1993

Collaborative Multimedia Scientific Design In SHASTRA

Vinod Anupam

Chandrajit L. Bajaj

Report Number:

93-007

COLLABORATIVE MULTIMEDIA SCIENTIFIC DESIGN IN SHASTRA

Vinod Anupam
Chandrajit L. Bajaj

CSD-TR-93-007
January 1993

Collaborative Multimedia Scientific Design in SHASTRA *

Vinod Anupam   *Chandrajit L. Bajaj*

Department of Computer Science
Purdue University
West Lafayette, IN 47907

Abstract

We address the issue of design of architectures and abstractions to implement multimedia scientific manipulation systems, propose a model for the integration of software tools into a multi-user distributed and collaborative environment on the multimedia desktop, and briefly describe a prototype CSCW infrastructure which we have used to implement a scientific manipulation environment. Finally, we present example design systems to exhibit that multimedia interfaces, incorporating text, graphics, audio and video, greatly facilitate distributed and collaborative scientific design effort.

SHASTRA [1] is a distributed and collaborative geometric design and scientific manipulation environment. In this system we address the research and development of the next generation of scientific software environments where multiple users (say, a collaborative engineering design team) create, share, manipulate, analyze, simulate, and visualize complex three dimensional geometric designs over a distributed heterogeneous network of workstations and supercomputers. SHASTRA consists of a growing set of interoperable tools for geometric design and scientific analysis, networked into a highly extensible environment. It provides a unified framework for collaboration, session management, multimedia communication, and data sharing, 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 scientific data.

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. At the system level, SHASTRA specifies architectural guidelines and provides communication facilities that let toolkits cooperate to utilize the functionality they offer. At the application level, it provides collaboration and multimedia facilities that let users cooperate. A marriage of the two lets us design sophisticated problem solving environments.

*Supported in part by NSF Grant CCR 90-00028 and AFOSR-91-0276

[1]SHASTRA is the Sanskrit word for Science

1 Requirements of the Scientific Setting

Advances in computer technology, high speed networking, compression techniques, as well as audio and video processor architectures are making multimedia communication components like high speed full color graphics, high resolution images and motion video, CD quality sound, and real time audio-video synthesis readily available on the desktop.

Harnessing current computing technology and utilizing multimedia functionality and performance has made it possible to tackle progressively larger problems. Scientific manipulation systems inherently have commonality. They often need interaction tools, display and visualization facilities, algebraic processing substrates, simulation tools etc. A large part of the effort in development of such systems goes into needless re-implementation of existing functionality due to lack of ease of integration into new technology. It is very productive to be able to use facilities provided by one toolkit within another by integrating a collection of highly function-specific tools into a loosely coupled and extensible environment where the applications themselves cooperate to perform tasks.

Scientific manipulation is often computation intensive. The infrastructure must intelligently distribute the input of low computation tasks – to obtain the parallelism benefit of distribution, and the output of high computation tasks – to emphasize sharing of resources among applications. The infrastructure for a collaborative scientific environment must provide mechanisms to support a variety of multi-user interactions spanning the range from demonstrations and walk-throughs to synchronous multi-user collaboration. In addition the infrastructure must facilitate the exchange of multimedia information which is often useful to successfully communicate at the time of design, and to share the results of scientific tasks, and often necessary to actually solve problems.

Since a scientific manipulation environment consists of multiple tools, the CSCW infrastructure must provide mechanisms for both homogeneous collaborations – multiple instances of the same application, and heterogeneous collaborations – cooperating instances of different applications. Finally, a powerful infrastructure for the design of collaborative scientific applications must provide a convenient abstraction to the application developer, shielding him from lower level details, while providing him with mechanisms to flexibly tailor cooperative interactions.

2 Overview

The rest of this paper is structured as follows. This section presents a brief introduction to SHASTRA in the context of related work. Section 3 describes the architecture briefly, and then highlights the system's features. Some SHASTRA Toolkits are briefly described in Section 4 and standard SHASTRA Services are described in Section 5. Specific details of example applications for distributed and collaborative problem solving are presented in Section 6. Section 7 addresses issues and future direction.

2.1 SHASTRA: A Scientific Manipulation 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 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. These software tools are tailored for use in both scientific experimentation and education, with an emphasis on distributed and collaborative geometric problem solving.

The SHASTRA environment consists of a group of interacting applications. Some applications are responsible for managing the collaborative environment (the Kernel applications), whereas others provide specific services (the Service Applications), while yet others provide scientific design and manipulation functionality (the SHASTRA Toolkits). Service applications are special purpose tools for multimedia support – providing textual, graphical, audio and video communication. Different 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 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.

SHASTRA provides a powerful substrate for the design of geometric manipulation toolkits. It supports rapid prototyping of such systems emphasizing distributed design and collaboration. Abstracting away from the scientific manipulation aspect of its implementation, SHASTRA represents a paradigm for interoperable software tool design.

2.2 Related Work

A teleconferencing approach to modeling and analysis of empirical data is presented in [19], where the authors hypothesize about a collaborative scientific visualization environment. A discussion of an interactive visualization environment for 3D imaging is presented in [35], where the the authors adopt the electron microscope to perform as a computer peripheral. An environment like SHASTRA makes it convenient to build collaborative visualization and manipulation facilities, that support resource sharing in a distributed setting.

Standardization in the digital multimedia domain for audio data, images and motion video like JPEG and MPEG [45], [26], [33] etc. in conjunction with T3 and FDDI networks are making it easy to build distributed conferencing systems or application development environments (e.g. NeXTstep [18]) on the desktop. Advances in computer technology are making it cost effective to integrate multimedia into the desktop as in DVI [29], [27]. Some hypertext systems have been used as effective archiving and information tools [44], [38]. Now, document structure standardization like SGML [17] is making hypermedia systems, with the capability to create and browse through complex networks of linked multimedia documents like HyTime, very attractive [37]. Intermedia [28] presents a system which supports multiuser access to hypermedia documents. Tools like PhoneTalk [39] and SuiteSound [41] present tested mechanisms for audio communication.

Out intention in SHASTRA is to provide a distributed heterogeneous environment for multimedia collaboration and to facilitate the development of collaborative applications by providing a rich substrate. Specifically, we focus on the scientific domain, though the collaboration facilities are flexibly adapted to other areas.

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 [23]. Research into computer and video fusion to support collaborative work has resulted in TeamWorkStation [30] which uses video-overlaid shared drawing surfaces in addition to wired video and audio communication links. The Capture Lab uses personal workstations with a large shared monitor, with one user controlling the monitor at a time for sketching [34]. Colab is a collaborative meeting facility which permits multiple users to simultaneously manipulate drawn objects [43]. GROVE [22] and ShrEdit [21] are editors that are designed to support group editing of documents. Quilt is a group editor that lets multiple users work on different parts of the same document [24]. DistEdit [31] supports building of interactive group editors by minor alterations to existing ones. LIZA is a collection of high-level tools for rapid groupware prototyping providing support for group editing and message transmission etc. [25]. Rendezvous proposes a powerful architecture for multi-user applications [40].

These systems either provide content independent sharing of drawing and viewing surfaces, or are very task specific. SHASTRA lets us build applications with shared drawing and viewing surfaces by supporting content dependent sharing – 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.

3 SHASTRA – System Features

3.1 System Architecture

The SHASTRA architecture is described in detail in [3]. Here, we present salient features. 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 that let 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. Utilization of these facilities in SHASTRA lets us design sophisticated problem solving virtual machines.

