CHAPTER 12

CONCURRENT ENGINEERING TECHNOLOGIES

V. Jagannathan
Y. V. Reddy
K. J. Cleetus
K. Srinivas
R. Karinthi
Concurrent Engineering Research Center
West Virginia University
Morgantown, West Virginia

12.1 INTRODUCTION

Effective collaboration among members of a team is the key to success, whether the team is a group of engineers designing a new engine or a group of physicians planning a medical procedure. This is now widely recognized, as can be seen by the numerous national initiatives emphasizing teamwork such as Concurrent Engineering (CE)[1], Total Quality Management (TQM), Integrated Product Development (IPD), Open System Architecture for CIM, and the Virtual Enterprise (VE).

Advances in database and networking technology, Internet technologies, groupware, multimedia, and graphical user interfaces, as well as a steep drop in the cost of computing, make possible the creation of a truly collaborative environment that transcends the barriers of distance, time, and heterogeneity of computer equipment. The ideal collaborative environment will enable any member of a team to communicate spontaneously, and thereby collaborate, with any other member of the team. This chapter provides an overview of technologies that facilitate geographically distributed teams to work together. Four primary categories of infrastructural services are needed to support collaboration: collocation services; coordination services; information-sharing and integration services; and corporate history management services. These are discussed in the remainder of the chapter.

Numerous people over the past eight years contributed to the development of the material presented here. In particular, Ravi Raman, Dan Nichols, and Felix Londono have contributed to various versions of the material.

This work was funded in part by DARPA grant number MDA972-91-J-1022 and NASA grant number NAG 5-2129, awarded to the Concurrent Engineering Research Center at West Virginia University.

Mechanical Engineers' Handbook, 2nd ed., Edited by Myer Kutz.
ISBN 0-471-13007-9

12.2 COLLOCATION SERVICES

Informal meetings and scheduled conferences are essential for teamwork, since they provide opportunities to inform others of ongoing work, consider cross-functional issues, and negotiate to harmonize viewpoints across multiple perspectives. Studies have shown that knowledge workers spend a large percentage (20–70%) of their time attending meetings and conferences. The parallelism implicit in concurrent engineering requires that team members perform simultaneous work activities without waiting for each subteam to report its results; the penalty for this method is a lack of consistency. Therefore, the basic concurrent engineering process itself envisages periodic meetings to bring about convergence.

When people who are geographically dispersed are required to meet at the same location, however, significant travel time, money, and energy are spent, leading to decreased productivity. Moreover, in most meetings and conferences, the participants are not equipped with all the information they might need to function effectively. In other words, they are dislocated from their ideal work environment. Furthermore, some meetings are totally unstructured and free-form, which diminishes their effectiveness. And many of these meetings have no effective mechanism for archiving all of the events that occur for future use.

The solution to these problems is to use existing computer and communications technology to overcome the distance barrier. The use of technology cannot only cut travel costs and time, it can also increase productivity by enabling individuals to, in a manner of speaking, bring their offices to their meetings.

This section describes the technologies available to support meetings involving distributed members of a virtual team.

12.2.1 Technology Overview

Several computer-based services are now deployed to support group communication. These services can be classified according to time and distance (Table 12.1).

Electronic Messaging

The simplest group communication service is electronic mail (e-mail), now widely used in industry, government research labs, and universities, and available to the general public through long distance carriers. Electronic mail is a useful facility for keeping members of a team in contact during a project. Minor structuring of the messages can provide a convenient way of performing daily work: disseminating task assignments, receiving notices of various kinds, and requesting information. A slight enhancement of electronic mail is the electronic bulletin board, which can serve as a discussion forum for recording and gathering views, ideas, analyses, and other information. An electronic bulletin board not only enables the rapid development of ideas and consensus, but it also generates an automatic corporate memory. A further enhancement indexes the discussion messages so that anyone who wishes to benefit from earlier knowledge can rapidly retrieve the archives.

Recent products, such as Lotus Notes, carry this structuring much further in two respects: first, in generating structured databases to serve the messages belonging to different categories (for example, engineering change notices, task assignments, customer complaints, meeting announcements, etc.) using organization-specific indexes; second, by making it possible for messages to contain attached graphic files produced by any application so that the recipient can view them as long as the application can be executed on his or her computer.

Computer-Supported Meetings

Computer-supported meetings come in several varieties.

Xerox Colab started with the idea of holding a problem-solving meeting around a "chalkboard" with several participants. They constructed a meeting room in which each participant had a computer in a connected network, and everyone could view and manipulate the contents of an electronic chalkboard on the computer screen via a "what you see is what I see" (WYSIWIS) chalkboard. The participants can also converse face to face. At the front of the room is a large electronic chalkboard—a

Table 12.1 Communication Services

	Same Time	Different Time
Same place	Computer support for face-to-face meetings	E-mail, computer supported asynchronous meetings
Different place	Desktop multimedia conferencing from the workplace, computer supported meetings	E-mail, computer supported asynchronous meetings

larger-scale replica of what is displayed on each screen—and a podium from which a speaker can manipulate that chalkboard directly.

The chalkboard provides a shared memory, enabling meetings to be focused and allowing direct and simultaneous participation. Later, Colab evolved in two directions.

The first direction involves a structured and recorded meeting. The meeting is organized into distinct phases—brainstorming for ideas, organizing by categorizing and ordering the ideas, and finally, evaluating the ideas and agreeing upon conclusions. The basic tool is a word processor with an outlining capability and a list manager. Once again, the participants operate from their own workstations. The team can determine whether the sessions are done synchronously with all the participants simultaneously present, at different times, or as soon as possible. With this kind of meeting facility, the entire group advances to the next stage only when the previous stage is complete.

