Information Management for Concurrent Engineering: Research Issues

Biren Prasad, Roger S. Morenc and Ravi M. Rangan

Concurrent Engineering 1993 1: 3
DOI: 10.1177/1063293X9300100102

Concurrent Engineering: Research and Applications (1993) **1**, 3–20

CONCURRENT ENGINEERING: Research and Applications

Information Management for Concurrent Engineering: Research Issues

Biren Prasad*, Roger S. Morenc†, and Ravi M. Rangan†‡

* *Electronic Data Systems, CPC GM Headquarters, P.O. Box 731, Bloomfield Hills, Michigan 48303-0731, USA, and † Structural Dynamics Research Corporation, Engineering Services Division, 2000 Eastman Drive, Milford, Ohio 45150, USA*

Received 5 September 1992; accepted in revised form 29 October 1992

The development of products in large industrial organizations involves numerous engineers from different disciplines working on interdependent components. Objectives are sometimes in conflict. The need for overall coordination, consistency, control, and integrity of data, design ideas, and design rationale is critical. The information generated by each designer must be viewed in the context of information generated by other designers, the enterprise historical data, and the organization as a whole.

The paper outlines major requirements facing concurrent engineering (CE). It focuses on the ability of collaborating designers to proceed independently, correlate interdependency, use existing information (data, knowledge, and processes), and negotiate conflicts arising from design inconsistencies. To provide information-based support for such environments a concept of design schemata is introduced to support the concurrent, collaborative, and historical aspects of CE environments from an enterprise perspective.

The need for a data dictionary that supports these schemata and its different dimensions is also recognized. The dictionary provides conceptual centralization of design information relative to the enterprise. This includes data, as well as its definition (meta-data), and must allow the design process to evolve in a global enterprise perspective. These discussions lead to a series of research issues that must be addressed by the CE research community.

Keywords: concurrent engineering, data dictionary, information management, product data models, product schemata, technical memory, enterprise integration.

1. Introduction

In recent years, many companies are beginning to view enterprise-wide product information as a valuable resource and commodity. This information refers not only to product data that describes designs and is used to build products, but also includes the build process and the enterprise knowledge (rationale and decisions) that underlie the product development life cycle. The product development process, like any other design process, is an evolutionary activity rather than a scientific activity [1]. The significance here is that whereas scientific processes are fairly rigid, structured, and are based on several irrefutable premises, evolutionary processes involve the application of synchronous, value engineering, and other organized knowledge (science, experience, organization, design knowledge and managerial skill, subjective analysis) within an organizational context. When these have withstood the test of time, they are usually embraced as rules of thumb (heuristics) or preferred processes. However, their contents are not absolute.

‡ To whom correspondence should be addressed.

This reliance on evolutionary processes, as defined above, makes the concurrent product design environment extremely difficult to model, and, as such, computationally challenging to support. Any such model must rely on intimate enterprise knowledge of design processes across and within several types of products. To design a generic suite of concurrent engineering tools that support elements such as data, knowledge, and processes, one must attempt to understand a full diversity of these issues. The ability to develop systems that support concurrent engineering requires that these design processes be well defined or some precedence is maintained. This is usually difficult to satisfy in the case of innovative designs. With routine design, there is normally some level of precedence. Researchers in conceptual design are often confronted with modeling unstructured domains and very *ad hoc* practices.

Numerous studies have attempted to rationalize the design process [2]–[7] by taking a theoretical approach. However, these are very difficult to implement in practice. Attempts at generalizing the product design process have yielded inconclusive results. Cross [8] mentions the characteristics of descriptive and prescriptive models of the design process. The formulation of descriptive models is expensive, time-consuming, and yields little information about the

1063-293X/93/010003+18 $08.00/0

nature of design [9], [10]. It usually involves protocol analysis [11] as a data gathering technique. Protocol analysis, due to its subjective techiques, is intrinsically controvertible. For example, Doblin [12] states: "Although I designed hundreds of successful products for major corporations, it suddenly occurred to me that I didn't understand what I had been doing . . . ". Designers tend not to articulate their processes fully. Often, they, themselves, do not understand them!

Several research studies have attempted to document comprehensively the product design process. These include well-established texts [13], [14], [8], which attempt to provide a systematic approach to design. Cross [8] and Pahl and Beitz [13] enumerate typical design methods and discuss, in depth, the techniques used during the initial and detail design phases. The primary purposes of these design methods is to formalize the design process and externalize design thinking. Cross asserts that such methods have the effect of aiding the creative process by ensuring that all avenues are fully explored in a systematic way. A contradictory viewpoint [15] observes that no such methodology should be imposed on the designer; that a chaotic, natural approach is in itself a methodology. Nevertheless, a thorough understanding of the design methods is a prerequisite to developing computational tools that are consistent with the designer's actions relative to the design environment. Research in descriptive cognitive modeling of the design process has revealed several theories on the cognitive nature of design [8], [10]. Several researchers have attempted to develop CAD-based support for such cognitive models [16], [10]. These efforts address the question: how do designers design?

Several collaborative design schemes involve management policies and practices. For example, several design-for-manufacturability (DFM) concepts rely upon human discipline [17]–[22]. The unified life-cycle engineering concept [23], [24] has been proposed as a concurrent design methodology where concurrence is introduced by a combination of managerial and technological strategies. Although analytical models may not always be available, the use of qualitative models, statistical data presentations, and heuristic-based advisory systems are suggested. The DARPA-sponsored DICE [25], [26] initiative has focused on computational aids for concurrent engineering. Initial prototypes are attempts at supporting concurrent and collaborative design processes such as design critics [25] and blackboard knowledge agents [27].

The importance of modeling and managing the enterprise design process has also gained prominence with the introduction of several product data management (PDM) systems. The PDM [28], [29] community has focused on developing products and tools to support processes in design/manufacturing organizations. The focus is on managing, controlling, and modelling some of the product-related activities associated with established design/manufacturing processes. These include capabilities such as tracking design files, controlling access to such files, maintaining revision histories, and notifying designers about changes. They also include information management, configuration management, and automation of predefined processes such as translation (using IGES, for example). The DICE PPO [26] attempts to model the product, process, and operations as is the PDES/STEP [30] effort, where configuration management is addressed in addition to product-related data.

While most of these efforts have stressed development of computational models of the concurrent design process, in this paper an attempt is made to understand the needs of infrastructure components in performing active services. How should one capture, manage, coordinate, and utilize the CE environment's constantly evolving data, knowledge, and processes? There are needs for support systems that manage, coordinate, and control these fine-grain design processes. There are needs of providing aids for developing initial values, utilizing previous design data, managing design variants, managing the concurrent processes, resolving conflicts between concurrent design actions, and providing mechanisms to check for design completeness. Any such infrastructure must allow product information (data, processses, knowledge) to be captured on a continuous basis in such a manner that the body of enterprise knowledge (technical memory) may grow with time. The information-based infrastructure required for realizing these needs entails several methods and research issues. This paper discusses these issues. Many of these methods and issues deal with technology that is currently maturing, while others are identified as potential research opportunities.

2. The concurrent engineering environment

Figure 1 shows that every enterprise carries with it some archival product data and knowledge, which is implicitly or explicitly used in every product design. In concurrent engineering (CE) environments, there are several concurrent contexts that are interdependent. Concurrence implies that these contexts can proceed in parallel. However, many of them are interdependent, requiring designers to proceed with partial information, with incomplete knowledge, and subjective interpretations.

The three areas that have significant impact in changing an engineering environment are:

- *Product and process classifications*: classifying the product and process into independent or semi-independent entities.
- *Life-cycle interactions*: timely availability of downstream information early in the design life cycle.
- *Knowledge propagation*: enhanced knowledge arising form the maturity of the product and processes.

2.1. Product and process classifications

In most large design organizations, the product development process may be classified into discrete functional areas such

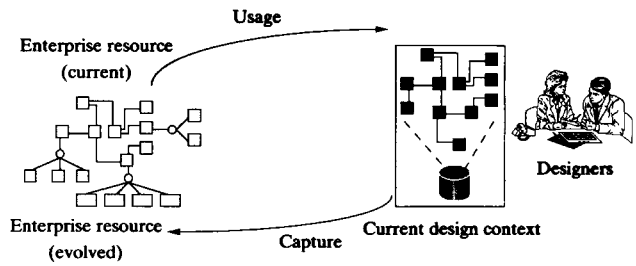


Figure 1. Evolution and use of enterprise design knowledge and procedure.

as conceptual design, detailed design, analysis, prototyping, process planning, manufacturing, production, maintenance, support, etc. Each of these functional areas may require several designers and engineers, and these functions usually operate in a concurrent mode. The involvement of several designers also implies a high degree of collaboration.

Figure 2 shows a typical product development process.

As seen in Figure 2, there is a high degree of overlap between functions. Consequently there exists considerable collaboration between designers, stylists, and layout engineers. System-level key attributes are often established *a priori*. Key part characteristics continually evolve. Analysis during these stages is driven by high-level part characterizations and idealizations, as well as key dimensions.

At times, it may be desirable to provide parallel process flows between the sub-functions. When concurrent operations are necessary, management practices such as inter-departmental committees, design guidelines, and inter-personal communication are employed. Today, this level of concurrence is supported by tools that are not designed for managing and controlling the flow of information along with the work flows. The possibility of concurrence is often restricted by the complexity of the process within each sub-function. For example, Figure 3 shows a further division between sub-functions. The evaluation of several performances areas within each sub-function may use different approaches and requires several attempts. This leads to many alternate possibilities being assessed. Furthermore, there are many relationships between the data in each of these sub-functions. Under these circumstances, current tools are incapable of paralleling these processes.

From the above discussion, one may note three important characteristics of tools and design aids:

- *Product parameters and function interdependence*: there are many inter- and intradependencies between the various functions and sub-functions.
- *Concurrent design process independence*: each function, sub-function, and designer, must be allowed to proceed independently to ensure natural concurrence.
- *Product design consistency*: the dichotomy of function interdependence introduces several design consistency issues. They result in the creation of design variants, while satisfying design conflicts and ensuring design completeness (and optimization).

