Artificial Intelligence for Engineering Design, Analysis and Manufacturing (1999), *13*, 185–204. Printed in the USA.
Copyright © 1999 Cambridge University Press 0890-0604/99 $12.50

Enabling principles of concurrency and simultaneity in concurrent engineering

BIREN PRASAD

Unigraphics Solutions, Knowledge-based Engineering (KBE) PBUs, CERA Institute, P.O. Box 250254,
West Bloomfield, MI 48325-0254, USA.

(Received May 20, 1997; Revised October 27, 1997; Final Revision November 20, 1998; Accepted February 12, 1999)

Abstract

The paper describes a set of seven *fundamental principles for achieving "best concurrency and simultaneity*." The concurrent approach is gaining worldwide attention at this moment. The paralleling of life-cycle activities and process restructuring are being deemed necessary by more and more industries. An automobile product development process example is used in this paper to illustrate many aspects of these seven principles. The principles help the concurrent teams, first, to define how to decompose the product, process and work activities and then, how to arrange these decomposed activities so that "*best concurrency and simultaneity*" can be achieved.

Keywords: CE Principles; Concurrent Engineering; Principles of Concurrency; Time-to-Market; Product Development Teams; Concurrent Product Design; Concurrent Process

1. INTRODUCTION

The concept of Concurrent Engineering was initially proposed as a potential means to minimize the product design, development, and delivery (PD^3) time. Since then, many interpretations of "Concurrent Engineering" (CE) have emerged in literature (Carter & Baker, 1992; Turino, 1992; Parsaei & Sullivan, 1993; Zhang & Zhang, 1995). For instance, Zhang and Zhang (1995) list over 123 papers dealing on this subject. Today, CE is much more encompassing. Expectations range from a modest productivity improvement (Chelsom, 1994) to a complete push-button type automation (Prasad, 1996, 1997), depending upon the views expressed. CE is a paralleled approach—replacing the time-consuming linear process of serial engineering and expensive prove-outs. CE is intended to encourage the product developers, from the start, to consider the "total job" including company's support functions (Carter & Baker, 1992; Turino, 1992).

CE has a major impact on process set up and the way an organization conducts the PD^3 business. As shown by Prasad (1996; Fig. 2.26), concurrent engineering replaces the traditional sequential "over-the-wall" approach with a simultaneous design and manufacture approach with parallel, less interdependent processes. It aims at reducing the total effort in bringing the product from its concept to delivery, while meeting the needs of both the consumers and industrial customers.

The four major phases of product design and development (as shown in Fig. 2.26; Prasad, 1996) have been detailed into nine tracks (shown in Fig. 1) running in parallel. Figure 1 illustrates the different tracks of the development process. These tracks are: mission definition, concept definition, engineering and analysis, product design, prototyping, production engineering and planning, production operation and control, manufacturing, and finally support and delivery. The continuous improvement—"support and delivery"—is an ongoing coordination track that runs for the full life cycle. This track provides normal project-management functions, sequencing, cooperation, and general support to the other tracks. These nine tracks are not unique to any particular product; individual steps and time overlaps may differ from product to product.

2. KEY DRIVERS FOR CE

Earlier, Prasad (1996) chose to divide forces that influence a CE domain into seven agents (called here as 7Ts): tal-

Reprint requests to: Biren Prasad, Unigraphics Solutions, Knowledge-based Engineering (KBE) PBUs, CERA Institute, P.O. Box 250254, West Bloomfield, MI 48325-0254, USA. E-mail: ⟨prasadb@ugsolutions.com⟩

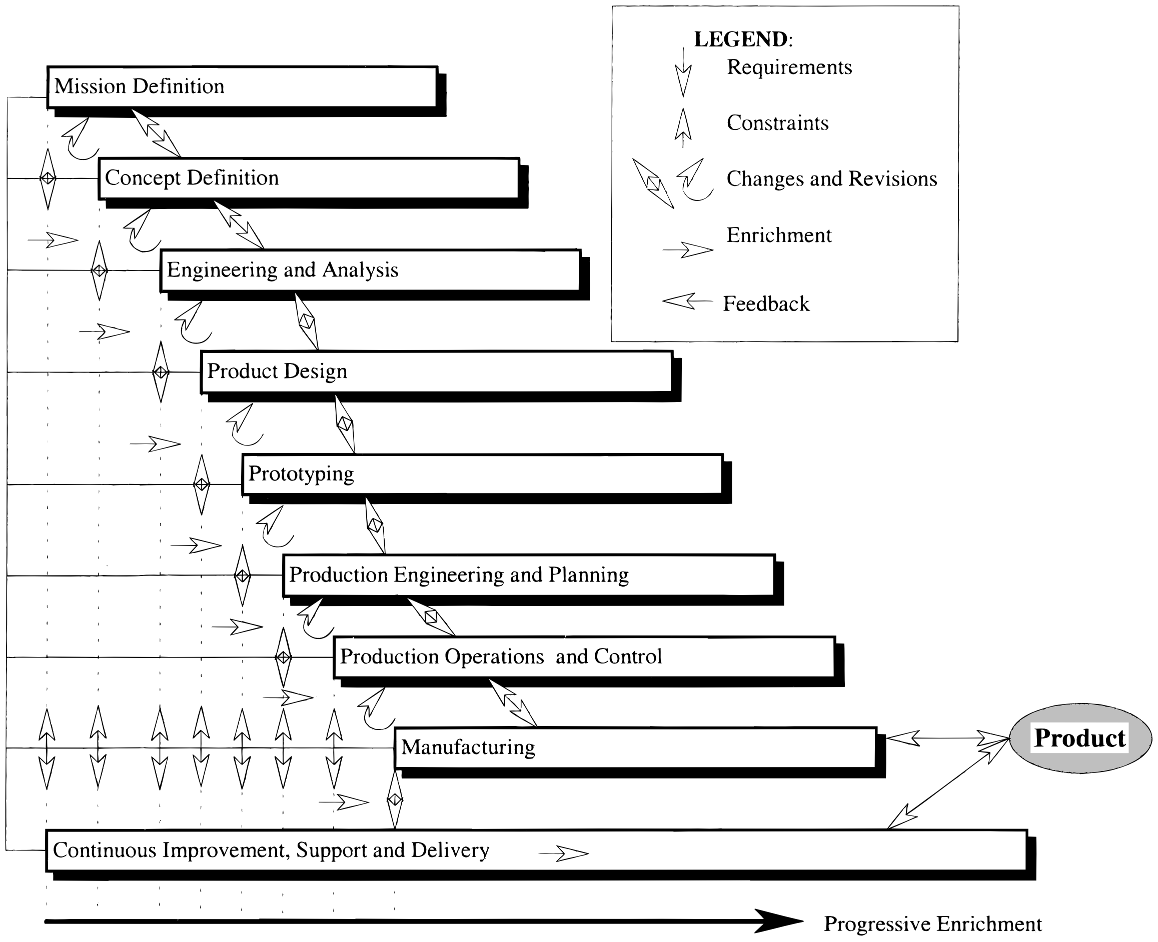

Fig. 1. Showing concurrency during phases of product design and development process.

ents, tasks, teams, techniques, technology, time and tools. One of the primary team issues is the decomposition of tasks. Another team issue is its composition. *Teams* are often used to cooperatively solve the problem. *Technology* issues arise from increased needs for higher operational efficiency and effectiveness. Examples of popular technologies in CE are soft prototyping, visualization, product data management, design for X-ability, multimedia, electronic data interchange (EDI), etc. *Tools* mean software, hardware, and networks that make CE practical in today's world of multinational corporations, multipartner projects, and virtual corporations. From the *time* point of view, CE is an initiative of the product-development community that has the goal of reducing the length of the product design and manufacturing cycle time by allowing teams of engineers to develop design modules concurrently from their perspectives (Pennell & Slusarczuk, 1989). *Training* also plays an important role in CE. A popular word in the business press is re-engineering, meaning, in short, revamp the processes by which one satisfies customers needs.

Timing is an important consideration in product design, development and delivery (PD^3) system. A lot rides on timing of decision making and problem discovery. Approximately 80% of a product's life-cycle cost is driven by decisions made in the first twenty percent of the program effort (DARPA, 1987, 1988).

Once a PD^3 process is decomposed into a set of tracks, and a track is decomposed into a set of activities, they become one full spectrum of steps leading to a product realization. The staggering of their (steps') start points and overlaps is indicative of partial information sharing. Orders are indicative of their precedence. The amount of overlap between any two consecutive activities is indicative of the degree of dependency that may exist between them (Krishnan, 1993). In general, there will be a greater affinity (close relationship and dependence) between pairs of activities, which are adjacent to each other. The farther away the activities are positioned from each other, the lesser would be the degree of affinity or the need for information transfer among them. For example, a *mission definition track* would be more closely related to *concept definition track*, but would have little bearing with activities such as those belonging to a *manufacturing track*. Similarly, *manufacturing track* would be closely related to *production operations and control track*, but would be less sensitive to activities belonging to farther tracks such as an *engineering and analysis* track. If the tracks and activities are completely independent, they all can be aligned along the left margin of the diagram (Fig. 1) keeping the precedence intact. The time-to-market in that case would be dominated by tracks that take the longest time to finish. This is a case of a true “Simultaneity” or a “Simultaneous Engineering (SE)” situation.

2.1. Measure of concurrency (MOC) or overlap

Let us denote the activities in a track, A-set, as a_1, a_2, a_3, …, a_{i-1}, a_i, a_{i+1}, … a_{n-1}, a_n, where a_i is the ith activity, and A-set is the activity set:

$$\textit{A-set} \equiv \{a_1, a_2, a_3, a_4, \ldots, a_i, a_j, a_k, \ldots, \ldots a_n\}. \tag{1}$$

An activity, a_i, itself can be a set of smaller tasks that an activity can be decomposed into. Let us also assume that these activities are arranged concurrently, meaning their start and end times are staggered. If we denote (Fig. 2):

ts_i as the start time, the time when an ith activity, a_i, starts;

te_i as the end time, the time when an ith activity, a_i, ends. (2)

Then, duration of an ith activity, also called the lead-time, d_i, can be expressed as:

$$d_i = (te_i - ts_i). \tag{3}$$

If we denote c_i as the “measure of concurrency” between any two consecutive activities, a_i and a_{i-1} the measure of concurrency or overlap can be expressed as follows:

$$c_i = 1 - (\,ts_i - ts_{i-1})/d_{i-1}, \tag{4}$$

where d_{i-1} is the duration of an activity a_{i-1}. Using Eq. (3), d_{i-1} can be expressed as

$$d_{i-1} = (te_{i-1} - ts_{i-1}). \tag{5}$$

$(ts_i - ts_{i-1})$ is the time-delay in the start of an activity, a_i, with respect to its predecessor activity a_{i-1}.

The above definitions yield the following characteristics. If the two activities, a_i and a_{i-1}, are arranged such that

(a) they run sequentially or serially, then $ts_i = te_{i-1}$ and $c_i = 0$; and (6)

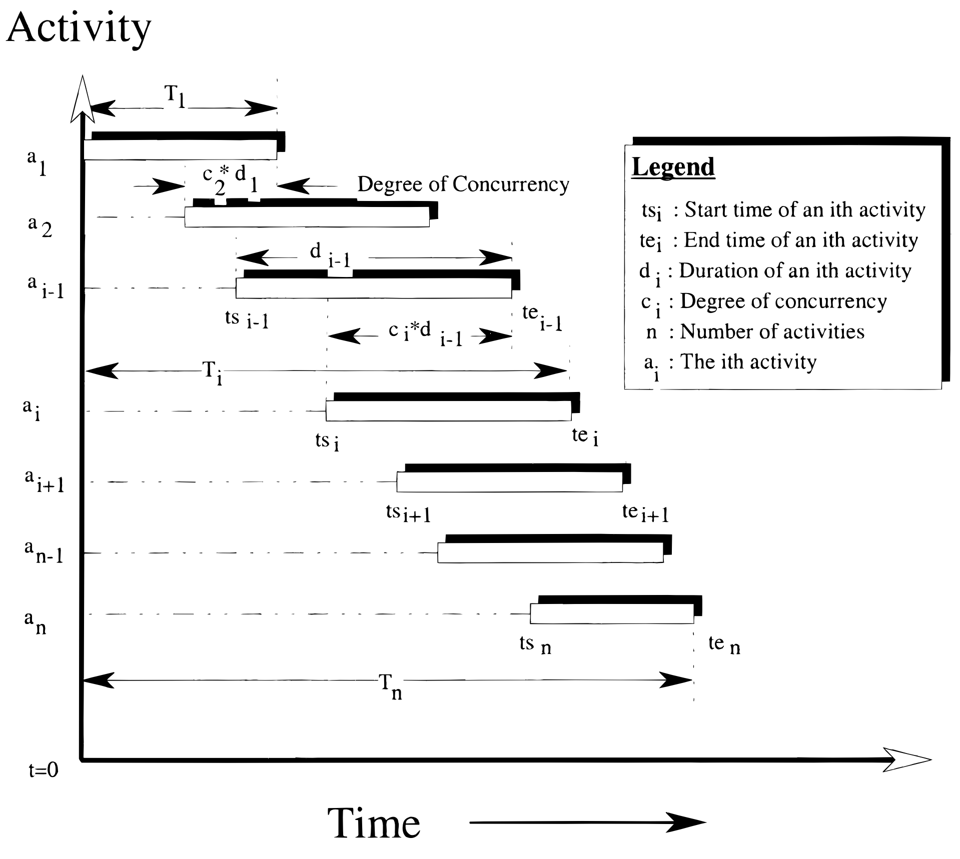

Fig. 2. Defining measures for concurrency and overlap.

