DD701890 *CI-02705798-3*

DD701890 **CISTI ICIST** CI-02705798-3

Document Delivery Service in partnership with the **Canadian Agriculture Library**

Service de fourniture de Documents en collaboration avec **la Bibliothèque canadienne de l'agriculture**

THIS IS NOT AN INVOICE / CECI N'EST PAS UNE FACTURE

SHADY PEYVAN
MILLIKAN MEMORIAL LIBRARY
CALIFORNIA INST OF TECHNOLOGY
1200 CALIFORNIA BLVD
PASADENA, CA 91125
UNITED STATES

ORDER NUMBER:	**CI-02705798-3**
Account Number:	DD701890
Delivery Mode:	ARI
Delivery Address:	131.215.226.3
Submitted:	2002/04/11 09:31:29
Received:	2002/04/11 09:31:29
Printed:	2002/04/11 22:21:18

Direct	**Periodical**	**OCLC**	**UNITED STATES**
$75.00			
$ 75.00			

Client Number:	6370119 /PRASAD, BRIAN
Title:	**JOURNAL OF SYSTEMS INTEGRATION.**
Vol./Issue:	VOL 9 NO 2
Date:	1999
Pages:	115–140
Article Title:	BIREN PRASAD: SYSTEM INTEGRATION TECHNIQUES OF SHARING AND COLLABORATION AMONG WORK-GROUPS, COMPUTERS AND PROCESSES
Report Number:	OCLC 25443749
Publisher:	BOSTON/U.S.A. : KLUWER ACADEMIC PUBLISHERS,
Edition:	*** PLEASE ARIEL OR SAY NO. THANKS.***
Information Source:	<TN LENDER: UPM,*CAI

INSTRUCTIONS: PLEASE ARIEL 131.215.226.3

Estimated cost for this 25 page document: $9.5 document supply fee + $21.5 copyright = $31

Phone/Téléphone: 1-800-668-1222 (Canada - U.S.) (613) 993-9251 (International)
Fax/Télécopieur: (613) 993-7619 www.nrc.ca/cisti producthelp.CISTI@nrc.ca

National Research Council Canada Conseil national de recherches Canada

Journal of Systems Integration, 9, 115–139 (1999)

System Integration Techniques of Sharing and Collaboration among Work-groups, Computers and Processes

BIREN PRASAD prasadb@ugsolutions.com
Unigraphics Solutions, Knowledge-based Engineering (KBE) PBU, P.O. Box 250254, West Bloomfield, MI 48325-0254, USA

Received December 10, 1997; Revised December 2, 1998

Abstract. Systems Engineering (SE) and Concurrent Engineering (CE) implies sharing of information, data, process and knowledge [20] across different levels in an enterprise. Division of work into classes and then to the work-groups or to the concurrent sub-teams is one form of sharing large organizations like Ford Motor and General Motors have used [29]. Other forms of sharing are governed by the state of computer communications, workstation and database technologies [9]. For accomplishing the needed collaboration (and to facilitate concurrent engineering), several concepts and models for work-group computing were tried at Delphi Divisional units of Electronic Data Systems (EDS) and General Motors. Based on such experiences, the paper describes Systems Engineering and CE techniques for sharing information in a concurrent engineering organization, which has been found most effective for collaborating knowledge among a set of multidisciplinary work-groups, array of computers and processes.

Keywords: knowledge sharing and collaboration, concurrent engineering, work-group computing, work breakdown structure

1. Introduction

In the information age that we live in, civilization depends on information and the dominant weapon for success is timely information sharing. Today, electronically, it is easy to connect to anyone in any part of the world at any time. We are bombarded with all sorts of information. However, we have very little knowledge of how to use them effectively. As Deming said [6], ***information is not knowledge***. Knowledge comes from theory. Without theory there is no rational basis to apply the information. The types of knowledge that are useful in meeting product realization goals are:

- ***System-knowledge***: Knowledge of how the system works in terms of concepts, facts, principles, methodology, or technology.
- ***Interface-knowledge***: Knowledge of how the underlying components and their interfaces work in terms of concepts, facts, principles, methodology, or technology.
- ***Transformation-knowledge***: This includes the knowledge of how to transform a set of specifications into a physical artifact or service.
- ***Behavior-knowledge***: This relates to "what-if" behavior of a model or a concept (assuming the model or concept has captured the desired knowledge) as some of the model attributes change.

- ***Environment-knowledge***: This includes the attitudes, values, visions, and beliefs of the participating teams, including the virtual teams, logical teams and technology teams, and those of the organizations.

In other words,

$$\textbf{Knowledge} = \cup\,[\textit{System-knowledge, Interface-knowledge, Transformation-knowledge, Behavior-knowledge, Environment-knowledge}]. \quad (1)$$

Where, $\bigcup$ means Union-of and the terms in the square brackets represent the possible sets. In this age of competitive advantage, what differentiates one Concurrent Engineering work-group from the rest is how one goes about performing *"knowledge sharing"* among teams, machines (or computers), and processes. Effective communication is the key to developing a knowledgeable and committed work force and setting a common set of consistent goals. Clear and supporting goals provide "constancy-of-purpose." They allow everyone in a company to set aside frivolous issues and focus on what is really important to the "total system."

Communication is a two-way street. Effective communication takes place both vertically (in spite of differences in responsibility or ranks) and horizontally (in spite of teams' functional differences) in an organization. An ineffective communication environment (that is, giving partial information and holding the rest of it) discourages free exchange of ideas up, down, and across organizational lines. The author working on system integration projects at Electronic Data Systems accounts with General Motors had encountered a number of such communication environments. Due to ineffective communication among the product development teams (PDTs), there is a danger that deficiencies discovered in the downstream activities of a product development process may not be rightly communicated to the upstream activities. This inhibits innovation, retracts teamwork, and strangulates opportunities for continuous improvement.

Many such real life examples of inhibitions and setbacks, this author had encountered working with product development groups during his fifteen years of combined tenure at Ford Motor Company and General Motors. Based on such experiences, the author describes in the paper a set of generalized forms of knowledge sharing and collaborations that can happen among typical teams, machines and processes in such large organizations.

2. Forms of Knowledge Sharing and Collaboration

Figure 1 shows nine forms of knowledge sharing and collaboration. These nine forms involve three-by-three combinations of concurrent teams (or employees), machines (or computers), and processes.

- ***Teams-to-teams sharing*** involves inter-team communication (across business units) and intra-team communication (among personal, technological, logical, and virtual teams). For instance, there is a need to establish an effective method for communicating knowledge and information among all personal teams (such as operators, team leaders, and application support personnel) [5]. The larger the organization, the more important it is to establish a formal method of sharing within (intra-team) and among (inter-team) disciplines. A proven old method is to hold group meetings at frequent intervals that include all the personnel team members. This can be time-consuming and may get out of hand, if not effectively managed.

- ***Teams-to-computer sharing*** involves human-machine interactions, such as user interface, information retrieval, data inputs, natural language, visual-based programming [13].
- ***Teams-to-process sharing*** entails interactive modeling, capture of geometry, such as CAD/CAM, or analysis (for example FEA/FEM/CAE) techniques including conferencing [20].
- ***Computer-to-teams sharing*** involves display, on-line help, desktop applications, computer graphics, output plots [22].
- ***Process-to-teams sharing*** includes communication capabilities among virtual teams, such as computational agents, system administration, network administration, groupware, blackboard architectures, advisors of all types, expert systems, knowledge-based engineering, optimization) [21]. It may be hard to visualize something as abstract as a process actually sharing something with a human team. In most cases a human team creates, modifies or makes use of a process. With the advent of knowledge-based engineering (KBE), intelligent data management, and KBE tools, processes are becoming more and more intelligent. A process for money transacting using an ATM (automated teller machine) card is an example of a two-way meaningful sharing. Ability of the personnel team members to access virtually and immediately all in-process or released data, as required, is an example of process-to-team communication [18].
- ***Computer-to-computer sharing*** involves inter-computer communications or inter-machine interactions such as LAN, penetration of E-mail, Internet, networks of all types, messaging [24].
- ***Process-to-computer sharing*** involves trouble shooting, remote diagnostic, fault management, feedback, process control, date, time, warning flags [16].
- ***Process-to-process sharing*** entails inter-process communications, such as groupware, remote applications, X-window applications, multi-media, Lotus Notes and work-flow based systems, process automation [19].
- ***Computer-to-process sharing*** means machine communications such as Distributed Numerical Control (DNC)/post-processing, multiple processes, and concurrent sessions [25].

