An approach to design knowledge capture for the Space Station

D. B. Wechsler

The MITRE Corporation, 1120 NASA Road One, Houston, Texas 77058

K. R. Crouse

NASA, Johnson Space Center, Mail Code EF5, Houston, Texas 77058

Abstract

Design of NASA's Space Station has begun. During the design cycle, and after activation of the Space Station, the reoccuring need will exist to access not only designs; but also deeper knowledge about the designs, which is only hinted in the design definition. Areas benefiting from this knowledge include training, fault management, and onboard automation. NASA's Artificial Intelligence Office at Johnson Space Center and The MITRE Corporation have conceptualized an approach for capture and storage of design knowledge.

Background

NASA's Space Station Program (SSP) is presently completing its System Design Review milestone. With a program preliminary design, NASA will move forward to detailed design, testing, evaluations, and launch. First element launch is planned for 1992. The environment for generation and capture of design information over this period is characterized as follows:

a) Continuing engineering efforts will result in many changes. Modifications in configuration are expected to occur.

b) Many personnel will phase in and out of the program

c) Technology developments will bring opportunities to apply design information in powerful and hardly-imagined ways.

The challenge of design knowledge capture is to create and populate a base of design knowledge, using approaches which are realistic within their implementation timeframes; but which will provide a sufficient informational foundation to support applications of the year 2000, and beyond.

Needs and benefits

The capture, storage, and availability of design knowledge can benefit the Space Station Program throughout its life cycle. The need for this capability has been indicated in the following representative areas.

Support for continuing engineering analysis

As the development of the Space Station progresses, engineering models will be exercised with complex "what-if" simulations. If the root values and inputs are retained, together with a definition of the engineering analysis description, then the same or similar analysis types can be reproduced using different parameters [1]. If the same analysis program is available, the original analysis can be re-created.

The importance of integration through a common database is emphasized, for "team engineering" approaches to reducing product design lead time. In a team engineering environment, the work of an individual will affect many others on a project. As the size of the team increases, cooperative interaction leads to productivity improvement, while standalone solutions reach a point of diminishing returns, due to the overhead costs of redundant communication [2].

Support for manufacturing

The integrative benefits of design knowledge capture are embodied in the term "CIM" (Computer-Integrated Manufacturing), adopted in principle by industrial corporations worldwide.

CIM is defined as a concept involving intelligent combination and use of "... computer and information/ communication technologies, to effectively integrate all of:

a) the engineering/design functions,
b) the manufacturing planning functions,
c) the equipment/process technologies,
d) the manufacturing control processes, and
e) the management functions,

necessary to convert raw materials, labor, energy, and information into a high quality, profitable product, within a reasonable amount of time." [3]

The Corporate CIM Committee at Garrett Corporation has observed that the manufacturing organizations operating in 1995 and beyond will be fundamentally different from today's typical

manufacturing company [3]. Donald Manor, Division Manager of Deere and Company, furthers this view [4]:

> "The face of manufacturing is changing at an incredible rate... And because of these profound technological changes, manufacturers must develop well-defined long-range plans with a full understanding and commitment to CIM. Those firms that are unwilling or are unprepared to make such forward-thinking commitments will simply not survive."

Support for logistics and field operations

In its introduction to the Integrated Design Support Project, the U. S. Air Force Logistics Command reports [5]:

> "In its wake, high technology has created massive amounts of technical information-- a mountain of paper which today must be managed manually... engineering technical data is volatile, complex, iterative, and addressable by a variety of applications. These special requirements make manual data-handling extremely labor- and cost-intensive. To date, application [of CAD/CAM/CAE] has been focused on design and manufacturing, with no consideration being given to integrating the overall... life cycle process."

Support for on-board applications

The initial findings of NASA's Automation and Robotics Panel stated that [6]:

> "... Shuttle operators rely heavily on paper backup for every mission. This mass of documents must be condensed, coordinated, and unified into a usable database if the Space Station is to reach its planned level of capability."

When the actual behavior of a system fails to match its intended behavior, the reason for failure is more easily localized if a record is available of how the system specification was decomposed and implemented [1]. However, this debugging cannot be conducted using the design definition only. The designer's knowledge must also be applied.