(b) the two activities completely overlap, then $ts_i = ts_{i-1}$ and $c_i = 1$. (7)

For a partial overlap, c_i may range in between 0 and 1. Based on the definitions of c_i, the cycle-time for designing and developing a product, whose activities, a_i through a_n, when arranged in parallel, can be expressed as follows:

If T_i is the clock time of an ith activity. The clock time is the time an ith activity, a_i, takes from start ($t = 0$) to its finish. Following this definition, then $T_1, T_2, T_3, T_4, \ldots T_n$, can be expressed as

$$\begin{aligned} T_1 &= d_1, \\ T_2 &= \{d_2 + d_1 * (1 - c_2)\}, \\ T_3 &= \{d_3 + d_1 * (1 - c_2) + d_2 * (1 - c_3)\}, \\ T_k &= \{d_k + d_1 * (1 - c_2) + d_2 * (1 - c_3) \\ &\quad + d_3 * (1 - c_4) + *** + d_{k-1} * (1 - c_k)\}, \end{aligned} \tag{8}$$

$$T_n = \left\{ d_n + \sum_{i=1}^{i=n-1} d_i * (1 - c_{i+1}) \right\} \tag{9}$$

The Eqs. (8–9) provide a basis for computing the total product development time, T_k^*. If the activities in the A-set are arranged concurrently. The term "$d_i * (1 - c_{i+1})$" represents a time delay, a fraction of the time-duration (d_i) when two activities, a_i and a_{i+1}, do not overlap with each other.

$$T_k^* = \max[T_1, T_2, T_3, T_4, T_5, T_6, T_7, T_8, T_9, \ldots T_k, \ldots, T_n] \tag{10a}$$

$$\text{Or } T_k^* = \max[T_i];\ \forall\ i\ ; i = 1, 2, \ldots, k, \ldots, n. \tag{10b}$$

It is clear from the Eqs. (9) and (10) that the total cycle-time, T_k^*, depends upon the duration of each activity, d_i and its "degree of concurrency or overlap," c_i.

- The shortest cycle time, T_k^S can be reached when $c_i = 1\ \forall\ i, i = 1, 2, \ldots, n$. That is, when each and every one of the activities, a_1 through a_n is scheduled to start simultaneously. This means that the starting point of each activity is aligned to the leftmost point as far as possible in Figure 2.
- The longest cycle-time, T_k^L will occur when $c_i = 0\ \forall\ i$, i=1, 2, …, k, …, n. Meaning, when each and every one of the activities, a_1 through a_n, run serially.

The idea of "*best concurrency and simultaneity*" is to align each activity step to the farthest left of the diagram (Fig. 1) satisfying the following 3**M**s:

(a) **M**aintain the precedence of the activities in the A-set that were decomposed, that is,

$$t_{si+1} \geq t_{si} \quad \text{for} \quad \forall\ i, i = 1, 2, \ldots, n - 1. \tag{11}$$

(b) **M**aximize the horizontal overlap between the consecutive activities, a_i and a_{i+1}, that is,

$$\text{maximize } (d_i * c_{i+1}) \quad \text{for} \quad \forall\ i, i = 1, 2, \ldots, n - 1. \tag{12}$$

(c) **M**aximize the independence of the decomposed activities, a_i and a_{i+1} in the A-set, meaning,

$$a_i \cap a_{i+1} \approx 0. \quad \text{for} \quad \forall\ i;\ i = 1, n - 1, \tag{13}$$

where ∩ denotes an intersection of the adjacent activities in questions in the A-set. This is based on the understanding that an activity, a_i, itself can be a set of smaller tasks that an activity can be decomposed into (Prasad, 1996). In many practical situations, depending upon the level of decompositions used, it is possible to create a set of activities, which are not dependent on the rest. The three terms ts_i, d_i, and c_{i+1} are defined earlier in Eqs. (2) through (4).

This paper describes a set of seven *fundamental principles for achieving this "best concurrency and simultaneity*." An *automobile-product-development process* example is introduced first in the next section to introduce the basic nomenclatures and terminology. The same terms are used in the remainder of the paper to illustrate the abstract nature of these fundamental principles. The seven principles help the teams define:

(a) how to decompose the activities in the A-set and then

(b) how to arrange these decomposed activities in the A-set so that "best concurrency and simultaneity" can be achieved.

The concurrent approach is gaining worldwide attention at the moment. The paralleling of life-cycle activities is being deemed necessary by more and more industries to adapt quickly to changing market conditions and to achieve shrinking time-to-market targets. Section 3 describes an automobile-manufacturing-process example, and Section 4 extracts a number of key principles from this example to maximize c_i, minimize d_i, and minimize T_k^*.

3. AN AUTOMOBILE MANUFACTURING PROCESS EXAMPLE

Most automotive companies introduce a new car model every 2 to 3 years at a cost of a billion dollars per vehicle (Prasad, 1996). The new-car-development program in the United States now ranges in between 3 to 4 years, whereas in Japan it takes less than 3 years (Tsuda, 1995). Development is generally the responsibility of the operating platform groups, with the new product sold by one or more of their marketing units. The major elements of an automobile are:

- *Outside Body:* It includes major designs for outside body parts, structures, such as roof, moveable roof, body glass, quarter panel, fender, A-pillar, B-pillar, C-pillar, decklid, trunk, Apron, shot-gun, vehicle tools, paints, etc. They are often designed by a staff group.
- *Styling* is done by a central design staff with support from components' divisions and outside suppliers. They come up with the design of the outside contour, "look and feel," mostly from aesthetic considerations.
- *Detail design* of the parts and body panels are done by the CAD/CAM shop contractors.
- *Analysis* is proposed by the engineers but often performed by the analysts on contracts.
- *Tooling and dies* are handled by process engineers internally and by outside prototype shops.

The following systems are designed by a prototype shop, one or more of the components' groups, or first- and second-tier suppliers (Prasad, 1996):

- Interior Systems: instrument panels, air bag, steering wheel, door trims, door modules, and related hardware, latching, window regulators, power closures, power sliding door, seat systems—seat trim, adjusters, recliners, frames, head rest, arm support, etc.
- Vehicle Wiring Systems: ignition wiring, fiber-optic data transmission, fiber-optic lighting distribution, electrical/electronics connection, multiplex, wire harness, integration of electrical electronics into modular structures, temperature sensors, electronic modules, and switches.
- Brake Systems: antilock brake, traction control, intelligent brake control, power brake assemblies, electric brake, disc and drum, corner assemblies, wheel spindle bearings, knuckles, calipers, and rotors, etc.
- Suspension Systems: suspension assemblies, controlled suspension, structural composites, integrated chassis, module suspension, powertrain mounts, etc.
- Climate Control Systems: heating, ventilation and air conditioning, condensers, compressors, accumulator dehydrators, evaporators, heater cores, etc.
- Engine/Transmission Cooling Systems: radiators, oil coolers, engine cooling module, etc.
- Engine Management Systems: air fuel, ignition, fuel handling and evaporative emissions, electronic control modules and algorithms, exhaust system, valve train, etc.
- Energy Management System: power generation and storage, batteries, generators, sensors, and solenoids, electric vehicle, etc.
- Lighting Systems: forward lighting, signal lighting, center high mounted stop lamps, distributed lighting, high intensity discharge lamps, etc.
- Vehicle Control Systems: advanced steering, power steering, pumps, gears and hoses, variable effort steering, standard and adjustable steering columns, intermediate steering shafts, etc.
- Driveline Systems: axles, front and rear, propshaft, half-shaft assemblies, constant velocity joints, intermediate drive shafts, boot seals, etc.
- Engine: structural, crankdrain, valve train, cam drive, accessory drive, lubrication system, cooling system, air intake, PCV, combustion, exhaust, sealing and fastening assembly, etc.
- Transmission Systems: transmission (auto and manual), torque converter, case and cast components, gears and shafts, mechanical components (clutches, free wheelers, chain drive), cooling and lubrication, sealing and fastening, dress components, transfer case, etc.
- Powertrain controls and diagnostics: diagnostics, electrical, electronics, software, driver display, driver controls, sensors, actuators, and other misc. systems.
- Others: Entire program is supported by thousands of second and third tier suppliers that provide interior parts, bolt-in-parts, and hundreds of other components and materials.

An operating group or platform normally is responsible for a particular line of automobiles, small or sporty cars, for example. A corporation generally has engineering facilities in multiple cities and has assembly plants in multiple countries, (such as United States, Canada, and Mexico). Many of its plants are spread throughout the United States (e.g., the Midwest and South). Operations within an operating group are supported by an extensive vendor network or a supply chain.

During this 3-year cycle of a new vehicle or product development process, an operating group or a platform must also build other car lines. This means, in the current year $199X$–$199X + 1$ (say $X = 7$), manufacturing engineers would be building $199X + 1$ (1998) model cars. While process engineers will work on $199X + 2$ (1999) models, and product engineers concern themselves with $199X + 3$ (2000 if $X = 7$) product lines. Other groups within the company must support these four groups: design groups, process group, manufacturing group, and the operating group. For instance, design-support groups may seek a balance among piece cost, manufacturing, assembly, fuel consumption (mileage), emission and safety regulations. The planning group may balance investments with budgets. Marketing groups may seek competitive concerns, such as styling, vehicle content, quality, and numerous other issues.

These groups are often matrixed to each other to address these concerns. Because many of these groups are independent of each other, no one manager is likely to own the right or control the total program. Funding and control of resources are usually decided through committees. Each group, thus, ends up suboptimizing their own portions of the design with lack of overall coordination between the groups. The problem is typical of a situation where groups have too much independence, but not enough coordination. Systems

Engineering and QFD models are often used to simplify the problem in such cases (Tsuda, 1995).

