NBSIR 83-2671

The Standards Interface for Computer-Aided Design

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
National Engineering Laboratory
Center for Building Technology
Washington, DC 20234

April 1983

U.S. DEPARTMENT OF COMMERCE
NATIONAL BUREAU OF STANDARDS

NBSIR 83-2671

THE STANDARDS INTERFACE FOR COMPUTER-AIDED DESIGN

Fred I. Stahl

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
National Engineering Laboratory
Center for Building Technology
Washington, DC 20234

April 1983

U.S. DEPARTMENT OF COMMERCE, Malcolm Baldrige, *Secretary*
NATIONAL BUREAU OF STANDARDS, Ernest Ambler, *Director*

ABSTRACT

Building quality can be improved and building cost reduced through more effective computer utilization in design and construction. To accomplish these objectives improved interfaces are needed between building project databases and computer-based procedures for analysis and design, and between computer-based engineering procedures and applicable design standards. This latter task involves the Standards Interface for Computer-Aided Design (SI/CAD). The SI/CAD is the focus of the current report. The SI/CAD is shown to be a critical determinant of CAD system effectiveness, particularly in the domain of structural engineering design. This report examines the hypotheses that: (1) the ability to easily maintain design standards data is fundamental to CAD system effectiveness; (2) the configuration of presently available computer-aided structural design (CASD) system software inhibits efficient design standards data modification, requiring costly maintenance to avoid software obsolescence and limiting the overall usefulness of these systems; and (3) methods to enhance the efficiency of criterion checking and standards data maintenance are required to increase the utilization of CAD technology. Support for hypotheses (1) and (2) is developed from anecdotal engineering experience and the technical literature drawn principally from CASD. No evidence was found to support hypothesis (3).

KEY WORDS: building codes and standards; building delivery process; building design process; computer-aided building design; computer-aided design; computer integrated construction; engineering database management; structural engineering computer programs.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FREQUENTLY USED ACRONYMS

CAD	Computer-Aided Design
CAEADS	Computer-Aided Engineering and Architectural Design System
CAESE	Computer-Aided Engineering Software Environment
CASD	Computer-Aided Structural Design
CASP	Computer-Aided Space Planning
DICAD	Distributed-Integrated Computer-Aided Design
GENESYS	GENeral Engineering SYStem (proprietary to GENESYS, Ltd.)
GP/ICAD	General Purpose Integrated Computer-Aided Design
GT/ICES	Georgia Tech/Integrated Civil Engineering System (proprietary to Georgia Institute of Technology ICES Systems Laboratory)
GT/STRUDL	Georgia Tech/STRuctural Design Language (proprietary to Georiga Institute of Technology ICES Systems Laboratory)
ICAD	Integrated Computer-Aided Design
ICES/STRUDL	Integrated Civil Engineering System/STRuctural Design Language
IPAD	Integrated Programs for Aerospace (vehicle) Design
POLO/FINITE	Problem Oriented Language Organizer/Finite Element Analysis Program
SI/CAD	Standards Inteface for Computer-Aided Design
TSO/STRUDL	Time Share Option/STRuctural Design Language

ACKNOWLEDGMENTS

Ideas expressed in this report stem from the creative work of numerous individuals. The author is especially indebted to Dr. R.N. Wright, Director of the National Bureau of Standards' (NBS) Center for Building Technology (CBT), Dr. J.R. Harris of Structural Consultants Inc. in Denver, CO, Professors S.J. Fenves and D.R. Rehak of Carnegie-Mellon University's Department of Civil Engineering, and Professor L.A. Lopez of the University of Illinois' Civil Engineering Systems Laboratory. Their research in the areas of standards representation, constraint processing, and integrated systems for computer-aided design, which has spanned many years, provides the technical foundation for concepts presented here.

The author is also grateful to Mrs. J. Spoonamore and her colleagues of the U.S. Army Corps of Engineers' Construction Engineering Research Laboratory, Professor H. Borkin of the University of Michigan's Architectural Research Laboratory, Professor C.M. Eastman of Carnegie-Mellon University's Institute for Building Sciences, and Dr. L.Z. Emkin, Director of the Georgia Institute of Technology's ICES System Laboratory. These individuals shared valuable technical information describing their research efforts and products in the field of computer-aided building design.

In addition, the author wishes to thank Drs. K. Woodward W. Stone and E.V. Leyendecker of CBT, and Dr. B. Smith of the NBS Center for Manufacturing Engineering, for critically reviewing drafts of this report. Finally, the staff of the CBT Word Processing Center is gratefully acknowledged for typing the manuscript.

1. INTRODUCTION

1.1 OBJECTIVE AND SCOPE

Computer-aided design (CAD) soon will become the ordinary method for engineering design. Today, use of CAD has become commonplace in the development of integrated electronic circuits, aircraft and automobile components, and other machinery. However, building construction is a sector of the economy in which most of the potential increase in productivity from the utilization of computers has not yet been realized [29].

The National Bureau of Standards' (NBS) Center for Building Technology (CBT) is researching ways to improve building quality and reduce building cost through more effective computer utilization in design and construction. Pertinent to these overall objectives, CBT is seeking to improve information interfaces: (1) between building project data and the analysis and design procedures employed by various engineering disciplines, and (2) between computer-aided engineering procedures and applicable building design standards. This latter task concerns a set of problems termed by CBT researchers the "Standards Interface for Computer-Aided Design" (SI/CAD), and comprises the focus of the present discussion.

Principal subjects of this discussion are those technical problems arising from the automation of design checking procedures. Checking procedures compare design decisions and solutions against standards and other criteria, and are common to all design domains. CAD software systems typically expend considerable computational effort and memory to automate checking procedures. These procedures comprise the SI/CAD. In the present treatment technical problems diminishing the efficiency of the SI/CAD are explored within the context of structural engineering applied to the design and construction of buildings. Specifically, problems associated with the execution of checking procedures in computer-aided structural design (CASD) are considered in detail. Appendix A overviews the building delivery and structural design processes for readers not already familiar with these application settings.

1.2 THE STANDARDS INTERFACE FOR COMPUTER-AIDED DESIGN: TECHNICAL PROBLEMS

1.2.1 Technical Definition of the Standards Interface

Any discussion of CASD requires consideration of the relationship between generic standards and project-specific design descriptions: the SI/CAD. What is the SI/CAD, and why is it a determinant of CASD system effectiveness?

The essential process occurring at the standards interface is criterion checking. Criteria include provisions of design standards (e.g., maximum allowable shear stress in a beam) as well as user-defined requirements (e.g., minimum allowable ceiling height). During this process, individual elements of a candidate structural design solution, determined on the basis of their ability to withstand stresses resulting from anticipated project-specific loading and environmental conditions, are checked against generic criteria stipulating required structural qualities. A technical term for criterion checking in CASD is "constraint processing."

1.2.2 The Standards Interface as a Critical Determinant of CASD System Effectiveness: Hypotheses

Three hypotheses concerning the role of SI/CAD as a determinant of CASD system effectiveness are considered: (1) the ability to easily maintain design standards data is a fundamental determinant of CASD system effectiveness; (2) the configuration of presently available CASD system software inhibits efficient design standards data modification, requiring costly maintenance to avoid software obsolescence, and limiting the overall usefulness of these systems; and (3) methods to enhance the efficiency of criterion checking and the maintenance of standards data are required to increase the utilization of CASD technology.

These hypotheses arise from a number of practical engineering concerns, including: (1) the costs of design software downtime, (2) the risks of checking designs against obsolete or even incorrect standards, (3) the costs of being unable to use a desirable CASD software system because the system does not check designs against standards applicable to the problem at hand, (4) the costs of developing reasonable engineering design standards.

The ability to maintain a design standards database rapidly and at low cost provides a measure of a CASD system's cost-effectiveness, particularly when the costs associated with software "downtime" (required during standards database maintenance) are considered. For the most part, commercially available CASD software systems provide relatively inefficient mechanisms for maintaining design standards databases [53]. For example, while tables of structural steel shapes may be treated as data that can be quickly replaced by updated tables, design criteria and algorithms for checking instances against the criteria typically are "hard-coded" within CASD software systems. Thus, modifications to criteria and checking algorithms require altering system code, and this results in software downtime.

Building code design provisions tend to be diverse and unstable. Provisions often vary across geographic boundaries, and in addition, they undergo change periodically (three year cycles are typical). But the practice of structural engineering demands knowledge of, and adherence to, design standards which are both up-to-date and applicable to a given task and jurisdiction. Because CASD software programs provide no systematic means for maintaining design standards data, and because these standards change periodically, increases in design productivity usually assumed to result from CASD utilization may be called into question. Refer also to Appendix sections A.1 and A.3.

Another kind of problem arises in connection with the standards development process. Standards writers typically are unable to evaluate potential costs associated with new or modified provisions, primarily because there exist no mechanisms for simulating design performance under alternative versions of a standard. Standards writers could more effectively determine the impact of their decisions by developing alternative standards databases, employing each in the automated design of a structure, and then comparing design results produced under each version of the standard. However, no commercially available CASD software system presently permits such an application.

For these reasons, important changes in the conceptualization of the standards interface seem warranted. In particular, research is needed to: (1) develop a CASD system design philosophy which separates building design criteria and checking procedures from analysis and other components of CASD software systems, and which treats criteria and instructions for checking as data applied at system execution time; (2) develop an appropriate algorithm for constraint processing, and (3) develop standards for configuring design criteria databases.

Chapter 2 provides in-depth reviews of ICES/STRUDL[1] and GENESYS[2], two of the most widely used automated structural design software packages. The purpose of these reviews is to illustrate technical problems typically associated with the standards interface. The reader may obtain further background in the state-of-the-art of integrated computer-aided design systems (ICAD) and CASD by referring to Appendix section B.3.

1/ Integrated Civil Engineering System/STRuctural Design Language.

2/ GENeral Engineering SYStem.

2. CURRENT IMPLEMENTATIONS OF THE STANDARDS INTERFACE: SOME EXAMPLES ILLUSTRATING KEY PROBLEMS

2.1 INTEGRATED CIVIL ENGINEERING SYSTEM/STRUCTURAL DESIGN LANGUAGE: ICES/STRUDL

2.1.1 Overview

STRUDL is an engineering tool which automates complex and time consuming structural analysis and design tasks. STRUDL is based on the concepts that the design process is an iterative sequence of decisionmaking and computational tasks which are usually impossible to sequence a priori, and that any attempt to limit the designer to a single prescribed design strategy is undesirable. To implement these concepts, STRUDL permits the designer to configure any sequence of operations to suit the demands of an individual design problem [47].

The user communicates with STRUDL by means of a problem oriented language (POL), permitting the designer to structure English-like command phrases which control the flow of project information and engineering tasks. Principal components of STRUDL available to the structural designer include modules corresponding to the desired analysis mode (e.g., frame, finite element), modules permitting project data input (e.g., geometry, loads, desired properties), modules enabling the production of various STRUDL reports, and capabilities for invoking STRUDL's member selection and code-check functions. At any point during the design process, the engineer may call upon a STRUDL processor (in any logical sequence) to conduct a particular form of analysis, modify current geometric or loading configurations, select structural members from appropriate tables, or conduct code checks on the current design.

