CAD/CAM DATA MANAGEMENT: SOLIDS MODELING APPROACH TO ENGINEERING DATA MANAGEMENT

Dr. Harold U. Nwosu

Abstract

In engineering design, one deals with large sets of independent but interrelated objects. These objects are specified by data. One way to store and organize these data items is through the use of a database management system. For an engineering design database, the system must be able to not only effectively manage the design data, but also model the objects composing the design. The database management system needs to have some knowledge of the intended use of the data, and must provide an abstraction mechanism to represent and model the objects. Solids modeling techniques offer great promise for *CAD/CAM.* To enhance the productivity benefits associated with *CAD/CAM* using solids modeling, unique developments have been undertaken to integrate solids with existing software capabilities. This interactivity allows the *CAD/CAM* user to perform productively tasks previously considered beyond the scope of interactive graphics. This paper discusses solids modeling approach to engineering data management capabilities and requirements.

Introduction

Computer-Aided Design (CAD) and Computer-Aided Manufacturing (CAM) systems create and use a variety of data for such purposes as parts design, manufacturing, and management. Although, the integration of such information is desirable, the complexity that arises from the sharing of *CAD* data by both applications and users requires the use of data base management tools. To provide the best possible computer assistance for users in design, engineering, and manufacturing, database management for *CAD* systems must be specifically oriented to handling design objects, assisting in moving objects between application phases, and supporting adhoc information needs throughout a project.

Computer tools have made possible the design of exceedingly complex products. As system complexity grows, so do customer demands for reliability and maintainability. As such, the design of the product must be closely coupled with the design of the processes to manufacture, test, and maintain it. This means getting diverse teams of engineers working concurrently on the problem. These engineers may reside at different locations or even in different companies. Using a database to store and describe engineering data offers many benefits (Teorey, Yang, and Fry, 1986):

1. Ability to store or access data independent of its use.
2. Ability to represent (complex) relationships among the data.
3. Control of data redundancy.
4. Maintenance of data consistency and integrity.
5. Enhanced development of application software.
6. Support of file manipulation and report generation.

To maximize these benefits, a database management system, besides effectively managing the data, needs to have some knowledge of the intended use of the data. That is, the formal structure or model used for organizing the data must be capable of depicting the' important relationship among the data, and must be able to facilitate the maintenance of these relationships. Furthermore, the structure should be sufficiently flexible to allow a variety of design sequences and aid an engineer to understand the design.

Traditional relational database systems provide many interesting features for managing data; among them are the capabilities of set-oriented access, query optimizing and declarative languages. More importantly, from the user points of view, the relational model is completely independent of how data are physically organized. The relational model presents data item as records (tuples) which are organized in 2-dimensional tables (relations), and provides a manipulation language (relational

calculus) to combine and reorganize the tables or relations for processing. The relational approach is simple and effective, particularly for business record-oriented data processing.

The relationships among the data items describing an engineering design are often complex. The manageability of a complex application, which has too many relevant details to be intellectually represented, can be simplified be decomposing the model into a hierarchy of abstractions. Abstraction concepts are used so that the unnecessary details can be suppressed; only that information pertinent to the problem is emphasized. Focusing on a few selected important pieces of information enables an engineer to deal with design entities and their relationships effectively.

The design process is very complicated and differs with each engineer. The development of computer-based aids for design must take this into consideration and allow for the nuances of the individuals. A generic representation of the information used in design is required for the development of any database or supports the design aids. A properly designed database or set of database can help integrate *CAD,* expect system, and ancillary programmes, thereby forming an intelligent design system. A database for an intelligent design system can be developed based on these results.

In recent years, there has been a great deal of research emphasis on aiding the mechanical design process with computers. Mechanical designers spend the majority of their time attending to design details: sizing shafts, selecting motors, specifying bearings, etc, and a relatively small amount of time generation alternative designs in creating new ideas for products.

Design productivity must be enhanced before Nigerian companies can successfully compete in the expanding world market, where low cost and high quality are the state of the art. Manufacturing productivity has improved many times over since the turn of the century but design productivity has increased only slightly. This deficit creates a larger opportunity for the advancement of design. The reasons for this design productivity lag are known about the design process. As a result, there is no objective way to evaluate the design process since the mechanical engineer’s design process is complex and poorly understood. One factor which makes the mechanical design components which perform the same function from which the designer can choose.