3.1. How product complexity is handled today

Complexity of the products and of the processes present in the system (e.g., an automobile), often compels a product manufacturer to look for their products and processes breakdown structures. This breakdown structure is necessary to exploit any inherent concurrency, so that the individual activities can be overlapped and thus run in parallel. Physical-based decomposition is one way to achieve this parallelism, as shown in Figure 3. Perspectives represent the first level of physical-based description (PhD). PhD is also commonly referred to as "*Product Holistic Decomposition*" or PhD in short (Prasad, 1996). Other possible levels of PhD (into which product can be decomposed to exploit concurrency) are: hierarchy, multiplicity, alternatives, characteristics, and projects (see Fig. 3). Please note "decomposition" is not intended here to mean clustering the problem parameters in different ways. A typical case of this type occurs when a problem is decomposed simultaneously into a num-

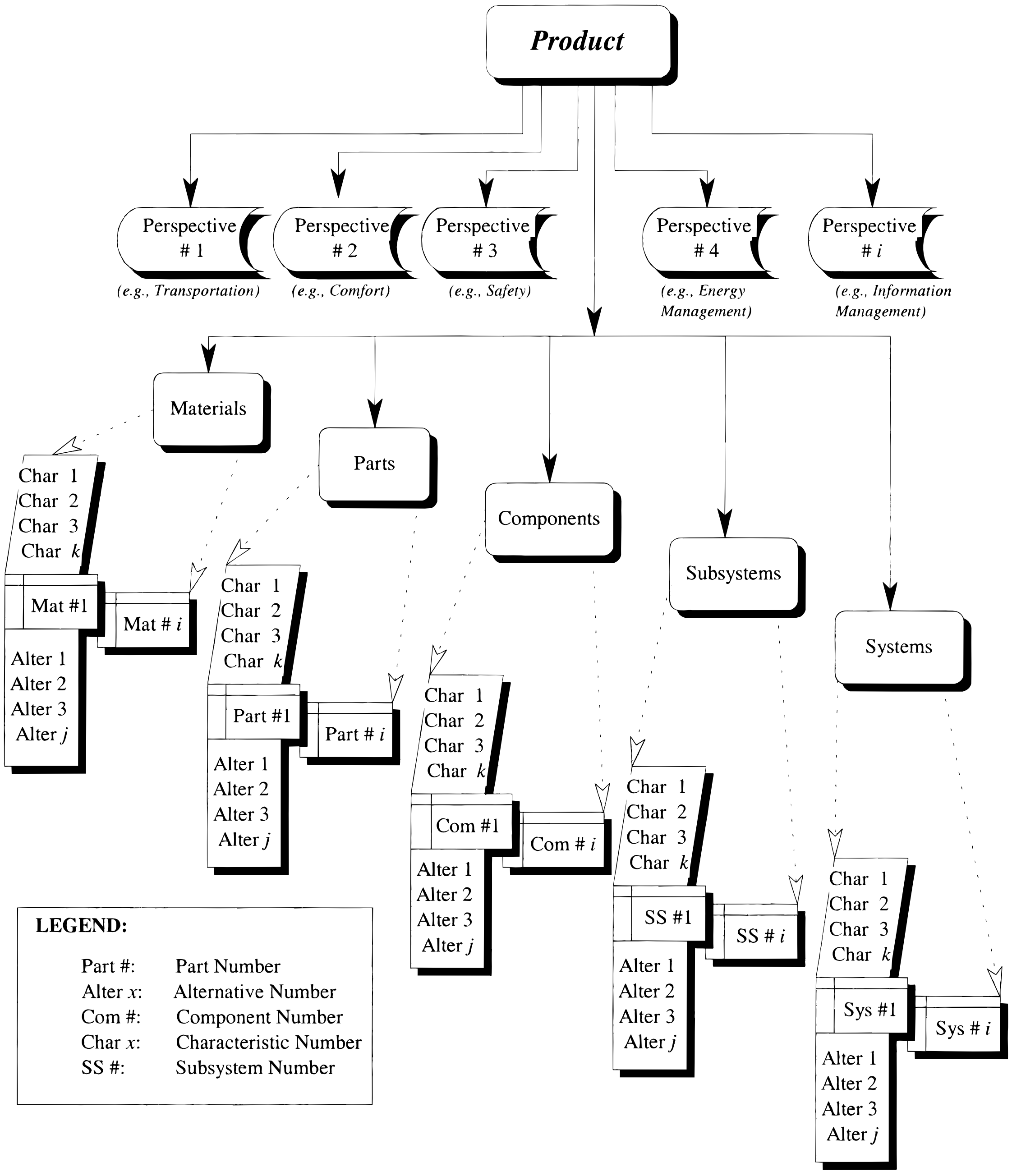

Fig. 3. Areas of concurrency during product synthesis (bottom-up representation).

ber of ways (such as program phase, subsystem, and discipline). The term "*decomposition*" is used here to mean "*product holistic decomposition (PhD)*." Parameters are not fragmented into separate decomposed sets. All parameters belonging to a particular class or a part family stay together (after decomposition) and collectively influence the decision-making process.

$$\begin{aligned}\text{PhD A-tree} \equiv [&\{\text{A-perspectives}\}, \{\text{A-hierarchy}\},\\ &\{\text{A-multiplicity}\}, \{\text{A-alternatives}\},\\ &\{\text{A-characteristics}\}, \{\text{A-projects}\}], \qquad (14)\end{aligned}$$

where the curly bracket, $\{x\}$, denotes the activity-set of quantities of type x and is governed by Eq. (1). An activity tree (short form is A-tree) is comprised of several activity sets. The following are examples of some typical decomposition scenarios of a PhD A-tree (Prasad, 1996):

A. *Perspective*: A design problem usually involves multiple perspectives. Each may have its own set of constraints and could interact with each other. At the highest level, different work groups can work in parallel on separate competing perspectives of product life-cycle concerns. Such concerns are often required for product evaluation or assessment. These perspectives include the intellectual process of commonality or class hierarchy between different families of products, such as:

- Size-wise (large, medium, or small)
- Model-wise (sporty, coupe, or luxury)
- Engine-wise (4-cylinders, 6-cylinders, 8-cylinders, etc.)

$$\begin{aligned}P_{\text{size-wise}} &\equiv \{\text{extra-large, large, medium, small}, \ldots\}\\ P_{\text{model-wise}} &\equiv \{\text{luxury, sporty, coupe}, \ldots\}\\ P_{\text{engine-wise}} &\equiv \{\text{4-cylinder, 6-cylinder, 8-cylinder}, \ldots\},\end{aligned} \qquad (15)$$

where, the letter P denotes the perspective. A class-hierarchy can be based on the

- usage [passenger car, commercial vehicles (jeep/trucks) and bus]. One commonly used perspective used during organization and management of information is by a combination of size, usage, and marketing perspectives. This for automotive industry has transpired into a triad, spanned by three axes, as shown in Figure 4.
- The vertical axis shows the division by platform types. The common platform types for automobiles are small car, mini-van, large car, Jeep/Truck. The subclasses of Jeep/trucks can be jeep, pick-up, vans, tractor-trailer, multi-purpose ve-

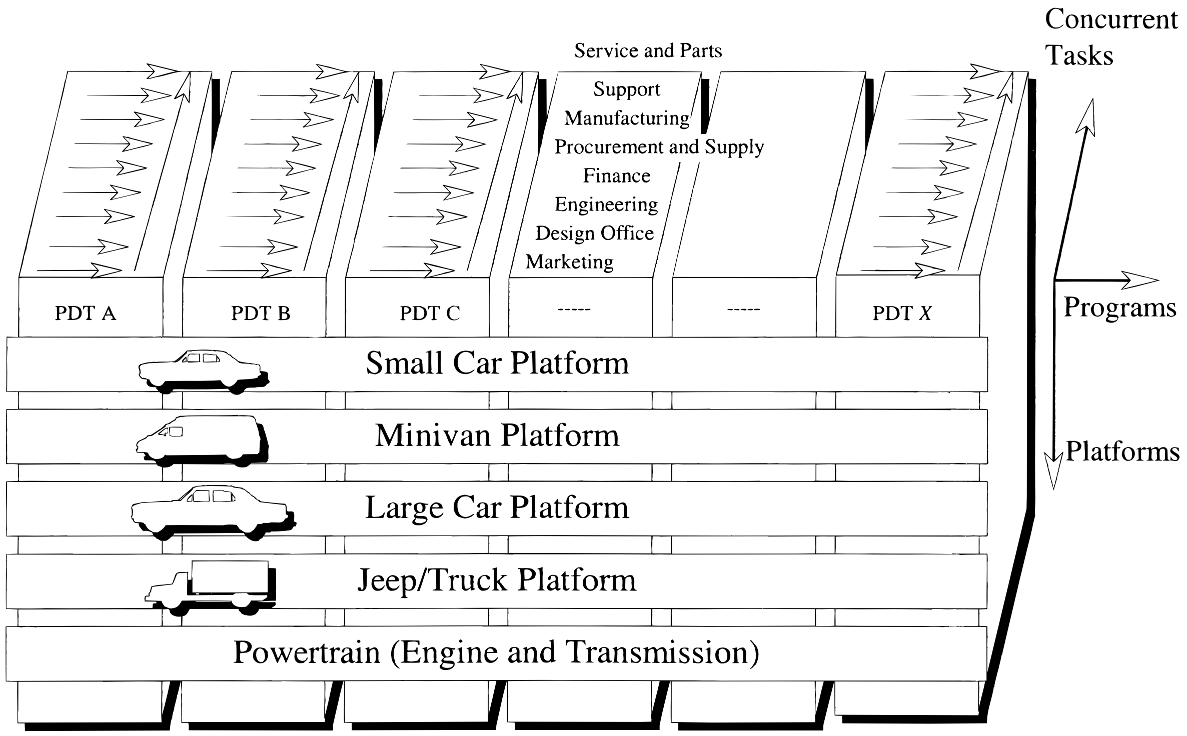

Fig. 4. A three-dimensional concurrent triad (automobile manufacturing example).

hicles, etc. Powertrain (Engines & Transmissions) is also categorized as a type of a platform, since it cuts across all major car lines.

- The horizontal axis lists the division by program types, such as P90, W, or C car programs. There is usually a Product Development Team (PDT) responsible for each car program. They are generically named here as PDT A, PDT B, PDT C, PDT D, …, X, etc.
- Concurrent to each program there usually is a third dimension, now commonly called centers. These centers perform concurrent tasks or activities such as marketing, design office, engineering, finance, procurement & supply, manufacturing, service & parts, and product support. They are either dedicated services to a program or are matrix across several programs: such as PDT A, PDT B, PDT C, etc.

Defining a product breakdown structure (PtBS) tree to perspectives attaches additional meaning and order to the complex product design process (Tsuda, 1995). A passenger automobile's basic product structure (e.g., four wheels, 4–8 cylinders, reciprocating gasoline engine in front, round steering wheel, 2–4 doors, 1–2 rows of seats, interior instrument panel, trims, etc.) has not changed much in three decades. Many new models have been introduced, inheriting the basic concept of the automobile.

B. *Hierarchy*: The physical product or the "product system" may be divided into several logical hierarchical blocks or classes depending upon its complexity. The advantage of this logical division is that different people can work in parallel in these different hierarchical blocks. The associated teaming between groups of people in a large manufacturing organization is discussed in Section 4. If separate teams are assigned to each class and subclass, they can work concurrently. PtBS example for an automobile class is shown in Figure 5.

$$\text{PtBS A-tree} \equiv [\{\text{System}\}, \{\text{Subsystems}\}, \{\text{Components}\}, \{\text{Parts}\}, \{\text{Materials}\}, \{\text{Characteristics}\}]. \quad (16)$$

The PtBS activity-tree can be superimposed on work groups involved in the system design, with supporting subteams dealing with subsystems design, and another set of subteams handling the remainders, such as components, parts' design, materials, form features, etc. A nested routing work-flow model can be drawn starting from the bottom and showing the activities of each of the PtBS's trees leading up to the system-flow model as information builds up. Some dependencies can exist between the branches.

An important job of the CE work groups is to recognize and manage interdependency between the PtBS nodes. Establishing common interface standards for communications and dictionary definitions (standard) of problem parameters and checkpoints can allow parallel groups to work concurrently.

Checkpoints are essential to ensure the smooth coupling of completed activities. This is accomplished by staggering the product breakdown structure (PtBS) tree as shown in Figure 3. For example, the system level activities can only begin when activities for subsystem track are already well underway. The subsystem level activities can begin only when tasks for component track are well underway and so on.

PtBS organizes a product hierarchy by using a stepwise refinement and differentiation technique. Stepwise refinement adds hierarchy to the structure and differentiation adds details at a particular level. Product or process features, materials, attributes, and parameters provide the lowest level of hierarchical abstraction. The amount of granularity present at each level is usually a function of the product and process complexity and their knowledge, such as knowledge of objects, functions, design cases and needs. Object knowledge for product (i.e., topology, geometry, etc.) provides attributes, structures, assembly, and their relationships. Functional knowledge produces evidence for hierarchical decomposition (systems, subsystems, components, parts, …, etc.). Design cases or case histories provide additional evidence of breaking the hierarchy into alternatives, characteristics, etc. During the differentiation technique, different characteristics and alternatives can be assigned to a PtBS tree, as shown in Figure 5. The PtBS tree drives the product or the process design to a manageable set of units and nodes that can be independently worked upon by the work groups or the concurrent subteams.

Figure 5 shows how a hierarchy of system decomposition would look if we started with system assembly of an automobile and worked our way down from this top level. Each decomposed element combines with other decomposed elements of about the same level to make up the next larger level. Strategy (what services to render and to whom) and processes (how to convert inputs to outputs and how to deliver outputs to the customer) practically determine expected quality level, productivity, costs, and profitability.