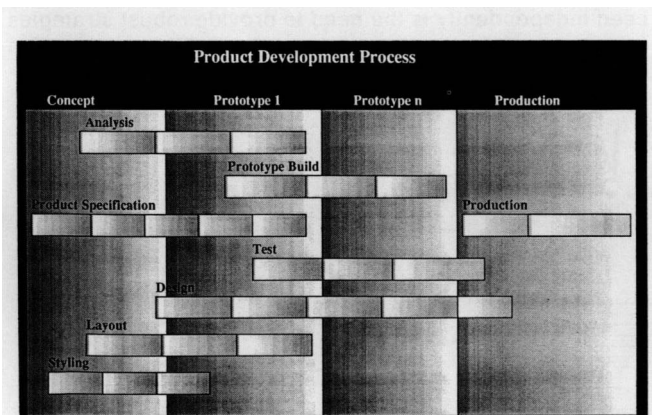


Figure 2. Product development process with sub-functions.

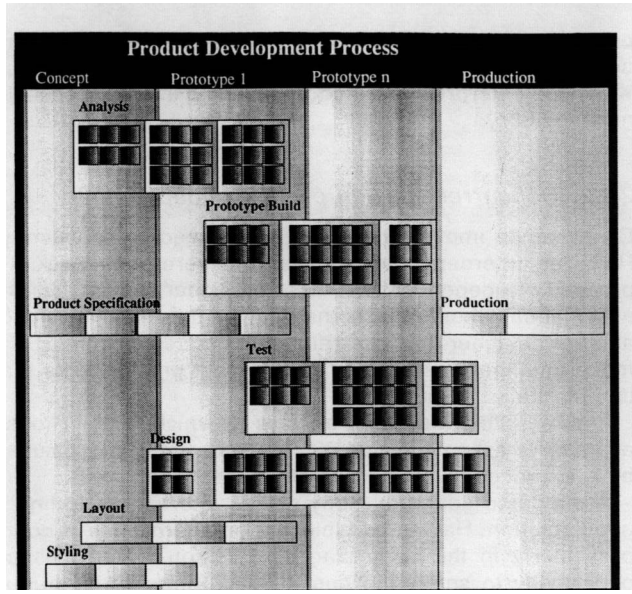


Figure 3. Concurrent function in the product development process.

2.1.1. Product parameters and function interdependence

Concurrent activities are usually accompanied by a high degree of product parameter and function interdependency. For example, Figure 4 shows that a designer responsible for analysis A may need input data X that is the result of activity B, the responsibility of another designer. Likewise, the designer responsible for producing X is dependent on the output of the designer responsible for producing Y.

Interdependence implies that these functions must also be coordinated. However, each organization and group may adopt different strategies to ensure such coordination. To facilitate this, tools and design aids are required to support these concurrent strategies. These tools may be different for different types of design situations. For example, innovative designs imply a suite of tools and capabilities that may be radically different from the needs of routine design processes. Furthermore, enterprise knowledge is continually evolving. New design concepts, data, and processes are

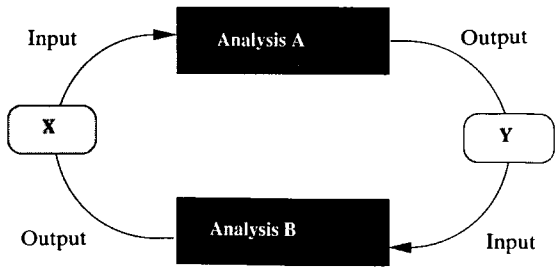


Figure 4. Interdependence between functions.

being generated and processed. They must also be managed, controlled, and captured, and must be assimilated within the enterprise knowledge, resulting in enhanced technical memory.

2.1.2. Concurrent design process independence

Concurrence implies the ability to proceed independently. However, interdependence poses a severe bottle-neck. To proceed independently, the designer must make assumptions about objects that may be the responsibility of another designer, and are related to his design context. To make intelligent assumptions, one must provide tools that can facilitate this process.

Figure 5 illustrates several ways by which A may choose an initial or assumed value, X1. This value may then be used by A to proceed with the design, independently of B.

A may also generate many design alternatives to find a good solution. Having multiple design alternatives is good, particularly in the early stages of design. It provides an opportunity to select the best balance among the various alternatives.

2.1.3. Product design consistency

The values within the designer's context must be consistent. This is an internal consistency requirement. Alternatively, external consistency is concerned with the interdependence between data from several designers, functions, or subfunctions. For a design to be complete, it must be consistent (both internal and external consistency requirements must be optimally satisfied). When individual design efforts and their dependencies are viewed against a collaborative context, several inconsistencies may result. Furthermore, each designer may generate several alternatives leading to the need for design sensitivity investigations.

Figure 6 shows that the independence associated with concurrent operations may generate multiple design alternatives, which may or may not be inconsistent. Note that X1 c X2; Y1 c Y2 leads to this inconsistency. Design independence in CE environments leads to design conflicts in the absence of consistency.

In Figure 7, one sees that design consistency is closely related to design sensitivity, and to ensure completeness, several iterations may be required. This is the feedback loop that is common in most design environments.

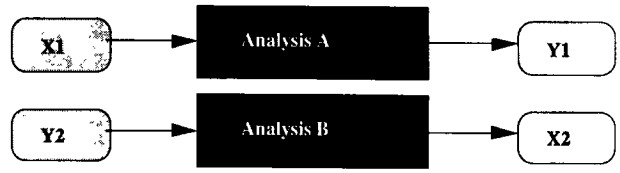
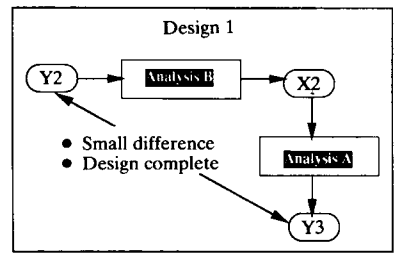


Figure 6. Design inconsistencies resulting from concurrence.



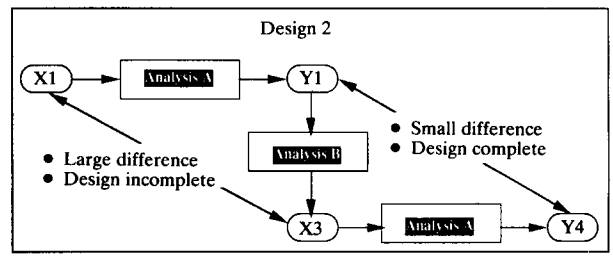


Figure 7. Design sensitivities resulting from concurrence.

Often, such design inconsistencies are resolved using strategies. For example, Figure 8 below illustrates a conflict resolution strategy, where the inconsistencies arising from independent concurrent design activities are resolved by using a technique similar to quality function deployment. The resulting matrix provides a basis by which negotiation and resolution can proceed. Such a strategy may be used to evaluate design alternatives by a single designer.

It is important to note that coupled with the ability to proceed independently is the need to provide robust strategies for resolving design inconsistencies, conflicts, and design

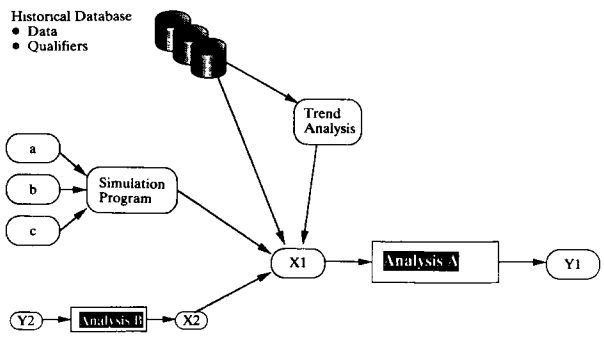


Figure 5. Different ways to select initial and assumed values.

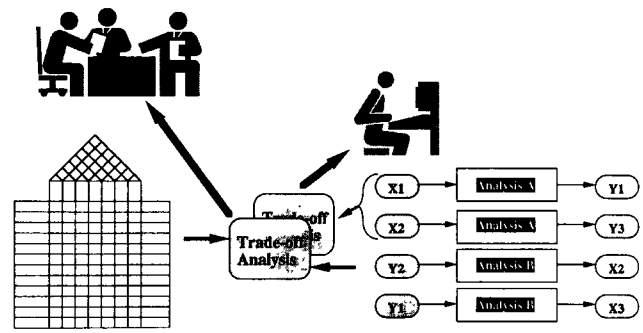


Figure 8. Design conflict resolution strategy.

alternatives. The environment must be flexible enough to accommodate various strategies and manage and control the execution of the process.

2.2. Life-cycle interactions

Interactions are an integral part of a CE environment, and, at times, are conflicting. For example, the interaction between downstream activities and upstream activities often leads to design inconsistencies. For some industries, manufacturing or manufaturing planning is not considered a downstream task, but is in fact an integral part of the early design and analysis phase. In such cases, design and manufacturing engineers must collaborate to understand the effects of manufacturing on design. However, once this collaboration has been achieved, the knowledge from it can be incorporated into the design process as tasks to consider during subsequent designs. Alternatively, it may be built into analysis or simulation programs to be used during design and analysis. Subsequent designs, therefore, may not need the same levels of collaboration between these two areas. However, new information created in the downstream activity must find a way to flow back upstream. Traditionally, this influence has often been captured in the form of rules, part-features, simulation and analysis tools, etc., providing a way by which knowledge of these influences may be captured and further utilized. Simulation and feature-based tools can be implemented today and can be very effective in facilitating concurrent engineering.

However, such knowledge is typically unique to the organization's product and process. A typical product or process may require many generic programs and features to represent fully the inter-relationships among different functions. The methods used to capture the enterprise knowledge are expensive to implement. They are usually developed in isolation—in the absence of any strategy. Any framework for CE must accommodate the development of an integrated information architecture and the capture of knowledge that includes both upstream and downstream information.

2.3. Knowledge propagation

The way in which one designs a product changes as our knowledge about that product is enhanced. In general, one may classify design processes following Pahl and Beitz [13] as follows:

- Original design.
- Adaptive design.
- Variant design.
- Fixed principle design.