3.1.1 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. 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]. All communication between Fronts, Kernels and Session Managers occurs via their Network Interfaces. The SHASTRA architecture is depicted in Figure 1. Systems 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.

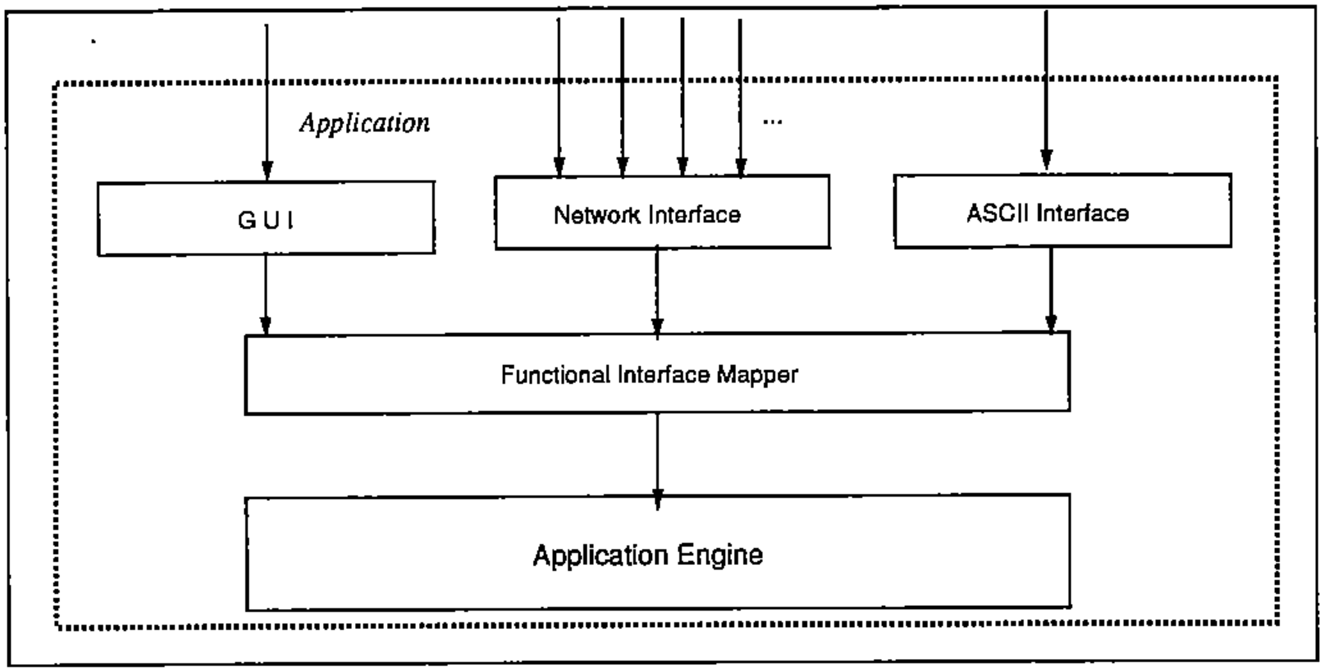

Figure 1: The Architecture of a SHASTRA Application

The entire set of connected Network Interfaces of Kernels, Session Managers and Fronts implements the abstract SHASTRA layer (see Figure 4) which maintains the collaborative environment, provides access to functionality of different systems, and provides facilities for initiating, terminating, joining, leaving and conducting collaborations.

3.1.2 The SHASTRA 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.

3.1.3 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.

3.1.4 SHASTRA 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.

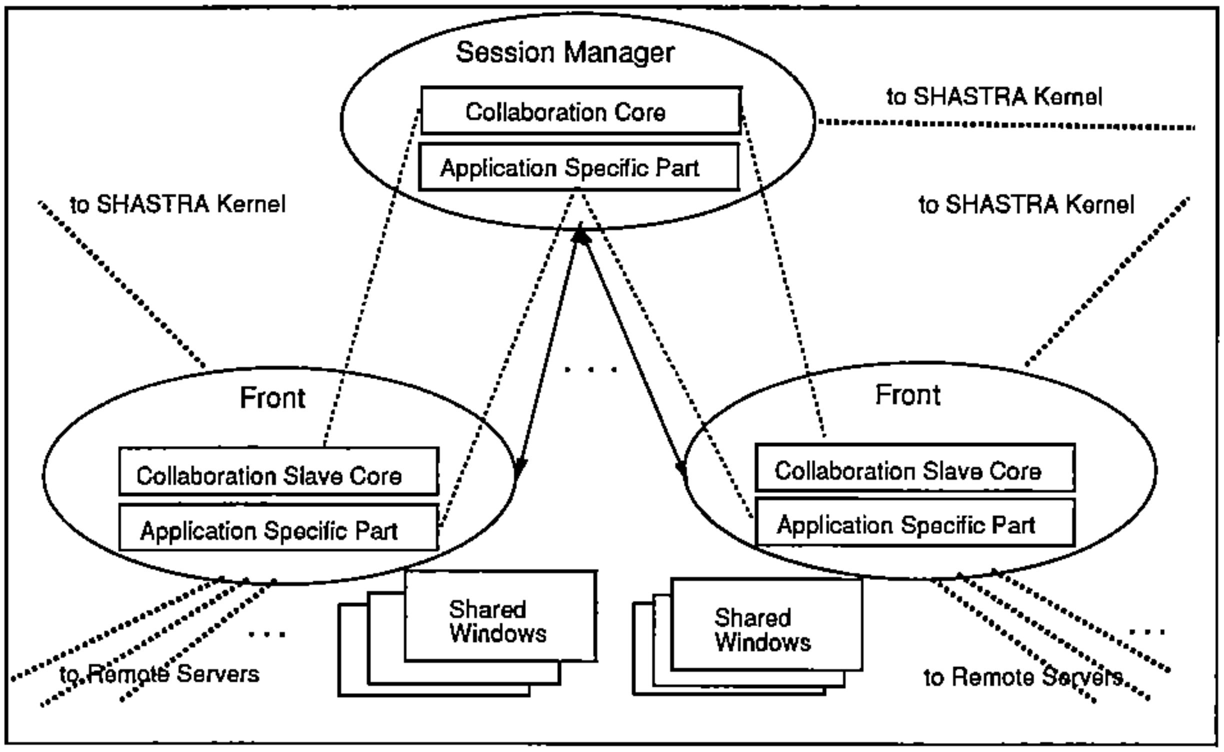

Figure 2: Architecture of a Collaborative Session in SHASTRA

The architecture of a typical collaborative session in SHASTRA is as in Figure 2. 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.

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. A token passing scheme is used to select a master who has total control over the session interaction as long as he has the token. Identity of the master is broadcast to all collaborators. In the preemptive mode, a Front can take the token away from the master. In the non-preemptive mode other participants can request the master to release the token. Token passing is performed in the Session Manager, This guarantees that the token will not get stuck if any participant gets computation bound. Fronts communicate with the session manager to capture and relinquish the token, in conjunction with the regulatory subsystem.

3.2 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.

3.2.1 Access Regulation Mechanism

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 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.

The regulatory mechanism aims at being able to support a variety of interactions ranging from master-slave mode blackboarding to multiple site collaboration.

