# A Short Survey of Models of the Design Process

by

Steven L. Levin

Technical Report #71

October 1975

Department of Information and Computer Science
University of California, Irvine

This research was supported in part by the National Science Foundation under GJ-36414.

## ABSTRACT

A taxonomy for describing models of the design process is presented. The taxonomy is used in comparing and discussing several types of design models. The paper's appendix contains a summary description for each model that is discussed.

CONTENTS

# INTRODUCTION

Design is an activity with which we are all familiar. We know that architects and engineers do design in preparing drawings for clients and manufacturers. We recognize that design is done in writing computer programs, deciding upon the administrative structure for an organization and in rearranging the furniture in a room. Design is much like our ability to understand language. It is a process that constantly surrounds us, which we manage to do and yet do not really understand how it is done.

In the late nineteen fifties, designers, predominantly in architecture and engineering, began to study the design process. Their goals were to attempt to understand what took place during the process called "design" and from the attempts to devise ways of improvement. These improvements would lead to more reliable, lower cost and hopefully "better" designs.

More recently in computer science concern has been raised over the problems of creating software that is economically acceptable and that works! This interest has lead software practitioners to look at how programs might be better designed, what information is used in program design and finally, how to interface man and machine in the design process. This paper examines these three questions by reviewing the research done on models of the design process.

The first section of this paper discusses and contrasts several models of design within the framework of the taxonomy presented in that section. The appendix contains synopses of the models discussed in section one and may be read by readers unfamiliar with a particular design model. Section two contains the conclusions which suggest further lines of research in the study of design behavior.

## DISCUSSION and COMPARISON of DESIGN MODELS

This section compares the better known design models using a simple taxonomy. To better distinquish the models their similarities and contrasts have been condensed into three charts. This section begins with a description of the taxonomy and is followed by the charts. A discussion of the charts and a comparison of models concludes the section.

### Taxonomy for Design Models

Type of model: Is the model prescriptive (P), descriptive (D) or a hybrid (H).

Descriptive: Models that attempt to provide an explanation for actual human design behavior.

Prescriptive: Models that provide directions for doing design.

Hybrid: Models that both prescribe and describe forms of design behavior.

Level: Describes the scope and detail at which the model analyzes the design process (1).

Molar: This is the highest level of detail and it describes the entire length of the design task. The types of information collected include: the general sequence in which major design decisions are made, what those decisions are, and the sequence in which information is received and from whom.

Molecular: Subsets of design decisions are analyzed. The problems are looked upon as the province of a single decision maker. The actions at this level can be characterized as design "problem solving" and are amenable to the analysis of Newell and Simon [21]. Actions at this level assume as primitives both standard methods of analysis and cognitive processes.

Atomic: The lowest level of design analysis is concerned with the primitive abilities required by

- - - - - - - - - -

(1) Eastman [10] originated these categories.

designers to analyze and synthesize physical systems. Includes: structure of human memory about physical and visual world and strategies for accessing information within this structure.

Type of Design Problems: The taxonomy provides for six types of design problems as categorized by Reitman [22]. The problem type is determined by the information given the designer about the initial object, the terminal object and the sequence of operations that can be used in going from the initial object to the terminal object.

Type 1: The initial object and terminal object are well specified and the problem is to discover a sequence of transformation operations.

Type 2: The terminal object is not specified well and there is no statement of the initial object but any sequence of operations is allowable.

Type 3: The initial object is a set of one or more subcomponents to be used in creating a loosely defined state or object.

Type 4: Both the initial object and terminal object may be represented as divided into subcomponents.

Type 5: The initial object is given by reference to some well-specified object and the terminal object by a combination of similarity and difference statements made with respect to the initial object.

Type 6: There is a reasonably well specified terminal state and essentially an empty initial state and processes.

Formulation of the Model: Is the model a program (P) that simulates or predicts design behavior or a non-program (NP).

Designer Information and Processing: How does the model handle design information and actions for:

Problem requirements: Their recognition and interrelation.

Subproblem determination: Identification and isolation of independent subproblems in the design.

Design quantification: Assignment of quantitative values to different parts of the design. Examples: Assigning relative weights of importance to a set of requirements, providing

cost functions for alternative solutions, etc.

Solution typologies: Availability of alternative solution schema for solving a particular design subproblem. Example: Linear search, hashing, binary search are solution typologies for solving an item search problem.

Critical Information and Processes: Does the model depend upon the existence of any type of critical design information or process?

Design Sequencing: Describes the control flow amongst activities constituting the design process: Sequential (S), concurrent (C), and variable sequential (V).

Sequential: Design follows a sequence of well-defined operations.

Concurrent: Design performed by alternating between one or more kinds of design activity. A particular model may impose greater importance to one activity over another.

Variable: A well-defined sequence of activities but branching back to earlier steps in the sequence is allowable.

Termination Criteria: The provision in the model for determining when to halt the design process.

Origin Date: Approximate date when model was originated.

Discipline: Academic area of model's author (typically engineering, architecture, computer science or psychology).

Evidence: A brief notation indicating the type of supporting or non-supporting evidence that exists for that particular model.

