CIFE CENTER FOR INTEGRATED FACILITY ENGINEERING

Knowledge-Based Creation of an Architectural 3-D Model from 2-D Drawings

by

Hisao Goto, Kincho H. Law and Greg Brickey

TECHNICAL REPORT
Number 59

December, 1991

Stanford University

CIFE Center for Integrated Facility Engineering • Stanford University

If you would like to contact the authors please write to:

c/o CIFE, Civil Engineering,
Stanford University,
Terman Engineering Center
Mail Code: 4020
Stanford, CA 95305-4020

Knowledge-Based Creation of an Architectural 3-D Model from 2-D Drawings[1]

by
Hisao Goto[2], Kincho H. Law[3] and Greg Brickey[4]

1 This research was supported by a seed research grant from the *Center for Integrated Facility Engineering*

2 Research Engineer, Takenaka Corporation and Visiting Fellow, *Center for Integrated Facility Engineering*
3 Associate Professor of Civil Engineering, Stanford University
4 PhD Student, Civil Engineering, Stanford University

Abstract

Drawings have been the primary means of communication in architectural design for centuries. Architectural drawings of a building are typically two dimensional orthogonal projections except for perspective and axonometric views. Each drawing alone does not represent the whole building; however, a collection of these drawings includes all the details necessary to describe a building.

3-D object-oriented design systems have been gaining popularity in the last few years. A 3-D object system requires features, geometry, relationships and attributes of 3-D objects. This information must be accessed quickly and efficiently in order to fully take advantage of a 3-D object system. However, there is no system today that is able to interpret multiple 2-D architectural drawings and to automatically create a 3-D building model. Presently, designers manually transform the drawings into a 3-D model which is a time-consuming and inefficient exercise. This paper describes a methodology for developing such a system.

To study the issues involved in the interpretation process, we implemented a prototype knowledge-based system called AutoMOD (Automatic 3-D Modeler). The prototype system utilizes a number of existing tools, including Nexpert Object[1] (for object representation and reasoning), AutoCAD[2] (for making input drawings and displaying the output 3-D model), THINK C[3] (for building the interface between Nexpert Object and AutoCAD) and HyperCard[4] (for controlling the overall interpretation process).

The AutoMOD system has been demonstrated successfully for interpreting simplified 2-D architectural drawings and for building a 3-D object model based on interpretations of the 2-D drawings.

---

1 Nexpert Object is a product of *NEURON DATA*
2 AutoCAD is a product of *AUTODESK Inc.*
3 THINK C is a product of *SYMANTEC Corp.*
4 HyperCard is a product of *Apple Computer Inc.*

ACKNOWLEDGEMENTS

This research is the result of collaboration of Mr. Hisao Goto of Takenaka Corporation and Professor Kincho Law of Stanford University. Many individuals have directly and indirectly provided advices and encouragements in this research effort. The authors would like to acknowledge the advice and encouragement of the following individuals:

- Professor Paul M. Teicholz, Director of CIFE, Stanford University.
- Dr. John C. Kunz, Senior Research Associate of CIFE, Stanford University
- Professor Raymond E. Levitt, Department of Civil Engineering, Stanford University.
- Professor Jean-Claude Latombe, Robotics Laboratory, Computer Science Department, Stanford University.
- Professor Robert D. Logcher, Department of Civil Engineering, Massachusetts Institute of Technology.
- Dr. Jonathan M. Cherneff, Department of Civil Engineering, Massachusetts Institute of Technology.
- Mr. Fujio Nishikawa, General Manager of Information Management Center, Takenaka Corporation, Japan.
- Mr. Yuzo Kato, Senior Manager of Information Management Center, Takenaka Corporation.
- Mr. Yasuo Endo, Manager of Information Management Center, Takenaka Corporation.
- Mr. Yasunaga Tatsumi, Deputy Manager of Information Management Center, Takenaka Corporation and Former Visiting Fellow of CIFE.
- Mr. Ken Ito, Senior Research Engineer, Shimizu Corporation and Former Visiting Fellow of CIFE.
- Mr. Hiroyuki Fuyama, Senior Research Engineer, Takasago Research and Development Center, Mitsubishi Heavy Industries, LTD., and Ph.D. student, Dept. of Civil Engineering, Stanford University.
- Dr. Yumi Iwasaki, Research Associate, Knowledge System Laboratory, Computer Science Department, Stanford University.
- Mr. Shigeomi Nishigaki, Manager of Information Processing Systems, Hazama Corporation.

We would also like to thank Ms. Chris Madison, Ms. Leyla Unerdem and the visiting fellows for making CIFE an enjoyable place to work.

TABLE OF CONTENTS

Section 1
Introduction

This section describes the background of this research work and introduces the objective and the scope of this study.

1.1 Background

Drawings have been the primary means of communication in architectural design for centuries. Architectural drawings of a building are typically two dimensional orthogonal projections except for perspective and axonometric views. Each drawing alone does not represent the whole building; however, a collection of these orthogonal projection drawings includes all the details necessary to describe a building.

3-D object-oriented design systems are gaining popularity in the last few years. An object design system requires a 3-D model that contains object information such as:

- features (what the object is - e.g., columns, beams, etc.)
- geometry (where the object is and its dimensions - e.g., (x,y,z) coordinates)
- relationships (how the objects relate to each other - e.g., connects to, contained by, etc.) and
- attributes (e.g., material, cost, etc.).

The 3-D model has to be constructed quickly and efficiently in order to take full advantage of the 3-D object design systems. However, there is no system today that is able to interpret multiple 2-D architectural drawings and to automatically create a 3-D model. Presently, designers manually transform the drawings into a 3-D model which is a time-consuming and inefficient exercise.

One notable work on a knowledge-based approach toward drawing interpretation was performed by Prof. Logcher and Dr. Jonathan Cherneff at MIT [4]. They implemented a Knowledge-Based system for Interpretation of Architectural Drawings (KBIAD) using the knowledge engineering environment, KEE[1]. The KBIAD system uses the geometric information of a simplified 2-D plan drawing to produce information regarding spaces, walls, doors, windows and millwork. However, the system was not intended to produce a 3-D model from multiple 2-D drawings.

There are two basic ways to automate the transformation process from 2-D drawings to a 3-D model by machine:

(1) Static interpretation which obtains the object information from interpreting existing drawings

Drawings may exist either on paper or in a drafting CAD system. In case where the drawings are on paper, scanning systems such as Scorpion's Raster to Vector Conversion System[2] can be used to pre-process the drawings into machine-readable form. Typically, a scanning system scans a drawing and records the drawing as raster data which are then converted into vector format such as DXF[3]. Existing commercial scanning systems are capable of normalizing a drawing and eliminating the "noises" to some extent; however, they are not capable of identifying features and relationships of objects nor building a 3-D model from multiple drawings. Consequently, the output DXF file contains only geometric information such as (x, y) coordinates. Likewise, drawings in drafting CAD systems store only geometric information.

(2) Dynamic interpretation which obtains object information while the drawings are being drawn

CAD systems have become popular, and some unique approaches are being taken in developing user friendly interfaces. The T-Board[4] system [8] is a free-hand drawing system that enables the user to draw graphic elements such as lines and circles on a liquid crystal display (LCD) with a stylus-pen, almost in the same way he/she draws pictures on paper with pencil. The system reasons about the graphic elements drawn and replaces them with precise vector information in real time. The T-Board system succeeded in removing traditional "pick and click" menu-based operations to declare the user's intention before taking any drawing action. The dynamic interpretation of features and relationships of

---

1 KEE is a prooduct of *IntelliCorp*.
2 Scorpion System is a product of *Scorpion Technology*.
3 DXF is Drawing Interchange File Format developed by *AUTODESK Inc.*
4 T-Board is a product of *Takenaka Corp.*

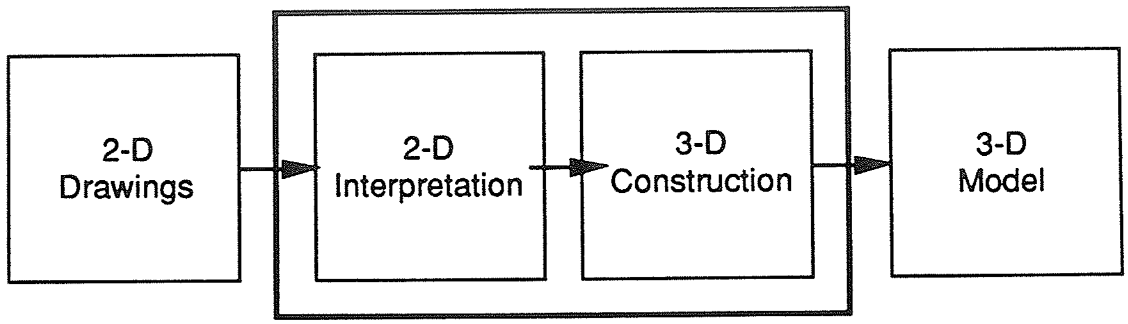


Figure 1-1: **Research Objective**

objects might be possible by embedding a knowledge-based interpretation system with a drawing system such as T-Board.

1.2 Objective

While a methodology for automatically creating a 3-D model from multiple 2-D drawings would be useful for both static and dynamic interpretations of drawings, the dynamic interpretation process is a difficult long-term problem. The static interpretation, taking existing drawings as an input to the AutoMOD system, is the primary focus of this research.

The objective of this research is to explore the issues in creating a knowledge-based system (AutoMOD) that:

- interprets existing multiple architectural drawings and
- constructs a 3-D model of a building using object-oriented data representation.

1.3 Scope of Study

The following describes the scope of study to facilitate our initial investigation in this research area.

(1) Utilization of commercial expert system shell

Commercial expert system shells provide ready-to-use rules, objects to represent domain knowledge, and a variety of inference mechanisms. The Nexpert Object system was chosen as a knowledge engineering development tool in this work for rapid prototyping.

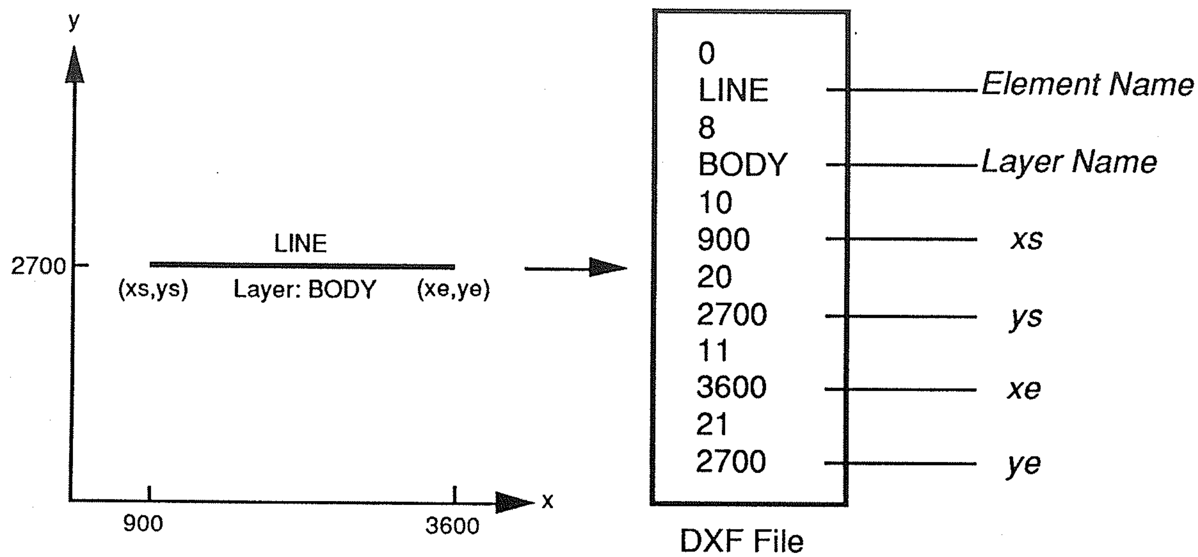


Figure 1-2: **DXF Information (Input to AutoMOD).** The figure on the left shows a single line in a x-y coordinate system. The figure on the right shows the DXF format for this line.

---

(2) Assumptions of input drawing

- **Kind of drawing:**

The contents of architectural drawings vary somewhat, depending on the type of facility and the phase of the project. This work focuses on architectural drawings of an office building drawn at preliminary design stage. Since it is difficult at this initial stage of investigation to deal directly with real drawings, we focus on the drawings that are simplified.

- **Data format of drawing:**

The AutoMOD system assumes that input drawings are in DXF files. Any drawing in DXF format can be an input to the AutoMOD system. DXF files contain only geometric information such as element type, layer name and (x,y) coordinates. Figure 1-2 shows an example of how a single line is stored in a DXF file.

- **Way of getting DXF information:**

Given only a paper drawing, DXF file may be produced by a scanning system which does automatic raster to vector conversion and then produces a DXF format output file. In

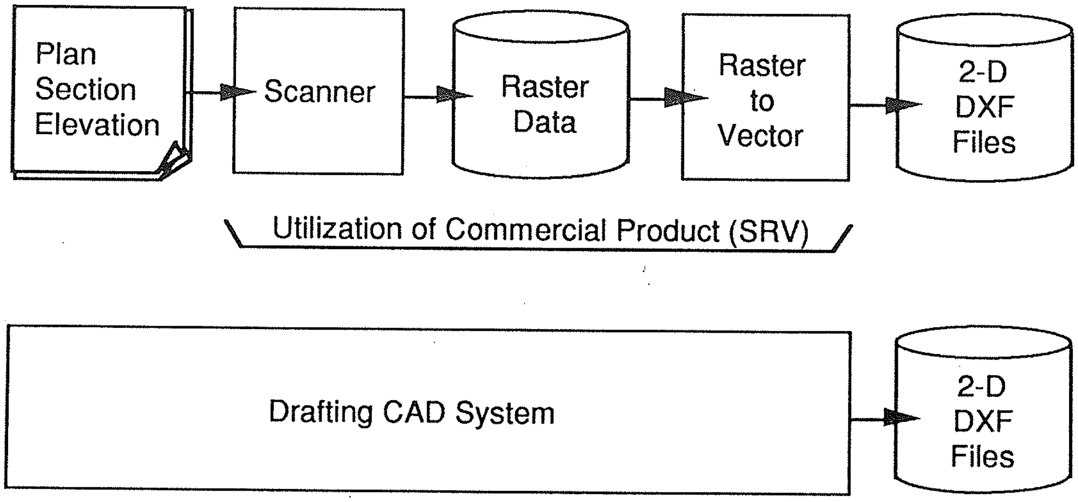


Figure 1-3: **Commercial Scanning and Drafting Systems.** Both systems produce 2-D DXF files which can be used as input to AutoMOD.

---

addition, most drafting CAD system can produce DXF file format outputs. In case where the drawings are on paper, utilization of a commercial product such as Scorpion is assumed. In case where the drawings are in a drafting CAD system, it is easy to obtain the DXF file directly from the CAD system since most CAD systems provide options to produce output in DXF format (Figure 1-3). In this work, AutoCAD is used for making input drawings and generating DXF information.

- **Accuracy and consistency of drawing:**

As long as the drawings are manually drawn or a drafting system is manually operated, inaccuracies and inconsistencies of the drawings are inevitable. Dealing with unpredictable errors on the drawings made by humans would be a difficult problem to solve in interpretation. Within the context of this work, input drawings to the AutoMOD system are assumed to be accurate and logically consistent among them.

- **Variation of drawing set:**

Different users may have different drawing sets. For instance, a user may want the system to create a 3-D model from the plan, section and elevation drawings while another user may have only plan and elevation drawings. Dealing with the infinite variations of drawing sets would be a difficult problem. The drawing sets that the AutoMOD system can handle are shown in Figure 1-4.

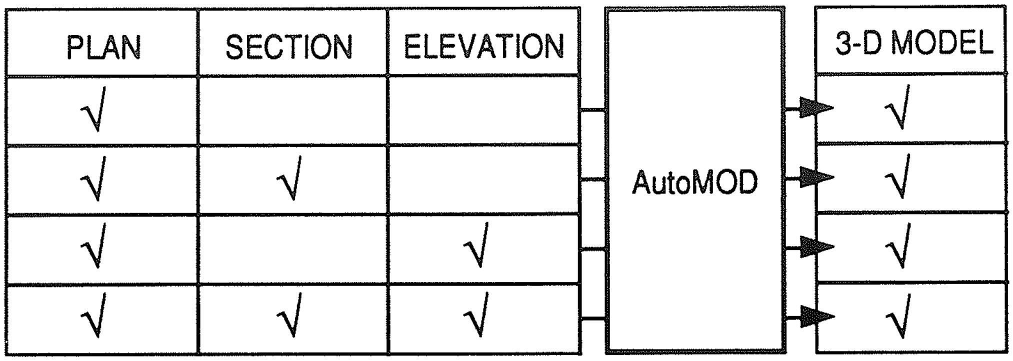


Figure 1-4: **Variation of Drawing Sets Supported by AutoMOD.** AutoMOD uses built-in assumptions when it detects ambiguities in a partial set of drawings.