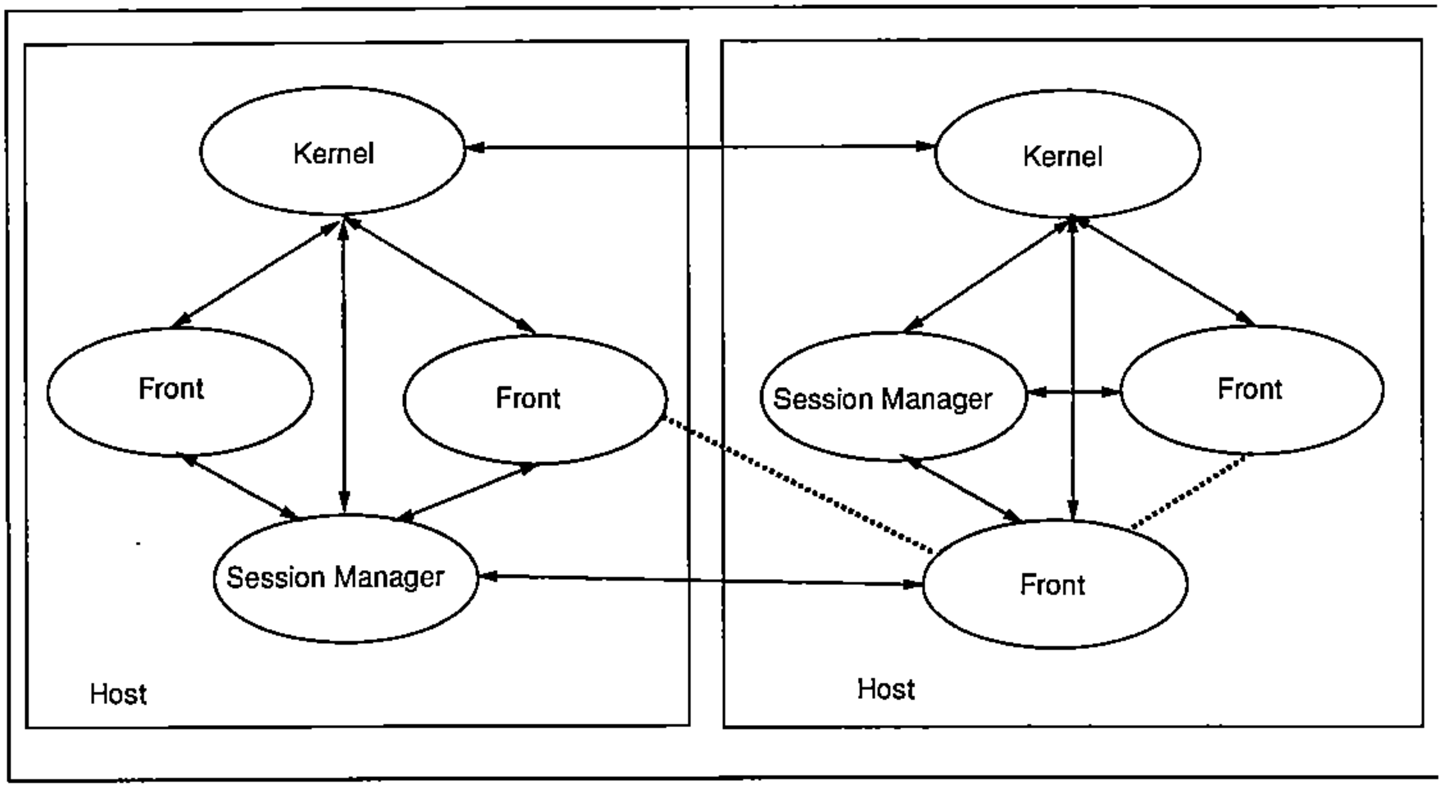

Figure 3: Information Flow in the SHASTRA Environment

3.2.2 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.

3.2.3 The Multimedia Aspect

Current functionality in SHASTRA supports simultaneous but independent, unsynchronized virtual channels for transmission of text, 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. Section 5 presents more details about multimedia service applications that have been modularly integrated into the environment. They will be replaced with newer tools as we upgrade the hardware and software multimedia facilities in the environment.

3.2.4 Information Flow

Figure 3 shows the paths of information flow in SHASTRA. 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].

3.3 Highlights

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 4 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 centralizes 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 im-

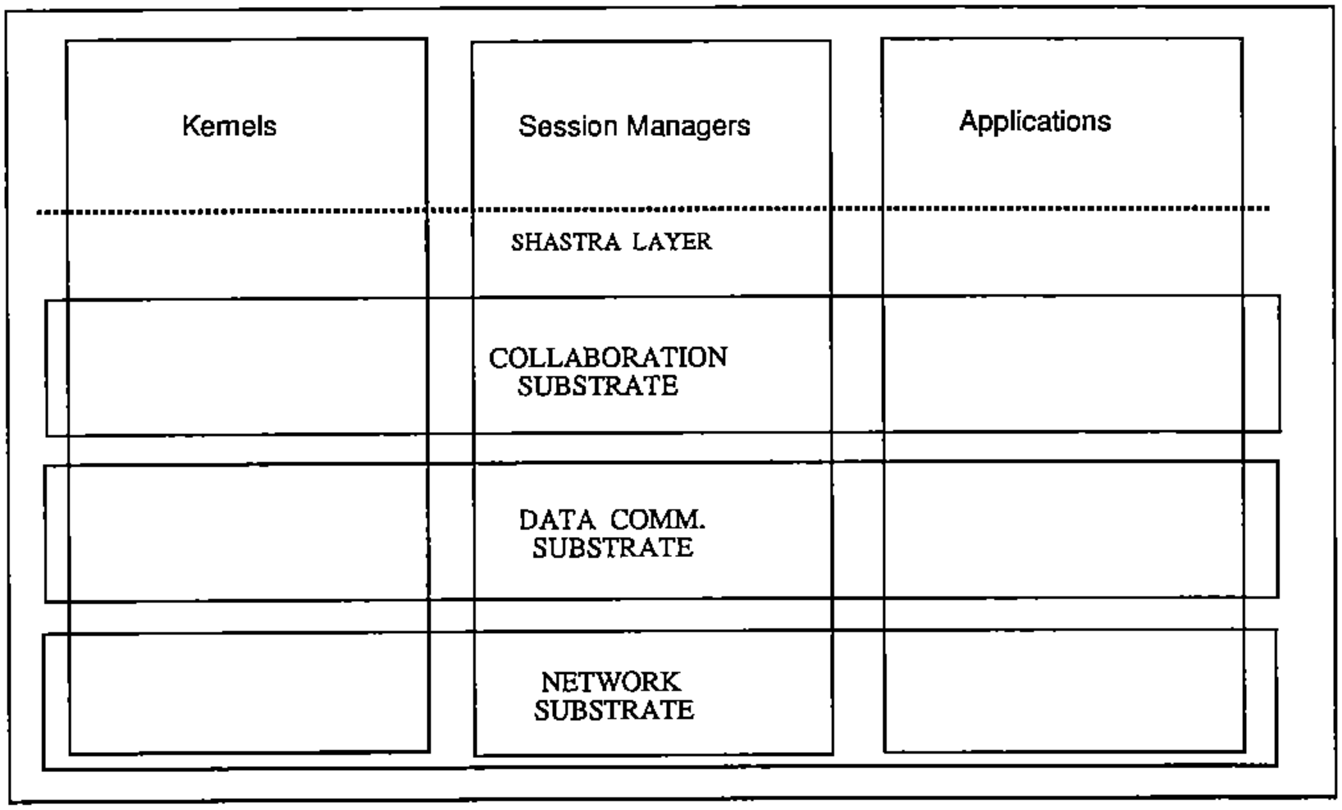

Figure 4: The SHASTRA Layer

plemented 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.

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. 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.