Comparative Features Charts

The information on each model has been summarized into three charts which are grouped by type of model (e.g. prescriptive, hybrid and descriptive). Blank areas are left in the charts where the model builder has not provided that information. A fuller description of each model is given in the appendix.

| | Alexander | Alexander | Archer | Jones |
|---|---|---|---|---|
| Type of Model | Prescriptive | Prescriptive | Prescriptive | Prescriptive |
| Level | Molar/Molec | Molar/Molec | Molar/Molec | Molar/Molec |
| Design Prob Type | Type 2,6 | Type 3 | Type 2 | Type 2 |
| Formulation | Non-----program | Non-program | Non-program | Non-program |
| Origin Date | 1963 | 1968 | 1963 | 1963 |
| Discipline | Architecture | Architecture | Architecture | Architecture |
| Dsgn Info/Proc Prob Reqs | identify and cross relate reqs | accounted for in patterns | lists goals and properties | identify all possible interactions |
| Subprblm Deter | obtain thru set decomposition | exist as patterns | | identified as p-specs |
| Dsgn Quan | assess strength of reltns amongst reqs either 1,0, or -1 | | assign values to dsgn variables, set acceptable limits | |
| Sol Typol | | given by patterns | | |
| Info - Processes | listing misfit var and decomposing eqs into sets | avail of patterns and rules for composition | identifying all properties,possible states, acceptable limits for goals | listing and relating requirements to perform specs |
| Dsgn Seqning | Sequen: analysis, synthesis | | Sequen: analysis, synthesis, commun | Sequen: analysis synthesis, eval |
| Term Criteria | tree of diagrams | set of patterns | | plot solutions to find "best" |
| Evidence | some designs done by Alexander | limited use by colleagues | | use of some of the analysis techns |

Chart 1:Prescriptive Models

| | Freeman/Newell | Moran |
|---|---|---|
| Type of Model | Hybrid | Hybrid |
| Level | Molar | Molar |
| Design Prob Type | Type 1 | Type 1 |
| Formulation | Program | Program |
| Origin Date | 1971 | 1970 |
| Discipline | Comp Sci | Arch/Comp Sci |
| Dsgn Info/Proc Prob Reqs | input as set of fnc reqs, explicit for each strux | stored explicitly for each concept |
| Subprblm Deter | explicit as fcn reqs | certain concepts maybe interprted as subproblems |
| Dsgn Quan | allowance for in modl, not in prg | constraints must eval to T or F |
| Sol Typol | explict as struxs | explicit as concepts |
| Info - Processes | avail of fcns and strux base | association memory |
| Dsgn Seqning | Sequen: matching fcns to struxs | Sequen: problem space search, ideally formul then search |
| Term Criteria | reach lowest level of avail struxs | "given as part of problm spec" |
| Evidence | none | none |

Chart 2: Hybrid Models

| | Hillier | Mitroff | Rittel |
|---|---|---|---|
| Type of Model | Descriptive | Descriptive | Descriptive |
| Level | Molar | Molecular | Molar |
| Design Prob Type | Type 2 | Type 1 | Type 2 |
| Formulation | Non-program | Program | Non-program |
| Origin Date | 1972 | 1968 | 1972 |
| Discipline | Architecture | Social Science | Architecture |
| Dsgn Info/Proc Prob Reqs | | explicit inputs to the simulation | |
| Subprblm Deter | | | |
| Dsgn Quan | | | |
| Sol Typol | | | |
| Info - Processes | designers pool of sol types, informal codes | heuristics for alt gen, ranking criteria | dynamic classif of info, perform prediction |
| Dsgn Seqning | Conc: altern gen and reduction | Sequen: alt gen then ranking | Conc: solut conjec and analysis |
| Term Criteria | | | |
| Evidence | | heuristics inferred from prtcls, informal interviews | |

Chart 3: Descriptive Models

Prescriptive Models

The area of design research came into being with the claim that the traditional methods of design were too simple for the growing complexity of the problems facing designers (2). The response to the feelings of inadequacy of traditional design took two directions. The first is investigated here by looking at proposed design methods (or prescriptive design models). We will later look at parallel and more recent work by theorists who were trying to examine what the designer actually did in practice and improve design by formalizing the design process on that basis.

In 1962 a book by Morris Asimow entitled Introduction to Design was published. In it Asimow set forth a number of steps that he conjectured as the principal phases in engineering design. They included:

1. Feasibility study-Phase I
2. Preliminary design-Phase II
3. Detailed design
4. Planning the production process
5. Planning for distribution
6. Planning for consumption
7. Planning for retirement of the product

The detailed design phase was further subdivided:

1. Prepartion for design
2. Overall design of subsystems
3. Overall design of components
4. Detailed design of parts
5. Preparation of assembly drawings
6. Experimental construction

- - - - - - - - - - -

(2) The merits of this claim and a discussion of what constitutes traditional design may be found in Design Methods by J. Christopher Jones.

7. Product test program
8. Analysis and prediction
9. Redesign

He also outlined a process for solving problems at each stage:

1. Analysis
2. Synthesis
3. Evaluation and decision-which extended into
4. Optimization
5. Revision
6. Implementation

Asimow's book contributed to two significant events. First, it aroused considerable interest in describing and operationalizing the actions of design. Secondly, and more important, it was to view design as a systematic process involving a sequence of decisions that moved from one defined step to another. Within each step occurred a basic sequence involving analysis, synthesis and evaluation. This view of the sequence and nature of design set the basic pattern for design models developed in the early and middle sixties.

