12. Intelligent Systems For Supporting Architectural Design

GianfrancoCarrara(°),YehudaE.Kalay(°°),Gabriele Novembri(°°°)

(°)Department of Building Techniques and Environmental Control,
"La Sapienza" University, Rome
(°°) School of Architecture and Planning,
State University of New York at Buffalo, Buffalo, NY, USA
(°°°) Cartesiana Consortium, Rome

Design can be considered a process leading to the definition of a physical form that achieves a certain predefined set of objectives. The process comprises three distinct operations: (1) Definition of the desired set of performance criteria (design goals); (2) production of alternative design solutions; (3) evaluation of the expected performances of alternative design solutions, and comparing them to predefined criteria. Difficulties arise in performing each one of the three operations, as well as in combining them into a purposeful, unified process. First, it is difficult to define the desired performance criteria prior to and independently of the search for an acceptable solution that achieves them, since many aspects of the desired criteria can only be discovered through the search for an acceptable solution. Furthermore, the search for such a solution may well alter the definition of these criteria, as new criteria and incompatibilities between existing criteria are discovered. Second the generation of design solution is a task demanding creativity, judgement, and experience, all three of which are difficult to define, teach, and otherwise capture in some explicit manner. Third, it is difficult to evaluate the expected performances of alternative design solutions and to compare them to the predefined criteria. Design parameters interact with each other in complex ways, which cause effects an side effects. Predicting the expected performances of even primary effects involves extrapolating non-physical characteristics from the proposed solution's physical organization, a process which relies on a host of assumptions (physical, sociological, psychological, etc.) an hence is seldom a reliable measure. A fourth problem arises from the need to coordinate the three operations in an iterative process that will converge on an acceptable design solution in reasonable time. Computational techniques that were developed in the past to assist designers in performing the above mentioned activities have shown limitations and proved inadequate to a large degree. In this paper we discuss the work in progress aimed at developing an intelligent support system for building and architectural design, which will be able to play a decisive role in the definition, evaluation and putting into effect of the design choices.

The methodological approach

Current design practice calls for the constant presence and the reciprocal close integration of an ever growing number of different disciplinary competences of rapidly evolving scientific, technological and organizational content. Such specialization and integration include significant problems, due both to the need for a constant, widespread professional updating, and to the increasing difficulty of communicating between experts in different fields, even when they are often closely related. Furthermore the increased "intrinsic" difficulties of the field are exacerbated by other difficulties, often no less imposing, derived from the methods of implementing the building process, that is, the procedures governing all the phases of designing, constructing and managing the building asset.

Faced with these situations, all the parties concerned and the operators in the field feel the need for formulating building information in a clear and technically flawless manner provided in a timely fashion. It must meet the information needs of the building process phase in which it is used, and be verifiable, modifiable, and updatable in real time, so it can optimize the solution an respond to the complex ensemble of variables, while enabling the solution to be compared with other solutions.

It has been shown (Carrara et alt. 1985, Kalay 1985) that the development of software systems to support design must be based on a thorough analysis of the methodologies and the logical procedures of the design process itself, aimed at identifying those activities that can or should be computer supported.

Researchers responding to this challenge demonstrated the importance of the techniques of representing knowledge, and incorporating it in CAAD tools (Carrara et alt. 1985, Carrara et alt. 1986, Novembri 1989, Kalay 1985). They have also demonstrated the close copuling between the concepts of prototype refinement, prototype adaptation on one hand and between prototype generation and Routine and Creative Design (Gero et alt. 1988) on the other. Routine and creative design are applied alternatively by the designer. Situations can in fact occur in which, starting with existing solutions, it might become necessary to define new partial solutions, while a new design solution could to be perfected and completed in all its parts to arrive at the final design.

Different methods have been proposed to implement the process of seeking design solutions (Shaviv 1979, Woodbury 1987, Swerdloff et alt. 1987, Flemming 1987, Stiny 1980). Each of these systems has shown interesting potentials but also strong limitations emerging clearly if the results they produce are compared to the real situations in which the designer works, rendering these potentials limited and basically theoretical.