Support for future designs

Captured and retained design knowledge can provide a basis for solving new design problems, by using design rationale to replay

design histories of similar problems. If a goal in the old problem was met using some plan, and the reasons the old design worked also hold true in the new problem; then the plan can assist in determining the new solutions as well [1].

Current state of design knowledge capture

Definition of capture

Capture is the process of obtaining data for retention in computer-interpretable form. Captured information is organized in meaningful context for retention and use.

Information currently held in manual media is not considered as captured. Neither is data which was read into storage without association of meaning, as with the case of scanned code, or unidentified text strings. Additional conversion or interpretation of these data forms will be required.

Impediment to capture

Short-sighted emphasis of Computer-Aided Design (CAD) as a drafting tool can prolong resistance to the effective organization of design definition data. Chester Fleszar, an Applicon Marketing Product Specialist, elaborates [7]:

> "The problem is that we act as if we're making blueprints rather than parts. Once a company becomes involved in CAD/CAM, the electronic representation of the part becomes all-important, while the blueprint becomes obsolete. Translating all the information on the original paper drawing does nothing to improve manufacturing quality or efficiency..."

Summary of condition

With increased use of CAD/CAM, the electronic retention of design definitions is on the increase. The manufacturing community has recognized the benefits of integrating their applications through the use of design definition information.

But efforts to capture the designer's knowledge are rare. Design knowledge is volatile. Unless it is captured, it will be available only to the extent that the designer is accessible. For customers, logistics organizations, and field operators, the design definition in itself is inadequate. This condition results either in their attempts to manipulate the definition information, or in their efforts to obtain additional information to suit their particular needs.

Elements of design knowledge

Overview

Design knowledge encompasses not only what designs are, but how and why they satisfy the functional requirement [8]. Design knowledge is represented as a linked design object structure, composed of design objects, object attributes, and declarations or assertions called "designer's knowledge".

Design object structures

The principal unit of design knowledge organization is the design object. The linkage of design objects defines the design arrangement. Design objects are defined over a range of abstraction levels, in which each object is linked with its constituents. For example, a representation for a physical design object of a component assembly is decomposed to the constituent components. The components are decomposed to constituent features of each component. The features can be implemented as graphical elements, if the feature has a graphical interpretation.

Each design object accommodates its attributes: the definitions of its own characteristics.

For the SSP, the Reference Configuration to be provided at the conclusion of NASA's Preliminary Design Phase can serve as an initial, high-level set of design objects.

Graphics of the design object

The U. S. CAD/CAM community has in cooperation defined a computing system-independent means of exchanging CAD graphics files. This evolutionary standards effort, coordinated by the National Bureau of Standards, is called Initial Graphics Exchange Specification (IGES) [9].

The IGES approach defines standard data structures called entities, for geometric and other graphics-based elements. A graphics system that supports IGES can translate an IGES data structure into a geometric pictorial.

The IGES data structures corresponding with the design object's geometric elements are attached to the object, at the highest applicable level of abstraction. These data structures are represented as attributes of the design object. This accommodation allows a design object possessing this information to "draw itself", using methods including IGES translation by the CAD delivery system [10].

Designer's knowledge

Design objects are the parent representations for designer's knowledge. Designer's knowledge attaches to the design object to which the knowledge refers, at the highest applicable level of abstraction.

Designer's knowledge includes the information the designers used; the analysis they conducted; and the decisions they made to develop the design object. Designer's knowledge defines what the design does, and why it does so. Such definition is dependent upon functional and behavioral descriptions [11].

Examples of designer's knowledge include declarations of functional requirements for the design object; criteria or intent for selection of a design approach or solution; declarations of analysis results and conclusions; and assertions concerning expected design object behavior.

Bounding of design knowledge

The potential size of a comprehensive SSP design knowledge aggregation demands that approaches be defined to eliminate the capture and proliferation of unnecessary knowledge. Three parameters for knowledge organization and bounding have been identified.

Bounding of knowledge content

The "perspective" parameter determines a category for design knowledge, by requiring that only knowledge be captured for which a use can be identified. Uses are defined in terms of analytic disciplines or problem classifications called "users' views".

Perspective allows for the retrieval of design knowledge, according to users' views. A view can be based on an intermediate design, such as required for an engineering analysis [12]; or on an implemented design, such as for on-board SSP applications.