2.1.2 Implementation of the Standards Interface

STRUDL supports structural design through its code-check, parameter and geometric constraint processing, member selection, and design summary output capabilities. The system's code-checking and constraint processing features are of special interest here. The commercially available versions of STRUDL perform generalized steel frame design in accordance with versions of the American Institute for Steel Construction (AISC) specifications, and with Canadian, British and other foreign specifications. Specifications for the design of off-shore drilling platforms, steel transmission towers, and other special structures also are available within these versions of STRUDL. The design of concrete structures is performed in accordance with applicable standards of the American Concrete Institute (ACI).

When initiating a design session, the engineer declares the desired design specification by invoking a single STRUDL command. During the session, the engineer may invoke an alternative specification contained within STRUDL (e.g., AISC-1978 may be substituted for AISC-1969), and the results of design checks based on each specification can be readily compared. Depending upon specific options requested by the user, code-check summary output may include: controlling code provision values for critical design loading conditions at

critical sections along members, and actual and allowable values of all design code provisions and parameters for each active design loading condition at every section along a member [32, p. 25].

In general standards consist of (1) tables of structural shapes and members (in the case of steel frame design), (2) specifications of minimum requirements, and (3) rules for determining whether a selected (or designed) member conforms with applicable criteria. Each must be represented within the CASD environment. Structural tables are contained by STRUDL as data, and these may be updated or modified with relative ease. Specifications of minimum requirements and rules for determining design conformance, however, are not treated as data, but rather are "hard-coded" within STRUDL's program logic. As a result, periodic changes in criteria and/or algorithms for measuring design conformance, as may be promulgated by standards-writing organizations or building code jurisdications, require modifications to STRUDL software modules. Thus, while STRUDL provides numerous capabilities useful to the structural engineer, and while the program's application is likely to noticeably improve design productivity, it remains incumbent upon the user to ascertain that design standards employed by STRUDL during code-checking are current and valid for the project at hand. The commercial versions of STRUDL do not permit the individual user to update or otherwise modify portions of design standards, nor to "swap-in" proposed new versions of standards for purposes of comparative analysis. For this reason STRUDL in its present form would not be an effective aid to standards development and evaluation (purposes for which the program was never intended by its developers).

2.1.3 Availability

At present, there are two versions of STRUDL supported for widespread commercial use. The Georgia Institute of Technology's GT/ICES System Laboratory has developed and supported GT/ICES and GT/STRUDL [19, 32, 33]. GT/STRUDL may be implemented on user-owned Digital Equipment Corporation (DEC) VAX or Control Data Corporation (CDC) CYBER machines, and it also may be accessed on a conversational time-share basis through CDC. A second version, Time Sharing Option (TSO) STRUDL has been implemented in an IBM environment, and is available commercially on a time-share basis through the McDonnell Douglas Automation Company [63].

2.2 GENERAL ENGINEERING SYSTEM: THE GENESYS SOFTWARE LIBRARY

2.2.1 Overview

GENESYS, Ltd., a software development organization in the United Kingdom, provides an extensive collection of programs to support the engineering design and construction of buildings and other structures. Many of the products contained in the GENESYS library are versions of programs originally developed at universities and research laboratories. Other programs are proprietary to GENESYS, Ltd.

For the most part, GENESYS programs are configured into "suites" providing specialized engineering capabilities. Individual suites may be obtained for

structural analysis, concrete design and detailing, structural steelwork, ground engineering, etc. Each suite contains all the facilities necessary for developing and maintaining a project database, providing user-machine communications (including tabular and graphic output) and conducting applicable analysis, design, and checking tasks. However, a single data model is not common to all suites.

As with STRUDL, the user communicates with a GENESYS Suite through a POL. Sample commands from a suite developed for designing reinforced concrete structures are:

```
DEFINE GEOMETRY USING 'BASIC-GRIDS','GRIDS','LEVELS';
SET MATERIAL PROPERTIES 'STEEL','CONCRETE'; and
COLUMN DESIGN AND DETAIL OF GROUP 'TYPED'.
```

This last command causes a column element to be designed, in accordance with applicable design standards (in this case the British Code of Practice 110), and a detailed drawing of the column (illustrating the location of reinforcement) to be generated. Unlike GT/STRUDL and TSO/STRUDL, however, GENESYS suites may be used in batch mode only. Consequently, the GENESYS user is more apt to utilize the computer for automated production design, detailing, and drafting, while the STRUDL user is likely to use the system as an interactive aid to design decisionmaking.

2.2.2 Implementation of the Standards Interface

As with various versions of STRUDL, GENESYS supports structural design through code-checking and parameter and constraint processing. Moreover, GENESYS also contains specifications of minimum requirements and rules for determining design conformance (provided by applicable code provisions) as "hard-code" within a design suite's program logic. Thus, to update or maintain design standards, to design under an alternative version of a standard, or to employ GENESYS in situations requiring other than the British Codes of Practice each necessitates fundamental software modification. For these reasons, GENESYS seems best suited to local, volume-oriented engineering production work, and appears largely ineffective as a tool to assist standards development and evaluation (the latter tasks are not intended by GENESYS, Ltd.). The inability to link versions of American Standards easily is a principal deterent to the system's use in the United States.

2.3 ADDITIONAL CONSIDERATIONS

STRUDL and GENESYS are modern software systems for automating time consuming engineering tasks. The commercially available design systems are specifically intended to automate certain aspects of the structural engineer's work, most notably the proportioning and selection of frame members, in addition to analyzing structures. These systems also maintain project data, although this capability typically is limited in utility by the local requirements of any individual software system. Thus, data describing a building's frame geometry developed using STRUDL and stored electronically under a STRUDL-imposed data format may be quite difficult--and costly--to transfer to some other software

system used by a mechanical engineer or interior space planner requiring geometric data describing the same project. The concept that data should be developed and stored in a manner making them accessible to the numerous and diverse participants of a project has led to consideration of Integrated CAD (ICAD) technology. This subject has been explored in substantial depth by Rehak and Lopez [53], and is overviewed in Appendix sections A.1, B.1.1, and B.2 of the present report.

3. RECOMMENDED APPROACH FOR RESOLVING THE STANDARDS INTERFACE: CURRENT RESEARCH DIRECTIONS

3.1 INTRODUCTION

Researchers at CBT are working to determine standard measures, test methods, recommended practices, and performance criteria for algorithms and data structures applicable to the SI/CAD. CBT is particularly concerned with practices pertaining to automated constraint processing and design standards representation. These immediate concerns are consistent with the broader research goal of facilitating development of cost effective computer technology for building design and construction.

In recent years CBT investigators and other workers have developed a number of computer-based techniques that can be used to perform automated constraint processing for engineering structures [31, 38, 66]. There have been advances in the areas of computerized standards processing [35, 62], engineering database management, and automated structural analysis. Each of these areas has developed via relatively separate research paths, and it now is possible to integrate these technologies in the solution of problems associated with efficient constraint processing in engineering design. Essential to the evolution and implementation of constraint processing technology is the representation of building codes and standards in an appropriate computer-processible form. Toward this end, NBS has developed the Standards Analysis, Synthesis, and Expression (SASE) software package, a computer program based on a systematic methodology for machine-coding provisions of standards and the hierarchical relationships among provisions [22, 23, 34, 35, 62]. Technical approaches for resolving the standards interface, emphasizing the study of constraint processing and standards representation are discussed below.

3.2 CONSTRAINT PROCESSING

Although automated constraint processing [66] and supporting data structures [31] were analyzed in considerable depth more than a decade ago, no constraint processor based on these concepts has since been developed and implemented either commercially or on a research basis. CBT is currently seeking to develop performance requirements for such a device. The core of this research effort is CBT's experimental CASD software, based on the POLO/FINITE system. POLO provides database management and language development capabilities needed to support CASD system configuration. FINITE provides an initial structural analysis capability for a research CASD system. The immediate goals of CBT investigators are to implement member sizing, constraint processing, and optimization algorithms under POLO, and then exercise the experimental CASD software system to determine useful performance requirements for constraint processor design. Technical objectives of CBT's constraint processing research are to develop performance requirements for algorithms designed to: (1) reference design standards as external data by commercial CASD systems, (2) interrelate standards criteria, design data, and constraints, (3) determine the correct, and most efficient, sequence of checks under given design conditions, and (4) determine the status of constraints during the design process, using these data to guide design optimization. A detailed technical treatment of constraint processing is given in appendix section B.1.2 of this report.

3.2 STANDARDS REPRESENTATION

Pioneering work by Fenves [20] on modeling standards led to the concepts that: (1) individual provisions of a standard could be efficiently and advantageously modeled by decision logic tables which explicitly reveal rules linking conditions with actions, and (2) interrelationships among provisions of a standard could be modeled by information networks which systematically organize provisions according to their relative precedence (i.e., their links to "dependent" and "ingredient" provisions). This overall approach has since been expanded by Harris and Wright [35] to include complex organizational aspects of standards documents. Moreover, it has been used to model specific standards, including the Applied Technology Council's Tentative Seismic Standard [34], the American Institute of Steel Construction Specification [24, 26] and the American Concrete Institute's Building Code for Reinforced Concrete [51].

Beyond modeling standards to systematically examine their clarity, consistency, and completeness, Goel and Fenves [31] and Fenves [21] demonstrated that representations of standards consisting of networks of decision tables are applicable to constraint processing procedures. Following from this work CBT researchers, collaborating with other workers, have developed a technology for computerencoding the contents of standards [62]. The Standards Analysis, Synthesis, and Expression (SASE) software package was initially developed to support the analysis, formulation, and expression of standards, as an aid to the standards writing community. The system permits: (1) the coding of a complete standard in a single database, (2) the analysis of provisions to check for clarity and completeness, (3) the modeling and evaluation of relations among provisions of a standard, and (4) the formulation of alternative organizations of a standard's contents.

Because decision tables (the basic mode for representing individual provisions of a standard under the SASE model) are readily processed into computer program code [42], standards databases prepared using SASE are readily adaptable to a form required for automated constraint processing. CBT's ongoing research in the interrelated areas of constraint processing and standards representation, therefore, shall seek to determine performance requirements for algorithms which express the contents and organization of standards in a form permitting generic standards data to be conveniently linked to CASD systems at the constraint processing juncture. In this way, CBT expects to play a key role in improving the SI/CAD, and thereby in helping to remove known impediments to the effective utilization of CASD technology.

4. CONCLUDING REMARKS

4.1 REVIEW OF THE HYPOTHESIS

The SI/CAD has been defined as the juncture between standards data which are generic, and design descriptions which are project-specific. On a conceptual level the SI/CAD involves the nature of interrelationships between data describing a proposed (or existing) structure and those stipulating the structure's required qualities. On a working level the SI/CAD involves the implementation of algorithms for constraint processing, and thus is concerned with criterion checking employing procedures for checking individual elements of a candidate design solution against generic criteria found in regulatory documents and project specifications. CBT researchers believe the SI/CAD to be a critical determinant of CASD system effectiveness, and on the basis of practical engineering concerns have hypothesized that: (1) the ability to easily maintain design standards data is fundamental to CASD system effectiveness; (2) the configuration of presently available CASD system software inhibits efficient design standards data modification, requiring costly maintenance to avoid software obsolescence and limiting the overall usefulness of these systems; and (3) methods to enhance the efficiency of criterion checking and standards data maintenance are required to increase the utilization of CASD technology.