The models of Alexander [1], Archer [4] and Jones [15] are representative of that period and similar to Asimow's formulation of the design process. The similarities are best seen in the information needs of the model's and their sequencing properties. Because all three models emphasize analysis they concentrate on having the designer accumulate, specify and sometimes quantify design information using some

formal technique. Alexander used sets of misfit variables, Jones advocated interaction matrices and performance specifications, and Archer used extensive operations research methods to analyze the design. The effectiveness of each method, just in the analysis phase, depended on the completeness and availability of information to the designer.

Knowledge that describes the interactions of different solutions, goals and constraints is listed as critical information in these models. Each model explicitly requires and uses such information to direct the synthesis phase of the design. Lack of this knowledge impairs the synthesis process in those models.

The emphasis of these models in having a systematic procedure is carried into the synthesis phase. In Alexander's model the designer uses set theory to find the most strongly connected subproblems which when solved independently offers the "best" solution. The output and terminating criteria is a tree of diagrams which is a hierarchical solution to the isolated subproblems. Jones uses the performance specifications to identify sets of potential solutions which are plotted to find the "best" solution. Archer proposes to link the decision variables identified in the analysis phase of his model and use them in evaluating proposed solutions.

Evidence to support the effectiveness of these models is limited. Partial success may be attributed to some models since they did develop several analysis techniques that are in frequent use today. The main criticism leveled at these "first generation" models is the fallacy of their basic assumption that viewed design as a linear process of analysis, synthesis, and evaluation (3). Later design theorists attacked these "rational" models saying that their intuitive appeal and flavor of the scientific method do not match the reality of commonplace design. In practice, perfect and complete information for analysis is not obtainable and while systems of design that work from such bases are desirable actual design methods must be more realistic.

Two other prescriptive models should also be considered at this point. The first is the later work by Alexander [3] wherein he abandoned the strict and mathematical formulations of Synthesis of Form.

Alexander did not abandon his attempt to resolve conflicts between form and the environment started in Synthesis of Form. He ingeniously moved many of the details of recognizing interactions, isolating subproblems and finding solution forms and hid them in a set of schemas

- - - - - - - - - - -

(3) We shall subsequently argue that this "fallacy" is not a fallacy at all but merely reflects the nature of these prescriptive models.

called "patterns."

Patterns form a combination of the design information involving analysis, decision and solution forms into one unit. Patterns are designed by specialists and then used by designers. Each pattern forms a discrete solution or part of a solution specification and it is the designer's job to construct a complete design solution using the patterns. A fuller description of this model is given in the appendix.

It is interesting to note that in this method the obvious need for specific information still exists. The burden of recognizing the need for specific information has been shifted to the specialist who must design this feature in the pattern. Thus the specialist, in defining the pattern, defines its set of interactions and the solution typologies that will satisfy the pattern's conditions.

There has been some success using the pattern language in design [8] b·t only amongst Alexander's fellow workers. The principal difficulties with this model are the definition of an appropriate set of patterns and the unresolved problem of then connecting them together. Nevertheless, pattern languages remain an interesting idea on how to do design.

Structured programming is a method for writing programs of greater reliability and clarity. It is not normally viewed as a model of design but we included it here due to a

similarity of goals and ideas with other prescriptive models of design. We do not offer any definition of structured programming except that it is to approach design using a limited set of control sequences and to do the design hierarchically.

To succeed structured programming depends like other design models on the designer's ability to recognize relational information in the design involving problem requirements, and design interactions. The designer must then synthesize a design to minimize control and information flow between various defined modules of the design.

Most definitions of structured programming stop at this point. The surface similarity between structured programming and other design models is that all require the same basic high level information. Unlike some of the other models, structured programming does not say how to obtain or use that information. Structured programming is also the least detailed with respect to the level of analysis compared to other models.

## Descriptive Models

The published literature of research in design theory does not distinquish between the purpose of prescriptive and descriptive design models. Descriptive models attempt to characterize the actions of human designers. The prescriptive models were developed to teach people how to

design more effectively. The prescriptive models attempt to describe systematic and consequently algorithmic-like processes for doing design. An algorithm is a set of discrete steps done in some well defined order with defined inputs and outputs along with terminating conditions. Thus, the basic nature of such descriptions, which were the goals of that earlier research, became the basis for later criticism.

In one review of models of the design process [6] it is stated "the criticisms of the early models of the design process can be summarized in the following way: (a) design is not a strictly sequential process, and (b) design problems are 'wicked'." Given the purpose of the early "first generation" models then such criticisms are unfair. The models do not purport to detail how people do design but prescribe a method which if followed would result in better design. The nature of the descriptions are sequential.

The second complaint recognizes the lack of success in formulating processes or programs that can solve "ill structured" problems or "wicked" problems. These are problems where there is not perfect information, defined operations and definite terminating criteria.

The remainder of the this section discusses the descriptive models of Hillier [13], Rittel [23,24] and Mitroff [18]. Generally, the prescriptive models contain less detail on what information designers use, how the

design process proceeds and how decisions are made. An exception is Mitroff's simulation of engineering design. Hillier's and Rittel's models are paper proposals of the design process and Mitroff's is a computer simulation.

Hillier's and Rittel's models contain similar views on the basic pattern of the design process. Common to both is the notion of generating and evaluating alternative solutions as the basic design sequence. Hillier sees designers as alternating between the activity of conjecturing solutions and analyzing their effectiveness. Rittel describes the design process as alternating between the generation of alternatives and their evaluation. Both models conclude that the final form of the design solution is strongly influenced by pre-structuring (Hillier) and early problem formulation (Rittel). Neither theorist offers any supporting data.