The AutoMOD system employs either default values or existing values to derive missing information. For example, the system uses default values for generating height information of the 3-D objects if the user has only a plan drawing. An example of utilizing existing information is that the system retrieves the height information of east-side windows from south-side windows if the east elevation drawing is missing while the south elevation drawing exists.

(3) Representation of 3-D model

The AutoMOD system represents the resulting 3-D model by the descriptions of its objects. Attributes, one of four object information mentioned in the very beginning in this paper, greatly depend on words and descriptive information such as specifications (which is beyond the scope of this initial investigation). Therefore, object information in AutoMOD includes:

- features
- geometry
- relationships

Figure 1-5 schematically shows examples of 3-D objects. A column, for instance, includes ID number, Label which indicates the two column lines (i.e., column lines "1" and "A") where the column exists and its (x,y,z) coordinates. A girder , for instance, includes also the geometry and the relationships naming the two connecting columns (i.e., columns "1A" and "2A").

In order to visually examine the resulting 3-D model, the AutoMOD system converts 3-D object information to a 3-D DXF file. Again, AutoCAD displays the model.

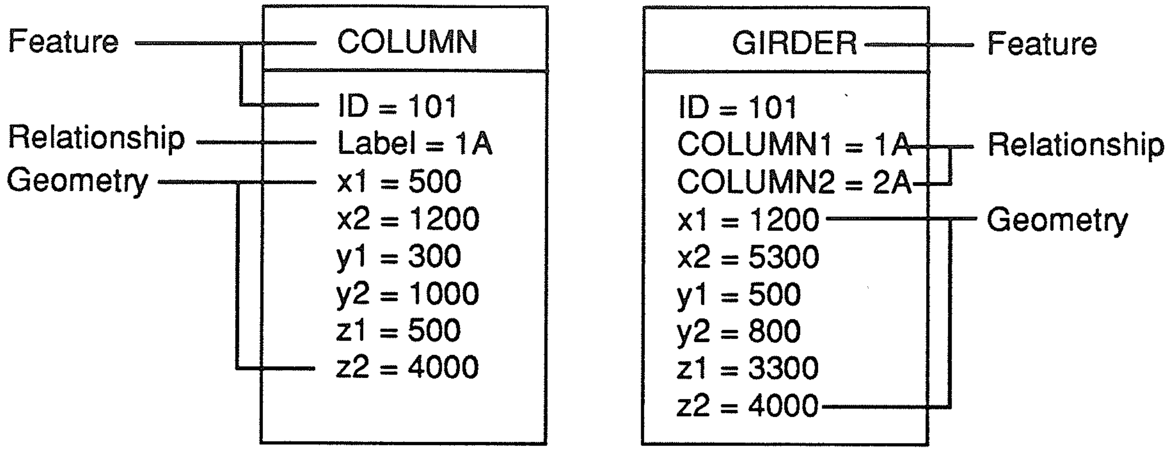


Figure 1-5: **Examples of 3-D Objects**

1.4 Summary of Results

The AutoMOD system successfully interpreted simplified multiple architectural drawings and constructed a 3-D model based on 2-D interpretations of the drawings. The overall schematic diagram of the system is shown in Figure 1-6.

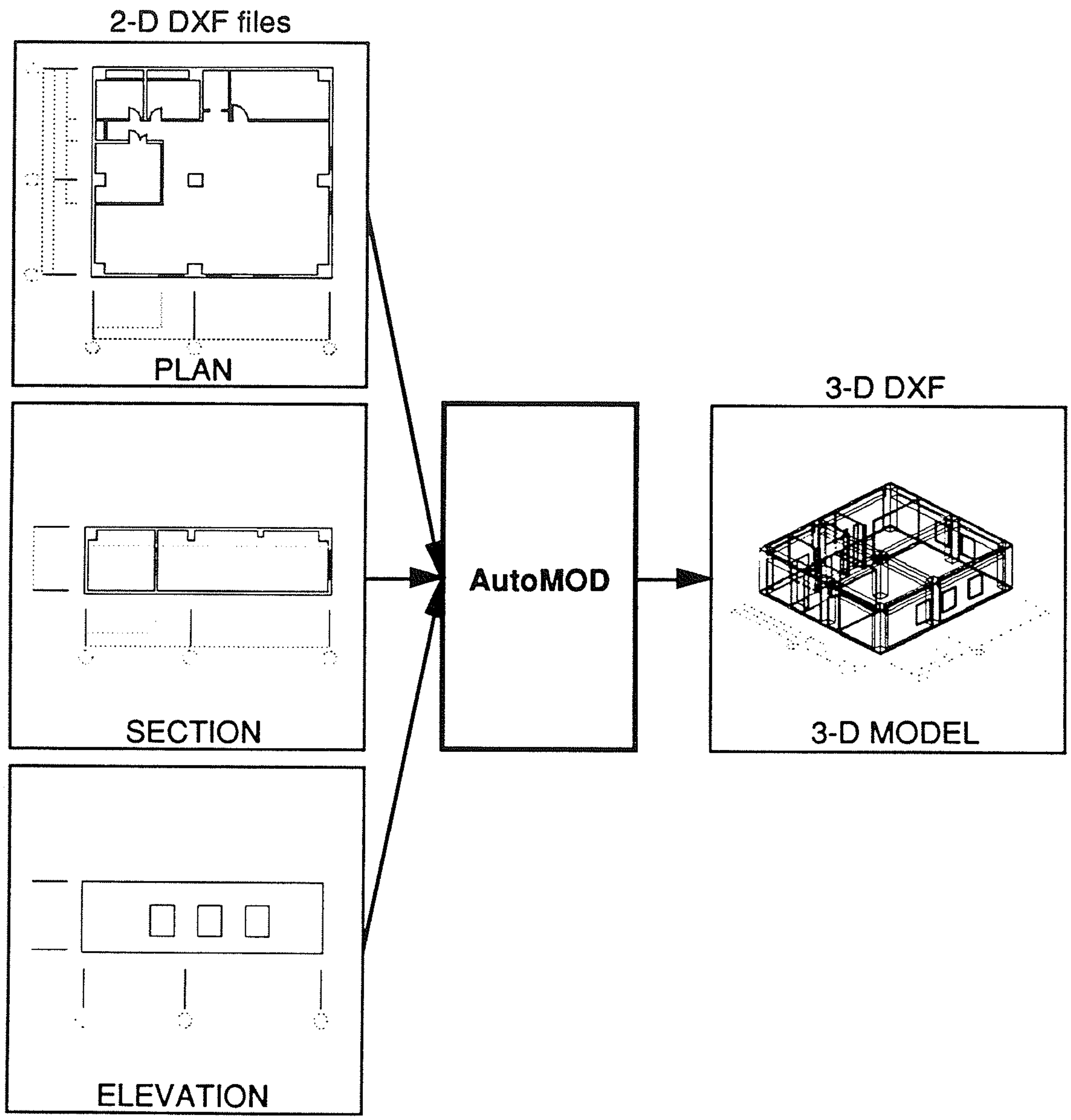


Figure 1-6: **Overall Diagram of AutoMOD.** 2-D drawings are interpreted to produce a 3-D model. The 3-D model is displayed by AutoCAD.

Section 2
Methodology

This section describes the methodology developed for 2-D interpretation and 3-D model construction. In this present study, we examine three basic issues relevant to this research problem.

(1) defining input drawings
(2) defining a human's drawing recognition model and
(3) converting the drawing recognition model into a computational model and defining the knowledge modules necessary to perform the (model-based) reasoning.

These issues are discussed in the following sections.

2.1 Input drawings

The drawing manual of Takenaka Corp., for instance, defines approximately twenty building components that a plan drawing should include at a preliminary design stage [21]. In this present work, however, we focus on identifying the following building components from the respective drawings:

- Plan Drawing: COLUMN, WALL, DOOR, WINDOW
- *Section Drawing:* WALL, WINDOW, GIRDER, BEAM, SLAB
- *Elevation Drawing:* WALL, WINDOW

The term "building component" refers to object including information such as feature, geometry and relationship.

Name of Layer	Type of Graphic Element	Type of Building Component
BODY	Either LINE or ARC	COLUMN, WALL DOOR, WINDOW GIRDER, BEAM, SLAB
COLLINE	LINE	COLUMN LINE FLOOR LINE
Any name except above two	Any type	Any type

Figure 2-1: **Rules on Drawing Organization.** This rule specifies the AutoMOD assumptions regarding CAD drawing layer, use of graphic elements, and allowed types of building components.

In addition, there are a few rules that must be followed when making the input drawings; they are summarized in Figure 2-1. The graphic elements of the building components are assumed to be either lines or arcs. Furthermore, graphic elements that constitute building components must be oriented in either the horizontal or vertical direction and stored in a layer called "BODY" in the DXF file.

The drawings include some crucial information for making it possible to correctly match different drawings such as plan and section drawings. For example, a column line is supposed to be common and consistent among the different drawings. The column lines in the AutoMOD system are straight lines stored in a layer called "COLLINE". In other words, the column lines need to be distinguished manually before the interpretation process (although this restriction should be eliminated in future development).

Building components other than the ones shown in Figure 2-1 can be drawn with any type of graphic elements and stored in any layer with name other than "BODY" and "COLLINE"; this information is ignored in the interpretation process.

An example of a real drawing is shown in Figure 2-2. Simplified versions of plan, section and elevation drawings are shown in Figure 2-3, 2-4 and 2-5 respectively. AutoMOD successfully interprets these simplified drawings.

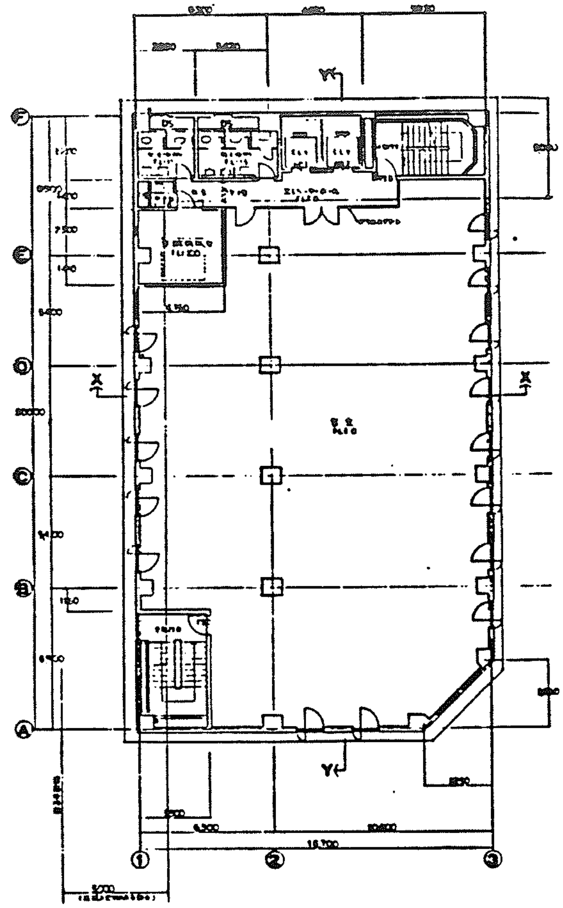


Figure 2-2: **Example of Real Drawing**

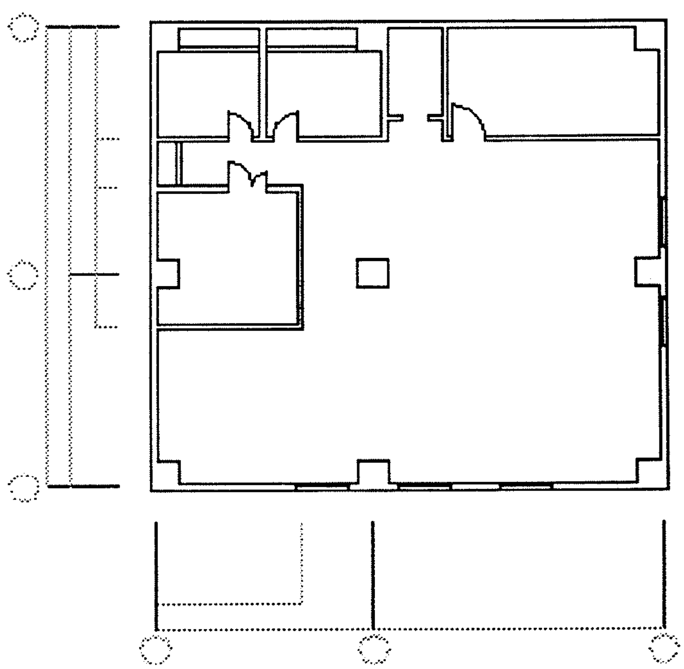


Figure 2-3: **Simplified Plan Drawing.** AutoMOD interpretes columns, walls, doors and windows in a plan drawing. Light color lines are ignored in interpretation.

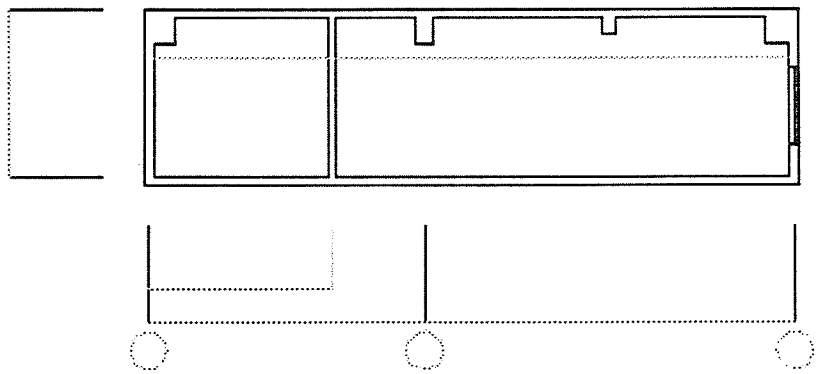


Figure 2-4: **Simplified Section Drawing.** AutoMOD interpretes walls, windows, girders, beams and slabs in a section drawing. Light color lines are ignored in interpretation.

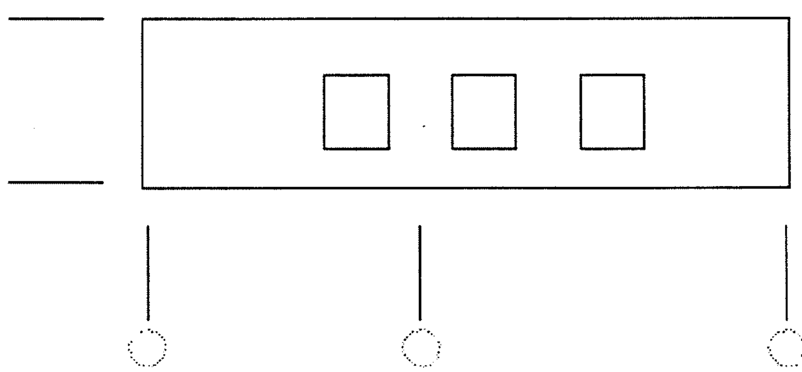


Figure 2-5: **Simplified Elevation Drawing.** AutoMOD interpretes walls and windows in an elevation drawing. Light color lines are ignored in interpretation.