Hypothesis (1) is supported by anecdotal data from professional engineering practice suggesting general concern for design software which is either obsolete (i.e., checks design instances against outdated standards criteria) or is too limited in the range of standards and code criteria available for checking. These anecdotal data also suggest frustration on the part of some engineering practitioners, who find the task of updating or modifying in-house design software to be impractical, beyond their own technical capabilities or too costly. On the other hand, computer service bureaus offering CASD packages to designers on a time share basis, and very large architecture/engineering firms capable of maintaining a knowledgeable support staff, seem reasonably well equipped--and willing--to maintain CASD programs, even though this occasionally requires costly reprogramming, debugging, and testing.

The current review of available CASD software also provides support for hypothesis (2). The facts that the available systems incorporate standards criteria and checking algorithms as "hard code," and that the hard coding of standards data is one cause of high CASD software maintenance costs, were explored in this report by reference to two systems in widespread commercial use: ICES/STRUDL in the United States and GENESYS in the United Kingdom. Because large-scale investments in CASD software can often lead to frustrations associated with maintaining these systems, it has seemed reasonable to assume that hitherto undeveloped methods to enhance the efficiency of constraint processing and the maintainability of standards data are required to increase the utilization of CASD technology by the structural engineering community. However, CBT investigators have found no direct evidence to define a relationship between the effective resolution of the SI/CAD and CASD technology usage (hypothesis 3).

CBT's immediate objective is to mitigate certain important technical deficiencies of computer-based structural design tools, most notably those involving constraint processing and standards representation within CASD software systems. In the long range, however, CBT seeks to improve the overall efficiency of building delivery by formulating performance requirements for the computer-based integration of the entire building delivery process. To provide an overall framework for CBT's current and planned research on the SI/CAD, and to place SI/CAD research in the proper perspective, a brief exploration of "computer-integrated construction" and of the role of NBS in this important area is provided in Appendix C of the current report.

4.2 SUMMARY AND CONCLUSIONS

In little more than a decade, CAD system development has become big business. Properly conceived and utilized, CAD systems have considerable potential for improving the quality and reducing the cost of buildings. Incorrectly conceived and implemented, however, CAD and particularly CASD may create new and very complex problems. Of central concern are potential problems surrounding algorithms for checking designs against criteria derived from building codes and standards. At present, CASD system configurations frustrate the maintenance of internally stored standards data. Moreover, lacking a general model for standards representation, it is possible that a single standard may be interpreted and implemented differently within different software systems. As a result of such conditions, building designs developed using CASD software may not necessarily conform with the latest versions of applicable standards, and in some instances their validity may even be called into question.

CBT is seeking to minimize the potential for these and similar problems, and thereby to assist industry in its development of useful and effective design tools. To accomplish this, investigators are seeking to: (1) improve interfaces between building project databases and procedures for analysis and design employed by various engineering disciplines, and (2) define the interface between computer-based engineering procedures and applicable building design standards. This latter task concerns a set of problems termed the Standards Interface for Computer-Aided Design, or SI/CAD. Technical problems associated with SI/CAD implementation are of immediate concern to CBT researchers, and comprise the central focus of this report. To place the SI/CAD in the proper perspective, the report also introduces the wider concept of computer-integrated construction, and explores the appropriate role of NBS in this area of national concern.

A number of hypotheses were advanced concerning relationships between the effective resolution of the SI/CAD, the ability to efficiently maintain standards databases, and CAD usage in the building community. These hypotheses were partially supported by anecdotal data from engineering experience, and from a review of the CASD literature.

During recent years CBT has become active in two interrelated areas of research essential to the resolution of the SI/CAD. These are automated constraint processing and computer-based standards representation. With the implementation during 1982 of the Standards Analysis, Synthesis and Expression software

package, it now is possible to machine-code provisions of standards and the hierarchical relationships among these provisions. The link between standards representation and constraint processing involves the incorporation of rules contained within individual provisions of a standard, and of the hierarchical structure of provisions, into executable program logic. This link is studied in a current project.

In summary, it is widely felt that significant gains in building delivery productivity and building quality may be derived from increased utilization of computer technology in various segments of the building industry. Within its mission as the nation's central engineering and measurement laboratory, NBS is seeking to characterize critical aspects of design and construction automation, develop measures of design and construction technology performance, and specify effective performance requirements necessary in the manufacture and procurement of such technologies. Within this framework, CBT researchers are exploring new approaches to automated structural design and are considering ways of integrating by computer the many diverse construction tasks.

REFERENCES

1. Abel, J.F., McGuire, W., and Ingraffea, A.R., In the Vanguard of Structural Engineering. Engineering: Cornell Quarterly, 1981-1982, 16, 3 (Winter).

2. Anderton, G.L., Integration of Design Information. Washington, DC: NASA, Publication CP-2143, September 1980.

3. ANSI, American National Standard X3.9-1978: FORTRAN. New York: American National Standards Institute, 1978.

4. Armitage, B.S., and Hall, P.A.V., Conceptual Schema for CAD. Computer-Aided Design, 1977, 9, 3, 194-198.

5. Baer, A., Eastman, C.M., and Henrion, M., Geometric Modeling: A Survey. Computer-Aided Design, 1979, 11, 5, 252-272.

6. Bennett, J., Creary, L., Englemore, R., and Melosh, R., SACON: A Knowledge-Based Consultant for Structural Analysis. Palo Alto, CA: Stanford University, Computer Science Dept., Stanford Heuristic Programming Project, Memo HPP-78-23, September 1978.

7. Borkin, H.J., McIntosh, J.F., McIntosh, P.G., and Turner, J.A., Geometric Modeling Relational Database System. Ann Arbor: University of Michigan, Architectural Research Laboratory, July 1981.

8. Bylinsky, G., CAD/CAM: The Industrial Evolution. TWA Ambassador, July 1982.

9. CEPA, National Institute for Computers in Engineering. Rockville, MD: Civil Engineering Programming Applications, Inc., 1975.

10. Churchman, C.W., Wicked Problems. Management Science, 1967, 4, 14, 141-142.

11. D'Arcy, R., The Oxford System and its Applications. Proceedings of the International Conference on Computers in Architecture, Berlin, W. Germany, May 1979.

12. Davis, N.G., Interchangeability of Data Between Computer Graphic Systems Using Standard Interchange Format (SIF). Huntsville, AL: US Army Corps of Engineers, Huntsville Division, unpublished report, 1982.

13. Diggins, R.M., Preliminary Design of a Future Integrated Design System. Washington, DC: NASA, Report No. CP-2143, September 1980.

14. Dudnik, E.E., and Schachtel, W., Assessing and Evaluating the Desirability and Suitability of the Natural Environment for Human Activity or Development. In Carson, D.H. (ed.), Man-Environment Interactions, Part 3. Stroudsburg, PA: Dowden, Hutchinson and Ross, 1974.

15. Eastman, C.M., Prototype Integrated Building Model. Computer-Aided Design, 1980, 12, 3, 115-119.

16. Eastman, C.M., System Facilities for CAD Databases. Proceedings of the 17th Design Automation Conference, ACM/IEEE, Minneapolis, June 1980.

17. Eastman, C.M., Recent Developments in Representation in the Science of Design. Proceedings of the 18th Design Automation Conference, ACM/IEEE, Nashville, June 1981.

18. Eastman, C.M. and Lafue, G.M.E., Semantic Integrity Transactions in Design Databases. Pittsburgh: Carnegie-Mellon University, Institute of Building Sciences, Report No. 13, March 1981.

19. Emkin, L.Z. and Green, D.B., GT/ICES Concepts: A Modern Systems Approach. Atlanta: Georgia Institute of Technology, Civil Engineering Department, unpublished report, 1981.

20. Fenves, S.J., Tabular Decision Logic for Structural Design, Journal of the Structural Division, Amer. Soc. of Civil Engineers, 1966, 92, ST6, 473-490.

21. Fenves, S.J., Representations of the Computer-Aided Design Process by a Network of Decision Tables. Computers and Structures, 1973, 3, 1099-1107.

22. Fenves, S.J., Performance Requirements for Standards Processing Software. Pittsburgh: Carnegie-Mellon University, Dept. of Civil Engineering, Report No. R-79-11, April 1979.

23. Fenves, S.J., Functional Specifications for Standards Processing Software. Pittsburgh: Carnegie-Mellon University, Dept. of Civil Engineering, Report No. R-120-629, June 1979.

24. Fenves, S.J., Goel, S.K., and Gaylord, E.H., Decision Table Formulation of the AISC Specification. Champaign-Urbana, IL: University of Illinois, Civil Engineering Studies, Structural Research Series 347, August 1969.

25. Fenves, S.J., and Norabhoompipat, T., Potentials for Artificial Intelligence Applications in Structural Engineering Design and detailing. In Latombe, J.C. (ed.), Articial Intelligence and Pattern Recognition in Computer-Aided Design. New York: North-Holland Publishing Company, 1978, 105-121.

26. Fenves, S.J., Nyman, D.J., and Wright, R.N., Restructuring of the AISC Specification. Champaign-Urbana, IL: University of Illinois, Civil Engineering Studies, Structural Research Series 393, January 1973.

27. Fullenwider, D. and Lefever, J.P., Computer Graphics and the Practice of Architecture. IEEE CG&A, October 1981, 19-26.

28. GAO, Computer-Aided Building Design. Washington, DC: US Government Accounting Office, Report LCD-78-300, July 1978.

29. GAO. Agencies Should Encourage a Greater Computer Use on Federal Design Projects. Washington, DC: US Government Accounting Office, Report LCD-81-7, October 1980.

30. Goble, G.G. and Moses, F. Practical Applications of Structural Optimization. Journal of the Structural Division, Amer. Society of Civil Engineers, 1975, 101, ST4, 635-648.

31. Goel, S.K. and Fenves, S.J. Computer-Aided Processing of Design Specifications. Journal of the Structural Division, Amer. Society of Civil Engineers, 1971, 97, ST1, 463-479.

32. GT/ICES Systems Laboratory. Brief Overview of and Contrast Between GT/ICES and Conventional Engineering Software. Atlanta, GA: Georgia Institute of Technology, Dept. of Civil Engineering, GT/ICES Systems Laboratory, unpublished report, August 1981.

33. GT/ICES Systems Laboratory. GT/STRUDL Overview and Summary of Main Features. Atlanta, GA: Georgia Institute of Technology, Dept. of Civil Engineering, GT/ICES Systems Laboratory, unpublished report, August 1981b.

34. Harris, J.R., Fenves, S.J. and Wright, R.N. Analysis of Tentative Seismic Design Provisions for Buildings. Washington, DC: US Dept. of Commerce, National Bureau of Standards, TN-1100, 1979.

35. Harris, J.R. and Wright, R.N. Organization of Building Standards. Washington, DC: US Dept. of Commerce, National Bureau of Standards, BSS-136, 1981.

36. Hayward, P. Building Descriptions for Data Processing. In Hawkes, D. (ed.), Models and Systems in Architecture and Building. Cambridge, UK: Cambridge University Press, 1975, 76-80.

37. Henrion, M. Automatic Space-Planning: A Postmortem? Pittsburgh: Carnegie-Mellon University, Institute of Physical Planning, Report RR-72, January 1978.

38. Holtz, N.M. and Fenves, S.J. Symbolic Solution of Design Constraints. Proceedings of the 1981 Canadian Society of Civil Engineers Conference, Frederickton, New Brunswick, May 1981.

39. Hoskins, E.M. Design Development and Descriptions Using 3-D Box Geometries. Computer-Aided Design, 1979, 11, 6, 329-336.