This structured decision support capability is carried much further in some recent commercial software, compressing the time taken to arrive at decisions, particularly in the synchronous mode of operation, by a factor of 10! Group Systems V (from Ventana Corporation) is a Group Decision Support System (GDSS) allowing synchronous or asynchronous meetings to take place with some structured phases of decision-making; that is electronic brainstorming. Other options include Categorizer, Voting (seven kinds), Topic Commenter, Group Dictionary, Alternative Evaluation (multiple criteria voting), Policy Formation (group writing to devise a short mission or policy), Idea Organization (powerful list-building and organization, e.g., nominal group technique), Group Outliner, Questionnaire (on-line, fill-in-the-blanks, survey tool), Stakeholder Identification, Group Writer (everybody works on different sections), and Group Matrix (two-way analysis of agreement between values for different criteria for different alternatives). VisionQuest (from Collaborative Tech Corp.) is much like Group Systems V and is in regular use at an electronic meeting room for rent at a Marriott Hotel in Washington, D.C.

The second direction Xerox Colab took involved the addition of an argumentation facility, the Argnoter. With this facility, someone proposes an idea, someone else then raises an argument for or against the idea, others ask questions regarding the proposal, and yet others raise issues that come up in the consideration of the proposal. This sequence of argumentation is structured and made commonly visible to all the participants, whether or not they are currently signed on to the discussion or arrive later and wish to take part.

Many incarnations of Argnoter now exist. IBIS (Issue Based Information System) and gIBIS (Graphical IBIS) are two examples. CM/1, a commercialization of gIBIS designed at the Microelectronics Computer Corporation (MCC), is a groupware system for qualitative decision-making support, shared issue exploration, decision mapping by a group, and documentation of decision rationale. The anticipated results include organizational learning, better decision-making, and greatly enhanced productivity for collaborative work groups.

Desktop Conferencing

Another type of tool for virtual meetings is the *desktop conferencing system.* A desktop conferencing system consists of hardware and software that enable real-time, full-motion video and real-time audio conferencing. Some systems also enable application sharing and the archival of meeting minutes.

Desktop conferencing systems provide a significant advantage to professionals who need to frequently consult and cooperate with team members at other sites. Users can conveniently and effectively communicate in face-to-face meetings because they can see each other, notice each other's facial expressions, hear each other's voices clearly, and use whiteboards and other media to draw pictures, take notes, and point to items on the screen.

There are currently a number of desktop conferencing tools available commercially, including Intel's ProShare and CU-SeeMe (Cornell University and White Pine Software). Research prototypes include the Meeting on the Network (MONET) system at the Concurrent Engineering Research Center, West Virginia University.

Application Sharing

Application-sharing technology makes the information displayed on one computer simultaneously available on multiple computers. This is a very powerful technology for collaborative work and it has innumerable applications. This technology has been used to develop group editing tools, whereby a number of people can jointly work on developing a document. It can also allow people to present their data, viewgraphs, spreadsheets, design documents, and other materials to other people, all from their own workstations. It also addresses how multiple participants can interact with an application program (such as a finite element modeler), make changes, and see the effects of the changes. Example of application-sharing tools include the COMIX system (West Virginia University), XTV (Old Dominion University) and Shared-X (Hewlett-Packard).

Conferencing and application-sharing technology are rapidly maturing and show great potential in supporting collaboration over the network. Flexible support for latecomers to such technology-assisted meetings still raises some hurdles. Some interesting solutions are suggested by Abdel-Wahab

of Old Dominion University. For instance, how can such a person be rapidly briefed on what has transpired up to that point? There is also the issue of managing multiple applications simultaneously. For instance, in a meeting that involves marketing, design, and customer support, marketing may want to share a document and design the output from a CAD tool, and customer support may want to open a spreadsheet or database with failure rate information. Finally, the most significant challenge in deploying this technology is how to deal with the heterogeneity in hardware and software. Building tools that work on different hardware platforms and can cooperate with the software that exists on all of these platforms is still a difficult problem.

Audio Technology

Advances in the field of digital audio are opening new doors in improving productivity in our group work environments. From one's desktop computer, it is now possible to participate in a conference call, send and receive voice mail, annotate documents with voice clips, and give remote viewgraph presentations. Advances in voice synthesis have resulted in more realistic synthetic speech. There are some programs (e.g., at MIT's Media Lab) that can even provide the speech inflections associated with laughter and other emotions. Advances in speech recognition will one day make possible the conversion of voice annotations and speech to text. That day is not far off!

Some of the remaining challenges include managing audio quality over the network, delay and jitter control, and satisfying the hard real-time constraints posed by the nature of audio data.

Video Technology

Rapid advances in video and compression hardware technologies are making it feasible to develop and deploy multimedia applications. These include video conferencing over a computer network, support for multimedia mail, and use of this technology for a variety of other collaborative work.

Transmission Technology

One method for transmitting data over networks efficiently to a large number of computers is *multicasting*. Like radio and television broadcasts, computers can tune to specific frequencies to intercept messages destined for multiple host machines located worldwide. Ideally, only a single message is needed to contact all host machines. For unreliable delivery, the sender can simply send the message continuously, as in the case of audio and video data. For reliable delivery, the sender can request positive acknowledgments from a specific number of recipients.

Multicasting is currently being used experimentally to send audio, video, and shared data over *wide area networks* (WANs). The experiment, known as the *Multicast Backbone* (MBONE), has been using multicasting successfully since 1993 via Level 2 IP (Internet protocol) packets. The bandwidth required is at least T1 for a limited number of conferences. The same software and protocols, however, should be usable given larger-capacity networks (i.e., T3). Indeed, many MBONE sites (Xerox, Bellcore, Lawrence Berkeley Laboratory) are currently running on an experimental Gigabit network known as Xunet. The IP protocols are also evolving and the future versions, such as IP.v6, have better support for multicasting.

Asynchronous Transfer Mode (ATM) technology is also evolving rapidly and is inherently better suited for the high-bandwidth and real-time needs of high-quality conferencing and video-on-demand applications.

On the other end of the bandwidth spectrum are mobile and wireless links.

12.3 COORDINATION SERVICES

Traditionally, task coordination has been largely a human process. With the significant growth in the employment of multidisciplinary tiger teams, however, computer support is critical for group decision-making and negotiation, especially over a geographically dispersed network. Particular features of task coordination systems include common visibility of activities and data, planning and scheduling of activities, change notification, and constraint management across multiple perspectives.