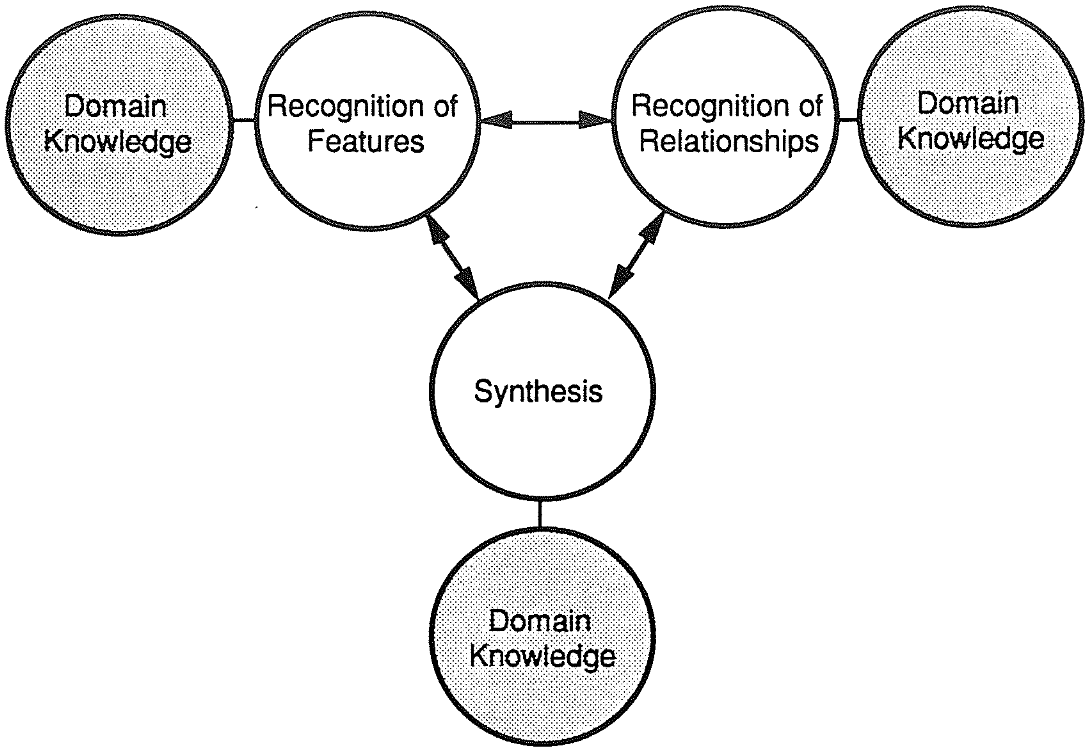


Figure 2-6: **Drawing Recognition by Humans**

2.2 Drawing Recognition Process

Architects/engineers construct 3-D images by examining 2-D drawings. The human recognition involves a *spiral-up* process toward a 3-D representation. This recognition process involves arbitrary interactions among three different activities:

- Recognition of Features - extracting features of building components based on their geometry from each of multiple drawings;
- Recognition of Relationships - recognizing relationships among the building components; and
- Synthesis - constructing a 3-D model by combining the features, geometry and relationships.

Each activity is carried out using the domain knowledge that architects/engineers possess (Figure 2-6).

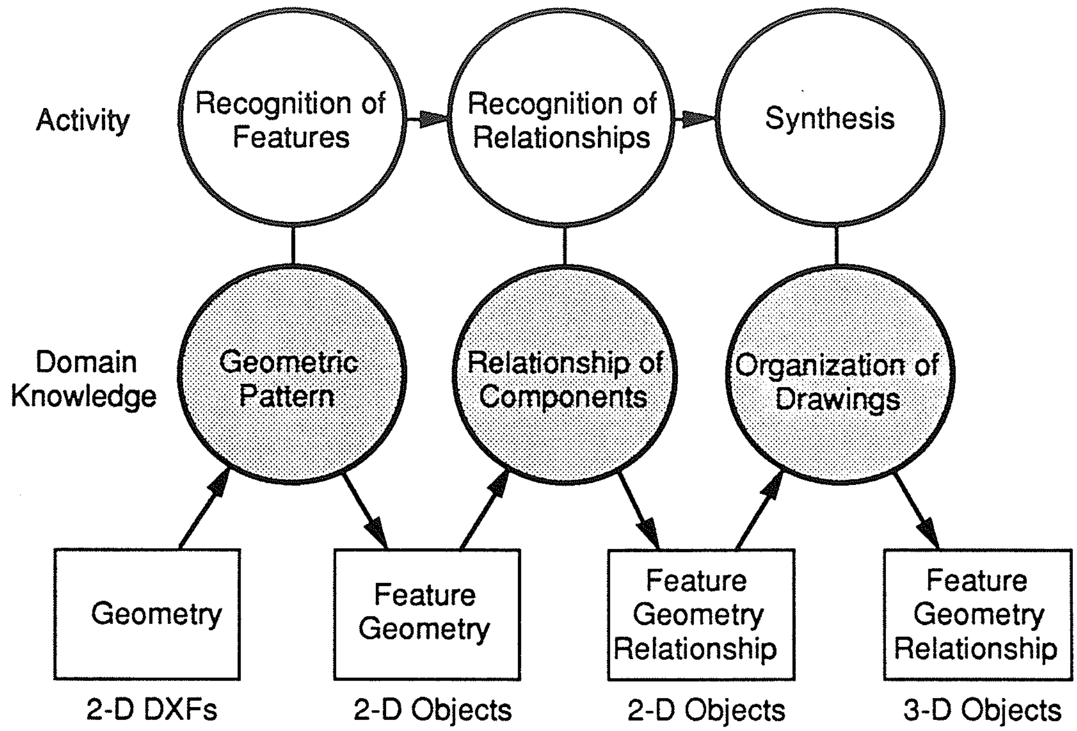


Figure 2-7: **Drawing Recognition by AutoMOD**

2.3 Knowledge-Based System

Since it is quite difficult to implement the human's recognition process, the recognition process needs to be transformed into a computational model.

One of the difficult issues for the machine to realize is the *spiral-up* process involving arbitrary interactions among the three activities shown in Figure 2-6. However, as long as the drawings are all accurate and logically consistent, the *spiral-up* process can be replaced with a linear process. In other words, all building components can be identified and a 3-D model can be built in a deterministic manner (Deterministic Reasoning). Therefore, in the drawing recognition model in AutoMOD, the three activities are executed in a linear fashion and each kind of domain knowledge is employed based on the result of the preceding activity (Figure 2-7).

Three knowledge modules corresponding to the three activities are implemented in the AutoMOD system:

- Geometric Pattern - describing how each component is drawn on a drawing in terms of its geometric features
- Relationship of Components - describing how each component is related to each other on a drawing
- Organization of Drawings - describing the relevant information contained in each drawing.

The first two modules concern with 2-D interpretation while the third module is for 3-D construction. In the next chapter, we will describe these three knowledge modules in detail.

Section 3
Implementation

This Chapter describes the implementation of the AutoMOD system. First, we provide an overview of the system. Three issues of the system design are then described:

(1) interface between drawings and reasoning
(2) knowledge modules and
(3) interface between 3-D objects and 3-D graphics

3.1 System Overview

The AutoMOD system utilizes a number of existing tools, including Nexpert Object [18] (for object representation and reasoning), AutoCAD [1] (for making input drawings and displaying the resulting 3-D model), THINK C [20] (for building the interface between Nexpert Object and AutoCAD) and HyperCard [9] (for controlling the overall interpretation process).

Input 2-D drawings made by AutoCAD are transformed into DXF files by using AutoCAD's native command (DXFOUT). An interface program written in THINK C is then used to transform the DXF files into Nexpert Object readable (NXP) files which contain only vector information, (x, y) coordinates. The reasoning process interprets the vector information and produces 3-D objects. The object information is converted back to a DXF file so that the resulting 3-D model can be displayed with AutoCAD. Figure 3-1 shows a schematic view of the AutoMOD system.

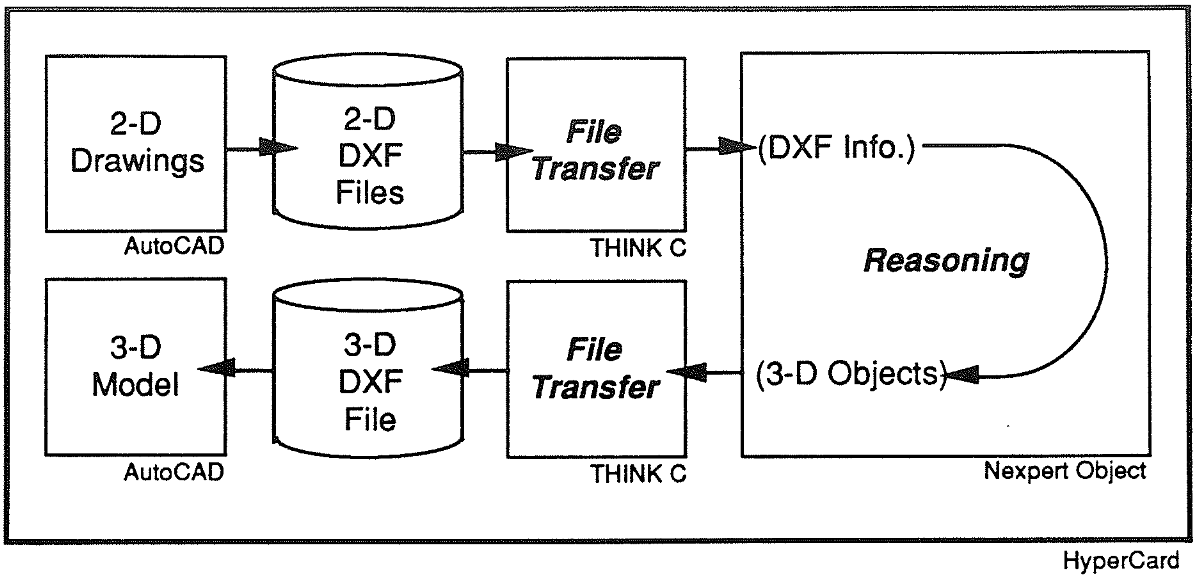


Figure 3-1: **Overview of AutoMOD**

3.2 Interface between Drawings and Reasoning

The following is an example of converting the DXF file of the plan drawing shown in Figure 2-3. The drawing contains 9 grid points in the layer COLLINE (three column lines in both vertical and horizontal directions), 128 lines in the layer BODY and 5 arcs in layer BODY while 22 elements in the other layers are ignored (the light color lines in Figures 2-3, 2-4 and 2-5).

Grids, lines and arcs are stored in three separate files in NXP file format as shown in Figure 3-2. The AutoMOD system reads in this information and transforms it into object representation as shown in Figure 3-3 and 3-4. The section and elevation drawings are transformed to the object representation in a similar manner.

```
     index|     label|        xc|         yc|
*****************************************
         1|        1A|     10000|      11000|
         2|        1B|     10000|      17000|
             |
             |
         8|        3B|     24000|      17000|
         9|        3C|     24000|      24000|
*****************************************
```

GRID

```
     index|         xs|         xe|         ys|         ye|
***************************************************
         1|      13075|      15600|      23900|      23900|
         2|      18175|      18175|      20875|      21775|
                 |
                 |
       127|      10700|      10700|      11100|      11700|
       128|      10700|      10100|      11700|      11700|
***************************************************
```

LINE

```
     index|        xc|        yc|          r|      angs|      ange|
********************************************************
         1|     18175|     20875|        900|         0|        90|
         2|     13125|     19475|        400|        90|       180|
                  |
                  |
         5|     12025|     20875|        700|         0|        90|
********************************************************
```

ARC

Figure 3-2: NXP Format

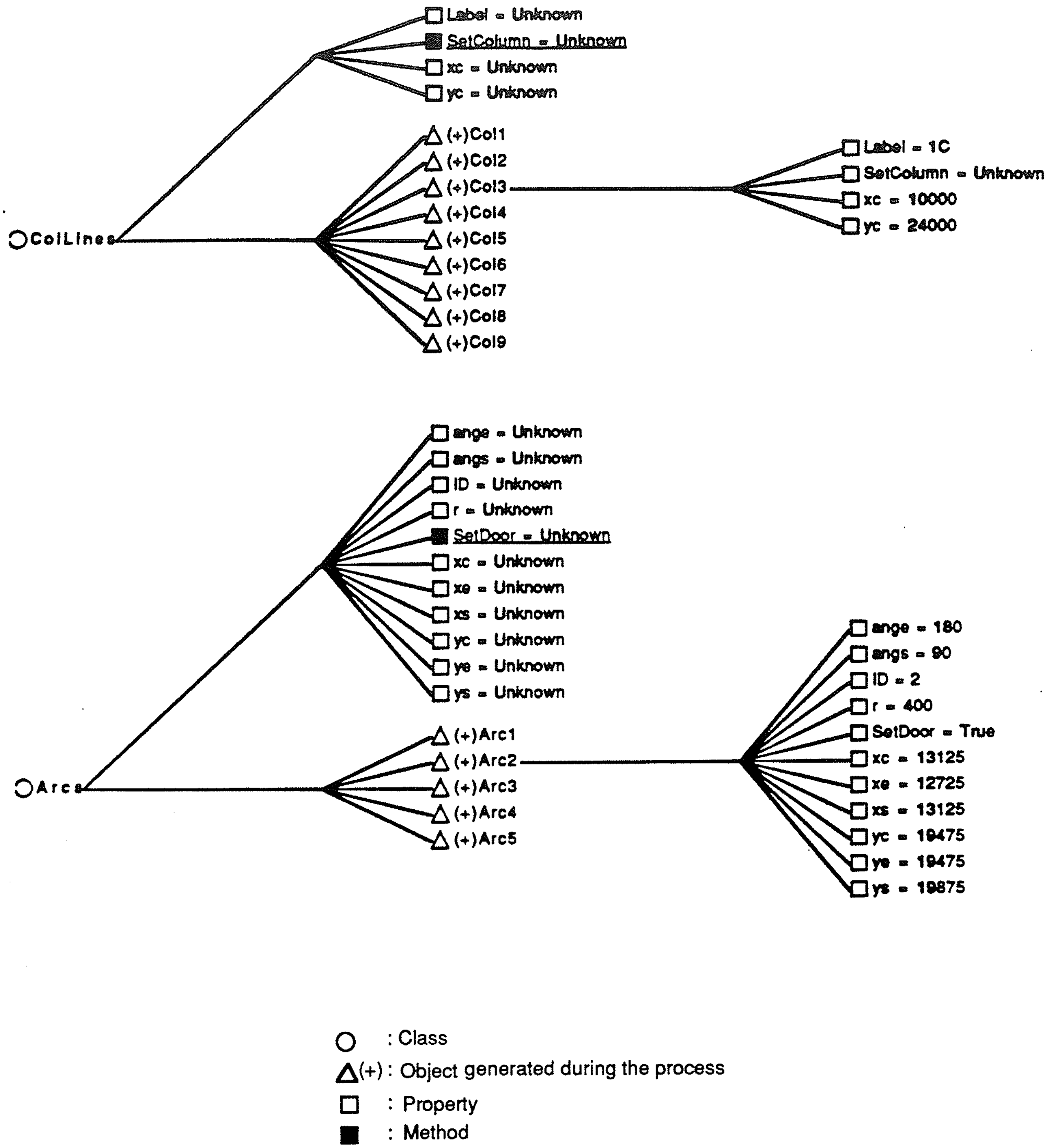


Figure 3-3: **Grid and Arc.** NXP information is transformed into objects whose names represent the type of graphic elements and whose properties contain 2-D graphic information such as (x,y) coordinates.