40. Johnson, J. Pushing the State-of-the-Art. Datamation, February 1982, 112-114.

41. Johnston, B.G., Lin, F.J. and Galambos, T.V. Basic Steel Design (2nd Edition). Englewood cliffs, NJ: Prentice-Hall, 1980.

42. King, P.J.H. and Johnson, R.G. The Conversion of Decision Tables to Sequential Testing Procedures. The Computer Journal, 1975, 18, 4.

43. Lerro, J.P., Jr. CAD/CAM System: Start of the Productivity Revolution. Design News, November 16, 1981, 46-65.

44. Lev, O.E. Structural Optimization: Recent Developments and Applications. New York: Amer. Society of Civil Engineers, 1981.

45. Liggett, R.S. and Mitchell, W.J. Optimal Space Planning in Practice. Computer-Aided Design, 1981, 13, 5, 277-288.

46. Liggett, R.S. and Mitchell, W.J. Interactive Graphic Floorplan Layout Method. Computer-Aided Design, 1981, 13, 5, 289-298.

47. Logcher, R.D. ICES/STRUDL: An Integrated Approach to a Computer System for Structural Engineering. In Moore, G.T. (ed.), Emerging Methods in Environmental Design and Planning. Cambridge, MA: MIT Press, 1970.

48. Mitchell, W.J. Computer-Aided Architectural Design. New York: Petrocelli-Charter, 1977.

49. Mitchell, W.J. and Oliverson, M. Computer Representation of Three-Dimensional Structures for CAEADS. Champaign-Urbana, IL: US Army Corps of Engineers, Construction Engineering Research Laboratory, Report CERL-TR-R-86, 1978 (NTIS order No. AD/A-052 040).

50. Moses, F. and Goble, G.G. Minimum Cost Structures by Dynamic Programming. Amer. Institute of Steel Construction Journal, July 1970.

51. Noland, J.L. and Feng, C.C. ACI Building Code in Decision Table Format. Jorurnal of the Structural Division, Amer. Society of Civil Engineers, 1975, 101, ST4, 677-695.

52. Phillips, R.J., Beaumont, M.J., Richardson, D. and Bartly, J. Geometry for Computer-Aided Architectural Design. Computer-Aided Design, 1981, 13, 2, 89-97. Rehak, D.R. and Lopez, L.A. Computer-Aided Engineering: University of Illinois, Civil Engineering Systems Laboratory, Civil Engineering Studies, Research Series No. 8, July 1981.

53. Rehak, D.R. and Lopez, L.A., Computer-Aided Engineering: Problems and Prospects, Urbana, IL: University of Illinois, Civil Engineering Systems Laboratory, Civil Engineering Studies, Research Series #8, July 1981.

54. Richens, P. OXSYS-BDS. Bulletin of Computer-Aided Architectural Design, 1977, 25, 20-44.

55. Richens, P. The OXSYS System for the Design of Buildings. Proceedings of the 3rd International Conference on Computers in Engineering and Building Design, Brighton, UK, March 1978, 633-665.

56. Rosenthal, D.S.H. Building Description Techniques for Rationalized Traditional Construction. Proceedings of the International Conference on Applications of Computers in Architecture, Building Design and Urban Planning, Berlin, W. Germany, May 1979, 565-572.

57. Ruegg, R.T., McConnaughey, J., Sav, G.T. and Hockenberry, K.A. Life-Cycle Costing: A Guide for Selecting Energy Conservation Projects for Public Buildings. Washington, DC: US Dept. of Commerce, National Bureau of Standards, BSS-113, 1978.

58. Simon, H.A. The Structure of Ill-Structured Problems. Artificial Intelligence, 1973, no. 4, 181-201.

50. Simpson, H. A Human Factors Style Guide for Program Design. Byte, April 1982, 108-132.

60. Skidmore-Owings-Merrill. Computer Capability. Chicago: Skidmore, Owings and Merrill, unpublished report, 1980.

61. Smillie, K.W. and Shave, M.J.R. Converting Decision Tables to Computer Programs. The Computer Journal, 1975, 18, 2, 108-111.

62. Stahl, F.I., Wright, R.N., Fenves, S.J. and Harris, J.R. Standards Processing Software System: A Technique for Computer-Processing Representations of Standards. Paper presented at the Advisory Board on the Built Environment/ World Computer Graphics Association Joint Conference on Computers/Graphics in the Building Process, Washington, DC, March 23, 1982.

63. Thelen, J.F. A Tool for Steel Design: TSO/STRUDL. Proceedings of the Conference on Computing in Civil Engineering, Atlanta, June 1978, 513-526.

64. Tompkins, J.A. and Moore, J.M. Computer-Aided Layout: A User's Guide. Norcross, GA: Amer. Inst. of Industrial Engineers, 1978.

65. Tracy, F.I. A 3-D Stability Analysis/Design Program (3DSAD): General Geometry Module. Washington, DC: Office of the Chief of Engineers, US Army Corps of Engineers, Instructional Report K-80-4, June, 1980.

66. Wright, R. N., Boyer, L. T., and Melin, J. W., Constraint Processing in Design. Journal of the Structural Division, Amer. Society of Civil Engineers, 1971, 91, ST1, 481-494.

APPENDIX A: A BRIEF OVERVIEW OF THE BUILDING DELIVERY AND STRUCTURAL DESIGN PROCESSES

A.1 THE BUILDING DELIVERY PROCESS

The process of producing a complete building differs substantially from that of manufacturing a discrete machine (or even building) component. The production of a building involves a far larger network of tasks that often are more complex, diverse, and geographically distributed than those involved in the production of discrete industrial conponents (see figure A.1). In addition, criteria for building design and construction tend to be considerably more diverse and subject to change than those governing the design and manufacture of machine components. Finally, building industry participants typically are unable to invest capital at levels necessary to support large-scale process automation and integration. Taken together, these factors thwart some gains in building industry productivity which might be obtained through the effective use of computer technology. The special problems of building process complexity and design criteria changeability now may be described in greater detail.

Complexity of the building delivery process. The complex nature of the building process arises, in large part, from the wide diversity of approaches for creating and communicating design data, and from the fragmentation of the building enterprise into numerous distinct and often widely distributed organizations. The creation of a single building typically requires the coordinated efforts of numerous and diverse participants, including architects, space planners, engineers of various kinds, regulatory officials, construction managers, building contractors, and others. However, there presently exists no shared model for creating and communicating design information among these participants. As a result, much of the data describing a project often is either redundant or conflicting. For example, both the structural engineer and the space planner play a vital role in the building delivery process. In many ways, moreover, decisions made by either of these professionals affect those of the other. Drawings prepared by the space planner which illustrate the location of columns and other structural elements of a building duplicate the efforts of the structural engineer, who has also prepared drawings locating such elements. If the space planner incorrectly locates on a space plan drawing even one structural element, then subsequent space plan data will conflict with structural data describing the same building. Such conflicts are common throughout the building delivery process, and are often very costly to isolate and correct.

Although many individual participants in the building process presently employ computers for specific tasks (e.g., benefit/cost analysis, structural analysis and design, space planning), computer-based task integration and distribution among participants has not yet been achieved in the building industry. An important reason for this is the fragmentation of building delivery tasks among, for the most part, independent small businesses and specialists. Thus, to succeed in the building marketplace, providers of software for various analysis and design tasks have tended to limit their offerings to specialized, task-specific products executable on computing machinery most likely to be found in architectural, engineering, and construction firms. This market environment

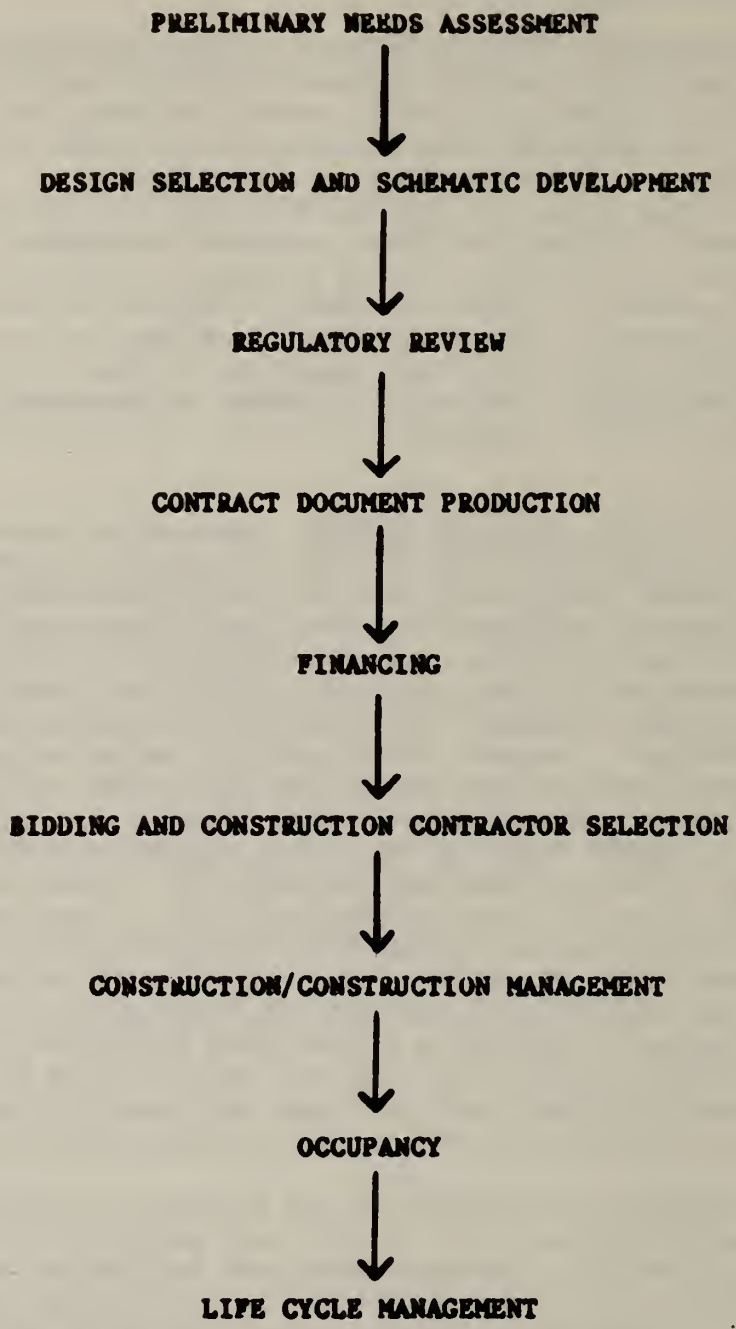


Figure A.1. A simplified overview of the building delivery process

provides little encouragement for computer-based building delivery task integration and distribution. As a result, it is generally not possible at present for data created on, say, the architect's desktop microcomputer to be transported to--and processed by--the structural engineer's time-share system.

In the present context Integrated CAD (ICAD) connotes the ability of diverse participants in a single building project to create and manipulate data, and to share these data through a common project database. Distributed-Integrated CAD (DICAD) extends this concept, distributing terminals and/or computers among project participants who are geographically separated, employing specialized software, using computing machinery of different configuration or manufacture, but are interacting through a common project database. Although DICAD is not currently practiced in the building industry, very limited forms of ICAD do exist today. Thus, a multidisciplinary architectural and engineering firm typically assembles a given project team from all disciplines needed to design the structure and prepare the contract documents, whether or not any of the tasks have been computerized. A number of such firms have already developed their own "in house" ICAD capability, enabling various participants in a given building project to create and manipulate project data residing in a central project database [13, 27, 60].