Computer-aided support for electric circuit design has been attempted with promising results. Typical electric circuits use fewer than ten types of components, each of which can be described using only a few parameters (Ward and Seering, 1987). The mechanical design process is far more complex, which makes the development of computer assistance more difficult. The design of mechanical systems involves component selection from “hundreds of types of available components, and some may require thirty or more parameters for specification. Many of these components may perform the same function, thereby greatly compounding the difficulties.

To overcome this design productivity lag, intelligent *CAD* workstations must be developed that aid the designer in both the design of novel parts and the selection of standard or purchased components. Such aid can only be implemented through the integration of drafting packages, expert systems, and databases, thereby creating next-generation *CAD* systems and increasing a designer’s productivity.

The layered object model overcomes another problem of integration. The integration of *CAD* packages, expert systems, and databases requires that a standardized communication data structure be implemented. A modular system which utilizes a interchangeable software needs. Thus, a quantifiable and complete information needed by system components in a standard, organized manner.

Thus, a generic model which represents design information is key to system development and overcomes integration difficulties. However, unless the model interacts with both the design process and the supporting database, it will be useless. Any model that is developed for systems which aid design must incorporate an information structure that supports integration as well as interacts with the design process and domain heuristic. The layered object model incorporates this structure by interacting with the design process, the database, and design for manufacturing guidelines.

Data Sharing

CAD database management must also address such project and design management concerns as controlling shared *CAD* data. The teams involved in a project may need to download part of a design object from a central storage location to a workstation to perform design work. The design objects must be tracked throughout the design cycle and eventually merged back into a central storage location, which requires mechanisms for controlling the reentry of the design objects. As a means of controlling checkout from central storage, a copy of a design may be issued to team members and used later to check in approved design changes.

During the design process, team members tell one another about the data. Because several teams may check out a design for different projects or reasons, a mechanism is needed to notify the design users about the status of all the copies. In the absence of alternative control, changes from several teams must be validated before they are posted against the master copy in central storage.

Data Recovery

Recovery issues arise any time there are changes to data, whether to a copy for reworking or to the data in a common repository. Updates to a database require a mechanism for backup or logging of transactions. In the event of a system problem, a backup copy would be used to restore the data. Many business-oriented transaction systems track transaction activity on a separate medium. One common method is to track transaction activity by recording the before and after images of the database record. This recorded data can then be used to restore the database to a previous state. The large amount of data used in *CAD*, however, may make such a logging scheme very inefficient.

Integration

CAD database management varies by vendor, project, and application. *CAD* systems that reside on large time-shared computer systems or on several compatible systems from the same vendor can ease the integration. The widespread use of microcomputers, workstations, and several different and generally incompatible specialized processors to support various specialized *CAD* functions has complicated integration and highlighted the need for standard interfaces and capabilities.

A *CAD* system with workstations of various machine types and in different locations requires both public and private copies of design object data. The public copy, which resides on the department or host system, can be read but allows only limited updating. The private copy, which resides at the workstation, can be manipulated at will. When the design or analysis is completed, the data is validated and checked in at the central repository. Each server -workstation department system or host system -may have different data manipulation capabilities and representation formats, which require data transformation or, when the differences are great enough, complete reentry of data to move an object from one server to another.

File Transfer

Although in many cases file transfer is sufficient for moving data between machines, the problem of translation still exists. The lowest level of translation is between different representations of data types, especially with vendor-specific bit methods. Bit order or even the representations that results from the machine architecture can be a problem. A common interchange format helps but may increase redundancy and processing time. Another issue is naming standards for objects, especially between many applications and types of systems. Finally, one problem is difficult to solve: it may be possible to generate a *2D* version of a *3D* object, but the reverse operation would require adding more information to the *2D* version. In other cases, it may not be possible to use the *2D* data at all, and the *3D* version may need to be manually generated.