Some systems, such as bulletin boards and electronic mail, provide an initial underpinning to support group working, but are very limited and informal. Fundamentally, they only allow for the exchange of messages, although they are being adapted to support brainstorming and group discussions. However, they do not support structured decision group working.

The team structure must be expressible in computer structures which mirror the organization of the project into a number of teams spanning many functional areas and ultimately many organizations. This imposes a substantial requirement that a project-coordination intelligence be pervasive in the network, so that wherever a person is located, that person can be deputed to belong to several teams at once. The team's membership profiles, constraints, common workspace, and tasks thus become visible, making it possible for a person to belong to any project, serve any role, and participate in all the team interactions at once, without leaving the workstation.

Coordination theory and technology are topics of high interest in current and recent research activities.

As part of the DARPA Initiative in Concurrent Engineering at the Concurrent Engineering Research Center, a system known as the Project Coordination Board was developed to support coordination of product development activities.

Coordination is being considered as part of the Computer-Supported Cooperative Work (CSCW) research efforts. The National Science Foundation has launched the Coordination Theory and Collaboration Technology Initiative under the Computer and Information Science and Engineering Directorate (CISE).

The Center for Coordination Science at the Massachusetts Institute of Technology is an interdisciplinary team studying new ways to organize human activity and developing new technologies to help people work together more effectively. In their view, coordination technology will provide benefits to humanity equivalent to those provided by the economies of production and transportation during the Industrial Revolution.

The technical approach of the Coordination Theory and Technology Project at MCC uses distributed systems technology to support the flexible automation necessary for coordinating people, tasks, and resources involved in organizational activities.

12.3.1 Technology Overview

Particular features of task-coordination systems include common visibility of activities and data, planning and scheduling of activities, change notification, and constraint management across multiple perspectives.

Common Visibility and Change Notification

Concurrent work involving many functional areas must be coordinated via a common workspace in which the actual work of product developers is made visible to assure structured group working. Conceptually, the common workspace is equivalent to the meeting table around which product developers gather to discuss and reach consensus in traditional engineering environments.

The common workspace must provide constant visibility of a unified cross-functional product model that provides directives for the information needed by product developers. Product developers can view and access components of the product structure within each domain of specialization. The common workspace, through nodes in the product structure, provides access to design information required during the product development cycle. It is also through iteration with this product model structure that product developers assert their design decisions onto the common workspace: a product developer locates some information required for analysis, executes a tool, and obtains the results by selecting appropriate nodes in the product structure. In principle, the common workspace provides immediate access to the information required by product developers to do their work. It also allows product developers to share the results of their work with their peer team members.

Product developers are concerned about how design decisions made by others influence their work. The product-development effort is driven, in part, by the existence of customer requirements, rules of design, policies, constitutive equations of engineering relating variables in different domains, and so on. They help guide and shape the product-development process, but they also raise conflicts and inconsistencies across perspectives. The common workspace must provide product developers with visibility of conflicts and inconsistencies across perspectives that affect their work. This aspect of visibility is supported by functionality that manages all types of dependencies, relationships, and constraints that exist between components of the product model. Notification mechanisms are implanted to support visibility of design decisions as they affect work of the virtual team members. This is key in assuring that everyone affected by a design decision participates in the final decision-making process.

Visibility of work concerns visibility of the activities performed by the group. Management of the relationship between the product and the process models must result in a dynamic approach to process management in which tasks are planned, scheduled, and monitored over the computer network. The common workspace is the place where the network of activities becomes visible. Product developers use the common workspace to view and respond to "work units" assigned to them by a project leader during the product development life cycle.

Finally, being the medium for interaction among product developers, the common workspace is also the place where consensus about conflicting design decisions is reached by product developers. This negotiation framework in the common workspace allows for the resolution of conflicts while exploiting trade-off analysis information to assure that the best design alternative is selected from the various design decisions posted into the common workspace. Functionality to support negotiation to reach consensus has two requirements. First, the framework for negotiation must provide means for human interaction across the computer network. This requirement was covered above in Section 12.2. Second, the framework for negotiation must exploit information available from the system to facilitate trade-off and multiobjective analysis in the decision-making process.

Managing Workflow

Coordination of the virtual team involves management of the ongoing concurrent work in many functional areas. It involves managing (over the computer network) the workflow of activities performed by the virtual team members.

Existing project-management packages are all oriented to repetitive and completely foreseen sequences of tasks from beginning to end. A concurrent engineering approach to product development calls for exploration, opportunistic contributions, and joint planning of work. Therefore, dynamic workflow management is required to support the planning, scheduling, and monitoring of tasks.

In principle, workflow management should extend the meaning of "manufacturing process" to all the processes occurring in the product development cycle. Also, workflow management must provide for the management of the critical relationship between the product model and the process model.

Workflow management involves the ability to reuse process models from previous projects. Each process or activity must be linked to the part or sub-part that is the focus of the activity, in order to provide for management of the relationship between the product model and the process model. Process models must be refined into atomic activities woven into a network of task scheduling units, providing for the initialization, dissemination, retrieval, monitoring, and overall management of these task units over the computer network. It means that team members receive electronic work orders via the computer screen and respond to them appropriately. This is core functionality to provide for dynamic management of the workflow in a CE environment

Tracking Design Progress

One aspect of assessing progress in a design is the quality measure attached to the results of the tasks. If assistance to track the results against product performance goals is not present, task leaders will be able to see much activity, but they will be unable to determine whether the activity is converging to the customer-desired product. A system to evaluate product performance metrics whenever the leader desires or when sufficient data are available would lessen the burden of tracking progress.

Assessment involves the ability to judge the quality of the evolving design based on initial specifications such as customer requirements, safety regulations, and standards. Evaluating these criteria requires that they be known, and this information can come from the standards and guidelines adopted and constraints created by specific customer requirements. Performance metrics, used to measure the performance of various components, also offer assessment capabilities. This can involve optimization of parameter values and requires the ability to specify the desired criteria.