Since complex future problems might involve several disciplines, the use of perspectives will serve to clarify the boundaries of expert knowledge in multi-discipline problem-solving.

Bounding of knowledge volume

The "visibility" parameter of a design object can be used to determine how much design knowledge to capture. Visibility can indicate the depth of knowledge detail to be captured, and aid in selection of design knowledge capture tools to be employed.

Visibility is a combined valuation of the probability of failure within the design object, factored with the results of such failure. This valuation can be taken from reliability and redundancy projections. For example, if a design object, with consideration of its design redundancies, is determined to have a high reliability, and least serious results from failure, then this object has a low visibility rating.

Bounding of capture frequency

The design knowledge "rolling baseline" will evolve towards complete design knowledge over an extended period. Although the state of the design knowledge will be "snapshot" for delivery at major SSP milestones, engineering analysis will continue in progress [13].

The "version" parameter establishes temporal support in the design knowledge structure, to accommodate multiple versions of design object structures, together with the associated source knowledge for each. Newly added or modified knowledge will include an accurate accounting of changes and their rationale.

Design knowledge system components

Relational database storage

Captured knowledge is stored in relational databases. Relational storage technology will be available to both NASA and the Space Station contractor community. The facilities of the database manager are then available as a capture tool. Additional capture tools are described in the following paragraphs.

For applications having human interfaces, the database management facilities of the relational storage system can be used for knowledge query and reporting. For applications with fully-automated interfaces, transformation of the database content is required.

Capture tools

CAD and engineering analysis. As CAD provides for automation of the graphical representation process, Computer-Aided Engineering (CAE) provides for automation of the engineering analysis. Engineering analysis methods are used to predict the behavior of the design object. The Computer-Aided Engineering (CAE) analysis perspective becomes a knowledge bounding parameter.

In integrated CAE, the output of the CAD system is recognized by the CAE program. The integration of CAD and CAE provides a common view of the information, which facilitates reasoning and analysis of the design object, in terms of its expected behavior.

CAD/CAE documentation. Basic documentation can be originated from within the CAD system. Attribute values are entered when the graphical representations of the design objects are created. These object-attribute structures are then extracted. The extracted information is loaded to the database system.

With the addition of integrated documentation, the results of the CAE analysis can be included with the CAD information [14]. The analysis program is easily identified. Further, the program itself can be retained for future use.

Specification Language System. A Specification Language System (SLS) is a specification development environment based on a formal notation for expressing design requirements, in terms of function, structure, or behavioral descriptions. The methodology of the SLS provides an organized approach for capturing design knowledge early in the development cycle, when insufficient system design definition exists to apply Computer-Aided Design (CAD) approaches.

The automated tool set of an SLS can provide valuable assistance in assuring consistency of identifiers and terms, and in enforcing documentation and project standards. The following SLS functions are supported with automated tools [15]:

- Data flow checker
- Formal specification language
- Interactive graphics
- Program design language system
- Simulation/prototyping
- Static analyzer
- Specification tree
- Traceability analyzer

Where well-defined relationships exist between the functional and physical definitions, certain SLS's synthesize and simulate physical structure, given a functional definition [16, 17]. An example in VLSI circuitry is the VHSIC Hardware Description Language (VHDL) [18]. Executable hardware description languages should be considered only for well-structured design problems, in which the physical implications of the functional description are unambiguous.

Designer's apprentice system. The designer's apprentice captures design knowledge as a by-product of its interaction with the designer, in the role of an intelligent design aid [19].

The designer's apprentice performs the following functions:

- Suggestion of goals and constraints, using an inferencing mechanism to process the design rule base
- Recognition of past successful solutions
- Conducting and recording of dialogues with the designer, through an explanation facility
- Assistance with tedious details

Although designer's apprentices now exist and hold considerable future promise, the design process is barely understood well enough to effectively apply this tool. For the present, the strongest candidate application areas are narrow domains of expertise, in which design rationale is codified. Since such areas are often likely to involve routine design, attempted justifications would be based upon productivity improvement. The benefit of the designer's apprentice as a reasoning aid would be ignored.

Technology for future knowledge access

In this section, emerging software technologies are described, which future systems will employ to access design knowledge from the relational database.