The knowledge influences on designs are shown in the form of bar charts (see Figure 9). As noted, fixed principle

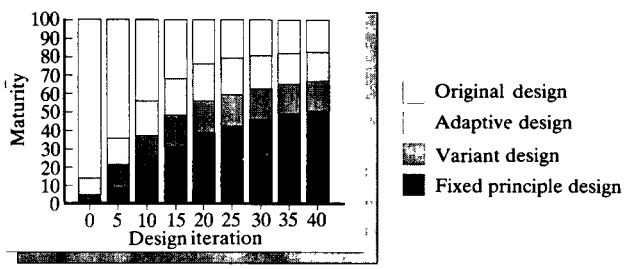


Figure 9. Design knowledge maturity.

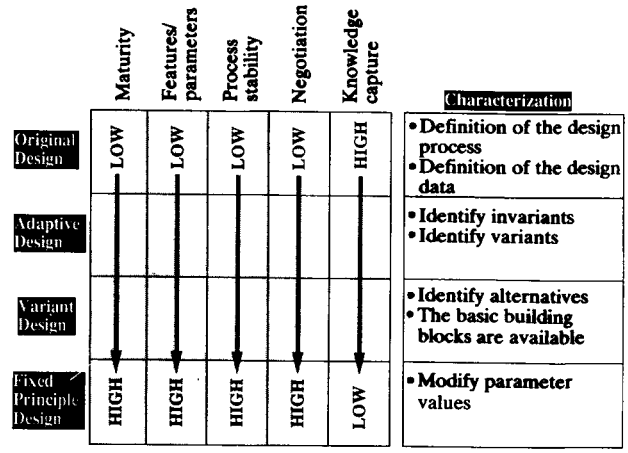


Figure 10. Different design types.

design is the most suited for automation. Original design is best suited for knowledge capture. Adaptive and variant designs help to solidify the generality of a design and bridge the gap between original and fixed principle. The next diagram summarizes some of the characteristics of these different design types. In addition, it illustrates some of the differences between the types.

The potential automation for fixed principle design is more likely since it contains stable design information in the form of features and parameters and because design processes are well defined. However, achieving this level of automation must start during original design and must be refined during adaptive and variant designs. To accomplish such automation, it is necessary to have tools available to the engineers for each type of design, and these tools should reflect the specific nature within the design type.

2.4. CE requirements

The CE requirements for information management can be broken down into the following categories: information modeling, teaming and sharing, planning and scheduling, networking and distribution, reasoning and negotiation, collaborative decision making, and organization and management of CE.

2.4.1. Information modeling

In this category, one should have, for example, the ability to:

- Capture and represent different forms of information: the design intent, the process, and the knowledge.
- Define different design representations (features, parameters, characteristics, specification, rules, etc.).
- Browse another designer's representation.
- View and navigate across design representations maintained by different designers and the enterprise.
- Store design information for future use (technical memory).
- Generically model and refine design concepts.
- Model products in an integrated manner as the design process evolves at different levels of granularity.

2.4.2. Teaming and sharing

In this category, one should have, for example, the ability to:

- Define teaming or sharing agents, which may be a person, a process, or a computer program running in the background.
- Share these design representations with others on the network.
- Share design process knowledge with relationships between design attributes and objects.
- Combine *ad hoc* design representations into a more structured form (i.e. generic or normalized) over time.
- Merge the knowledge gained from a new design with the previous designs.

2.4.3. Planning and scheduling

In this category, one should have, for example, the ability to:

- Identify and distribute independent product design activities.
- Identify and exploit existing information (e.g. design history) for a given design.
- Monitor design process events, and schedule appropriate actions to be taken, and the objects and subjects of these actions (change management).
- Select existing aspects from a previous design that can be applied to the current design.

2.4.4. Networking and distribution

In this category, one should have, for example, the ability to:

- Exchange data among and between designers, agents, tools, and systems.
- Design a file format for an analysis program.
- Associate rules, equations, and algorithms or programs at targeted decision points interchangeably.

2.4.5. Reasoning and negotiation

In this category, one should have, for example, the ability to:

- Define negotiation and conflict resolution strategies [quality function deployment (QFD), Taguchi's, etc.] at different levels of granularity.
- Identify interdependence and inconsistencies between and within design representations.
- Redefine design representations as a result of the negotation process.
- Identify affected design attributes and parties as a result of the negotiation process.

2.4.6. Collaborative decision making

In this category, one should have, for example, the ability to:

- Capture design intent knowledge, rationale, heuristics, and decision processes.
- Test the responsiveness (sensitivies) relative to different events (e.g. impact of changes, how they propagate, and who is affected).
- Enforce strategies (optimization, trade-off, sensitivities, inferences, backward and forward chaining) using manual or computer-based methods relative to the design representations.
- Trigger events and actions automatically, e.g. execute a design verification algorithm or allow automatic execution of an analysis program.

2.4.7. Organization and management:

In this category, one should have, for example, the ability to:

- Set up various interfaces and multidisciplinary multigroups.
- Manage and control the design activities at different levels of granularity.

3. Concurrent engineering enablers

Concurrency is the key for concurrent engineering. Paralleling work-groups represents one of its most important enablers.

3.1. Paralleling work-groups

A concurrent engineering construct must comprise several distinct parallel work-groups that specialize in a variety of disciplines, including product engineering, design, prototyping, computer-integrated manufacturing (CIM), and assembly technology. Experts in the field of mechanical, electrical, industrial, chemical, and materials engineering, as well as a variety of other fields, work together on creative solutions to meet today's demands and anticipate tomorrow's challenges. This diversity of disciplines is essential to address the growing complexity of today's product requirements and global manufacturing trends. Each must work closely with other groups to identify and develop techniques that are more cost-effective, innovative, and simple to use. Suppliers must work with their clients and vice versa to design the manufacturing processes, tooling, equipment, and systems needed to fit the production and plant limitations. Each unit gains the strengths provided by the resources of all the groups combined.

There are six additional enablers of CE:

- *Multiple processes or concurrent sessions*—relates to dividing the work into sizable chunks of semi-independent activities.
- *Paralleling activities*—relates to assigning the activities to the individual work-groups so that each one can work in parallel.
- *Fast processing*—relates to performing individual activities as fast as practicable.
- *Simultaneous processing*—relates to simultaneously starting as many activities as possible.
- *Minimize interfaces*—relates to keeping the number of interfaces (manual or data transfer) through the "product process" cycle to a minimum.
- *Transparent communication*—means a seamless (virtual or computer-supported) communication between and across the activities that are segmented.

These are elaborated further in Section 5 of this paper.

3.2. Schemata for enabling concurrency

Concurrency can be exploited by employing one or more of the following schemata during a product development cycle (see Figure 11):

1. *Perspective*: at the highest level different work-groups can work on different perspectives of the design in parallel.
2. *Hierarchy*: the product may be divided into logical hierarchical blocks depending upon its complexity (see Figure 12). Different teams can work in parallel in these different hierarchical groups. People in each level can work concurrently. Some dependencies can exist between the layers. Establishing the common interface standards for communications and definitions of common problem parameters and checkpoints can allow parallel groups to work concurrently. Checkpoints are essential to facilitate couplings of completed tasks to take place smoothly. This is shown in Figure 11 by the staggering of the product breakdown structure tree. For example, the system-level tasks can only begin when tasks for the subsystems's phase are already well underway.
3. *Multiplicity*: within each different hierarchical group, say a part or a component group, multiple parts or components going into the final product may be worked upon simultaneously.
4. *Alternatives*: within one hierarchy, a group of designers guided by the hierarchy leader may be working on several alternative ideas in parallel.
5. *Characteristics*: each alternative may involve functional characteristics from different disciplines dictating its designs, such as compliance with characteristics like aerodynamics, noise, ride quality, etc., at the same time. People from different disciplines may be needed to support these requirements. These teams can work in parallel.
6. *Projects*: multiple analysis may be required to evaluate product compliance to a functional requirement. Many project teams may be working in parallel to determine the integrity of the design with respect to this requirement.

Many of these representations are common in large organization, programs, and products. In a small organization, a simple product may not have the same level of sophistication to warrant its full implementation. Figure 11 schematically shows a cluster of concurrence schemata. Perspectives are shown as the top layer consideration followed by hierarchical breakdown of a product into systems, subsystems, components, parts, and, finally, materials. Figure 11 also shows further breakdowns of hierarchical structure into three-dimensional representation. Possible "multiplicity" of parts is represented along the *x*-axis, "alternatives" on the *y*-axis, and "characteristics" along the *z*-axis. Each of the axes can further be broken down into individual projects, not shown in Figure 11 for clarity reasons. These notions of concurrence help reduce the time-to-market aspect. However, several technical issues remain unresolved. Consistency or interdependence is one such issue that makes concurrence difficult to achieve. Current tools that support these levels of interdependence are very limited in capabilities. For example, typical DBMS including ROSE [31] deal with concurrence using a simplistic unit of work transactions, which are just not robust enough for the types

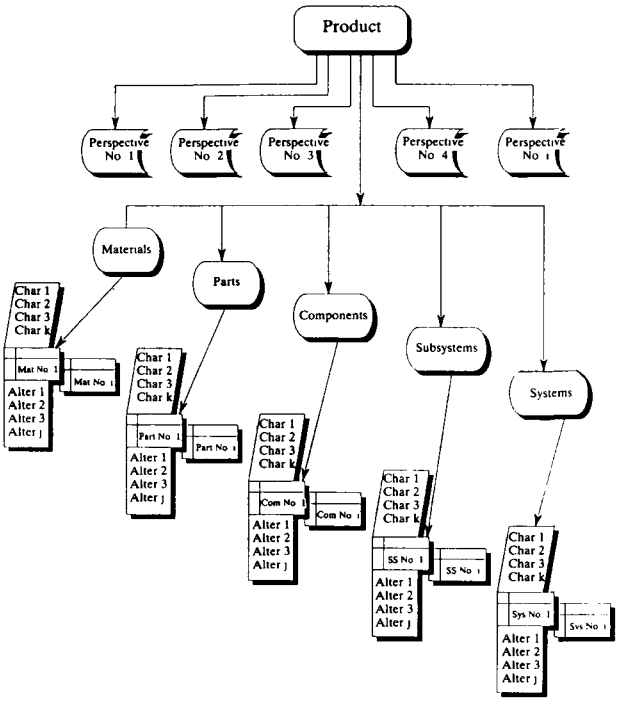


Figure 11. Schemata for enabling concurrency during product synthesis.