It may be hard to visualize something as abstract as process actually sharing something with a human team. In most cases, a human team originally creates, modifies or make use of a process, and only after a series of rules' capture and validations, if some intelligence are built-into a process, then only a human team feels confident to share information. Sharing with and among computers seems more rational, because human teams often instruct computers to perform such activities. The effectiveness of collaboration is discussed next.

Modes	Team	Machine/ Computer	Process
Team	Team-to-Team Sharing (e.g:, Help Desk, Customer Support, Intra-teams, Work-group Applications, and Inter-team Cooperations)	Team-to-Computer Sharing (e.g., User Interface, Information Retrieval, Data Inputs, etc.)	Team-to-Process Sharing (e.g., Interactive Modeling, Analysis, CAD/CAM, CAE, CAD Conferencing, etc.)
Machine/ Computer	Computer-to-Team Sharing (e.g., Display, On-line Help, Desktop Applications, Computer Graphics, Plots, etc.)	Computer-to-Computer Sharing (e.g., Inter-computer, LAN, Internet, e-mail/Messaging, Network, etc.)	Computer-to-Process Sharing (e.g., Concurrent Multiple Processes, DNC/Postprocessing, Concurrent Sessions)
Process	Process-to-Team Sharing (e.g., Computational Agents, System Adm., Network Adm., Groupware, Virtual Teams, Blackboard, Expert Systems, Advisors, etc.)	Process-to-Computer Sharing (e.g., Feedback, Trouble Shooting, Remote Diagnostics, Fault Mgt., Process Control, Date, Time, Warning Flags, Storage, Retrieval, File Management, etc.)	Process-to-Process Sharing Inter-process (e.g., Groupware, Remote Applications, X-window Applications, Multimedia, Automation, etc.)

Figure 1. Forms of sharing & collaboration.

3. Effectiveness of Collaboration

One of the premises on which CE is based is the "paralleling" of activities so that multiple groups of people (work-groups) can work in teams. A work-group is a collection of sub-teams or people having common interests or expertise [30]. Work-groups usually have a short life in a CE organization. Such work-groups or sub-teams must collaborate with each other, in a timely fashion, so that they all are working towards a common goal. The need for collaboration and communication in a project grows as more and more sub-teams are involved. It is usually not advantageous to have multiple processes and concurrent sessions without a coherent communication pipeline [5]. More processes can slow down a collaborative decision-making process due to increased "needs" for timely communication and coordination [23] among teams, processes and computers. Thus, there is a limit to how far paralleling in CE can stretch. It would seem logical to adopt a modular communication

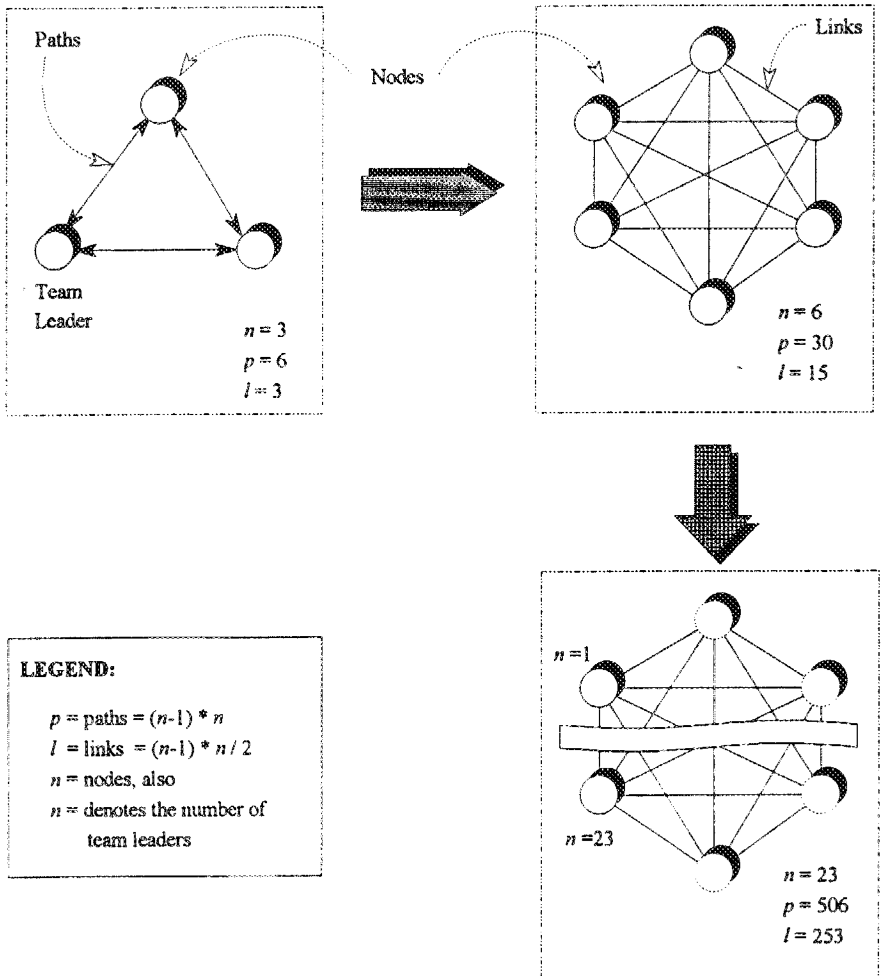


Figure 2. Growth in communication paths in a matrix of agents.

infrastructure, which can grow in an evolutionary manner. As the work-group grows and needs arise for more compute power and timely communication, newer (and modular) infrastructure can be added. The efficiency of communication is, thus, one of the major barriers to "paralleling" needed to do things and tasks in a Concurrent Engineering way [29]. Difficulties arise if the project is large and the activities are closely interdependent. There would be a need for skills at each vertical level corresponding to each horizontal function. This situation leads a skill matrix into a communication network that may be fully populated causing cross-links in all directions. This is shown in Figure 2. In a real situation, this may translate into a large number of time-consuming communication links that are required for tasks' completion. This may grind the progress of product development to a halt.

3.1. *Matrix of Communicating Agents*

Let us assume that a set of vertical agents (v columns) is interacting with a set of horizontal agents (h rows). Each agent could be a set of skilled groups (teams), a set of machines (or computers) or a set of processes. The matrix organization forms a network of interacting links or agents. Each cell of the matrix represents a communication node, which reports back on two different paths: one horizontal and one vertical. For example, agent $(1, 1)$ must communicate with agents $(1, 2), (1, 3), \ldots, (1, h)$ along the horizontal row, and with agents $(2, 1), (3, 1), \ldots, (v, 1)$ along the vertical column. Each agent requires $[(h-1)+(v-1)]$ communication paths. Since the total number of agents is $h * v$, the total number of paths p:

$$p = \text{total Number of Communication paths} \equiv [(h+v-2) * h * v],$$

$$\text{or } p = [h^2 * v + h * v^2 - 2 * h * v], \quad (2)$$

$$\iota = \text{Total Number of Links} = p/2 \quad (3)$$

Link is the union of the two paths, one going in forward direction and the other in reverse direction.

If $h = 12$; and $v = 11$, such that $(h + v) = 23$,

$$\text{total number of paths } (p) = 21 * 12 * 11 = 2772 \text{ and } l = 1386 \quad (4)$$

This discussion centers on the needs for communication among the various agents not just the interactions of "humans in teams." If "computers or machines" alone are part of this matrix of communicating agents, the number of communication paths may not present much problem. Computers do interact as long as communication circuits are open, power is on and they are properly programmed. The moment, the communicating agents comprising of elements such as human teams and processes enter the picture, the number of communication paths will be governed by equation (3).

