SHASTRA - An Architecture for Development of Collaborative Applications

Vinod Anupam Chandrajit Bajaj

Department of Computer Sciences
Purdue University
West Lafayette, IN 47907

Abstract

We address the issue of design of architectures and abstractions to implement multimedia scientific manipulation systems in a Concurrent Engineering setting, where experts in a cooperating group communicate and interact to solve problems. We propose a model for the integration of software tools into a multi-user distributed and collaborative environment on the multimedia desktop, and describe a prototype CSCW infrastructure which we have used to implement scientific problem solving tools. Finally, we briefly describe a prototype CE system built on this infrastructure. SHASTRA presents a unified prototype for some crucial enabling technologies for Concurrent Engineering - Multimedia Communication, Framework Integration, Coordination, and Enterprise Integration.

1 Overview

This section introduces SHASTRA in the context of related work. Section 2 describes the architecture and highlights the main features of the system. Section 3 introduces SHASTRA Multimedia Services and Section 4 briefly describes SHASTRA toolkits. An example Concurrent Engineering application, implemented in SHASTRA, is briefly presented in Section 5. Section 6 addresses issues and future direction.

1.1 The SHASTRA CE Environment

SHASTRA is a highly extensible, collaborative, distributed, geometric design and scientific manipulation environment. At its core is a powerful collaboration substrate – to support synchronous multi-user applications, and a distribution substrate – which emphasizes distributed problem solving.

SHASTRA presents a prototype for some crucial enabling technologies for Concurrent Engineering – Multimedia Communication, Framework Integration, Coordination, and Enterprise Integration in a synchronous setting. SHASTRA consists of a growing set of interoperable tools for geometric design networked into a highly extensible environment. It provides a unified framework for collaboration, session management, data sharing and multimedia communication along with a powerful numeric, symbolic and graphics substrate, enabling the rapid prototyping and development of efficient software tools for the creation, manipulation and visualization of multi-dimensional geometric data.

The SHASTRA environment consists of a group of interacting applications. Some applications are responsible for managing the collaborative environment (the Kernel applications and Session Managers), whereas others provide specific services (the Service Applications), while yet others provide scientific design and manipulation functionality (the SHASTRA Toolkits). Services and Tools register with the environment at startup providing information about what kind of services they offer (Directory), and how and where they can be contacted for those services (Location). The environment provides mechanisms to create remote instances of applications and connect to them in client-server mode (Distribution). In addition, the environment provides support for a variety of multi-user interactions spanning the range from master-slave electronic blackboarding to simultaneous multiple-user interaction (Collaboration). It provides mechanisms for starting and terminating collaborative sessions, and joining or leaving them.

Though the toolkits are independent processes with separate user interfaces, they share a common infrastructure of numeric, symbolic, and graphics algorithms. The toolkit processes connect to each other

0-8186-4082-0/93 $3.00

and communicate via the SHASTRA environment. SHASTRA provides these systems with connection management and data communication facilities enabling component systems to use facilities and operations provided by sibling systems, effectively integrating them into a large scientific manipulation system. It also provides them with a collaboration substrate to support cooperative and collaborative design effort.

1.2 Related Work

Our intention in SHASTRA is to provide a distributed heterogeneous environment for multimedia collaboration and to facilitate the development of collaborative applications for concurrent engineering by providing a rich substrate. Specifically, we focus on the scientific domain. DICE(MIT) [17] discusses issues for realizing frameworks to support collaborative engineering activities, and [8] presents an overview of integration technology from the perspective of concurrent engineering.

Groupware emphasizes on using the computer to facilitate human interaction for problem solving. Ellis *et al* present an overview of the state-of-the-art in [6]. Research into computer and video fusion to support collaborative work has resulted in TeamWorkStation [7] which uses video-overlaid shared drawing surfaces in addition to wired video and audio communication links. Rendezvous proposes a powerful architecture for multi-user applications [12], and the Touring Machine provides an infrastructure for development of applications that require multimedia communication. Collocation facilities have received a lot of interest of late *e.g.* MMConf [4], COMET, Rapport [1], Capture [10] etc. MONET [16] presents a powerful architecture that facilitates collocation and cooperation for virtual teams, and transparently lets multiple users cooperatively interact over a single-user X-based application using a simple turn-taking protocol. The system provides content independent sharing for collocation. Suite [5] proposes a technology for enabling CE. A powerful paradigm for modeling multimedia collaborations is presented in [13]. PCB proposes a general system to facilitate coordination of group work [9].

SHASTRA lets us build applications with shared drawing and viewing surfaces by supporting content dependent sharing for concurrent engineering – the applications are collaboration aware, and support synchronous multi-user manipulations of application-specific objects. This adds a new dimension to the kind of cooperation that can occur in collaborative problem solving, because it permits cooperative manipulation and browsing of objects in the context of applications that manipulate them. It supports cooperation in the design (problem-solving) phase, as well as in the analysis (review) phase.

2 SHASTRA – System Features

2.1 System Architecture

The design of SHASTRA is the embodiment of a simple idea – scientific manipulation toolkits can abstractly be thought of as objects that provide specific functionality. The objects exchange messages, automatically or under user command, to request operations of other objects. At the system level, SHASTRA specifies architectural guidelines and provides communication facilities to set up an integration framework that lets toolkits cooperate and exchange information to utilize the functionality they offer. At the application level, it provides collaboration and multimedia facilities allowing the development of applications in which users cooperate to solve problems. A synergistic union of these aspects lets us design sophisticated concurrent engineering environments.