CAD Application and Database Management Requirements

CAD systems place special requirements on database management. For example, the graphic representations of objects require the support of complex data structures and the ability to perform several types of operations on the objects. Because an object may be a composite of other objects as well as part of another object, this hierarchical structure requirement further complicates the data and data management needs. *CAD* also requires that objects be acted on in their entirety -for example, when a designer is moving a subassembly in a composite view of a design for display, comparing two versions of a design for improvements, or using an older, comparable design to start or enhance a new one. The last is a time-and cost-saving method for developing new products that could benefit from the use of a data manipulation language. *CAD* graphic data must support multiple functions, including plotting blueprints or displaying a design object at a graphics terminal, and support multiple functions, including plotting blueprints or displaying a design object at a graphics terminal, and support multiple representations of an object. For example, in electrical circuit design, both the structural and behavioural aspects of a design must be supported -that is, how the design looks, especially in layout, and how it works. A *CAD DBMS* should also provide a means for maintaining consistency between multiple representations, this need has resulted in such standards as *IGES* (initial graphics exchange specifications), which provides an intermediate format for moving objects between the hardware of various vendors.

One major problem in *CAD* database management is transferring data from one phase or application to another. Throughout the design, engineering, and manufacturing processes, various users require support for obtaining or using information about a product. Programmers frequently write code to obtain reports from the *CAD* database, but a data manipulation language oriented to adhoc information needs would support applications programmers and users by providing both higher-level and more powerful object-level commands.

Basically, data management is done at the application level, but the diversity of application data views and machines used indicates that it must continue to address lower-level data-handling issues. With effective data management, applications could be written with a view exclusive of its machine aspects. A standard relations language (e.g. *SQL*) is one trend toward providing a common interface to the applications to assist with data handling.

The Layered Object Model

In the mechanical design world, there is a vast amount of information specified during the design of part or an assembly. Information is needed in a certain format and at various levels for a given design process state. It is difficult to capture all of the needed information, especially when revision after revision of a design is made. A checklist of information which describes the needed information to the designer, regardless of the present design state or level in the manufacturing or design process, would greatly facilitate his/her work.

The layered object model identifies the minimum set of information needed at these various states. It not only shows the information and its location in the sub-assembly or part, but also the relationships between it and other information at different levels in the model. This feature provides the designer with a better understanding of how and why information is generated for a given design, as well as how to abstract this information from the design. The object model has meaning by itself because it represents design information in a structure that captures both information and relationships. However, the model cannot stand-alone. It is tied to the outside design and manufacturing worlds by a series of input/output points. These points are represented in the model diagram as arrowed lines which originate at entities and terminate at either design-state or manufacturing blocks.

The design process interacts with the object model through what are termed design states. These states correspond to the level of the design process at any given time. Similarly, manufacturing interacts with the model at different levels and requires certain information abstracted from the part's geometry, material, function, etc. Thus, the importance of this, layered object model is that it represents not only the minimum amount of information needed from a part to ensure manufacturability and aid in design, but also its relationship to other information in the model. This information exists at different levels associated with the level of abstraction of the product: the sub- assembly, part, and feature levels.

Manufacturing Interaction

In order to aid the engineer with the design of novel parts, manufacturing guidelines interact with the layered object model at various levels and with various objects within the levels. The following discussion explains these interactions. Manufacturing consists of two major categories which interact with the object model: materials processing and assembly. Of course, manufacturing encompasses more than these categories, which have been chosen as representative.

Materials processing involves the production of features. To determine if a part can be manufactured, one must analyse the part's geometry, feature's function, and material in light of the potential manufacturing operations. This interaction between the materials procession operation and the object model reflects the multi-layered abstraction required for complete information from several layers of the model for operation. The interaction spans both the part and feature levels, but does not require information from the sub-assembly level, because all of the information needed for process determination is located at the feature and part levels.

For example, features have a specific set of manufacturing processes available for their manufacture; yet, the material chosen for the part and the spatial relationship of features may impose limits on these operations. Assembly analyses the fastening of parts by various connection methods. The assembly category requires information about the part geometry, including the features, part materials, spatial relationships of parts, as well as the chosen connection methods, to successfully complete an analysis. For the successful analysis of assembly, the object model needs to supply information from the sub-assembly and part levels.