In actual design work, many factors concerning the design process are unknown to the designer and would not seem to be determinable a priori. Rather, the search for a design solution is a learning process, through which facts, possibilities and compromises are discovered, which can influence an modify choices an decisions already made, as well as radically change objectives that have already been set.

Moreover, design solutions cannot be defined separately from the process that determines their viability, since it is the evaluation of the choices that were made that leads to achieving the objectives (Simon 1969). The difficulty of forecasting all the effects and consequences of the choices that were made precludes planning of the process, except at the most abstract level. Rather the designer must weigh the positive or negative effects of all the solutions adopted and to derive operational indications from this evaluation. The choices must be assessed not only also in relation to achieving the objectives, but also in light of the choices that were made earlier.

The difficulties of formalizing and fully understanding the mechanisms of learning, creativity, and the critical capacities necessary to develop an activity such as designing, seem to be (under current conditions) insurmountable. These processes are and will for long remain the exclusive prerogative of humans.

Recognizing these difficulties and at the same time faced with the real necessity of producing tools capable of helping the designer led some time ago to the definition of a different approach aimed at producing software instruments that able to facilitate design work as a whole without attempting in any way to supplant it. This approach is based on the fact that designers are capable of handling extremely complex problems and processes, and have, in the course of the centuries, achieved results that are there for all eyes to see.

Perhaps, to improve the quality and the reliability of the responses supplied by the designer, it is not necessary to think of automating the design process at all levels. It seems more prudent and worthwhile to consider producing tools that are able to establish a symbiosis between the capabilities of the designer and those of the machine.

Developing such a system is still a task of outstanding difficulty, requiring the identification of the operations that can in reality be carried out by the computer and, among these, the choice of the ones that a design system must or should carry out, while at the same time defining the procedures of integration and dialogue with the designer.

The following pages illustrate the structure of a Knowledge-Based Assistant (which we call KAAD), which is being developed within the framework of a combined research project of the CAD laboratory of "La Sapienza" University's Faculty of Engineering - Rome , the Cartesiana Consortium and the State University of New York at Buffalo.

The purpose of this research is to produce of a software system that can establish an effective working symbiosis with the designer by making a complex, articulated representation of building objects, simulate their behaviour and verifying the design choices. The proposed system will be able to cooperate with the designer in:

- guarantee the rational nature of the design solutions;
- fostering the competence of the action and the quality of the proposal;
- permit the updating of technological and organizational solutions
- making an "historical memory" of solutions elaborated earlier and of the assessments of their behaviour with respect to the requisites laid down as the basis for any subsequent development.

The system can attribute a class (and hence a meaning) to every graphic note used by the designer, thereby enabling the semantic interpretation of sketches. Every declarative phase is matched by a series of deductive processes intended to determine all the consequences of the declarations made with the objective of defining a correct integrated representation of the design solution and determining the compliance of the choices made with regard to the assumed set of constraints. This process entails the determination of all the quantities necessary for the purpose. The designer is able to modify interactively the definitions given both at the level of the general scheme and regarding aspects of detail, observing the variations that these modifications entail in the quantities concerned. Choices not in keeping with the system of defined constraints are pointed out to the designer.

The system operates on a knowledge base aimed at producing a complex model of the objects as the designer defines them. The knowledge-base enables knowledge of the objects to be represented by defining classes of concepts and instances. The classes are defined by prototypes which correspond to an intentional representation of the objects involved in the design activity (Bobrow et al. 1975). Implementation of the system is based on the formalism of frames implemented in the Common lisp language and it uses a hybrid object-oriented programming paradigm.

The representation of the design goals is performed by expressing their objective aspects (requirements) and by defining their allowable values (performance specifications). The resulting system of requirements defines the set of all the possible building objects consisting of the set of characteristics (attributes and relations) which are considered relevant to represent the particular kind of desired building object with respect to the consistency check against the designer goals.

Generally speaking, there is no single object corresponding to an abstract representation but the whole class of the building objects that are equivalent with respect to the values assumed by the considered characteristics. The more we increase their number and their specifications, the smaller becomes the membership of the class, until it coincides with one single object - given that the assessed specifications are fully consistent. Moreover, the corresponding representation evolves up to the total prefiguration of the building object.