2.1.1 The Integration Framework

SHASTRA addresses the issue of integration in a Concurrent Engineering setting by adopting a specific high-level Application Architecture. All systems designed to run in the SHASTRA environment (Fronts, Kernels and Session Managers) have certain features which make them amenable to inter-operation. A typical system has an application specific core – the Application Engine which implements all the functionality offered by the system as a tool or service. This part typically encapsulates the basic computational operations and services provided by the toolkit.

On top of the Engine is a Functional Interface Mapper which actually calls upon functionality embedded in the engine in response to requests from the the Graphical User Interface, ASCII Interface or the Network Interfaces. The GUI is application specific. The ASCII interface is a shell-like front end for the application. The Network Interfaces communicate with the outside world using the SHASTRA communication protocol [3]. The availability of multiple network interfaces supports a flexible regulation mechanism – toolkits can offer different functionality at different interfaces.

All communication between Fronts, Kernels and Session Managers occurs via their Network Interfaces. The SHASTRA architecture is depicted in Fig-

ure 1. Toolkits talk to the communication subsystem through an abstract interface, which multiplexes multiple simultaneous network connections. This architecture makes it easy for other systems to connect to a system via the network interface and request operations, synchronously or asynchronously – it supports interoperability for standardization in CE settings, and lets us build large applications modularly.

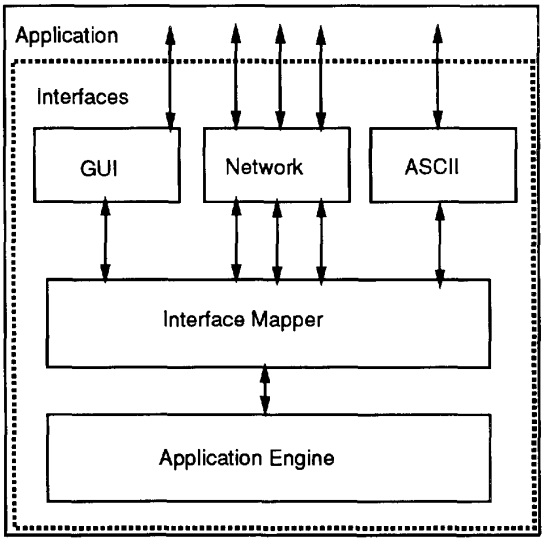

Figure 1: SHASTRA Application Architecture

The entire set of connected Network Interfaces of Kernels, Session Managers and Fronts implements the abstract SHASTRA layer (see Figure 2) which maintains the collaborative environment, provides automatic access to functionality of different systems, and provides facilities for initiating, terminating, joining, leaving and conducting collaborations. It provides transparent access for data sharing between different tools and services.

2.1.2 The Kernel

The SHASTRA kernel is responsible for maintenance of the distributed environment. It consists of a group of cooperating kernel processes. One kernel process runs at a well known port on every host that is active in the environment. The SHASTRA kernel spawns all service and toolkit processes on request from users or applications. A Directory facility lets users dynamically discover what services and applications are active in the environment at any time. A Location facility provides contact information about where all the services and toolkits are running in the environment, letting applications dynamically connect to each other to access toolkit functionality or to access services.

SHASTRA kernels maintain information about the state of the environment on the local host, and exchange information with all other kernels to obtain state of the distributed environment. They also maintain information about the collaborative sessions in the environment and their membership.

2.1.3 SHASTRA Tools

Application and service processes running in the SHASTRA environment, also called Fronts, are the tools in the system. Fronts exist on a per-user basis, and there is no restriction on the number of instances (besides operating system imposed limits). Any Front can access the SHASTRA environment to instantiate tools locally or on remote sites, and to terminate previously created tools. Fronts can also connect directly to each other to exchange data in client-server settings using the connection management facilities of SHASTRA. Users at Fronts can initiate and terminate collaborations, join or leave current collaborations and invite other users to join ongoing sessions via the Kernel process and the Session Managers.

2.1.4 Session Managers

A collaborative session in SHASTRA is started by a user through a Front. One instance of a Session Manager runs per collaborative session. A Session Manager is a specialized service in the SHASTRA environment. It maintains the collaboration and handles details of connection and session management. It is a repository of the objects in the collaboration, and keeps track of membership of the collaborative group. It communicates with the local SHASTRA Kernel to keep it up to date with current membership of the session. The session manager provides the broadcast facility needed for information exchange in multi-user synchronous conferencing. This is the core functionality of all session managers.

The application-specific part of the session manager has a constraint management subsystem which resolves conflicts that arise as a result of multi-user interaction, and therefore maintains mutual consistency of multiple operations. This part is a function of the collaborative application being constructed.

Every Front participating in the collaboration communicates directly with the session manager. It regis-

ters collaboration objects with the manager and transmits to it all operations it performs on these objects. The session manager performs constraint checks and relays requisite information to all other participants in the collaboration.

Every Front in a collaborative session performs certain operations on behalf of the session manager. It thus executes in two modes – local mode, when it responds to directives from the user interface and remote mode, when it responds to directives from remote systems (the session manager, in this case). A Front may be party to multiple collaborative sessions simultaneously. All such sessions are independent.

The session manager also provides a blackboarding substrate to collaborating Fronts. This is an asymmetric operation in which one Front is master and all other Fronts are slaves. Baton passing is used for floor control. The Front with the baton is the master who has total control over the session interaction as long as he has the baton. Identity of the master is broadcast to all collaborators. In the preemptive mode, a Front can take the baton away from the master. In the non-preemptive mode other participants can request the master to release the baton. Baton passing is performed in the Session Manager, Fronts communicate with the session manager to capture and relinquish the baton, in conjunction with the regulatory subsystem.