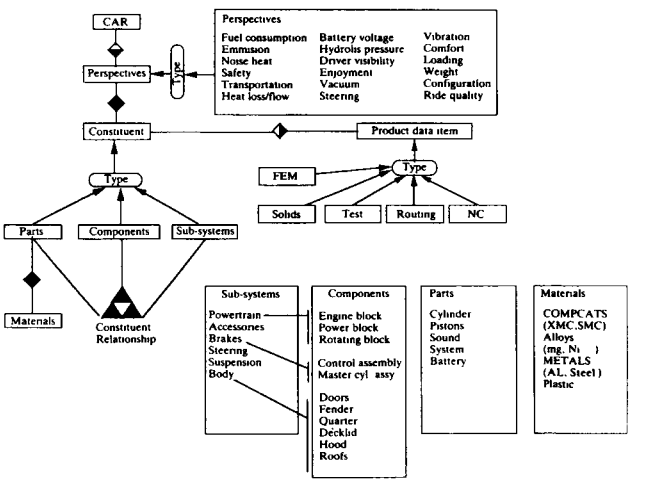


Figure 12. A product schema for a car (E+R model).

of interdependencies that prevail in several of the aforementioned concurrent settings. This is discussed at length in Section 4. The other major issue is "synergy", which is discussed next.

3.3. Matrix of concurrency

Synergy is the cornerstone of any CE organization, combining individual capabilities to produce results greater than any single effort. For synergy to take root and make teamwork cooperation effective, companies need information

Table 1. Matrix of concurrency.

ID No.	Mode of interactions	Work-group configurations: Single user (1)	Cooperating users (2)	Simultaneous users: Different versions (3)	Simultaneous users: Same version (4)
1	Access own products' interaction tools or applications (PITA)	Sequential engineering (SE)	SE	SE	SE
2	Run against their own DATA	SE	SE	SE	Sequential/ concurrent engineering (SE/CE)
3	Access PITA belonging to other work-groups	SE/CE	CE	CE	CE
4	Access DATA belonging to other work-groups	Concurrent engineering (CE)	CE	CE	CE
5	Access both PITA and DATA from other work-groups	CE	CE	CE	CE

management tools that perform in concert. The human side of the information management is an important carrier for making CE really work. Table 1 shows a matrix of concurrency that are possible in an information management of CE. The horizontal row shows the various modes of operations. The vertical column shows the possible configuration-setting of the work-groups.

1. Single user: a single user is responsible for design decisions.
2. Cooperating user: often, in a team enviroment, some processes are sequential. A cooperating user is a person who completes the work left unfinished by previous users. Such a person could be a customer, a member of another product team, or a member of any downstream organizations, who may require to work on the uncompleted task. Other responsibilities may include exploring or replaying a logic and analysis of the specified product form or the understanding of the processes that produce it.
3. Simultaneous users: this means multiple users or product developers are accessing the product information tools or application. There are two situations in which this may occur. The users may access the same design, tool, or application concurrently, or different users may access or edit different versions of product information tools or applications (PITA) at the same time.

Along the vertical axis in column one, the different possible modes of interaction for these levels of users are shown. Any of the work-group configurations can have five modes of interaction, not all of which leads to concurrency. For example, if a single user accesses his or her own PITA, the process is termed as sequential engineering (SE). Even if his partner (a cooperating user) accesses the information following his design, the process will still be sequential [see Figure 13(a)]. Similar situations occur when they try to run these functions against their own data. In other situations, concurrency is present in varying degrees. For example, concurrency occurs when a single user or cooperating user accesses PITA from other groups and runs them against the data from other groups in a computer environment tailored to their perspective on the design [see Figure 13(b)]. Concurrency is enhanced still when users at two different locations perform the aforementioned operations simultaneously.

The degree of concurrency increases as one moves from top to bottom and from left to right (see Table 1). The situation depicted by the bottom rightmost rectangle provides the largest degree of concurrency. It may be noted from this table that a style concurrency, where simultaneous users access the same version of PITA and run their own DATA—the location (2, 4)—is characterized as both SE and CE. A similar situation occurs when a single user accesses PITA belonging to other work groups—location (3, 1). Some computational environments for CE are such that they prevent their clients from editing a design module until another user is finished with that design. Even though the two users can work in parallel, the changes cannot be posted until the latter has had the chance to review the changes made by the first user. The latter continues only after the first user is finished with his version of the design module. For a critical module in the design, this particular slot may mean that concurrent engineering is no faster than sequential engineering. Some database systems, e.g. ROSE [31], thus support this type of concurrency by allowing multiple users and applications to edit different versions of the same design module. It reduces the length of the design cycle, provided that the benefit of concurrently editing the module is greater than the cost of having to merge multiple versions of that module at a later time.

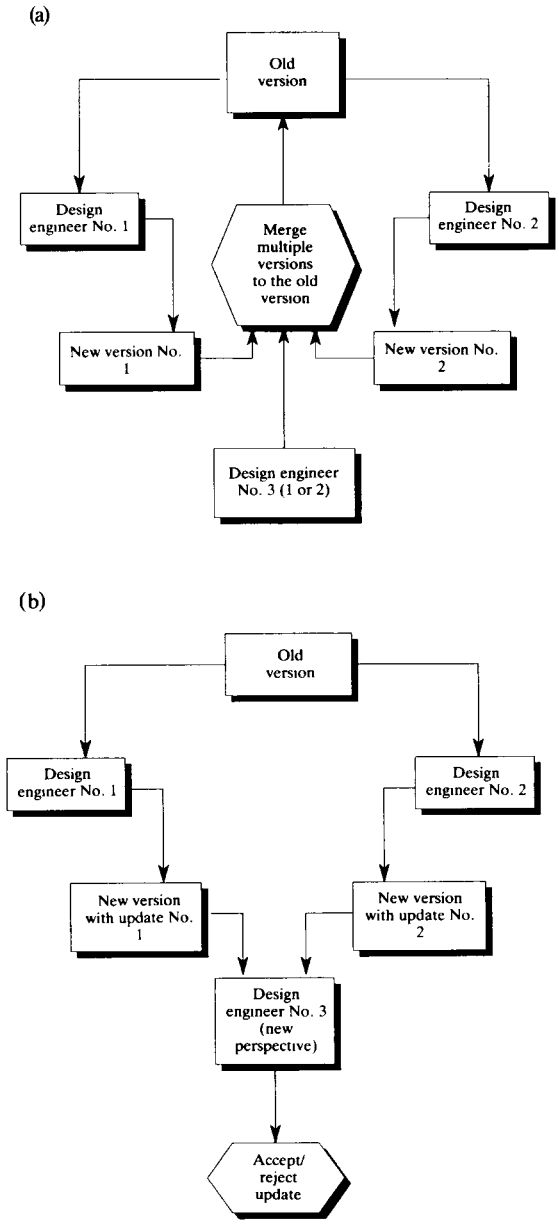


Figure 13. (a) Simultaneous users working with multiple versions of designs (sequential engineering). (b) Simultaneous users working with multiple versions of designs (concurrent engineering).

4. Information management in CE

Information management of CE is a global problem and thus it requires a global strategy and solution. Most solutions today tend to be isolated. To appreciate the various modes of operations fully, it is necessary to consider the needs of a mature design environment across many departmental and discipline boundaries, with historical data that will evolve over time. The information management stategy should include how design information is represented in the computer and how different processes interact with this information—whether these processes are automated or manually controlled. The following discussion is intended to give a more detailed insight into possible ways of looking at some mechanisms that may be needed to achieve a robust solution.

In the context of this discussion we define a schema as a logical representation of a structure of information. Information includes data, processes, and knowledge. A design may be thought of as a schema that relates different design object classes and its eventual population. To support the CE environment in its entirety, several types of schemata are required. The relative dynamism, language dependency, design-type dependency, and stability of these schemata require diverse implementation strategies.

The need to accommodate various types of design representations as well as design data forces us to deal with different levels of abstractions. The interaction between data, process, and knowledge presents representation–integration issues. The problem of scale cannot be ignored. Typically, design organizations deal with a tremendous amount of information. This section will review some information technology components required to support the needs of the CE environment, specifically:

- The data, process, and knowledge dimensions of design information.
- The various types of schemata required to support the CE design environment.
- The use of a dictionary as a necessary framework component.

4.1. Dimensions of enterprise information

In product environments, design data representation is concerned with what a design is. The design processes are concerned with how a design may be realized; whereas the design knowledge is concerned with why certain actions are carried out. Current trends have been aimed at capturing design data and to some extent the knowledge but have largely ignored the process. In the future, it is important that one captures all these aspects.

4.1.1. Design data

There are several types of data that are created, used, and processed during the product life cycle. Associated with each part (detail, assembly, delivery) is a myraid of data relevant to the enterprise, the individual designers, or groups of designers. These data items may include key dimensions, geometry, analysis models, drawings, test data, manufacturing fits and tolerances, NC files, process plans, machining, manufacturing, etc. From an information management standpoint, the structural semantics of data may be represented using schemata. A detailed account of the type of data that occur in engineering environments and the semantics required to represent them is presented in Morenc and Rangan [32].

4.1.2. Design processes

The design process may also be captured and represented in much the same way as design data. There may be different types of processes associated with a design process. For example, a designer may execute a series of actions (decisions, simulations, analyses, etc.) while performing a

design. The conflict resolution negotiations may follow a series of actions to ensure design completeness. The enterprise may impose a process to facilitate a controlled design process. While each of these processes implies actions in response to events, these actions may apply to objects at different levels of granularity.

The need for a process schema that links events with actions and the data objects is apparent. Several methodologies may be used to model processes [33]. These process schemata describe how the product is designed. As these processes are validated with time, they may become candidate enterprise schemata.