It is not possible, therefore, to completely define a building object deductively in advance, since it is inferred from the considered goals and is itself a result of the design process. All that can be established in advance in that the characteristics assumed to represent any building object are hierarchical, topological, geometrical and functional relations between the parts of the object - at any level of aggregation, components, space units, building units, the whole building (Carrara et al. 1990).

The system comprises four modules: the knowledge base, the evaluator system, the design goal database and the user interface and design control system.

The functional scheme of the proposed system is illustrated in figure 1.

The Knowledge Base (Kb)

The KB is the system software component making it possible to construct a conceptual model of the building organism and of its evolution at the necessary level of detail in the various phases of the design process. The KB contains three categories of objects which together create the representation which in the future will be termed prototypes, instances or objects and groups, respectively.

Prototypes

The KAAD system's knowledge base has been created by implementing a set of prototypes, slots and calculating procedures. The system of prototypes is a basic component of a frame-based knowledge representation system because it guides the inferential process of the system through the slot filling process.

Prototypes represent the generalization of a group of objects in the design sphere and constitute the knowledge base of the system (Gero et alt. 1988] In this work it has been assumed that the type of information the designer normally uses derives from a vision of reality of intentional type (Bobrow et alt. 1975, Hofstadter 1979). The representation adopted is therefore based on a discrete number of attributes, relations and "methods" made up of rules of the IF - THEN type or of calculation procedures.

The description of the characteristics of the objects is made by pairs of slot-facets. Each slot represents a particular characteristic of the object, while the facets represent the set of the values that can be assigned to that characteristic. It is possible through the definition of appropriate slots to represent information such as default values or maximum and minimum values.

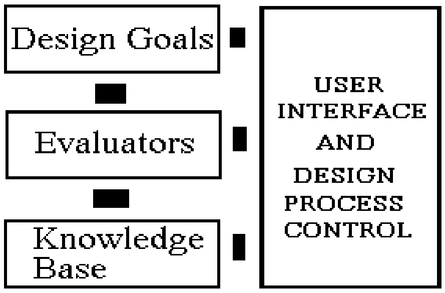

Figure 1. General structure of the system

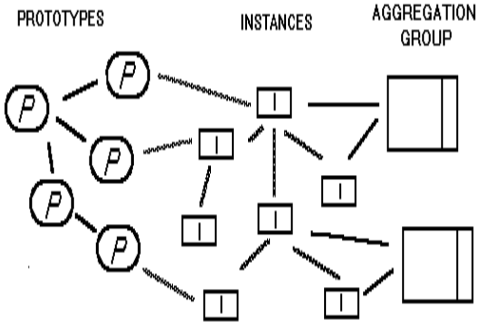

Figure 2. Internal structure of the Database.

A large number of prototypes relating to physical objects or existing spaces together defining a building organism have been pin-pointed and made operative in the kaad system. To describe the results achieved it is necessary to first set out certain definitions. In the following, a building will be considered as a structured set of spaces created by a structured set of physical elements. We will define a space unit as an equivalence class of spaces intended for the performance of the same activities for which the same environmental and dimensional characteristics are required. A building unit (B.U.) is an equivalence class of structured sets of spaces which recursively corresponds to a structured set of Building Units and Space Units (Carrara85).

A functional element (F.E.) is an equivalence class of physical objects (Building Elements) able to play a role in delimiting and classifying the space, in achieving an adequate level of safety, environmental comfort and visual perception. A Functional Subsystem (F.S.) is an equivalence class of structured sets of Physical Objects (Building Subsystems) which recursively corresponds to a structured set of Functional Subsystems or Functional Elements

The Space Unit Prototype

The Space Unit prototype is made up of SLOTS and FACETS which stem in part from the definition of the formalism adopted. Through the ISA and AKO slots (Winston 1984, Bobrow et alt. 1975, Carrara et alt. 1985) a hierarchical organization of the prototypes is made corresponding to different levels of generality of knowledge.