2.2 The Collaboration Infrastructure

The SHASTRA collaboration architecture uses a replicated computation model for the multiple user system – a copy of the application (the Front) runs at each site involved in the collaboration. Since most scientific manipulation tasks we address are graphics intensive, this gives a performance advantage over a centralized model. Application developers utilize the broadcast facility of SHASTRA to distribute input of low computation tasks and the output of high computation tasks to benefit from the distributed setting. The Session Manager for a collaborative session regulates only collaboration specific windows of the Front. This, coupled with the replicated model, permits easy separation of the private and public aspects of the Front at each site.

2.2.1 The Coordination Facility

A two tiered permissions based access regulation mechanism is used to structure a variety of multi-user interaction modes at run-time. It allows a high degree of tailorability and flexibility in SHASTRA's CSCW applications in the domain of interaction as well as data sharing and access control. Permissions are specifiable on a per-site as well as a per-object basis. A Site Permission defines the scope of collaboration operations available at each site in the session. An Object Permission specifies the operations permissible on each object in the collaboration. Permissions are maintained at the Session Manager which centrally coordinates the access regulation mechanism. The group leader (typically the session initiator) specifies permissions at the start of the collaboration. He can alter permissions dynamically. The permissions system is intended to be a regulatory mechanism, rather than a security mechanism, in a cooperative setting. The regulatory mechanism aims at being able to support a variety of interactions ranging from master-slave mode blackboarding to multiple site collaboration. The regulatory subsystem supports the following permissions.

1. Access – This regulates the view at a collaboration site. For a participant, it specifies whether or not he will observe any collaborative interaction. For an object it specifies whether or not it will appear as part of the view. E.g. it decides whether or not a site in an audio/video conference will receive sound bites or video images.

2. Browse – This permission controls whether the site can browse through objects independently, e.g. in the scenario of solid modeling, independent control of viewing transformations on the model or independent viewing of a a video clip in the object is regulated by this permission.

3. Modify – This controls participation in a collaboration. Only the sites with this permission can collaborate through the session manager (i.e. modify the state of the collaboration).

4. Copy – This permission regulates copy propagation. It permits sites to obtain a private, local copy of a shared object.

5. Grant – This is a site permission. The initiator of the session grants this permission to other trusted sites giving them the right to grant permissions to other members.

2.2.2 Collaboration Maintenance

A collaboration is initiated through the local SHASTRA Kernel by one of the Fronts, by specifying the participants (other Fronts), and the capabilities they will have during the collaboration. Collaboration is performed under the auspices of a Session Manager.

The application specific part of the Session Manager is a repository of the objects in that collaboration, which are introduced into the collaboration by participating Fronts. This part has a constraint management subsystem which resolves conflicts that arise as a result of multi-user interaction over the shared objects, and therefore maintains mutual consistency of multiple operations. Constraint management is fairly straightforward since there is only one site performing the arbitration viz. the Session Manager.

2.2.3 Information Flow

SHASTRA Kernels are a repository for contact information for ongoing collaborations and other concurrently executing applications in the environment. Every Kernel maintains a control link to all systems on the local host. A Session Manager connects to its local Kernel, and maintains data links to all the members of the collaboration it is conducting. It is at the hub of a star topology. This setup tends to suffer from performance degradation when the number of collaborators is large and data volume is large, but works well for typical collaborative groups in the design setting. (We currently simulate broadcast using point-to-point transmissions.) We are studying reliable broadcast protocols, and multicast mechanisms to alleviate this problem.

In a non-collaborative setting for distributed problem solving, applications connect directly to each other, using the contact information stored by SHASTRA, to exchange data and utilize functionality offered. All communication is done using TCP/IP, and data transfers are performed via the XDR based SHASTRA protocol [3].

2.2.4 Networked Collocation

Multimedia communication facilities greatly facilitate cooperative design effort, by permitting rapid exchange of ideas. Current functionality for virtual team support and collocation in SHASTRA supports simultaneous but independent, unsynchronized virtual channels for transmission of text, 2D and 3D graphics, audio and video information over the network during the collaboration. The SHASTRA layer provides facilities for transfer of all those forms of data in a client-server setting, between two Fronts, for visualization or exchange of multimedia components of objects in the environment. The same data can be transferred among all the participating Fronts in a collaborative setting for conferencing. The objective is to provide a media-rich communication substrate for the design of multimedia applications, by relieving application developers of the burden of low-level manipulation. Service tools are described in Section 3. All conferencing applications are collaborating instances of SHASTRA Service Applications – they are built on top of the collaborative substrate of SHASTRA.