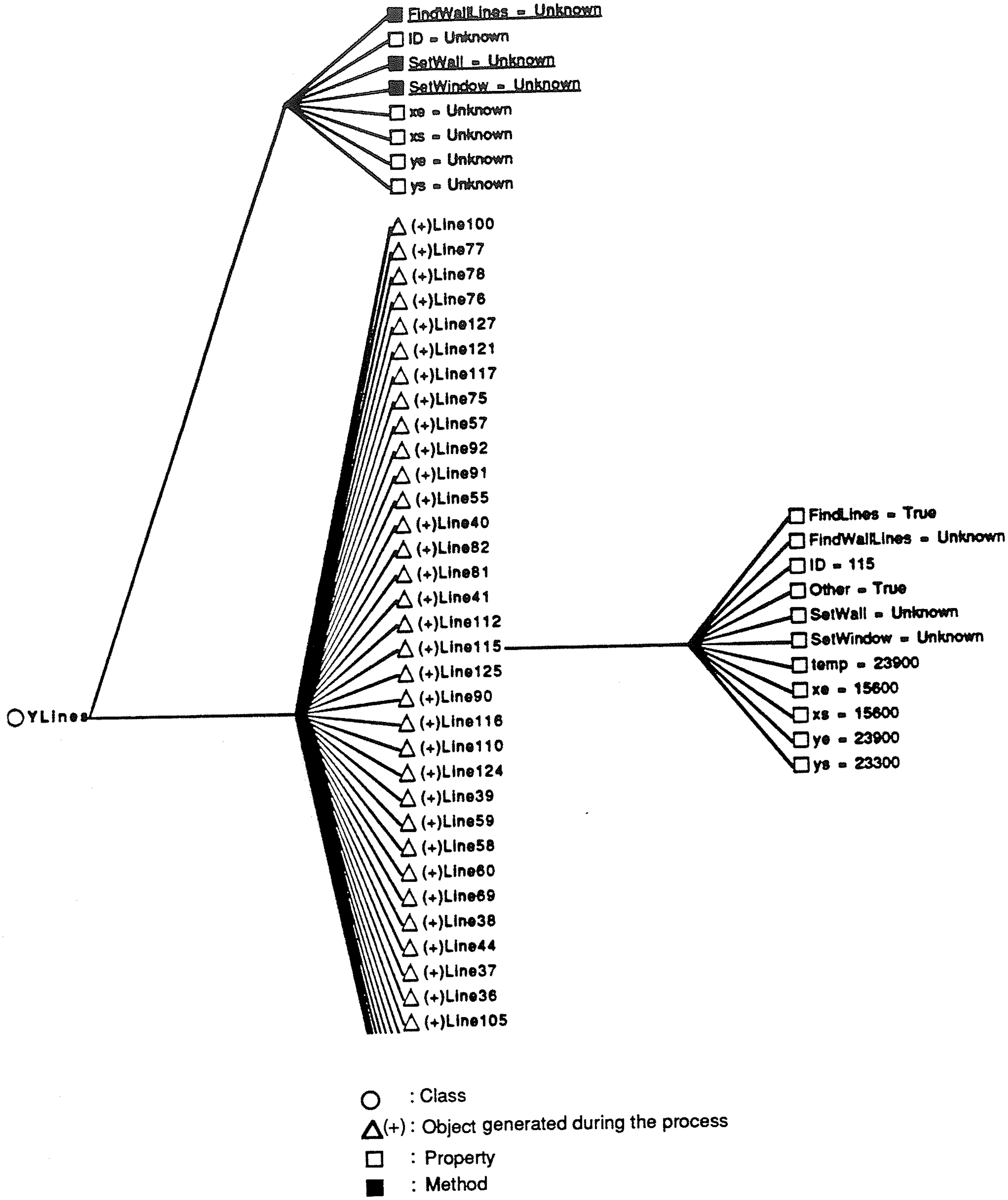


Figure 3-4: **Lines.** NXP information is transformed into objects whose names represent the type of graphic elements and whose properties contain 2-D graphic information such as (x,y) coordinates.

3.3 Knowledge Modules

There are three knowledge modules in the AutoMOD system:

- Geometric Pattern - for extracting building components (e.g., columns and walls) from each of multiple drawings
- Relationship of Components - for recognizing relationships among the building components and
- Organization of Drawings - for constructing a 3-D model by combining the building components and the relationships among them.

These three modules are described in the following subsections.

3.3.1 Geometric Pattern

The Geometric Pattern module consists of two components (Figure 3-5):

- Template: describing how each building component is drawn on a drawing in terms of its geometric features.
- Controller: providing the processing order of components to be found in each Template.

As soon as a building component is derived from the input information, the module removes the graphic elements or parts of graphic elements corresponding to the information which compose the building component (Figure 3-6).

The combination of the processing order provided by the Controller and the reduction of input data enables the Template to extract building components in a deterministic way. As an example, Figure 3-7 illustrates the interpretation of a door and walls on the plan drawing.

The Controller includes a set of rules describing the building components which must be preceded before a particular type of components so that the Template module can work properly. The preceding components are the surroundings of the component to be interpreted. The Controller allows the user to add a template for new building component to the system because it allows the user to focus on the geometric features of the new component to be interpreted. Figure 3-8 shows an example of the Controller for the plan drawing.

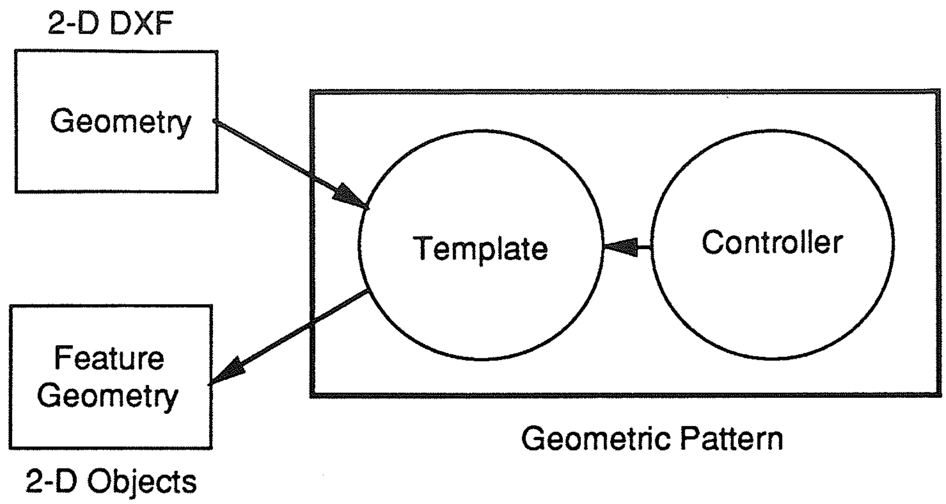


Figure 3-5: **Template and Controller**

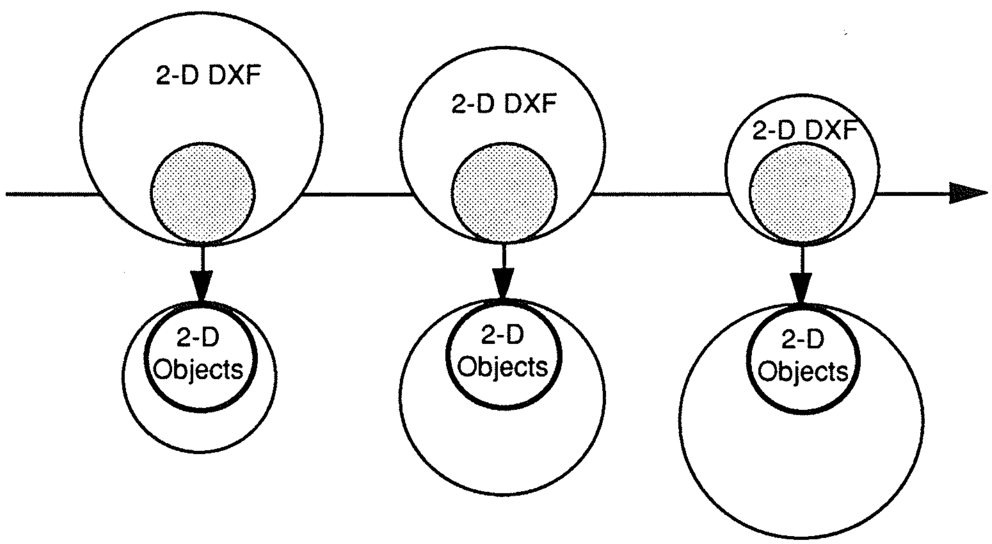


Figure 3-6: **Reduction of Input Data.** Graphic elements are recognized in the input, removed from the input and added to the output.

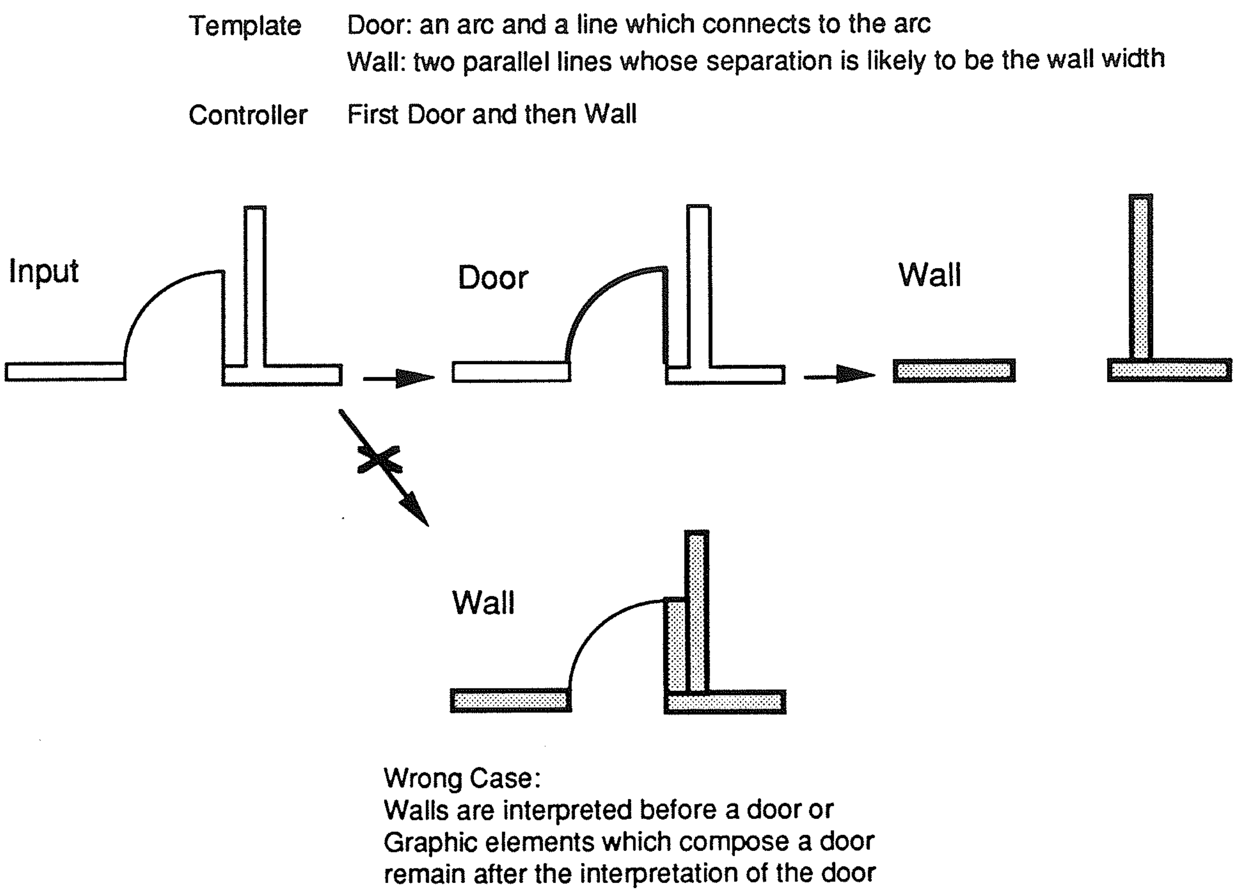


Figure 3-7: **Deterministic Reasoning.** Features are extracted in a fixed sequence set by the controller. If features are extracted in an incorrect order, as shown in the bottom of the figure, ambiguities may arise.

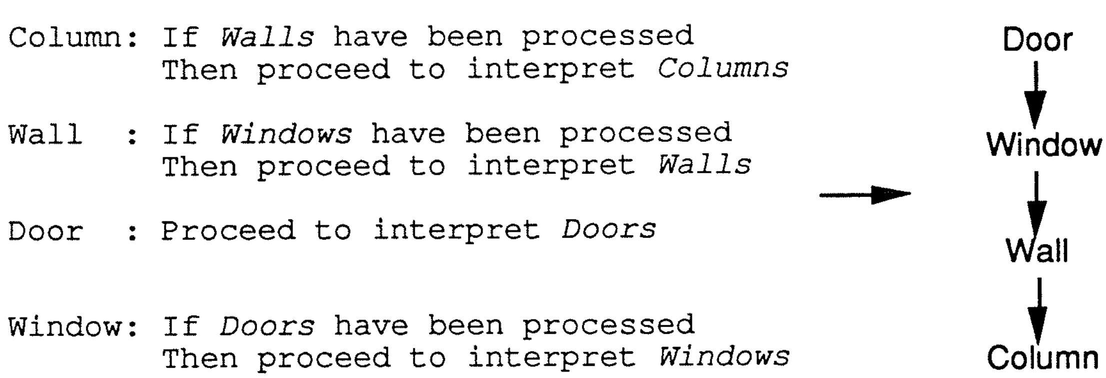


Figure 3-8: **Example of Controller.** The Controller specifies th eorder in which building components are interpreted.

Along with the actual processing order, the following summarizes the Template for interpreting the building components on each drawing.

1. Plan Drawing

(1) Door

The system extracts a door by finding an arc and a line which connects to the arc at one end and the other end is equal to the center point of the arc. The resulting object is represented as a rectangle to depict an opening in a wall and a line to represent the door itself. The line overlaps one of the longer sides of the rectangle and whose first point is at the hinge of the door.

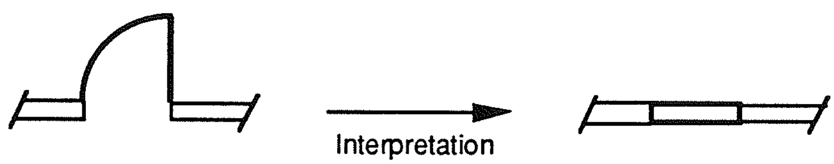


Figure 3-9: **Door on Plan**

(2) Window

The system identifies a window by finding three parallel lines which have same lengths and the separation among them is equally spaced within certain range (say <= 250mm). The resulting object is represented as a rectangle to depict an opening in a wall and a line to depict the glass of the window.

Figure 3-10: **Window on Plan**

(3) Wall

The system extracts a wall by searching for two parallel lines whose lengths are not necessarily the same but whose separation is less than 250mm. The resulting object is represented as a rectangle.

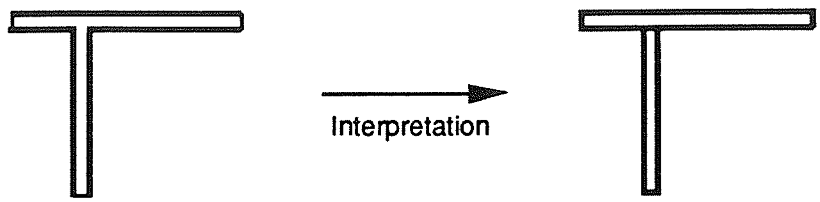


Figure 3-11: **Wall on Plan**

(4) Column

When the system reaches the interpretation of columns, building components which possibly connect to the columns have been already identified and relevant graphic elements have been removed from the input information.

Based on the assumption made in this work that each column includes an intersection point of two column lines, the system searches for lines which are completely included in a certain range around an intersection point of two column lines. The resulting object is represented as a rectangle.

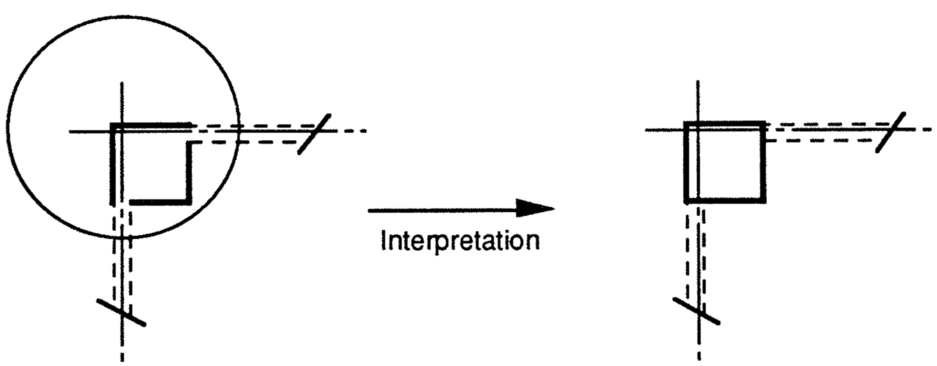


Figure 3-12: **Column on Plan**

2. Section Drawing

The Template for the section drawing is almost the same as the one for the plan drawing, since the two drawings are similar in terms of geometric features of the building components.

(1) Window

The Template used for the windows on the plan drawing is utilized for the section drawing except that the system only searches for vertical lines. The resulting object has a line which represents the glass of the window.

(2) Wall and Slab

The Template used for the walls on the plan drawing is utilized for the section drawing. The system searches for only vertical lines for the walls while horizontal lines are searched for the slabs. The resulting object is represented as a rectangle.

(3) Girder

The girders can be regarded as the columns on the plan drawing in terms of their geometric features. The only difference is that a girder includes an intersection point of a column line and a floor line instead of two column lines.