3.2. *Cooperating Matrix of Agents*

In the above it is assumed that each agent (a team, a machine or a process), represented by a cell or node is required to communicate with its counterpart agent along the horizontal or vertical lines. Human team is the weakest link of this communication channel. This means, this occurs whenever interaction takes place with human teams—six out of nine forms of sharing and collaborations are in use. Since computers and processes can interact as long as communication circuits and networks are open and powered, the remaining four forms of communications would not be impacted as long as the human teams were not part of this process. In actual practice, it is hard to program the

ısuming communication links
ɔgress of product development

acting with a set of horizontal
(teams), a set of machines (or
ɔrms a network of interacting
ınication node, which reports
or example, agent (1, 1) must
orizontal row, and with agents
ıt requires $[(h-1)+(v-1)]$
v, the total number of paths p:

$h+v-2)*h*v]$,

(2)

(3)

ction and the other in reverse

$= 1386$ (4)

ımong the various agents not
or machines" alone are part
ɔmmunication paths may not
ımunication circuits are open,
ıt, the communicating agents
enter the picture, the number

ne or a process), represented
ıterpart agent along the hor-
of this communication chan-
lace with human teams—six
. Since computers and pro-
works are open and powered,
impacted as long as the hu-
e, it is hard to program the

computers and processes so those automatic interactions can occur without human interventions. In situations, when human interventions are required even if the collaborations are happening amongst computers/machines and processes, the number of communications links would still be governed by the rules of Equation (3). However, if we elect a leader-agent in each respective horizontal or vertical functional groups, in such a way, that each agent cooperates and communicates only with their respective horizontal or vertical leader-agents, the number of communications links can be minimized as follows:

There will be "h" leader-agents along the horizontal row, and "v" leader-agents along the vertical column. The communication can occur in 3 ways:

(a) between "v vertical functional agents or groups" and a representative leader-agent,

(b) between "h horizontal agents or groups" and a representative leader-agent, and

(c) among "the n leader-agents" themselves,

where, n is the total number of leader-agents in this **matrix of agents**. For an (h, v) matrix, n can be written as,

$$n = (h + v) \tag{5}$$

number of communication paths between "v" vertical functional agents or groups and the leader-agent $\equiv v$, (6)

number of Communication paths between "h" horizontal functional agents or groups and the leader-agent $\equiv h$, and

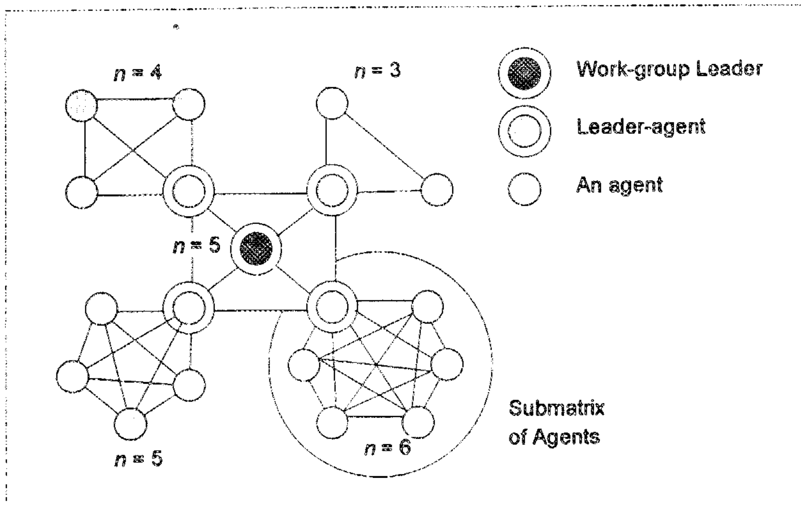
number of Communication paths among the leader-agents $\equiv (n-1)*n$. (7)

$$\text{Thus, total number of possible interactions} = (h+v)+(n-1)*n$$

$$= (n*n). \tag{8}$$

$$\text{Number of links} = = (n*n)/2. \tag{9}$$

Thus, for $n = 23$, the number of paths could be as large as 529. As illustrated by the equations (7–9) given above, the number of possible interactions is proportional to the square of the sum of the rows and the columns of a matrix. This number (e.g., 529) is however, many times smaller compared to communication path numbers (e.g., 2772) for a **matrix of agents**. The number of such interactions can be used as an indicator to determine whether or not the size of a chosen matrix of agents is practical or manageable. More thought is necessary in organizing such interactive agent-based projects. One solution is to divide the projects into smaller sub-units of several matrixes of agents that can be tackled relatively independently. This situation is shown in Figure 3. By dividing the initial matrix of agents of node-size 23 into five sub-matrixes (or network) of agents (say node-sizes 3, 4, 5, 6, and 5), the number of interactions can be significantly reduced. The number of communication paths is reduced from 529 to 111. This leads to a concept of ***cooperative matrix of agents***, where the number of communication paths amongst teams, computers and



Relation	Communication Path Counts					Submatrix Total	Flat Total
n =	3	4	5	6	5	23	23
p = n * n	9	16	25	36	25	111	529

Figure 3. Concept of cooperative matrix of agents.

processes is manageable. With this cooperative setup, matrix of agents no longer represents a bottleneck in getting the project done on time.

4. Modes of Cooperation

One of the CE challenges is to find which combination of tools and technology (such as distributed computing, networking, information sharing, process concurrence, hardware and software), will create an environment conducive to efficient communication, sharing, and cooperation. The good news is that in today's environment, there are many technological options available to choose this from. The bad news is that most of these technical approaches to information sharing are primitives—i.e., they are not based on sound Concurrent Engineering principles (see for instance [29]). Also, a concept such as cooperative problem solving has not been fully leveraged into newer generation of C4 (CAD/CAM/CIM/CAE) tools. There are many instances of cooperative problem solving in product design. This is discussed next.

4.1. *Interaction Matrix*

Figure 4 depicts an interaction matrix between ***mode of cooperation*** and ***degree of involvement***. The inner cells of the matrix show the various types of concurrency that are possible in a team environment. On the horizontal axis, in top row, four possible sets of work-group configuration are chosen to represent different modes of cooperation. A work-group configuration may consist of:

- ***Single User:*** A single user is responsible for design decisions.
- ***Co-operating user:*** Often in a team environment, some processes are sequential. A co-operating 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 of the downstream organizations who may be required to work on the uncompleted task. Other responsibilities may include exploring or replaying the logic and analysis of the specified product form or an understanding of the processes that produce it.
- ***Simultaneous Users:*** This means multiple users or product developers are accessing the ***Product Information Tools or Applications*** (PITA). The two situations in which this may occur are:
  - (a) The users may access the same design, tool, or application concurrently, or
  - (b) Different users may access or edit different versions of product information tool or application (PITA) at the same time.

On the vertical axis in the title column of Figure 4, the different possible degrees of involvement for these levels of users are shown. Each of the work-group configurations can have five levels of interaction, not all of which leads to concurrency. For example, a single user accessing his or her own PITA is sequential. Even if his partner (a co-operating user) accesses the information following his design works, the process is still sequential. Similar situations occur when they try to run these functions against their own data [18]. In other situations, concurrency is present in varying degrees—mild-to-strong. For example, concurrency occurs when a single user or co-operating user accesses data from other groups and runs them against the data from other groups in a computer environment tailored to their perspective on the design. Concurrency is further enhanced when users, at geographically separated locations, perform the aforementioned operations simultaneously.

The degree of concurrency increases as we move from top to bottom and from left to right in Figure 4. The situation depicted by the bottom rightmost rectangle (location [5,4]) provides the largest degree of concurrency. It may be noted from the table that a style of concurrency, where simultaneous users run the same version of PITA against their own data (the location [2,4]) is characterized as both Sequential Engineering (SE) and Concurrent Engineering (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 module. Even though the two users can work in parallel, the

Modes of Cooperation → / Degree of Involvements ↓		Work-group Configuration			
				Simultaneous Users	
		Single User	Co-operating Users	Different Versions	Same Version
		1	2	3	4
Access own product interaction tools or applications (PITA)	1	Sequential Engineering (SE)	(SE)	(SE)	(SE)
Run against their own DATA	2	(SE)	(SE)	(SE)	Sequential/ Concurrent Engineering (SE/CE)
Access PITA belonging to other work-groups	3	(SE/CE)	(CE)	(CE)	(CE)
Access DATA belonging to other work-groups	4	Concurrent Engineering (CE)	(CE)	(CE)	(CE)
Access both PITA and DATA from other work-groups	5	(CE)	(CE)	(CE)	(CE)

Figure 4. Interaction matrix between modes of cooperation & degree of involvement.