4.1.3. Design knowledge

As a result of these design processes, it is possible to represent the knowledge and rationale associated with the design process in knowledge schemata. Several knowledge representation methods are currently available [34]; they capture why a designer or the environment carries out an action on an object. For the designer, these may be justifications for actions. At the collaborative level, there may be reasons and justifications for the tradeoffs and negotiations. At the enterprise level, there may be justifications for the approval and review processes. As these knowledge components are validated with time, they may be absorbed as historical enterprise knowledge.

4.2. Types of schemeta

To provide an information-driven infrastructure to support CE environments, several types of schemata are required. These schemata must describe the data, process, and knowledge dimensions, and support important CE characteristics such as design independence, collaboration, consistency enforcement, knowledge capture, and utilization, etc. The following schemata are introduced in this section:

- Local design context schemata.
- Integrated product schemata.
- Integrated resource schemata.
- Integrated enterprise resource schema.
- Data representation schemata.
- Consolidated enterprise schema.

Each of these schemata is populated with data objects and may be represented using several languages.

4.2.1. Local design context schemata

The CE environment must support several concurrent design activities. A local design context schema is introduced as a representation of the independent needs of a designer or sub-function. As a result of the interrelationships between the various functions, sub-functions, and designers, a CE environment must support several concurrent local design context schemata (A, B, C), (see Figure 14).

These schemata evolve gradually as the design becomes more concrete. Design independence requires that all objects that lie within a local design context schema are under the exclusive control of the designer. A designer may generate several design variants within his context. Each of these variants must be consistent. However, design interrelationships establish constraints at the interfaces that must

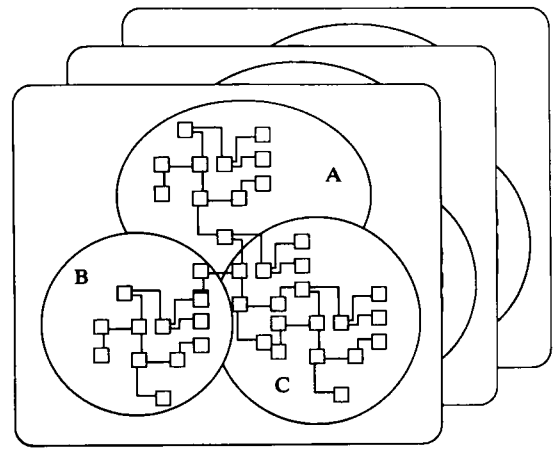


Figure 14. Inter-related local design context schemata.

be resolved to ensure design consistency. The layers in Figure 14 represent consistent designs after all conflicts between participating designers have been resolved. Commensurate with the representation of design data is the need to represent the processes used to create this data, and their justification and rationale. There are dimensions of the local design context schema that capture design data, processes, and knowledge.

4.2.2. Integrated product schemata

A product schema represents several design objects for a given product. This schema, and its population, contains all the data required to define different aspects of the product for the different functions within the enterprise. In CE environments, since the product definition results from the resolution of conflicts between various local design contexts, the product schema is effectively an integration of the various local context schemata. This integration process is aimed at ensuring design completeness (conflict resolution, negotiation, tradeoffs, optimization, etc.).

For example, Figure 12 shows a product schema for a car. It uses the extended entity relationship model constructs [35], and identifies specific components of the car for this design and the data associated with these components. An instantiation of this schema represents the design of the car. This schema and its population are a result of the efforts of several groups (engine development, exhaust, body engineering, etc.), and their consistency enforcement policies.

The description of the product by means of a schema (the product schema) is a dynamic process. For example, as the car's design becomes more concrete, its schema evolves dynamically until it finally describes the car. It also determines the design information that must be available, such as analysis, geometry, documentation, etc. Although there may be much commonality between different cars, each car's schema (and its population) may be thought to be unique. It is appropriate to refer to a part of the product schema as a design specification schema. This schema captures the design requirements and provides a reference point for design intent. This is an evolving schema, since the requirements become more concrete as the design progresses.

The process dimension of the product schema defines how the resolutions of the local design context schemata were achieved. It defines what events occurred, what actions were carried out in response, and determines the processes carried out by the enterprise to control and manage the overall design process. For example, the collaborative process required to ensure design consistency across the design contexts may adopt a conflict resolution strategy. The strategy may be represented in the appropriate conflict resolution process schema. The implementation of this strategy is a service component and will not be considered here. Similarly, the overall design process indicating how the functions coordinate with each other may also be represented in another schema, such as a process graph [36].

The knowledge dimension of the product schema captures why and how conflicts were resolved.

4.2.3. Integrated resource schemata

One may also recognize the existence of an integrated resource schema (as a library schema representing the enterprise's data, process, and knowledge resources) from which the product schema is composed. This is true, although the product schema may contain novel elements (newly designed components, newly acquired processes, and knowledge) not available in the resource schema. The resource schema contains building blocks from a historical perspective and may be thought of as a reusable library. For example, several cars may use the same engine. The engine selection process is similar to previous processes and may use existing knowledge in the process. The engine's process, data, and knowledge dimensions are captured in the resource schema.

The resource schema represents a schema for capturing the knowledge base (technical memory) for a given domain. For example, there may be a resource schema for automobiles that captures the complete historical data of all components and assemblies used in cars, their types, combinations, and variations. It is possible to consider this as a global parts' list (library of parts)—a historic knowledge base that captures the full details of all components supported by an enterprise, and their associated processes and knowledge. One may note that the resource schema is dynamic since it evolves with every new design. As more designs are completed they must be posted within the resource schema. An enterprise may maintain several resource schemata for different topical domains.

Figure 15 illustrates a simple resource schema for an automobile.

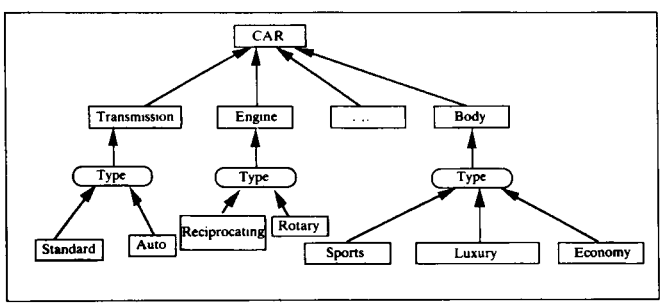


Figure 15. A simple integrated resource schema for an automobile.

The term integrated resource schemata is used because a resource schema is of limited use by itself. Once it is integrated with various product schemata, its domain of applicability increases manifold. As new products and concepts are defined, they must be integrated within the resource schema. This integration is aimed at identifying new designs and determining how they must be added to the resource schema. Such integration requires domain knowledge. For example, how should a new design concept such as a rotary engine be integrated with a resource schema for automobiles that currently supports only reciprocating engines?

4.2.4. Integrated enterprise resource schema

In the previous section it was mentioned that an enterprise may maintain several integrated resource schemata for the different topical domains of the enterprise. The integration of the various topical resource schemata within an enterprise results in an integrated enterprise resource schema. This integration is aimed at eliminating redundancies and centralizing the resources of the enterprise. It defines an enterprise schema for capturing one's enterprise knowledge (technical memory). Such enterprise resource is often derived from the evolution of the resource schemata. To understand what it meant by evolution of an integrated resource schema, consider the simple example shown in Figure 16.

Option 1 implies that parts may be made up parts, and so on. Option 2 shows that an assembly is made up of one or more reusable parts and subassemblies and that subassemblies may be composed of further parts. Option 3 implies that a part is either a detail part, a subassembly, or a product, and that a subassembly may contain one or more parts for each product that it is associated with. Option 4 explicitly identifies the key assembly components in an automobile—with rich semantic implications.

All the schemata above represent different levels of semantics. A designer who is designing an automobile today will most likely work with schema "integrated product schemata". But, how has this schema been formed? It is a culmination of more than 100 years of experience in engineering automobiles. The schema is relatively stable, and new designs may employ sub-graphs of this schema. For example, the product schema of a car—described in Figure 12—may be built from this schema. Similarly, what one

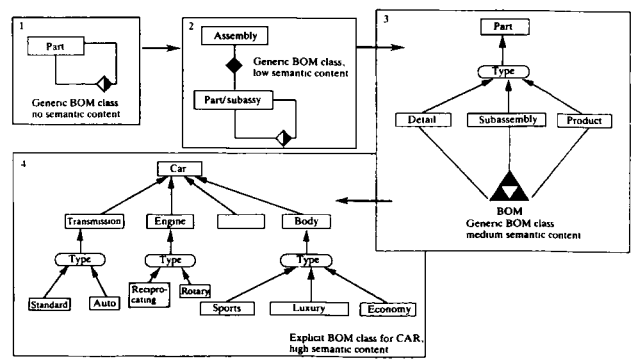


Figure 16. Evolution of integrated resource schema.

refers to as manufacturing features today are parameters that have been recognized as features due to the frequency of their use over time. With the maturity of the design process, and with time as the relationships become better understood, the schema may evolve to a relatively rich representation.

Just as the design data in the resource schema evolves with time and experience, the processes and design knowledge associated with an enterprise also evolve. For example, practices and knowledge from the past are captured and evolve into structured processes that may be used by the designers in solving a problem. For example, in the automotive industry, most processes are very well defined. Most project managers have a very good idea of how a project's engineering activities will be conducted, and the same goes for the individual designer. Given a design problem, experienced designers usually have a good idea of how to proceed. Novice designers will have difficulties. However, with the existence of a resource schema that describes both processes and knowledge, novice designers may not face much problem.

4.2.5. Data representation schemata

The local design context schemata, the product schema, and the resource schema must support richer representations of data used to describe a design. Figure 12 illustrates that a component is described using several rich representations. For example, the finite element model and solid geometry model that describe the designs are instances of a fairly rich FEM and geometry schema. For example, Figure 17 represents a rich schema for a FEM model. This is essentially a complex class, and each instance of this class represents a FEM model. The constructs used to describe this class are adapted from Hull and King [37].

The data schemata are very stable, change as a result of changes in technology, and are independent of the design process. For example, the FEM schema is relatively stable and undergoes revisions resulting from changes in FEM technology (e.g. new elements, capabilities, etc.). The STEP/PDES [30] effort is currently attempting to define standardized representations of these rich schemata (geometry, etc.).