2.3 Highlights

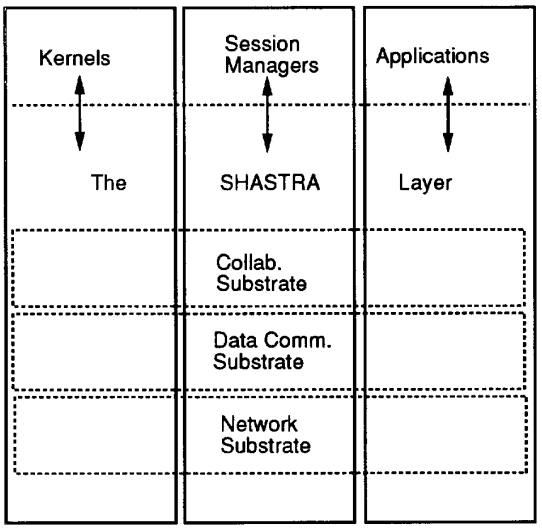

Figure 2: The SHASTRA Layer

SHASTRA provides a framework for the implementation of the Object–Multiple View–Multiple Controller paradigm for multi-user applications where information shared between collaborating participants can be viewed and altered independently, but consistently, at the different sites. It provides a rich substrate for the development of distributed and collaborative applications by abstracting away the details of the underlying communication subsystem from the application developer. Figure 2 shows this abstract layering.

The SHASTRA architecture uses a replicated computation model for the multiple user system with a copy of the application running at each site involved in the collaboration. It provides collaborators with multiple channels of communication (text, graphics, audio and video) to aid the collaboration effort. Centralizing the shared data in the session manager permits late joiners of ongoing sessions to be brought up-to-date immediately. The session manager also cen-

tralizes broadcasts alleviating the worries of misordering of input events in distributed settings. SHASTRA provides a framework for specifying a constraint management subsystem which can be adapted to diverse applications. The environment promotes rapid prototyping of multi-user collaborative tools.

We use the X Window System (X11) as our UIMS because of its widespread availability. The communication subsystem uses TCP as its reliable communication protocol. Device dependence woes arising from machine representation are circumvented by using XDR to encode all information. The implementation of a scientific manipulation environment involves a large symbolic manipulation substrate which has been implemented in Common Lisp, which again makes it very portable. The LISP and C processes communicate via a protocol to remove the implementation dependence of foreign function calls. SHASTRA provides a rich substrate of graphics algorithms and decomposition techniques as well as a powerful numeric substrate as readily available C libraries. The current implementation of SHASTRA runs on a mix of Sun, HP and SGI workstations.

The distributed aspect of SHASTRA, with its emphasis on interoperability of tools provides a powerful environment for development of distributed problem solving applications.

3 Shastra Services

SHASTRA services are applications in their own right which provide specific functionality of great utility to all other Fronts in the environment. The current set of services contains communication tools. The objective is to provide a media-rich communication substrate for the design of multimedia applications, by relieving application developers of the burden of low-level manipulation. In the scientific setting, especially in design and analysis, most of the information shared by a collaborating team is oriented towards structured 3D graphics, and is typically application specific. However, inclusion of facilities for text, 2D and 3D graphics, audio and video communication has greatly enhanced the quality of communication, enabling the design of more sophisticated applications. For details of the architecture of these service tools, see [18].

3.1 SHA-TALK

This is a SHASTRA environment tool built to demonstrate the ease of creation of collaborative applications in the SHASTRA environment. SHA-TALK is a text communication tool which uses text message bites generated from the participating Fronts as the collaboration objects. A collaborative session consisting of SHA-TALK applications, started by any Front, lets a group of collaborators synchronously exchange text messages in a multi-way communication session. The Session Manager causes the creation of a text buffer window for every remote site that has Modify permissions for the session at every Front that has Access permission, and continually updates the message boards as messages arrive. It is a simple communication facility. Figure 3 shows one site in a 3-way talk to support a design session.

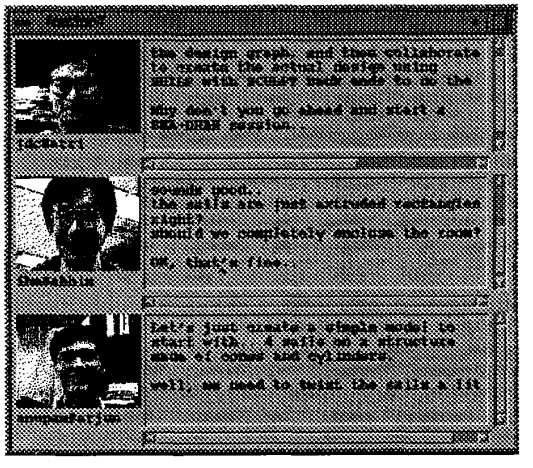

Figure 3: A Text Conference using SHA-TALK

3.2 SHA-DRAW

SHA-DRAW is a SHASTRA environment sketching tool, which facilitates the generation and display of simple 2D pictures [18]. In keeping with SHASTRA philosophy, SHA-DRAW isolates the functionality of 2D sketching, and in the client-server setting is used as a back end to display sketches stored in objects in the SHASTRA environment. SHA-DRAW is also a collaborative graphical communication tool which uses simple 2D graphics objects generated from the participating Fronts as the collaboration objects and displays them in hardware independent XS windows. A collaboration session consisting of SHA-DRAW applications lets a group of collaborators synchronously create and edit simple 2D sketches. The Session Manager causes

the creation of a shared XS Graphics window [19] at every site that has Access permission for the session. Collaborators with Modify permission for the collaboration can draw into the shared window. The Session Manager is responsible for broadcasting images as soon as they are created, so that the shared window becomes a common drawing surface. Collaborators can specify permissions for the picture primitives, which controls whether or not other collaborators can alter or delete them. In the master-slave mode which is activated by a Front with appropriate permissions, only one user draws at a time, and everybody else just views the results. Users take turns by passing the baton. Figure 4 depicts one site in a collaborative sketching session to support a design session.