Diversity and Instability of Design Criteria. Another important impediment to the widespread application of computers by the building industry concerns the diversity and rapid change of design criteria affecting building projects. Design criteria may vary as a function of geographic, jurisdictional, technological, social, and other considerations. They also change over time. For example, design criteria effecting the thermal characteristics of a building vary from one geographic region to another. Similarly, criteria stipulated by local building codes often vary from one jurisdiction to another. In many instances, moreover, criteria may be interpreted differently by various participants in the building delivery process. Thus, a criterion governing ceiling height connotes spatial quality to the architect and space planner, while indicating minimum available clearances to the mechanical systems designer. Finally, many design criteria vary over time, as when the social or economic conditions affecting a project change during the design phase, or when standards bodies act to modify or add provisions of model codes and standards.

The diversity and changeability of design criteria have tended to shorten the useful life of certain important software tools, most notably engineering design programs. The checking of interim solutions against constraints and design criteria is a necessary component of the engineering design process. As indicated earlier in this report, applicable constraints and criteria are subject to considerable variation. Engineering design software in current use incorporates criterion values and checking algorithms as "hard code." As a result, changes brought about by regulatory, consensus, and other processes may render software obsolete (at least temporarily) and require costly maintenance [53].

A.2 THE STRUCTURAL DESIGN PROCESS

Structural analysis concerns the determination and resolution of the effects of forces acting upon and within a structure. Such force effects include those

arising from within the structure itself (dead and live loads), as well as those external to the structure (e.g., wind and seismic loads). Structural design is a process of configuring physical elements to resist forces and loadings in relation to known boundary conditions and materials properties, while satisfying desired functional requirements and specifications. Products of structural analysis include reactions, shears, and moments for beams, columns and other members. These data, when employed in conjunction with constraints, design criteria and material characteristics, enable members to be proportioned and configured. In traditional engineering practice (and in the commercially available CASD systems) structural steel member selection, for example, is accomplished by "looking-up" the desired properties in pre-coded standards tables, and then accepting the lightest member providing these properties.

Final design always results from some sequence of problem-solving operations coupled with various optimizations [41]. Refer to figures A.2 and A.3. The design objective usually is to maximize structural serviceability and safety while minimizing design, construction, and maintenance costs. This is accomplished through the simultaneous consideration of building configuration, loadings, materials, and code requirements. Structural optimization has been defined as designing and constructing a structure at lowest cost, while fulfilling well defined constraints [44]. All research and practice in structural engineering is directed toward this goal; structural optimization particularly connotes the development and application of automated techniques for improving designs. Thus, the computer becomes an essential tool for searching and sorting through design concepts, and for proportioning and detailing individual structural elements.

A.3 THE ROLE OF STANDARDS IN THE BUILDING DESIGN PROCESS

Standards are primary mechanisms for communicating technical information in the construction community. For purposes of this discussion the term "standard" includes all types of formal documents used to define the qualities of buildings, building products, or construction processes. During the design process, standards primarily serve to ensure provision of minimum acceptable levels of public health, safety, and welfare. The domain of structural engineering standards includes provisions specifying allowable and ultimate design stresses, and stipulating methods to be employed in the sizing, configuration, and connection of structural members.

A standard is usually developed by a small group of experts who, upon completing a draft standard, submit the document to the organization responsible for its promulgation and maintenance. However, the processes of promulgating and maintaining a standard may be of long duration, perhaps on the order of three to four years or longer between versions. In addition, rapidly changing societal demands (such as those concerning life safety or energy conservation), and the emergence of new technologies (such as electronic computation and new materials) frequently result in the need for new standards and changes in existing standards. While such factors contribute to the high cost of maintaining standards, they also increase the complexity, and therefore the cost, of maintaining CASD software systems. This is because checking design properties against standards

ANALYSIS	Collect and classify data; define the problem; develop criteria for evaluating solutions.
SYNTHESIS	Hypothesize alternative solutions to the problem.
EVALUATION	Select and develop the optimal solution.

Figure A.2 Fundamental elements of the design process

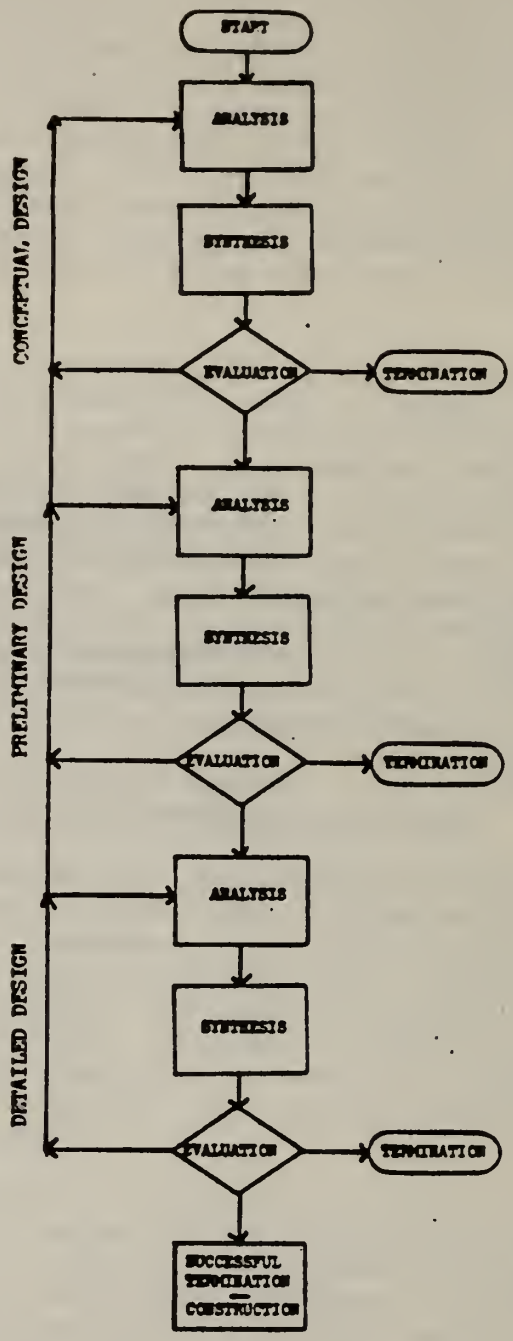


Figure A.3. Iterative engineering design process

is a fundamental feature of the design process (and therefore a necessary element of CASD software systems), and because much of the effort in maintaining CASD software involves keeping pace with and implementing periodic revisions to standards.

APPENDIX B. GENERAL TECHNICAL CONSIDERATIONS IN THE IMPLEMENTATION OF SOFTWARE FOR COMPUTER-AIDED STRUCTURAL DESIGN

Structural design is a complex task involving many procedures which are based on scientific principles, professional experience, judgment and intuition. When designing a structure, the engineer must adequately define the problem at hand, specify the criteria by which candidate solutions will be judged, synthesize and evaluate candidate designs, and ultimately decide upon a single acceptable solution. Some of these steps are more well-defined than others. For example, while the engineer may rely upon intuition or past experience to a considerable degree when selecting a conceptual approach or specifying overall design criteria, selecting individual structural components and detailing individual connections are likely to be governed by scientifically and empirically supported engineering standards.

Humans can solve complex engineering problems because they are intelligent; that is, humans can logically derive specific instances from general concepts, employ relatively efficient strategies for searching large solution spaces, and improve their own task performance by learning from past experiences. These abilities seem particularly necessary during the least well-defined portions of the design task, but may be relatively unnecessarily in carrying out such activities as calculating stresses and selecting structural members from codified tables. Currently available CASD software generally is not artifically intelligent[3], and as a result these systems focus only on aspects of design which lend themselves to determinate mathematical formulation. In general, these aspects have been limited to analyzing candidate structures (i.e., resolving forces acting on a structure under specified loading conditions) and selecting and configuring predefined structural members and components (i.e., identifying components which most efficiently satisfy all applicable constraints upon the structure, on the basis of generally accepted engineering standards). Thus, CASD software presently in commercial use is intended to assist the engineer by automating only those aspects of design which are least open to subjective interpretation.

The remainder of the present discussion considers a number of problems associated with available CASD systems. Of particular interest are difficulties in applying codified engineering standards during automated structural design. Technical considerations in CASD first are discussed, and several established computer-based systems then are reviewed.

B.1 TECHNICAL CONSIDERATIONS

The performance of CASD software systems may be specified and evaluated in relation to four issues. These include: (1) the efficiency of engineering data management, (2) the effectiveness of structural analysis, proportioning, and optimization procedures, (3) the efficiency of procedures for constraint processing, and (4) the ease with which the user interacts with the automated design system.

3/ An exception is SACON, a structural analysis "consulting system" [6].

B.1.1 Data Structures and Database Management

Overview. There are two general approaches to maintaining data generated or used by structural analysis and design software. These are application-specific files, and centralized application-independent databases. Application-specific data files are specially written (either manually or by an application program) to be read by either a single application program, or by a small group of highly interrelated programs which, taken together, comprise a unique application. The main advantages of application-specific files include their low development cost and their ability to insure high levels of data security and integrity. Essential disadvantages of the application-specific approach are the facts that data produced by one application may not be readily available to other applications (and as a result, there may exist considerable--and costly--redundancy in data used by similar applications), and data stored in application-specific files may be difficult and costly to maintain [53].

Centralized application-independent databases, in contrast, maintain data in a form making them acceptable to the widest practical range of application programs. Advantages of centralized databases include the facts that they substantially reduce data redundancy (since data required by many application programs need be stored only once), they provide special facilities for accessing and maintaining data (the so-called "database management system," or DBMS), and they enforce standards for representing information within individual applications. However, DBMS are usually difficult and costly to create, requirements for data security and integrity may be difficult to satisfy, the form of data most appropriate to the widest range of applications may be difficult to determine, and access time to data may restrict computational efficiency.

In most complex application areas, including structural engineering, it has already become quite clear in practice that the benefits of application-independent databases far outweigh those of application-specific files [53]. Some years ago, a number of architectural/engineering firms in the United States, desiring to automate certain analytical and design functions, began this process by procuring various independently-developed software packages and hardware systems. Many of these programs are still in use today, and they primarily deal with energy analysis, structural analysis, and automated drafting. The output from such packages typically are stand-alone reports and drawings. The operation of each program requires a unique data input protocol. Early users of such software quickly found, however, that much of the input data required for each program is both redundant across the programs, and is incompatible with those programs for which they were not specifically intended. For example, while each analysis or drafting program reads geometric descriptors into the computer, few programs read these data in the same form. Moreover, users noted that output from one program could usefully serve as input to another. Thus, certain data produced during structural design might appear in a final drawing, schedule, or specification. But to accomplish this, a single geometric description must be reformated to "fit" each application program, and in addition, data generated automatically by one program often must be reformatted manually before they can be used by another program.

Initial attempts to solve these problems within the production-oriented environment of architectural/engineering practice involved writing data manipulation "pre-" and "post-processors," and placing these between the production programs. The processors are special programs that rewrite data files making project data compatible across relevant application programs, and also less redundant. More recently, a number of integrated building design systems have been developed which maintain a single, readily accessible, project-wide database. Examples of such systems include ICES, POLO, and GENESYS for structural engineering data, BDS for architectural design data, and CAEADS and IPAD for both design and project management data. In general, the goal in configuring such data systems is to support the design of complex structures from early specification to full design development and construction instructions [18]. With the exception of IPAD and CAEADS, however, systems currently available focus on only a portion of the design/development process. Database management aspects of integrated design systems are reviewed in considerable detail by Mitchell and Oliverson [49], Rehak and Lopez [53], and Eastman [15, 16].