Another area of assessment involves such "ility" components as manufacturability and maintainability, which are not explicit aspects of customer requirements but are determined by engineering domains and are an integral part of the design. Finally, product development performance measurements, such as adherence to the task schedule, prove a key area of assessment. These assessment capabilities partly depend on the availability of domain specific tools to evaluate performance requirements, especially with "ility" components and performance metrics.

The key requirement for monitoring the progress of a design is passing the "ACID" test: the ability to ACCESS information about the current design and past, similar designs; the ability to CHOOSE the parameters of the design to monitor and the ability to choose analysis routines to further analyze the information; the ability to INTERPRET the information that are collected; and the ability to DISPLAY the information gathered in an intuitive fashion.

Assessments can be valuable only with accurate, complete information. Therefore, to monitor the progress of the design, any source of information that can contribute to the assessment must be accessible. This includes information about the current design as well as past designs of a similar nature. Often, past designs can lend themselves to comparisons of current design situations.

Access stands not only for access to information, but also for access to analysis routines. These routines can provide more than just a filtered view of the data. Quality evaluation methods can provide valuable insight into the current state of the design based on the intended goals.

Once access is provided to the information and the analysis tools, users must be able to choose what information to monitor and which routines to invoke to analyze the data. Choosing the information to monitor implies determination of the key parameters crucial to the design. Techniques that can aid a designer in choosing the key parameters include Quality Function Deployment (QFD) and constraint management.

Constraint Management

QFD indicates customer requirements; by weighting those requirements for importance, the key ones can be determined. Through successive iterations of QFD, these can be translated into lower-level constraints on product parameters which can be maintained by a constraint manager. Through this translation, low-level product parameters can be monitored and their values propagated to determine

their effect on customer requirements. In this manner, the progress of the design can be determined based on customer requirements.

The constraints maintained for a design come from a variety of sources—not just customer requirements. In fact, one method of tracking design progress (and design "goodness") is by monitoring constraint satisfaction. Through the constraints, allowable/optimal values for parameters can be determined. Therefore, by using both QFD and constraint management, a designer can determine the key parameters to monitor, and, in some cases, the measurements to determine the direction of progress made on those parameters.

Once the user has determined the key parameters to monitor and indicated the tools and methods that should be invoked to evaluate those parameters, the collection and interpretation of those parameters should occur automatically. As sufficient real-time data are collected for the methods to interpret the data, they should be invoked.

The information that is being monitored should be displayed in a manner intuitive to the decision-making process. That is, the user should see the results in a natural manner without the need to sift through the reports/information the tool generates. To accomplish this, the tool should have access to graphical (plots, charts, histograms, etc.) and textual (tables, text, etc.) libraries to compose reports in a variety of manners. In addition, these results should be displayed in a timely manner. Reports concerning the assessments should be available as needed.

Another feature of display is the ability to "navigate" through the results to view aspects of the monitored design to the required level of detail. That is, users should see a high-level report of the design progress, but should be able to view aspects of the monitored information in sufficient detail to make quality decisions about the design.

As an example, consider an assessment which monitors constraint activity, collecting all of the constraint violations. At the end of the week the Project Leader (PL) notices in the report provided that a constraint on the blade's strength has been violated ten times whereas no other constraint has been violated more than twice. The PL may want to know which designers have violated that constraint and what tasks caused the constraint violations. The desired information should be available through the report through requesting more information about the constraint's history.

Once this information is displayed, the PL notices that the aerodynamics engineer has violated the constraint seven of the ten times when attempting to determine the optimal flow path for the blade. This information may cause the PL to request a meeting between the engineer who set the constraint (in this case, the mechanical engineer) and the aerodynamics engineer to resolve the problem so that the strength constraint can be satisfied and the flow path can be optimized.

12.4 INFORMATION SHARING

In addition to communicating with each other and coordinating teamwork, team members must have ready access to the information necessary and appropriate for their tasks. Because corporate information exists in a variety of computer information repositories, such as databases, documents, drawings, and data files, it is imperative that an information-sharing system be utilized to provide a single interface to these sources of information.

The information must also be indexed and accessible via a type of electronic card catalog. This is a recognized need for which an ANSI standard solution exists—Information Resource Dictionary System (IRDS). An IRDS contains metadata—data describing other data in the organization. It consists of a dictionary which identifies various information resources and describes their logical structures, and a directory that describes the location and protocol by which such information may be accessed.

Some of the major criteria for information-sharing technologies include:

- Ease of use
- Performance objectives such as response times
- Integration with existing/fielded systems/environments
- Cost-effectiveness, economic viability, scalability: some of the major factors relating to scalability are the number and types of information, the number (total and simultaneous) of users (providers), and the geographic distribution of the elements of the system
- Ability to address security, integrity, consistency, and proprietary and intellectual property rights concerns
- Need for legislative mandate—some of the technological solutions such as digital signatures have not been put to test in a court of law and may require legislative support for implementation

12.4.1 Technology Overview

A variety of techniques have been developed to enable wide-area access to information. Increasingly, organizations are beginning to rely on the Internet-based information technologies to communicate

and share information. Particularly effective are tools that provide access to enterprise databases over the Internet, especially over the emerging World Wide Web.

Current and Emerging Approaches and Systems

Organizations have typically employed centralized information servers based on large mainframe computers. These solutions have migrated to smaller machines that effectively employ the client–server approach to provide information to distributed users.

Organizations that do not have to deal with a significant number of fielded/deployed systems, organizations that have complete control over their entire operations, and small businesses can adopt this strategy.

Centralized solutions are also available for unstructured information, and document-imaging solutions are available commercially that can store and serve gigabits of information. Examples of research systems include the object-oriented database management system—OMEGA, the multimedia object presentation manager MINOS, and the multimedia office server MULTOS.