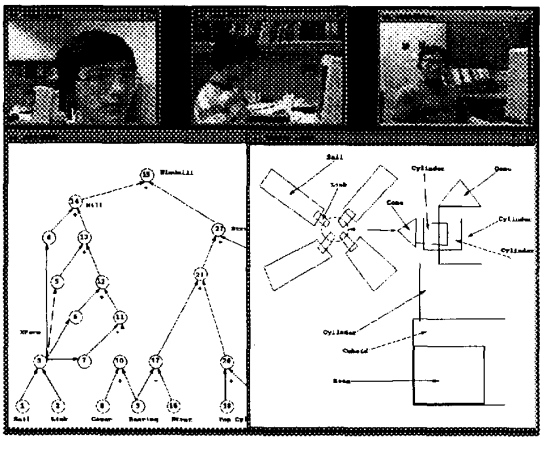

Figure 4: Collaborative Sketching using SHA-DRAW

SHA-DRAW supports modes for drawing graphs and flowcharts, and has proved to be very useful as a brainstorming tool in the SHASTRA environment, especially in conjunction with the other services.

3.3 SHA-POLY

SHA-POLY provides a substrate for collaborative visualization of graphical output of different toolkits in the environment. Toolkits can connect to an instance of SHA-POLY and use it as a back end to browse through graphical information generated by using its windows like a 3D drawing surface. A collaborative session involving instances of SHA-POLY is used to collaboratively visualize results of physical simulations, and is a very useful tool in a design environment. The Session Manager causes the creation of a shared XS Graphics window [19] at every site that has Access permission for the session. Collaborators with Browse permission can interact with the displyed 3D information. Collaborators with Modify permission for the collaboration can alter visualization parameters in the shared window. The Session Manager is responsible for broadcasting graphical data as soon as it enters the system, so that the shared window becomes a common viewing surface.

3.4 SHA-PHONE

The SHA-PHONE application serves as the Audio Processing toolkit in the environment [18]. In a client-server setting, Fronts connect to an instance of SHA-PHONE, which runs on a machine with audio hardware, and record and playback sound data. For example, SHILP automatically requests a local SHA-PHONE instance to play back comments about design aspects of a solid model which are stored in it as audio data. Similarly, a user can store his interpretation of the results of a finite element analysis problem in BHAUTIK in audio form with the results of the analysis, by requesting SHA-PHONE to record his voice. This is in keeping with the SHASTRA paradigm of isolating functionality in a specific tool, and subsequently making it available to all other systems in the environment. Application developers are afforded the facility of building audio aspects into their applications through an abstract interface, without having to worry about implementation details.

A collaboration consisting of SHA-PHONE applications provides the mechanism to conduct an audio conference. The Session Manager causes all the participating SHA-PHONE instances to activate their audio hardware for playback and recording. Sound bites are recorded by each Front with Modify permission for the session and passed to the Session Manager, which passes them on to all Fronts that have access permission for the session. The received sound bites are superposed and played back on the audio hardware. In the master-slave mode, only the site with the baton broadcasts, and the sound is played back at all other sites. Users take turns by passing the baton. The SHA-PHONE application is currently supported on SGI and Sun workstations, and understands the AIFF(SGI) and AU(Sun) sound formats.

3.5 SHA-VIDEO

SHA-VIDEO is the video image processing toolkit in the SHASTRA environment [18]. It handles image data (without sound) – both still images and mo-

tion video. In a client-server setting, Fronts connect to an instance of SHA-VIDEO, which runs on a machine with video hardware, and record and playback video image data. For example, SHILP requests SHAPHONE to play back the video data stored in one of its solid models which shows a picture of the intended design, or a video clip that inspired the model. Similarly, a user stores a video clip of a hip implant operation in a reconstructed model of the femur in VAIDAK, for use in design of an implant for that femur in SHILP, by requesting SHA-VIDEO to record the data off a video camera or video cassette player. In a similar manner, SHA-VIDEO stores still images in multimedia objects in the SHASTRA environment.

A collaboration consisting of SHA-VIDEO applications provides the mechanism to conduct a silent video conference. The Session Manager causes all the participating SHA-VIDEO instances to activate their video hardware (if available) for recording. It then causes the creation of an image display window at each site for every site that is capturing video frames. Video frames are recorded by each Front with Modify permission for the session and passed to the Session Manager, which passes them on to all Fronts that have access permission for the session. The received image frames are displayed in the image window.

Video conferences are compute and communication intensive (especially since in our case, image processing is done in software). Currently, JPEG compression is performed only on still images, since we do it in software. In the master-slave mode, only the site with the baton broadcasts, and the image is displayed at all other sites. Users take turns by passing the baton.

SHA-VIDEO uses X11 to display images, and thus is supported on all X11 platforms. However, currently we have digitizing hardware for video image capture only on Sun workstations. The digitizing hardware understands PAL and NTSC formats. It is fairly low end and digitizes upto 10 frames per second, but currently this seems to be sufficient.

Video conferencing is very useful as a communication tool for collaborative sessions in the design setting, as it is faster to show a collaborator what a design looks like, as opposed to drawing it or describing it in words. In conjunction with an audio conference, a video conference makes design situations very natural.

4 Shastra Toolkits

Designed for extensibility, the SHASTRA environment is continuously growing. Currently GANITH, SHILP, VAIDAK, BHAUTIK, SPLINEX and GATI are scientific applications under the SHASTRA umbrella (see Figure 5). GANITH, SHILP, and VAIDAK existed before SHASTRA, and had to be modified to adhere to the Application Architecture specifications. Once integrated into the SHASTRA environment, these toolkits provide their functionality to the others to permit distributed problem solving.