Hillier's and Rittel's models discuss at a high level the use of knowledge by the designer. Different types of information are identified and labelled in each model but nothing is said about how such information is acquired and what determines when that information is relevant to the decision at hand.

What constitutes critical information differs in Rittel's and Hillier's models. Hillier emphasizes the importance of having many different solution typologies and sets of "informal codes" for selection. Rittel emphasizes

the designer's ability to use and dynamically reclassify causal information during the design. Neither theorist provides empirical evidence to support their conjectures.

Mitroff's model does not describe the entire engineering process. It simulates the behavior of a specific engineer in generating and ranking design alternatives for pressure vessels. The model is a computer program consisting of heuristic subroutines that embody the decision procedures of the engineer. Input to the model are values for selected technical variables and output is the set of ranked design alternatives. The heuristic procedures of the model were inferred from various types of empirical data (i.e. protocols, constrained problem-solving, interviews).

Hillier's and Rittel's models are almost so general that they are useless. They speculate on sequences for the design process but have not detail or empirical corroboration as in Mitroff's model. Mitroff's model is explicit in the information needs of the engineer. The model details what parameters are important and how weights are given them (or the problems involved in assigning weights).

Mitroff's model studies a segment of the design process and provides empirical information on what data the engineer uses and how. This is in contrast to the introspective and empirically unsupported models of Hillier and Rittel.

Hybrid Models

Hybrid models are a class between models prescribing a formula for design and models describing human design behavior. Moran [19] and Freeman [11] suggest their models resemble human behavior but neither claims to be a strict simulation. The models are formulated as programs that produce designs but neither are automatic design systems.

The general goal of both models is to better explain what goes on during design but the specific goals of each leads to different emphasis. Moran is trying to determine what amounts of knowledge, what form and what sophistication of strategy are necessary to achieve different levels of design. Freeman looks at the phenomena of functional reasoning in design to see what role it plays and its applicability to automated design. Moran's interests are structural and concerned with representing the designer's knowledge. Freeman concentrates on the behavioral aspects of his model resulting from the functional reasoning paradigm.

Outwardly the two models differ substantially in how knowledge is represented and utilized. Freeman's model has sets of structures and functions. Structures supply certain functions and functions require certain structures. The design process is to recognize functional dependencies and use them to build larger constructed structures. The computer formulated model is restricted to sets of functions

and structures (4). It serves as information retrieval system and history keeper. When executing, the program supplies needed functions by searching through the set of available structures (5).

Moran represents conceptual classes, constraints, and assorted facts in an "association memory." This is a graph explicitly linking all knowledge the designer posseses. Concepts may be concrete objects such as door, chair, room, or may be abstract entitities such as proportion, axis, or module. Concepts have a name, a list of attributes and a list of component parts with their interactions.

By using association links amongst concepts as production rules the memory may be used to generate designs. A problem in the form "given an X" may be reformulated using the association links of X as "design the subcomponents of X." The association links for the subcomponents of X direct the next level of problems to be solved.

Superficially Moran's and Freeman's model differ in how design is carried out and the form and structure of design knowledge. The differences begin to disappear as the association links are interpreted as functions linking the

- - - - - - - - - - -

(4) The paper describing the model has provisions for constraint and resource laws but they are not implemented in the computer program.

(5) It is implausible that humans would employ a exhaustive search of their memory for such knowledge.

designer's concepts (structures). The explicit linkage of concepts is replaced by search in Freeman's model. The design strategy component in both models governs the process and not the model's knowledge representation. Knowledge bases are static as is the assertional representation used for design information.

The ability to switch amongst several representations is a major point in Moran's model. An example would be translating part of the design problem into a linear programming language convention and employing a simplex algorithm. The critical aspects of change of representation are the ranges of representation obtainable and the effort needed to make the transformation. Moran's representation of a floor plan and the transformation of that representation to simplex form is a limited success in solving the representation problem.

## CONCLUSIONS

The lack of success of prescriptive models (design methods) in the fields of architecture and engineering should be heeded by software engineers. The warning is against techniques that become overly formalistic and so self-important that the main task, the design, is lost from sight.

The initial prescriptive models which based the success of solution synthesis upon initial problem analysis failed. Theorists now recognize that the formulation of the design problem (analysis) is a major activity that is continuous throughout the design. Design systems that interact with humans must adapt to the restriction that design problems and decision criteria cannot be fully defined at the onset of the design because the designer does not have or cannot communicate that information. Mitroff emphasized this point as a problem in constructing his simulation.

Alexander's model using pattern languages should be further explored in the context of program writing. One idea might be to create an automatic programming system whose principal database consists of software patterns. Software patterns would be analogous to Alexander's patterns for building design. The problem remains of devising a strategy component for assembling the patterns.

The descriptive models in this report, except for Mitroff's, were specious and unsupported. The discussions of what information designers used and the general structure of the design process are of little practical use. Development of knowledge-based design systems such as that of Green and Barstow [12] require much more information than these models provide.

Mitroff's model is a better indication for further research. That model provides detail on the pool of knowledge and strategies employed by the designer. Furthermore there is actual data to support the model.

Mitroff's model starts to deal with how the designer makes decisions that must always be comprimises. This is a critical feature in design. This problem and several others (how is planning done, how does the designer select relevant information to use in decisions and planning, etc.) remain unanswered by current descriptive models.