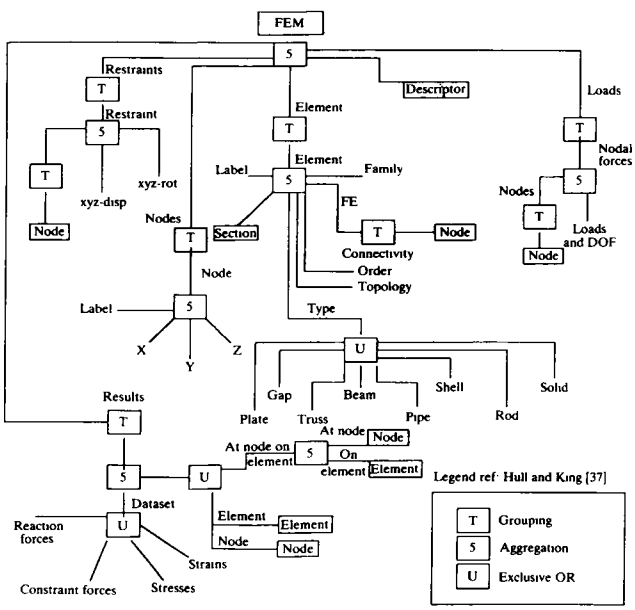


Figure 17. A typical FEM data schema.

Figure 18 summarizes the relationship between these schemata.

For a new design effort, one may perceive the integrated resource schema to be a schema definition library. When a design process starts, initial design context schemata may be defined using the design requirements' schema and the integrated resource schema. The local design context schemata evolve independently and integrate (through conflict resolution) to form the product schema. Every design is thus described by a dynamic product schema. This schema references several rich data schemata. As the design is completed (the product schema stabilizes), its contents must be integrated into the integrated resource schema.

Potentially, each new design may imply a radically different schema (and its population) as in the case of original designs. Alternatively, it may imply a modification to an existing schema (and its population), as in the case of variant designs, or merely a population of a given schema for fixed principle designs. At this stage, one also recognizes the existence of a dictionary that provides an information framework and supports these schemata, their populations, and their relationships. The CE environment dictates several specific requirements to this dictionary. The dictionary provides an active service by managing the information needs of the environment from a global and temporal perspective.

4.2.6. Consolidated enterprise schema

The trilogy of data, process, and knowledge implies the existence of another integrated schema—the consolidated enterprise schema resulting from their integration. In effect, this schema captures the entire knowledge of the enterprise. This is shown in Figure 19.

This integration aims at ensuring that the data entities are tied with the processes that created them and the knowledge used in the process.

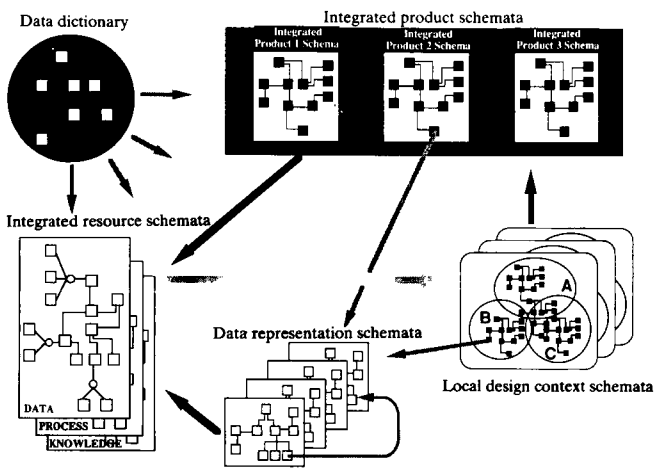


Figure 18. Relationship between data, product, resource, and design context schemata.

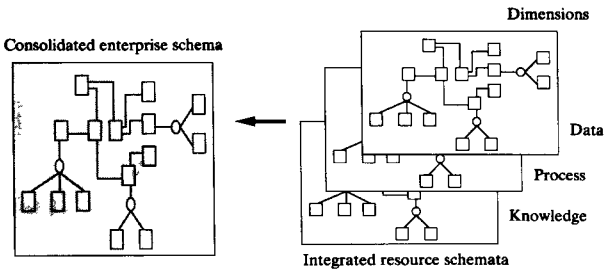


Figure 19. A consolidated enterprise schema.

4.3. Modeling enterprise information

The need to model data, processes, and knowledge together requires suitable methodologies. Traditional approaches to system analysis and system design have focused on modeling the process first [38]. The identification of the data required to support the processes, and the knowledge to enforce it, have been secondary issues. Currently, this process-first approach is being supplanted by ongoing object-oriented analysis methodologies and techniques [39] that stress a thorough understanding of the static component (the data) before attempting to identify the dynamic aspects—the processes. These processes are consequently resolved into operations that act on the data objects resulting in operational objects. These objects capture both the data as well as the behavior of the objects, and the separation becomes indistinguishable.

As noted, one may view design as an activity by which the product schemata and data schemata may be defined, extended, and populated. This does not, however, imply that the data precedes the process. For example, in the case of original designs, one must first create (a process) the schema. The process thus precedes the data. In effect, this process is preceded by the existence of the meta-schema class. This duality indicates that the process and data are very tightly integrated, and, consequently, the implementing paradigm must also support this integration. The ability of the object-oriented model to support the integration of data with the processes makes it a suitable methodology for modeling and implementing the CE schemata.

4.3.1. Design representation languages

In CE environments, one must recognize the existence of several languages of different types (data, process, knowledge), and variations within these types: IDEF 1X [33], NIAM [40], entity relationship models [35], EXPRESS [30], etc. Several data modeling languages, knowledge representation schemes, and process modeling languages coexist in practice. The need for different types of information (data, process, knowledge) required to support the design process and the different granularity (at the designer's level, the organization level, etc.), and language representation preferences of the designers imply that several languages may be used to specify the local design context schemata. These languages are technology-based, not a function of the design process, and therefore will evolve very gradually. The ability to support different languages while maintaining the relationships between the schemata poses several research challanges.

To model these design schemata, an important concept is the notion of meta-schema: a meta-schema is a schema to represent schemata. The idea of representing schemata for their own sakes is very attractive. Traditionally, schemata have been viewed as templates for persistent objects. However, several concepts introduced in this section propose schemata to be objects that are themselves manipulated to support the CE environment. This duality of schemata is useful in CE environments and raises several important research issues.

4.4. Dictionary: a framework component

Earlier, the concept of a dictionary was introduced as an active framework component to support the needs of the CE environment. The word active implies that the infrastructure component must assume responsibility for several CE-based services, allowing the applications and users to proceed independently of these services. The dictionary must provide conceptual centralization of design information relative to the enterprise. This includes data, as well as its definition (meta-data), and must allow the design process to interact with a global enterprise perspective. In this section an attempt to identify characteristics of this dictionary is made.

The dictionary, as introduced here, must support the coexistence of languages, dialects, schemata, and data in CE environments. The dictionary must contain an abstraction dimension to support this coexistence. This concept parallels the Information Resource Dictionary System [41] philosophy.

Figure 20 summarizes the IRDS four-level information dictionary architecture. At the highest level, L1, are defined the constructs of language components with which various modeling languages may be defined. The L2 level is concerned with the grammar of the language being used. The vertical lines imply that there may be several languages used for different purposes. The L3 level defines the meta-data for the problem at hand, e.g. a schema for the product data and a meta-schema for this schema. The L4 are the actual instances of these schemata, e.g. the actual product data (product schema instances) and the schema description (product data meta-schema instances). The information infrastructure must support capabilities that support the infrastructure as well as the environment. For example, schema integration, as described earlier, is an infrastructure

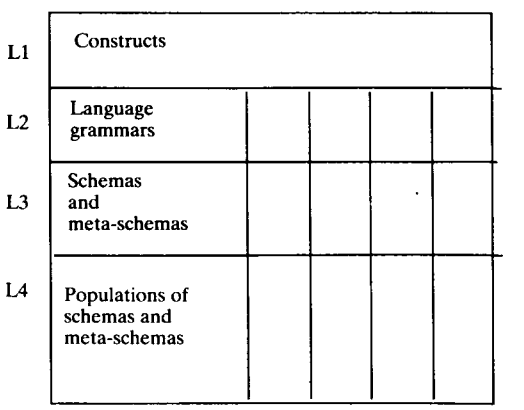


Figure 20. The abstraction dimension of data dictionary.

requirement—whereas collaborations is an environment requirement (the navigation of schemata relative to user views). Schema integration is intimately related to views. Just as schemata can be integrated to form new schemata, it is equally desirable to generate views from integrated schemata. This introduces a view dimension to the dictionary. The view dimension parallels the three schema architectures [42], where the integrated schemata are analogs of the conceptual schema, and the component schemata define the external view. The internal view is provided by the implementation.

Both the abstraction and view dimensions are critical to the CE environment. The view dimension supports the collaborative, interdependent nature of CE environments by allowing designers to navigate schemata generating preferred views and integrating schemata by resolving conflicts and enhancing the enterprise knowledge. The abstraction dimension supports design independence providing a unifying framework for all the CE components (data, schema, language) to coexist. These dimensions of the dictionary allow anyone to view design from an organizational and a historical perspective—permitting design actions to be related to a global enterprise perspective.

5. Research issues

Section 4 has introduced several interesting concepts that are useful in supporting design activities from a global enterprise perspective. The relevancy of the schemata to the CE environment present several research challenges. These are briefly outlined in this section.

5.1. Capture and use design information in an unobtrusive manner

What is a good representation? How can designers interact with these representations in a natural manner? How can one implement such systems in a given CE environment? The ability to customize generic tools to a given environment requires work in modeling and generalization. In addition, toolkits to support customized development of design processes must also be made available. Research in this area must focus on identifying generalities and modularizing systems. These topics require research in the nature of interfaces between designers and computer-based tools, in addition to new languages and representation techniques.