In addition to the slots and the facets foreseen by the formalism, further characteristics have been defined which derive from the nature and the particularity of the application that has been made thereof. In the case of the SPACE UNITS, the following slots are considered:

- IMS (immediate successor): this identifies the objects that together define the one under examination. The type of objects that can be contained in this slot are only functional elements. corresponding to the imp slot, a facet termed all (allowable set) is defined, identifying the ensemble of values admissible for the slot considered. In the case under examination the only values permitted are frames of the functional element type;
- IMP, which identifies the building units that the space unit considered helps to make. As in the preceding case, an „all facet“ has been defined which specifies only frames of building unit type as admissable values.
- ADJ („Adjacent“), which relates the unit considered with other environmental units making up the Building Organism. In this case a function is supplied in the prototype enabling the system to calculate the list of ADJ of the space unit considered on the basis of the arrangement of the constituent elements. This procedure, which within the context of frames formalism takes the name of procedural attachment, calculates the above values using the inheritance procedure activated by the inquiry functions. Each functional unit which contributes towards defining the space unit considered is "requested" for the list of the IMPs from which the space units adjacent to the one considered are obtained.

Those defined are only some of the possible relations of hierarchical and topological type that generally come into play in the definition of a building object. Generally, in fact, all flows of information, of persons, energy or things lead to the consideration of a high number of complex relations whose detailed description falls beyond the scope of the present paper.

ISA	VALUE	S.U.
AKO	VALUE	PROTOTYPE
IMS	VALUE	Some Functional subsystems
IMP	VALUE	Some Functional subsystems
SHAPE	VALUE	some LISP procedures
	TO-CALC	some LISP procedures
ADJ	VALUE	Some adjacent space units
	TO-CALC	some LISP procedures
NETSUP	VALUE	14.50 Mq
	TO-CALC	some LISP procedures

Apart from characteristics such as those referred to above, which we could define as being of "relational" type, physical characteristics of the objects have also been defined, such as for example net and gross areas, net volume and gross volume, and dispersed heating power. These characteristics are mainly represented by calculation procedures which, however, thanks to the procedural attachment allowed by the frames formalism, are handled by the system in the manner of data. All the procedures defined base their functioning on cooperation with other objects of the knowledge base.

Thus for example the procedure that determines the dispersed power simply asks for the necessary heat characteristics, through a series of functions, from the various functional elements which together define it. Each functional element is able to answer this type of request because it too possesses procedural attachments permitting this type of procedure. This type of approach, called object oriented (Tello 1989), offers countless advantages and in the case under examination it enables a highly flexible system to be produced.

The Building Unit Prototype

A building unit, as stated in the definitions, is made up of a structured ensemble of space units. As the space unit prototype, the building unit prototype foresees the isa, ako and ims slots. In this however the admissible values for the imp slot are only frames of building unit type, while those for the imp slot may be of space unit and building unit type.
In the case of building units, too, all the construction characteristics such as shape, high and squ (area) are defined by procedural attachments which contrary to Those relating to the space unit do not directly calculate the relative values but only "request" and summate the corresponding values of the elements hierarchically subordinate thereto. By definition, a building unit does not in fact possess form and dimensions of its own, these deriving from the respective characteristics of the subordinate elements.

FLAT-F1		
ISA	VALUE	BU
AKO	VALUE	PROTOTYPE
IMS	VALUE	- some space units and/or building units

Functional Subsystem Prototype

The Functional Subsystem prototype is structurally similar to that used to represent building units. In this case, too, the IMS and IMP slots define, respectively, the objects that compose them or that the functional Subsystem helps to define. On the basis of the definitions given, each functional element may be made up either of functional Subsystem or of Functional Elements.

FS		
ISA	VALUE	Functional Subsystem
AKO	VALUE	PROTOTYPE
IMS	VALUE	- some space units and/or building units

The Functional Element Prototype

The prototype that represents a Functional Element is from a conceptual stand-point wholly similar to the one relative to space units. Whereas the space unit represents the "minimum" element of what is generally defined as space sub-system, the Functional Element may be thought of as the basic element of the technological sub-system. Contrary to what happens for the space unit prototype, in this case no IMS slot is foreseen. That is, the Functional Element is regarded as a directly available physical object or in any case not requiring any specific design measure except in the definition of the ways whereby it helps to define more complex physical objects. The IMP slot contains the indication of the functional elements which the building element helps to define.