4.1 GANITH

The GANITH algebraic geometry toolkit manipulates arbitrary degree polynomials and power series [9]. 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 rational 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 B-Rep based solid modeling system [5]. The boundary representation data structure of SHILP solid models allows curve and surface patches to be represented either implicitly or in rational parametric form (or both when available), with either power or Bernstein-Bezier polynomial bases. The current functionality of the toolkit includes restricted extrude, revolve and offset operations, edit operations on laminas and solids, pattern/feature 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.

As an example of distributed problem solving, SHILP calls upon instances of GANITH to interpolate a polyhedral object and produce a curved surface solid model, as is shown in Figure 5.

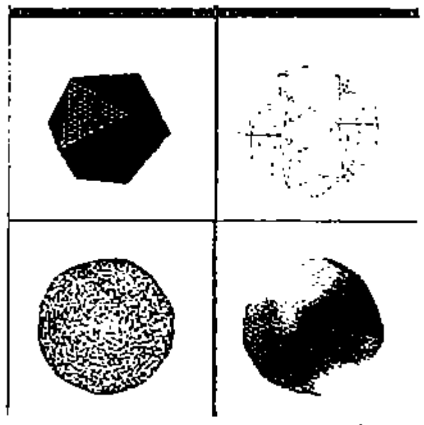
Figure 5: A Polyhedron Smoothed in SHILP via Remote Calls to GANITH

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 [7], [13]. 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.

In a distributed problem solving scenario, a medical images of the parts of the human body can be reconstructed in VAIDAK, solid models created and manipulated in SHILP. See Figure 6.

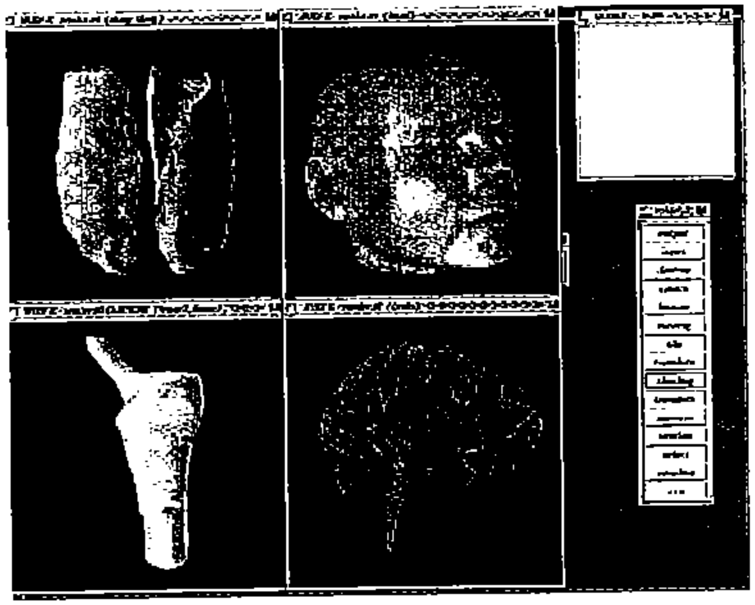

Figure 6: Medical models reconstructed in VAIDAK, manipulated in SHILP

4.4 BHAUTIK

BHAUTIK provides the tools necessary to set up and perform scientific and engineering simulations on geometric models [14]. Problems can be 2 or 3 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.

For example, a curved cross-section is created in GANITH, to obey certain continuity requirements. SHILP performs a revolve operation on the cross-section to generate a solid model for a jet engine cowling. Subsequently, BHAUTIK imports the model and generates an external mesh for fluid flow analysis, as depicted in Figure 7.

4.5 SPLINEX

SPLINEX is a toolkit that manipulates different geometric patches in Bernstein basis [12]. Its main use is as a tool for interative design with geometric patches. One can model a geometric object by subdividing

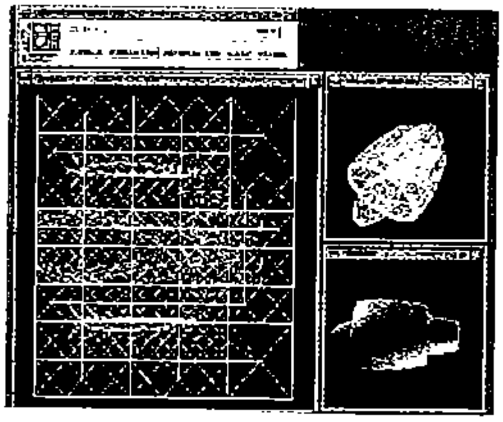

Figure 7: Mesh Generation in BHAUTIK of a jet cowling designed in SHILP

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 [11]. The system supports a high level animation language based on a commands/event paradigm.

5 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, picture, sound 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 [2].

5.1 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.

5.2 SHA-DRAW

SHA-DRAW is a SHASTRA environment sketching tool, which facilitates the generation and display of simple 2D pictures [2]. In keeping with SHASTRA philosopy, 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 [6] 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 token.

Figure 8 depicts one site in a collaborative sketching session.

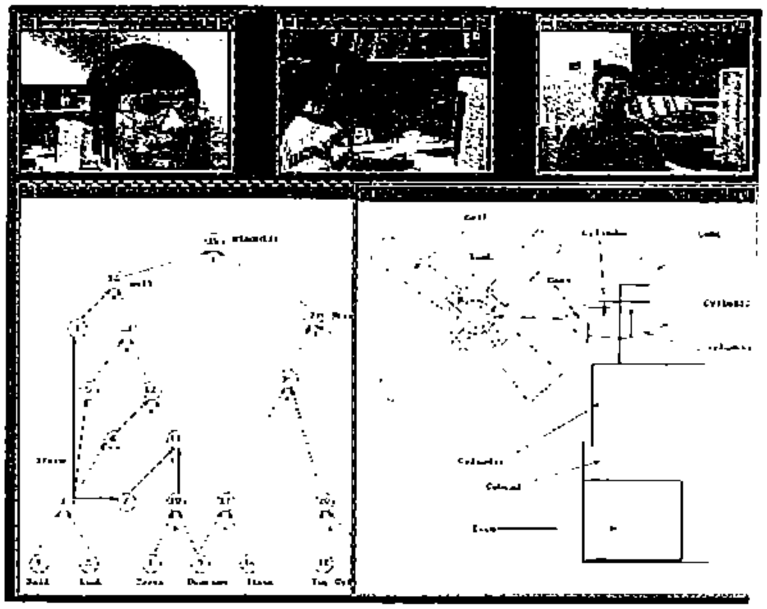

Figure 8: Using a shared drawing surface in SHA-DRAW for brainstorming

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

5.3 SHA-PHONE

The SHA-PHONE application serves as the Audio Processing toolkit in the environment [2]. 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 in 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 token broadcasts, and the sound is played back at all other sites. Users take turns by passing the token. The SHA-PHONE application is currently supported on SGI and Sun workstations, and understands the AIFF(SGI) and AU(Sun) sound formats.

