects of Scientific Discovery -BACON system"* Technical report CMU-RI-TR-84-20 Carnegie Mellon University [1984].

Lenat, Douglas *"The role of Heuristics in Learning by Discovery"*. In Michalski, Carbonell & Mitchell (ed.) Machine Learning, Palo Alto CA; Tioga Pub. Co. [1983]

Lenat, D. & Davis, R. *"Knowledge-based Systems in Artificial Intelligence"* A presentation for the Ph.D. dissertation of the authors that describe in details the AM system for Learning by Discovery in the domain of mathematics, and Tereisias the reasoning system of MYCIN expert system. New York: McGraw Hill [1982].

- Lozano-Perez, Tomas *"Automatic Planning of Manipulator Transfer Movements"* IEEE Transactions on Systems, Man, Cybernetics, SMC-11(6), 681-698 [1981].
- Marr, David *"Vision-A Computational Investigation into Human Representation and processing of Visual Information"*; San Francisco, CA: W.H. Freeman & Co. [1982].
- Michalski, Ryszard *"A Theory and Methodology for Inductive Learning"* In Michalski, R. et al. (ed.) Machine Learning, Palo-Alto CA; Tioga Pub. [1984].
- Minsky, Marvin *"The Society of the Mind"* Cambridge MA: Simon & Schuster [1987].
- Minsky, Marvin *"NATO Conference Series: Systems Science"* Vol.16, In Selfridge, Oliver, Edwina Rissland, & Michael Arbib (ed.) "Adaptive Control of Ill-defined systems" New York: Plenum Press [1984].
- Minsky, Marvin *"A Framework for Representing Knowledge"* In Winston's (ed.) "The Psychology of Computer Vision" New York: McGraw Hill [1975].
- Minsky, Marvin *"Semantic Information Processing"* MIT press; MA. [1969].
- Mitchell, W. *"Computer-Aided Architectural Design"* : VNR [1979].
- Mitchell, w. et al. *"Synthesis and Optimization of Small Rectangular Floor-plans"* Environment and Planning B 3:37-70 [1976].
- Mitchell, Kellar & Kedar-Cabelli *"Explanation-based Generalization, A Unifying View "* Technical Report ML-TR-2, Lab of C.S. Research, The State University of N.J. , Rutgers [1985].
- Negroponte, N. *"Soft Architecture Machine"* Cambridge, MA. [1975].
- Negroponte, N. *"The Architecture Machine"* Cambridge, MA. [1973].
- Negroponte, N. & Groisser, L. *"URBAN5 - A Machine that Discusses Design"* In "Emerging Methods in Environmental Design and Planning" G.Moore (ed.) pp. 105-115, Cambridge: MIT Press [1970].
- Newell, A. & Simon, H. *"GPS, a Program that Simulates Human Thought"* In "Computers and Thought" FEigenbaum & Feldman (ed.) McGraw Hill [1963].
- Nii, Penny *"The Blackboard Model Of Problem Solving-part.1."* AI Magazine, VII(2), 38-53; summer [1986].
- Nilsson, N. J. *"Principles of Artificial Intelligence"*: CA, Tioga Press [1980].
- Pena, W. *"Problem Seeking"* Cahners Books International [1977].
- Pevsner, N. *"A History of Building Types"* Princeton University Press [1976].
- Pople, H. *"Heuristic Methods for Imposing Structure on Ill-Structured Problems"* The Structuring of Medical Diagnosis; in P. Szolovits (ed.) 'Artificial Intelligence in Medicine' AAAS [1979].

Radford, R. *"The Harness Hospital Development Programme"* Building International 7 (1) pp. 43 - 56 [1974].

Sackman, H. *"Man-Computer Problem Solving"* Auerbach, Philadelphia [1980].

Schon, D. *"Designing in the Light of the year 2000"* Student Technologist (Autumn) pp. 20 - 24 (extracts reprinted in Cross, N. Elliot, D. Roy, R. -ed.) "Man Made Futures"; Hutshinson, London [1969].

Schnack, R. *"Dynamic Memory"* Cambridge University Press; Cambridge, England [1982].

Sharpe, R. & Marksjo, B.S. *"Facility Layout Optimization Using the Metropolis Algorithm"* Tecnical Report CSIRO Australia [1974].

Simon, H. *"The Science of the Artificial"* Cambridge MIT Press [1969].

Simon, H.A. *"Style in Design"* In "Spatial Synthesis In Computer Aided Building Design", C. Eastman (ed.) pp.98-147 New York; Wiley [1975].

Stallman, R. & G. J. Sussman *"Toward Reasoning and Dependency Directed Backtracking in a System for Computer-Aided Circuit Analysis"* MIT, A.I. Technical report [1977].

Sussman, G. J. & G. L. Steele *"Constraints. A Language for Expressing almost Heirarchical description"* (Journal 14), [1980].

Sussman, Gerald J. *"SLICES: At the Boundary between Analysis and Synthesis"* In laTombe, Jean-Claude (ed.) "Artificial Intelligence and Pattern Recognition in Computer Aided Design", Amestterdam, the netherland: North-Holland Publishing Co., [1978].

Sutherland, I. *"SKETCHPAD - A Man-Machine Graphical Communication System"* Technical Report No. 296, Lincoln-lab, MIT Cambridge [1963].

Szolovitz, P. & Pauker, S. *"Categorical and probablistic Reasoning in Medical Diagnosis"* Artificial Intelligence 11, 115-144 [1978].

Thomson, B. & Hughes, J. *"An experimental Investigationof the Performance of a Computer-aided Design Building design System"* Proceedings of IFIP Congress 84; International Federation of Information Processing Societies; Stockholm [1984].

Weinzapfel, G.& Johnson, T. *"The IMAGE System and Its Role in Design"* Proceedings of International Conference on Computers in Architecture; York, British Computer Society- London [1971].

Winston, P. *"Artificial Intelligence"* MA: Addison Wesley Pub. [1984].

Winston, Patrick H. *"Learning and Reasoning by Analogy: The Details"* Technical Report AIM520 MIT [1980].

Winston, Patrick H. & Prendergast (ed.) *"The AI Business"* MIT press [1986].

Yamagata, Y. ; Yoshitsugu, Aoki ; Sanae Ikeda *"A Computer Aided Room Allocation Model, with Concepts of Framableness and Flexible Constraints"* CIB [1986].