FE		
ISA	VALUE	PROTOTYPE
IMP	VALUE	Some Functional Subsystems
SHAPE	VALUE	Some Views

Views Of Objects

In The Functional Element Prototype The Geometrical Characteristics Of The Objects Represented Are Considered. In Fact The Slot That Contains An Indication Of The Views That Define The Physical Shape Of The Object Is Called The Shape Slot. The Views Are Flat Parametric Views (In Section Or 3-Dimensional) Of The Building Elements. Each View Is Provided With Conditions Of Activation And With One Or More Procedural Attachments Which Have As Their Side Effect That Of Displaying The Geometry Of The Objects. The Views Are Defined In A System Of Local Coordinates And Are Repositioned In Space Every Time An Instance Of The Building Element Prototype To Which Said View Is Linked Is Defined. The Procedural Attachment Present In The View Is Accommodated In The Shape Slot And Is Constituted By A Lisp Function That Contains The Calls To The Functions Of The Graphic Subsystem Which Enable The Object To Be Drawn. The graphic functions which the user can use are all operators defined on the hybrid edge data structure (kalay). Formally every shape is a function of p1....pn points in the space and Of An Unlimited Number Of Parameters. Although The Views Are Mainly Accommodated By Functional Elements, The Possibility Has Nevertheless Been Foreseen Of Giving Their Own Shape Also To Objects Of The Knowledge Base Which, At Least From A Logical Stand-Point, Do Not Possess This As Functional Elements, Space Units And Building Units. In This Way, These Objects Of The Knowledge Base Globally Through A Simple Graphic Interaction.

View		
Isa	Value	Prototype
Shape	Value	
	To-Calc	- Calls To The Functions Of The Graphic Subsystem

Groups

A Further Prototype That Has Been Defined In The Knowledge Base Is The Group Prototype. A Group Is Made Up Of The List Of The Objects Together Defining The Group. When The List Defining The Group Includes Instances, It Is The Single Object That Becomes A Member Of That Particular Group. If Instead A Prototype Is Included In The List, The Whole Class Of Objects Depending On It Becomes Part Of The Group.

Group		
Isa	Value	Prototype
Elements	Value	- Some Prototypes Or Instances Of Objects

Definition Of The Groups Takes On Great Importance Because Often The Specification Of Certain Characteristics Is Supplied Precisely By Using Aggregations Of Objects Which Follow Different Logics From That On The Basis Of Which The Prototypes Of The Knowledge Base Have Been Defined

The Evaluator Subsystem

It Has Been Shown In The Preceding Pages How, Through The Potentials Offered By The Procedural Attachment, It Is Possible To Represent A Number Of Aspects Of The Building Objects Which Imply The Determination Of New Values Based On The Information Contained In The Knowledge Base. There Are However Situations For Which This Type Of Approach Cannot Be Followed. Many Of The Indications That Are Normally Necessary In Design Activity Have Mostly Already Been Codified In Efficient, Reliable Calculation Codes. Therefore It Does Not Appear Realistic To Think Of Rewriting Them In The Context Of A New More General System Because Such An Effort Would Be Prohibitive.

In Addition To The Preceding Considerations, Which We Could Call Operative In Character, There Exist Also Methodological Type Motivations. The Use Of A Complex Calculation Code Always Implies On The Designer's Part A Schematization Of The Problem It Is Intended To Tackle. The Schematization Of The Problem Is In Fact Another Design Activity Calling For All The Designer's Critical Capacity And Experience. It Does Not Therefore Appear Possible To Crystallize The Carrying Out Of This Process In A Calculation Procedure Or Method, Otherwise There Would Be The Risk Of Obtaining Results That Would Be Formally Correct But Absolutely Senseless.

Goals

One Of The Aspects Characterizing The Proposed System Is The Possibility Of Defining Constraints That Have To Be Respected By The Choices Adopted During The Course Of The Design Activity. Two Types Of Constraints Have Been Considered: Constraints Of Natural Type And Those Of Design Type.