C. *Multiplicity*: Within each different hierarchical group, for example, a part or a component group, multiple parts or components going into the final product may be worked upon simultaneously.

$$\{\text{Parts}\} \text{ of a PtBS A-set} \equiv \{\text{part\_1}, \text{part\_2}, \text{part\_3}, \ldots, \text{part\_}n\}. \quad (17)$$

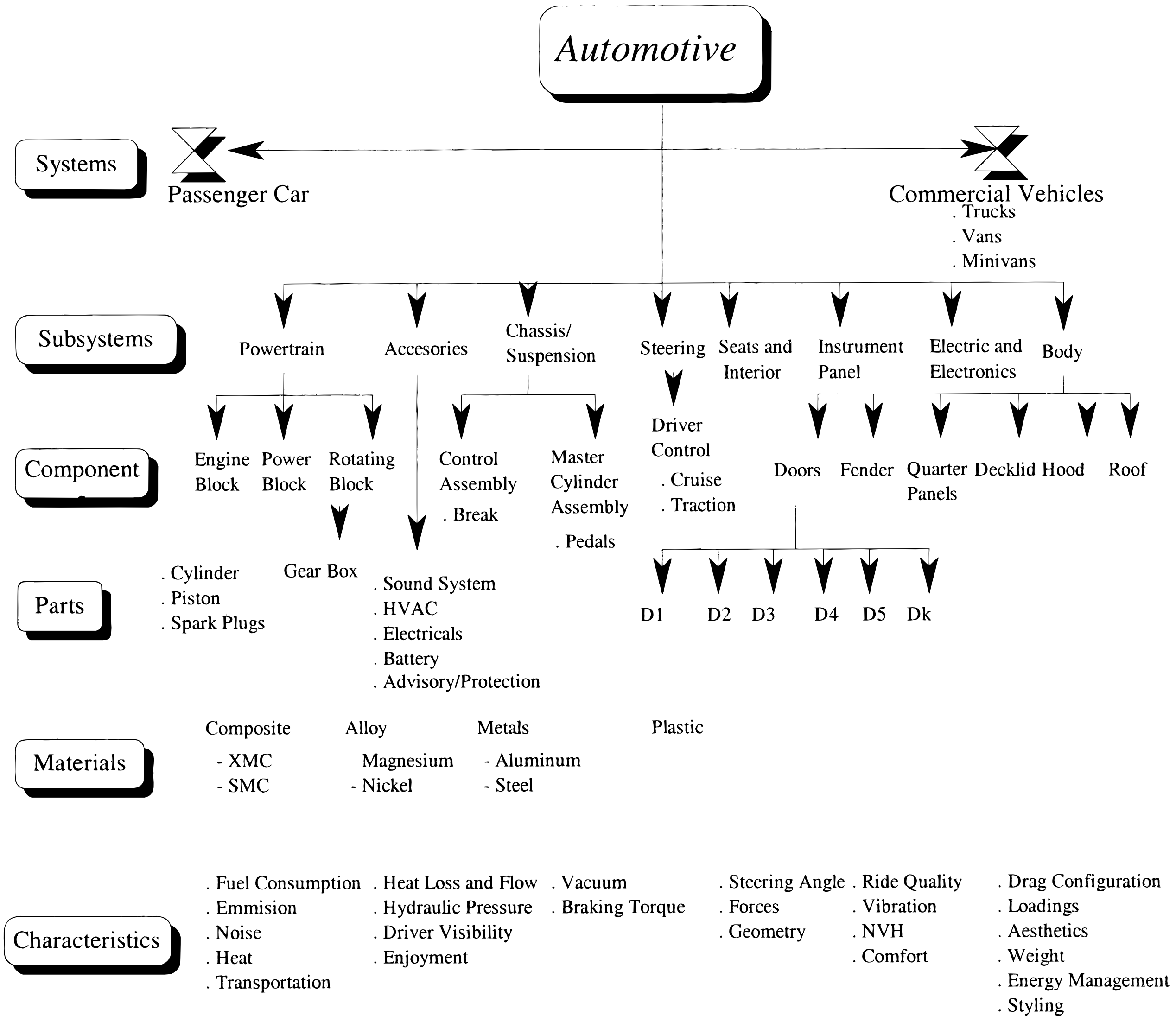

Fig. 5. Areas of concurrency in an automotive top-down decomposition—An example.

Similarly, the teams in the work groups may work concurrently on a multiplicity of models used to represent a multidimensional enrichment of a part. For example, the geometry of parts may be modeled, first in the early design stages using sketches, then by means of a solid model using CAD/CAM tool, and later by an orthographic projection drawing. As such, a decomposed element of a PtBS tree can be a quantified set.

D. *Alternatives*: Within one hierarchy level, a group of designers guided by its hierarchy leader may work on several alternative ideas in parallel.

E. *Characteristics or Aspects*: Each alternative idea may involve integrating some life-cycle aspects, that is, validating its output through compliance from multiple characteristic views. Where, each characteristic view may represent a different life-cycle aspect such as aerodynamics, noise, ride quality; NVH (noise, vibration, and harness), strength, stiffness, energy management, packaging, etc. Each life-cycle concern may further be looked upon from different view points: from a well-defined view points (e.g., rigid body dynamics), to an ill- or vaguely defined viewpoints (e.g., manufacturability). Subteams from different disciplines and background may be needed to support these aspects or viewpoints. These subteams can work in parallel on each characteristic view.

F. *Projects*: Multiple projects, such as predictive analyses, fault-tree analysis (FTA), QFD (Tsuda, 1995), DFMA/FMEA analyses, may be required to evaluate product compliance to functional specifications. Many analysis subteams may be working in parallel to determine the integrity of the design with respect to these specifications. Additional details, increased accuracy, and other aspects of alternative designs may be considered as typical examples for projects.

The next section describes a set of enabling principles of concurrency and simultaneity. The information is extracted from this automobile case history and presented in a generic form to be applicable across many other product families.

4. BEST CONCURRENCY AND SIMULTANEITY

Concurrency and simultaneity are the major force of Concurrent Engineering. There are seven enabling principles to achieve the best concurrency and simultaneity in Concurrent Engineering:

4.1. Principle 1: Parallel work group

Parallel work group is one of the well-known and widely used methods of achieving concurrency. There are many concepts of such parallel work groups have been described in the literature (DARPA, 1988; Prasad, 1996, 1997). Paralleling describes a "time overlap" of one or more activities in the A-set, tasks, etc. CE is structured around multifunctional teams that bring specialized knowledge necessary for the program.

4.1.1. Multidisciplinary team

The multidisciplinary setup, called product development team (PDT), is composed of several distinct technical subunits specializing in a variety of disciplines:

- Product Planners (T_{pp}),
- Product concept engineers (T_{ce}),
- Engineers and analysts (T_{ea}),
- Product designers (T_{pd}),
- Prototype engineers (T_{pe}),
- Production engineering planners (T_{ep}),
- Management & control (T_{mc}),
- Computer integrated manufacturing (CIM) and Assemblers (T_{ma}),
- Delivery & support (T_{ds}) teams:

$$\text{PDT-set} \equiv [\{T_{pp}\}, \{T_{ce}\}, \{T_{ea}\}, \{T_{pd}\}, \{T_{pe}\}, \{T_{ep}\}, \{T_{mc}\}, \{T_{ma}\}, \text{and } \{T_{ds}\}] \tag{18}$$

where, the square bracket signifies union-of sets contained within the curly brackets, and T stands for "talents." In the above, nine concurrent sets of teams are intentionally chosen to show the actual correspondence with each of the nine concurrent tracks of Figure 1. Each track is responsible for developing and integrating its own aspect to the product's lifecycle as the program requires. However, there could be as many activities per track (referred here as A-set) and teams (referred here as PDT-set) as needs arise. For example, experts from the volume production area must be involved in prototype production to identify as early as possible opportunities to improve process reliability. By using this multifunctional team approach to merge design and manufacturing, GE Aircraft engine division reduced design and fabrication lead time for some GE engine components from 22 to 3 weeks (*Machine Design*, 1993).

A product design and development process is not a concurrent-engineering process unless it involves all parties that are responsible for its making, regardless of to whom they administratively report. Subcontracting companies must be included as participants in the CE teams, at least until the interface design requirements have been determined, evaluated, and are firmed up. The distributors or retailers often tell the product manufacturers exactly what the consumer wants. The product manufacturers then are able to communicate upstream to the suppliers what parts or materials they would require to manufacture the product. For the organization to function as a unit and product to compete globally, all participants have to know what is expected from each other and in what time frame. Correct communication links have to be in place. This will ensure a complete integration of the OEM's product needs with supply chain capabilities. With such integration, the subcontractors can influence requirements before it is too late. Requirements can be stated in joint terms that the subcontractors can effectively satisfy, and that they are reasonably stable and unlikely to undergo any significant change.

4.1.2. Inclusion of outside trade partners

An effective inclusion of outside (trade) partners in cooperative development is frequently one of the underemphasized issues related to the implementation of a CE process. In today's environment, because of the growth in the complexity of consumer products and the increased reliance on specialized technologies and methods to manufacture them, partnership has become an increasingly important issue. Companies often rely on outside partners to supply expertise, services, and products in various specialized disciplines. Many examples exist (Womack et al., 1990).

In conjunction with the United States Council for Automotive Research (USCAR), GM, Ford, and Chrysler are working to establish voluntary parts standards on items like light bulbs, car jacks, radiator caps, switches, handles, and other noncompetitive parts to reduce the influx of parts and boost global competitiveness. Taken to its extreme, this will mean the emergence of a concept called "virtual company," where the core company has only a limited staff. They are the financiers or planners of ideas of a product. The major work force is comprised of individuals from various other companies that have the appropriate skills to transform these ideas into a useful product. A list of typical participants of a virtual company is shown in Figure 6.

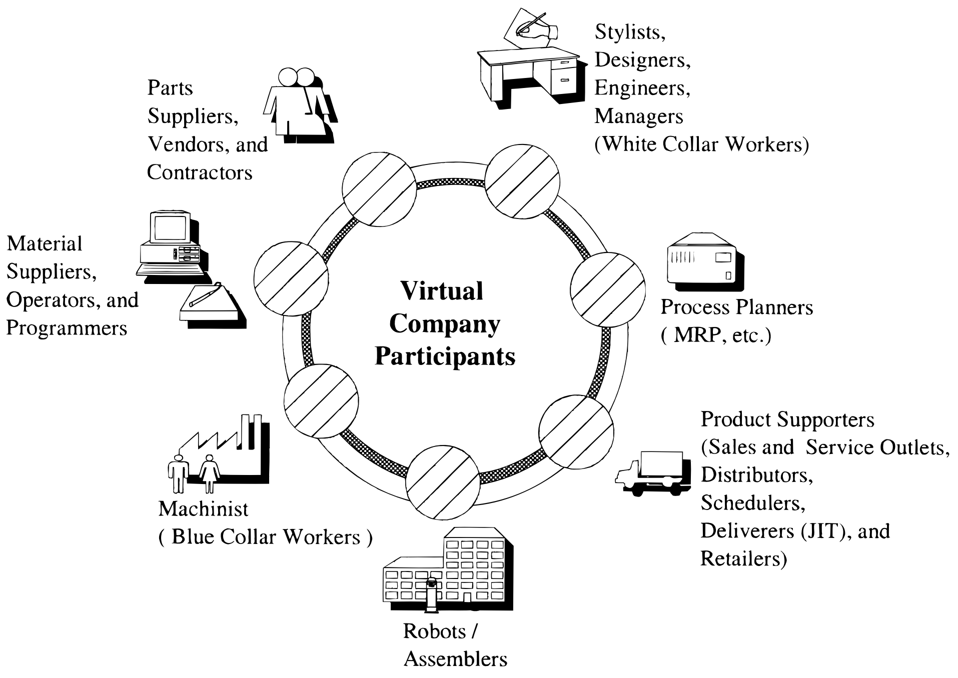

Fig. 6. Typical participants in a virtual company.