Design Process Modeling Implications

The layered object model interacts with the design process through individual design states that resides at each layer. The design states determine where the designer is within the design process and at what information, a design controller can direct the engineer through the design process. The design process is based on the information presented to the designer, as well as the information she/he generates. Information is unique at each level within the model. In other words, at any given level, information known by generated due to some actions at level can be identified as an output of that level. Thus, through the identification of this information flow and generation, the design process can be modeled and given to the designer.

At each level of the layered object of the layered object model, information is presented along with its relationship to other information. Information passed to a level resides there before the designer initiates any action; other information is generated within the level as a consequence of design and is passed to the other levels. For example, at the beginning of the design process, functional requirements and constraints are passed from the external world to the product level. This information and other information generated at the product level is then passed to the sub-assembly level and similarly to the part level. Once a part is in the conceptual design stage, functional information generated at the part level is considered at the feature level. Features are selected and featural information is passed to the part level for assimilation. This information flow begins as a top- down propagation beginning with the product level. Once at the feature level, it becomes a bottom-up propagation ending at the product level. Information propagates between levels along information channels. It is necessary to note that some of the information, although generated at one level, has no meaning until it is assimilated into the level to which it is being passed. For example, some featural information such as spatial relationships of features on a part, although generated at the feature level, has no meaning until assimilated at the part level.

Why Have Solids Modeling Techniques Evolved?

Solids modeling has come to be viewed as the solution to all the *CAD/CAM* users needs. Much has been written concerning the benefits to be realized with solids modeling techniques, but most of the implementations to date have not successfully addressed the requirements of complete process engineering. One of the great benefits of solids modeling systems is their ability to define an unambiguous mathematical definition of a real world part. This ability can be used to provide product definition data which is instrumental in describing the complete description of a product, including all process and administrative information. When this product definition information is stored and managed by a Relational Database Management System *(RDMS)*, it can provide a powerful tool in an organization’s attempt to control the product life cycle.

A clear definition of how solids modeling evolved will help describe the benefits obtained and problems that are solved. Basically, interactive graphics was developed as a tool to allow the designer or engineer to see his data in a graphical rather than alpha-numeric mode. Prior to this time, any geometric data utilized on a computer system was input, operated on, and output in alpha-numeric, or coded form. By being able to actually visualize this information graphically, the user could greatly improve the quality of his work and the speed with which he could perform it.

These early systems were basically two-dimensional and only displayed primitive graphics, such as lines, points, or arcs. It did not take long for these systems to expand and allow the input of actual three dimensional information. This data was still rather primitive and the resultant graphics images were the basic stick figure types that we are all familiar with. This database type has come to be known as “wire-frame” geometry. Wire-frame geometry can also be thought of as the first order of complexity in the definition of geometry models. The types of data created using wire-frame techniques has progressed to the point where today we have the highly refined capacities of complex B-Spline and Bezier curves. Geometry has reached its *APEX* with the Non-Uniform Rational B-Spline *(NURB). NURB* type curves form the basis on which to build the complex modelers of today.

As interactive graphics systems moved out of the world of simple visualization and drafting and into the world of engineering analysis and manufacturing, particularly numerical control, it became critical to develop techniques that could handle the additional complexities required. The second order of geometric modeling, or surfacing, evolved. Once created a *NURB*-based surface can be meshed for finite element analysis, evaluated for area properties, “machined” for output to numerical control, and have practically any other required operation performed with it.

However, geometry models may be thought to possess three orders of complexity. First order, or wire frame elements, represent a stick figure approach to the design. The second order, or surfaces, deal with infinitely thin-bounded sheets which have area properties associated with them. The third order, or solids, introduces completely enclosed shapes, which contain volume, or mass hierarchy, it can be seen that the order of complexity of the model increases. It is this complexity which can describe a mathematically complete unambiguous model that can feed a product definition database.

Engineering Solids Modeling System

Engineering Solid Modeling System *(ESMS)* is an example of the latest in solids modeling technology. *ESMS* combines cost-effective dedicated processing power and an innovative object- oriented environment to offer designers an integrated, full featured tool for generating wire frame, sculptured surface and solid models of mechanical parts and assemblies.