...tion

Simultaneous Users	
Different Versions	Same Version
3	4
(SE)	(SE)
(SE)	Sequential/ Concurrent Engineering (SE/CE)
(CE)	(CE)
(CE)	(CE)
(CE)	(CE)

...olvement.

changes cannot be posted until the latter has had the chance to review the changes made by the first user. The second user continues only when the first user has finished with his version of the design module. For a critical module in the design, this particular mode of execution may indicate that CE is no faster than sequential engineering (see Figure 5). Many CAD/CAM vendors are resorting to standards to provide them with a cross-platform, software-independent environment for collaborative cooperation [3]. Autotrol technology in Denver, for instance, recently announced its Mozaic architecture that implements industry standards like STEP and the Object Management Group's Common Object Request Broker (CORBA) [4]. Some database systems, e.g., ROSE [13], thus supports this type of concurrency by allowing multiple users and applications to edit different versions of the same design module (see Figure 6). Some mechanical design automation CAD/CAM vendors (e.g., SDRC, Milford, Ohio) have unveiled a new generation of software products to support concurrent associativity for multi-user, multi-application product development team [26]. Concurrent associativity enables continuity amongst the master model and its linked application data sets. For example, a designer creates a master model for "version A." A draftsman can work on drawings, an analyst can do a stress analysis, a manufacturing engineer can do a process plan or a tool path all for "Version A," while the designer does refinements towards an eventual "Version B." Yet when "Version B" comes around, each specified application will be automatically regenerated to reflect changes made to the master model. This 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 [18].

4.2. Managing Interactions

Managing interactions between a user and an application such as PITA are not easy. Several methods are proposed to manage interactions [18]. The two popular methods are: (a) via linguistic description; and (b) icon displays and direct manipulation. A linguistic description provides an intermediary communication interface between a user and a PITA. Icon-based displays provide a model-world communication interface and enable the users to directly engage and manipulate PITA objects.

Interactions are further complicated by the multiplicity of media in which such interactions can possibly take place, such as voice, video, audio and scripts. There are many techniques for coordinating the presentation of information in different media. Usually, types of information or knowledge determine what media would be ideally suited.

Enabling tools plays an important role in facilitating the cooperation between the teams that may not be co-located. With the lack of such cooperative tools, the performance of a team that works effectively face-to-face in a conference room deteriorates when they communicate via computers. Teams need all sorts of knowledge that provide or replicate the same feeling or environment as they get in an interactive conference room. Participating teams need both—the type of knowledge that is necessary to be an effective collaborative partner, and the knowledge that has proved useful in practice. In addition, teams need an appropriate architecture for deploying these knowledge bases into tools that can be used as an effective collaborative partner. Technology of collaboration, such as Electronic mail

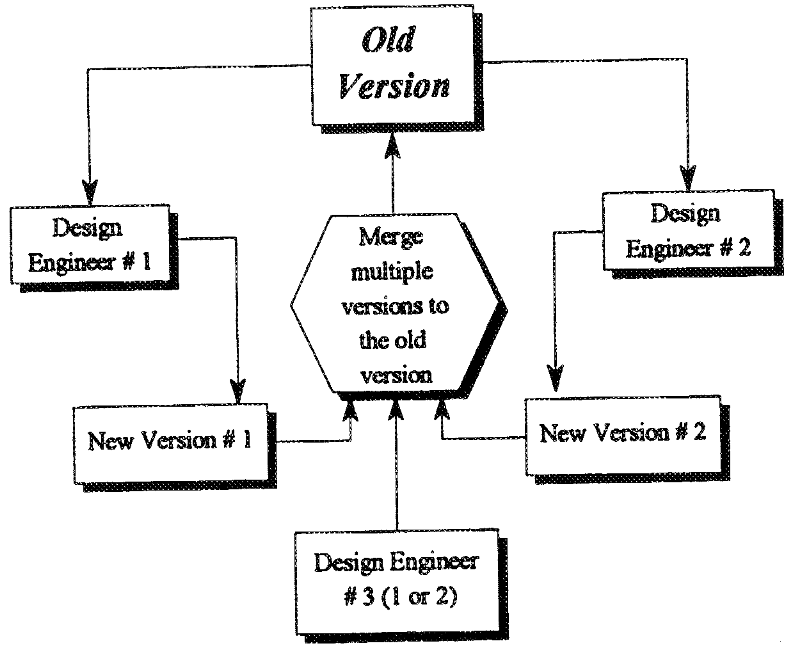


Figure 5. Simultaneous users working with multiple versions of design (Sequential Engineering).

(E-mail) systems, Lotus Notes and workflow-based systems can assist in this process. The level of technological support for collaboration must correspond to the degree of cooperative effort required and the complexity of the CE environment [18].

5. Paralleling of Responsibility

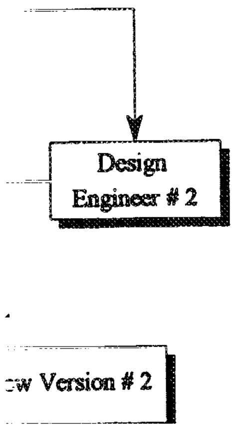
Distribution of responsibility among the teams, machines (or computer), products, and processes should be such that they maximize the effectiveness of the "joint cognitive system [30]." A joint cognitive system means a complex product manufacturing environment where:

⇒ Business partners and other CE work-groups (such as personnel, virtual, logical and technology teams) share product design and development responsibility.

⇒ The product is differentiated using a systematization approach.

⇒ Modular manufacturing and flexible production concepts are utilized.

Joint cognitive system requires distribution of responsibility among sharing partners. Work breakdown structures (WBS) establish a framework through which work-group re-



equential Engineering).

can assist in this process. The
ond to the degree of cooperative
18].

(or computer), products, and
ss of the "joint cognitive system
ct manufacturing environment

personnel, virtual, logical and
nt responsibility.

proach.

nts are utilized.

bility among sharing partners.
through which work-group re-

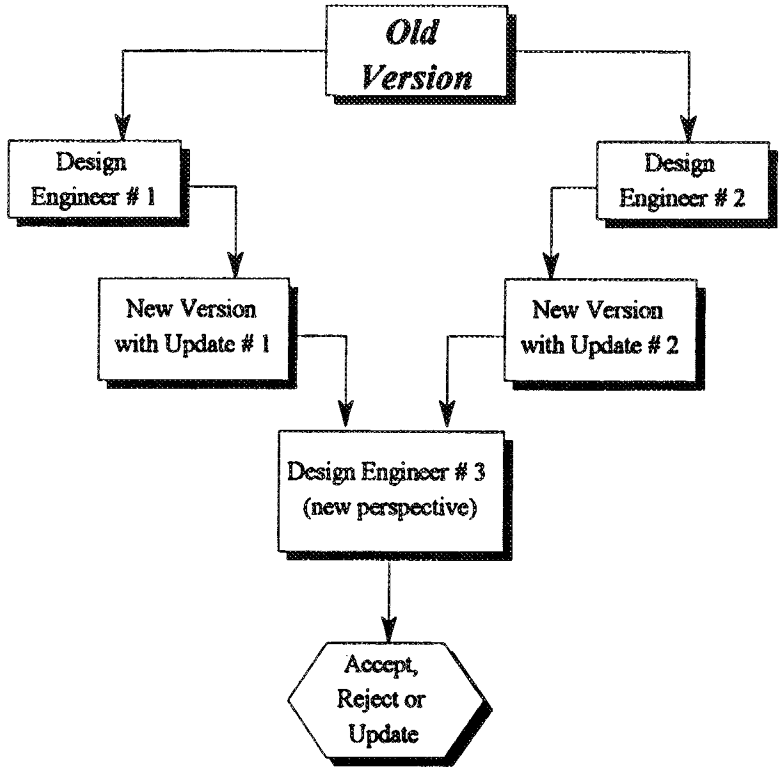


Figure 6. Simultaneous users working with multiple versions of design (Concurrent Engineering).

sources can be allocated [29] in accordance with the responsibility and shared among partners. The following factors influence how such allocations can be made:

- ***Interactions and Consensus:*** Interactions between the teams must occur in advance, and a consensus reached in deciding as to how the sets of responsibilities in WBS will be distributed among the sharing partners.
- ***Communication and Coordination:*** Types of communication and coordination necessary to collaborate on a task determine the actions and basis of how to distribute responsibility among the agents so that they can work in parallel [12]. Examples of coordination include delegating responsibility, reporting, or evaluating results.
- ***Knowing human behavior:*** Knowing how teams of people work. Studies have shown that humans interweave problem-solving tasks with problem-setting tasks [15]. They create personalized workspaces to organize their thoughts and tasks [19].

- *Level of skills:* The shift of responsibility from human to the intelligent machines (computers) depends upon the types of work, such as routine or repetitive tasks [19]. Repetitive tasks can be better handled by a computer system [27]. However, if it is a skilled job or one of a kind, it could be handled better by a human being with an ingenious skill set [28].