5.2. Schemata organizations

The schemata introduced previously evolve at different rates, depending upon the type of design supported (innovative or routine). They are related through complex operations such as schema integration, conflict resolution strategies, knowledge capture, semantic equivalence, etc. How can one manage these relationships? Furthermore, since schemata exhibit a dual role (as persistent object stores and as instances of meta-schema classes), one must maintain the identity relationship as a result of integration, evolution, and view generation.

There are meta-schemata to represent each of the design schemata. These meta-schemata may impose their own representation schemes. How can one uniformly model meta-schemata? What operations must these meta-schemata support? For example, the design context meta-schema must allow for the local design context schemata to evolve, generate variants, invoke processes to ensure design consistencies, resolve conflicts across schemata, and integrate schemata. Is the integration operation supported by the local design schema meta-schema or the product schema meta-schema? These are some questions that must be investigated in more detail.

5.3. Product and process organizations

The focus of this paper has been to view design from a global organizational perspective. More research is needed to understand the relationships between the individual designer and the engineering enterprise. Research in product design must focus on design process issues that are organizationally motivated in addition to cognitive aspects. The needs dynamically to define relationships between design attributes are very important. For example, how can the design information (process and knowledge) be used to define data relationships? Approaches must investigate the applicability of neural networks and inference methods.

5.4. Virtual teams

The word virtual in this paper does not mean 100% machine-supported, neither does it mean hands-off (0% human-based) operation. It is used here to indicate computer-based cooperative work (CBCW) practices and a CBCW team concept. A virtual team in this setting relates to a CE concept that maximizes the role of information technology in controlling the design process or minimizes the manual interfaces and supports. This definition raises several interesting issues. Can a computer program replace a designer? What level of automation can one achieve? Is there a need for a distinction between different skill levels, say a system analyst from that of design or project engineer? When do the design representations become an integral part of the design? This may introduce several conflicts relative to the knowledge and usage of such systems.

5.5. Virtual integration and mapping

The consolidation of views into integrated views and the generation of views from integrated views are common themes of virtual realities. Let us investigate these in more detail.

5.5.1. Virtual integration

Here, the term virtual integration is used to indicate types of integration operations, which are demand-driven—meaning they are automatically activated on demand. Traditionally, schema integration is viewed as the process of integrating various component schemata into a unified schema while maintaining the component semantics. Schema integration research [43], [44] has been driven by problems arising in distributed database environments. In the CE environment, the ability to integrate component schema is complicated by

the dynamic and evolving nature of the component schemata (the product schema and the design context schemata). Each of these schemata evolves with the design process, and their constituents may be novel. Schema integration in CE environments can have several purposes:

- The integration of the design context schemata into the product schema is aimed at ensuring design consistency and completeness. This may be complicated by different languages in each of these contexts.
- The integration of product schemata into the integrated resource schema is aimed at ensuring domain semantic integrity. Domain knowledge is essential in guiding this process. For example, when innovative designs are created, how are these designs integrated into the resource schema? The ability to understanding why and how the design was created, as well as the design function, is useful in this regard.
- The integration of domain-specific resource schemata into an integrated enterprise schema is aimed at eliminating redundancy and retaining the domain-specific semantics.
- The integration of the different resource dimensions (data, processes, and knowledge) into the consolidated enterprise schema requires integration of different language constructs and ability to correlate form with function and procedure.

Realization of these virtual integration strategies provides scope for varied research in this area. Relevant approaches may include knowledge-based techniques, parametric, feature-based techniques, model-based or case-based reasoning, geometric or features recognitions, AI, optimization and neural-net techniques, and the ability to mechanize certain activities (including human interaction and decision making). The ability to resolve conflicts and provide automatic creation of the integrated schemata introduces several additional research issues such as schema comprehension, domain knowledge utilization, language translation, data translation, etc. The integration process may rely on other related schemata. For example, data schema integration may utilize the corresponding process and knowledge schemata to resolve conflicts.

5.5.2. Virtual mapping

The ability to view data on demand and in different perspectives must transcend language, schema, and data barriers. Views may be realized by virtual mapping. Mapping may involve data mapping, schema mapping, and language mapping.

- Data mapping. To map between different data sets. For example, one may translate a NURBS surface to a Bezier surface at the data level by applying a data transformation, e.g. a translation algorithm.
- Schema mapping. One may achieve the same translation by applying a mapping between the NURBS schema and the Bezier surface schemata. For example, designer A may be using a solid modeler A and may need to access the results generated by designer B using system B. This translation is currently being dealt with through neutral file translations such as IGES, or neutral schemata such as the PDES schemata. However, the translation between schema A to schema B may also be achieved by integrating the two schemata and extracting a particular view (using schema A or B) from the integrated schema [45]. In this case, the schema stores the data and becomes the view relative to which translation occurs.
- Language mapping. The local design context schemata may employ different representation languages. How can one navigate between different contexts from a given perspective? A translation between two languages of the same types, such as an ER modeling language and an IDEF 1X [33b] methodology, will require language mapping. In this way, an enterprise may support two designers who work with different modeling languages, and allow them to share information in the native schema.

In the presence of an object-oriented framework, it becomes possible to define the view dimension capabilities as "virtual operations" on the abstraction dimension (schemata, data, languages). The definition and processing of the languages, schemata, and their populations are linked through referencing chains. The virtual integration and mapping provide the road map. The ability of the dictionary to support both these dimensions virtually is a useful idea. The dictionary in this case may serve as a fundamental infrastructure for enabling CE environments.

5.6. Abstract dimension

Current database management system support only one language, and therefore allow manipulation only at the schema and data levels of the dictionary (L3 and L4) [20]. Access to the language level and the construct definition level is currently unavailable and must be built independently. Research in this area must be focused on the ability to support multiple languages and mechanisms to translate across languages.

5.7. Virtual view dimension and the abstraction dimension

The view dimension represents several operations that must be applied on the objects represented in the abstraction dimension. The ability to model the dictionary objects' structural semantics as well as the operational semantics is very attractive. The object-oriented paradigm supports this capability. Further research is required to understand how these aspects may be integrated. For example, is the integration operation associated with the product schema or the local design context schemata?

5.8. Virtual navigation

Often there may be a need to navigate diffent design contexts in concurrent or collaborative environments. Since each context may potentially use a different language, the ability to map between different schemata and different languages is important. This permits a designer to realize views of other design context schemata defined in the same language (schema mapping), or other design context schemata defined in a different language (language mapping and schema mapping) or the same type (e.g. one of data, process or knowledge). A virtual navigation is the ability to realize

views across languages, schemata, and data almost instantaneously.

5.9. Virtual processing

When is a process valid? When is knowledge valid? When is a design representation valid? When these aspects are initially modeled, they may be crude representations, not fully mature. When do these aspects become mature and enter the resource schema, for use in subsequent designs? Virtual processing implies capture and extraction of data, processes, and knowledge for subsequent use. If a dictionary is defined using virtual terms (such as virtual teams, virtual integration, virtual mappings, virtual views, abstract dimensions, and virtual navigation), virtual processing for CE can emerge in some form. A vast history of product and process knowledge are then available to all subsequent product design and development. Many issues remain. How can one extract relevant data, knowledge, or process from an enterprise and resource schemata relative to a given design? For example, during the design process, how will a designer identify and extract relevant aspects of the resource schema? Prasad [46] identifies some system-level paradigms that enable virtual processing.

6. Conclusions

This paper has discussed in detail several characteristics of concurrent engineering environments and abstracted several requirements based on this characterization. An information-based architecture is introduced to support these requirements. This architecture results in the identification of several data, knowledge, and process schemata that together support the design process from a global enterprise and historic perspective. This is suggested in the form of an active information dictionary component that provides essential information-related services for CE. The realization of these services translates into several research issues that must be addressed by the research community. Among others, these include various data, schema, and language integration tasks, meta-schema representation, and manipulation tasks. The application of dictionary schema supports two complementary dimensions: the view dimension and the abstraction dimension. The result is a design data dictionary that functions as a framework component for the centralization of data and its definitions.

References

[1] N. Cross, J. Naughton, and D. Walker, "Design Method and Scientific Method in Design Science Method", Proceedings of the 1980 Design Research Societies Conference, Edited by R. Jacques and A. Powell, Westbury House, 1981.

[2] M.M. Andreasen, "The Use of Systematic Design in Practice". Design Synthesis, Edited by H. Yoshikawa, Elsevier, North-Holland, pp. 138–144, 1985.

[3] H. Grabowski and W. Seiler, "Techniques, Operations and Modes for Functional and Preliminary Design Phrases", Design Synthesis, Edited by H. Yoshikawa, Elsevier, North-Holland, pp. 17–22, 1985.

[4] V. Hubka, "Attempts and Possibilities for Rationalization of Engineering Design", Design Synthesis, Edited by H. Yoshikawa, Elsevier, North-Holland, pp. 133–138, 1985.

[5] J.R. Rinderle and N.P. Suh, Measures of Functional Coupling in Design, ASME Paper 82-PROD-27, 1982.

[6] N.P. Suh, "Development of the Sciences Base for the Manufacturing Field through the Axiomatic Approach", Robotics and Computer Integrated Manufacturing, Vol. 1, No. 3/4, 1984, pp. 397–415.

[7] H. Yoshikawa, "General Design Theory as a Formal Theory of Design", Intelligent CAD, Proceedings of the IFIP TC5/WG 5.2 Workshop on Intelligent CAD, Edited by H. Yoshikawa and D. Gossard, Boston, Massachusetts, October, 1987.

[8] N. Cross, "Anyone for Design? Participation and Performance in Designing", 1987 International Congress on Planning and Design Theory, Boston, Massachusetts, 1987.

[9] K.M. Wallace, "Studying the Engineering Design Process in Practice", 1987 International Congress on Planning and Design Theory, Boston, Massachusetts, 1987.

[10] S.L. Newsome, W.R. Spillers, and S. Finger, (eds), Design Theory '88, Proceedings of the 1988 NSF Grantee Workshop on Design Theory and Methodology, Springer-Verlag, 1989.

[11] K.A. Ericsson and H.A. Simon, Protocol Analysis: Verbal Reports as Data, MIT Press, 1984.

[12] J. Doblin, "What Designers Do", Designer, June 1980, London.