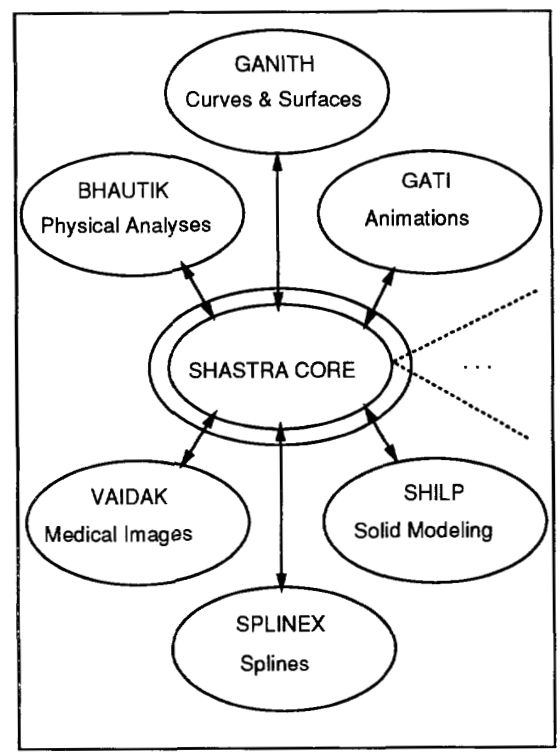

Figure 5: The SHASTRA World

4.1 GANITH

The GANITH algebraic geometry toolkit manipulates arbitrary degree polynomials and power series [22]. It can be used to solve a system of algebraic equations and visualize its multiple solutions. Example applications of this for geometric modeling and computer graphics are curve and surface display, curve-curve intersections, surface-surface intersections and global and local parameterizations. Power series manipulations are used to generate piecewise ra-

tional approximations to algebraic curves and surfaces. GANITH incorporates techniques for multivariate interpolation and least-squares approximation to an arbitrary collection of points and curves, and C1-smoothing of polyhedra by low-degree implicit patches. Arbitrary rational parametric surfaces can be displayed in GANITH, taking care of poles and base points. Animation facilities allow the visualization of entire families of algebraic curves and surfaces.

4.2 SHILP

SHILP is a boundary representation based solid modeling system [20]. The boundary representation data structure of SHILP solid models allows curve and surface patches to be represented either implicitly or in rational parametric form with either power or Bernstein-Bezier polynomial bases. The current functionality of the toolkit includes extrude, revolve and offset operations, edit operations on laminas and solids, pattern matching and replacement, boolean set operations, fleshing of wireframes with smooth algebraic surface patches, blending and rounding of solid corners and edges, and shaded display of solids.

4.3 VAIDAK

VAIDAK can be used to construct accurate surface and solid models of skeletal and soft tissue structures from CT (Computed tomography), MRI (Magnetic Resonance Imaging) or LSI (Laser Surface Imaging) data [25], [25]. VAIDAK incorporates both heuristic and exact methods of contouring image data, active thresholding, tiling (polygon reconstruction), and rendering reconstructed models. It also incorporates a browse feature to modify the contours, and a scanner to view image data and interactively pick threshold values.

4.4 BHAUTIK

BHAUTIK provides the tools necessary to set up and perform scientific and engineering simulations on geometric models [26]. Problems can be two or three dimensional, using objects created in VAIDAK, SHILP, or other model creation toolkits. BHAUTIK has several finite element mesh generation options, including good bounded aspect ratio triangulations. Meshes can be subdivided repeatedly to gain more accuracy in analysis and visualization. A material database is maintained which contains both structural and heat properties of various materials. Boundary conditions including external forces, fixed nodes, and external heat sources can be specified interactively. Results obtained from interfacing to finite element solvers are displayed for the user.

4.5 SPLINEX

SPLINEX is a toolkit that manipulates different geometric patches in Bernstein-Bezier basis [24]. Its main use is as a tool for interative design with geometric patches. One can model a geometric object by subdividing the object into simplex domains and then defining the patches inside these domains so that the patches can be merged to form the surface of the object.

4.6 GATI

GATI is an animation server that provides for distributed and collaborative real-time interactive animation in two and three dimensions [23]. The system supports a high level animation language based on a commands/event paradigm.

5 Collaborative Problem Solving

We briefly describe a prototype CE application implemented in SHASTRA. A more detailed treatment of this and other design applications is available in [2].

5.1 Collaborative Design in SHASTRA

An example of multi-user cooperative design in the context of SHASTRA is Collaborative Set Operation Based Design using SHILP and SCULPT toolkits. This permits a group of designers to cooperatively create a 3-D model by performing set operations on simpler models in SHILP (automatically using SCULPT, through SHASTRA, as a back end to perform the set operations). The SCULPT system is optimized to perform set operations – Union, Intersection, Difference and Complementation on polyhedral geometric models [11]. It was modified and integrated into SHASTRA. This application presents a departure from the traditional method – single user systems – by providing a collaborative design environment where a group of designers synchronously and cooperatively create large designs. It improves throughput of the design operation by providing a collaborative environment from the conception phase through the final design realization phase.