As noted earlier, DBMS to support CAD tasks and applications may be both complex and costly to develop. In general, engineering-oriented DBMS may be based upon hierarchical, network, or relational data models. A succinct summary of these models, and a consideration of their ramifications for CASD, have been provided by Rehak and Lopez [53]. Some investigators have suggested that existing commercially available DBMS are quite capable of supporting facility design databases (e.g., Armatage and Hall [4] in connection with computer-aided space planning and facility layout). Others have argued that more specialized DBMS are likely to be required because of the volume of data involved and the nature of the design process (e.g., Rehak and Lopez [53] concerning structural design, and Eastman [15, 16] concerning integrated CAD).

Representing project-specific information. Project-specific information generally includes design constraints, attributes of building subsystems and components, and representations of geometric configuration. Design constraints involve functional requirements (minimum clearances, dimensional constraints, spatial requirements, etc.), requirements imposed by codes and standards, and limitations concerning site usage, costs, and such factors as access to solar energy. The representation of functional requirements and spatial topology has been considered by Mitchell [48], Mitchell and Oliverson [49], Liggett and Mitchell [45, 46], and by Eastman [17]. Automated engineering design poses somewhat different problems in the representation of design constraints, primarily because engineering decisions require precise dimensional descriptions of discrete building components. In structural engineering, constraints are defined as project-specific applications of criteria imposed by building codes and standards [66]. To evaluate constraints on a structural system, it is necessary to apply applicable criteria at appropriate points in space corresponding to joints or other locations along structural members. The systematic checking of design instances against generic criteria is termed constraint processing. This task requires that project geometric data be maintained with sufficient precision to permit valid design decisions. Moreover, because complex engineered structures may require literally thousands of constraints to be evaluated, constraint processing also requires that project data be rapidly and efficiently retrievable. Problems associated with designing algorithms

and data structures for constraint processing in structural design have been considered by Wright et al., [66], Holtz and Fenves [38] and Rehak and Lopez [53].

Attributes of building subsystems and components include descriptions of subsystem and component types, materials, unit dimensions and weights, performance data, relationships among components of individual subsystems (e.g., ducts-to-furnace), and relationships among subsystems to a building (e.g., HVAC-to-structural). According to Hayward [36], a complete building description may best be represented by data contained in two separate files, one holding descriptions of components, the other holding information about spatial occurrences of these components. Since a particular component is likely to occur many times in a given building, this separation permits considerable economy in data storage. The representation of such project information has been accommodated in the development of such integrated design DBMS as OXSYS/BDS [11, 54, 55], IPAD [2], and also is evident in the work Borkin et al., [7] on geometric modeling.

Because buildings are artifacts with specific shapes and finite boundaries, geometric modeling and representation is fundamental to the objectives of CASD. Specification of structural form requires definition of spatial enclosure (i.e., topological relationships among components), location of components and subsystems in three-dimensional space, and description of shapes and boundary conditions. Geometric modeling and representation applicable to CASD also have been treated [5, 7, 39, 49, 52, 56, 65].

Representing codes and standards. In contrast to project-specific data which describe aspects of a finite physical entity, the contents of building codes, standards, and specifications are generic data applicable across all projects in any given class. Constraint processing merges design data and standards data to create, conduct, and record the results of tests of specific loading and resistance conditions at given spatial locations. To accomplish this, it is necessary that both design data and standards share a logical structure [66]. Thus, descriptors for discrete structural elements in a given project may be subscripted to denote member identifications, segment lengths, shape categories, and the like. Similarly, standards data may be subscripted to enable active standards data to be identified, to connect active provisions with associated design data, and to establish precedence (i.e., dependence and ingredience) relationships among provisions of a standard. The purpose of a shared logical structure, then, is to assure that correct and complete data are available for any given criterion check, and that checks are conducted in the proper sequence.

Efficient constraint processing requires explicit relations between elements of standards data (i.e., individual provisions). This objective is most readily achieved when the provisions of a standard have been modeled as decision tables, and the relations among provisions have been structured as networks. Such formulations of standards are feasible and have been demonstrated, and with the availability of SASE appropriate organizations and formats for CAD standards databases are now within the reach of researchers.

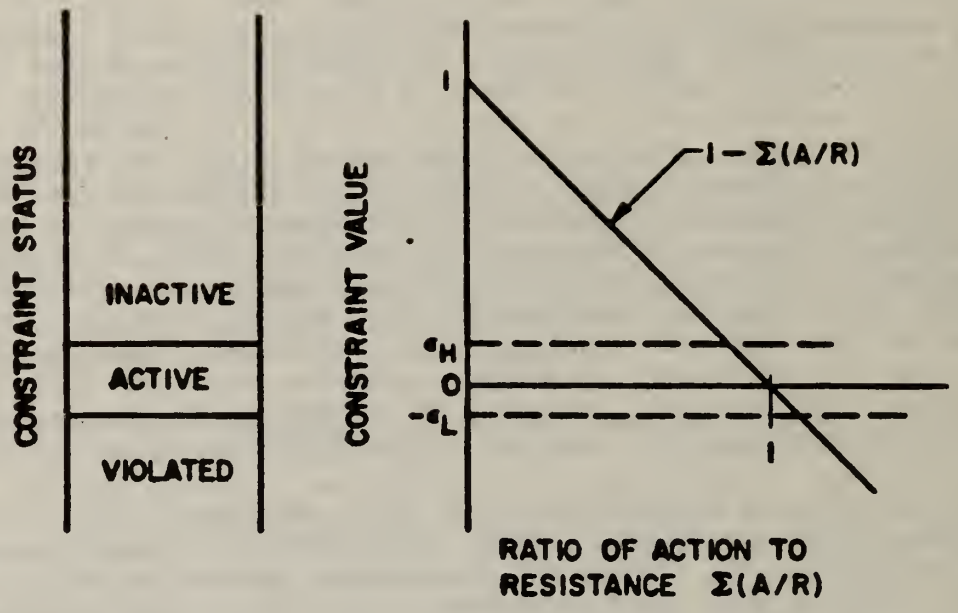
B.1.2 Constraint Processing

The constraint processing algorithm discussed by Wright et al., [66] specifies a procedure independent of both generic standards data (design criteria) and project-specific design data. Ideally, therefore, software implementations of the constraint processing algorithm should permit standards to be referenced as external data, as are project data. In this way, software maintenance to ensure the validity of standards employed during criterion checking can be reduced to the seemingly trivial task of exchanging one standards database for another (the author recognizes that the complementary task of generating such external standards databases in the useable format is by no means trivial; see Harris and Wright, [35]). However, current implementations of constraint processing in available CASD software systems do not possess such general purpose capabilities.

In the procedure initially envisioned by Wright et al., [66] a constraint is defined as a particular application of a design criterion imposed by a standard or user requirement. Within the domain of structural engineering, a constraint typically is characterized by the mode of failure specified in the criterion (e.g., vertical shear), the point of application of the criterion (e.g., beam endpoint), the physical location of the point of application, and the loading or environmental condition producing the responses referenced in the criterion. The function of the constraint processing procedure, then, is to evaluate all critical points in a structure and determine whether, under the given loading conditions, applicable constraints are: (1) "active", that is, the element as designed provides required resistance within allowable tolerances, (2) "violated", that is, the element as designed provides resistance below the minimum permitted, or (3) "inactive", that is, the element as designed provides resistance in excess of that required. Note figure B.1.

In the formulation by Wright et al., two basic operations recur in constraint processing. These are SEEKing a value for an absent attribute (e.g., area, moment, deflection) and WARNing that the current value of an attribute has been changed. After an iteration of the design cycle has been completed, the SEEK function searches all ingredient attributes defining a specific design condition and identifies those voided (or otherwise absent) during the cycle and requiring (re)computation. SEEKing is performed by tracing design parameters backward through a constraint's ingredients, until a basic input value is provided. For example, modification of a beam during one design cycle may void the area of a column determined earlier. This area must be flagged for recomputation, the ingredients of the area computation (width and depth) must be identified, and the values for these variables must be provided. Similarly, the complementary WARN function keeps track of changes in design parameters and attributes, and voids other dependent attributes of a specific design condition for which an antecedent parameter has been changed. WARNing is accomplished by tracing design parameters forward through a constraint's dependents, until a logical termination point is reached. Taken together, SEEK and WARN provide the link between the design database, the standards database, and the rules for checking design conformance.

The status values of constraints themselves are data which describe important characteristics of a candidate design solution. These data indicate precisely



where ε_H and ε_L denote tolerances

(from Wright et al. [66])

Figure B.1. General form for design criterion

the points at which the design is sufficient (denoted by active constraints), insufficient (violated constraints), and "overdesigned" (inactive constraints). Thus, they indicate to the designer (whether a human engineer or an automated structural design system) objectives for design optimization.

Commercially available CASD software systems such as ICES/STRUDL and GENESYS implement variants of such a procedure. Accordingly, these systems automatically design or select structural members on the basis of minimum weight or other objectives specified by the user. They do not, however, physically separate criteria and the checking algorithm from analysis and other CASD components, a key goal for enhancing the effectiveness of CASD technology [21, 53, 66].

B.1.3 Structural Analysis, Design, and Optimization

In general, automated design procedures arrive at a design the engineer could equally as well have obtained if time and money were available to directly search among the design alternatives [44]. Thus, CASD software systems are typically configured to design assemblages by first performing a structural analysis to obtain force effects in the elements, and then by optimizing each element separately. The resulting structure is then reanalyzed, and the procedure repeated. In most cases, this process converges rapidly to an acceptable solution.

Goble and Moses [30] point out, however, that element design procedures are limited to constraints which describe only element behavior, and therefore to structures where design is based strictly on fixed codes and procedures. Accordingly, optimization based on element design may be relatively ineffective where overall system constraints such as stability, natural frequency, aeroelastic behavior, and overall deflection must be considered. Moreover, the element design procedure is not applicable when additional variables relating to the whole structure, such as geometry, interelement sensitivity to stiffness, material, fabrication, and construction, are introduced. In these special cases, element optimization is useful only as a subroutine within a broader and far more complex set of mathematical operations. One final difficulty noted with element optimization based CASD concerns the possibility that designing one element at a time might lead to large number of members of different size being specified in a single structure. Solutions to this problem have been developed using dynamic programming routines which consider the results of element optimization, and then optimally select elements, and the number of each, to be used [30, 50]. Similar solutions are implemented in the commercially available CASD packages.

B.1.4 Human Factors Considerations in CASD

Technical conserations in the user-machine interface also are important to an effective standards interface. Human interactions with software systems and computer-based engineering design tools typically concern (1) data entry, (2) display (screen) layout and design, and (3) task sequence control. Experience and research in human factors engineering suggests that optimal system performance can best be assured when the needs of system users have been effectively accommodated [59].