[13] G. Pahl and W. Beitz, Engineering Design, Design Council, London, 1989.

[14] M. Asimow, Introduction to Design, Prentice-Hall, New Jersey, 1962.

[15] G. Broadbent, "The Morality of Design", Proceedings of the 1980 Design Research Societies Conference, Edited by R. Jacques and A. Powell, Westbury House, 1981.

[16] D.G. Ullman, T.G. Dietterich, and L. Stauffer; "A Model of the Mechanical Design Process Based on Empirical Data: A Summary", AI Eng '88 Conference, Palo Alto, pp. 193–215, August 1988.

[17] R.E. DeVor *et al.* "A Methodology for the Simultaneous Engineering of Products and Manufacturing Processes", Proceedings of the NAMRC, SME, 28–30 May 1986.

[18] S.N. Dwivedi and B.R. Klein, 'Design for Manufacturability makes Dollars and Sense", CIM Review, pp. 53–59, Spring 1986.

[19] J. Corbett, "Design for Economic Manufacture", Annals of the CIRP, Vol. 35, No. 1, 1986.

[20] K.M. Gardiner, "Characteristics of an Ideal Manufacturing System", ASME Winter Annual Meeting, Computer Integrated Manufacturing & Robotics, PED-V.13, December 1984.

[21] J.M. Starkey and G.J. Florin, "Design for Manufacturability", ASME Paper, 86-DET-121, New York, 1986.

[22] F. Turner and D.L. Carter, "Manufacturing Influence on Design and Its Importance to Successful Automation", Autofact Europe, SME, Geneva, 13–15 September 1983.

[23] H.M. Burte and A. Copolla, "Unified Life Cycle Engineering", Proceedings of the Annual Reliability and Maintainability Symposium, 1987.

[24] H. Bloom, "Design for Manufacturing and the Life Cycle", Design Theory '88, Proceedings of the 1988 NSF Grantee Workshop on Design Theory and Methodology, 1988.

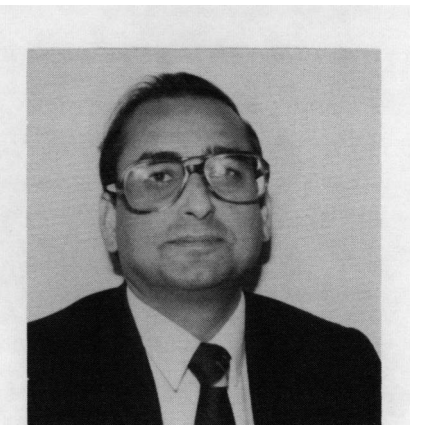
[25] DICE DARPA Initiative in Concurrent Engineering: Second National Symposium on Concurrent Engineering, February 1990, Morgantown, West Virginia.

[26] M. Kinstrey *et al.*, "PPO Management: CE Enabling Technology", Selected Technical Papers, CERC, West Virginia, 23 April 1991.

[27] D. Sriram *et al.*, "Knowledge-Based System Applications in Engineering Design: Research at MIT", AI Magazine, Fall 1989.

[28] R.H. Johnson, "Engineering Data Management P What's Needed and Expected for the 1990's", ASME Computers in Engineering, 1989.

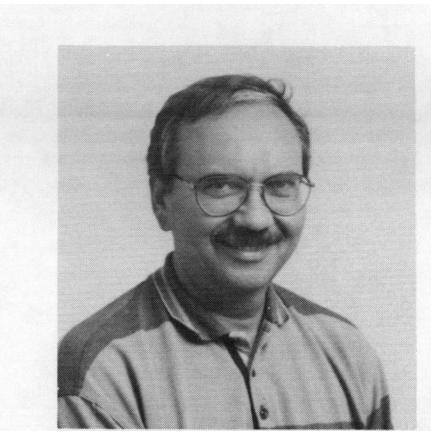
[29] B.J. Gray *et al.*, "Establishing Requirements for Data Management Control Applications to Achieve a Successful System Implementation and Configuration", ASME Computers in Engineering, 1989.
[30] PDES: Product Data Exchange Specification. First Working Draft, US Department of Commerce, National Technical Information Service, December 1988.
[31] M. Hardwick *et al.*, "ROSE: A Database System for Concurrent Engineering Applications", Proceedings of the Second National Symposium on Concurrent Engineering, Morgantown, West Virginia, 7–9 February, pp. 33–65, 1990.
[32] R.S. Morenc and R.M. Rangan, Object Oriented Databases in the Engineering Enterprise, SDRC Internal Paper, Milford, Ohio, 1991.
[33a] IDEF0: Integrated Computer Aided Manufacturing (ICAM) Architecture, Part II, Vol. IV, P Functional Modeling Manual (IDEF 0), Softech Inc., Waltham, Massachusetts, 1981.
[33b] IDEF 1X: Integrated Information Support System (IISS) Information Modeling Manual, Extended (IDEF 1X). D. Appleton Company, Inc., Manhattan Beach, California, 1985.
[34] R.J. Brachman and H.J. Levesque, Readings in Knowledge Representation, M. Kaufmann, Los Altos, California, 1985.
[35] T.J. Teorey, D. Yang, and J.P. Fry, "A Logical Design Methodology for Relational Databases Using the Extended Entity-Relationship Model", ACM Computing Survey, Vol. 18, No. 2, June 1986.
[36] DMCS: Data Management and Control System, Structural Dynamics Research Corporation, Milford, Ohio, 1992.
[37] R. Hull and R. King, Semantic Database Modeling: Survey, Applications, and Research Issues", ACM Computing Surveys, Vol. 19, No. 3, September 1987.
[38] T. DeMarco, Structured Analysis and System Specification, Prentice-Hall, New Jersey, 1978.
[39] J. Rambaugh *et al.*, Object-Oriented Modeling and Design. Prentice-Hall, New Jersey, 1991.
[40] S.H. Gadre, "Building an Enterprise & Information Model", Database Programming and Design, December, 1987.
[41] IRDS: ANSI X3, 138–1988, Information Resource Dictionary System (IRDS), American National Standard Institute, New York, 1989.
[42] ANSI: ANSI/X3/SPARC, Study Group on Data Base Management Systems, Interim Report FTD 7, ACM, New York, 1975.
[43] J.A. Larson, S.B. Navathe, and R. Elmasri, "A Theory of Attribute Equivalence in Databases with Application to Schema Integration", IEEE Transactions of S/W Engineering, Vol. 15, No. 4, April 1989.
[44] C. Batini, M. Lenzerni, and S.B. Navathe, "A Comparative Analysis of Methodologies for Database Schema Integration", ACM Computing Survey, Vol. 18, No. 4, December 1986.
[45] R.M. Rangan, "An Object Oriented Dictionary based CAD/CAM Data Exchange Environment ASME Paper 92-WA/EDB-4, ASME Winter Annual Meeting, Anaheim, California, 8–13 November 1992, New York.
[46] B. Prasad, "On Systems Paradigms", International Journal of Systems Automation: Research & Applications, Vol. 2, No. 1, pp. iii–iv, 1992.



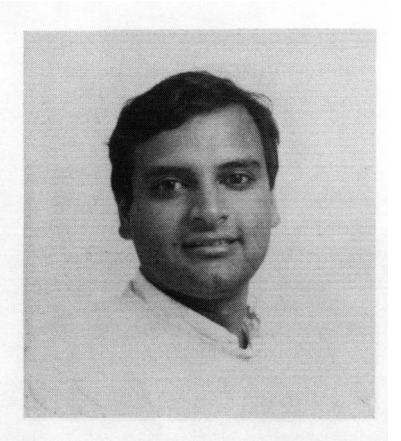
Dr Prasad is a Senior Engineering Staff member at Electronic Data Systems (EDS), a subsidary of General Motors, where he is in charge of the Technology Support Group. Dr Prasad's research interests include CAE, structures AI, intelligent vehicle/highway systems, structural optimization, systems identification, CAD/CAM automation, and simultaneous or concurrent engineering.

He has written or coauthored over 70 technical publications, including four books. His most recent book is *CAD/CAM Robotics and Factories of the Future*, 1989, Springer-Verlag. He has served as editor for several additional texts, monographs, and proceedings. He supports professional societies in several editorial and organizational roles. He served as the General Chairperson on two international conferences: ISPE's CARS & FOF '88 Conference held in Detroit, Michigan (1988) and the 31st AIAA/ASME/ASCE/AHS/ASC Structural, Structural Dynamics, and Materials Conference (SDM '90), held in Long Beach, California (1990). He has received three awards: AIAA's Survey Paper Citation Plaque and Award (1982), a NASA Award and a Certificate for Creative Development of a Technical Innovation on "PARS"—Programs for Analysis and Resizing of Structures (1981)—and the ABI (American Biographical Institute) Commemorative Medal of Honor (1987).

Dr Prasad earned his Ph.D. from Illinois Institute of Technology, Chicago, and a Degree of Engineering from Stanford University, California.



Mr Roger Morenc is currently the Technical Director for the SDRC Engineering Services Division. Roger holds a B.S. in Phycology from Wayne State University. Before joining SDRC, he had worked with the Ford Motor Company and General Motors. Roger has over 25 years of experience in computer application systems architecture design, enterprise modeling, and integration technology. He is currently involved in PDES-related activities and is addressing future architecture issues in the data management and control areas from the standpoint of data dictionary and object-oriented technology. He has spent several years working with a large Japanese automotive company to help design an integrated system that would support their vehicle design process. Roger's past project work also brings in practical experience from the concurrent engineering environment.



Ravi Rangan is a Staff member within the Integration Services organization of SDRC Engineering Services. He holds a Ph.D. in Mechanical Engineering from the Georgia Institute of Technology with a specialization in engineering data management and design automation. He serves on the executive steering committee of the ASME engineering database program and has chaired several ASME conference sessions in the area of research advances in engineering and product data management. Ravi has consulted for several major engineering and computer companies in the areas of engineering data management and design automation. Ravi's research interests include information management in design and manufacturing environments, design automation, engineering data management, design theory, and engineering data management and modeling methodologies. He is a member of ASME, ACM, IEEE, Tau Beta Pi, and Pi Tau Sigma.