Programming in Logic (PROLOG) [20]

Because PROLOG itself is a software environment, it does not require a development shell. This environment contains generally accepted rules of inference as the "inference engine".

In PROLOG, the same set of statements can be used both to declare a structure, such as a part breakdown hierarchy; and also state relationships of cause and effect. PROLOG creates its own application-specific data structure. The goal statement, which drives the execution of PROLOG's inferencing, could be described as a command to query PROLOG's knowledge base according to the specific application.

The task of the PROLOG application developer is to provide the problem goal and description. For example, if a device design requirement goal is provided, together with descriptions of alternative device designs; then PROLOG can select from among the described devices those which meet the goal conditions, without being explicitly programmed for search.

Object oriented programming [21, 22]

Object-oriented programming complements design knowledge organization by characterizing systems in terms of a configuration; centering descriptions around the objects that are pieced together, rather than centering on transformations of data about these systems [23]. This approach to organization is similar to the previously-described linked design object structures in the database system.

Object-oriented programming holds promise for greater use because of claims for improved programmer productivity and easier program maintenance. More important from a user viewpoint, the program organization allows those not initially familiar with the program to rapidly and accurately understand its content.

An object consists of data private to the object, and of a set of operations which can access that private data. A "consumer" object requests a "provider" object to perform one of its operations, by sending it a message telling it what to do. The provider object responds by choosing and executing the operation, and returning control to the consumer.

Database-inferencing systems

The bridging needed between expert system knowledge bases and database management systems is receiving increasing attention. Although it is now possible for knowledge-based applications to initiate a database query [24], the time to search large databases can severely limit application size.

A promising current approach involves schema translation. In comparing programming language commands with database operators and query commands, researchers have developed a mapping of schema between the two [25, 26].

Schema translation will lead to development of database-inferencing systems which share schema. These systems will use a common database for a number of knowledge-based applications. The inferencing procedure will be integrated with the database management capabilities. Application development will consist of developing the appropriate goal statements, and confirming that the supporting descriptions are in the database.

Accommodating knowledge access technologies

Applications based on these advanced technologies are not now in wide use. For an interim period, design knowledge will be retained in relational storage, in a non-executable, object-oriented form. When these applications become established, they will be supported by captured SSP design knowledge, made compatible through economically tolerable modification.

Future tasks and issues

Overview

Development of a design knowledge capture system is a project which involves substantial planning and preparation.

A serious implementation issue is the coordination of timing between capture system installation and Space Station development. Until capture is implemented, the risk of losing designer's knowledge is ongoing [6].

Many of the facilities for design knowledge will be provided as part of NASA's Technical Information Management System (TMIS). The TMIS will be used to support technical management functions of the overall Space Station Program, including the design, development, and operation of the orbital facility. The TMIS user community will include all NASA personnel involved with the Space Station, all primary contractor personnel, and all personnel representing the international partners. The TMIS resources will be based on commercial,"off-the-shelf" technology.

Following are major near-term tasks for implementation of design knowledge capture:

Design knowledge computing facilities

Relational database facility. A TMIS-compatible relational database facility will be employed as the development contractors' design knowledge repository. Adequate description of this facility will be provided in ample time to allow for development contractors' knowledge capture planning.

Standardization issues. Standardization issues will be resolved, which arise from the resources to be provided. Such issues include common methods for CAD data exchange.

Design knowledge organization

Schemes for knowledge bounding. To support the development contractors' planning for capture resources and methods, initial valuations of visibility parameters will be provided for identified design objects. A classification of engineering analysis perspectives will be supplied. A common contractor approach for implementation of knowledge versioning will be defined.

Design knowledge content. The design knowledge base must be defined and organized, before it can be populated. Guidelines will be established for common semantics and input definitions. Available application developers will assist by providing requirements.

Existing NASA databases will be evaluated for compatibility with requirements for design knowledge content. Conforming portions will be integrated within a design knowledge context.

An evaluation of the planned content of future milestone deliverables [27] will also be conducted, for suitability as design knowledge.

Conclusions

The benefits of design knowledge availability are identifiable and have considerable breadth. The implementation of design knowledge capture and storage using current technology increases the probability for success, while providing for a degree of access compatibility with future applications. The Space Station design definition should be expanded to include design knowledge. Design knowledge should be captured. A critical timing relationship exists between the Space Station development program, and the implementation of this project.