They are mostly contract employees. However, they are responsible for delivering the services or products that they are contracted for on time and at budget. There is a global partnership among the participants: the product manufacturers (stylists, designers, engineers, managers, white-collar workers), process planners (Manufacturing resource planning—MRP, Computer-aided Process planning—CAPP), robots/assemblers, machinists (blue-collar workers), parts suppliers (vendors, contractors) or materials suppliers (operators, programmers), and product supporters (sales and service outlets, distributors, schedulers, deliverers, retailers).

$$\text{Virtual\_Company-set} \equiv [\{T_{wc}\}, \{T_{pp}\}, \{T_{sd}\}, \{T_{ra}\}, \{T_{bc}\}, \{T_{cop}\}, \{T_{svc}\}], \quad (19)$$

where

T_{wc} denotes white-collar,

T_{pp} denotes process planners,

T_{sd}: schedulers & delivery,

T_{ra}: robots assemblers,

T_{bc}: blue-collar,

T_{sop}: materials suppliers, operators and programmers,

T_{svc} are parts' suppliers, vendors and contractors,

and where the square brackets denote "union-of" the individual sets. The curly brackets indicate the presence of sets.

It will do little good for a company to adopt a CE environment (or to control its product definition process) without including its trade partners, if a major or significant portion of its production is performed by outside suppliers. Establishing a partnership can be strategically very important. It can eliminate or minimize the needs for in-house inspection. By establishing some type of partnership, where the certification program is a part of the deal one can ensure the delivery of quality incoming materials. In that case, the cost-benefits of inspecting incoming materials and sorting out defective parts for return to vendors must be weighted against the supplier's cost of acquiring defect-free parts. Successful partnership requires a harmonious communication environment characterized by rapid, accurate and "paperless" business transaction. Other claimed benefits of partnership include greater satisfaction to the customer, simplified recycling, fewer computer entries, smaller inventories, and greater economy of scale. The increased use of electronic commerce technologies, such as Electronic Data interchange, or EDI, *via* Wide Area Networks, Value Added Networks, and Electronic Vendor Bulletin Boards are paving the ways of making this partnership painless. They are widely used in the auto industry to exchange purchase orders, shipping notices and payments, particularly with first-tier supply-chain partners that deliver directly to OEMs. A first-tier supplier of instrument panel, for example, may be required to deliver product within a few hours of receiving an order, and deliver it in the assembled order needed on the assembly line. This close partnership has directly reduced inventory industry-wide.

Figure 7 shows a bi-directional sandwiched structure for an Integrated Product Development (IPD) System. In one direction, IPD-set are supported by the customer on the top, and the infrastructure (organization) at the bottom. In a perpendicular direction, PDT-set are sandwiched between product and process on one side, and tools and technology on the other side.

$$\text{IPD-Set} \equiv [\{\text{Customers}\}, \{\text{Product}\}, \{\text{Process}\}, \{\text{Tools}\}, \{\text{Technology}\}, \{\text{Infrastructure}\}, \{\text{PDT-set}\}] \quad (20)$$

where, square brackets indicate union-of several sets. PDT stands for product development team. An example of a PDT-set is defined earlier in Eq. (18). The curly brackets indicate the presence of several sets given in Eq. (20). The infrastructure involves a wide range of disciplines including multi-functional teams, strategic business units (SBUs), culture and practices, business process re-engineering, logistics, finance, information technology, education and training, and synchronous manufacturing organization. As shown in Figure 7, the customers help establish the requirements for IPD including QFD, marketing, strategic/tactical business planning (Tsuda, 1995). In the IPD-set system, PDT-set integrates the customer's inputs, their product and processes with their own experience in business solutions (tools), knowledge of future directions (technology), and the organiza-

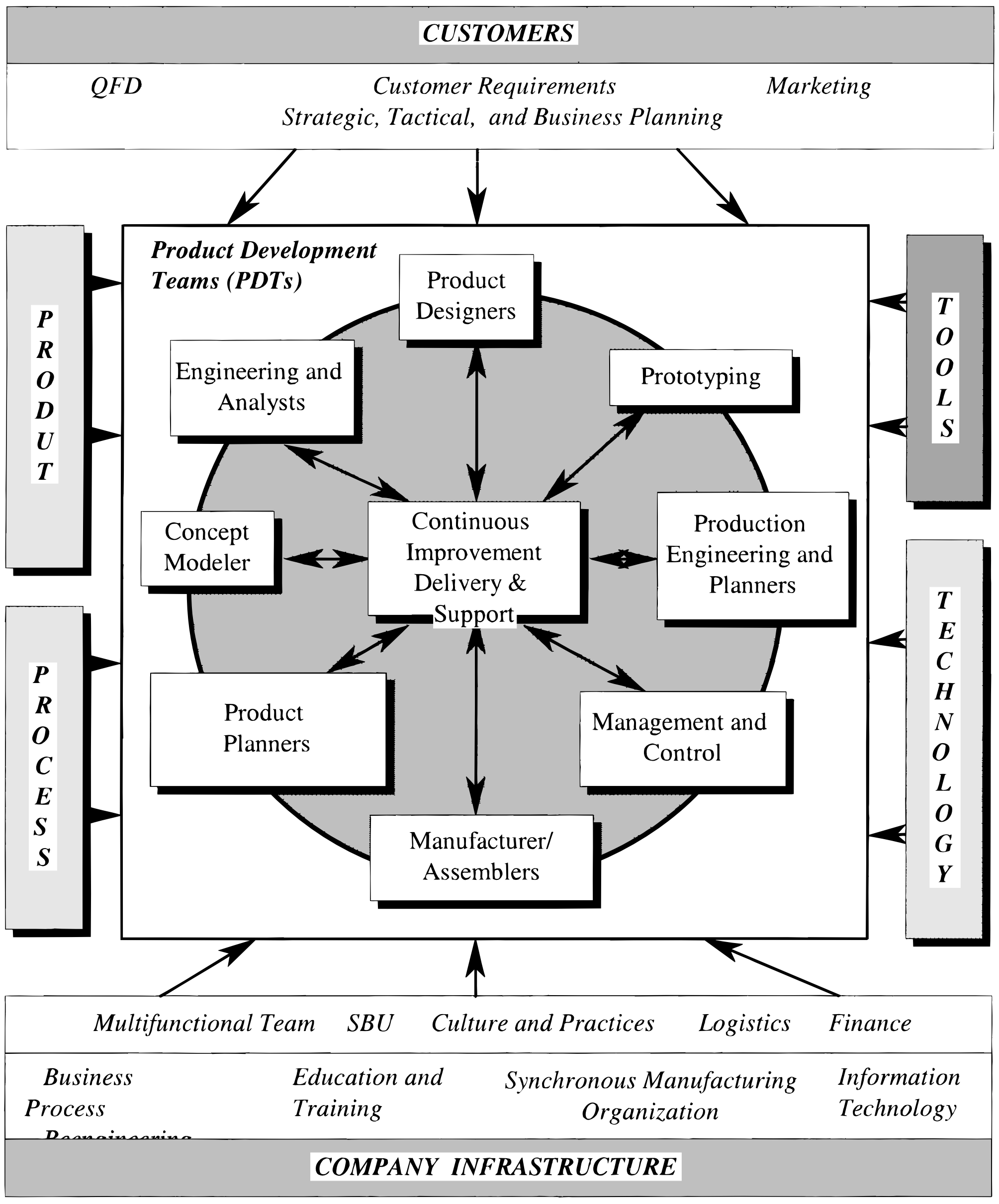

Fig. 7. A typical product development team (PDT)—An example.

tional infrastructure to provide worldwide competitive advantage. The PDT-set, which replaces the traditional functional department, are often organized along goal-oriented principles. Experts in the field of mechanical, electrical, industrial, chemical and material engineering, as well as a variety of other fields, work together. Removals of barriers to cooperation and resolution of conflicts are responsibilities of the PDT manager (Deming, 1993).

The demands of today's ever-changing international marketplace are immense. Goals are moving targets, undergoing constant changes and shifting in response to market conditions. The diversity of disciplines in CE is essential to leverage core competency to address the growing complexity of today's product needs and global manufacturing trends. CE requires a new approach to project management. Each team must work closely with other teams to identify and develop techniques that are more cost-effective, innovative, and simple to use.

4.2. Principle 2: Parallel product decomposition

Smith and Browne (1993) describe decomposition as a fundamental approach to handling complexity in engineering design. Product decomposition means viewing the product realization process as a part of the whole and then aggregating (summing) the decomposed A-sets to recreate or reconstruct the whole set (IPD-set) from its parts (A-sets). In other words,

$$\begin{aligned}&\text{Product Realization}\\ &\quad\Leftrightarrow [\{\textit{Decomposing parts-from-the-whole}\}\\ &\quad\oplus \{\text{Reconstructing the whole-from-the-parts}\}]. \qquad (21)\end{aligned}$$

The symbol $\Leftrightarrow$ signifies that product realization is logically "*composed or made out of*" two essential sets contained in the square brackets and shown in Eq. (21). The term "whole" also includes multiple characteristics of life-cycle concerns (e.g., X-ability). Although not all life-cycle activities are independent, many sets can be decomposed safely. For example, it is not necessary to delay the start of an activity if the information required for that activity is not dependent on the rest. Due to an increased global pressure to bring a product into the marketplace early, parallel processing in CE is becoming a necessity. There are, however, many ways a product, process or work information can be decomposed and overlaid in parallel (Kusiak & Wang, 1993). If a product, process, or a work information activity does not affect other parameters or processes, it can be performed locally. If it does, it can be performed in a distributed fashion. Local or distributed processing, to a large extent, depends on how a product structure is originally broken up or decomposed (D'Ambrosio et al., 1996). Do the decomposed parts exhibit independent or semi-independent characteristics? Decomposition allows the scheduling of activities to proceed in parallel.

The two (decomposition + concurrency) allows one to identify activities that can be overlapped or performed simultaneously. It also allows one to formulate product realization strategies, for example, indexing, alternate decomposition, teaming, or restructuring leading to satisfaction of all intermediate life-cycle requirements and constraints during this realization process.

Previously, in Eq. (1), an activity set, A-set, was broken up into activities: a_i, a_j, a_k, etc. There are four possible ways such activities can be related to each other (Prasad, 1996). They are shown in Figure 8. This means that A-set in Eq. (1) can be split into four subgroups. The corresponding sets for these four subgroups are: (a) dependent activities set, $\{A_{dep}\}$; (b) semi-independent activities set, $\{A_{sin}\}$; (c) independent activities set, $\{A_{ind}\}$; and (d) inter-dependent activities set, $\{A_{int}\}$.

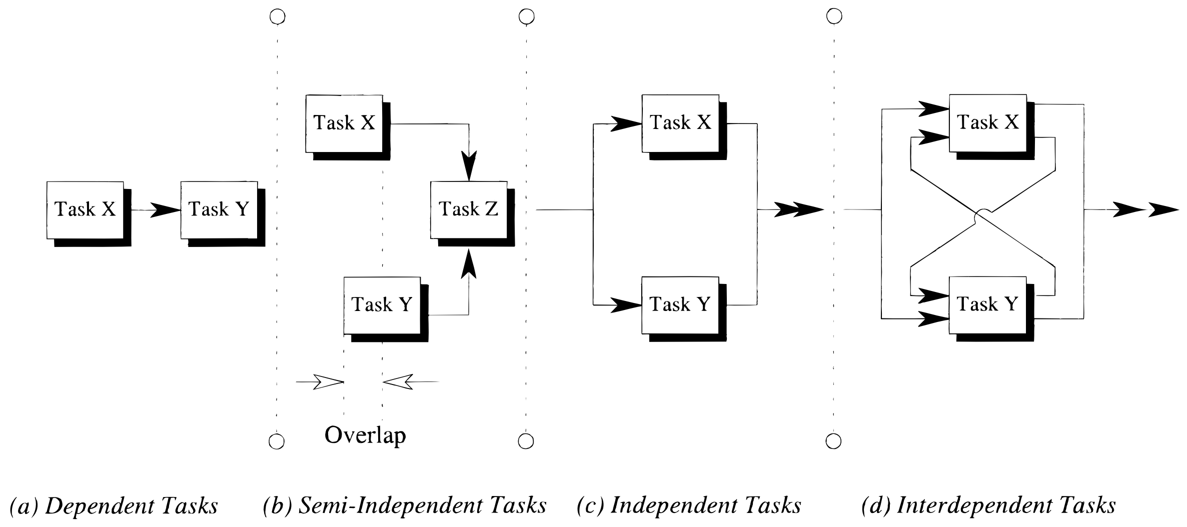

Fig. 8. Possible relationships between pair of activities.

