1988

Descriptive models of design projects

Sarosh Talukdar
Carnegie Mellon University

Jim Rehg

Alberto Elfes

Carnegie Mellon University.Engineering Design Research Center.

Descriptive Models of Design Projects

by

Sarosh Talukdar, Jim Rehg, Alberto Elfes

EDRC 05-23-88

DESCRIPTIVE MODELS OF DESIGN PROJECTS

Sarosh Talukdar, Jim Rehg, Alberto Eifes

**Engineering Design Research Center, Carnegie Mellon University,
Pittsburgh, PA 15213**

ABSTRACT Engineering design efforts are usually organized into projects. Traditional CAD tools leave large gaps in their coverage of these projects and their coordination. A necessary first step in closing these gaps is the development of descriptive models of projects. This paper suggests the use of directed networks for such models. Nodes in these networks are data structures customized for storing descriptions of aspects of the artifact being designed; arcs are design processes. The networks have several advantages including the capabilities for arbitrarily detailed description of a project's micro-structure, arbitrarily great expansion and the seamless integration of new processes and aspects. The status and plans of an effort to build a network model are included in the paper as an illustration.

INTRODUCTION

Many of the ideas described in this paper were developed through a study of processes used in the Fisher Guide division of GM for the design of window regulators. A window regulator is a device that raises and lowers the glass in an automobile door. We will allude to window regulators whenever we need to illustrate concepts in the succeeding material. This is done for the purposes of continuity and coherence. The concepts themselves are general, and apply to more than window regulators.

Projects

Design efforts can be examined at granularities ranging from the moment-by-moment thoughts and actions of a single designer to the collective efforts of many designers over many years. In this paper we will examine efforts of a fairly coarse grain we call projects. Several such projects are undertaken in the life cycle of the typical artifact Each project covers a coherent let of the artifacts features, involves a team of people and usually lasts for a period of weeks to months. With the window regulator, for instance, one project encompass the design of its shape, and another, the design of the process by which it is to be manufactured.

Existing CAD tools address only a few isolated tasks that arise in the typical design project and virtually none of the tasks that arise in coordinating projects. In essence, these tools

provide islands of automation in a sea of human activity. Before the level of automation can be significantly increased, descriptive and computational models of entire projects will have to be developed. The succeeding material discusses such models, beginning with an explanation of the terminology we will use.

Aspects

By "aspect" we mean the appearance of an artifact to any of the many actors who play some port in the artifact's life cycle. These actors can either be human-structural designers and production engineers, for instance-or programmed-finite dement analysis and solid modeling packages, for instance. To the structural designer the artifact first appears as a set of behavioral specifications and later, as a set of blueprints that meet these specifications. To the production engineer, the artifact first appears as a manufacturing problem and later as a process to solve this problem. To a finite element analysis program the artifact appears as a finite element mesh, and so on. In fact, there is no limit to the number of aspects that an srtifact can have and each person or computer tool usually sees at least two, corresponding to the inputs and outputs of her/its part in the artifact's life cycle.

Design-results (products)

By "design-result" we mean the information that instantiates (fleshes out or describes) an aspect For instance, the design-result of the above mentioned production engineer is the information that describes the particular manufacturing process he has devised from all his knowledge of manufacturing processes.

Design-activities (processes)

Neither practitioners nor researchers seem to be able to agree on what constitutes design activity. For instance, Susan Finger, while director of the Design Theory and Methodology program in the National Science Foundation, asked researchers for their definitions and collected the following answers:

- design is satisfying constraints and meeting objectives
- design is problem solving
- design is decision making
- design is reasoning under uncertainty
- design is search
- design *is* planning
- design is an iterative process
- design is a parallel process
- design is an evolutionary process
- design is a mapping from functional space to physical space
- design is like a game
- design is creative and inexplicable

We prefer a broad definition that includes all the intellectual activities that lead to design-results. These activities form a continuum with uninspired copying at one end and invention at another.

Routine and Nonroutine Activities

The typical project employs a mix of intellectual activities from the continuum mentioned above. This mix can be thought to consist of two fuzzily separated categories: routine and nonroutine activities. By "routine activity" we mean efforts that follow established, well defined design methods; for instance, the use of formulas from handbooks. By "nonroutine activities" we mean everything else, but especially the design of new design methods and the introduction of new technologies.