- ***Efficiency of communications among the agents:*** The best way to figure out inefficiencies of communications is, first, to ask the individual units or departments how they are performing their business. An IFLOW "flow charting" methodology may be used to capture the flow of information through various departments [29]. One way to gain some insight into a process is for everyone to cooperate and communicate in real time with what they are doing, how they are doing it, and who needs access to the information. A full commitment and determination on the part of the management may be required for its success. It is vital to agree on a common set of symbols and methods for depicting the information derived from the teams.

6. Modeling Collaborative Behavior

The cognitive model captures human behavior in teams and collaborative situations. Most CE environments involve a carefully orchestrated interplay among CE teams, processes and machines [23]. Information models are incomplete without a common framework of cognitive understanding. Takeda, Hamada, Tomiyama and Yoshikawa [34] conducted design experiments and protocol analyses from which a cognitive design process model was derived. There are no useful cognitive models for accommodating human behavior such as different points of view, capturing thinking rationale, and various mind-sets in a product realization process [14, 19].

6.1. Convergence of Collaborative Thinking

Convergence of collaborative thinking is an important feature of a constancy-of-purpose-oriented work-group [32, 29]. Figure 7 describes the stages of teams' progress through which a convergence of collaborative thinking takes root. Five stages are shown to describe a progression from a closed state of mind to a converged mind set [32, 29]. The stages represent a change in the mental state or an attitude, expressed psychologically, towards building a consensus amongst the CE teams [15]. These stages are merely pointers or intermediate steps and do not necessary represent actual phases.

- In the **beginning stage**, most teams possess a closed mind, "don't want, don't ask" attitude, afraid of unknowns and often feel threatened. With time, members of each team develop an understanding of each other's point of view. They begin to appreciate importance of their disciplinary contributions at various points along the way and their impact to the product goals' realization. The mental state or attitude sails through a series of changes:

n to the intelligent machines
ıtine or repetitious tasks [19].
ystem [27]. However, if it is
er by a human being with an

: best way to figure out inef-
inal units or departments how
harting" methodology may be
s departments [29]. One way
ooperate and communicate in
t, and who needs access to the
e part of the management may
on set of symbols and methods

collaborative situations. Most
among CE teams, processes
without a common framework
nd Yoshikawa [34] conducted
ognitive design process model
commodating human behavior
le, and various mind-sets in a

ire of a constancy-of-purpose-
es of teams' progress through
ve stages are shown to describe
mind set [32, 29]. The stages
essed psychologically, towards
stages are merely pointers or
ases.

mind, "don't want, don't ask"
. With time, members of each
view. They begin to appreciate
; points along the way and their
state or attitude sails through a

- **Stage second**, it moves from a "*don't want, don't ask*" attitude initially to a "*don't want but ask*" attitude.
- **Stage third**, it goes from "*don't want but ask.*" to "*want and don't ask*" attitude.
- **Stage fourth**, it goes from "*want and don't ask.*" to "*want and ask* attitude. Collaborative thinking extends teamwork concepts and communication capabilities and applies them to formulate how a work will actually be done. Considering this, the teams form expanded group knowledge of the whole in such a way that it allows each team member to completely understand the needs and goals of his fellow members. This collective mind-set allows members to grasp each other's differing point of views, which results in confidence building and increased level of cooperation.
- In **stage fifth**, it moves to "*seek, desire and listen*" (see Figure 7). At this stage, their minds get converged——teams feel more secured than ever before. With convergence of collaborative thinking, the impact of cross-functional teaming on the product realization is maximized [7]. The progression through these states or phases is motivated from an initial feeling of being threatened to a final feeling of security and welcome.

⇒ ***Design Reviews:*** Design reviews (made out of select cross-functional review teams) are an efficient method to [21]:

(a) Monitor the progress of a project;

(b) Facilitate reporting and appraisal of results to management; and

(c) Keep the teams' interest in line with the common set of consistent project goals. Design reviews promote a team oriented review strategy, which optimizes the team's collective talents in problem solving. A carefully timed and organized design review is not an engineering inspection, but rather a value-added process of improving the design features. During a design review process, teams collectively add monetary values to a product or a process function. The design reviews allow teams to build consensus and is often considered a first step towards teams' progress leading to convergence in collaborative thinking [29]. During design review, it is important to stick with a standard review format and timing.

6.2. *Factors that Affect Collaborative Behaviors*

To develop goal-oriented or "constancy-of-purpose" solutions, a successful collaborative environment—among the participating teams and their members—is required [32, 6]. However, it is not easy to model the cognitive behavior of a collaborative and motivated goal-oriented work-group [14]. Empowerment and reward systems can help in motivating the teams and hopefully can remove some common barriers of communication. Implementing such tasks may not be difficult. However, developing collaborative behavior in a goal-oriented work-group is a more challenging task, since these are more related to the social and cultural roots of team-members, which may be difficult to change [32]. There are some

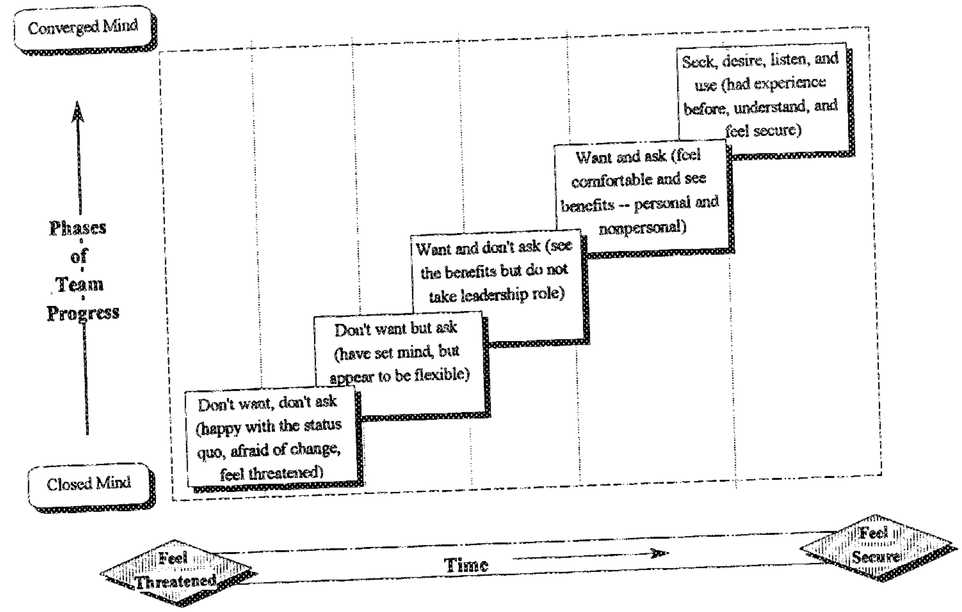


Figure 7. Phases of teams' progress leading to convergence in collaborative thinking.

behavioral patterns that help in collaboration more than others. Knowing and understanding the differences in behavior patterns can be helpful in organizing a better work-group for CE. The factors that affect collaborative behaviors are:

- Cognitive Framework
- Occupational Culture
- Education & Background
- Value System

This is shown in Figure 8. Cognitive framework, or nature, is how a team member thinks and acts during decision making. In individuals, certain cognitive skills tend to be more developed than others with one half of the brain (hemisphere) being more utilized than the other half [1]. Individuals with creative minds are primarily right brained, while those with judicial/analytical minds are primarily left brained. Rarely are people exclusively right or left brained. More often they are in-between, perhaps leaning towards one side more than the other. For example, engineers generally have a more developed left brain, while designers have a more developed right brain [1]. Human culture belongs to the affective aspect of one's creative mind. The topic is debatable and how these postulated functional differences impact the actual thinking in individuals is at best indeterminate at this time. Occupational and organizational culture refers to a common tradition of functionally oriented departments