- *Dependent Activities*: A pair of activities (say, a_i, a_j) is said to be dependent, if an activity requires information, which is an output from another activity. The information required could be a complete output transfer or it may represent only a portion of the output. If the transfer of information is complete, they are usually run in a series. This is shown in Figure 8(a).
- *Semi-independent Activities*: A pair of dependent activities (say, a_i, a_j) is said to be semi-independent if the transfer of output from one activity to the other is only a partial transfer (pseudo-parallel). The pseudo-parallel structure means that there exist weak interactions among groups of activities. In Figure 8(b), an activity, a_k is said to be dependent upon the activities, a_i and a_j because partial outputs from both a_i and a_j are used to complete activity a_k.
- *Independent Activities*: A pair of activities (say, a_i, a_j) is said to be independent if no portion of the output from one activity or the other is required for the completion of both activities. Figure 8(c) shows a pair of activities, a_i and a_j that is independent.
- *Interdependent Activities*: A pair of activities (say, a_i, a_j) is said to be interdependent, if a two-way information exchange is required for the completion of the job. Meaning, information from one activity (say, a_i) is used to complete the second activity (say, a_j) and the information from the second activity (a_j) is used to complete the first activity (a_i). This is shown in Figure 8(d).

In other words, if the activities in the A-set [shown in Eq. (1)] are reorganized along the above four subgroups, Eq. (1) can be expressed in the following way:

$$\text{A-set} \equiv [\{\mathrm{A_{dep}}\}, \{\mathrm{A_{sin}}\}, \{\mathrm{A_{ind}}\}, \{\mathrm{A_{int}}\}]. \quad (22)$$

The square bracket implies union-of the individual sets, which are contained within the curly brackets of Eq. (22).

Paralleling activities and the amounts of overlap depend upon the types of relationship and the degree of dependency that exist between them (Kusiak & Wang, 1993). The overlap between two intermediate activities or specifications/outputs represents the time elapse to build the information required for the start of the subsequent activities. Coordinating activities that exhibit dependent $\{\mathrm{A_{dep}}\}$ or independent characteristics $\{\mathrm{A_{ind}}\}$ are quite straightforward. The dependent activities, belonging to the set $\{\mathrm{A_{dep}}\}$, are arranged in series and independent activities, belonging to the set $\{\mathrm{A_{ind}}\}$, are stacked in parallel. For the work groups, however, the challenges of CE are extremely difficult when many activities are interdependent, those that belong to the set $\{\mathrm{A_{int}}\}$. Meaning they are coupled and cannot be separated explicitly either in a series or in a parallel mode. As discussed earlier, interdependent (or coupled) activities take more design time and many iterations (of information transfer back and forth) before they finally converge.

CE strives for simultaneity and immediacy. In practice, however, mutually independent group of activities seldom exist. Strategically, decomposing the interdependent activities, which belong to the set $\{\mathrm{A_{int}}\}$, into a series of dependent: $\{\mathrm{A_{dep}}\}$, semi-independent: $\{\mathrm{A_{sin}}\}$, and independent activities: $\{\mathrm{A_{ind}}\}$, can reduce the size of the working groups and the number of iterations required to obtain a reasonable solution.

4.3. Principle 3: Concurrent resource scheduling

Facilitating the transfer of work information among work groups is an essential organizational responsibility of any company. Concurrent resource scheduling involves scheduling the distributed activities, A-set, so that they can be performed in parallel. Paralleling is simple for activities exhibiting independent or semi-independent characteristics: $\{\mathrm{A_{ind}}\}$, $\{\mathrm{A_{sin}}\}$. However, it is not so simple for dependent activities set, $\{\mathrm{A_{dep}}\}$. There are many cases when activities are dependent (not yet coupled), but need to be scheduled in parallel with other activities. A simple case is that of an overlap. Even though an activity is dependent on another, there is no need for one to wait until the other task ends. If an activity precedes and generates the needed information for a later activity, the next task can start as soon as the needed information is made available. There is no need to wait for the completion of the former task. If the two activities are independent, they can be scheduled in any order necessary. The other options that address these issues more precisely are: optimal scheduling (minimizing time, resource, cost, etc.), backward scheduling (meeting target time), and team-based project management. Sanborn Manufacturing Company employed a backward scheduling to set up major milestones consisting of hard and fast dates and worked back from those dates as a planning mechanism (*Machine Design*, 1993).

If the set of activities in the A-set are independent, that is the activities belong solely to the $\{\mathrm{A_{ind}}\}$ set, a pair of activities (say a_i, a_j) can start immediately, meaning

$$t_{si} \leftrightarrow t_{sj}. \quad (23)$$

The symbol $\leftrightarrow$ means starting time is coincident with respect to initial timing. The term '*ts*' in Eq. (23) denotes a time of start for an activity '*a*', where A-set was defined in Eq. (1),

$$\text{A-set} \equiv \{a_1, a_2, a_3, a_4, \ldots, a_i, a_j, a_k, \ldots, \ldots a_n\}.$$

Frequently, the "product and process" are radically redesigned to achieve parallelism. Paralleling of activities provides management team with opportunities to reorganize and control the resources applied during CE. These resources fall into three main categories:

- teams (e.g., people, machines, facilities, outside firms);
- tasks (activities or projects they work on, knowledge of the projects, information they need to work with); and
- time.

The trio provides a basis for defining a work breakdown structure (WBS). A WBS is really a series of interrelated work tasks initially set in motion by the planning track. New tasks are added or created by the subsequent tracks when put into motion. The latest series of tasks are mostly due to support and delivery track. These tasks are over only when that product is finally disposed at the end of its useful life.

A good WBS contains all three elements: *paralleling of tasks, paralleling of teams (work groups), and optimal time schedules*. A good WBS uses tasks' decomposition knowledge that designers commonly use as much as possible. It contains knowledge that aggregates the existing evidence for concurrent work scheduling. Techniques such as optimal resource planning, cost accounting, level balancing, OPT, and other load management approaches are considered integral to WBS in achieving concurrent resource scheduling. The types of WBS required within an organization dictate how 7Ts should be developed and used. Figure 9 also shows how CE activities and work groups should be organized into loops, linked (digitally connected) together by a product breakdown structure (PtBS) and/or process breakdown structure (PsBS) hierarchy. The product decomposition details have been integrated into such loops. Concurrent resource scheduling is shown in Figure 9 as a central block, where arrows to and from the nested loops or decision blocks either emanate or terminate. The outer loop starts with the multiple perspectives of design and the innermost loop ends with multiple analyses (or projects). There is a series of nested loops to prune the elements or the information envelope required to build a total product or process model.

4.4. Principle 4: Concurrent processing

Managing time is the fulcrum of concurrent engineering. Some companies rely on milestones. Others use strategic routing and queuing as another way to manage time. Concurrent processing means optimal routing and queuing of activities both from the work-group distribution and information build-up standpoint. This is essential to guide the design of the product and its processes toward a quality end. Concurrent processing is never easy, particularly in industrial settings where solvable technical problems are prevailed upon by cultural considerations. Resistance to change is quite predominant. This is seen, for example, in the automotive industry, and more generally, in companies where the age profile of the technical staff is high (Womack et al., 1990; Prasad, 1996). The three most important concepts associated with concurrent processing are: creation of "variable-driven" product/process models, route management, and queue management.

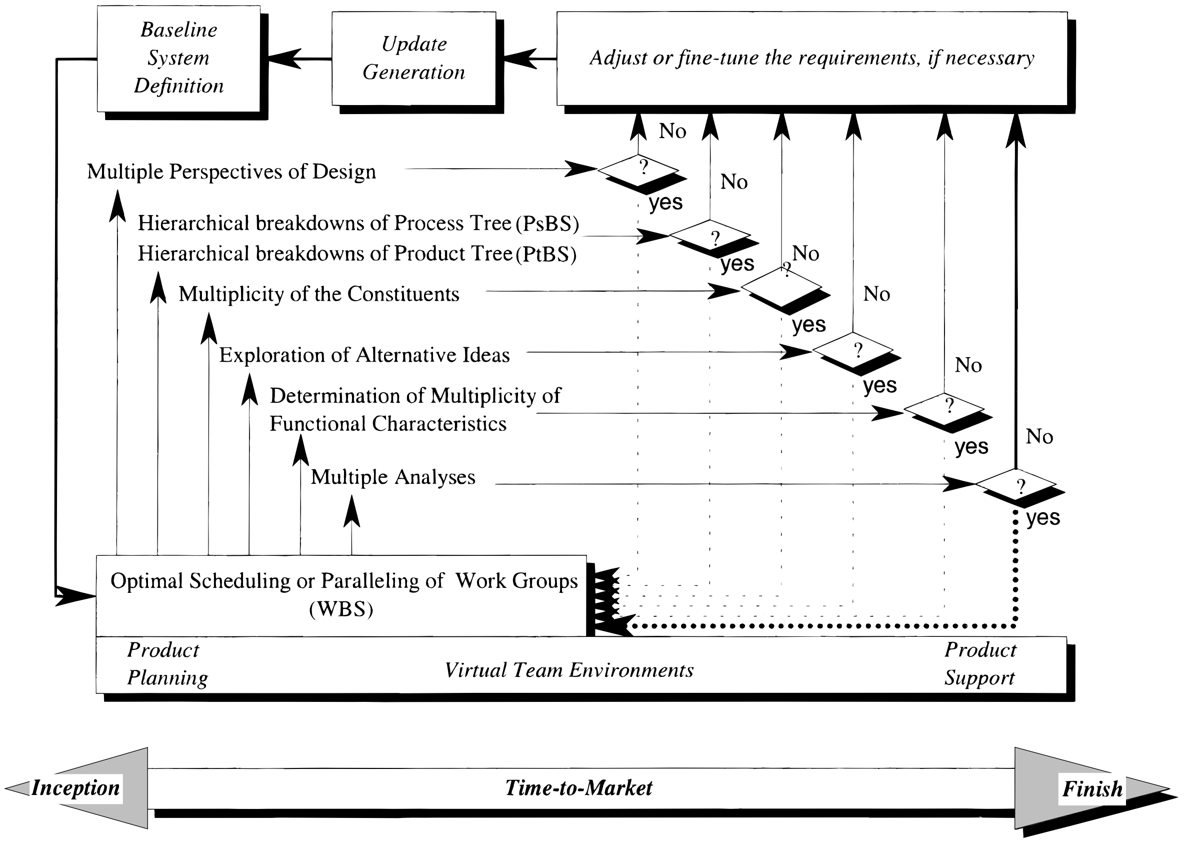
Baseline System Definition
Update Generation
Adjust or fine-tune the requirements, if necessary
Multiple Perspectives of Design
Hierarchical breakdowns of Process Tree (PsBS)
Hierarchical breakdowns of Product Tree (PtBS)
Multiplicity of the Constituents
Exploration of Alternative Ideas
Determination of Multiplicity of Functional Characteristics
Multiple Analyses
?
No
yes
Optimal Scheduling or Paralleling of Work Groups (WBS)
Product Planning
Virtual Team Environments
Product Support
Inception
Time-to-Market
Finish

Fig. 9. Schemata for exploiting concurrency.

In concurrent processing, activities are staggered (performed simultaneously or overlapped) rather than carried out sequentially. Keeping track of those complex dependencies that vary with time is a critical task in concurrent processing. Appropriate synchronization efforts between different CE teams have to be made.

If the set of activities in A-set is performed simultaneously (complete overlap), it implies a pair of activities, a_i and a_j can start together, that is

$$t_{si} \leftrightarrow t_{sj}. \tag{24}$$

If the set of activities in A-set is overlapped (partial), it implies that a pair of activities, a_i and a_j, start and end times are related as follows,

$$\begin{aligned} t_{si} &< t_{sj} \\ t_{ei} &> t_{sj}, \end{aligned} \tag{25}$$

where, the symbol $\leftrightarrow$ means that the beginning or the start timing of the two activities "coincides."

4.5. Principle 5: Minimize interfaces

This entails reducing (or minimizing) all sorts of interfaces during a "product realization process" (PRP) subject to a given set of constraints imposed on the IPD-set [defined earlier in Eq. (20)]. This can be expressed symbolically as:

$$\text{Number-of PRP}_{\text{interfaces}} \Rightarrow \text{Minimum} \tag{26}$$

Subject to "IPD-set" [expressed by Eq. (20)] converging to a feasible or compatible set.