Simultaneous Engineering
Simultaneous engineering (sometimes called concurrent design), refers to the process of coordinating design projects. When the projects run concurrently, the objective is to prevent incompatibilities in their results like a window regulator that will not fit in its door. When the projects are displaced in time, the objective is to keep upstream projects from making choices that would unduly increase the difficulty of downstream projects; for instance, a window regulator shape that is particularly difficult to manufacture.

Summary and Objectives

We view design as the process of assembling information about any and every aspect of an artifact. These aspects cover the life cycle of the artifact from its creation, through its use, to its disposal and after effects. Usually, design activity continues throughout the life cycle but at levels that are highest in the early stages. Organizationally, the activity can be decomposed into a set of projects some of which may run concurrently. The inputs and goals of each project tend to change dynamically. Within each project, a mix of intellectual activities is used to achieve its goals. This mix can be divided into two categories: routine and nonroutine activities. The former involves following established paths to the project's goals, the latter involves finding new paths. The task of coordinating projects, particularly to ensure that in achieving the goals of one project the goals of another are not made unreachable, is called simultaneous engineering.

The objective of this paper is to develop an architecture for models that capture routine and nonroutine activities of projects, and the notion of simultaneous engineering.

NETWORK MODELS

Representations

Anyone involved in engineering or scientific problem solving is aware of the importance of representations. A good problem representation can make the solution obvious; a poor representation can make it impossible to find.

To be good, a representation must contain neither too much information nor too little, and must express the information in terms that are readily comprehensible to the problem solver (1, 2). In other words, to be good, a representation must be tailored to the needs of both the problem and the problem solver.

The implications for design are clear. The representation for every aspect must be customized for the problem solver (human or computerized) that is to work on it. By way of an illustration, consider a window regulator (recall that a window regulator raises and lowers the glass in an automobile door). The type of regulator we have in mind has three major components-a "lift arm" to move the glass up and down, a "back plate[1]* to provide a pivot for the lift arm on the inside of the door, and a toothed "sector" that connects the lift arm through a pinion to a handle that drives the mechanism.

A manual design sequence begins with a set of specifications (as in Fig.l) and moves progressively through more detailed representations (as in Figs. 2 and 3), ending in the most detailed representation-! set of blue prints or a solid model (as in Fig.4). Each of these representations captures a different aspect of the window regulator.

The overall goal of this design sequence is to obtain the solid modeL The purpose of the intermediate aspects (Figs. 2 and 3) is to serve as stepping stones or subgoals on the way to the overall goal. To fulfill this purpose, the intermediate aspects must have two properties. Fust, they must be placed to decompose the overall problem into subproblems of manageable complexity. Second they must use representation schemes that are meaningful to the associated problem solvers. The pictorial schemes of Figs. 2 and 3 are well suited to human problem solvers. However, other types of problem solvers might demand quite different schemes. As an illustration, consider the problem of optimizing the tolerances of a window regulator for manufacturing and maintenance costs. The geometric information needed to formulate this problem is contained in the solid model. However, the most appropriate problem solver is a nonlinear optimization program that does not understand solid models but requires information to be supplied in the form of algebraic relations, like those in Fig. 5. Therefore, if tolerances are to be optimized, provisions must be made to include both solid model and algebraic representation schemes. Other window regulator problems demand still other representations like block diagrams, graphs and differential equations.

In summary, goals and subgoals correspond to aspects. The representations of aspects must be customized for the problem solvers that are to work on them. As a result, even relatively small projects can require a considerable number of different representation schemes. At present, very few of the translations among schemes are automated.

Design Projects, Products and Processes Revisited

In light of the previous discussion, a design project can be seen as an effort to identify a set of goals and related subgoals (say N in total), associate an aspect with each goal and subgoal, select a representation scheme for each aspect, and deploy problem solvers to instantiate the resulting representation schemes (i.e. to produce actual representations of the selected aspects).

Let A(n,m) be the set of aD the possible instances of the n-th aspect expressed in the m-th representation scheme, and let a(n^njc), k«l,2,..JC, be the k-th instance (element) of A(n,m). The product of a design project is the N-tuple:

$$D=\{a(1,r(1),k(1)),\ldots\ldots,a(N,r(N),k(K))\}$$

where *r(n)* is the representation scheme and k(n) is the element identifier for the n-th aspect For example, the product of a project to design the solid model of a lift arm, given its specifications, is the 4-tuple whose elements are Figs. 1-4.