Distributed DBMS solutions have now been available commercially for several years. Organizations that can dictate a homogenous solution have fielded such ways of accessing information successfully. For instance, a company can dictate that their entire distributed databases should be based on Oracle™ or Sybase™ products. Using commercial off-the-shelf (COTS) products does involve a significant amount of customization and front-end application work, however.

Security in current-day systems is handled primarily through private networks. The use of such private networks will eventually become obsolete as the Internet becomes more prevalent and cost-effective. Solutions for addressing security concerns on the Internet are beginning to appear in the marketplace.

In the context of integrating multiple databases and information repositories, several approaches have emerged over the last decade. The *federated* or *multidatabase* approaches (see Ref. 2) present the user with a collection of local schemas along with tools for information sharing among the databases. In this case, the user integrates only the necessary portions of the databases. There are several advantages to this approach, such as increased security, easier maintenance, and the ability to deal with inconsistent databases. Such an approach is suitable when the different databases in the federation contain similar data, but not when a wide variety of information repositories (not all of which are databases) need to be integrated. In such cases, the user needs to be guided through the information available via a model.

The World Wide Web

The World Wide Web (WWW or Web) is accessible through commercial on-line services and through popular Web browsers, such as Netscape and Mosaic (available in the public domain). Web browsers provide a point-and-click metaphor for accessing a hyperlinked collection of documents written using the Hypertext Markup Language (HTML) and available on the Internet. HTML is one of the family of languages that conform to the Standard Generalized Markup Language (SGML), an international standard for specifying neutral-format documents. HTML documents are served by servers that adhere to the Hypertext Transfer Protocol (HTTP), which was designed to efficiently support multiple independent requests for documents. These servers do not maintain any state information; each request for a document is an independent transaction. To support the dynamic creation of HTML documents, the HTTP servers support a Common Gateway Interface (CGI). Typically, the HTTP servers invoke CGI programs—frequently called CGI scripts—when requested to serve specific documents. CGI scripts can be written to provide access to, and present information coming from, a variety of sources.

CORBA Standards

The Common Object Request Broker Architecture (CORBA) standard was developed out of the need for interoperable solutions that work across multiple hardware and software platforms. This standard is promoted by the Object Management Group (OMG), whose membership includes over 500 hardware and software vendors. The CORBA architecture, and particularly the Version 2.0 standard, promotes interoperability to a hitherto unprecedented level: it promotes independence in hardware architecture, language, and location. For instance, by complying with CORBA, software services can be written in any language (e.g., C, C++, Scheme, or even Fortran), run on any machine (e.g., Sparc, Silicon Graphics, Macintosh, PC), use any operating system (e.g., Windows NT, Unix), and be accessed by client software, which could be in turn be written in any language. Of course, the success of this standard—and it already has achieved a fair measure of success—depends upon the availability of support for developing Object Request Broker (ORB) services in the respective languages and operating environments. This support is currently commercially available for all of the major hardware platforms.

The CORBA standard defines services that are based on object-orientation principles and requires the definition of services using the Interface Definition Language (IDL).

The major components of CORBA are:

Object Request Broker (ORB) core and interface

Interface Definition Language (IDL)

Dynamic Invocation Interface (DII)

Object Adapters

The architecture defined by the CORBA standard is shown in Fig. 12.1.

Web*

Web* is a software that is part of the Information Sharing System in West Virginia University. The goal of the Information Sharing System (ISS) is to provide the means for an organization to effectively disseminate information, thus enabling effective work in collaborative endeavors. Because corporate information exists in a variety of computer information repositories, such as databases, documents, drawings, and data files, it is imperative that these sources be integrated with an information-sharing system to enable wider use. CERC's Web* (pronounced "WebStar") software, currently released into the public domain, can be used to integrate multiple information sources. Web* allows the exploitation of the World Wide Web and the CORBA environment.

The Web* software allows the linking of any information source to a World Wide Web client, such as Mosaic or Netscape, by allowing a person to specify HTML or other ASCII-based templates which are dynamically filled in upon the request of a user. The templates contain embedded TCL commands, which are interpreted and can be used to retrieve and dynamically fill the templates with information. Web* comes with interfaces to call CORBA-compliant services. One of the key features of Web* is that it provides mechanisms to deal with the stateless nature of the HTTP protocol. The architecture of Web* is shown in Fig. 12.2.

Scripting Languages

Scripting languages have long been used for routine chores that do not need the full power of a programming language such as C or C++. Some examples of scripting languages include shell scripts in Unix, the Practical Extraction and Report Language (Perl), and the Tool Command Language (Tcl). One characteristic that most scripting languages have in common is that they are usually interpreted. Because of this fact, interpreted programming languages such as Basic, Common Lisp, or Scheme can also be used as scripting languages.

Mediators

In future information systems, mediators and mediator-like systems will play an important role in providing enterprise-wide information sharing. This emerging subfield of computer science, originally pioneered by Distributed Artificial Intelligence (DAI) researchers, advocates the notion of intelligent

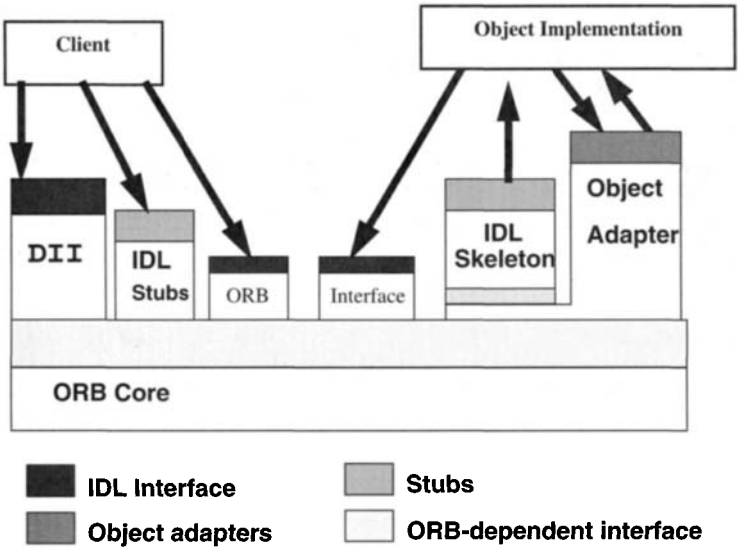

Fig. 12.1 CORBA architecture.