5.4 SHA-VIDEO

SHA-VIDEO is the video image processing toolkit in the SHASTRA environment [2]. It handles image data (without sound) – both still images and motion 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 SHA-PHONE 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. For example, in Figure 9 a researcher uses a live video window to confirm the topological accuracy of a reconstructed femur in VAIDAK. In exactly the same 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-PHONE 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. For motion video, more than 4 collaborators broadcasting full-size images simultaneously can choke the star-connected Session Managers. Performance is significantly better with small images. This problem will partly be solved when we start to use real multicasts over the network, instead of simulating

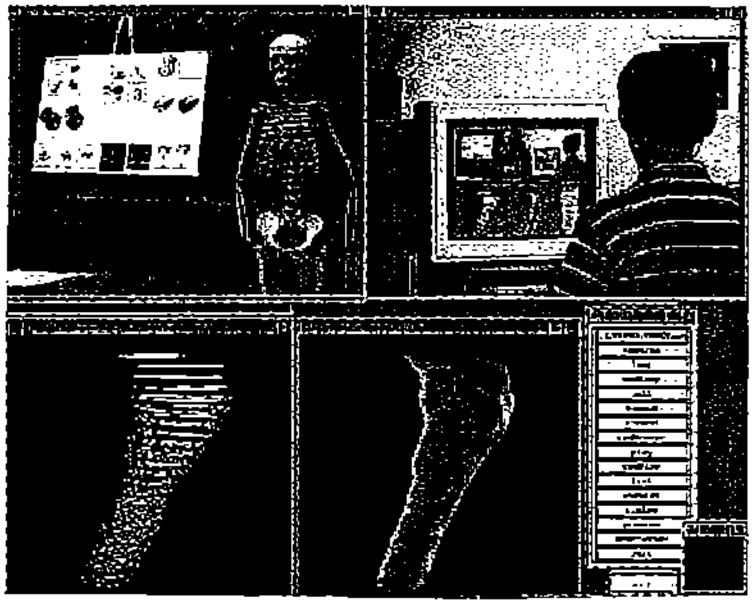

Figure 9: Using video images to confirm femoral topology in VAIDAK

them. Advances in high-speed networking will alleviate this problem. In the master-slave mode, only the site with the token broadcasts, and the image is displayed at all other sites. Users take turns by passing the token. Here again, performance suffers if the number of collaborators is very large, but the problem will be solved by using true multicasts.

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.

6 Distributed and Collaborative Problem Solving

6.1 Collaborative Smoothing in SHASTRA

An example of multi-user cooperative design in the context of SHASTRA is Collaborative Smoothing using SHILP and GANITH toolkits. This permits a group of users to collectively smooth out a rough polyhedral model by fitting C1 or C0 continuous patches using Hermite interpolation [8], [10]. GANITH is optimized to perform algebraic manipulation – curve-curve, curve-surface, and surface-surface intersection, as well as interpolation. SHILP is optimized for Boundary Representation based solid modeling. A coordinated nexus between the two applications lets us add a powerful design facility to the environment by drawing upon the functionality-sharing application level cooperation substrate, and the user level collaboration substrate of the SHASTRA environment.

6.1.1 Motivation

The smoothing operation we describe arises in our geometric design environment in two different situations. It provides an easy method for generating solid models with curved surfaces from approximate polyhedral models that have been created interactively. The operation is also used as the last phase of solid model creation from reconstruction of medical images. Medical image reconstruction results in polyhedral models with very high feature density(vertices, edges, and faces). A density reduction step generates a rough polyhedral model from the dense model by collecting features into groups. The smoothing operation results in a low feature density model which accurately represents the medical image [5].

Generation of the surface patch is a compute intensive operation. Also, patch computation for a face is independent of that for other faces, except for continuity requirements, and can be done in parallel. However, surface curvature parameters often require interactive twiddling by the designer, in order to adhere to global or local requirements, and to control the goodness of fit. Collaborative Smoothing parallelizes this step, by allowing multiple designers concurrent access, and thus significantly improves design throughput.

6.1.2 Operation Outline

Smoothing is performed in two phases – curved wireframe generation and interpolating surface computation. The curved wireframe is generated in SHILP by specifying continuity parameters, as well as edge curvature and vertex normals. The curved wireframe specifies the intersection of the interpolating surfaces, where they satisfy the specified continuity requirement (C1 or C0). Parameters for controlling normal values and edge curvature are specified graphically through the SHILP user interface. Actual fleshing of the wireframe is done by requesting service from GANITH, which is an algebraic geometry toolkit. In a single user setting, the designer specifies the parameters for all the faces, based on the requirements for the design, and subsequently computes the interpolating surfaces by making calls to GANITH servers. The obtained model is checked for goodness using still images, motion video or some calculated metric for reference, parameters are twiddled and the computation process repeated till a satisfactory model is obtained.

6.1.3 The SHASTRA setting

The SHASTRA environment for this operation consists of a collection of instances of the SHASTRA Kernel, SHILP and GANITH. A collaborative session is initiated by one of the SHILP users in the environment. He specifies, to the local Kernel, the list of SHILP users that will be invited to participate in the session, and becomes the group leader. The Kernel instantiates a Session Manager, which starts a session with the group leader as its sole participant, and then invites the specified users of concurrently executing remote SHILP sessions to participate. Users that accept are incorporated into the session. The Group Leader uses the access regulation mechanism to specify what the other participants can/can not do. He can invite other users to join the session at any point in the collaboration. Other remote users can request to join the session, and current participants can leave the session at any time.

6.1.4 The Collaborative Operation

Every participating SHILP session creates a shared window in which all the cooperative interaction occurs. A local window which displays only the user's sub-problem can also be created. A user introduces the object to be smoothed by selecting it into the Collaboration Window. The Session Manager is responsible for providing access to the object at all participating sites which have the Access permission, and for permitting interaction relevant to the operation at sites which have Modify permission for the collaboration.

The Session Manager partitions the object into non-overlapping zones when the smoothing operation is initiated. The partitioning is based on the number of people in the collaboration, and on the number of subtasks left in the operation (the number of uninterpolated faces, in this case). It aims to minimize the number of boundary edges of partitions, which are regions of contention in this collaboration, since adjacent faces have to obey the continuity requirement along shared edges. The partitioning defines a scenario for minimal-conflict cooperative interaction. The partitioning can be dynamically altered as users join/leave the session. The group leader can explicitly specify and alter the partitioning.

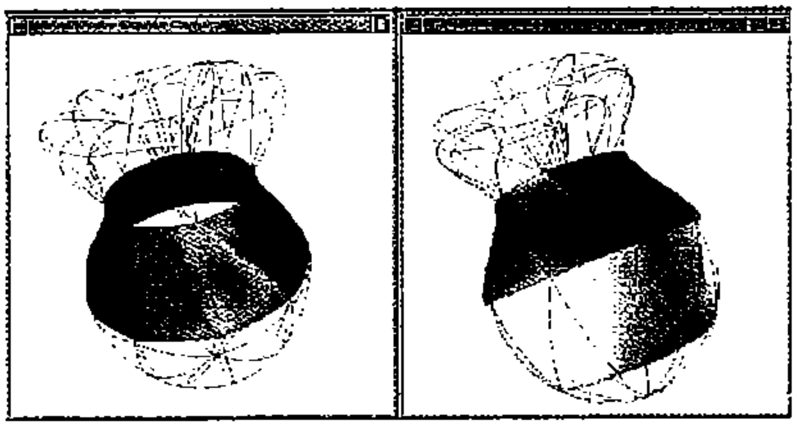
Figure 10: One site in a Collaborative Smoothing scenario in SHILP

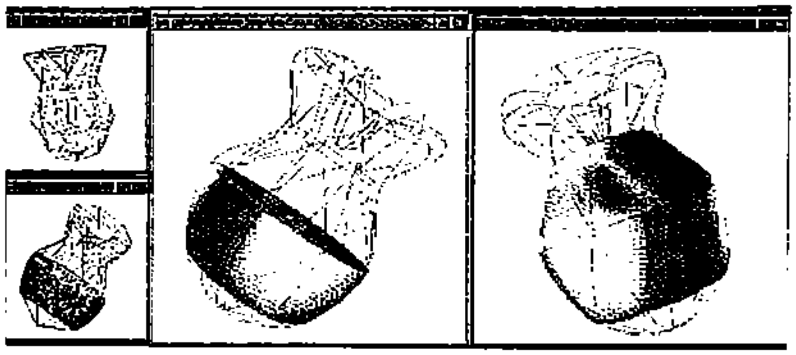
Figure 11: Another site in the Collaborative Smoothing session

The partitioned zones are assigned to the collaborating designers, and are colored differently for identification. Every user is responsible for smoothing his zone by first generating a satisfactory curved wireframe and subsequently using instances of the algebraic geometry toolkit, GANITH, to perform the actual interpolation and cycling through this process till a satisfactory design is created. Figures 10 and 11 show a view of the interaction at two sites, with the users' own zone in a local window, and the status of the operation in the shared window.