Clearly, the N elements of D must be chosen to be consistent with one another. For example, the solid model of Fig. 4 must have geometric properties and mechanical strengths sufficient to satisfy the specifications. The geometric properties are easy to test The mechanical strengths, however, are more difficult First "sufficient strength" needs to be quantified and then analytical or empirical procedures devised to test for it Note that the results will be situation-specific; what seems sufficient to one company may seem quite inadequate to another.

In our experience, much of the effort expended in a project goes into developing computational standards of consistency and devising ways to generate the elements of the N-tuple, D, so they meet these standards.

The usual approach to enforcing consistency standards is to treat one aspect (i.e. one element of D) as a "given" and generate the other aspects progressively from it Often, the starting aspect is the least detailed and most abstract, as is the case in generating Figs. 2, 3, and 4 from 1. However, this does not always have to be so. One could start with the most detailed aspect (reverse engineering) or one of intermediate detail.

The generation of aspects, whether in forward, reverse or intermediate modes, is done by design processes. Thus a process can be thought as an operator or directed link between two aspects. The function of the link is to wait till the aspect at its input end has been instantiated and then generate a consistent instantiatiation of the aspect at its output end. If a process in a forward path (i.e. a path leading away from the starting aspect) is not powerful enough to guarantee the consistency of its output, reverse processes can be added to create "generate and test" loops that improve consistency through interaction.

Networks

It follows from the discussions on aspects and processes that the computational activities of a design project can be modeled by a directed graph or network whose arcs are processes and whose nodes are data structures. Each of these structures is used to store an aspect of the artifact being designed. The processors that execute the processes can be either human or mechanical, and most projects use some of each.

A path between a pair of nodes defines a sequence of processes by which the end node can be filled (instantiated), given the contents of the starting node. The presence of multiple paths between a pair of nodes indicates the existence of multiple design sequences, each of which could produce a quite different result Loops correspond to consistency verification cycles, such as: specifications->bliieprint->pro^ype->specifications.

Routine project activities correspond to the operation of existing networks, that is, to the use of known design processes to fill established data structures.

Nbnroutine activity corresponds to network building, that is, to modifying or adding nodes and arcs. More specifically, nonroutine activity consists of: -identifying new aspects; -designing new data objects for aspects; -developing new processes.

Simultaneous engineering-the development of schemes to coordinate distinct design projects-corresponds to building ties among the projects' network models. When the actual projects are to be displaced in time, implementing such ties will require simulating the activities of the later projects while the earlier ones are in progress. The computational

Class in fuD up position, x coordinate • 250 mm

Class in full up position, *y* coordinate • 450 mm

Class in full up position, t coordinate • 310 mm

Glass in fuD down position, x coordinate • 900 mm

Class in fuO down position, *y* coordinate • 50 mm

Glass in full down position, x coordinate • 270 mm

Glass weight-151b

Handle location, x coordinate • 0 mm

Handle location, *y* coordinate • 250 mm

Handle location, *z* coordinate • 290 mm

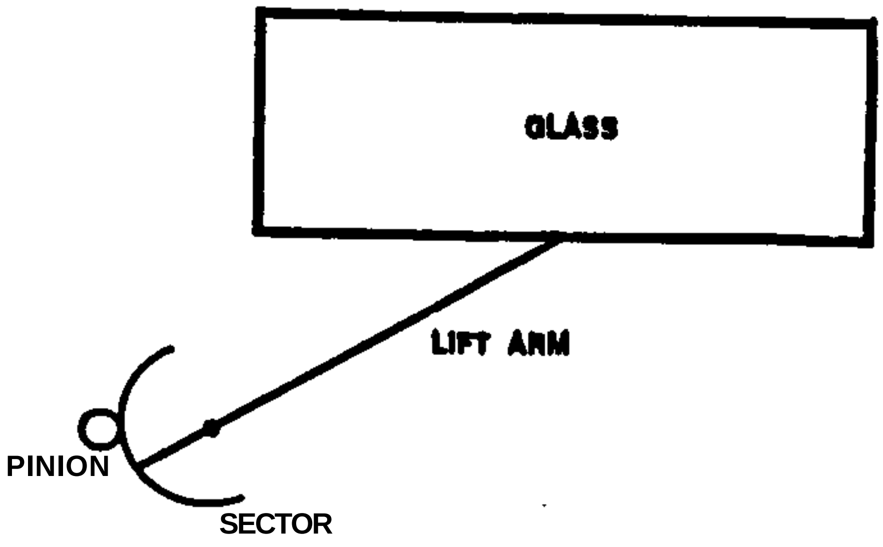
Location of center of gravity with respect to edge of gtass • 126 mm