(4) Beam

When the system reaches the interpretation of beams, only relevant input information remain. The system identifies the beams by searching the graphic elements within a certain area. The resulting object is represented as a rectangle.

3 Elevation Drawing

The elevation drawing in this work has only two kinds of building components. To facilitate the interpretation, the system first distinguishes the lines into long lines and short lines.

(1) Wall

The system identifies a wall by searching for a rectangle composed by the longer lines. The resulting object is represented as a rectangle.

(2) Window

The system identifies a window by searching for a rectangle composed by shorter lines. The resulting object is represented as a rectangle.

3.3.2 Relationship of Components[1]

Relationship of Components describes how each component is related to each other in a drawing. Two types of relationships, "Connect to" and "Contain (or Contained by)" are defined in each drawing. The AutoMOD system extracts the relationships based on the 2-D object information such as features and geometry obtained by the Geometric Pattern module. The following shows the definition of relationships among the components per each drawing.

(Drawing)	(Object)	(Relationship)	
PLAN	COLUMN	Connect to	COLUMN LINEs(2)
		Connect to	WALLs (0-4)
	WALL	Connect to	COLUMNs (0-2)
		Connect to	WALLs (0-2)
		Contain	DOORs (n)
		Contain	DOORs (n)
	DOOR	Contained by	WALL (1)
	WINDOW	Contained by	WALL (1)
SECTION	GIRDER	Connect to	WALL (0-1)
		Connect to	SLAB (1)
	BEAM	Connect to	WALL (0-1)
		Connect to	SLAB (1)
	SLAB	Connect to	GIRDERs (n)
		Connect to	BEAMs (n)
	WALL	Connect to	GIRDER (0-1)
		Connect to	BEAM (0-1)
		Connect to	SLAB (0-1)
		Contain	WINDOW (0-1)
	WINDOW	Contained by	WALL (1)
ELEVATION	WALL	Contain	WINDOW (n)
	WINDOW	Contained by	WALL (1)

1 This feature has only been partially implemented.

3.3.3 Organization of Drawings

Organization of Drawings provides the knowledge about how to synthesize different drawings in order to build 3-D objects. The following shows the drawings that each component needs to refer in order to obtain its 3-D information. The drawing in parentheses is optional for the user to have.

(Object)	(Drawing)
COLUMN	PLAN, (SECTION)
WALL	PLAN, (SECTION)
DOOR	PLAN
WINDOW	PLAN, (ELEVATION)
GIRDER	(SECTION), PLAN
BEAM	(SECTION), PLAN
SLAB	(SECTION), PLAN

The key information to properly synthesize different drawings is Column Line which is common and consistent across the drawings. Figure 3-13 illustrates the synthesis of plan, section and elevation using column lines.

3.1 Interface between 3-D Objects and 3-D Graphics

In order to visually examine the resulting 3-D model with a CAD system, an interface program written in THINK C has been developed to convert the 3-D object information to a DXF file. The program puts the feature information of the object into layer of the DXF file. A column object, for instance, is converted to 12 lines in DXF format and then displayed as shown in Figure 3-14.

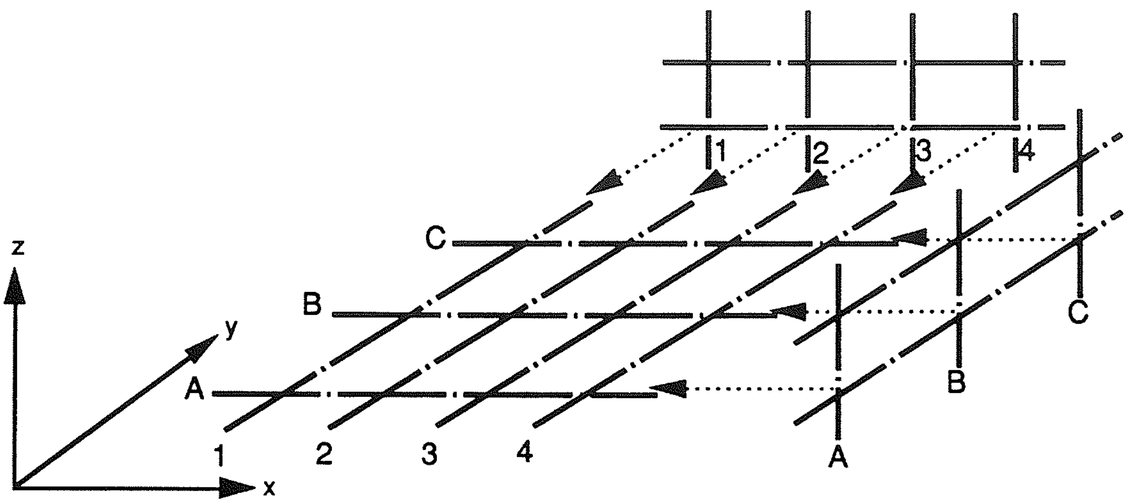


Figure 3-13: **Synthesis by Column Line's Information**

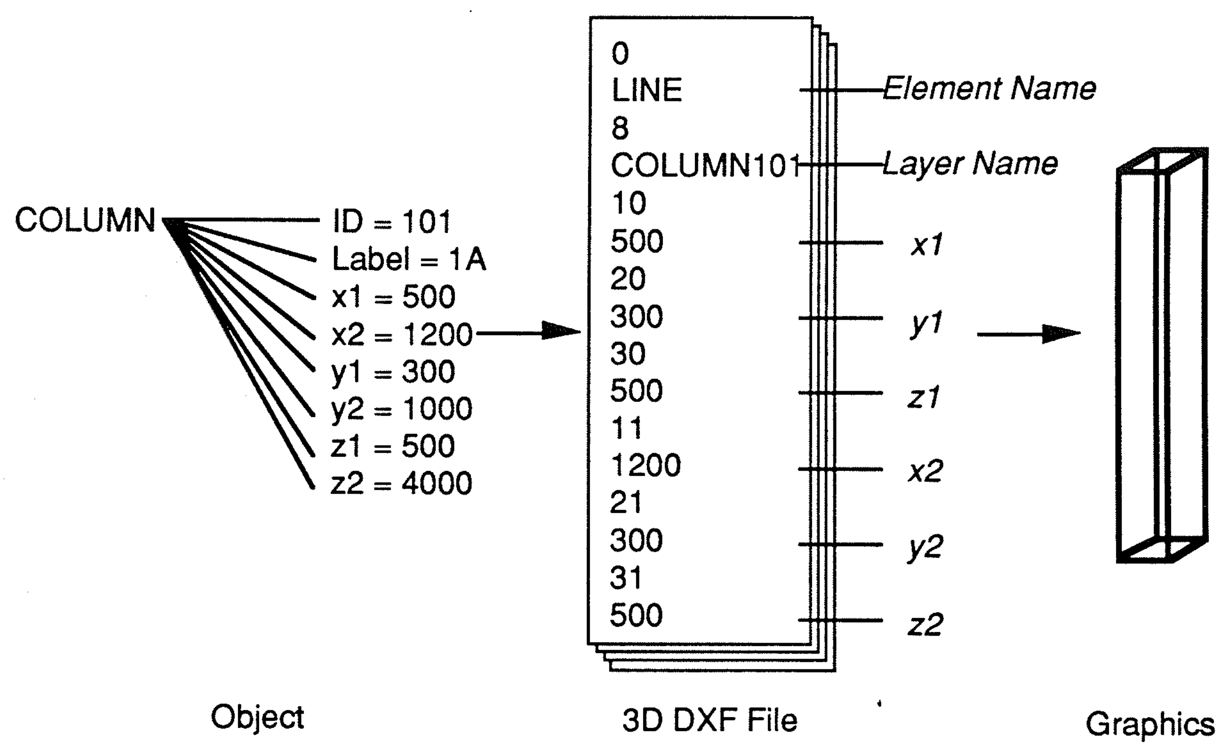


Figure 3-14: **Object to Graphics**

Section 4
Example

This section shows how the AutoMOD system looks during the process of interpreting 2-D drawings and constructing a 3-D model. In this example, the drawings shown in Figures 2-3 to 2-5 are used as input drawings.

AutoMOD has an user interface built in HyperCard. The interface allows the following functions:

(1) Showing 2-D Drawings
(2) Transforming DXF Files into NXP Files
(3) Reasoning
(4) Transforming 3-D Objects into a 3-D DXF File and
(5) Displaying a 3-D Model.

The AutoMOD takes about four minutes to convert the 2-D drawings of Figures 2-3 to 2-5 to the 3-D model of Figure 4-13 on Macintosh IIcx.

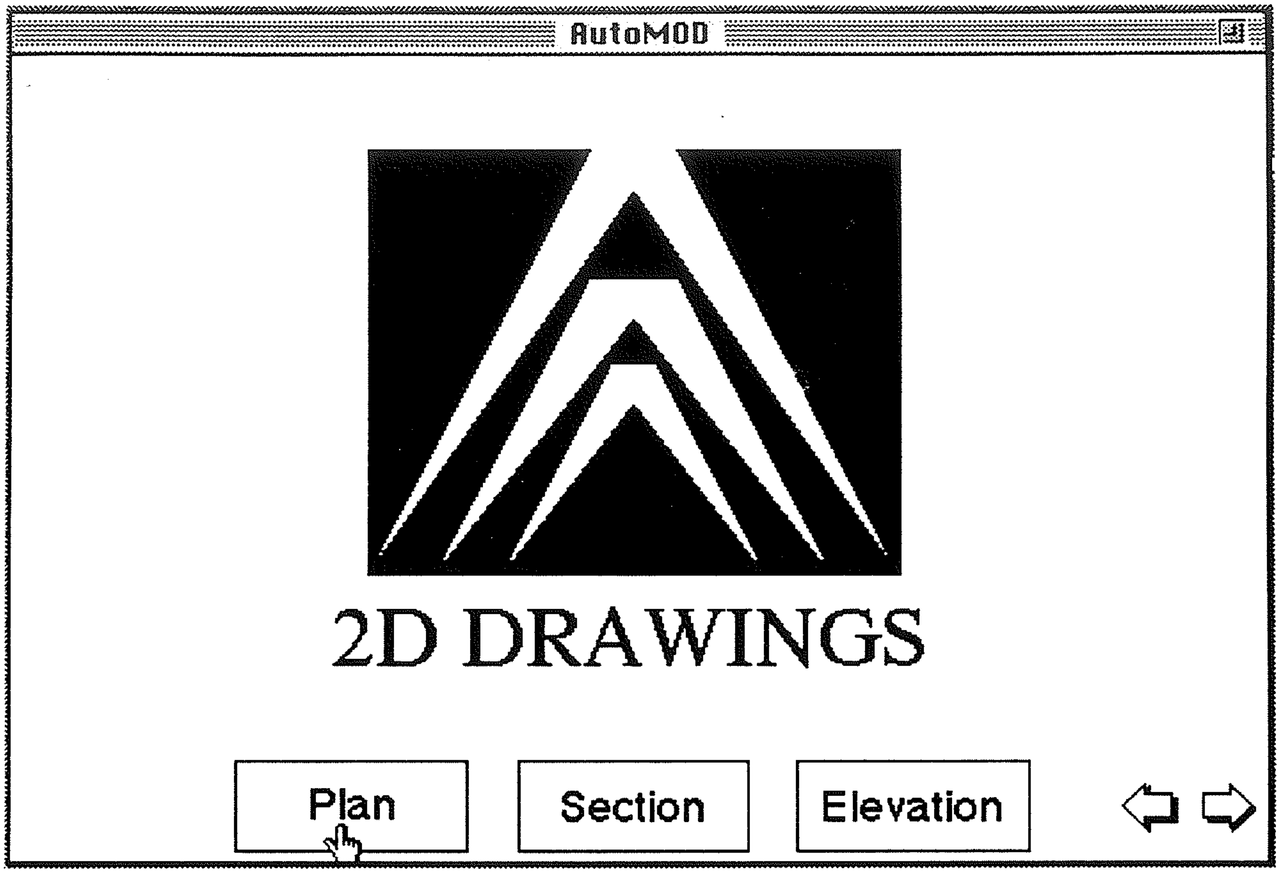
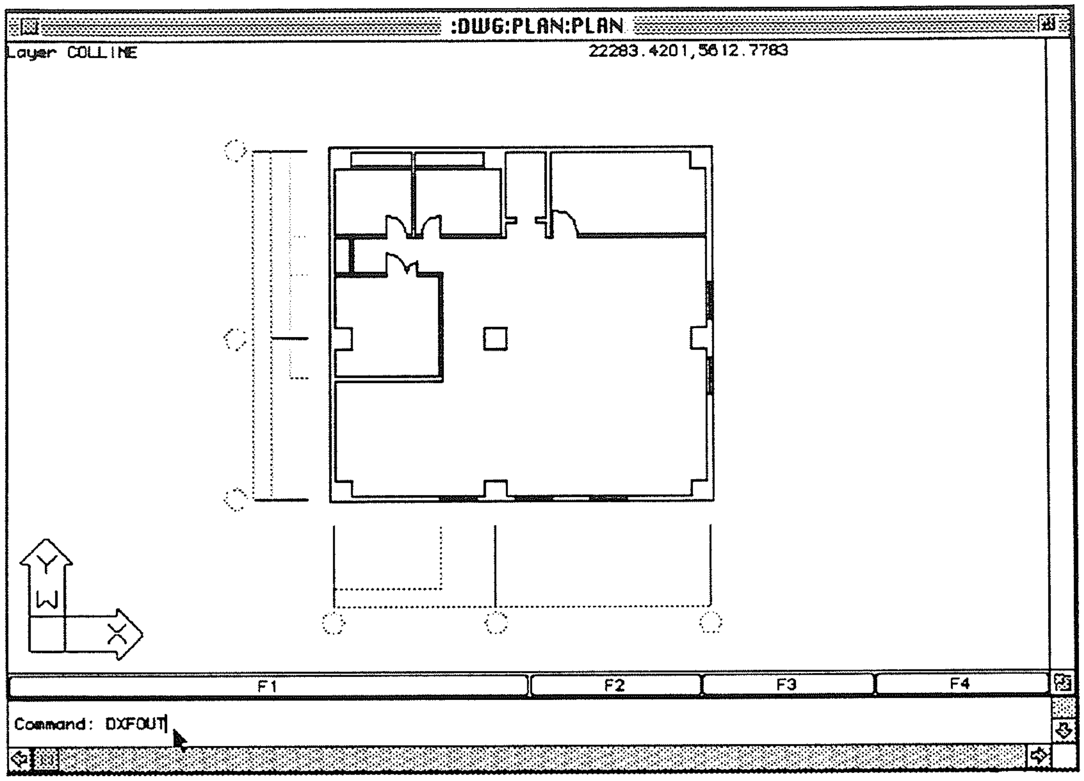


Figure 4-1: **HyperCard for Bringing Up Input Drawings on Screen**

(1) 2-D Drawings

When the AutoMOD system is activated, the screen shown in Figure 4-1 is displayed. The user can bring up existing drawings by clicking any one of three buttons on the bottom of the screen to look at the drawing and to generate the DXF file with AutoCAD's native command, *DXFOUT* (See Figure 4-2).

All building components are stored in the layer, "BODY" and all column lines are stored in the layer, "COLLINE" while other building components are stored in other layers. Figure 4-3 shows an example of layers for an input drawing.