Human factors considerations play a significant role in SI/CAD. In particular, it is important that the analyst be kept informed about the status of constraints as these are evaluated. Only with this information can the designer effectively intervene in the design process. Similarly, the engineering designer should be kept informed about the standard and version in effect for a given procedure, and should be provided continuous feedback regarding provisions being invoked at any point. These are especially important where processing is distributed among or being monitored by a number of participants. Such feedback permits analysts to study the constraint evaluation "trail", the basis for automatic design decisions and for intelligent user intervention in the design process. Clearly, it is up the the system software designer to provide such enhancements and capabilities in CASD systems.

While experts have not as yet adopted specific performance requirements for software design, they generally agree that "good" software systems are easy to learn, produce simple and readable displays, are supported by easy-to-follow documentation, and are written for less specialized users (not for computer experts). To develop software that may be correctly and efficiently used, human factors specialists consider a number of basic principles [59]: (1) The software should provide the user with feedback that is immediate and unambiguous. Feedback should be provided at the location, and in the form, expected by the user. The completion of some process, the failure of a process to successfully terminate, incorrect input syntax, and data entry errors are examples of points requiring feedback to the user. Such information should be sufficiently explicit to advise the user about what to do next. (2) Program organization and program-user dialog should be consistent. The system designer should establish and strictly follow rules which permit the user to learn a part of the system, and then to apply this new knowledge to other parts of the system. That is, one part of the system should not contradict any other part. (3) Use of the program should be matched to operators' skill levels and roles.

From a human factors viewpoint, software systems for structural engineering pose new problems for the programmer. These problems derive largely from the complexity and data intensity of most structural engineering problems, the predisposition of structural designers to think in visual terms, and the interrelationship between principal engineering tasks, particularly analysis, proportioning, and optimization. To reduce the impact of these sources of complexity, a number of different modes for controlling complex processing in CASD have been employed in practice. Each has different human factors implications. For example, the ICES and POLO program development environments rely on the use of POLs to simplify data entry and sequence process control, and thereby reduce the likelihood of user error. POLs are intended to simplify system use by reducing person-to-computer communications to an abbreviated English-like format. Consider the statement "JOINT COORDINATES" in ICES/STRUDL, which tells the computer that the following data specify spatial locations of member endpoints, or "STIFFNESS ANALYSIS," another STRUDL command, which requests a particular process to be initiated.

POLs have been augmented to better suit graphically oriented users such as design engineers. For example, graphic input mode permits the designer literally to draw a frame, solid, or mesh directly onto the screen, while the

computer automatically records the two- or three-dimensional coordinates of member (or element) endpoints. Similarly, "menus" of all available modes and commands can be continuously displayed on the screen, and the user can automatically be prompted to select a command from the menu at appropriate occasions during a design session. In addition, menus do not require the user to memorize lengthy lists of commands and mnemonics (or to frequently consult the user's manual).

Human factors considerations also have resulted in the better utilization of graphics in CASD output displays. Most significant is the application of interactive graphics in CASD, in which graphic output from one design cycle may be interpreted and manipulated by the user, and then dispatched for processing as input to the next cycle. Among the most innovative applications of interactive graphics has been in the area of structural mechanics, in which it is now possible to simulate and visualize in three dimensions load displacements, deformations, and failure modes [1].

B.2 INTEGRATED COMPUTER-AIDED STRUCTURAL DESIGN SYSTEMS

B.2.1 State-of-the-Art

Potential economies derivable through the integration and distribution of CASD tasks were discussed in appendix section A.1 of this report. In particular, cost savings resulting from minimizing redundancy and error in a project database were considered. However, it also was noted that many of the goals associated with ICAD and DICAD may be somewhat difficult and costly to obtain. At present, impediments to ICAD include: (1) lack of a uniform approach to developing integrated engineering design software; (2) lack of consensus regarding the best model for a task-independent database for building design; and (3) lack of accepted data definition and data manipulation language standards applicable to building industry participants. An additional technical impediment to the implementation of effective DICAD concerns the lack of standards for interfacing hardware and operating systems of diverse design and manufacture, although this is not the focus of the present report.

Rehak and Lopez [53] have argued that the development of integrated computer-based engineering aids has been hampered further by the proliferation of singlepurpose software tools. According to these investigators, singlepurpose software tends to work reasonably well in practice, under the very limited sets of conditions for which it was designed. But because each individual program usually is based on a unique view of a limited problem domain, and because engineering practitioners themselves are limited in their technical concerns and responsibilities in connection with a building project, potential economies deriving from a more sophisticated engineering software concept have not proved obvious in practice. It may be that future widespread development of ICAD and DICAD, and the economies derivable through their implementation, must follow certain fundamental changes in the building process itself. This concept is supported by the previously cited pioneering of in-house ICAD by architectural/engineering firms already organized to integrate essential building development tasks. Recent experiences with ICAD, and to a lesser degree DICAD, cluster into three broad categories: (1) industry-wide systems

for computer-aided design, design management, and manufacture, represented by IPAD; (2) environments for CAD system development, represented by ICES and POLO; and (3) systems for linking disparate CAD software components, represented by GENESYS and CAEADS. General aspects of these approaches applicable to research on SI/CAD now can be reviewed.

IPAD is perhaps the most ambitious undertaking to develop an integrated and distributed computer-based engineering environment. IPAD was developed by the Boeing Computer Services Company for the National Aeronautics and Space Administration. The system consists of computer programs intended to support the aerospace-vehicle design process by aiding design management tasks, technical tasks, and supporting requirements. All project management and technical data, together with standards data, are stored in a single task-independent database. IPAD functions permit project personnel to query the database and perform operations on project data. IPAD sequences the task elements of a complex operation, providing common access to a single database by the various participants in the design/development process. Such capabilities can be provided either on a single host computer, or across heterogenous machines on a distributed basis [13].

Under IPAD, engineering analysis and design are treated either by IPAD-supplied functions and programs, or by user-defined software components. When developing special-purpose software, the user must also develop and implement the necessary interfaces between the application program, the IPAD data manager, and the IPAD operating system. Each added application program must, moreover, contain its own user interface.

While IPAD provides computer-based support for long-term multi-user design/development process management, in contrast ICES and POLO provide environments for engineering software development. These systems provide data management and interface facilities; linking new application programs is accomplished through the database and support system software. An approach combining both a program development environment, and a cadre of built-in applications, all linked through an application-specific executive system is illustrated by GENESYS.

In contrast with IPAD, ICES, and POLO, the United States Army Corps of Engineers' CAEADS consists primarily of a collection of single-purpose (albeit interconnected) programs to assist "pre-concept" building design. These include SKETCH (floor plan creation and editing), SEARCH (design evaluation), BLAST (energy analysis), and similar functions. CAEADS (and GENESYS, which also links individually developed software components) illustrates certain problems which may arise when system integration is sought by linking independently developed computer programs. An example problem in CAEADS concerns invoking the BLAST energy analysis option. Current data describing the project must first be passed to an intermediate processor which develops a BLAST data file and run-stream. This new off-line run-stream is then submitted as a batch job on a separate computer. Results of the analysis are provided in a report, and do not automatically update the CAEADS project database (in contrast with SEARCH, which does update the database). Thus, database maintenance on the basis of BLAST results is a manual task. CAEADS has no

structural analysis and design capability at present, and therefore will not be treated further in the present report.

B.2.2 Future Prospects for General Purpose Integrated CASD

Thus, ICES, POLO, GENESYS, and to a lesser extent CAEADS may be viewed as ICAD support systems [53]. Each provides database management, language translation, and other run-time features, and was primarily designed to support large analysis applications (e.g., STRUDL under ICES, and FINITE under POLO). None of these support systems contains application-independent constraint-processing capabilities, and both code-checking algorithms as well as standards data must be provided within individual application programs. Under each of these systems, therefore, program maintenance to assure the currency of applicable building codes and standards may be time-consuming and costly. Similarly, users of these systems are "locked into" applying only those standards that have been pre-programmed into the design software.

If ICES, POLO, and GENESYS exemplify the current generation of CAD system integration, then the next generation of such systems will be distinguished by their considerably more general purpose capabilities. Ideally, general purpose ICAD (GP/ICAD) systems will be functionally independent of computing hardware, application discipline, design task, and standards and other design constraints.

Thus, they will ensure that diverse discipline-specific application programs can be incorporated at any point during the development of a complex building project, and that project knowledge can be efficiently and correctly shared among the numerous applications. Moreover, future GP/ICAD systems will readily enable simultaneous design processing by various participants, and will permit the design of large or complex structures to be distributed geographically (even where diverse contributors employ different computing machinery, data structures and data definition languages, etc.).

Rehak and Lopez [53] have offered a GP/ICAD concept design intended to mitigate most important problems associated with current generation integrated engineering design systems. The remainder of this section describes the approach taken by these investigators. Designated "Computer Aided Engineering Software Environment" (CAESE) by Rehak and Lopez, the system is intended to perform neither as a single ICAD device nor as a collection of diverse programs. Rather, it is a collection of system components applicable to CASD and other highly complex engineering problems. CAESE is a prototypical design which has not as yet been implemented as a working software system.

CAESE is envisioned to consist of two levels. The top level is the applications environment, in which reside all domain-specific tools, programs, and data required to perform actual design and engineering computations in connection with a given project. In this context, the term "domain" denotes a particular project, discipline, field of knowledge, or area of engineering practice. The bottom level contains the system environment, in which reside all components, support software, and data which are both domain- and application-independent. This level provides engineering design and computational support, but itself performs no engineering tasks. Also at the bottom level is the support

environment, which provides tools for developing and maintaining both the system and application levels.

The system environment (bottom level) is envisioned to consist of components required to provide system and run-time support features needed to solve engineering design problems. Components include an engineering-oriented database management system, an engineering design process controller, a constraint processor (i.e., standards interface), a project manager, and user-system interfaces. The support environment (also at the bottom level) assists the user and system programmer in developing, managing, and maintaining various software components. This environment includes software for encoding building codes and standards into machine-processible form, software for developing and machine-encoding knowledge-based rules for driving the design process, programming languages for developing system and application components, and other similar support features.

All engineering analysis and design is envisioned to occur within the application environment of CAESE (the top level). CAESE is intended for those applications encompassing large, multi-discipline problem domains. These are typified as being large-scale projects which are complete systems containing a number of integrated subsystems from various engineering disciplines, which derive from relatively ill-structured problems, and which are governed by a variety of standards. Examples of such projects include nuclear power plants, off-shore drilling platforms, high-rise buildings, aircraft, and even large-scale software.

Each application domain is viewed as an individual application environment (e.g., CAESE-Nuclear Power Plants; CAESE-Aircraft; etc.). Within each application environment, various subsystem software modules and databases representing numerous engineering disciplines are integrated to form a complete engineering design system. Many subsystems and modules are cross-disciplinary in their function, and need only be stored once in either the system or support environments. Examples include programs for finite element analysis, proportioning, and constraint processing, which are applicable to the design of office buildings, off-shore platforms, ships, and so on. All such elements are then linked to create a unique application environment which appears unique to the user.

Rehak and Lopez's conceptual design for CAESE represents only an initial step in defining the next generation of engineering computer aids. CAESE software development, however, is expected to involve numerous tasks and require hundreds of person-years. Once system and support environments have been developed and tested, it will be necessary to select characteristic applications and develop the necessary application environment software and interfaces. Finally, actual engineering problems for which there are known solutions must be redesigned under CAESE, so that the validity and effectiveness of the entire system may be evaluated [53].