ESMS beings a sophisticated, technologically advanced approach to geometric modeling which serves the current and future needs of design engineers. The object-oriented nucleus makes possible a full-integrated database in which wire frame, surface, and solid elements co-exist with associated non-geometric data, forming a complete engineering model which supports analysis, detailing, documentation manufacturing, and other processes throughout the product life cycle. The system’s object basis builds more intelligence into geometric elements allowing design to perform complex construction and graphic manipulation with high-level commands which support a natural and more productive design process.

The data structure also maintains associative design process. The data structure also maintains associative relationships between geometric elements (connectivity, tangency, etc.) and provides an inherent capability for associative dimensioning. The object base is complemented by an advanced, unified mathematics which delivers extreme precision and consistency of results in modeling operations. The fundamental geometric element in *ESMS* is the non-uniform rational B-spline *(NURB),* ideal for use in geometric modeling because it can exactly represent both complex and simple geometric entities. The *NURB* math structure in *ESMS* is common to wireframe, surface, and solid elements, whether they are basic lines and conics or doubly curved sculptured surfaces.

The system’s solid modeler builds complete and unambiguous geometric models through an approach called Design Process Representation *(DPR),* a hybrid of the Constructive Solid Geometry *(CSG)* and Boundary Representation *(B-Rep)* approaches which combines the advantages of both. Initial construction is accomplished by placing, modifying and combining primitive solid elements. The designer can make both global and local modifications with Boolean operations and high-level commands (e.g., “move face”, “chamfer edge”) which reflect the designer’s natural thoughts process. The resulting edges and faces are automatically computed and stored as B-Splines, along with a record of the operations used to form them. With this representation, the designer has the flexibility to step back in the design process with an “undo” function to make revisions.

ESMS includes a full detailing system to fully and efficiently document models for manufacturing and other requirements. View extraction and sectioning capabilities enable the user to define ad arrange multiple views (including planner sections) for drawing creating, with view-specific control of symbology and hidden line removal display. Automated dimensioning, including an associative dimensioning capability, speeds dimensional descriptions of the model and assures the integrity of drawing data when changes are made to the design geometry.

The system provides a uniform environment for the creation and manipulation of geometric and non-geometric data in "files" common to all product-oriented design, analysis and manufacturing processes. Also, *ESMS* provides a selective interface to other related process automation tools including:

1. Specialized task packages (sheet metal, plastics, etc).
2. Database management programmes.
3. Project management software.
4. Analysis.
5. Manufacturing.
6. Robotics.
7. *MRP,* process planning, etc.

Product Definition Data

There is a tremendous interest in creating and managing Product Definition Data *(PDD)* in the graphics community. A key ingredient in establishing the *PDD* is the use of parametric features, sometimes called "form features". One of the problems in analyzing the needs of form features is to separate the generation of the picture from computer data. Although there is a continuing need to generate drawing the standards investigations are identifying the data needs of downstream processes. This is then used to establish data description requirements. At this time, there is a gaining recognition that the computer cannot reliably scan the drawing (with its text dimensions) to identify the feature.

Features must be stored and described in a standardized manner. It is then the feature's recognized name and its control parameters that will be used by the computer applications. The implementation of a valid *PDD* environment is still very dependent on the data structures of the *CAD* suppliers. A typical implementation would require a "parameter data file" which contains the part programme definition, the parameter descriptions, and the parameter data. The supplier programme must be very smart to process the "part programme description".

Product Definition Data Implementation

The mission of *ESMS* is to facilitate design, engineering, and manufacturing automation. As such, *ESMS* will be capable of creating and manipulating complete Product Definition Data. However, *PDD* is not restricted to geometry alone. One extensive study broke the nebulous term "product data" down into seven slightly less nebulous categories: Geometry, Topology, Tolerance, Feature, Assembly, Non-Shape and Notes, and Administrative. Of course, all this data is related in one way or a lother. A *CAD* system must allow the user to record, apply, and maintain functional relationships ' >etween product data elements. This is the critical component accessory for automation. A brief analysis of these seven *PDD* categories will help to better define how the tools provide a key advantage to the user (Rangan, Whelan, and Fulton, 1998):

1. Geometry: Development of easier and more powerful geometric modeling tools was the original concern of *CAD,* and still consumes a significant portion of the industry 's efforts. The ultimate goal is to allow complete design freedom. This simple means that the geometric modeling capability of *CAD* system must not be a design constraint. The designer must not be a design constraint. The designer must be able to efficiently create a geometric model of any geometric data span the entire product life cycle from concept evolution through engineering analysis, manufacturing, and product support, to reference data for the next product generation. Clearly, the industry has only just begun developing application functionality that taps the value of the model geometry. Hence, geometry will continue to play an important role in future developments.