6.1.5 Collaborative Interaction

Interaction can occur in two modes. In the Regulated mode, the user responsible for a zone controls the normals and curvature control parameters for vertices, edges and faces internal to the zone. All other users are denied access to the interior by the session manager. In the Unregulated mode, the partitioning merely suggests a minimal-conflict setting, and the session manager doesn't regulate interaction beyond what is specified by the permission settings for the site and the object. In this mode any user can alter any parameter if he has Access and Modify permissions for the collaboration.

The algebraic continuity requirement imposes constraints at the boundary of a partition. Users of adjacent zones must agree about parameter settings for boundary edges so that continuity requirements are not violated. A good group design protocol for this setting is to resolve boundary condition issues at the start of the operation, to prevent unnecessary cycles due to inconsistencies later on.

The session manager allows only one user to manipulate a "hot spot" – where there is a possibility of contention – at any particular instant. It uses the first-come-first-served paradigm to decide which user gets temporary exclusive control. The last validly specified parameter value takes effect. Designers can agree on a parameter adjustment protocol using the token passing facility of the system to take turns to specify vertex normals and edge curvature along boundary edges. Alternately, they can use auxiliary communication channels (audio/text) by initiating SHA-PHONE or SHA-TALK sessions to decide mutually acceptable values, and which users will set those values. A SHA-VIDEO session can be used to inspect a physical model or picture to visually establish goodness of the smoothing operation.

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 window – the curved wireframe and interpolated patches of the object. Changing a parameter requires all of its dependencies to be re-evaluated. The operation is completed when all the polyhedral faces have been smoothed. Any site with Copy permission can then extract the model from the session and save it.

6.1.6 Regulation Context

A point to note is that the topology of the object, as defined by the connectivity of vertices and edges, does not change in the entire operation. Thus the wireframe skeleton of the desired result serves as a context for the collaborative task and is always available to the collaborating designers – in some sense they know what the resulting object will look like, and it provides a very convenient medium to express partitioning information as well as collaborative task status information. A collaborator who joins an ongoing collaboration late can quickly come up-to-date, and infer the status of the operation.

6.2 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 using SCULPT as a back end to perform the actual operations. The SCULPT system is optimized to perform set operations – Union, Intersection, Difference and Complementation on polyhedral geometric models [36]. It was modified and integrated into SHASTRA. Linking the two applications, in the context of the SHASTRA's functionality-sharing application level cooperation substrate, and user level collaboration substrate, lets us add a powerful design facility to the environment.

6.2.1 Motivation

The design paradigm in SHILP emphasizes on creation of complex models by performing operations on simpler models in a hierarchical fashion. Set Operation based design (Constructive Solid Geometry) is a very powerful and flexible way of creating intricate designs. Conventional design systems support this mechanism in the single user setting. One designer creates the design by going through the multiple steps involved. This application presents a departure from the traditional method – a collaborative design environment where a group of designers cooperatively create large designs.

In boundary representation based solid modeling systems, generation of the results of set operations is compute intensive. Also, such a design process can be represented as a tree in which the lower levels are often parallelizable – a group of designers can work independently on those parts. This application improves throughput of the design operation by providing a collaborative environment from the conception phase through the final design realization phase.

6.2.2 The Startup Problem

One of the challenges of this design scenario is that before the design process is started, some members of the team may have a mental picture of the object that they are attempting to design, while others may not. This issue is traditionally resolved by conducting a physical meeting where the team members communicate with each other, and this information exchange enables them to synchronize at a starting point, and gives them all an idea of the final design. SHASTRA eliminates the need for such a physical meeting, by providing support for a design brainstorming session. This is the first step of the design process. One of the designers initiates a collaborative 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 8). The group interactively, and quickly, creates a rough sketch of the intended design. Alternately, the image of an actual physical object or picture thereof can be broadcast to the group using SHA-VIDEO. At the end of this phase, the entire team has a reasonable idea of the task at hand.

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 leaf(0 in-degree) nodes are existing or primitive solid models, and internal nodes are intermediate models in the design process. A designated root node symbolizes the final design. Directed edges indicate that the destination node is the result of an operation on all the source nodes. Annotations in the graph indicate the operation needed to obtain the destination node from the source nodes (see Figure 8).

6.2.3 Design Outline

The operation is performed in two phases – design graph generation and model computation. The design graph is initially created in a SHA-DRAW collaboration in the context of a sketch or video image of the final model, and is a succinct summary of the entire design task. The image and/or the sketch is stored with the graph for future reference. The graph is converted into a form amenable to this operation, with maximum in-degree of the nodes being 2, since only unary and binary operations are supported. An automatic DNF (Disjunctive Normal Form) decomposition is the simplest transformation, but doesn't produce the most efficient design graph. The design team cooperatively restructures the design graph to meet the requirements.

Positioning of models is performed graphically through the SHILP user interface, to set up models in appropriate locations for set operations. The actual operations to generate intermediate and final models of the design graph are performed by requesting service from SCULPT, a Geometric Design Toolkit. In a single user setting, the designer computes the various nodes of the graph sequentially by making calls to SCULPT servers on models that have been positioned using SHILP. The final model is checked for goodness, and the computation process is possibly repeated till a satisfactory model is obtained.

6.2.4 The SHASTRA setting

The SHASTRA environment for this operation consists of a collection of instances of the SHASTRA Kernel, SHILP and SCULPT. A collaborative session is initiated by one of the SHILP users in the environment. He specifies, to the local Kernel, the list of SHILP users that will be invited to participate in the session, and becomes the group leader. The Kernel instantiates a Session Manager, which starts a session with the group leader as its sole participant, and then invites the specified users of concurrently executing remote SHILP sessions to participate. Users that accept are incorporated into the session. The Group Leader uses the access regulation mechanism to specify what the other participants can/can not do. He can invite other users to join the session at any point in the collaboration. Other remote users can request to join the session, and current participants can leave the session at any time.