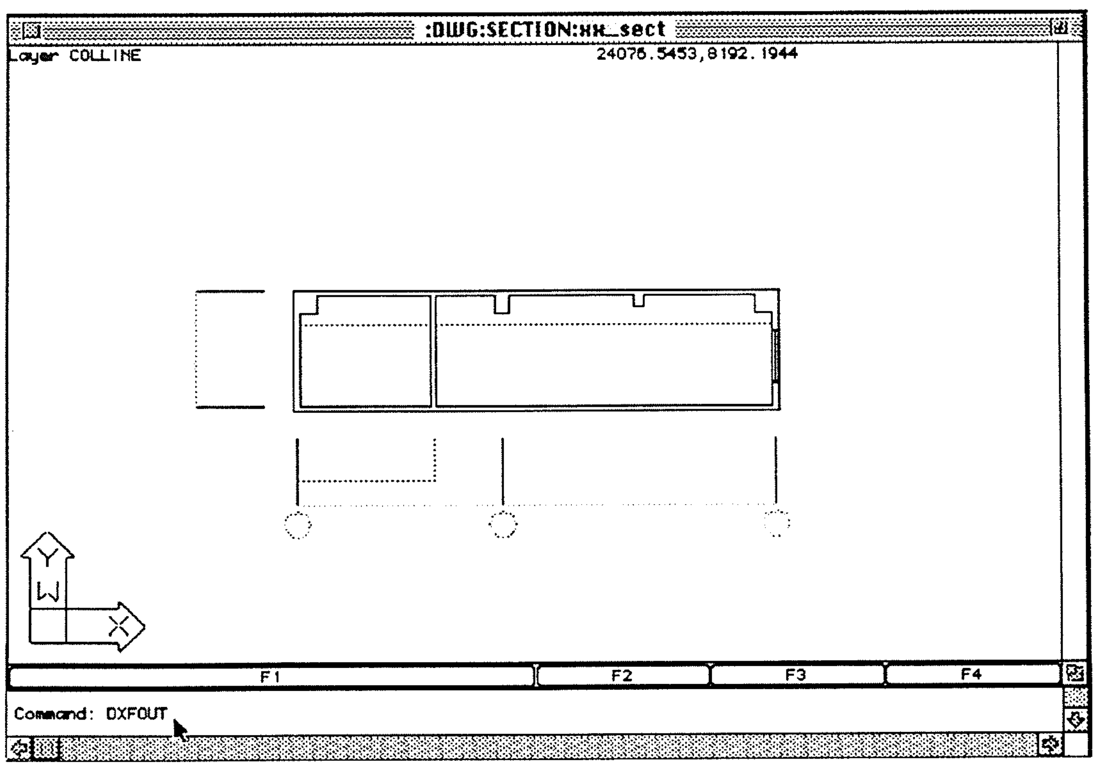


Figure 4-2: **Input Drawings on AutoCAD (Plan and Section)**

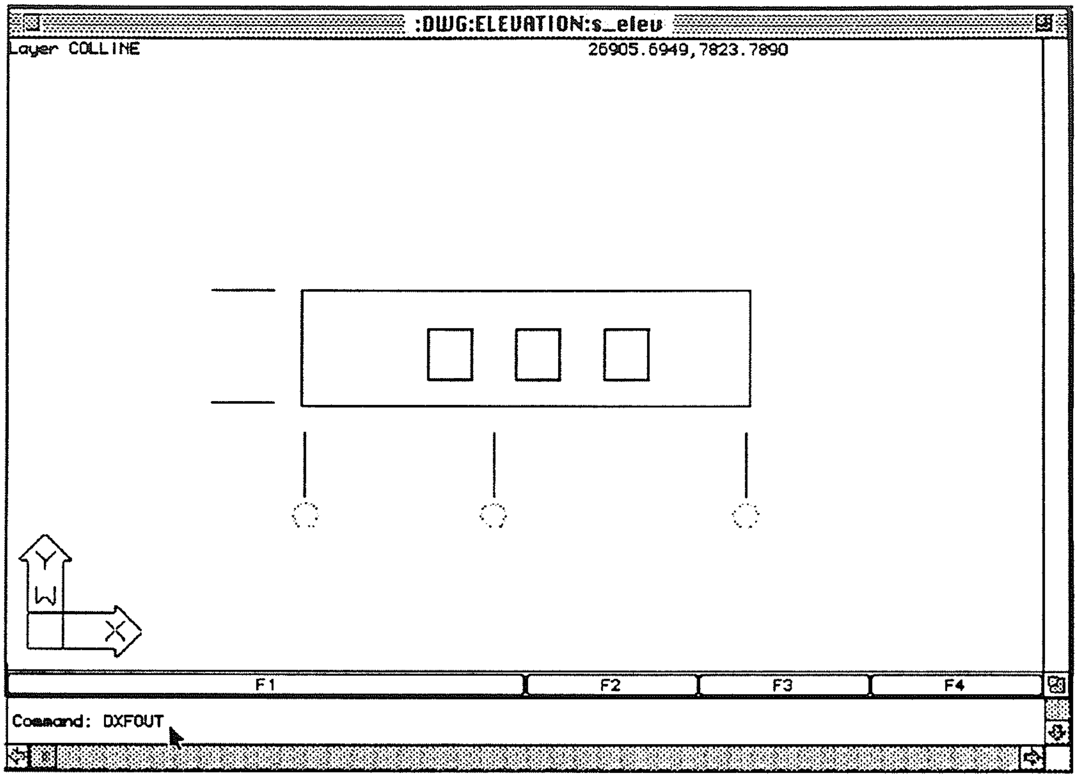


Figure 4-2 (Cont'd): **Input Drawings on AutoCAD (Elevation)**

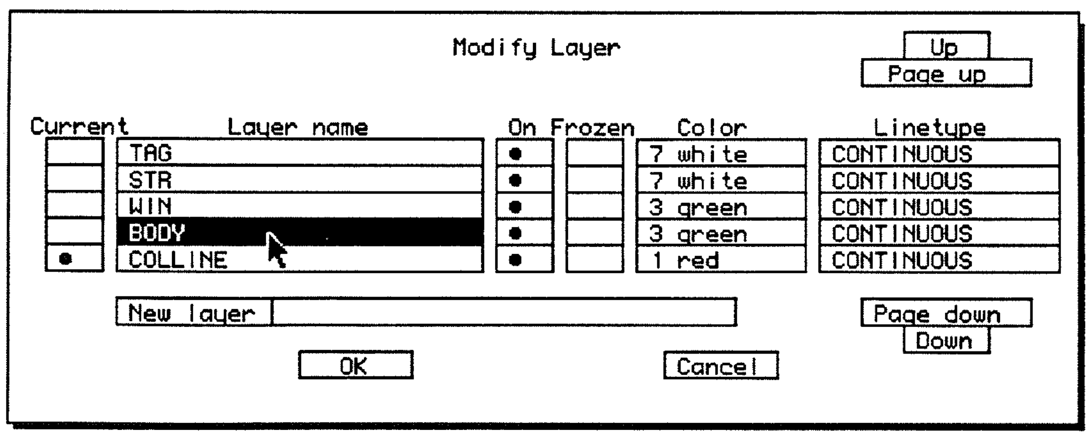


Figure 4-3: **Example of Layer of Input Drawing.** Building components are all stored in the BODY layer.

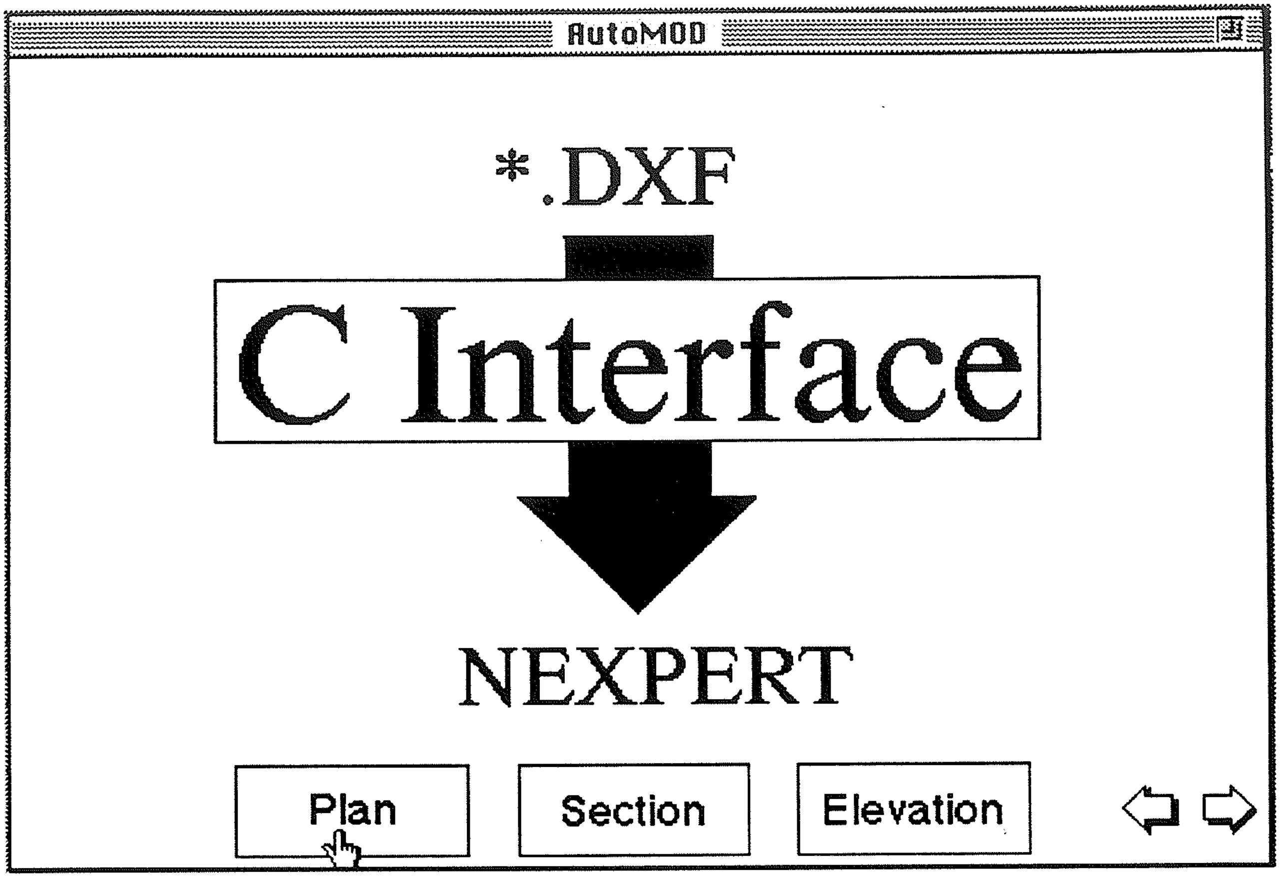


Figure 4-4: **HyperCard for Invoking the C Interface to Transforming DXF Files into Nexpert NXP Files.**

(2) Transforming DXF Files into NXP Files

As shown in Figure 4-4, the user can now select any one of the three interface programs for transforming a DXF file into a NXP file. The results are displayed in Figure 4-5.

```
press «return» to exit

        ********************************************
        *            FILE CONVERSION               *
        *   AUTOCAD DXF FILE TO NEXPERT NXP FILE   *
        *           FOR  PLAN DRAWING              *
        ********************************************

Input DXF file :? plan.dxf

*** input DXF file          =: GOTO HD:DATA:PLAN_DXF:plan.dxf
*** output NXP file (grid ) =: GOTO HD:DATA:PLAN_NXP:table.nxp
*** output NXP file (line ) =: GOTO HD:DATA:PLAN_NXP:lines.nxp
*** output NXP file (arc  ) =: GOTO HD:DATA:PLAN_NXP:arcs.nxp

*** number of grid  = 9
*** number of line  = 128
*** number of arc   = 5
*** number of other = 22
|
```

```
press «return» to exit

        ********************************************
        *            FILE CONVERSION               *
        *   AUTOCAD DXF FILE TO NEXPERT NXP FILE   *
        *          FOR SECTION DRAWING             *
        ********************************************

Input DXF file :? x_sect.dxf
Orientation    :?
                1 for x-x
                2 for y-y          :1

*** input  DXF file         =: GOTO HD:DATA:X_SECT_DXF:x_sect.dxf
*** output NXP file (grid ) =: GOTO HD:DATA:X_SECT_NXP:table.nxp
*** output NXP file (line ) =: GOTO HD:DATA:X_SECT_NXP:lines.nxp

*** number of grid  = 3
*** number of line  = 31
*** number of other = 10
```

```
press «return» to exit

        ********************************************
        *            FILE CONVERSION               *
        *   AUTOCAD DXF FILE TO NEXPERT NXP FILE   *
        *        FOR ELEVATION DRAWING             *
        ********************************************

Input DXF file :? s_elev.dxf
Orientation    :?
                1 for south
                2 for east
                3 for north
                4 for west        :1

*** input  DXF file         =: GOTO HD:DATA:S_ELEV_DXF:s_elev.dxf
*** output NXP file (grid ) =: GOTO HD:DATA:S_ELEV_NXP:table.nxp
*** output NXP file (line ) =: GOTO HD:DATA:S_ELEV_NXP:lines.nxp

*** number of grid  = 3
*** number of line  = 16
*** number of other = 3
|
```

Figure 4-5: **Interface Programs (DXF to NXP)**

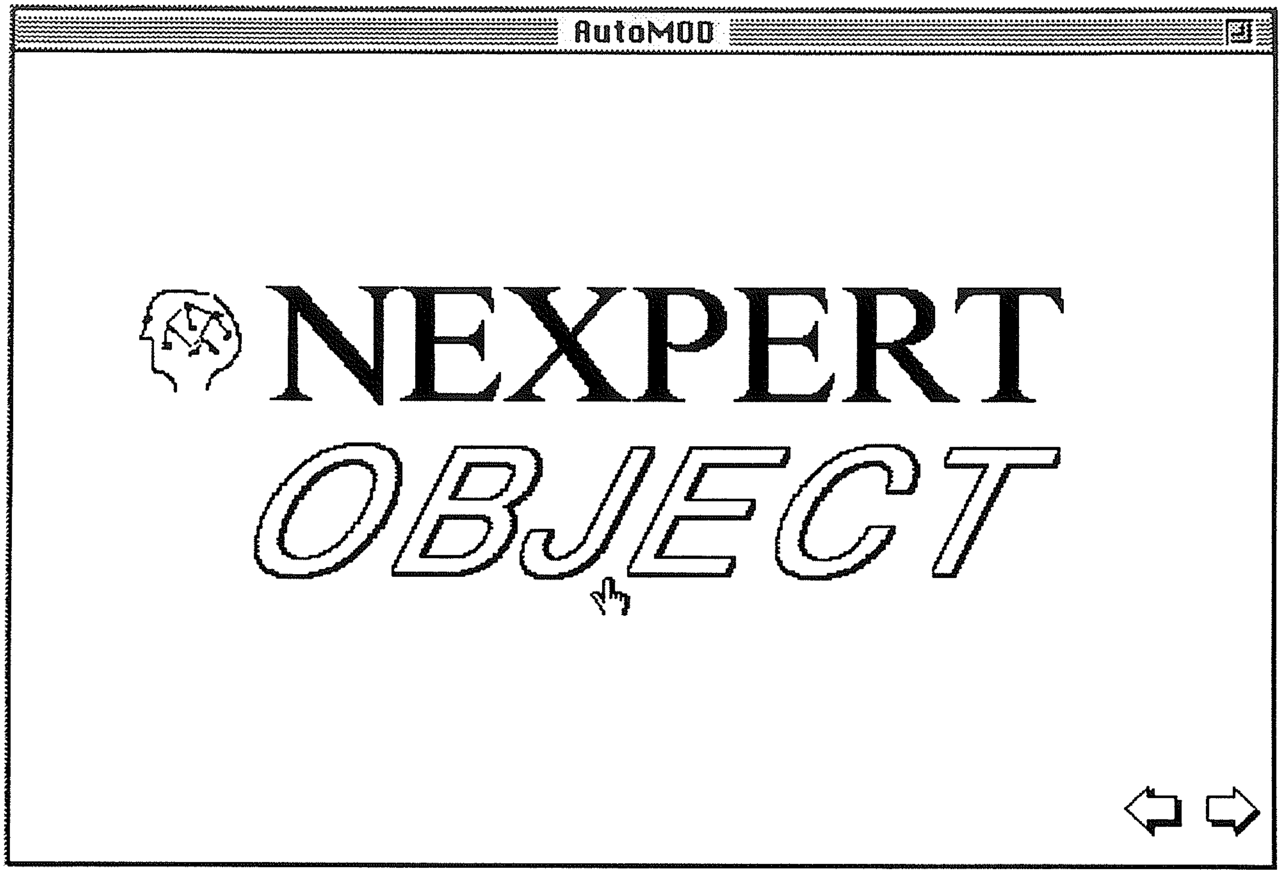


Figure 4-6: **HyperCard to Invoke Reasoning**

(3) Reasoning

Once the information is stored in Nexpert's readable NXP file format, the user can load all the rules and objects for interpretation by clicking the central portion of the screen shown in Figure 4-6. The reasoning process is carried out by AutoMOD as depicted in Figure 4-7. Examples of interpretation results are shown in Figures 4-8 and 4-9.