Maximum allowed number of handle turns » 4.5

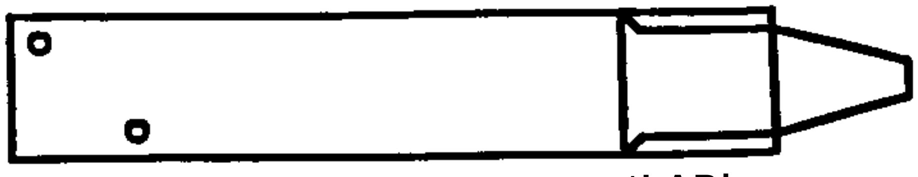
Maximum allowed handle effort • 2.0 N-m

Minimum force requirement for spindle abuse test • 18.5 N-m

Fig 1. Typical specifications for **s window** regulator·



Hg · **2. A stick tfaonm or skeleton of s window regulator. This dtogram dtepiays the prtndpsJ ** and Oear dimensions while suprtttJng al other gsometric dstafl.**



rig 3. A fctluradbgramoltw Warm of •«**»» regulator. 1Nt«aor«t**Bfeyf tht main features of lh« •mi ki oft«tr otonuMe fetal than It tvalat* from t» tfcfc dtogram. fewtvar.twMaitoml complete. tapMtady h tw *«wnaton *pmptnttoAm* to »• plant ol ta dtaoram

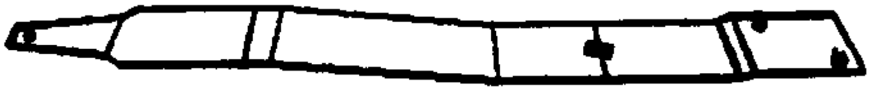


Fig *m* *m* rendering of a solid a^ik^fc^Al —* ^ Aft ^i^i^. The solid model provides a complete geometric description of the arm

$$z_4 = z_1 + r_1(1,0)$$

$$z_5 = z_4 - r_2(\cos\gamma, \sin\gamma)$$

$$p(\theta) = z_5 + r_2(\cos(\theta-\gamma), \sin(\theta-\gamma))$$

$$d(z_1, p(\theta)) = \{(r_1 + r_2(\cos(\theta-\gamma) - \cos\gamma))^2 + (r_2(\sin(\theta-\gamma) - \sin\gamma))^2\}^{1/2}$$

$$gap = d(z_1, z_2) - d(z_1, p(\theta)) - r_2$$

Hf.S. Some of the equations that relate the gap between the sector and pinion teeth to tolerances talhe principal dimensionsl of te window itgulttor. These equations •t Inputs to t pregnro *lot* oprtialTlnf the tolerances for manufacturing costs.

intensity of these simulations could be reduced by replacing full network models with simpler equivalents, as is done in studying interconnections of electrical networks. Of course, the technology for simplifying design project networks would have to be developed first

Comments

The above described networks have two attractive features. First, they seem to capture the actual representations, processes and architectures of design projects. Second, they are arbitrarily expandable. Among the advantages that result are:

- **the micro-structures of the routine parts of projects can be described in arbitrarily great detail. (Arcs can be expanded into subnetworks.)**
- **the history of a project can be recorded by taking snapshots of the network.**
- **new processes and aspects can be added without disturbing the existing network.**

The distinguishing characteristic of the network is its set of distributed data structures. Each structure can be designed independently and tailored to the needs of the aspect it is to contain. Moreover, at any time the structures are allowed to store aspects that are inconsistent with one another, as often happens in the early stages of a project It is only towards the end of most projects that glaring inconsistencies are eliminated through iteration and reverse processes. Models that would force all the aspects into a centralized database with a single representation scheme and no provisions to accommodate inconsistencies cannot capture the actual behaviours of projects.