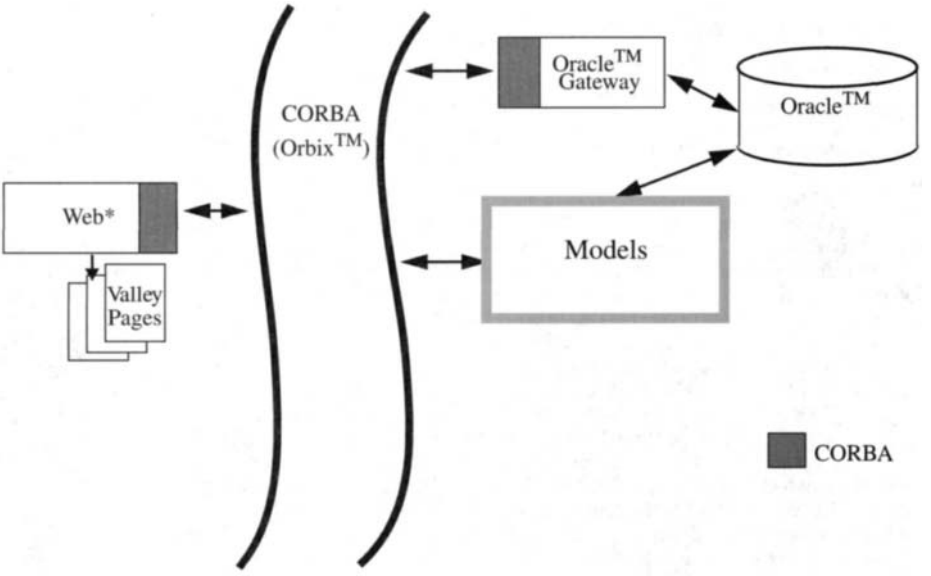

Fig. 12.2 Web* architecture.

computing agents that cooperate over a network to accomplish a set of objectives. Each agent is independent and maintains its own view of the world (in DAI terminology, a model of itself and others). Recently, agent-based architectures have come to the forefront of enterprise information systems and are currently referred to as *mediators*, based on the term used by Wiederhold.[3] The notion of mediators in the context of developing an information system hinges on the premise that intelligent assistants can be built that

- Filter and forward information
- Maintain a model of what information is available and in what format
- Facilitate the specification of user profiles on what information they are interested in and what they should be notified about
- Anticipate the needs of the user it is supporting
- Retrieve and provide the information needed

12.4.2 Related Research

Carnot

The Carnot project[4] at MCC uses a large body of knowledge or ontology that is available from the Cyc knowledge base (see Refs. 5 and 6), which has a rich representational structure. The bulk of the knowledge in Cyc is common sense facts about the world (such as how tall humans are) and does not include enterprise-specific information. Such information is added to Cyc, thus expanding Cyc's ontology. The Carnot project is also aimed at integrating multiple databases.

ISS

The Information Sharing System (ISS)[7] at the CERC was designed to provide the user with a system that is integrated with representative information repositories along with methodologies that enable the implementation in a phased manner. The major themes in this project include:

- Use of a model of enterprise information
- Uniform way to access information from heterogeneous information repositories including multimedia repositories and legacy database systems
- Use of a commercial off-the-shelf object-oriented database systems and relational database systems
- Use of model-based and case-based retrieval techniques for enterprise integration

12.5 CORPORATE HISTORY MANAGEMENT SERVICES

In any product development environment, there is a need to electronically capture the design intent and the evolution of the product from conceptual design to retirement. Corporate history is useful for future product design as well as for documenting existing products. Indexing, linking, and storing various types of documents (design, manufacturing, specifications, etc.) and archiving decisions reached in meetings among team members are some of the features of corporate history management systems.

Traditionally, the history of a product has been captured in notebooks and the minds of the designers and other team members involved in product development at various stages. They documented the tasks and activities and the decisions taken at each step. Often, such information is in the form of the daily notes of the designer, memos, pictures or a piece of conversation. The following are some of the reasons for capturing product history.

- In order to be able to produce good-quality products efficiently, a virtual team member must be able to look up the history of related products. This way, the team member gets to know the good decisions, the rationale behind the decisions, the lessons learned in the earlier ventures, why a particular decision was made in the context of several alternatives, etc. Access to such information is vital for developing good-quality products in a competitive world.
- Product history is vital in improving product quality. For example, if the designer of a new engine of a car had the maintenance records on the previous versions of the car readily available, then he or she would be able to evaluate how the previous designs of the engine fared and use this information in the new design.
- Product history is important even in the context of a single product. A product goes through an entire life cycle from conceptual design to maintenance and retirement. There will be several versions of the product during its lifetime. Unless the rationale behind each decision is documented and understood, team members with various perspectives will not be able to respond to these decisions. When problems arise, it becomes especially important to know why a certain design decision was made. Knowing the rationale helps in altering the decisions if need be.
- Product history is also important, for legal reasons. Even beyond the useful life of a product, a corporation needs to keep a record of the events in the product development, such as the studies conducted and decisions made. This is to provide some immunity against liability lawsuits that may come much later. Documented product history will also be useful in patent law in proving the originality and precedence of a particular work.

With the above motivation for product history, let us examine how it is currently being done, and why we want a computer-supported tool for doing this. Even though computers have become commonplace in every office, they are still not actively being used for capturing design history. The current mechanism for capturing decisions is the engineering notebook, into which an engineer writes down all the design decisions and sometimes pastes additional material. Below are some of the virtues of capturing design history electronically.