Constraints Of Natural Type Define Limits Of Variation Dictated By Common Sense, And Their Purpose Is To Guarantee The General Coherence Of The Knowledge Base. A Natural Constraint Is Represented By Means Of The Introduction Into The Prototypes Of Specific Facets Relating To The Characteristics On Which It Is Intended To Place A Constraint. In The Following Example Natural Constraints (Max And Min Range) Are Defined For Some Characteristics Of The Apartment Flat-F1.

Flat-F1		
Isa	Value	Bu
Elements	Value	Some Prototypes Or Instances Of Objects
Sup	Value	Actual Value
	Max	Max Value
	Min	Min Value
°C	Range	Range Values

The "Design Constraints" (Which We Call Requirement) Are On The Other Hand Defined By The Designer In Order To Establish Performances Required Of The Building Object, And They May Be Represented By Special Frames. Design Constraints May Be External, If Due To Cogent Normative, And Internal If Defined Directly By The Designer. Design Type Constraints Enable The Designer To Define Complex Relations Among The Objects Of The Knowledge Base. The Use Of This Mechanism Makes It Possible For Example To Specify Constraints Relating To The Objects Which Together Define The Spaces Subsystem (Space Units And Building Units) And The Physical Elements Subsystem (Functional Elements And Building Elements). Thus For Example Conditions Can Be Defined Which Establish That A Given Building Unit Must Consist Of A Given Ensemble Of Space Units And Building Units Or That A Given Group Of Building Elements Must Be Correlated On The Basis Of A Given System Of Rules.

Req-11		
Isa	Value	Requirement
Subject	Value	- Some Prototypes Or Instances Of Objects
Sup	Max	Max Value
	Min	Min Value

The Definition Of The Constraints Of Design Type Assumes Great Importance In Defining The Modalities Whereby The Objects Of The Knowledge Base Interact Together At The Level Of Both Prototypes And Instances.

The Requisite Prototype Foresees The Definition Of Two Slots, The First Of Which Defines The Field Of Application Of The Requisite Or, In Other Words, The Ensemble Of Objects To Which It Is Applied, While The Second One Establishes Which Characteristic Is To Be Constrained And The Values There Should Be For This.

The Definition Of A Constraint Requires The Specification For A Given Slot Of The Fields Of Variation Of The Values That This Can Accommodate. To This End The Frames Formalism Offers The Possibility Of Defining Facets Dedicated To This Purpose. A Number Of Examples Of Slots Are Set Out Below.

- Range. The Range Facet Permits The Definition Of A Continuous Variation Interval Of The Values Associated With A Slot. In The Range Facet The Minimum And Maximum Values Of This Interval Are Therefore Stored.
- Set. The Set Facet Serves Instead To Define A Discrete Variation Interval Of The Values. If A Value Is Specified For Said Characteristic, The Program Will Thus Have To Compare The Given Value With The Ensemble Of Those Permitted.
- Class. The Introduction Of A Class Facet Is In Answer To The Need To Define That A Value Belongs To A Given Class Of Values. If A Number Of Classes Are Specified, The Programme, Beside Accepting The Values Specified, Will Make Provision Also For Their Classification With Respect Thereto.
- Max, Min. The Max And Min Facets Define The Maximum And Minimum Thresholds Of Acceptance For The Values Of The Slots. This Is The Simplest Type Of Constraint Among Those Considered Because It Asks The Program To Control Only The Acceptability Or Otherwise Of The Values Introduced.

The User Interface And The Control System

The User Interface Is Responsible For Communications Between User And System And It Enables The Functioning Thereof To Be Controlled. The Interface Has Been Made Using Development Instruments Based On The Use Of Dialogue Windows And Pointing Systems Such As The Mouse And The Graphic Panel.