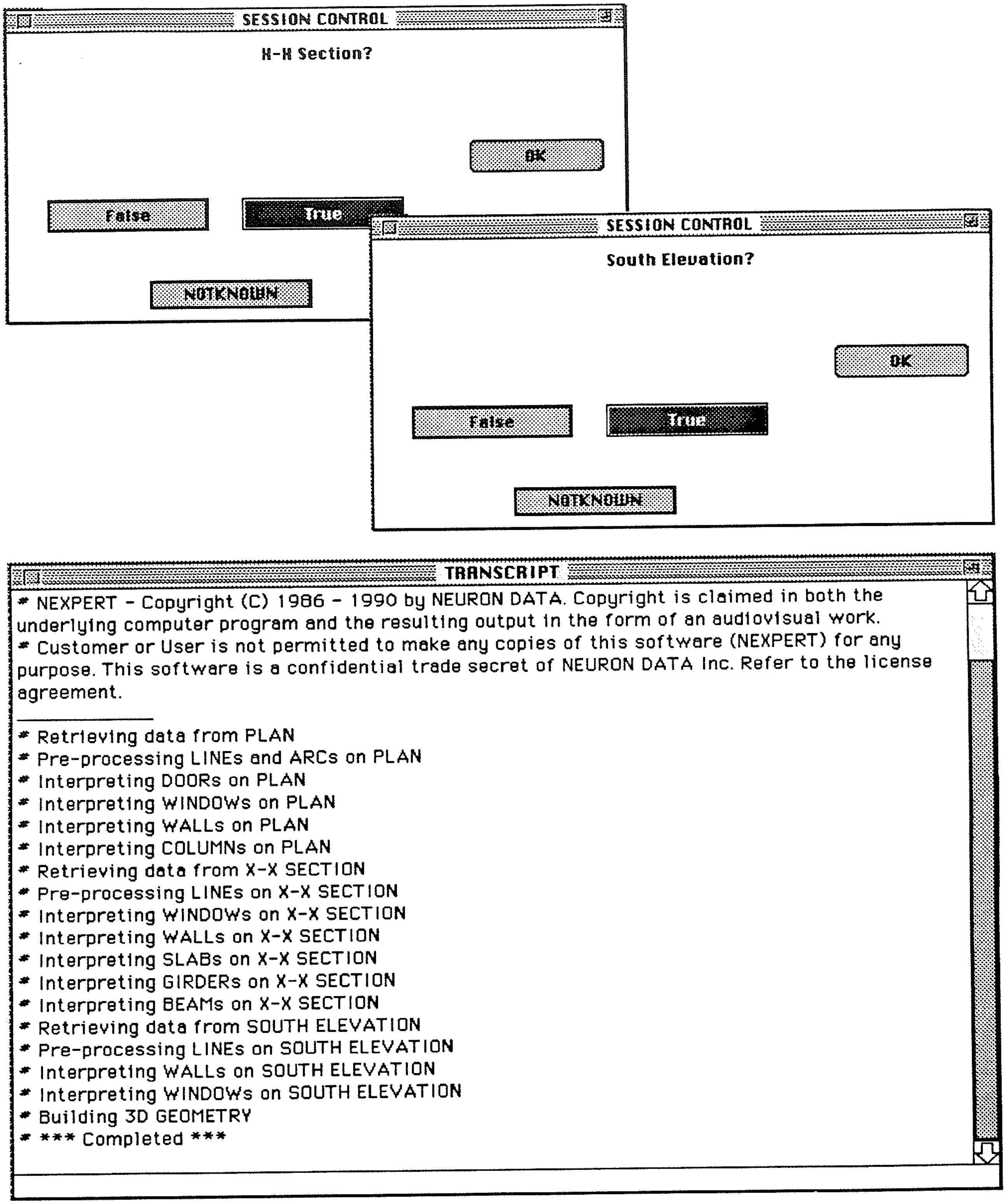


Figure 4-7: **Prompts of Reasoning**

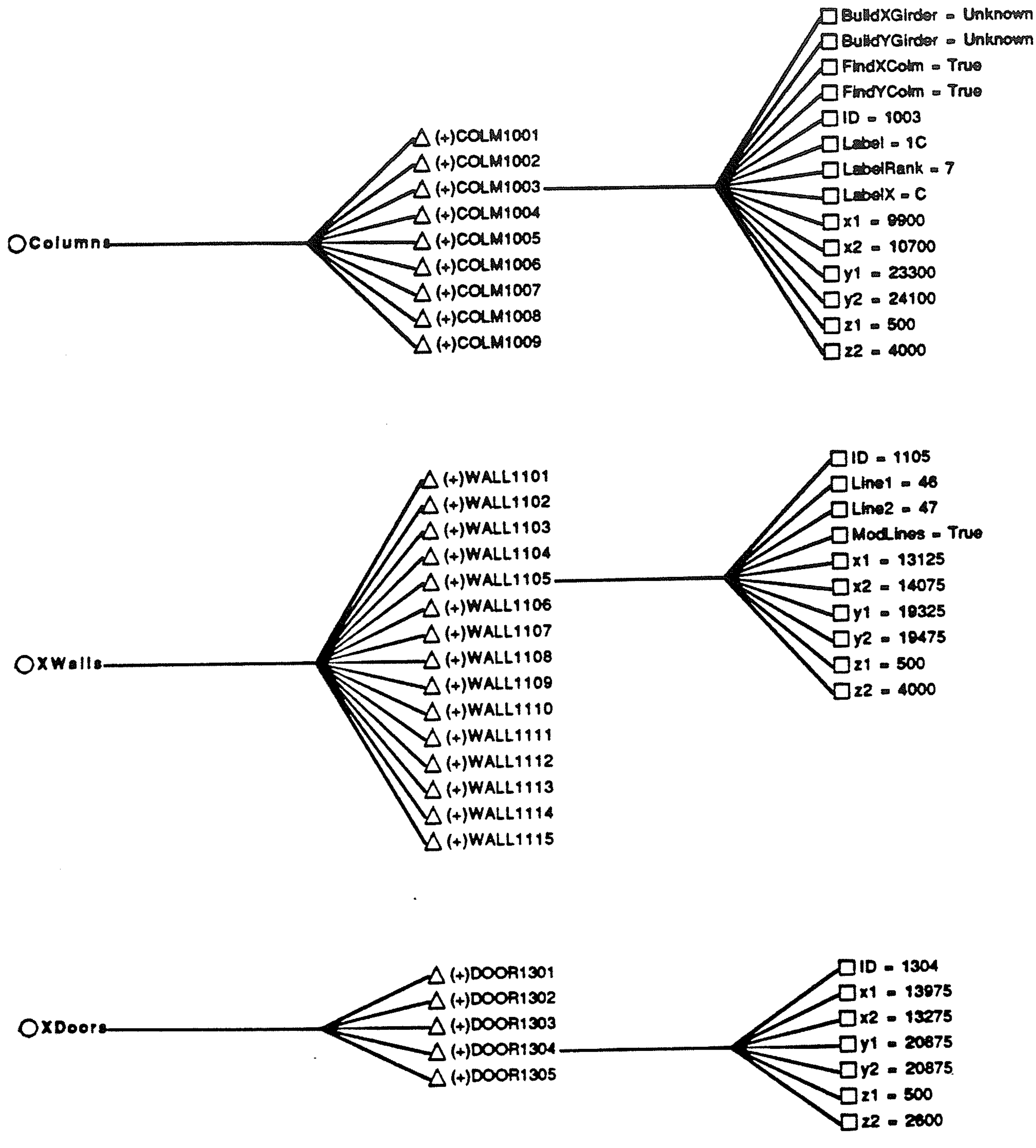


Figure 4-8: **Examples of 3-D Objects (Column, Wall and Doors)**

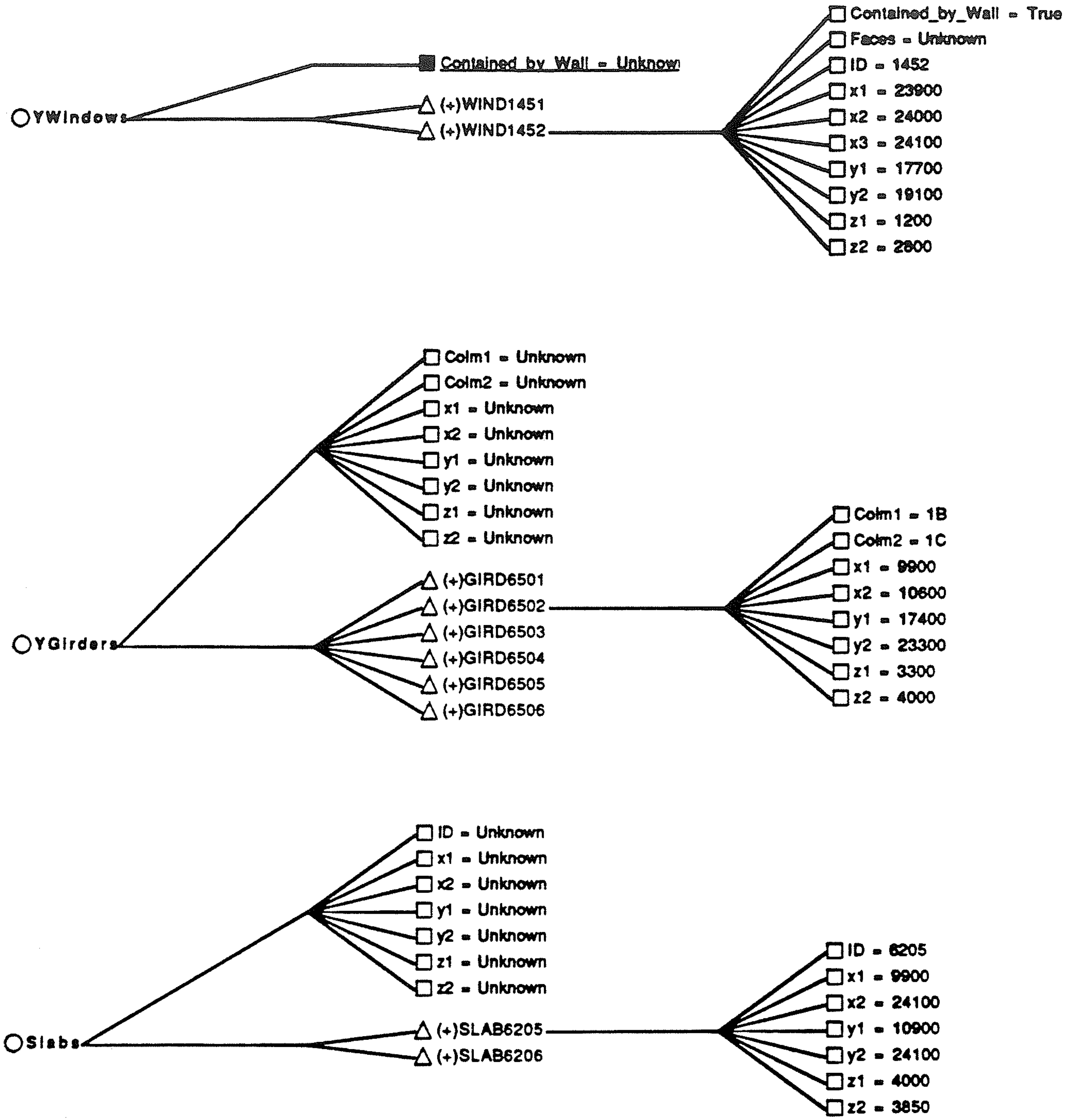


Figure 4-9: **Examples of 3-D Objects (Windows, Girders and Slabs)**

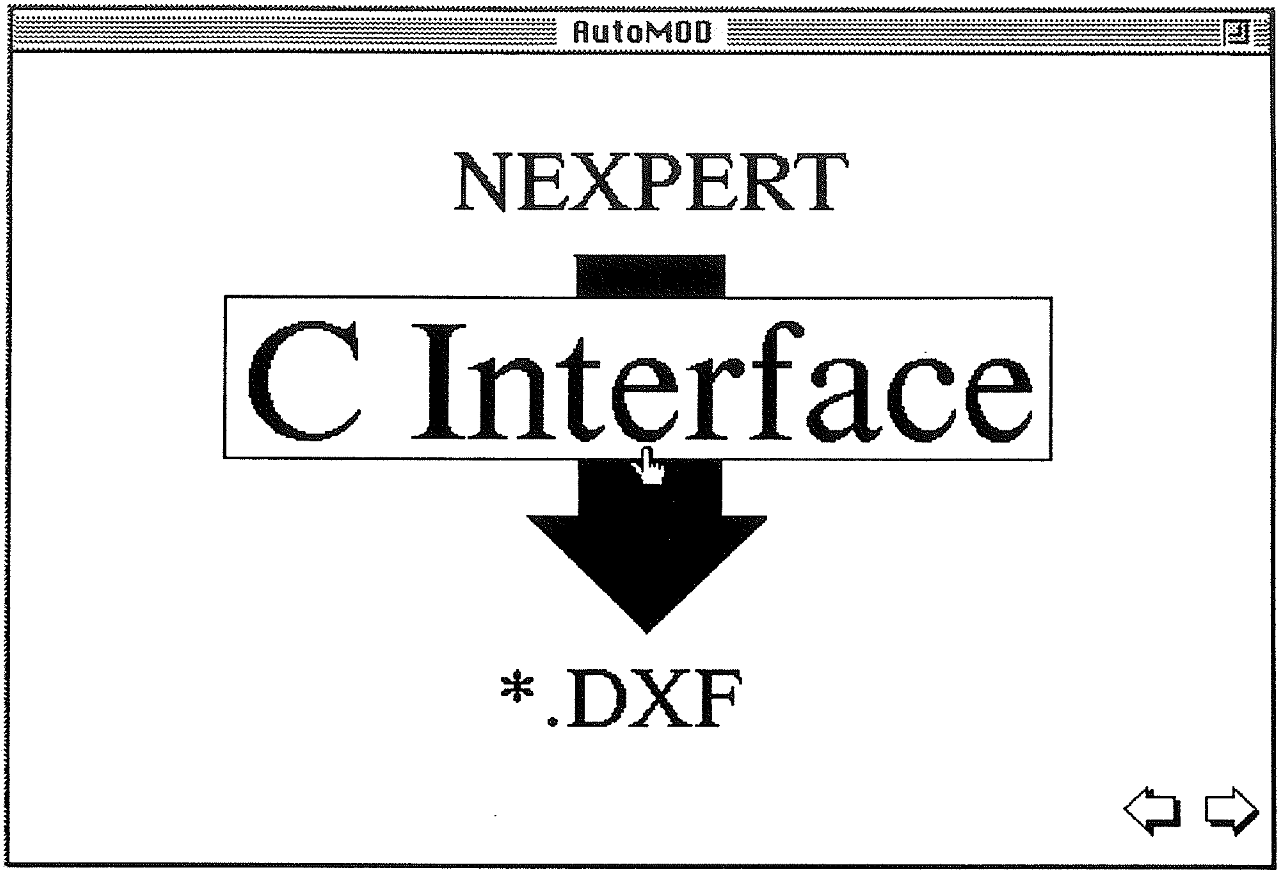


Figure 4-10: **HyperCard to Invoke Transforming 3-D Objects into a 3-D DXF File**

(4) Transforming 3-D Objects into a 3-D DXF File

After the 2-D drawings are interpreted and the object information about each building component is obtained, the user can run an interface program for transforming 3-D objects into a 3-D DXF file by clicking the central portion of the screen shown in Figure 4-10. As shown in Figure 4-11, for this example, the interface program retrieves the object information of 9 columns, 24 walls, 5 doors, 5 windows, 12 girders and 2 slabs and stores them in a 3-D DXF file.

```
press «return» to exit

        ************************************************
        *                                              *
        *               FILE CONVERSION                *
        *  NEXPERT 3D NXP FILE TO AUTOCAD 3D DXF FILE  *
        *                                              *
        ************************************************


DXF file to output :? model3d.dxf

*** reading objects from MODEL_NXP folder..
*** writing objects to   MODEL_DXF folder..

*** number of COLUMN  =: 9
*** number of WALL    =: 24
*** number of DOOR    =: 5
*** number of WINDOW  =: 5
*** number of GIRDER  =: 12
*** number of SLAB    =: 2
```