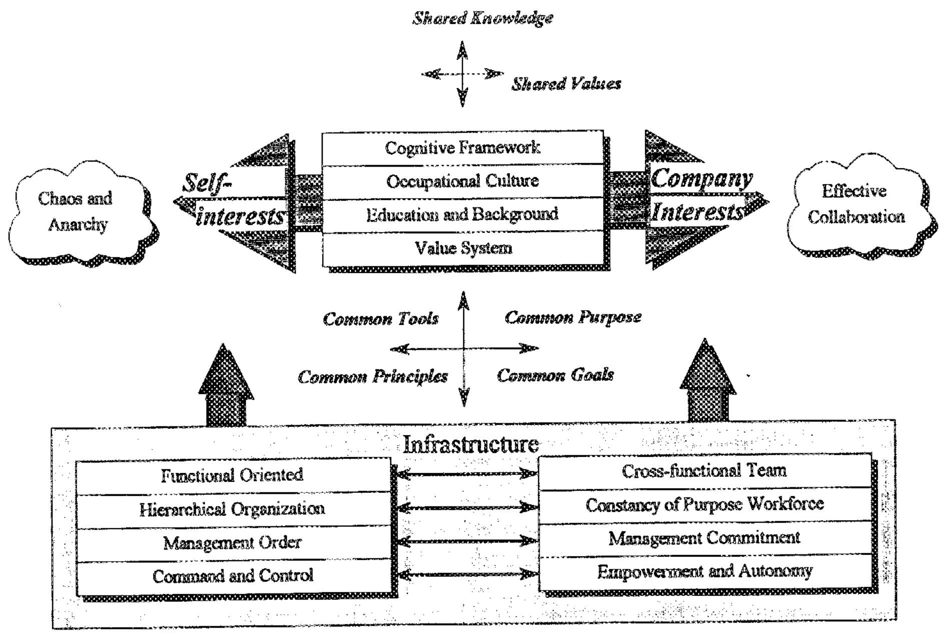


Figure 8. Factors that affect collaborative behaviors.

in which the work is often *like "deeply ingrained routine activities* [21]." When teams of work-groups are brought together in a CE setting having initially different occupational and divisional cultures, they have a limited, and often restricted, view [15]. Often, this is inherited with "native-view" paradigms that create organizational cultural conflicts [15]. Differences in educational background in the work-groups also result in differing "thought-views," creating further possibilities of conflict. Each team member may have his/her own value system, depending upon what each team believes in. Some may have a stronger belief in one area than others.

It is important to realize that the above behavioral factors are the forces behind creating strong cognitive skills. It would be desirable to develop the same level of cognitive skills working at the "*company level*" as on the "*personal level*." This way, the teams would be able to deliver the same level of enthusiasm and dedication that existed on a personal level. All of the above behavior frameworks are important ingredients in making a company successful. For example, a value system that is based on the combined interests of the employees, customers, and the company would be ideal. Similarly, a company culture that is oriented towards building on the strengths of each member, group, department or division of the company would be a desirable thing to have. Cultural views of the teams that are based on "local or native needs" would be detrimental, however, developing a cultural view that is conducive to global or company-wide interests would be a good use of culture. An example of this scenario is when work-groups bring their own occupational or divisional "native-views" (of the culture) to define an "outcome" that is based on a "global sense of priorities" or a "constancy-of-purpose."

7. Physical and Electronic Proximity

Communication is the cornerstone of success in CE. The ability to interact constantly with teammates or work-groups, while the design is taking shape, lays the foundation for consensus building and rapid decision-making. There are many ways to communicate. One approach is to move the teammates together, i.e., physically locate them on the same premises. This creates ***physical proximity***. An equivalent measure would be an "***electronic proximity***." Electronic proximity is a class of virtual links that could make the travel involved quite painless. Proximity could be achieved through electronic interchange of messages, CAD models, drawings, sketches, and other communication modes (such as data, voice, video) [31]. Virtual team approach, networking, and groupware tools are available today to aid team members in communication and decision making [11]. However, this may not be matured enough to replace a natural interactive environment, which comes from physical proximity. Meetings and physical proximity do maximize various modes of communication (such as body language, group dynamics, and personal touch). Geographical collocation develops a natural appreciation of each team's capabilities, problems, and opportunities. When people work side by side, they begin appreciating each other's opinions. Cultural or language barriers that traditionally prevent them from cooperating and optimizing their overall output begin breaking down. However, when the groups get larger, the advantage diminishes because there may be too many people to deal with at one time [21]. Teammates from diverse disciplines (such as electronics, mechanical, civil, and industrial) also face difficulties in prioritizing the aspects of the problem and their relative importance. The impact of physical proximity on turf wars and territorial issues has produced mixed results. It is not obvious when members stop seeking credit for their own contributions and begin working towards a common set of consistent goals. Breaking down the barriers, getting people outside their organizational boxes, and working together as a coherent ***team of teams***, are some primary challenges, most CE organizations are facing today. The next section discusses methods and techniques of using computing and communication devices to facilitate cooperation and concurrency during an integrated product development.

8. Resource Sharing Techniques and Modeling

The information generated in CE—such as product specifications, goals, enterprise data, process, and knowledge—must be shared among work-groups [17]. It is useful to describe common information as shared resources so that each work-group member can access this information irrespective of their tasks, sources, or place of work. Such shared resources may include product knowledge bases, product specification taxonomy [18], process knowledge bases, PDMS, process specification taxonomy [23]. Such resources may also include work breakdown structure (WBS), product breakdown structure (PtBS) tree, process breakdown ɪt is based on a "global sense of

ne ability to interact constantly
ıg shape, lays the foundation for
re many ways to communicate.
ysically locate them on the same
ent measure would be an "***elec-***
l links that could make the travel
rough electronic interchange of
communication modes (such as
g, and groupware tools are avail-
lecision making [11]. However,
ctive environment, which comes
ity do maximize various modes
ics, and personal touch). Geo-
ch team's capabilities, problems,
begin appreciating each other's
prevent them from cooperating
. However, when the groups get
many people to deal with at one
lectronics, mechanical, civil, and
of the problem and their relative
rs and territorial issues has pro-
top seeking credit for their own
consistent goals. Breaking down
oxes, and working together as a
ıost CE organizations are facing
es of using computing and com-
ncy during an integrated product

ifications, goals, enterprise data,
oups [17]. It is useful to describe
rk-group member can access this
work. Such shared resources may
xonomy [18], process knowledge
resources may also include work
: (PtBS) tree, process breakdown

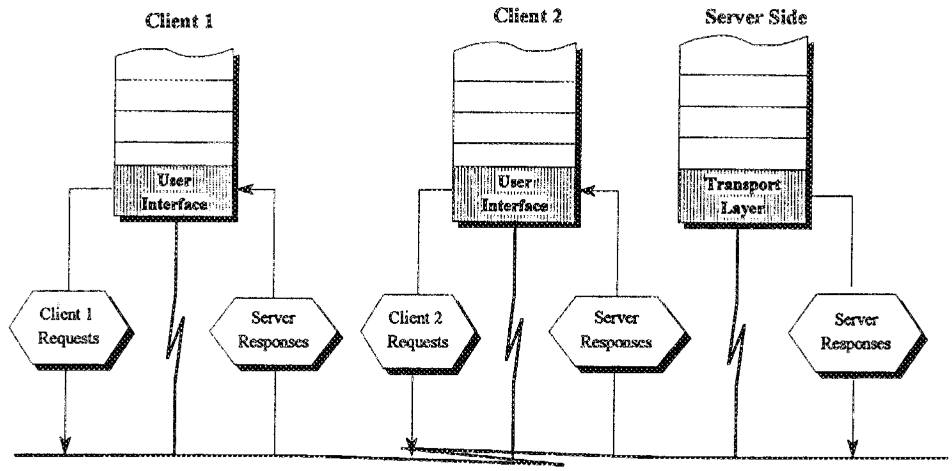


Figure 9. Client/server model.

structure (PsBS) tree, and network taxonomy (see section 6.4.1 of volume I [29]). Shared resources can include the concurrent work-group taxonomy (also referred as WBS) for both human and computer supported activities. Work-group taxonomy can include distribution of work for each work-group within each CE team, portable applications for distributed processing, common services or execution environment, and common shared resources. Graphical User Interface (GUI) mainly provides a friendly front-end for accomplishing the man-machine interactions [22]. It is not a means to accomplish computational resource sharing. The five most popular approaches to resource sharing are discussed in the following.

8.1. Client/Server Traits

The client/server trait represents a software-defined model for information sharing. It is made out of two entities, a client and a server. The client requests services from a server, such as retrieval of data, or printing of a document. The server processes the request, performs the service, and returns the results to the client. Establishing a streamlined communication involves configuring an "open-system" architecture that provides distributed computing in which data and resources can be shared easily. One type of architecture that provides such openness is a client/server model [35]. A client-server approach is shown in Figure 9. It is generally characterized by a division of an application into components with the flexibility that each component could run on a different network or a computer. A transport layer provides the necessary communication protocol among the client-team interface and the server applications.