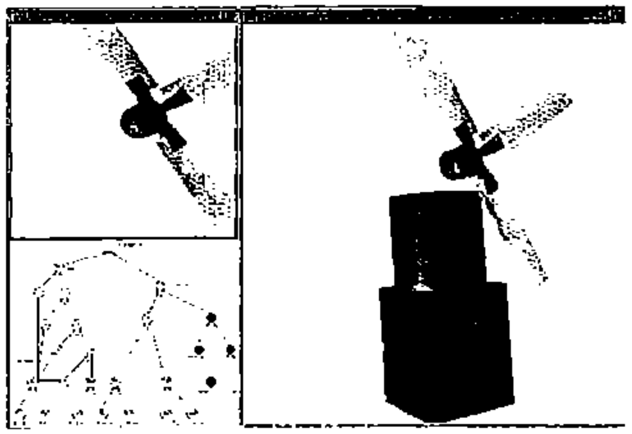

Figure 12: One site in a design collaboration in SHILP

6.2.5 The Collaborative Operation

Every participating SHILP session creates two shared windows in which all the cooperative interaction occurs, and if desired more local windows. One shared window contains the design graph that is used to regulate the entire operation. The second window contains models as they are introduced to or created in the session. Users introduce leaf node objects into the session by selecting them into the Collaboration Window. The Session Manager is responsible for providing access to the objects at all participating sites which have the Access permission, and for permitting interaction relevant to the operation at sites which have Modify permission for the collaboration.

The Session Manager partitions the design graph into slightly overlapping zones when the design operation is initiated. The partitioning is based on the number of people in the collaboration, and on the number of subtasks left in the operation (the number of uncomputed nodes in this case). It aims to minimize the number of shared nodes of partitions, which are regions of contention in this collaboration since they constitute dependencies in an otherwise parallelizable situation. The partitioning also aims to distribute the leaf nodes equally among the designers, since they usually represent nodes that have to be interactively created. The partitioning defines a scenario for fair, minimal-conflict cooperative interaction. The partitioning can be dynamically altered as users join/leave the session. The group leader can explicitly specify and alter the partitioning.

The partitioned zones are assigned to the collaborating designers, and are colored differently for identification. Every user is responsible for filling the intermediate nodes in his zone by first positioning the models on the incoming edges, and subsequently using instances of the Geometric Design toolkit, SCULPT, to perform the actual set operation. This process is repeated till a satisfactory design is created. Figure 12 shows one site in the design of a simple windmill model. Figure 13 shows another site at the end of the operation.

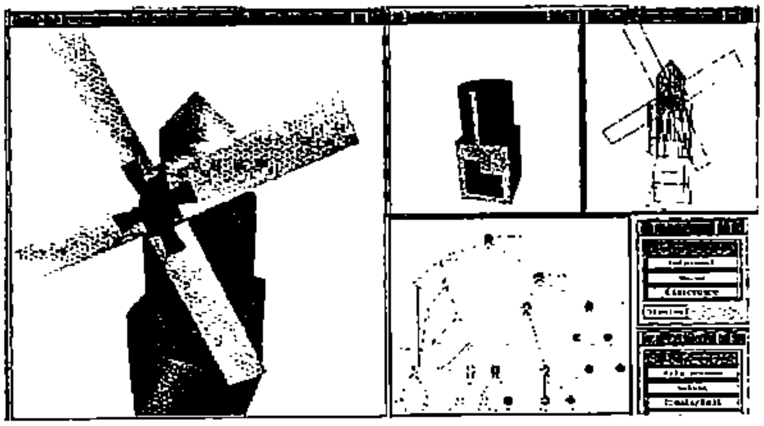

Figure 13: Another site, at the end of a Collaborative Design session

6.2.6 Collaborative Interaction

Interaction can occur in two modes. In the Regulated mode, the user responsible for a zone creates all the models internal to the zone. If the source nodes for an intermediate node are filled, a user locks the intermediate node by selecting it in the Graph Window. The session manager allows him access to the models in source nodes. The user interactively positions these models and performs the appropriate set operation, and the resulting model is assigned to the intermediate node, which is subsequently unlocked. All other users are denied access to the interior of a zone by the session manager. In the Unregulated mode, the partitioning merely suggests a minimal-conflict setting, and the session manager doesn't regulate interaction beyond what is specified by the permission settings for the site and the object. In this mode any user can access any node if he has Access and Modify permissions for the collaboration.

At the boundary of a partition, users of adjacent zones must agree about the models at the boundary nodes so that inconsistencies are not created in the design. A good group design protocol for this setting is to resolve boundary condition issues at the start of the operation, to prevent unnecessary cycles due to inconsistencies later on. This involves computing the subgraphs rooted at boundary nodes first, till satisfactory models at those nodes are obtained.

The session manager allows only one user to manipulate a "hot spot" in the graph– where there is a possibility of contention – at any particular instant. It uses the first-come-first-served paradigm to decide which user gets temporary exclusive control. The last completed operation specifies the model associated with the node.

The token passing facility of the system can be used to take turns to create boundary nodes. Alternately, designers can use the auxiliary communication channels (audio/video/text) by initiating SHA-PHONE, SHA-VIDEO or SHA-TALK sessions to regulate access, and to decide which users will set those nodes.

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. Changing a node requires all of its dependencies to be re-evaluated. The operation is completed when all the nodes of the design graph have been evaluated. Any site with Copy permission can then extract the model from the session and save it.

6.2.7 The Design Graph

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 – in some sense they know what the design objective is, and it 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. A collaborator who joins an ongoing collaboration late can quickly come up-to-date, and infer the status of the operation. He can use the stored sketches or images to figure out what is being designed, and participate actively in the collaboration.

A sufficiently detailed design graph of a model stored along with the leaf node objects is a compact and efficient way to represent the design. The model can be reconstructed automatically in the SHASTRA environment by SHILP.

6.3 Collaboration Modes

At one extreme, the SHASTRA implementation for Collaborative Smoothing can be used by a single designer to design a curved surface solid model much like in a non-collaborative setting. Allowing other users to join the session with only Access and Browse permissions sets up the environment like an electronic blackboard to teach novice users the basics of the smoothing mechanism. An appropriate setting of collaboration permissions and turn-taking can be used to allow hands on experience with the task. In conjunction with the audio and video communication services of SHASTRA, this becomes a powerful instructional environment.

In a different situation, for Collaborative Design, a group of n designers can set up a Regulated collaborative design session and collaborate to design an object. Each designer performs only the designated part of the shared design, and a speedup of as much as *problem size / maximum partition size*, can be achieved. Novice designers can join the ongoing session with only Access and Browse permissions, and get familiar with the group dynamics of a collaborative session.

In yet another situation, a group of n designers can start an Unregulated collaborative design session. Judicious use of the auxiliary communication facilities (Audio/Video/Text) to regulate design operations in a cooperative manner can let the team acquire a speedup factor of up to n.

6.4 Collaborative Chess

Another example of multi-user cooperative interaction is the collaborative Chess substrate, SHA-CHESS, which is described in detail in [2]. SHA-CHESS exploits the permissions mechanism of SHASTRA to support a variety of modes in which multiple users can interact over a virtual 3D chessboard – from tournament-like regulated conditions to an n-sided free-for-all, to a chess instruction tool.

If the user at a site has copy permission, he can acquire a copy of the current game, and initiate a chess playing program on a local board with the current game loaded, to determine a good move, as is depicted in Figure 14.

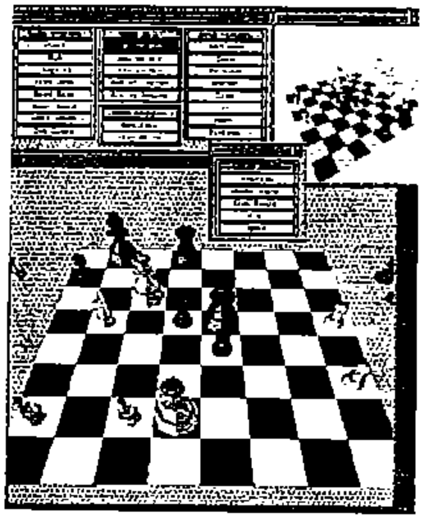

Figure 14: Exploiting the Copy permission in a SHA-CHESS collaboration

7 Issues

We are currently in the process of building an environment for Collaborative Design of Artificial Implants for the Human Body, using a solid modeling system, a medical image reconstruction system, a finite element analysis system and a computer aided manufacturing facility. This puts us in the realm of heterogeneous collaborations – which are supported on top of different applications, and we believe that the SHASTRA application architecture paradigm will alleviate many problems.