## ACKNOWLEDGEMENTS

I gratefully acknowledge the assistance and suggestions of Rob Kling and Peter Freeman in preparing this paper.

# APPENDIX

## DESIGN MODEL SUMMARIES

### Christopher Alexander

Alexander's is probably the best known of the early models of the design process. This model was eventually abandoned by Alexander for reasons described in [2] and is fully described in his book Notes on the Synthesis of Form.

Alexander claimed that there was a very important underlying structural correspondence between the pattern of a problem and the process of designing a physical form which answers the problem. He felt the form of the problem/path which lead to the solution was just as important as the solution itself.

Under Alexander's definition the design process had two phases: analysis and synthesis. Every problem could be decomposed in a unique way. In the analysis phase the designer decomposed the problem into problem elements and problem subsets and established a hierarchy amongst them. This process started with the analysis of problem requirements.

The synthesis phase derived a form from the program. Here the starting point was a "diagram" and the result a tree of diagrams which represented a realization of the

program. This phase was carried out by matching program requirements with corresponding diagrams. The solution to the problem was a synthesized diagram which contained all diagrams in the tree.

Alexander's model might be best conveyed using an example of how it would be employed in the design of a family residence. We start with the general requirement for the residence that it provide shelter for the family. This requirement is then broken down to providing privacy, offering comfort, matching lifestyles, etc. The designer in turn then breaks down these requirements into other subrequirements. The designer then organizes the complete list of requirements, enumerates elements in the list and then cross-relates all those elements.

The elements can then be formed into sets with a hierarchy established amongst them. The designer then knows which requirements should be satisfied before other requirements and what their interactions will be. Each requirement is an element in Alexander's "program" and they all form a hierarchical structure--thus before we can arrange seats for a view we must have a window.

From the program the designer proceeds to find pictorial diagrams that satisfy each element and then each set. The diagrams are organized into a tree which dictates the actual form of the house.

Christopher Alexander (Pattern Languages Model)

Pattern languages are the second major prescriptive model that Alexander has proposed and depart radically from the complications and formality he used in Synthesis of Form. Alexander has found that he can go directly to the earlier diagrams without doing the previous explicit decomposition. The diagrams have become "patterns" and it is now they that determine the form of the design.

The key element in the new model is the availability of knowledge with which to make decisions (6). Design knowledge takes the form of solution typologies, causal information, and relative importance of different orderings of design decisions. Alexander embeds these types of information into "patterns."

Patterns are designed by specialists and then used by "object designers." The specialist knows what is important and includes that information in the pattern.

Part of Alexander's philosophy is that every human is a designer and operates with some set of patterns. People make decisions with little and sometimes incorrectly formed

- - - - - - - - - - -

(6) The availability of knowledge is undoubtedly critical in every design process model. How a prescriptive model directs the designer to acquire that information or how the model fills the designer's need for the knowledge is important.

patterns. Patterns designed by specialists both correct old patterns people have and provide new patterns and additional information.

A designer engaged in designing a house may not be aware that the large window that is to provide a view will also present heating problems for the room. The pattern(s) dealing with windows (or large windows) informs the designer of the side-effect and offers one or more solutions to the problem. The designer may act on the pattern's advice or may choose to ignore it due to other information. The designer may know the house is to be air conditioned.

Patterns bring together many diverse pieces of information that assist the designer. Use of patterns does not depend on a separate analysis step in the design process. Alot of the analysis directed at identifying side-effects and solutions is directly coded into the patterns.

L. Bruce Archer

Archer's goal is broader than that of any other design theorist for he attempts to create the foundations for a science of design in addition to describing a design process. His model is complex and borrows techniques and terminology from operations research, management science, statistics and systems engineering.

He starts by assigning specific interpretations to terms such as values, goals, problems, objectives and properties. Particular properties of the environment give rise to certain desires and the attainment of such desires are goals. When it is not clear what action to take in attaining a goal there is a problem. Problem-solving is then an activity used to remove problems. The aims of a designer's problem-solving are objectives.

The process starts with the designer plotting a range of correlation charts of the designer's objectives against the properties of the environment. These charts, in addition to plots of other mathematical relationships (exponential, parabolic, etc.), are supposed to determine how to fulfill goals with respect to certain properties.

Archer continues by then having the designer plot the acceptability limits for the goals. For some goals such as material strength this is reasonable but for other goals such as comfort the designer is left to devise an appropriate scale.

The final outline of Archer's model has thirty items with several feed-back loops. These steps when grouped correspond to three phases the first two of which are like those of Alexander's first model. The first phase is analytical and includes:

1. Agreeing on goals

2. Rating goals

3. Identifying the properties required to be exhibited in the end result

4. Determining the relationships between the varying states of the property and the varying degrees of fulfillment of respective goals

5. Setting acceptability limits for goals

6. Identifying decision variables available to the designer and the scope of resources

Synthesis or the creative phase was:

7. Linking decision variables to properties and properties of goals

8. Ensuring independence of properties is in the range of acceptability

9. Proposing solutions

10. Evaluating solutions

11. Selecting optimal solution

and finally the third phase:

12. Communicating the design description

Peter Freeman and Allen Newell (FRD Model)

Freeman and Newell advance a model of the design process based on the tendency of humans to design in terms of functions. They call this mode of design, <u>functional reasoning</u>.

The basic model defines a task environment in which the designer possesses a set of structures and functions. The different structures provide functions (e.g. a core memory

provides memory) and different structures may require sets of functions (e.g. a physical object requires space).