CASE: A NETWORK DEVELOPMENT PROJECT

To study the implications of using network models for design projects, a team of people from Fisher Guide and Carnegie Mellon has set out to build a network, called CASE, for the design of window regulator geometries (WRG). Window regulators were selected for two reasons. First, they are simple enough to serve as research specimens simple to be trivial nor too complicated to be overwhelming. Second, they are typical of a large class of multi-part objects that are common in automobiles and other electro-mechanical systems, and should lead to generalizable results.

A fairly detailed description of CASE is given in a companion paper (4). Here we will provide a much briefer description from a different viewpoint Our purpose is to illustrate the ideas presented in the proceeding sections.

Objectives

The objectives in building CASE are:

- **to shorten the time and improve the quality of design processes for WRG and related projects;**
- **to provide a lest bed for investigating the automation of design projects in general. Some important issues are: What tasks can and should be automated? What tools should be provided to facilitate this automation? How should human-computer interactions occur? How can simultaneous engineering be achieved? How should the computations be carried out?**

Status

In its present state (see Fig. 6). CASE contains four main computational subnetworks, the first is a completely automatic path (nodes l->5->6->7->8->4) that generates a solid model from any set of specifications for a window regulator. The second is also an automatic path l->5->6->7->8->4) that generates a solid model from any set of specifications for a window regulator. The second is also an automatic path (nodes 7->10->ll->9) that calculates and displays stresses in the regulator. The third is a manual loop (nodes 8->13->8) that optimizes tolerances for manufacturing and maintenance costs. The fourth (nodes 8->12->9 with a tie (the door is designed in a concurrent project).

The human-computer interface contains several nodes that have been described earlier and one node-the bulletin board-that has not This bulletin board is for messages that summarize important results and draw the designer's attention to consequences that might be beyond his ability to forsee. For instance, a message may point out that a recently made change in the door-project (which runs concurrently with the window regulator project) will cause an interference between the door and window regulator. We see such bulletin boards, backed up by a comprehensive set of expert systems for the generation of bulletins, as a key ingredient in automating simultaneous engineering.

Synthesis Processes

Arcs (design processes) can be classified by a number of attributes, including their directions (forward, lateral or reverse, with respect to the starting aspect), their effects on information content (expansion, translation or reduction of output relative to input), and the computational methods they employ (which include synthesis, optimization and analysis).

We will devote a few words here to the automatic synthesis processes in CASE. Our reasons are that synthesis processes largely determine the quality of a project's results, and also, automatic synthesis processes are relatively rare while automatic optimization and analysis processes are commonplace.

Synthesis processes work by generating alternatives and then selecting from among them. Three schemes by which these alternatives can be generated are: Each automatic synthesis process in the current version of CASE is identified by an S in Fig. 6. It has two functions. First, to identify alternatives for its output-aspect, and second, to select one of these alternatives that is consistent with the given input aspect

Since the number of different alternatives is usually large and often infinite, it is not possible to store all of them. Three more compact ways for characterizing alternatives are:

- sample sets, dun is sees of sample-alternatives, usually obtained from previous design efforts. New alternatives are generated by interpolating or otherwise modifying these samples.
- aspect kits, that is, sets of primitives (also called features) and sets of laws (constraints, rules and procedures). Alternatives are generated by combining the primitives in accordance with die laws. Figs. 8 and 9 illustrate some primitives and constraints for generating lift aims of die sort shown in Fig. 3.
- templates (also called generic parts and variational geometries), that fix the overall pattern of the alternatives but allow certain parameters to vary within this pattern (see, for instance, reference 3).