Because of the unified math basis, *ESMS* has extremely well-integrated wireframe, surface, and solid modeling capabilities. Many other systems have physical integration, in which all entitles reside in the same database. The bottom line result to the designer is true geometry modeling flexibility, accuracy, and consistency that is missing in many systems.

2. Topology: Topology describes how the geometric elements of the model are related in space. In *ESMS,* topology is defined by a generalized composite capability. A composite curve defines connectivity of curves at endpoints. A composite surface defines connectivity of surfaces along edges.

A solid is simply a special case of a composite surface; the case where the surfaces of the composite completely enclose a volume. Surfaces maybe placed together to form a closed volume or solid. The system will automatically recognize a closed volume from the topological data. It can also provide feedback on where a composite is not closed. This consistent treatment of topological relationships helps make the system's integrated modeling environment easy to understand and use.

Many applications are possible only if the model has complete geometric and topological definition. One valuable application of this information is in the creation of engineering drawings. Since the drawing is derived from views of a valid solid model, all views of the drawing are guaranteed to be consistent. Other applications include, but are not limited to, mass property analysis, interference analysis, cross-section extraction, automatic meshing for Finite Element Modeling *(FEM)* and mold flow analysis. Interference Analysis can be valuable even if the effects of tolerances are not included. The user is given an easy way to check for interference that maybe soused by nominal size and position errors in a large and complex assembly.

3. **Tolerance:** Geometry and topology describe the nominal geometric configuration. Since nothing is (or will ever be) manufactured with exact nominal dimensions, the acceptable dimensional variation of the nominal geometry must be defined for the part or assembly. *ESMS* currently allows placement of the traditional tolerated dimension to note this important manufacturing information. The dimensions are generally placed in the context of a detailed drawing of the part.

4. **Feature:** A feature, as mentioned earlier, is a high-level concept of great flexible and power. It is critical to maximizing productivity in the design-for-manufacture work process. Feature data allows the identification of a particular area of a part to be regarded as a unit entity of known characteristics by some application. The concept of features encompasses manufacturing features, design features, inspection features, and other application views. Feature-based design or feature modeling allows the user to create part geometry and topology by specifying the location and size parameters for features instead of working directly with the basic geometric entitles.

In the mechanical design application, most features can be thought of as "geometric details" that exist only in relation to an existing geometric shape. As such, features are modifiers to a geometric shape (holes, slots, ribs, gussets etc). Features may also be of the type which simply attach attribute or property data to a specific region of the part but do not actually change its geometric form or data structure. Some geometric tolerancing definitions, such as "flatness" may utilize this technique of feature attachment. A feature can be a combination of shape definition and attribute information.

A parametric part in more of a stand alone entity of a given form but variable dimensions. Actually the only real difference between a parametric part and on intelligent feature is the level of abstraction. The part is defined as a single manufacturable entity. It can also, however, by considered a feature of an assembly with the part's size parameters and/or placement defined by some functional relationship to other parts of the assembly. A feature and parametric part are "intelligent" if they are automatically modified to conform to the rules of the original placement whenever the data required for their construction is modified. The implementation of intelligent feature modeling capability in *ESMS* solid modeling can significantly improve modeling productivity in many application areas.