The symbol, ⇒, means "drive to." The PRP should be redesigned with CE in such a way that the number of interfaces present in a new "redesigned PRP with this CE principle" is driven to a bare minimum. These interfaces include the relationships between management and design, supplier interface, design-development interface, design-manufacturing interface, production interface, etc. Such interfaces depend upon the size of the company, the product and process complexity, and decomposition. Shared product design can be facilitated by introducing adequate interface management. The main focus is on identifying various sources of interfaces and determining whether they are actually needed. The goal is to reduce the number of design and manufacturing interfaces to a minimum. "Reducing interfaces" means taking steps to redesign and simplify business systems and processes, search out best practices (3Ps), to develop a more competitive work force, and to explore new business methods. This principle fosters out-of-comfort-zone thinking, relies on value-added benefits to both the customer and the business, and focuses heavily on 7Ts (talents, tasks, teams, techniques, technology, time, and tools). It requires follow through until the new process is firmly entrenched. Unlike organizational restructuring, "minimize interfaces" principle involves alterations in the level of abstraction to reconfigure the subject PRP system. It may involve reconstituting this PRP into a new form or to a new level of abstract descriptions, and a new implementation of an altered form of PRP. This saves time, reduces design costs, and gets the needed partners involved early in the process. "Minimize interfaces" involves at a minimum reducing or eliminating three types of interfaces that are commonly found in a PD^3 process: product interfaces, process interfaces, and computer interfaces (Prasad, 1996).

4.5.1. *Minimize product interfaces*

The product design problem is often decomposed into sub-domains, each having its own design variables and constraints.

$$\begin{aligned} \text{Product-set} \equiv [&\{\text{System}\}, \{\text{Subsystem}\}, \{\text{Components}\}, \\ &\{\text{Parts}\}, \{\text{Features}\}]. \end{aligned} \tag{27}$$

Product-set is defined as a union of the sets contained within the square brackets. Equation (27) is similar to what was shown for an automobile example in Section 3, Eq. (16). Here {Features} represents a combination of {materials} and {Characteristics}. These subdomains can be quite independent of each other except in a limited number of common interfaces. The PtBS tree thus drives the product design to an interface-driven integration technique.

The PtBS also serves as the model index structure and helps keep the digital equivalent organized and easier to cross-reference with other indexes. The problems of each subdomain can be solved in parallel and the results brought back to satisfy global needs at a later time. Such a decomposition of design, as represented by the structured PtBS tree, can be achieved in a number of ways. For example, the design problem can be divided into a four-step process as shown in Figure 10. The first step is to develop a functional system. It yields system characteristics, which is input to the next step to identify and develop subsystems. The subsystems characteristics are then input to the third step to identify and develop components. Finally, the components' characteristics are then fed into the fourth step to identify and develop parts (see Fig. 10). There are four decision blocks, corresponding to four loops: conceptual design, layout design, subassembly design, and an assembly design, which checks whether the corresponding design is satisfactory or not. The other aspect of the PtBS tree is the minimization of interfaces among these five steps: system, subsystems, components, parts and features. This was illustrated in Section 3 by an automobile example.

$$\begin{aligned} \text{Product-interface-set} \equiv\ &[\{\text{System}\} \cap \{\text{Subsystems}\}] \\ &\cup [\{\text{Subsystems}\} \cap \{\text{Components}\}] \\ &\cup [\{\text{Components}\} \cap \{\text{Parts}\}] \\ &\cup [\{\text{Parts}\} \cap \{\text{Features}\}]. \end{aligned} \tag{28}$$

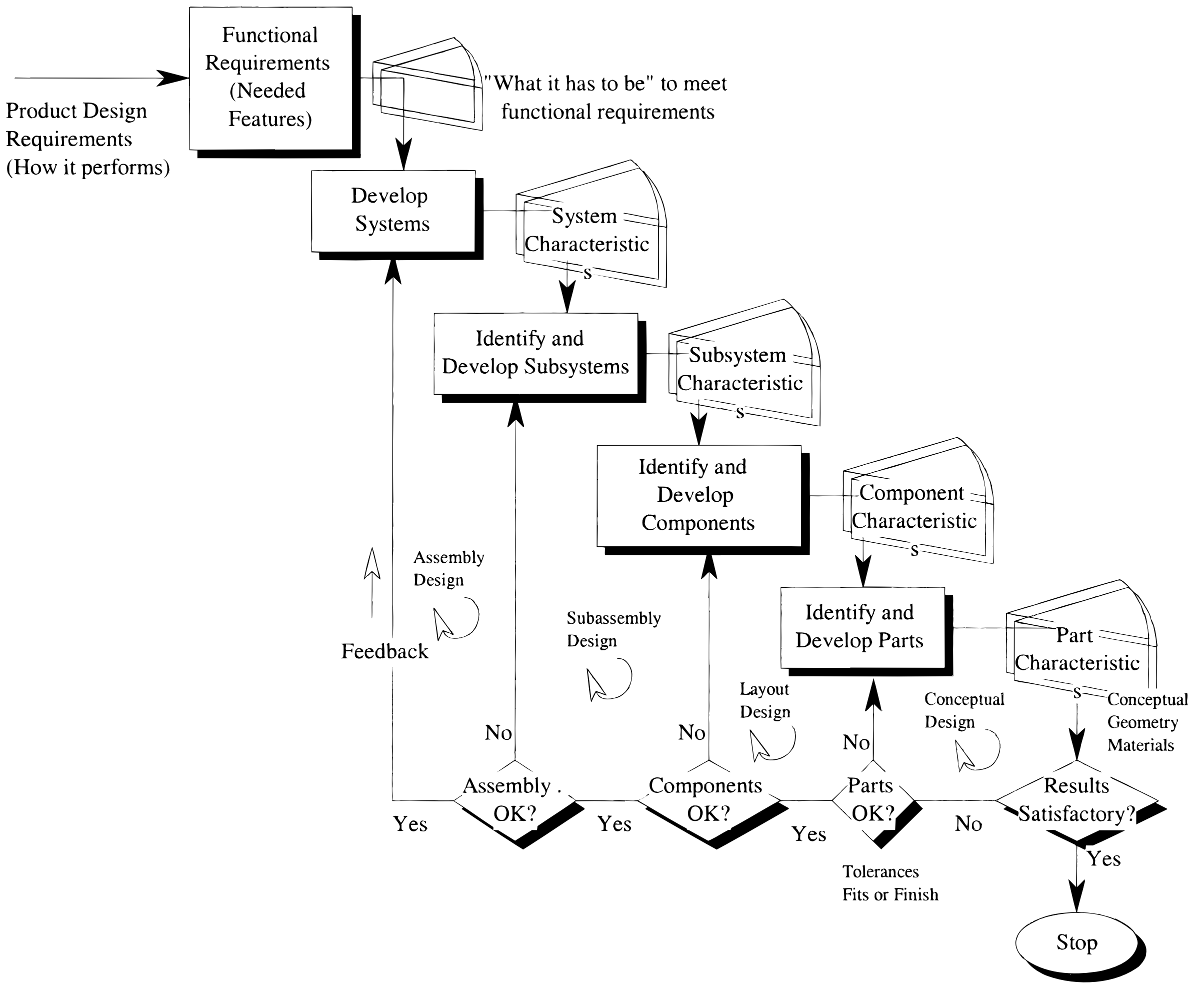
Functional Requirements (Needed Features)
Product Design Requirements (How it performs)
"What it has to be" to meet functional requirements
Develop Systems
System Characteristics
Identify and Develop Subsystems
Subsystem Characteristics
Identify and Develop Components
Component Characteristics
Identify and Develop Parts
Part Characteristics
Conceptual Geometry Materials
Assembly Design
Feedback
Subassembly Design
Layout Design
Conceptual Design
No
No
No
Assembly OK?
Components OK?
Parts OK?
Results Satisfactory?
Yes
Yes
Yes
No
Yes
Tolerances Fits or Finish
Stop

Fig. 10. Minimize product interfaces.

Product-interface-set is defined as a union of the sets contained within the square brackets. Where ∪ means union of the sets in the square brackets. The symbol ∩ means intersection of the two sets identified within the curly brackets. If a decomposed element is decoupled (or loosely connected), the tasks of interface definition are simple and straightforward. Tolerances, finish, and fit requirements have little or no impact on conceptual design, assembly, or components' functions. Material types, as represented by the structured bill-of-materials, can often be modified without jeopardizing the part, component, or assembly function. The convenience of processing design problems in parallel can lead to a converged design much faster than conventionally possible.

However, if decomposed elements of PtBS tree overlap (cross areas' boundaries), the interface definitions could be quite complex and intertwined. The major product development challenge in such cases is to integrate the many (decomposed) subproblem solutions into a well-connected system. Some organizations address this by assigning teams of analysts or conflict resolution engineers to handle the interactions between the decomposed subproblems. The trouble is that such interactions are rarely known in advance or their implications are not well understood. Interface management is the technique used to minimize interfaces. Management implies preparing the PtBS tree or its content so as to preclude possible interfaces between the decoupled elements. Through this approach, the design at the complete product level supports the next level of design, which supports the next level and so on. The decomposition is consistent with their interface requirements. This does not prematurely commit the product to a high cost.

4.5.2. Minimize process interfaces

Like in product design, the process design problem can also be decomposed into subdomains. These subdomains can be quite independent of each other except a limited number of interfaces. Similar to the product design case, problems of each process plan domain can be solved in parallel and results brought back to satisfy global needs at a later time. Such a decomposition of a process plan—a process breakdown structure (PsBS) is shown in Figure 11. Here the pro-

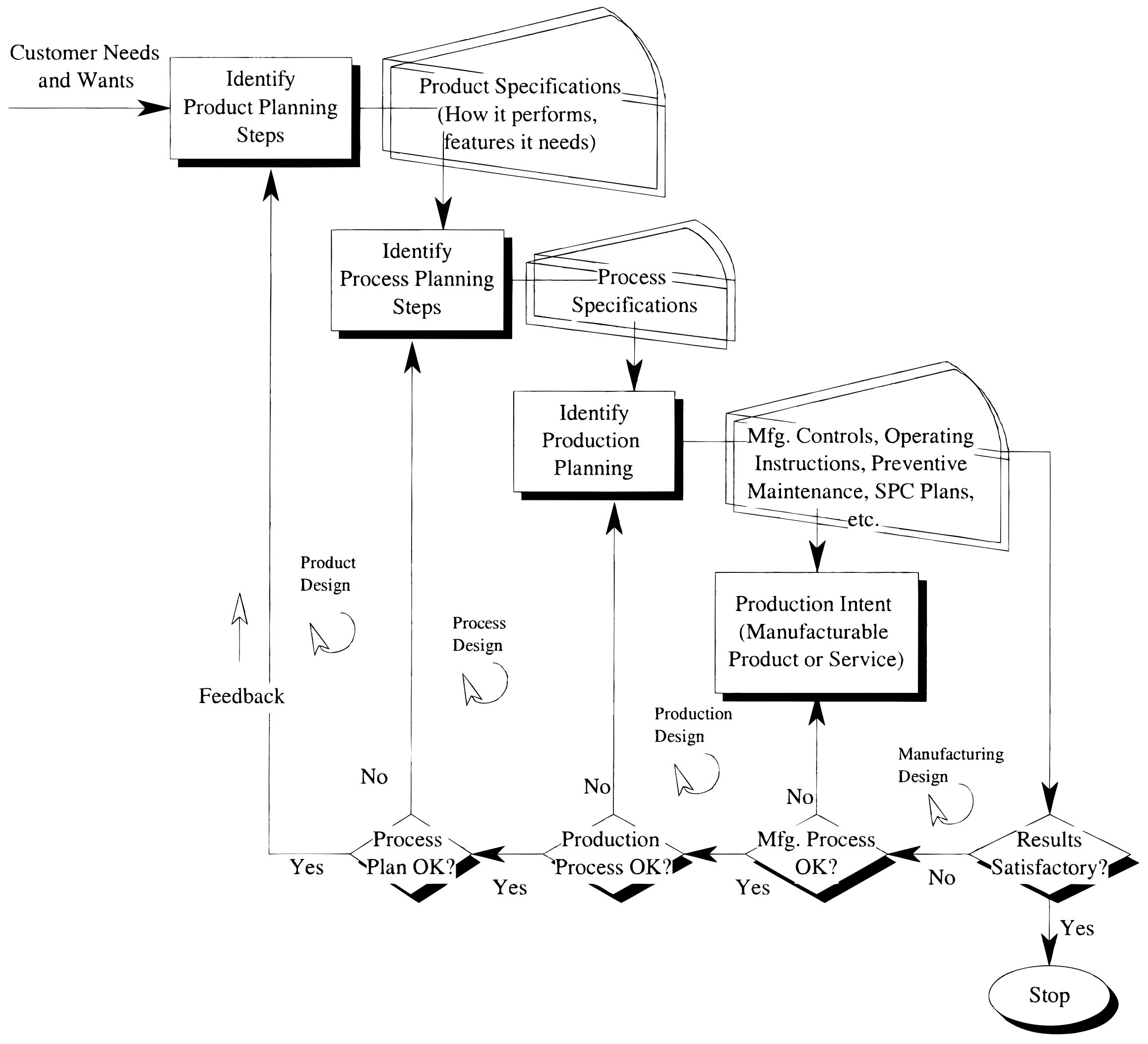
Customer Needs and Wants
Identify Product Planning Steps
Product Specifications (How it performs, features it needs)
Identify Process Planning Steps
Process Specifications
Identify Production Planning
Mfg. Controls, Operating Instructions, Preventive Maintenance, SPC Plans, etc.
Production Intent (Manufacturable Product or Service)
Product Design
Process Design
Production Design
Manufacturing Design
Feedback
No
No
No
Process Plan OK?
Production Process OK?
Mfg. Process OK?
Results Satisfactory?
Yes
Yes
Yes
No
Yes
Stop