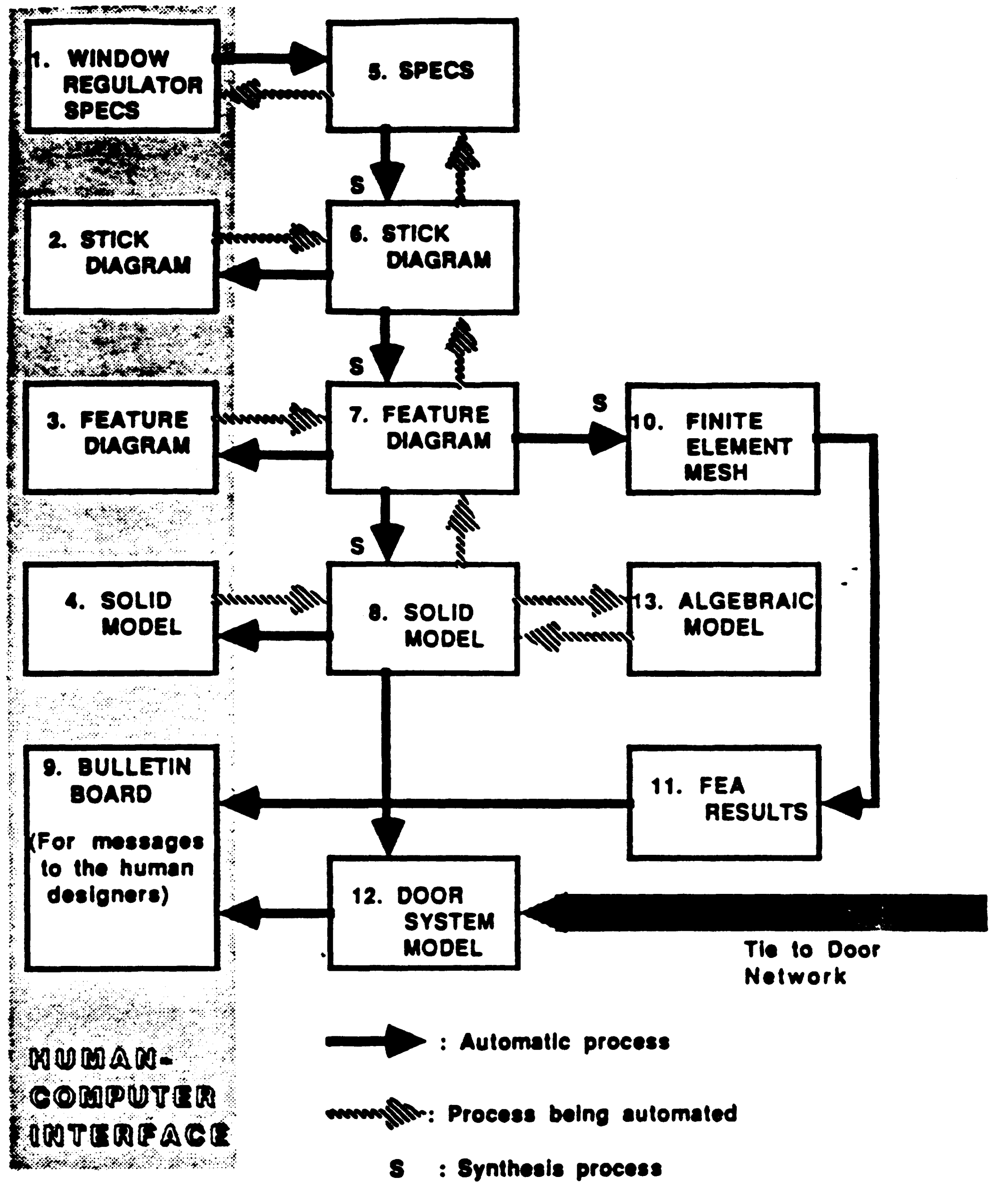


Fig. 6. The current state of CASE, a network for the design of window regulator geometries. Nodes in this network denote data structures for window regulator aspects; arcs denote design processes. Nodes 1-4 are data structures for human-oriented-representations of the specifications, stick diagram, feature diagram and solid model (Figs. 1-4); nodes 5-8 are for computer-oriented-representations of the same information. Node 9, the "bulletin board", is for displaying messages for the human designers from other parts of the network and other networks (other projects). Node 10 is for finite element meshes (see Fig. 7). Node 11 is for the results of the Finish Element Analysis. Node 12 is for a solid model from which any interferences between the door and window regulator can be calculated. Node 13 is for certain algebraic representations (see Fig. 5) that are needed to optimize the window regulator's tolerances.