- Engineering notebooks cannot handle multimedia information; voice and video especially cannot be archived effectively. This is a significant drawback of the notebooks, because voice annotations and graphical illustrations that support design decisions are never recorded formally and are lost after a meeting has ended. With the development of multimedia editors and high-resolution display workstations, however, capturing graphics and video is becoming easy.
- Notebooks cannot be shared and accessed easily, particularly in a geographically scattered organization. Current computer networks are making information easily accessible over long distances.
- Notebooks cannot aid in retrieving items of interest to a team member. Computer-supported tools can aid in retrieving and traversing information. Such searching capabilities are one of the key advantages of computer tools.

12.5.1 Issues in Product History

Process Modeling

Understanding the product-development process is the first step in building a tool to capture product history. A design progresses through several stages before it is developed into a product, and the design history must provide support for capturing all the significant events in this process. Therefore, it is important to understand what the customer (virtual team member) needs to support his work in its various stages. In this regard, providing the functionality of a design notebook is only a first step.

Books have several limitations when it comes to retrieval, and one cannot pose intelligent queries to them. Some level of organization can be achieved through the use of tables of contents, indices, and such, but the human still has to do most of the work. This could be cumbersome if one has to track through a whole pile of books, even if they are all physically co-located. A *to be* scenario must be created based on what the team members want given the state of the art of the computing technology.

Representation

A rich representation structure is necessary for capturing the complexity of product history. This includes representations for product history as well as the data types in product history. One of the key aspects of a product history data system is heterogeneity, which means that it is capable of handling a variety of data formats.

Representing product history involves the use of a variety of data formats. Plain text, text with special fonts, 2D and 3D drawings, IGES files, schemas in EXPRESS, program code, solid models, graphic images, and voice are all examples. In addition, some enterprises have document formats and forms that are specific to the enterprise, such as the Design Study Summary (DSS) and Item of Information (IOI) used by General Electric.

In addition to the above data formats, a product history tool should be capable of representing various conceptual entities such as a design, design rationale, project, team, experimental results, analytic results, constraints, and so on. One of the important notions to be captured is the rationale behind decisions in the course of product development. Design rationale is an explanation of why an artifact is designed the way it is. A product can be thought of as a physical realization of a set of goals. The product represents an optimization of the goals subject to resource and other constraints. For example, a manufacturer might be making a computer with the goals of large memory, high CPU performance, fast disk access, high-resolution monitor, and so on. However, all of these have to be achieved subject to cost constraints and application demands. Thus, within a spectrum of choices, the final product represents the best alternative. Product history can be decomposed along several axes, with each axis providing a view. The above decomposition of a product into a set of goals is a *structural* view. Thus, the design of a CAD workstation involves a high-resolution monitor, a processor board, frame buffers, graphics accelerators, track ball, and so on. Since products evolve, they go from one version to the next. For example, Version 2 of a chip design could point to Version 3 of the chip, and in order for one to fully appreciate the design decisions of Version 3 one must examine the previous version of the design. Thus, there is a *temporal* view: a slice through the time line. There is a third view, the *logical* or rational view, which shows the supporting facts behind each decision and, very often, a causal chain. For example, we might want to provide a high-resolution monitor because we want to display 3D solids accurately, which is in turn necessary because we want to develop a CAD workstation for mechanical designers, and mechanical designers deal with 3D solids. Any representation scheme must support these three axes of product history.

In fact, the logical view could also present several issues[8,9] that are relevant in a particular design. An issue could be argued by taking one of several positions, and each position in turn has an argument, or a sequence of logical assertions, that support that position.

Retrieval

A product's history data collection can become very large, and, unless efficient ways of retrieving it are provided, it will be quite useless. For this reason, computer-based tools have a distinct advantage over paper. When a team has a large collection of notebooks containing designs, it is not easy to search them for a specific item of interest. This is particularly true if only some constraints on the item are known. For example, a user might want to retrieve the design of a turbine blade for which John or Pete was in the design team and that was drawn in the late 1980s. Product history could be organized by time, projects, teams, leaders, events, and so on, and retrieval by any of these indices is required. In addition to standard index-based retrieval, users need querying capabilities as well as associative retrieval.

A design will have a number of attributes, such as the date, designer, team members, project lead, key words, and rationale. It should be possible to retrieve a design based on any one of them. In addition, if old designs are to be reused and refitted by people who are not aware of the specific design attributes, associative retrieval techniques must be provided. For this purpose, case-based reasoning[10] seems to offer a lot of promise. Case-based reasoning is a method of reasoning from analogies—that is, reasoning from old cases or experiences in an effort to solve problems, critique solutions, explain anomalous situations, or interpret situations. Normally, people tend to solve problems by analogy, but they are not good at retrieving the relevant cases, especially when dealing with a large number of them. In particular, medical-expertise and legal education are case-oriented. Since computers are good at searching large databases, augmenting their capabilities with case-based reasoning will make the relevant cases available to a human. The case-based approach is not intended to supplant humans in decision-making. Instead, it supports human decision-making by providing the cases based on analogies and cues rather than specific indices. One of the key issues here is to figure the set of abstract cases that a specific case typifies; otherwise, an archived case will be applicable

only when the new case matches it very closely. The design rationale can serve as a way of capturing the abstract cases. Once the cases relevant to a particular situation have been retrieved, the user can utilize conflict-resolution strategies to find the most appropriate case. Adapting the case to the current situation is the task of the human.

Navigation

Retrieval can enable one to access only one particular item. However, once the user has retrieved an item, he or she often needs to examine a related or consequential item. For example, while examining one version of a design, a team member might want to examine the next version or perhaps the components of the design. While examining a particular design decision, he or she might want to look at the alternative designs that were considered but not chosen. In a book, the table of contents, lists of figures and tables, glossary, index, and bibliography provide ways to navigate through the book. The sequence of pages, chapters, sections, and so on, in a book provide a simple linear order of navigation. When it comes to navigating information stored on the computer, however, hyperlinks provide a way of navigating complex documents.