References

1. Mostow, Jack, "Towards Better Models of the Design Process," AI Magazine. Spring, 1985.

2. Sullivan, Jerry S. and Rummler, David C., "Second-Generation CAE Tools Learn to Share One Database," Electronic Design. July 10, 1986.

3. Mize, Joe H., Seifort, Deborah J., and Settles, Stan F., "Formation and Workings of a Corporate-Wide CIM Committee at Garrett Corporation," Industrial Engineering. November, 1985.

4. Autofact Alert, Computer and Automated Systems Association of the Society of Manufacturing Engineers (CASA/SME) 1986.

5. Integrated Design Support, U. S. Air Force Wright Aeronautical Laboratories (Flight Dynamics Laboratory) 1986.

6. Advanced Technology Advisory Committee (Aaron Cohen, Chairman), Advancing Automation and Robotics Technology for the Space Station and for the U. S. Economy (NASA Technical Memorandum 87566). April 1, 1985.

7. Fleszar, Chester, "Is IGES the Problem, or are We?," CIM Technology. Fall, 1986.

8. Crouse, K. J. and Spitzer, J. F., "Design Knowledge Bases for the Space Station," Proceedings of ROBEXS '86. June, 1986.

9. IGES Organization, Initial Graphics Exchange Specification- Version 3.0, National Bureau of Standards. April, 1986.

10. Beazley, William G., Available Expert System Knowledge In CAD/CAM Databases, The MITRE Corporation Contract F19628-86-C-0001. July, 1986.

11. Walker, Robert A. and Thomas, Donald E., "A Model of Design Representation and Synthesis," Proceedings of the IEEE 22nd Design Automation Conference. 1985.

12. Brown, D. C. and Chandrasekaran, B., "Expert Systems for a Class of Mechanical Design Activity," Working Conference on Knowledge Engineering in Computer-Aided Design, Elsevier Science Publishing Company 1984.

13. Snodgrass, Richard and Ahn, Ilsoo, "Temporal Databases," IEEE Computer. September, 1986.

14. Steinke, George C. and Schussel, Martin D., "Engineering by the Book... and On-Line," Mechanical Engineering. November, 1985.

15. Addleman, David K., Davis, Margaret J., and Presson, Edward P., Specification Technology Guidebook, U. S. Air Force Contract F30602-84-C-0073. August, 1985.

16. Genesereth, Michael R., "The Use of Design Descriptions in Automated Diagnosis," Qualitative Reasoning About Physical Systems, The MIT Press 1985.

17. German, Steven M. and Lieberherr, Karl J., "Zeus: A Language for Expressing Algorithms in Hardware," IEEE Computer. February, 1985.

18. Ackley, Dave, et al., "VHSIC Hardware Description Language," IEEE Computer. February, 1985.

19. Rosenfeld, Lawrence W. and Belzer, Avrum P., "Breaking Through the Complexity Barrier--ICAD," Proceedings of Autofact '85. November, 1985.

20. Genesereth, Michael R. and Ginsberg, Matthew L. "Logic Programming," Communications of the ACM. September, 1985.

21. Cox, Brad J., Object-oriented Programming, Addison-Wesley Publishing Company 1986.

22. Stefik, Mark and Bobrow, Daniel, "Object-Oriented Programming: Themes and Variations," AI Magazine. Winter, 1986.

23. Clancey, William J., "Heuristic Classification," Artificial Intelligence Journal. December, 1985.
24. Sowa, J. F., Conceptual Structures: Information Processing In Mind And Machine, Addison-Wesley Publishing Company 1984.
25. Boas, Ghica van Emde, and Boas, Peter van Emde, "Storing and Evaluating Horn-Clause Rules in a Relational Database," IBM Journal of Research and Development, January, 1986.
26. Rehak, Daniel R., Howard, H. Craig, and Sriram, Duvvuru, "Architecture of an Integrated Knowledge-Based Environment for Structural Engineering Applications," Working Conference on Knowledge Engineering in Computer-Aided Design, Elsevier Science Publishing Company 1984.
27. Anderson, Sandra V., Bell, John B., and Smylie, Susan E., Information Asset Management (Volume IV), The MITRE Corporation Technical Report Number 8613. June, 1986.