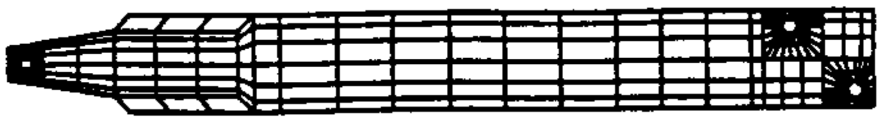


Fig. 7. A finite element mesh for a lift arm. This mesh is generated from the feature diagram (Fig. 3) and provides the input to a commercial finite element analysis program for calculating stresses.

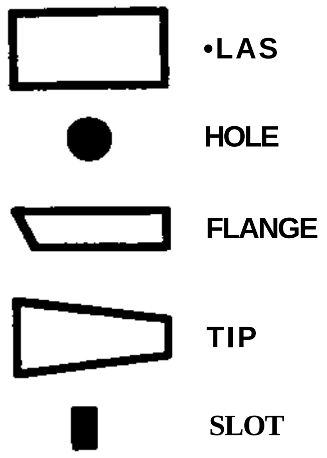
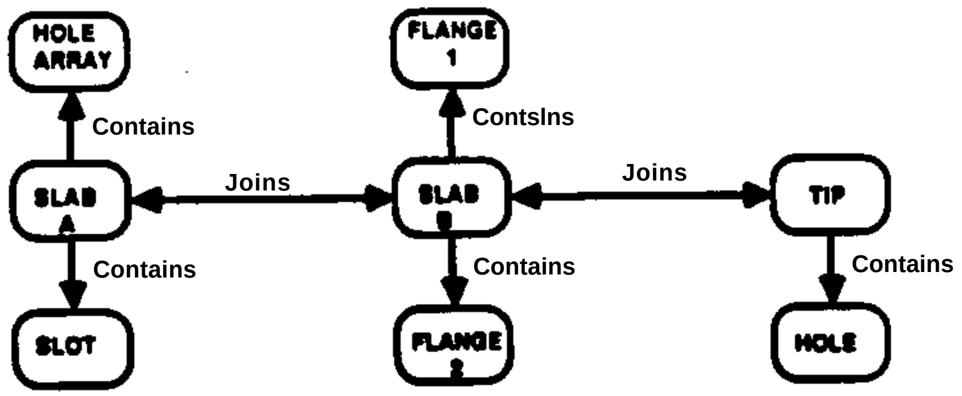


Fig. I. One tet of primitives from which feature diagmt (Fig. J) on be composed. The dimensions of the primidvts «c variable.



Hf.t. *Awmtmt* expressassi of topological constraints for synthesizing lift arms of thi aaR BBVB B Fig. 3 from the primitives afteaan shown in Fig. 6.

CASE uses all three schemes but in skeletal forms that allow it to produce acceptable designs though of limited range. The schemes will have to be packed with much more knowledge before CASE can produce the range and variety of designs of which an expert human is capable.

Plans

Our immediate plans are to add:

1. multiple entry points to the network;
2. automatic simultaneous engineering; and
3. distributed processing.

Some of the issues involved are mentioned below.

Multiple Entry Points and Boundry Value Problems

By "entry point" we mean an aspect that is at the root of a completely automatic path in the network. The current version of CASE has only one entry point-the window regulator specifications (node 1 in Fig. 6) and the feature diagram (Node 7 in Fig. 6). We intend to automate all the processes to the right of the "human-computer interface" in Fig. 6, and thereby, make it possible for a human designer to make changes at any of the four levels of detail contained in nodes 1-4 and have these changes propogate to all the other nodes. The human designer could also initiate reverse engineering efforts by providing the solid-model-description of a regulator at node 4, where upon the network would generate the specifications of the regulator as well as all its other aspects.

By a "boundry value problem" we mean one in which several aspects are given in part, and the network is required to complete these aspects and also, calculate the remaining aspects. For instance, one might be given all of the specifications (node 1 of Fig. 6) and some of the solid model (node 4 of fig. 6) and required to calculate the rest of the solid model and other nodes. Developing methods for solving such problems will be difficult but the payoff is high because many of the design changes that are invariably required in the latter stages of a project can be formulated as boundry value problems.

Simultaneous Engineering

To investigate simultaneous engineering we intend to consider two additional projects: (1) the design of door geometry, which usually runs concurrently with WRG, and (2) the design of manufacturing processes, which usually runs downstream from WRG.