The Designer Can Use The System In Various Ways. He Can, Through A Traditional Graphic Subsystem, Make Preliminary Layouts Of The Object Which He Defines Without Initially Specifying Any Information As To The Nature And Type Of The Signs That Have Been Drawn. At Any Moment, It Is Possible To Go Over To A Declaration Phase In Which The Geometrical Entities Being Drawn Are Declared To Belong To A Class Of Objects Already Defined In The System, Which The Designer Has Previously Defined Or Whose Definition And Creation Are Simultaneously Carried Out. Whenever The System Possesses Sufficient Data To Identify Clearly An Object As Belonging To A Given Class, Automatically, Through An Inheritance Process The Object Concerned Inherits All The Characteristics Specified For The Prototypes On Which It Depends. Through This Process, The Ensemble Of The Natural And Design Constraints Defined Is Propagated To The Single Instances. The Type And The Number Of Characteristics That The System Controls Are Chosen However By The Designer Who Can Define Monitoring Windows. This Type Of Entity Present In The Knowledge Base Simply Makes Provision, At Every Designer-System Interaction Or If Specifically Requested By The Designer, To Evaluate The Aspect/S Whose Monitoring Has Been Requested. If The Goal Contains A Procedural Attachment That Must Be Activated In The Event Of Non- Respecting Of The Conditions Specified (As In The Case Of Goals Relating To Geometrical Aspects) It Is Automatically Activated By The System.

Conclusions

The system proposed in the foregoing is the first result of the definition of a different approach to the problems of designing and of Computer Aided Design in particular, stemming from an analysis of the experiments made earlier within the CAD context. On the basis of the definitions given, the formal structure of a support system for building and architectural design has been defined, able to simulate the behaviour of building objects and to enable the designer to work on them in a natural way, i.e. without influencing his design methodology.

Although experimental, the prototype constructed on the basis of the foregoing considerations has evidenced interesting potentials which will be analyzed and further studied in the on-going research activity.

References.

Aksoylu Y. "Two different Systematic Approaches to Design" technical report, University of California, Berkeley 1982

Bobrow D. G., Collins A. "*Representation And Understanding*" New York Academic press 1975

Carrara G., Novembri G.- "Constraint Bounded Design Search" in "*CAAD Futures*"□edited by A.Pipes - Butterwoths London 1986

Carrara G., Novembri G.- "KAAD a Knowledge Based Assistant for Architectural Design" - eCAADe proceedings of 4th European conference on Research and Teaching Experiences - Roma 1985.

Carrara G., Novembri G.- "Knowledge Assistant in the process of Architectural Design" *Building And Environment Vol. 25, No. 3, 1990* - Pergamon Press

Carrara G.,Fioravanti A, Novembri G.- "Towards a New Generation of Computer Assistant For Architectural Design: An Existing Scenario," In *Proceedings of eCAADe Conference Aarhus Denmark*, September 1989

Gero J.S., Maher M. L., Zhang W. "Design Knowledge and Representation", in *Artificial Intelligence in Engineering an Design*, Gero J.S Editor, ELSEVIER 1988

Gero John□S. "Knowledge Based Design Systems In Architecture" *eCAADe proceedings of 4th European conference on Research and Teaching Experiences* - Roma 1985.

Hofstadter D. R. - *Goedel Escher, Bach : An Eternal Golden Braid*" - Basic Books inc. 1979

Kalay Y. E. "*Computability of Design*", John Wiley & Sons, New York, 1987

Kalay Y.E. "Redefining the Role of Computers in Architecture: From Drafting/Modeling to Knowledge Based Assistants" *Computer Aided Design September 1985*

Novembri G. "Progettazione Edilizia e Tecniche della Knowledge Engineering" , *Università di Roma* " La Sapienza□ 1989.

Shaviv E., Gali D. " A Model of Space Allocation in Complex Building" *Building International* June 1979

Simon H. A. " *The Sciences of Artificial* " MIT Press 1969

Stiny G. "Introduction to Shape and Shape Grammars" *Environment and Planning B 7:343-351*, 1980

Swerdloff L.C. and Kalay Y.E. "A Partnership Approach to Computer Aided Design" in *Computability of Design (Kalay Ed),* John Wiley & Sons, New York, 1987

Tello E. R., "*Object-Oriented Programming for Artificial Intelligence*: *A Guide to Tools and System Design*", Addison-Wesley Publishing Company, 1989

Winston P. H. "*Artificial Intelligence* " Addison Wesley 1984

Winston P. H. and Horn B. K. P. " *Lisp* " Addison Wesley 1984

Woodbury R.F. "Strategies for Interactive Design System" in *Computability of Design (Kalay Ed)*, John Wiley & Sons, New York, 1987