APPENDIX C: ELEMENTS OF COMPUTER-INTEGRATED CONSTRUCTION

C.1 A NATIONAL PROBLEM

The efficiency of the building delivery process can be improved through reliance on computers in many of the individual tasks comprising the whole, such as design, analysis, drafting, bid preparations, construction planning, and regulatory review. Moreover, the efficiency of the process can be improved through reliance on computers for communication and coordination of technical data between the various processes that comprise building delivery. Indeed, an important effect of bringing computers into the building delivery process is the anticipated improvement in buildings themselves, brought about by better organizing information required in the solution of complex building problems. A recent study of computer-aided design for Federal agencies concluded that computers enabled designers to produce better buildings [29].

C.2 INTEGRATED CONSTRUCTION PROJECT INFORMATION

Construction projects are characterized by a large number of participants (owners, architects, engineers of various kinds, regulatory officials, fabricators, etc.) who must create, use, and modify a large amount of interrelated data. In practice, considerable duplication of effort typically takes place. In preparing shop drawings, for example, the fabricator often repeats the calculations of the estimator, who repeated the calculations of the designer. This duplication of effort is expensive, and becomes doubly expensive when errors are made (a seemingly inevitable occurrence). The use of a computer as the central depository for all data describing a single project would greatly increase productivity by minimizing redundancy and opportunities for error, and has the possibility of revolutionizing construction bid preparation, since all competitors could work from identical, certified sets of material quantities.

C.3 ECONOMIC ANALYSIS AND LIFE CYCLE COSTING

The pre-design planning of large construction projects requires careful analysis of potential affects of the construction upon surrounding physical, social, and economic systems [14]. The need to evaluate construction project life cycle costs (LCC) provides additional rationale for a computer-integrated construction information system. LCC analysis is a method of economic evaluation of alternative project plans which considers all relevant costs and benefits associated with each alternative over its life [57].

The principal steps in performing an LCC analysis include: (1) identifying alternative design solutions; (2) identifying life cycle costs, benefits, and use scenarios; and (3) comparing alternatives. The success with which an LCC analysis will be accomplished depends upon the degree of detail and accuracy employed in describing design options, life cycle costs and benefits, and upon the correctness of scenarios on which the analysis is based. Thus, LCC analyses are highly dependent upon up-to-date and well organized information, and are also computationally complex. For these reasons, they tend to require computer aids. In order to make use of and build upon available project data, and to

reduce the likelihood that costly measures will be duplicated, it is imperative that LCC techniques be integrated with other aspects of the design process which affect the evaluation and selection of design solutions.

C.4 COMPUTER-AIDED SPACE PLANNING

Computer-aided space planning (CASP) techniques offer certain potential advantages over traditional, manual methods. In particular, they offer speed, accuracy, and versatility. Moreover, because CASP enables the rapid generation of a wide variety of feasible solutions, and because in some cases CASP techniques operate faster as the problem becomes more constrained, CASP may enable designers to conduct complex conceptual design analyses which are virtually impossible using manual methods [28, 37]. An inherent feature of the building design process is that decisions which critically affect building quality and life cycle cost usually occur early during the conceptual design phase. However, less than 20 percent of the overall design manpower and resources usually are devoted to this phase of a project [28]. Consequently, CASP offers the potential to greatly improve manpower and resource effectiveness during a critical stage of the design process.

However, CASP remains the rare exception, rather than the rule, in architectural practice today. Incomplete software documentation, hardware and software incompatibilities, unavailability of software packages, and unreliable or incomplete input data are some of the oft-cited impediments to wider CASP utilization [28, 64]. These impediments are typical for single-discipline CAD development within so broad, complex and multi-disciplinary an enterprise as the building delivery process. Indeed, the development of more effective CASP software systems, and the far more widespread commercial utilization of these sytems, is likely to benefit from large-scale computer-based integration of the construction process.

C.5 COMPUTER-AIDED REGULATORY ACCEPTANCE

One of the frequent complaints heard in America today is the hidden cost created by the combination of regulatory delays and inflation. One potential remedy is to speed the review of plans, specifications, and calculations that must be performed by local building officials preparatory to issuing building permits. The task of regulatory review involves checking design instances against criteria for safety and serviceability, and hence, is directly analogous to constraint processing, discussed earlier in this report. Computer-aided regulatory acceptance, like SI/CAD, will require the marriage of automated constraint processing and computer-based standards representation technologies. An important benefit of this approach concerns the opportunity for building officials to more thoroughly and systematically evaluate designs against ever-growing and more sophisticated requirements in building codes without adding a large number of technical specialists to their staffs, thus reducing the need to increase tax revenues or fees for building permits.

C.6 OPERATING AND SAFETY SYSTEMS

During the life of a building numerous problems arise in connection with both day-to-day operations (e.g., building heating and cooling) and emergency situations (e.g., fire). Computer systems often are installed in buildings to deal with such problems. Frequently, systems installed for such purposes require continuous or intermittent sampling of various aspects of the building and its environment, and conduct comparisons of samples against baseline or other data for the building. Solution techniques for operational and safety problems also may involve reviewing previous instances of similar problems. In order to facilitate the effectiveness of such systems, it is essential that integrated project information be maintained not only during a building's design and construction, but throughout its life as well.

It is clear to see that "computer-integrated construction," as briefly described above, is an expansion of the ICAD and DICAD concepts introduced earlier in this report. It is also clear that research on SI/CAD, the primary focus of this report, has ramifications for aspects of building delivery lying beyond computer-aided structural engineering. The importance of NBS participation in research concerning the general area of computer-integrated construction now may be considered.

C.7 MEASUREMENT TECHNOLOGY FOR COMPUTER INTEGRATED CONSTRUCTION

In the jargon of artificial intelligence, building delivery requires the solution of "ill-structured" problems [58], and even of "wicked" problems [10, 25]. The building industry has much to gain by working to make these problems tractable through the development and implementation of computer-integrated construction technologies. However, industry cannot be expected to bear the full cost of needed fundamental research. Working within its mission as the nation's engineering and measurement laboratory, NBS can bring considerable expertise and facilities to bear on identifying and characterizing the component parts of computer-integrated construction technology, on developing the necessary measurement methods, and on deriving effective performance requirements for future (privately developed) hardware and software products.

C.7.1 Problem Characterization

While each element of computer-integrated construction has specific technical components, there are a number of fundamental issues which cut across these elements. These include the need to: (1) evaluate the validity of analytical models, algorithms, and interfaces among algorithms which comprise computer-integrated construction software systems; (2) ensure the portability of needed software in view of the large number of diverse actors and systems involved; and (3) facilitate the communication of information among diverse actors and systems comprising complex building projects.

Analytical models underlying algorithms for computer-aided environmental analysis, economic analysis, engineering design, and functional design are extremely complex in the sense that they frequently contain many parameters which interact in ill-defined ways. As a result, validating models, algorithms,

and interfaces itself becomes a difficult task requiring basic research. But the process of validation is seen as two-directional: by systematically adjusting models against realworld events, the analyst also obtains a much deeper understanding of the nature of these events.

Because complex building projects involve the participation of numerous diverse participants, the enchancement of construction productivity requires that obstacles to communications among these actors be minimized. Such communications may be greatly facilitated through the implementation of: (1) standard languages for developing design and analysis software; (2) standard formats and procedures for maintaining project information databases; and (3) software and database systems which are portable across the variety of computers likely to be found at various points during the construction process (e.g., main-frames employed during large-scale design and engineering analysis; graphics devices and word processors used during later phases).

C.7.2 Performance Requirements

To be effective in use, the performance of software systems applicable to computer-integrated construction technology must be properly specified. Performance requirements for these systems must stipulate measures and measurement methods for their verification, validation, and certification, and must serve to ensure the overall reliability of the software (i.e., improving the probability that a program will perform its intended function, with minimal error, over some period of time). In addition, performance requirements are needed to specify the degree to which the design or construction activity should be enhanced or made more productive through the application of computer aids. New measures will be needed to ascertain such performance, as well.

Increasing productivity in the building industry will require a high level of portability for all types of software because of the fragmented nature of the construction community and the rich variety of computer hardware. In 1975 it was estimated that 1,836 organizations produced software for civil engineering applications alone [9]. The same study questioned the value of much of this software, in terms of its universal transferability. Since that time, however, a significant step has been taken with the introduction of a new standard for FORTRAN, the most widely used language in the engineering design field [3]. Such standards provide explicit measures of portability performance of individual software packages. Other, and newer, computer programming languages and types of data may eventually be used in the construction industry, and similar measures of portability performance will be needed when they appear.

The feasibility of an integrated project information system to maintain the database for the typical construction project depends to a great extent on a capability to measure several qualities of communications performance between those actors and the database itself. Similarly, measures of the quality of communications are needed for evaluating various computer-aided design systems. All these communications will tend to be dynamic, changing previously stored information: measures of the efficiency (speed and redundancy), reliability (propagation of errors), and basic format for data handling will be needed.

CBT long has been concerned with characterizing and measuring critical aspects of buildings which affect their usefulness, safety, and economy. Other components of NBS are concerned with specifying the performance of software and automation systems. The characterization of elements comprising computer-integrated construction technology and the specification of performance required of these technologies seem well within the grasp and mission of NBS research teams.

NBS-114A (REV. 2-8C)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET (See instructions)	1. PUBLICATION OR REPORT NO. NBSIR 83-2671	2. Performing Organ. Report No.	3. Publication Date March 1983

4. TITLE AND SUBTITLE

The Standards Interface for Computer-Aided Design

5. AUTHOR(S)

Fred I. Stahl

6. PERFORMING ORGANIZATION (If joint or other than NBS, see instructions)

NATIONAL BUREAU OF STANDARDS
DEPARTMENT OF COMMERCE
WASHINGTON, D.C. 20234

7. Contract/Grant No.

8. Type of Report & Period Covered

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (Street, City, State, ZIP)

same as item 6

10. SUPPLEMENTARY NOTES

☐ Document describes a computer program; SF-185, FIPS Software Summary, is attached.

11. ABSTRACT (A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here)

Building quality can be improved and building costs reduced through more effective computer utilization in design and construction. To accomplish these objectives improved interfaces are needed between building project databases and computer-based procedures for analysis and design, and between computer-based engineering procedures and applicable design standards. This latter task involves a set of problems termed the Standards Interface for Computer-Aided Design (SI/CAD). These problems comprise the focus of the current report. The SI/CAD is shown to be a critical determinant of computer-aided design (CAD) system effectiveness, particularly in the domain of structural engineering design. This report examines the hypotheses that: (1) the ability to easily maintain design standards data is fundamental to CAD system effectiveness; (2) the configuration of presently available computer-aided structural design (CASD) system software inhibits efficient design standards data modification, requiring costly maintenance to avoid software obsolescence and limiting the overall usefulness of these systems; and (3) methods to enhance the efficiency of criterion checking and standards data maintenance are required to increase the utilization of CAD technology. Support for hypotheses (1) and (2) is developed from anecdotal engineering experience and from the technical literature drawn principally from CASD. No evidence was found to support hypothesis (3).

12. KEY WORDS (Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons)

Building codes and standards; building delivery process; building design process; computer-aided building design; computer-aided design; computer-integrated construction; engineering database management; structural engineering computer programs

13. AVAILABILITY

☒ Unlimited

☐ For Official Distribution. Do Not Release to NTIS

☐ Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402.

☒ Order From National Technical Information Service (NTIS), Springfield, VA. 22161

14. NO. OF PRINTED PAGES

48

15. Price

$8.50