SHASTRA eliminates the need for a physical meeting to synchronize at a common starting point, by

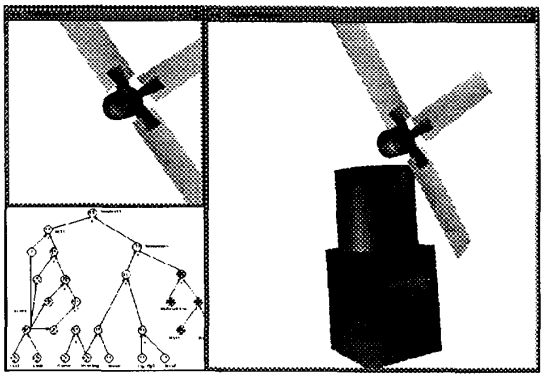

Figure 6: One site in a design collaboration in SHILP

providing support for a design brainstorming session using SHA-DRAW, the multi-user sketching tool. Audio-Visual exchange support is provided by concurrent SHA-PHONE and SHA-VIDEO sessions (see Figure 4). The group interactively creates a rough sketch of the intended design. The next step in the process uses SHA-DRAW to set up the dependencies of various parts of the design in graphical form, as a directed design graph, where nodes are solid models, and edges are dependencies of the destination node.

The SHASTRA environment for this operation consists of a collection of instances of the SHASTRA Kernel, SHILP and SCULPT. The Session Manager for the design session partitions the design graph into slightly overlapping zones when the design operation is initiated. The partitioning defines a scenario for fair, minimal-conflict cooperative interaction. Every user is responsible for filling the intermediate nodes in his zone by performing the required operation. Figure 6 shows one site in the design of a simple windmill model.

All operations are performed via the (central) session manager which is responsible for keeping all sites up-to-date, so that the users have a dynamically changing and continuously updated view of the operation in the shared windows – the design graph and intermediate models. The operation is completed when all the nodes of the design graph have been evaluated. The final model is checked for goodness, and the computation process is possibly repeated till a satisfactory model is obtained. Figure 7 shows a site at the end of the operation.

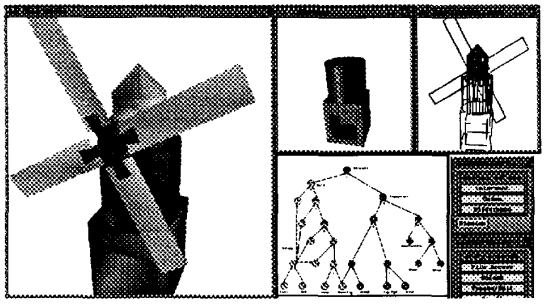

Figure 7: Another site, at the end of a Collaborative Design session

5.1.1 Enterprise Integration

The design operation can be performed without the auxiliary support of the design graph, but it makes the collaboration cumbersome, since it places the burden of visualizing all the steps of the design on all the designers. The design graph of the desired model is always available to the collaborating designers – it encapsulates the design objective and provides a very convenient medium to express partitioning information as well as collaborative task status information, in conjunction with the graphical display of intermediate steps. It gives a user a global picture of the task and shows him where he fits in. A collaborator who joins an ongoing collaboration late can quickly come up-to-date, via this information sharing and progress monitoring facility. He can use the stored sketches or images to figure out what is being designed, and participate actively in the collaboration.

5.2 Custom Implant Design

We are building an environment for Custom Design of Artificial Implants for the Human Body, using a solid modeling system (SHILP), a medical image reconstruction system (VAIDAK), a finite element analysis system (BHAUTIK) coupled with a computer aided manufacturing facility, for use by a group of surgeons, geometers, and physical analysts for custom-building limb implants with very low turnaround time. This puts us in the realm of heterogeneous collaborations – which are supported on top of different applications. The SHASTRA application architecture paradigm has greatly facilitated the task of building such a problem solving environment environment.

A medical expert uses VAIDAK to create a con-

tour model of the patients femur, from CAT Scan or MRI cross sectional images. A geometer then uses the femur model as a context in SHILP to design an implant model that satisfies geometric constraints. A physical analyst subsequently conducts a stress-strain simulation of the implant deployed in the femur, using BHAUTIK, to study load transfer characteristics. The design team iterates over this process till an optimal design is obtained. SHASTRA provides a framework for integration of the different toolkits and provides collocation facilities enabling rapid dissemination of design information.

6 Issues

We have demonstrated that collaboration in the scientific design setting is facilitated by multimedia support as well as information sharing. SHASTRA provides a rich substrate for design of CE systems. The multimedia aspect brings powerful communication primitives to the desktop. The integration of 3-D graphics into the environment adds a new dimension to the potency of this environment, as visual processing on powerful graphics engines becomes more common. Collaboration support in the environment, in the form of communication facilities and application development substrate, makes it easy to develop synchronous multi-user applications, and problem solving tools. The distribution aspect lets us build sophisticated problem solving environments consisting of interoperable tools. SHASTRA has let us integrate diverse software tools under a unifying umbrella, and provides transparent access and invocation in a distributed setting. The ease of integration of GANITH, SHILP, VAIDAK, BHAUTIK, SCULPT, SPLINEX and GATI into the SHASTRA environment demonstrates that the idea of an extensible collaborative environment for scientific manipulation is very viable.

Acknowledgements

This work was supported in part by NSF grants CCR 92-22467, DMS 91-01424, AFOSR grant F49620-93-10138, NASA grant NAG-1-1473 and a gift from AT&T.

References

[1] Ahuja, S., Ensor, J., Horn, D., (1988), "The Rapport Multimedia Conferencing System", *Proc. of Conference on Office Information Systems*,Mar. 1988.

[2] Anupam, V., Bajaj, C., (1993), "Collaborative Multimedia Scientific Design in SHASTRA", *Proc. ACM International Conf. on MultiMedia '93*, in press.