We are also studying synchronized audio-video communication facilities for incorporation in the system. A short term goal is to add an integrated collaborative multimedia browser and communication facility to the scientific manipulation environment, and to possibly link with a stable hypermedia system for archiving and reference of designs in SHASTRA to build a scientific information system.

We are adding functionality to the SHASTRA Kernels that will enable them to assign tasks to new instances of server systems, using runtime load analysis to distribute the computation to less busy machines (Load Balancing). SHASTRA can then function as a task-broker for distribution of scientific computations. We are also addressing issues of Session Manager state, which will make it possible to save session state and restart collaborations at a later point in time.

8 Conclusions

We have demonstrated that collaboration in the scientific design setting is facilitated by multimedia support as well as information sharing. SHASTRA is a powerful distributed and collaborative toolkit prototyping environment. It provides a rich substrate for design of such 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. The ease of integration of GANITH, SHILP, VAIDAK, BHAUTIK, SPLINEX and GATI into the SHASTRA environment demonstrates that the idea of an extensible collaborative environment for scientific manipulation is very viable.

Though a scientific manipulation environment has been the focus of our implementation, the facilities easily abstract out to a variety of situations requiring similar substrates. The collaborative layer is generic and can be used to implement the heart of systems for collaborative editing, code viewing and quality assurance tools, software development environments, multi-user electronics CAD, architecture CAD and mechanical CAD tools, and interactive multi-player games etc.

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., (1992), "Multimedia Communication Facilities in the SHASTRA Environment", *Computer Science Technical Report* [redacted], Computer Science Department, Purdue University.

[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] Anupam, V., (1991), "A Survey of Support Mechanisms for Collaboration in the Context of Geometric Design", *Unpublished*, Computer Science Department, Purdue University.

[5] 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.

[6] 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.

[7] Bailey, B., Bajaj, C., and Fields, M., (1991), "The VAIDAK medical imaging and model reconstruction toolkit", *CAPO Technical Report 91-31*, Computer Science Department, Purdue University.

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

[9] 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. Also as a CAPO Technical Report 91-30, Computer Science Department, Purdue University.

[10] Bajaj, C., Ihm, I., (1992), "Algebraic Surface Design with Hermite Interpolation", *ACM Transactions on Graphics*, 11:1, Jan. 1992, pp.61-91.

[11] Bajaj, C., Cutchin, S., (1992), "Interactive Animation using GATI", *Manuscript*, Computer Sciences Department, Purdue University, Dec. 1992.

[12] 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.

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

[14] Bajaj, C., Schikore, D., (1993), "Distributed Design of Hip Prosthesis Using BHAUTIK", *Proceedings of the 1993 Symposium on Applied Computing*, Feb. 1993, (to appear).

[15] Brothers, L., Sembugamoorthy, V., Muller, M., (1990), "ICICLE: Groupware For Code Inspection", *Proc. ACM CSCW 90*, 169-181.

[16] Crampton, C., (1987), "MUSK – a Multi-User Sketch Program", *Proceedings of the European UNIX systems User Group*, 17-29.

[17] Chamberlin, D., Goldfarb, C. (1987), "Graphic Applications of the standard generalized markup language (SGML)", *Computer Graphics 11, 4*, 1987.

[18] Cockroft, G., Hourvitz, L., "NeXTstep: Putting JPEG to Multiple Uses", *Communications of the ACM*, ACM, April 1991, pp. 44.

[19] Carlbom, I., Hsu, W., Klinkner, G., Szeliski, R., Waters, K., Doyle, M., Gettys, J., Harris, K., Levergood, T., Palmer, R., Palmer, L., Picart, M., Terzopoulos, D., Tonnesen, D., Vannier, M., Wallace G. (1992), "Modeling and Analysis of Empirical Data in Collaborative Environments", *Communications of the ACM*, ACM, June 1992, pp. 73-84.

[20] 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.

[21] Cognitive Science and Machine Intelligence Lab, U. of Michigan, Ann Arbor, (1990), "ShrEdit 1.0: A Shared Editor for the Apple Macintosh, User's Guide and Technical Description", 1990.

[22] Ellis, C., Gibbs, S., Rein, S., (1988), "Design and Use of a Group Editor", *Report STP-263-88*, MCC Software Technology Program.

[23] 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.

[24] Fish, R., Kraut, R., Leland, M., Cohen, M., (1988), "Quilt: A Collaborative Tool for Cooperative Writing", *Proc. ACM SIGOIS Conference*, 30-37.

[25] Gibbs, S., (1988), "LIZA : An Extensible Groupware Toolkit", *Report STP-042-88, MCC Software Technology Program.*

[26] Gall, D. (1991), "MPEG: A Video Compression Standard for Multimedia Applications", *Communications of the ACM*, ACM, April 1991, pp. 46-58.

[27] Green, J., (1992), "The Evolution of DVI System Software", *Communications of the ACM*, ACM, Jan. 1992, pp.52-67.

[28] Haan, B., Kahn, P., Riley, V., Coombs, J., Meyrowitz, N., (1992), "IRIS Hypermedia Services", *Communications of the ACM*, ACM, Jan. 1992, pp.36-51.

[29] Harney, K., Keith, M., Lavelle, G., Ryan, L., Stark, D. (1991), " The i750 Video Processor: A Total Multimedia Solution", *Communications of the ACM*, ACM, April 1991, pp. 64-78.

[30] 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.

[31] Knister, M., Prakash, A., (1990), "DistEdit: A Distributed Toolkit for Supporting Multiple Group Editors", *Proc. ACM CSCW 90*, 343-355.

[32] Lee, J., (1990), "Xsketch : A Multi-user Sketching Tool for X11", *ACM Office Information Systems*, 169-173.

[33] Liou, M. (1991), "Overview of the px64 kbit/s Video Coding Standard", *Communications of the ACM*, ACM, April 1991, pp.59-63.

[34] 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.

[35] Mercurio,P. , Elvine, T., Young, S., Cohen, P., Fall, K., Ellisman, M. (1992), "The Distributed Laboratory", *Communications of the ACM*, ACM, June 1992, pp. 54-63.

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

[37] Newcomb, S., Kipp, N., newcomb, V. (1991), "Hytime: Hypermedia/Time-based Document Structuring Language", *Communications of the ACM*, ACM, Nov. 1991., pp. 67-83.

[38] Nielsen, J. (1990), "The Art of Navigating through Hypertext" *Communications of the ACM*, ACM, March 1990, pp. 296-310.

[39] Nises, P., Wettby, J., (1990), "Phonetalk", *Tech. Report, Royal Institute of Technology*, Dec. 90.

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

[41] Riedl, J., Mashayekhi, V., (1991), "Continuous Media in Discrete Objects: Support for Collaborative Multimedia", *CS Dept. University of Minnesota*, Sept. 1991.

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

[43] 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.

[44] van Dam, A., 1987, "Hypertext'87 keynote address", *Communications of the ACM*, ACM, July 1988, pp. 887-895.

[45] Wallace, G. (1991), "The JPEG Still Picture Compression Standard", *Communications of the ACM*, ACM, April 1991, pp. 30-44.