Figure 4-11: **Interface Program (3-D Objects to DXF)**

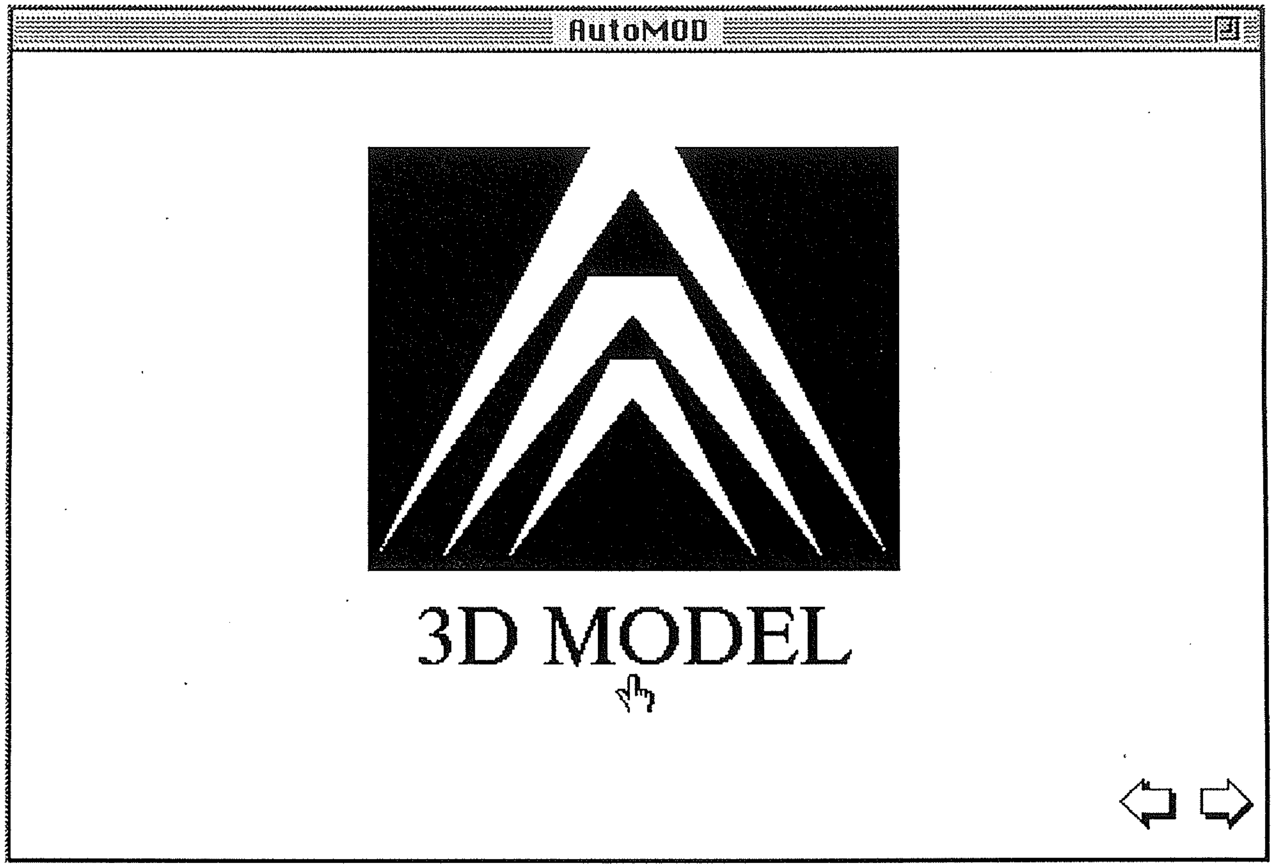


Figure 4-12: **HyperCard for Displaying a 3-D Model**

(5) Displaying a 3-D Model.

Finally, the user can display the resulting 3-D model via AutoCAD by clicking the central portion of the screen shown in Figure 4-12. The 3-D DXF file can be imported by AutoCAD's native command, *DXFIN*. The 3-D model is displayed as shown in Figure 4-13. Each building component is stored in a layer by its name (e.g., COLUMN1009, WALL1101, etc. See Figure 4-14).

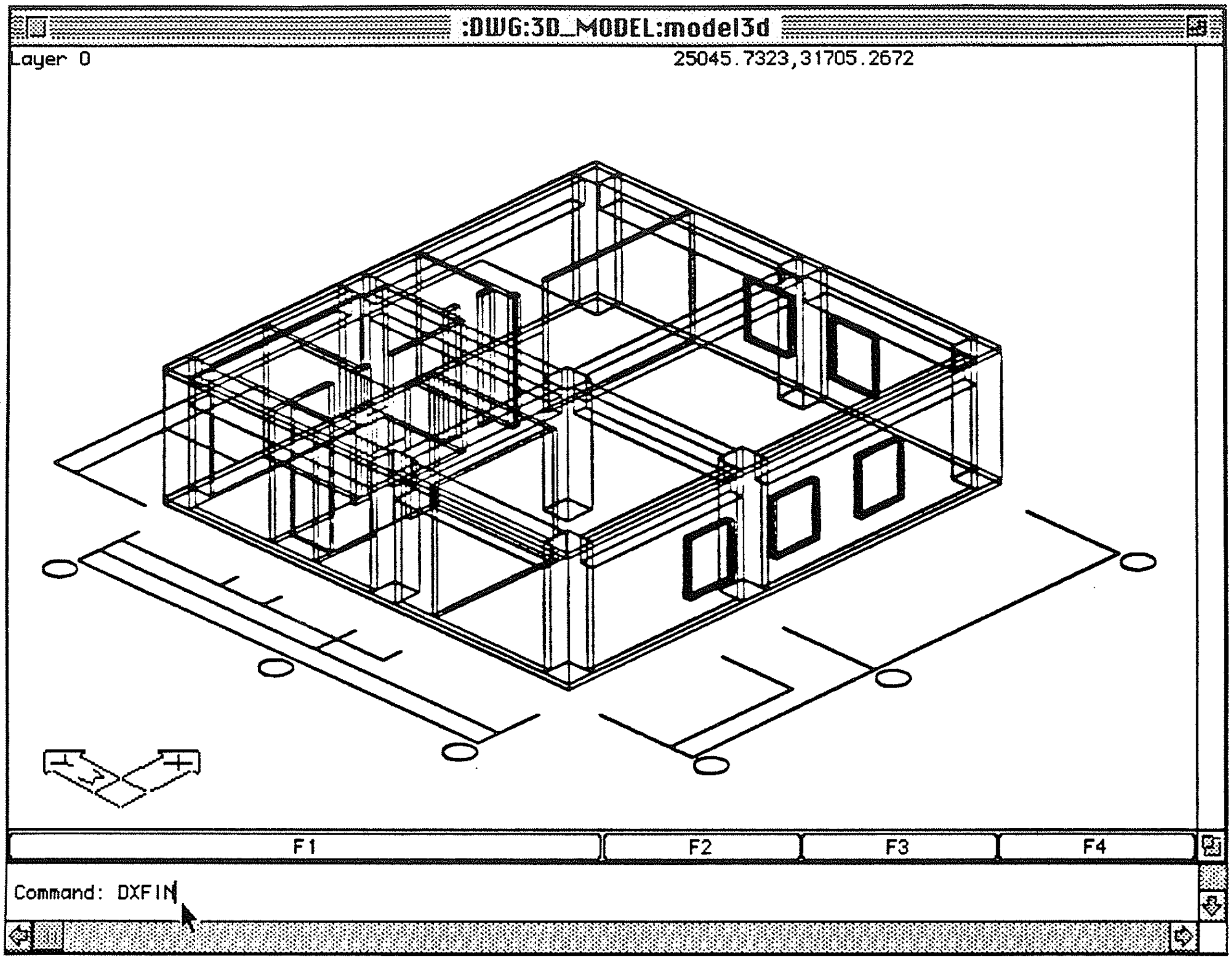


Figure 4-13: **3-D Model on AutoCAD**

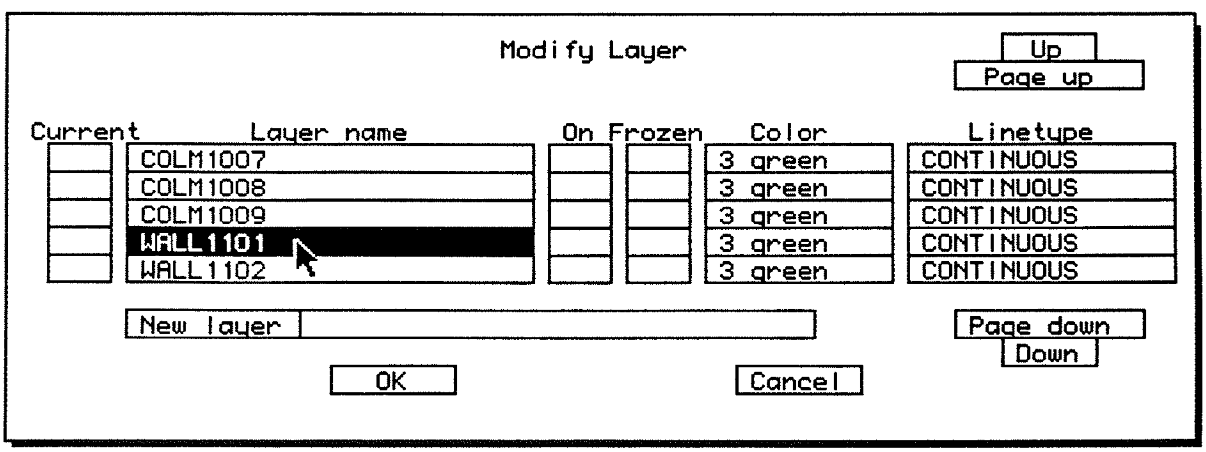


Figure 4-14: **Example of Layer of 3-D Model**

Section 5
Conclusions

This chapter presents the conclusions drawn from this seed research work. First, the advantages and limitations of the AutoMOD system are summarized. Then, directions for future work are discussed.

5.1 Advantages and Limitations of AutoMOD

The AutoMOD system has demonstrated successful interpretation of drawings with seven types of building components: columns, walls, girders, beams, slabs, doors and windows, drawn on multiple architectural drawings. The system is capable of dealing with four different drawing sets such as a plan only, a plan and a section, a plan and an elevation, and a plan, a section and an elevation.

Since the AutoMOD system takes DXF files as input, any drawing in DXF format derived from either a 2-D drafting system or a scanning system can be input to the system. However, the implementation is restricted to simplified rectilinear plan, section, and elevation drawings. Moreover, the drawings must follow some rules such as preserving accuracy and consistency, and classifying input information into appropriate layers of the DXF file before the interpretation.

The resulting 3-D model includes object information such as features, geometry and relationships. Any 3-D object system could use the results of AutoMOD for quantity take-off or design simulation. Presently, the results are converted back to a 3-D DXF file so that any CAD system can display the model.

HyperCard user interface controls a number of applications being used in the AutoMOD system so that the user can examine entire process from making input drawings to displaying the resulting model. However, the AutoMOD system is not capable of interacting with the user during the interpretation process.

5.2 Directions for Future Work

The subjects for future work can be classified into six categories based on the limitations of the current AutoMOD system.

(1) Complexity of input drawings

Expanding interpretation capability of the AutoMOD system to include:

- other building components
- multiple-story buildings and
- other combinations of drawing sets

would be next steps toward the practical use of the system.

(2) Flexible recognition model and user interface

In order to deal with complexities or ambiguities of input drawings, the three activities in the recognition process (i.e., Recognition of Features, Recognition of Relationships and Synthesis) should interact each other during the process instead of working as a sequential process. The user should be able to appropriately support this interaction through the graphic interface to directly access the object information.

(3) Addition of new components

A flexible means to add new building components for interpretation into the system, including automatic learning capability as a longer-term research issue. Individual project users will not add procedures for interpreting new components, but a goal is to allow non-programmers to add support for interpreting new building components.

(4) Extraction of attributes

To complete the object information that the current AutoMOD generates, attributes of objects such as material and cost information need to be extracted from both drawings and specifications.

(5) Integration with other systems

To prove the effectiveness and usefulness of the AutoMOD system, the integration with other 3-D object systems such as CIFE CAD[1] [10] [11] and OARPLAN[2] [16] would be useful where the resulting 3-D model from AutoMOD is used as input to those object systems.

(6) Dynamic interpretation

As mentioned earlier, interpreting building components while they are being drawn (dynamic interpretation) would be an interesting research subject. In addition to the knowledge modules in static interpretation, the drawing sequence (e.g., column lines are most likely the first elements to be drawn on a drawing) would also enable the system to interpret the building components effectively.

---

1 CIFE CAD is the system built on AutoCAD for creating architectural 3-D object model.
2 OARPLAN is the system for generating construction planning using a 3-D object model.

Bibliography

Listed below are references that are related to this study but not all of them are cited in the paper.

[1] AUTODESK Inc., "AutoCAD Reference Manual", 1998.

[2] Brachman, R. J. and Levesque, H. J., "Readings in Knowledge Representation", Morgan Kaufman Publishers, Inc., 1985.

[3] Bunke, H. and Sanfeliu, A., "Syntactic and Structural Pattern Recognition - Theory and Application", World Scientific Publishing Co. Pte. Ltd., 1990.

[4] Cherneff, J. M., "Knowledge Based Interpretation of Architectural Drawings", PhD Thesis, Civil Engineering Department, Massachusetts Institute of Technology, 1990.

[5] Chinowsky, P. S., "THE CAADIE PROJECT: Applying Knowledge-Based Paradigms to Architectural Layout Generation", PhD Thesis, Civil Engineering Department, Stanford University, 1991.

[6] Cohen, P.R. and Feigenbaum, E. A., "The handbook of Artificial Intelligence", William Kaufman, Inc., 1982.

[7] Dong, X., "Geometric Feature Extraction for Computer-Aided Process Planning", PhD Thesis, Rensselaer Polytechnic Institute, 1988.

[8] Endo, Y., Akimichi, S. and Milne, M., "The Context-Base Graphic Input System", Proceedings of IV-ICCCBE International Conference, Tokyo, Japan, 1991.

[9] Goodman, D., "The Complete HyperCard 2.0 Handbook", Bantam Books, 1990.

[10] Ito, K., Ueno, Y. and Levitt, R. E., "Linking Knowledge-Based Systems to CAD Design Data with an Object-Oriented Building Product Model", CIFE Technical Report No. 17, 1989.

[11] Ito, K., Law, K. H. and Levitt, R. E., "PMAPM: An Object Oriented Project Model for A/E/C Process with Multiple Views", CIFE Technical Report No. 34, 1990.

[12] Kato, Y., Ishii, T., Endo, Y., Morimoto, O., Tatsumi, Y. and Goto, H., "CIM Implementation in the Japanese Construction Industry", CIFE Affiliates Technical Report No. 1, 1990.

[13] Kunz, J. C., "Concurrent Knowledge Systems Engineering", CIFE Working Paper No. 5, 1989.

[14] Kunz, J. C., Stelzner, M. J. and Williams, M. D., "From Classic Expert Systems to Models: Introduction to a Methodology for Building Model-Based Systems", Topics in Expert Systems Design: Methodologies and Tools, pages 87-110, Elseviser Science Publishers B.V., 1989.

[15] Law, K. H. and Wentorf, R., "Natural Language Processing and Computer Graphics for Engineering Design", Proceedings of IV-Computing in Civil Engineering Conference, 1986.

[16] Levitt, R. E., Hayes-Roth, B. and Darwiche, A., "OARPLAN: Generating Project Plans by Reasoning about Objects, Actions and Resources", CIFE Technical Report No. 2, 1989.

[17] Nagasamy, V. and Langrana, N. A., "Recognition of Three-Dimensional Objects Using a Knowledge-Based Environment", Engineering with Computers, Vol. 7, pages 23-35, 1991.

[18] NEWRON DATA, "Nexpert Object Reference Manual", 1990.

[19] Sasada, T., "Open Design Environment", Proceedings of IV-ICCCBE International Conference, Tokyo, Japan, 1991.

[20] SYMANTEC, "THINK C User's Manual", 1990.

[21] Takenaka Corp., "Drawing Manual", 1985.

[22] Teicholz, P. M., "Technology Trends and Their Impact in The A/E/C Industry", CIFE Working Paper No. 2, 1989.

[23] Wilson, R. J. and Watkins, J. J., "Graphs - an Introductory Approach", John Wiley & Sons, Inc., 1990.

[24] Winblad, A. L., Edwards, S. D. and King, D. R., "Object-Oriented Software", Addison-Wesley Publishing Company, Inc., 1990.

[25] Wright, P. K. and Bourne, D. A., "Manufacturing Intelligence", Addison-Wesley Publishing Company, Inc., 1988.