[3] Anupam, V., Bajaj, C., Royappa, A., (1991), "The SHASTRA Distributed and Collaborative Geometric Design System", *Computer Science Technical Report 91-38*, Purdue University.

[4] Crowley, T., Milazzo, P., Baker, E., Forsdick, H., Tomlinson, R., (1990), "MMConf: An Infrastructure for Building Shared Multimedia Applications", *Proc. ACM Conference on CSCW*, Oct. 1990, pp. 329-342.

[5] Dewan, P., (1992), "Enabling Technologies for Concurrent Engineering: A Position Statement", *Proc. First Workshop on Enabling Technologies for Concurrent Engineering*, Vol. 1.

[6] Ellis, C., Gibbs, S., Rein, G., (1991), "Groupware: Some Issues and Experiences", *Comm. of the ACM*, Vol. 34 No. 1, Jan 1991, pp. 38-58.

[7] Ishii, H., Miyake, N., (1991), "Toward an Open Shared Workspace: Computer and Video fusion approach of TeamWorkstation",*Comm. of the ACM*, Vol. 34 No. 12, Dec. 1991, pp. 36-49.

[8] Jagannathan, V., Shank, R., Kannan, R., Cleetus, J., Brandt, W., (1992), "Integration Technology for Concurrent Engineering", *Proc. First Workshop on Enabling Technologies for Concurrent Engineering*, Vol. 1.

[9] Londono, F., Cleetus, K., Nichols, D., Iyer, S., Karandikar, H., Reddy, S., Potnis, S., Massey, B., Reddy, A., Ganti, V., (1992), "Managing Chaos: Coordinating a Virtual Team", *Proc. First Workshop on Enabling Technologies for Concurrent Engineering*, Vol. 1.

[10] Mantei, M., (1988), "Capturing the Capture Concepts: A Case Study in the Design of Computer-Supported Meeting Environments", *Proc. Conf. on Computer-Supported Cooperative Work*, 257-270.

[11] Naylor, B., and Thibault, W., (1987), " Set Operations on Polyhedra using Binary Space Partitioning Trees", Computer Graphics Vol. 21 No. 4, 1987.

[12] Patterson, J., Hill, R., Rohall, S., Meeks, M., (1990), "Rendezvous: An Architecture for Synchronous Multi-User Applications", *Proc. ACM CSCW 90*, 317-328.

[13] Rangan, V., Vin, H., (1992), "A Unified Framework for Modeling Synchronous and Asynchronous Multimedia Collaborations", *Proc. First Workshop on Enabling Technologies for Concurrent Engineering*, Vol. 1.

[14] Scheifler, R., Gettys, J., Newman, R., (1986), "The X Window System", *ACM Transactions on Graphics*, 5, 2, 79-109.

[15] Stefik, M., Foster, G., Bobrow, D., Kahn, K., Lanning, S., Suchman, L., (1987), "Beyond the Chalkboard: Computer Support for Collaboration and Problem Solving in Meetings", *Comm. of the ACM*, 30, (1), 32-47.

[16] Srinivas, K., Reddy, R., Babadi, A., Kamana, S., Kumar, V., Dai, Z., (1992), "MONET: A Multimedia System for Conferencing and Application Sharing in Distributed Systems", *Proc. First Workshop on Enabling Technologies for Concurrent Engineering*, Vol. 1.

[17] Sriram, D., (1992), "Design as a Collaborative Process", *Proc. First Workshop on Enabling Technologies for Concurrent Engineering*, Vol. 1.

[18] Anupam, V., Bajaj, C. (1993), "Multimedia Communication Facilities in the SHASTRA Environment", *Manuscript*, Computer Science Department, Purdue University.

[19] Anupam, V., Bajaj, C., Burnett, A., Fields, M., Royappa, A., Schikore, D., (1991), "XS: A Hardware Independent Graphics and Windows Library", *CAPO Technical Report 91-28*, Computer Science Department, Purdue University.

[20] Anupam, V., Bajaj, C., Dey, T., and Ihm, I., (1991), "The SHILP Solid Modeling and Display Toolkit", *CAPO Technical Report 91-29*, Computer Science Department, Purdue University.

[21] Bajaj, C., and Ihm, I., (1991), "Algebraic Surface Design with Hermite Interpolation", *ACM Transactions on Graphics*, (1991).

[22] Bajaj, C., and Royappa, A., (1989), "The GANITH Algebraic Geometry Toolkit", *in Proceedings of the First International Symposium on the Design and Implementation of Symbolic Computation Systems, Lecture Notes in Computer Science*, No. 429, Springer-Verlag (1990), 268–269.

[23] Bajaj, C., Cutchin, S., (1992), "Interactive Animation using GATI", *Proc. CGI '93, Lausanne, Switzerland*, Feb. 1993.

[24] Bajaj, C., Chen, J., Evans, S., (1992), "Distributed Modeling and Rendering of Splines using GANITH and SPLINEX", *Manuscript*, Computer Sciences Department, Purdue University, Dec. 1992.

[25] Bajaj, C., Fields, M., (1993), "The VAIDAK Medical Image Model Reconstruction Toolkit", *Proceedings of the 1993 Symposium on Applied Computing*, Feb. 1993.

[26] Bajaj, C., Schikore, D., (1993), "Distributed Design of Hip Prosthesis Using BHAUTIK", *Proceedings of the 1993 Symposium on Applied Computing*, Feb. 1993, pp. 36-39.