The structures can be combined by recognizing possible functional connection between structures where one requires functions provided by another. These combined structures form aggregates known as constructed structures.

Designing is the process of finding a feasible structure that can be composed from the designer's set of available structures. Stated another way,

> Given a set of structures and their functional specification,
>
> Construct a structure with the desired functional specifications.

The designer proceeds by interating between matching functions and structures until a stopping point is reached. Freeman points out that the model is independent of the design strategy and could be used in top-down, bottom-up, most critical function first, etc. The output is then a plan for realizing the actual object.

Constraints are expressed as laws governing the supply and capacity of structures. Some quantitative tradeoffs may be viewed as functions on the available functions and structures.

The model was formulated by Freeman as a computer program that could design some simple programs given a statement of required functions. The program selects a function to fill and searches its library to find a set of structures. An external agent (not the program) selects a structure and the program iterates by then filling that structure's functional demands.

Freeman and Newell view their model as a planning device that might be used in initial problem structuring in automatic design systems. There is no evidence of further work on this model or on it's conjectures of functional reasoning as a mode of human design.

Bill Hillier, et al

Hillier presents a descriptive design model which is representative of the shift away from the analysis-synthesis models.

In Hillier's model problems given to designers almost immediately are "pre-structured" by the problem constraints and the designer's own "cognitive map." Pre-structuring corresponds to the designer's earliest plans or alternatively, conceptions of how a solution will be achieved. From this activity (i.e. the designer's preconceptions) and not analysis of user requirements the design is derived. The authors argue that either of the two strategies that they perceive open to designers would

require pre-structuring. These strategies are either to pursue some definite plan (origin unknown) or to interrogate the designer's instrumental set by understanding how it relates to general objects which can be created (7).

The model is augmented by a description of the elements perceived to be in the designer's field such as knowledge of instrumental sets, knowledge of solution types, informal codes and information. Examples of each are given in the paper.

Designing consists of a designer given a problem, prestructuring it and from that conjecturing possible solutions or at least approximations of solutions. This allows the designer to structure an understanding of the problem.

The designer proceeds to collect data and in doing so is able to sharpen the conjectures. Conjectures arise from knowledge of instrumental sets, different solution typologies, informal codes, etc. Thus, conjecture and problem specification proceed side by side rather than in sequence.

As conjectures stand up to the test of increasingly specific problem data a solution in principle is agreed to exist.

- - - - - - - - - -

(7) Instrumental set is the designer's collection of design operations used in transforming one form into another.

J. Christopher Jones

At the 1962 conference on design methods Jones presented a paper in which he stated the aims of design methods and thus the purpose of his model. In it he said:

> 'The method is primarily a means of resolving a conflict that exists between logical analysis and creative thought. The difficulty is that the imagination does not work well unless it is free to alternate between all aspects of the problem, in any order, and at any time, whereas logical analysis breaks down if there is the least departure from a systematic step-by-step sequence. It follows that any design method must permit both kinds of thought to proceed together if any progress is to be made. Existing methods depend largely on keeping logic and imagination, problem and solution, apart only by an effort of will, and their failures can largely be ascribed to the difficulty of keeping both these processes going separately in the mind of on person. So systematic design is primarily a means of keeping logic and imagination separate by external rather than internal means.'

Jones attempts to achieve this aim by having the designer record <u>every</u> piece of design information. The recording of information develops in three stages (8):

1. Analysis in which all the design requirements are listed and reduced to a set of logically related performance specifications.
2. Synthesis in which solutions are found for individual performance specifications and then built up to form complete designs.
3. Evaluation in which alternative designs are tested against performance specifications.

- - - - - - - - - - -

(8) The reader should note the strong similarity in the stages and the design activity taking place.

At each stage Jones describes a specific technique to apply. Analysis is begun by listing the random factors of the design and plotting the interactions. The same information is then plotted using interaction nets and topological diagrams. From this the designer can write the performance specifications or p-specs in which the requirements are expressed in terms of performance and not shape, material, design, etc. As an example "mount the control panel at 45 degrees" would be a design specification whereas "the control panel should be visible from all operating positions" is a p-spec.

Synthesis is concerned with finding partial solutions for each p-spec versus traditional design which initially attempts to find an overall solution. The partial solutions are assembled in various permutations and their interactions plotted to detect incompatibilities and help predict performance.

The traditional use of experience and judgement in evaluation is replaced by statistical evaluation. This includes the collection and assessment of available experience and judgement, simulation and logical prediction.

Ian I. Mitroff

Mitroff's model is a detailed simulation of the design behavior for a specific engineer. The engineer is characterized as a system whose internal properties are

specified by a set of heuristic design rules. The role of the engineer is to transform a client's needs (the input to the engineer) into a product (the output) by using the design rules. Through the use of informal interviews, tape-recorded subject protocols and constrained problem solving sessions the engineer's design heuristics were inferred and then programmed into a computer.

The program, DESIGN, simulates how the engineer generates his entire design space of feasible design alternatives. The program does not simulate the search of the design space for a specific design alternative as a solution to the design needs of the client. The simulation greatly simplifies many of the engineer's design strategies and idealizes the format in which the form and content of client needs are expressed to the engineer.

DESIGN solves a type of problem in pressure vessel design. It is given a statement of basic space, material property and allowable thickness requirements as technical input variables defining a client's needs and proposes upto 57 design alternatives (9).