With concurrent projects, the issue is where to make interconnections so that the projects are most effectively prevented from producing incompatible results. If CASE and the door project are only connected at the solid model level, as indicated in Fig. 6, then inconsistencies between the window regulator and door will not be detected till both solid models have been obtained. Interconnections at higher levels of abstraction would permit the detection of inconsistencies at earlier stages.

With serial projects, the issue is how to develop simple equivalents for the downstream projects. The purpose is to make the concerns of the downstream projects apparent to the upstream projects without having to simulate the entire networks of the downstream projects.

Distributed Processing

Some of the important features of CASE and systems like it are:

- projects are represented by continually expanding networks;
- each network consists of subnetworks that can be operated in parallel;
- subnetworks can have very different computational needs (i.e. can require very different types of processors);
- simultaneous engineering requires interconnections among networks. Some of the associated research issues are: What strategy should be used? How should the data structures for aspects be designed? What languages and utilities should be provided to help set up the distributed processing?

Our initial ventures into distributed processing will be made with the aid of a package called DPSK (Distributed Problem Solving Kernel) for combining programs written in different languages and running on different machines (5).

SUMMARY AND CONCLUSIONS

The life cycles of artifacts typically contain several design projects. Traditional CAD tools leave large gaps in their coverage of these projects and especially, their coordination (simultaneous engineering). A necessary first step in closing these gaps is the development of descriptive and computational models for entire projects. This paper argues for the use of directed networks as the basis for such models. The nodes in these networks are data structures and the arcs are processors for filling (instantiating) the nodes. Each data structure is configured to hold a description of an aspect of the artifact being designed. If the project is to work well, it is important that the terms of description (the representation scheme) selected for each aspect be customized for the processes that are to use it The processors that do the processing can be humans or computers and most projects require both.

The functional goal of a design project is to instantiate cenain key aspects so they are consistent Consistency is a subjective, situation-specific attribute, and much of the effort involved in a project can be devoted to reaching agreement on ways to define and measure it

We have split the total effort involved in a project into two categories-routing and nonroutine. The former corresponds to operating an existing network, that is, to using established processes to fill or modify the contents of established data structures. The latter corresponds to network building, that is, to adding and modifying data structures and processes.

We envisage these uses for network models of the type described here:

1. as aids in analysing projects. Since the networks capture the routine activities of design projects at levels of detail that can be arbitrarily increased, they should serve as good tools for assessing projects (identifying their strengths nd weaknesses) and for recording their histories.
2. as vchitectures for project automation and expansion. The network models allow the seamless integration of new processes and aspects. Also, they allow parallel processes so new processes can be gradually introduced and tested against existing processes (even manual ones).

3. as architectures for automating simultaneous engineering. Coordinating projects is equivalent to building interties among their network representations. The technology for doing this should have much in common with the technology for automating individual projects.

We have begun to use networks as tools for analyzing projects in several engineering domains and, as demonstrated by the description of CASE, have made some progress in using them for project automation and simultaneous engineering. Our plans are to continue these efforts with particular emphasis on developing equivalents for netwoiks (for use in simultaneous engineering) and methods for solving boundry value problems.

References

[1] Laikin, J.H. and Simon, HA. (1985) "Why a Picture is Worth Ten Thousand Words[1]* Psychology Dept, Carnegie Mellon University, August 198S.

[2] Simon, H.A. (1978), "On the Forms of Mental Representation". In Savage, C.W. (ED.), *Minnesota Studies in the Philosophy of Science.* Vol. ix: *Perception and Cognition: Issues in the Foundations of Psychology.* University of Minnesota Press.

[3] South, N.E., Allison, J.E., (1987) "Generic Modeling for Optimum Materials Selection", presented to the Advanced Powertrain Subcommittee, October, 1987.

[4] J. Rehg, A. Elfes, S. N. Talukdar, R. Woodbury, M. Eisenberger, R. Edahl (1988) "CASE: Computer Aided Simultaneous Engineering," submitted to AEENG88.

[5] S. N. Talukdar, E. Cardozo, G. Podnar, "Coupling Symbolic and Numeric Programs in Large-Scale Software Organizations," Workshop on Coupling Symbolic and Numeric Computing in Knowledge-Based Systems, Bellevue, Washington, July 1987.