5. **Assembly:** Assembly data defines the relationships between the parts of a product. This includes product component lists, nominal component orientation, kinematic relationships between the parts of the assembly, assembly tolerance, and assembly notes. In *ESMS,* the reference file is the foundation of assembly definition, reference files of virtually unlimited file, nesting of references files are also supported. The assembly data may be managed by the Intergraph Product Data Management System *IPDM).*

6. **Non-Shape and Notes:** This is a catch-all type category. Non-geometric design and/or

manufacturing information may be associated with an aspect of a design. This data consists not only of the typical information traditionally recorded as notes on drawings to clarify design intent, but also deeper knowledge about the design rationales. The new products of a company are generally derivatives of one more previous products. Hence, over a period of time, the detailed engineering design experience of the company becomes a very valuable source of the technical information needed for the next generation of product. This information is usually retained by the company in the minds, texts, and notebooks of its engineers and designers.

Though long experience with their product line, these professionals have learned design techniques, shaping and dimensioning calculation approaches, and product design requirements that appropriate for the type of products produced by the company. The problems associated with this somewhat volatile data storage mechanism are obvious. The *CAD* system can provide a mechanism for the preservation of and access to engineering data and experience associated with the company product-line. The goal is an increase in design efficiency through the use of concepts and techniques proven by company experience. A designer or engineer needs to be able to survey the company design experience to see how certain functional problems were solved in past products.

7. Administrative: Administrative data provides the information required to manage and control the Product Definition Data through the product life cycle. It was one of the most neglected areas in the previous generation *CAD* systems and is now recognized as key to the productive implementation of Computer-Integrated Manufacturing *(CIM).*

Managing Product Definition Data

As product definition data is generated for individual components, the tools necessary to collect and manage the information become more critical. There are four major fields in which data management can become more critical:

1. In the design process, so that the designs product are both functionally sound and aesthetically pleasing;
2. In the production of documentation, so that the product is manufactured as intended, since ambiguities are reduced to a minimum;
3. In the manufacturing process, by providing information for the control of manufacturing machinery, materials purchasing, and stock issue; and
4. In handling the revision of design, since the manufacturer, and ultimately, the user must be able to feed back information to the designer.

In all cases, the basis for *CAD/CAM* is data: data about components, about testing, about existing designs which could be used as the basis for new products, about machines, about standard and procedures; the list is endless. The problem, in the main, is to know what data is available; and where it is, in fact, to present the user with information about his resources. There is a great deal of difference between data and information; information is data presented in some intelligent, intelligible form. What is required is a data management system which increases the amount of information available and also increase the ways in which that information can be manipulated, not only within function, but globally.

Summary and Conclusion

The object model described in this work presents a new way of viewing engineering information association with the design process. The relationships that exist between information are often complex and span different levels of the object representation. The representation engineering information in a layered object format allows these relationships to be viewed as they actually exist. As a result, a qualifyable amount of information needed to insure both competent and manufacturable design can be identified. Using this information, a design process model can be generated by identifying its generation and flow between the levels of the model.

The layered object model is an object-oriented information representation, which is necessary for the development of a useful design database and for smooth and error-free communication. The layered model is a standard model of the information generated during the mechanical design process. As a result, it can exert a great deal of influence on the development of integrated ‘next-generation’ intelligent design systems.

Relational data management systems provide a flexible foundation on which to build an effective Product Definition Database. Application specific data management tools, built on this foundation, allow the user to structure a system that can manage the information maintained in a Product Data Definition *(PDD)* database. The key to making this work is a design environment that
provides the capabilities of generating a complete *PDD* model. Current state-of-the-art solids

modeling systems provide these capabilities. The ability to create a mathematically complete, unambiguous model of a design, with all related attribute data, interfaced to the data management system, truly provides a product definition approach to engineering data management.

References

Rangan, R.M.; Whelan, P.T. and Fulton, R.E. (1998). Managing Design/Manufacturing Information in *CAD/CAM* Environment. *UPCAEDM,* Atlanta, GA, 228-235.

Starkey, J.M. and Florin, G.J. (1986). Design for Manufacturability, *ASME Paper, 86-DET-121,* New York.

Teorey, T.J.; Yang, D.; and Fry, J.P. (June, 1986). A Logical Methodology for Rational Bases Using the Extended Entity-Relationship Model. *Computing Surveys, 18* (2), 187-222.

Wavd, A. and Seering (1987). An Approach to Computational Aids for Mechanical Design. *International Conference on Engineering Design,* Boston, 2, 591-598.