The model was built as a set of subroutines each of which embodied some heuristic rule of the engineer. DESIGN begins by invoking the most fundamental of the subroutines, BASIC SHAPES, to generate an initial space of alternatives

- - - - - - - - - - -

(9) The maximum of 57 is a mystery.

from which all other alternatives are derived. Mitroff had previously determined that 50 percent of the engineer's design problems were solved using spherical and cyclindrical vessels so BASIC SHAPES only considered those types of vessel.

Mitroff recognizes that the engineer selects a size and shape of vessel under conflicting requirements. The program simulates the final compromise by starting the evolution of design alternatives with more than one vessel size and shape. DESIGN uses its other heuristic subroutines to propose further alternatives by considering additional modifications.

Lastly the simulation ranks all alternatives relative to one another with respect to the engineer's three most important performances parameters: pT, cost and time of construction (10). Mitroff indicates that coding the heuristic rules for the alternative generation subroutines was comparatively easy because they dealt with straightforward technical inputs like flask size, space requirements, etc. Inferring the rules for deciding between alternatives was very difficult. Neither the engineer nor his clients could communicate values to the performance variables and their relative weights. The problem according to Mitroff was that the variables in one sense do not exist

- - - - - - - - - -

(10) pT is a performance parameter of relative tradeoffs in the vessel's thickness in different directions.

because the clients did not describe their needs in that form. These variables emerged implicitly as the engineer gathered information from the client and as the design developed.

Mitroff concludes that "one of the engineer's biggest design problems is to have his client define the real problem and hence what is really required of him." Mitroff observes the strategies the engineer employs in the getting the problem defined and how this part of the design process influences the final design. He finds that "a statement of design requirements cannot be completely removed from a description of the design process as a whole."

Thomas P. Moran

Moran's work is based on earlier and contemporary work of a similar nature by Charles Eastman [9,10]. Moran was a student of Eastman's. The basic direction of their work was to devise computer programs to assist architects in design, specifically as it turned out, in solving a class of problems involving "space planning."

Moran viewed the designer as an information processor and fit the process and problem of design into a model of information-processing developed by Newell, Shaw and Simon [20]. Design in this model is systematic and has two major phases, problem formulation and search for the problem solution.

The problem formulation provides the specification or criteria for a solution. The formulation is integal to the model since it defines a problem space which is interpreted as a "subset of all conceivable forms including a form for the present state and hopefully, forms for possible solutions."

The second step is the search for a solution which proceeds by "stepping from point to point" in the problem space. The transition from one point (state) to another (state) is accomplished by applying by an operation which transforms that point into another form. The model has a fixed set of operators.

Moran notes that both aspects, formulation and search are both active throughout the entire design process but that formulation dominates the early stages and search the latter stages. Moran recognizes that it is the early stages of the design that are most difficult and crucial and states his model is primarily for the later stages, i.e. search for solution. Thus, it is to quote Moran "oriented toward only well-formulated problems."

The model is formulated as a computer program where the designer's knowledge is represented as a graph of nodes and links. Nodes are "concepts" which may be concrete objects or abstract entities. Constraints are links amongst concepts and may always be evaluated to true or false dependent on the current state of the design. The concepts

and constraints are connected by association links in an association memory. The links may be used as production rules to generate designs.

Horst W. J. Rittel

Rittel views designing as "thinking before acting." Designers attempt to develop a set of alternative courses of action, then to predict their outcomes and their likelihood, to evaluate them and finally to select one as the solution. Rittel views decision-making in design as distinctive from that of regular problem-solving due to the long time interval between the design process and the feedbacks resulting from the execution of the plan, the impossibility of solution through trial and error design and the ill-behaved nature of design problems (11).

Ill-behaved problems in design are contrasted with problems of arithmetic and chess (12). Design problems are not well-defined and every formulation of the problem is made in view of some particular solution principle. If the

- - - - - - - - - -

(11) In [24] Rittel later calls these ill-behaved problems "wicked problems" and lists eleven properties by which they can be recognized.

(12) An interesting view by Simon [25] is that the ill structured problems is vague and not susceptible to formalization. The appearance of being an ill structured problem results from having a large memory of potentially relevant information applicable to the problem process.

solution principle changes during the design process then new aspects become relevant and new kinds of information become pertinent.

Evaluating designs is based on personal values and there can be no determination of whether a solution is correct or false. Solutions can be good, bad or adequate but not correct or false, whereas an answer to an arithmetic problem is either right or wrong. Design problems are ill-behaved because there is no stopping rule for terminating the search for a better solution. Design is not like chess where checkmate is a stopping rule for the problem-solving process.

Rittel maintains that this problem formulation precludes attempts to design that start with an analytical phase that would end with the problem being well described. There cannot be a complete list of objectives and information available for a synthesis phase. Designing requires "permanent feedback with the problem environment; i.e., a perpetual information exchange about the conditions of the situation and about the 'ought to' state which is to be accomplished."

Rittel views design as an argumentative process in which the designer and possibly other parties argue towards a solution. Each party contributes to a better understanding of what the design is to do. Eventually solution principles evolve, are evaluated and decided upon.

This process allows for a better formulation of the problem to be developed simultaneously with a clearer and clearer image of the solution.

The process involves two alternating phases of activity which become invoked every time a problem which cannot be resolved in a routine fashion occurs. The first phase involves generating a set of relevant possibilites that might solve the problem. It results in a set of alternatives in which there is at least one element. Next the solution set is reduced by evaluating the alternatives for feasibility and desireability with a decision made in favor of the most desirable and feasible alternative.