Fig. 11. Minimize process interfaces.

cess plan is divided into four stages: product planning, process planning, production planning, and production intent (manufacturable product or service).

$$\begin{aligned}\text{Process-set} &\equiv [\{\text{product-planning}\}, \{\text{process-planning}\},\\ &\quad \{\text{production-planning}\}, \{\text{production-intent}\}], \text{ or}\\ \text{Process-set} &\equiv \{\text{product-planning}\} \cup \{\text{process-planning}\}\\ &\quad \cup \{\text{production-planning}\} \cup \{\text{production-intent}\}.\end{aligned} \tag{29}$$

Process-set is defined as a union of the sets contained within the square brackets in Eq. (29). The first stage is to identify a functional set of production planning steps. It yields product specifications, which are input for the next stage to identify process planning steps. The resulting process specifications are then input to the third stage to identify production planning steps. Finally, its outputs are then fed into the fourth stage to obtain a production intent (see Fig. 11).

There are four decision blocks, corresponding to such four loops: manufacturing design, production design, process design, and product design.

$$\begin{aligned}\text{Process-interface-set} \equiv [&\{\text{Product-design}\} \cup \{\text{Process-design}\}\\ &\cup \{\text{Production-design}\}\\ &\cup \{\text{Manufacturing-design}\}]\end{aligned} \tag{30}$$

where,

$$\text{Product-design} \equiv [\{\text{Product-planning}\} \cap \{\text{Process-planning}\}] \tag{31}$$

$$\text{Process-design} \equiv [\{\text{Process-planning}\} \cap \{\text{Production-planning}\}] \tag{32}$$

$$\text{Production-design} \equiv [\{\text{Production-planning}\} \cap \{\text{Production-intent}\}] \tag{33}$$

$$\text{Manufacturing-design} \equiv [\{\text{Production-intent}\} \cap \{\text{Desirable-specifications}\}]. \tag{34}$$

The process-interface-set is defined as the union of the four curly-bracket sets contained within the square brackets. The decision blocks check whether the corresponding plan is satisfactory or not. In essence, PsBS is the process by which a company designs and manufactures its products, while PtBS is the means used to describe or capture the inherent complexity of a product.

4.5.3. Minimize computer interfaces

Too many computer interfaces can create problems with the smooth flow of information. Each program has its own data, input, and output format requirements. For these programs to run seamlessly, the inputs and outputs of these programs must work in concert with each other. Manual data entry is error prone. Moreover, there should be a single data source from where all inputs originate, so that if a parameter is changed, the correct value is passed on to all interface-programs using them.

4.6. Principle 6: Transparent communication

This provides virtual communication between the individual activities that are partitioned (decomposed), and among the concurrent team members (discussed in Principle 1). Transparent communication involves identification and definition of mission-critical data. All members of the CE teams need to have the same common understanding of the frequently used terms and their meanings. It may require definition of "data dictionary and semantics" as a structured approach to resolving conflicts and for consensus building. The elements that contribute to transparent communications are (a) global access (b) Universal Product Code (c) Electronic Data Interchange (EDI) (d) Technical memory (Prasad, 1997).

4.7. Principle 7: Quick processing

Quick processing means performing individual activities, a_i, or device means to perform activities, a_i, [see Eq. (1)] in A-set as fast as possible using productivity tools or design aids. It also amounts to speeding up the preparation time in building up the information content before and after an execution of an activity. This emphasizes the mandate for shortening the pre- and postprocessing time and the time it takes for completing the decomposed activities themselves.

$$\text{Quick Processing} \Leftrightarrow \text{Minimize } (d_i) \quad \text{for} \quad \forall\ i;\ i = 1, n. \qquad (35)$$

Equation (35) symbolically states that "*quick processing*" is logically equivalent to minimizing all elements of d_i. Where, n is the number of activities in the A-set [see Eq. (1)]. The term d_i is the time duration—defined in Eq. (3). There is a difference between the complexity of the philosophies (such as product complexity, process complexity, enterprise complexity, or complexity of cognitive behavior), and the philosophies of their management. An organization committed to making complex products in the shortest possible time need not require an equally complex management philosophy. Organizations can still handle all that while following a simple management philosophy. This simple management philosophy is the philosophy of decomposition followed by concurrent processing. This is similar to what used to be at one time the European philosophy of "divide and concur." To apply this to a complex product, a systematic decomposition of the product and process, including 7Ts [see Fig. 4.1 in Prasad (1996)].

Fast processing can be accomplished through high-bandwidth and backbone technology or building flexibility into the process. Management techniques, which are the product of decades of corporate learning, can be captured as knowledge or rules. With high bandwidth and backbone technology, such as object-oriented databases, technical memory, parallel computers, multimedia, X-window, a large amount of information exchange can take place at a very high speed. Using such means, product development rules can be coded into knowledge-based design and manufacturing software programs (Prasad, 1997). Once these knowledge-based programs and technical memories are deployed as useful life-cycle aids, they can provide considerable competitive advantage to companies in terms of design speed, accuracy, and quality (Prasad, 1997). The competitive advantage earned through process management techniques is not only retained in this case but the methods are also readily available for future use, when market conditions suddenly change or a competitor develops a superior product. The flexibility inherent in knowledge-based systems can be exploited to overcome any such short-term market fluctuations.

5. CONCLUDING REMARKS

At the heart of any good product design, development, and delivery (PD^3) process, there lays a set of underlying principles for satisfying the interests of the customers and the company. The paper described a set of *seven principles of concurrency and simultaneity* (namely parallel work group, parallel product decomposition, concurrent resource scheduling, parallel processing, minimize interfaces, transparent communication, and quick processing). The company focus shows up in applying these seven principles initially to identify concurrent teams and then to organize the activities that can be overlapped or performed simultaneously.

The set of these principles provides companies significant competitive advantages and organizational abilities to manufacture a quality product in less time and cost that a customer would like to buy. CE Principles also help the teams to formulate significant product and process strategies leading to their separation; for example, indexing, alternate decomposition, teaming, or restructuring.

REFERENCES

Carter, D.E., & Baker, B.S. (1992). *Concurrent engineering: The product development environment for the 1990s*. Addison-Wesley Publishing Company, Reading, Massachusetts.

Chelsom, J.V. (1994). Concurrent engineering case studies: Lessons from Ford Motor Company experience. In *Concurrent Engineering: Concepts, Implementation and Practice*, (C.S. Syan and U. Menon, Eds.) pp. 24–48. Chapman & Hall.

D'Ambrosio, J., Darr, T., & Birmingham, W. (1996). Hierarchical concurrent engineering in a multiagent framework. *Concurr. Eng. Res. Applications Int. J. 4(1)*, 47–58.

DARPA. (1987). *Workshop on Concurrent Design* December 1–3, Key West, FL. Defence Advanced Research Projects Agency (DARPA), Washington, DC.

DARPA. (1988). *Workshop on Concurrent Design* December 2–4, Key West, FL. Defence Advanced Research Projects Agency (DARPA), Washington, DC.

Deming, W.E. (1993). *The new economics*. MIT Center for Advanced Engineering Study, Cambridge, Massachusetts.

Krishnan, V. (1993). *Design process improvement: Sequencing and overlapping activities in product development*. DSc Thesis. Massachusetts Institute of Technology, Cambridge, MA.

Kusiak A., & Wang, J. (1993). Decomposition of the design process. *J. of Mechanical Design, Trans. ASME 115(4)*, 687–695.

(1993). How to make concurrent engineering work, Part 6—Measuring success from CE. *Machine Design 65(23)*, 77–80.

Parsaei, H.R., & Sullivan, W.G., Eds. (1993). *Concurrent engineering: Contemporary issues and modern design tools*. Chapman & Hall, London.

Pennell, J., & Slusarczuk, M. (1989). *An annotated reading list for concurrent engineering*. Report No. HQ 89-034130. Institute for Defense Analyses, Alexandria, VA.

Prasad, B. (1996). *Concurrent engineering fundamentals, Volume I: Integrated product and process organization*. Prentice Hall, New Jersey.

Prasad, B. (1997). *Concurrent engineering fundamentals, Volume II: Integrated product development*. Prentice Hall, New Jersey.

Smith, G.F., & Browne, G.J. (1993). Conceptual foundations of design problem solving. *IEEE Trans. Sys. Man Cybern. 23(5)*.

Tsuda, Y. (1995). QFD models for concurrent engineering development processes of automobiles. *Concurr. Eng. Res. Applications—An Int. J. 3(3)*, 213–220.

Turino, J.L. (1992). *Managing concurrent engineering: Buying time to a market: A definitive guide to improved competitiveness in electronics design and manufacturing*. Van Nostrand Reinhold, New York.

Womack, J.P., Jones, D.T, & Roos, D. (1990). *The machine that changed the world*. Macmillan Publishing Co., New York.

Zhang, H.C., & Zhang, D. (1995). Concurrent engineering: An overview from manufacturing engineering perspectives. *Concurr. Eng. Res. Applications Int. J., 3(3)*, 221–236. .

Biren Prasad is the Director of Knowledge-based Engineering (KBE), Product business Unit at Unigraphics Solutions Inc. Before Joining Unigraphics, Dr. Prasad was *Senior Engineering Consultant at Electronic Data Systems* (*EDS*) at General Motors account, where he was in charge of Automated Concurrent Engineering (ACE) Consulting Group. He has written or co-authored over 100 technical publications, including 80 archival papers and twelve books. His most recent original contribution is a two-volume textbook on "Concurrent Engineering Fundamentals" with Prentice Hall, USA. This has become very popular for teaching CE best practices both in U.S. universities and abroad. His most referred edited text is *Modern Manufacturing: Information Management and Control (1994)*, published by Springer Verlag. In 1989, he edited a set of three-volume book entitled *CAD/CAM, Robotics and Factories of the Future, 1989*, Springer-Verlag. He has served as editors for several additional texts, monographs and proceedings.

Dr. Prasad earned his Ph.D. from Illinois Institute of Technology, Chicago, a *Degree of Engineer* from Stanford University, California. He received a Master (M.S.) degree from Indian Institute of Technology, Kanpur and a B.S.M.E. Degree from Bihar College of Engineering, Patna, both from India. While completing studies for his D'Engineer, he worked at *Xerox Research Center*, and *Failure Analysis Associates*, both at Palo Alto, California. He is an Adjunct Professor of Mechanical & Aerospace Engineering at West Virginia University. He also teaches at Wayne State University, Oakland University and Lawrence Tech University in Michigan. Dr. Prasad has served on numerous committees for ASCE, ASME and SAE. He is the Chairman of the SAE's Computer Readers' Committee. His professional honors include: Associate Fellow of AIAA, Fellow of ASME, ASCE, SAE, AAAI, and fellow and life member of ISPE. He is listed in International Men of Achievement, International Biographical Center (IBC), England (1986), Dictionary of International Biography, IBC (1986), Who's who in Frontier Science & Technology (1985), Who's who in Technology Today (1985), Who's who in Aviation & Aerospace (1983), International Directory of Engineering Analysts (IDEA) (1983), Personalities of America (1986).