8.2. Distributed Processing

In this environment, concurrent teams tap the local processing power of "client" computers (personal computers and low-end workstations, for example) to "servers" (other high-end workstations and multi-user platforms) [35]. When clients request some information, servers respond by running an application locally and then delivering the requested information over the network (Figure 9).

8.3. Work-group Computing

Globally interconnected clients and servers on standard-based networks can produce improved communication and greater access of data throughout an organization. The key benefit of the client/server form of integration lies in its ability to foster work-group computing (WC). In recent years, an architecture called "work-group computing" has emerged [25] due to the efforts of many workstation vendors competing for CE market share [SUN, HP, DEC, IBM among others]. It provides a better-integrated environment for CE compared to LAN based PC networks [35]. With "work-group computing," computing tasks are distributed among "clients," consisting of powerful desktop workstations and "servers" that store and manipulate information. Work-groups are provided a view into the computing complex through a "window" created by the client workstation. They may, however, be linked with virtually anyone else. The X-display servers can range from a printer, to a facsimile, to a workstation.

The WC implies a transparent access of data/system resources to the linked applications or work-groups, independent of their locations, installations, processor hardware, operating systems, and programming languages [18]. WC incorporates four different types of interactions among the work-group members with respect to dimensions of time and space [11]. Figure 10 shows schematically these four shared computing environments among the work-group members. These environments can be characterized as:

◇ *Face-to-face:* Interaction occurs at the same time and at the same place. This environment supports face-to-face interactions amongst work-group members regardless of temporal or geographical alignment with other group members. Examples include interactive meeting, collaboration laboratory, design reviews among others.

◇ *Distributed Synchronous:* Interaction occurs at the same time, but at different places. Lotus Notes, for instance, supports a type of distributed synchronous interaction environment. Using Notes work-group members can work on the same task regardless of their physical location or place of work. Video-conferencing is a groupware technology that enables geographically dispersed teams to conduct face-to-face meetings in real time, by combining interactive video, audio, and graphic/document display capability. Meeting on the network is a similar idea.

◇ *Asynchronous:* Interaction occurs at different times, but at the same place. Examples include multi-purpose equipment (for instance cutting, milling, and threading), and other resources installed in a plant, or database repository.

◇ *Distributed Asynchronous:* Interaction occurs at different times and at different places. "Distributed asynchronous" is useful in providing functions such as computer-intensive processing, electronic mail, automatic printing, and filing.

And while mainframes, mini and microcomputers will suffice for general needs, there will always be a need for more specialized machines. That is why work-group computing is built around client-server architectures. In addition, modern databases can be distributed over many different machines, so work-groups can create and execute applications locally on their own computers [33]. These applications could look and feel just like separate programs. Here, individual workstations—the clients—handle the local processing needs. While, the server has the power and capacities to access data from distributed databases beyond that of an individual workstation [3]. They may provide heavy-duty number crunching, distributed databases, and links to outside resources. When a work-group runs the applications from any workstation, it draws on the resources of all the related applications and databases over the network no matter where they physically reside or are stored. This distributed concept maximizes the computer power needs of the work-groups with least amount of investments [25].

8.4. Network Taxonomy

For a client and server to communicate, they must use a common network protocol, even though multiple protocols can be used by each individually. Examples of protocols include TCP/IP, DECnet, SNA, and others. Many computer vendors, in order to optimize the use of different computers, support tools that allow client/server interactions for different protocols [25]. Examples include the X-Window System, Distributed Name Service (DNS), Remote Procedure Call (RPC). RPC is a procedural language mechanism for distributing application program procedures to remote network locations [35]. The remote procedures become servers. The local client programs invoke the remote procedures as if they were local procedures. X-Windows is also a network-based windowing system. Applications developed for the X-Windows system are hardware independent, that is, the application developer uses standard graphics protocols [Krisnamurthy and Law, 1995]. The application is shielded from the details of individual implementation of the hardware-specific graphics display features.

8.5. Cooperative Computing

Another part of the CE solution is to provide cooperative computing among applications, work-groups and the computers using enterprise models or network taxonomy, as required [8]. Work-groups benefit because such models and taxonomy are common to all X-Window based workstations. Developers can move their applications from one UNIX platform to another or run the applications with relative ease, as long as both platforms support X-Window-based interface and are networked together. A work-group member can run one application on one workstation and, at the same time, open an X-window interface on another computer belonging to another team-member and run his or her application [10].

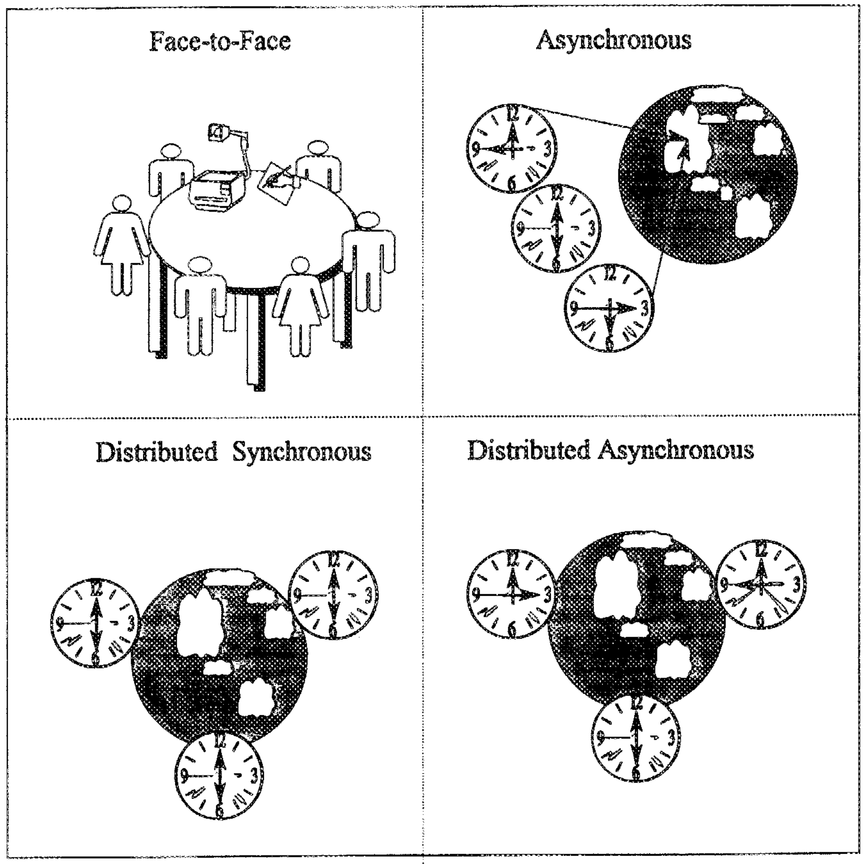


Figure 10. Types of interactions among members of the work-group computing.

The above features clearly provide a "freedom of choice" for information management as well as system and network administration to structure and cost-effectively manage the current and future (such as client/server) computational needs [2].

9. Concluding Remarks

The paper described five techniques of sharing and collaboration among various work-groups, computers and processes. For accomplishing the needed collaboration (and to facilitate concurrent engineering), a concept and a model for a work-group computing were implemented at Delphi Divisional units of EDS and General Motors. The intention was to help teams work together and automate work-group processes, including engineering, manufacturing, and other complex tasks. Since open systems are commonly built around industry standards, they can be integrated easily with existing equipment from various competing vendors. Work-group computing provides the power to perform individual tasks with ease while opening up the possibilities of information sharing among parallel work-groups [5].

Figure 11 illustrates the shift in key characteristics that were derived from its implementation when moving from a personal computing environment to a work-group computing environment. The shift pulled everyone—teams, computers, networks—transforming disparate computers into one flexible, easy-to-use client/server based system. The individual task-oriented environment of personal computing activities became a set of goal-oriented parallel work-group activities. The use of distributed database over the network became the mainstream norm for file management as opposed to local database residing in one's own personal PCs [18]. The use of personal planner and a calendar was replaced by a group scheduling system and an electronic workflow resource management system.

With work-group computing, a manufacturing organization—small and large—can employ CE concepts early into their product development process and therefore can realize all the benefits that come from using this.