Rittel characterizes the problem-solving cycle as a profliferating and nested network and not a linear sequence. In the early phases of the design the cycles are involved with determining the general form of the solution. Later phases, involving more detailed problems, involve specifying values for various parameters in the design which have been identified in the conceptual phase.

Design knowledge is "instrumental" if it is useful in the design process. In his model, instrumental knowledge relates performance variables, design variables and context variables. Performance variables express desired characteristics of the object under design, design variables describe the possibilities of the designer and context variables are those factors not controlled by the designer

which affect the object to be designed.

The variables are components occuring in items of learned causal information possessed by the designer. As particular design problems arise causal information is called into use. Items in that knowledge are classified at that time as being either design, performance or context variables. Once the search method for a table has been specified its speed is a given. If the search method is undecided then the speed is a design variable to be decided upon by the choice of search method. Knowledge reclassification is a continuous process in the model.

Rittel points out that this model offers a way of improving design through improving the argumentation process. If arguments are improved procedurally, (conducted in an orderly, structured fashion according to some rules), their content may improve resulting in better decisions.

# REFERENCES


1. Alexander, C., Notes on the Synthesis of Form, Harvard University Press, Cambridge, Mass., 1964.

2. Alexander, C., "A City is Not a Tree," Architectural Forum, Vol. 122, No. 1, April 1965, pp. 58-63; Vol. 122, No. 2, May 1965, pp. 58-61.

3. Alexander, C., et al., A Pattern Language which Generates Multi-service Centers, The Center for Environmental Structure, Berkeley, Ca., 1968.

4. Archer, L. B., "An Overview of the Structure of the Design Process," Emerging Methods in Environmental Design, Moore, G. T. (Ed.), M.I.T. Press, 1970.

5. Asimow, M., Introduction to Design, Prentice-Hall, 1962.

6. Spilliers, W., (Ed.), Basic Questions in Design Theory, North-Holland/American Elsevier, 1974.

7. Broadbent, G., Design in Architecture, Wiley, 1973.

8. Duffy, F. and J. Torrey, "A Progress Report on the Pattern Language," Emerging Methods in Environmental Design and Planning, Moore, G. T., (Ed.), M.I.T. Press, 1970.

9. Eastman, C., "Explorations in the Cognitive Processes of Design," Carnegie-Mellon University, Department of Computer Science, ARPA Report DDC No. 671-158, 1968.

10. Eastman, C., "Design Augmentation," Computer Science Research Review, Carnegie-Mellon University, 1972-1973.

11. Freeman, P. and A. Newell, "A Model for Functional Reasoning in Design," Proceedings Second International Joint Computer Conference on Artificial Intelligence, London, September 1-3, 1971.

12. Green, C. and D. Barstow, "A Hypothetical Dialogue Exhibiting a Knowledge Base for a Program-Understanding System," Computer Science report STAN-cs-75-476, AIM-258, Stanford University, January, 1975.

13. Hillier, B., Musgrove, J. and P. O'Sullivan, "Knowledge and Design," Environmental Design: Research and Practice, Mitchell, W. J., (Ed.), proceedings of the EDRA 3/AR 8 Conference, University of California at Los Angeles, January 1972.

14. Hillier, B. and A. Leaman, "How is Design Possible? A Sketch for a Theory," DMG-DRS Journal, Vol. 8, No. 1, January-March 1974, pp. 40-50.

15. Jones, J. C., "A Method of Systematic Design," Conference on Design Methods, Jones, J. C, and D. G. Thornly, (Eds.), Pergamon Press, 1963.

16. Jones, J. C., Design Methods, Wiley, 1970.

17. Mannheim, M., "Problem Solving Processes in Planning and Design," Emerging Methods in Environmental Design and Planning, Moore, G. T. (Ed.), M.I.T. Press, 1970.

18. Mitroff, I. I., "Simulating Engineering Design: A Case Study on the Interface Between the Technology and Social Psychology of Design," IEEE Transactions on Engineering Management, Vol. EM-15, No. 4, December 1968, pp. 178-187.

19. Moran, T. P., "A Model of a Multilingual Designer," Emerging Methods in Environmental Design, Moore, G. T. (Ed.), M.I.T. Press, 1970.

20. Newell, A., Shaw, J., and H. Simon, "The Processes of Creative Thinking," Contemporary Approaches to Creative Thinking, Gruber, H., Terrel, G. and M. Wertheimer, (Eds.), Atherton Press 1962, pp. 63-119.

21. Newell, A. and H. Simon, Human Problem Solving, Prentice-Hall, 1972.

22. Reitman, W. R., "Heuristic decision procedures, open constraints, and the structure of ill-defined problems," Human Judgements and Optimality, M. Shelly II and G. Bryan (Eds.), Wiley, pp. 282-315.

23. Rittel, H., "Some Principles for the Design of an Educational System for Design," Design Methods Group Newsletter, Vol. 4, No. 4 and Vol. 5, No. 1, December 1970 and January 1971.

24. Rittel, W. and M. Webber, "Dilemmas in a General Theory of Planning," DMG-DRS Journal, Vol. 8, No. 1, January-March 1974, pp. 31-38.

25. Simon, H., "The Structure of Ill Structured Problems," Artificial Intelligence 4, pp. 181-201, 1973.