Hyperlinks can provide all the linear navigational capabilities available in a book, and much more. A link can include information about its type structure, which identifies different types of links. For example, *immediate predecessor* is a relation between designs, which is a special case of the *predecessor* relation. If a design is represented as a node, then other menu items associated with the node can take the user to various other nodes, depending on the relation chosen. Effective ways of visualizing these hyperlinks must be provided.

Creation and Update

In general, the product history is meant to be read-only. However, this applies mostly to past history; the current events need to be updated into the product history. This could be at the level of conceptual design or in the later stages of product development. In each of these cases, the user must be able to manipulate heterogeneous data formats—for example, cutting and pasting text, voice, graphics, and so on—freely into the document being created or modified.

Even with the increasing use of computer tools, there is a reason why the traditional notebook is still the medium of choice in design. For the creation of new designs, no other medium offers the flexibility and agility offered by a pad of paper. During the conceptual design stage, the designer must be able to give free reign to his/her thoughts and visions. Many of the current editors do not allow multiple media and a wide variety of data formats. More importantly, they constrain the user to a set of entities depending on whether they are *text mode, graphics mode*, and so on. Such a scheme can be quite constraining for a designer who needs an environment that can capture his/her thoughts. Ideally, the designer would like a sketchpad-like facility in which he/she can draw freehand and the computer will interpret the drawings, i.e., a tool that acts both as a computer and as a pencil. Currently, tools such as the PenPoint system by GO Corporation are beginning to address this problem.

User Interface

In addition to sketchpad-like interface, any tool for capturing product history needs to include a multimedia editor, which enables communication and editing of audio, video, text, and graphics. The user-interface must be programmable to accept a variety of documents, such as process diagrams, design reports, and memos. One of the important features needed is the ability to establish as well as visualize hyperlinks. Say the link from one version of a design to the next is called the *next-version* hyperlink, and the link from a component to its parent is called the *part-of*. The user needs to be able to visualize these hyperlinks and to distinguish between the two kinds. For example, the two links could be shown in different colors. Such capabilities are required so the user can visualize a complex document at various levels. For example, he or she might want to look at the rationale for a particular design, or at the successive versions of a design at a coarse grain. Thus, there should be ways for a user to visualize a design along specific axes, such as the structural and logical axes, and ways to visualize a design at a more detailed level.

Security

The product data history will be organized at several levels. Certain kinds of information may be personal to an individual team member, and certain data may be exchanged among all the members of a team or within a project. As people move in and out of various projects, their access must be enabled and disabled accordingly.

A concept that addresses the issues of logical partitioning as well as selective access is that of a notebook. A notebook need not be a continuous document such as a single word-processed document. It is, rather, a collection of separate documents linked together in assorted ways. An example of such is a *lab notebook*, which contains the versioned documents of a team member and his/her notes. This workspace belongs exclusively to the owner. A different kind of notebook, such as a Project/

Patent notebook, is a read-only document, to which team members may submit entries and also read but cannot alter.

The Environment of a VTM

A tool for representing product history is an essential part of the environment of a Virtual Team Member. It should not be an isolated tool but should be integrated as part of the CE environment. A CE environment provides a team member with a single *shell* or working environment; within this environment, he or she can move freely from one CE tool to another and can drag the constraints from one design and apply them in another context. He/she should be able to communicate to this tool from the other tools in the environment. For example, the product history tool must communicate with a tool for multimedia conferencing over the network. This means that the minutes of a multimedia conference could be part of the archived documents and could become part of the product history. Likewise, in the middle of a conference, a team member should be able to access information about the product history and be able to include it as part of a conference session.

User Acceptance and Validation

As with any computer tool, the final benefits of a product history tool depend on the acceptance and use of, and feedback from, its end users. The tool must not alter the existing practices and protocols of recording design history, but must support the same practices in electronic form. It must not require team members to learn and use a lot of computer jargon, thereby distracting them from their main tasks. The tool also must be programmable to produce specialized documents that are specific to certain corporations. This way, the knowledge is captured in the electronic form, enabling easy subsequent access, and at the same time the user can make the transition smoothly to the computer tools.

12.6 CONCLUSION

This chapter has provided an overview of the technology elements needed to support concurrent engineering teams. A number of commercial tools now on the marketplace address some of the needs and requirements presented in this chapter. The emergence of high-speed networks, the explosion in the adoption of World Wide Web technology, and the maturation of integration frameworks now make it possible to support the notion of a virtual team—a geographically distributed team of engineers.

REFERENCES

1. K. J. Cleetus, "Definition of Concurrent Engineering," in *CERC Technical Report Series*, CERC-TR-RN-92-003, Concurrent Engineering Research Center, West Virginia University, Morgantown WV, May 1992.
2. W. Litwin, L. Mark, and N. Roussopoulos, "Interoperability of Multiple Autonomous Databases," *ACM Computing Surveys* **22**, 267–293 (1990).
3. G. Wiederhold, "Mediators in the Architecture of Future Information Systems," *IEEE Computer* **25** (3), 50–62 (March 1992).
4. C. Collet, M. Huhns, and W.-M. Shen, "Resource Integration Using a Large Knowledge Base in Carnot," *IEEE Computer*, **24** (12), 55–62 (December 1991).
5. D. Lenat and R. V. Guha, *Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project*, Addison-Wesley, Reading, MH, 1990.
6. R. V. Guha and D. Lenat, "Cyc: A Midterm Report," *AI Magazine* **11** (3), 32 (Fall 1990).
7. V. Jagannathan et al., "Model-Based Information Access," in *Proceedings of the Second Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, IEEE Computer Press, Los Alamitos, CA, Summer 1993, pp. 198–212.
8. W. Kunz and H. Rittel, *Issues as Elements of Information Systems*, Technical Report Working Paper No. 131, Institute of Urban and Regional Development, University of California, Berkeley, 1970.
9. H. Rittel, *Apis: A Concept for an Argumentative Planning Information System*, Technical Report Working Paper No. 324, Institute of Urban and Regional Development, University of California, 1980.
10. E. Rich and K. Knight, "Common Sense," in *Artificial Intelligence*, R. R. Donnelley & Sons, 1991.