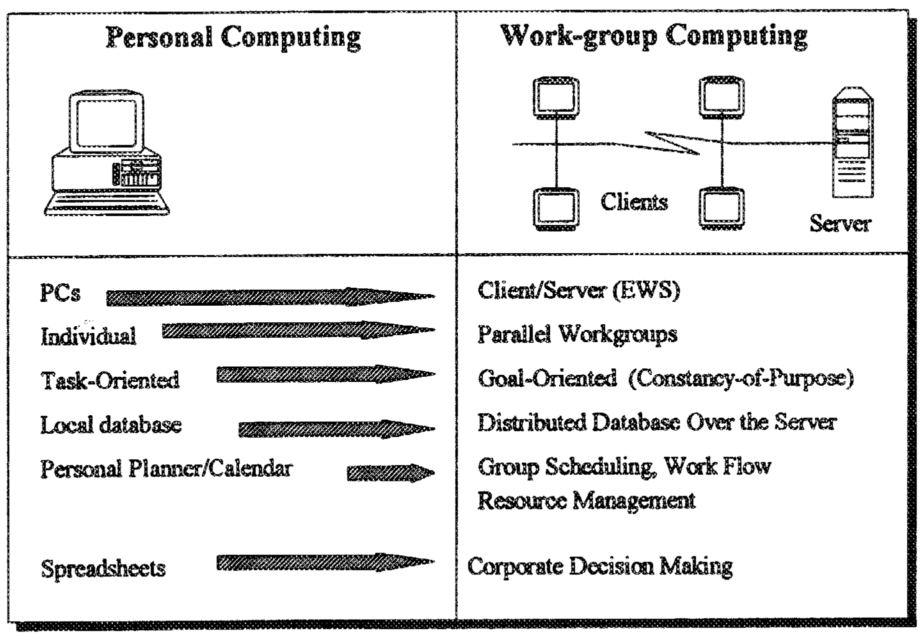


Figure 11. Shift from personal computing to work-group computing.

References

1. J. R. Anderson, 1993. *Rules of the Mind.* Lawrence Erlbaum Associates, Inc.: New Jersey, 1993.
2. N. Bajgoric, "Organizational systems integration: Management information systems perspective." *Concurrent Engineering: Research and Applications—An International Journal* 5(2), pp. 113–122, June 1997.
3. C. NE. Branki, "The acts of cooperative design." *Concurrent Engineering: Research & Applications (CERA)* 3(3), pp. 237–246, September 1995.
4. L. Curran, "STEP bridges the way to better product modeling." *Machine Design* 66(6), pp. 137–142, 1994.
5. M. R. Cutkosky and J. M. Tenenbaum, "Providing computational support for concurrent engineering." *International J. of Systems Automation: Research and Applications* 1(3), pp. 239–261, 1991.
6. W. E. Deming, 1993. *The New Economics.* MIT Center for Advanced Engineering Study: Cambridge, MA, November 1993.
7. D. Dimancesen, 1992. *Seamless Enterprise: Making Cross-functional Management Work.* Ist Edition, Harper Collins: New York, NY, 1992.
8. J. Dong, "Organization structures, concurrent engineering, and computerized enterprise integration." *Concurrent Engineering: Research & Applications (CERA)* 3(3), pp. 167–176, September 1995.
9. Mike Donlin, "Data management tools the framework to concurrent engineering." *Computer Design* 30, p. 28, April 1991.
10. DICE Initiative in Concurrent Engineering-DARPA, *Red Book of Functional Specifications for the DICE Architecture.* Technical Report, Morgantown, WV: Concurrent Engineering Research Center, West Virginia University, (February 28), 1989.
11. Ellis, Gibbs, and Rein, "Groupware: Some issues and experiences." *Communications of the ACM* 34(1), January 1991.
12. R. Fruchter, K. A. Reiner, G. Toye and L. J. Leifer, "Collaborative mechatronic system design." *Concurrent Engineering: Research & Applications (CERA)* 4(4), pp. 401–412, December 1996.
13. M. Hardwick, et al., "ROSE: A database system for concurrent engineering applications," in *Proceedings of the Second National Symposium on Concurrent Engineering*, Morgantown, West Virginia: Concurrent Engineering Research Center, February 7–9, 1990, pp. 33–65.
14. D. Herling, D. G. Ullman and B. D'Ambrosio, "Engineering decision support system (EDSS)," in *Proceedings of the Design Engineering Technical Conferences*, Boston, MA., Jadaan, Ward, Fukuda, Feldy and Gadh, (eds.), DE-Vol. 83, Volume 2, ASME Press: New York, September 17–20, 1995, pp. 619–626.
15. B. Huthwaite, 1994. *Strategic Design: A Guide to Managing Concurrent Engineering.* The Institute of Competitive Design: Rochester, MI., 1994
16. R. M. Jones and E. A. Edmonds, "Supporting collaborative design in a seamless environment." *Concurrent Engineering: Research & Applications (CERA)*, 3(3), pp. 203–212, September 1995.
17. D. Kahaner, S. Lu, F. Kimura, T. Kjellberg, F. Krause and M. Wozny, "First CIRP international workshop on concurrent engineering for product realization—Workshop results." *Concurrent Engineering Research in Review* 5, pp. 6–14, 1993.
18. K. Krishnamurthy and K. H. Law, "A data management model for design change control." *Concurrent Engineering: Research & Applications (CERA)* 3(4), pp. 329–343, December 1995.
19. M. Klein, "iDCSS: Integrating workflow, conflict and rational-based concurrent engineering coordination technologies." *Concurrent Engineering: Research & Applications (CERA)* 3(1), pp. 21–28, March 1995.
20. M. Klein, "Capturing design rationale in concurrent engineering teams." *IEEE Computer*, January 1993.
21. M. Klein, "Supporting conflict management in cooperative design teams." *Journal on Group Decision and Negotiation* 2, pp. 259–278, 1993.
22. J. W. Lewis, "An approach to applications integration for concurrent engineering." *Proceedings of the Second National Symposium on Concurrent Engineering*, Morgantown, West Virginia, February 7–9, 1990, pp. 141–154.
23. C. Menzel and R. J. Mayer, "Situations and processes." *Concurrent Engineering: Research & Applications (CERA)* 4(3), pp. 229–246, September 1996.
24. G. R. Olsen, M. Cutkosky, J. M. Tenenbaum and T. R. Gruber, "Collaborative engineering based on knowledge aharing agreements." *Concurrent Engineering: Research & Applications (CERA)* 3(2), pp. 145–160, June 1995.
25. R. Peters and H. Kress, "Distributed synchronous interaction examined closely." *Concurrent Engineering: Research & Applications (CERA)* 5(3), pp. 219–232 September 1997.

26. B. Prasad, R. S. Morenc and R. M. Rangan, "Information management for concurrent engineering: Research issues." *Concurrent Engineering: Research and Applications—An International Journal* 1(1), pp. 1–19, March 1993.
27. B. Prasad, "CE plus CIM equals IIS: Intelligent information systems," in *Proceedings of the 1995 ASME Mechanical Engineering Congress and Exposition*, San Francisco, CA, R. Rangan, (ed.), ASME Press: New York, NY, November 12–17, 1995.
28. B. Prasad, "On influencing agents of CE." *Concurrent Engineering: Research and Applications—An International Journal* 3(2), pp. 78–80, June 1995.
29. B. Prasad, 1996. *Concurrent Engineering Fundamentals, Volume I: Integrated Product and Process Organization*. Prentice Hall PTR: New Jersey, 1996.
30. B. Prasad, 1997. *Concurrent Engineering Fundamentals, Volume II: Integrated Product Development*. Prentice Hall PTR: New Jersey, 1997.
31. Jon Stark, 1992. *Engineering Information Management Systems: Beyond CAD/CAM to Concurrent Engineering*. Van Nostrand Reinhold: New York: NY, 1992.
32. J. B. Smith, 1994. *Collective Intelligence in Computer-based Collaboration*. Lawrence Erlbaum Associates, Inc.: New Jersey, 1994.
33. C. Sun, and Y. Zhang and Y. Yang, "Distributed synchronization of group operations in cooperative editing environment." *Concurrent Engineering: Research & Applications (CERA)* 4(3), pp. 293—302, September 1996.
34. H. Takeda, S. Hamada, T. Tomiyama and H. Yoshikawa, "A cognitive approach to the analyses of design processes," in *Proceedings of the ASME Design Theory and Methodology Conference, Design Theory and Methodology—DTM '90*, DE-Vol. 27, J. R. Rinderle, (ed.), ASME, pp. 153–160, 1990.
35. K. Willett, "Networking advances will anchor concurrent engineering." *Electronic Design* 40(1), p. 6, 1992.