Distributed Collaboration for Engineering and Scientific Applications Implemented in Habanero, a Java-Based Environment

Brent Driggers, The University of Kansas
Jay Alameda, National Center for Supercomputing Applications
Ken Bishop, Ph.D. The University of Kansas

Introduction:

Contemporary practice of engineering, science, and education places an ever-increasing emphasis on the temporary assembly of expertise that may be widely distributed geographically; intense collaborative application of that expertise to a technical challenge; and efficient dissemination of the results of the collaboration. A famous example of this emerging paradigm can be found in Apple Computer's "Knowledge Navigator," a twenty year old "futures piece," that showed geographically distributed expertise, "late breaking" literature information, and "current" experimental data being assembled in order to create an exciting educational experience – a classroom without walls.

The motivation for the current project has its roots in a related and equally exciting activity. The National Computational Science Alliance has identified a group of scientists and engineers to work on computationally challenging chemical engineering applications. The members of that team are located in Illinois, Massachusetts, Kansas, and South Dakota. Individually, they offer special expertise (*ab initio* physical property estimation, chemical reactor design, chemical process simulation, numerical methods, and parallel processing algorithms), lead diverse professional lives, and are committed to advancing the art and technology of chemical engineering design. Clearly, a research base without walls is necessary for the logistical support of such a group.

Although the concept of a collaborative session among peers, wherein computational tools and information may be shared freely is a familiar one, it has been implemented recently as a software product named Habanero (NCSA, 1996). The Software Development Group at the National Center for Supercomputing Applications is developing Habanero, an environment that supports collaborative Java applications. The Kansas Engineering and Scientific Interface (KESI) is a prototypical Java application intended to identify requirements for, and suggest tools that support distributed collaboration. In particular, KESI is focused on the issues associated with managing execution of third party codes written in FORTRAN (or another non-Java language), running on a high performance computer as an integral part of the collaborative session.

The Engineering Application:

Chemical intermediates, the building blocks for specialty chemicals, are frequently made from naturally occurring chemical species by catalyzed, partial oxidation reactions. These reactions are commonly carried out by passing the reactants, in the vapor phase, through a bed of solid catalyst particles. The heat generated by these exothermic chemical reactions is removed by conduction through the catalyst bed and convection from the reactor wall to a high temperature heat transfer medium, such as molten salt. The shell and tube heat exchanger configuration, shown in Figure 1, is common for such a reactor.

Temperature control of these reactors can be a delicate matter, depending upon a balance between the flow rate of gas through the bed, the temperature and composition of the feed, and the flow rate of the molten salt. If too much heat is removed then the reaction is quenched and production ceases; if too little heat is removed then a positive feedback relationship that exists between reactor bed temperature and chemical reaction rate is causes the reactor to "run away", ultimately destroying the reactor. A variation of as little as five degrees centigrade in the feed temperature can cause reactor operation to swing between the limits described! The combination of the economic potential and the safety challenges presented by this technology justifies a significant investment in the simulation of transient reactor performance in support of reactor design and control activities.

The specific reactor simulated in this project is a shell and tube configuration wherein the tubes are packed with a solid granular catalyst. The reaction studied is the non-isothermal, vapor phase, partial oxidation of ortho-xylene to form phthalic anhydride. The feed to the reactor is a very dilute mixture of ortho-xylene and air. The reactor operates at a pressure of about two atmospheres and a temperature of approximately 600 degrees Kelvin.

The mathematical model used in the simulator involves a set of simultaneous, non-linear, partial differential equations corresponding to the material and energy balances that describe the transient behavior of the reactor in two space coordinates (radial and axial position). The equations programmed in the simulator (Eggeman, 1989) are the set of simultaneous, non-linear, ordinary differential equations that result from replacing the spatial derivatives with finite differences. The equations in this “method of lines” algorithm are solved using a standard solver, LSODE, that is adept at solving stiff differential equations. The code is written in FORTRAN.

Resolution in the spatial field of interest for this problem is somewhat coarse (nine radial, thirty-three axial nodes) so as to permit use of a powerful workstation (SGI Indy) for this prototype development project rather than the multi-processor supercomputer (SGI Power Challenge Array) that would normally be used. Even so, a transient simulation run, one hundred seconds long, produces nearly thirty thousand computed results. The results of a simulation run are post-processed for display as a set of continuous hue, false color, iso-potential plots that represent the temperature and composition of three chemical species as a function of position in the reactor. The eight-bit color display used is adequate for analyzing and following trends in reactor performance; the tabular output produced directly from the simulator is available for resolving anomalous results.

An example of a portion of the simulator’s output is presented in Figure 2. Figure 2 shows the false color rendition of the temperature in the reactor bed at five different times following an increase in the feed temperature. The reactants flow from left to right through the reactor. The top of each frame corresponds to the centerline of the reactor; the bottom corresponds to the inner radius of the tube that holds the bed. A blue color indicates a low temperature; red indicates a high temperature. The increase in the size and intensity of red in the area located approximately twenty percent of the distance from reactor feed to effluent indicates formation of a “hot spot” (and its migration toward the feed port).

<u>Habanero</u>:

Habanero is a framework for sharing data and specific events among colleagues who are distributed across the Internet, possibly in remote locations. Sharing of these data and events allows the same material to be displayed on every computer in the session, thereby providing a synchronous collaborative work environment. Currently, Habanero is equipped with collaborative general tools that include communication, science and engineering, and entertainment applications.

Habanero provides the means to synchronize the states of every computer in a session. In order to determine the state of a remote computer, Habanero uses the idea of wrapping or “marshalling” the present state of that computer. Information is sent to collaborating computers using methods coded specifically to write and read the information. The class that has a requirement for sending information to every participant in the session must implement the Wrapped or Marshallable interface.

The Wrapped or Marshallable interfaces in Habanero have marshallSelf and unmarshallSelf methods that have been declared abstract. Thus, the marshallSelf and unmarshallSelf methods must be overridden in the class that sends information if that class is not going to be abstract. For example, if three variables were to be shared by all the computers in a session then a class necessarily would have a marshallSelf method that would write the variables. The unmarshallSelf method would read the variables in the same order as written in the marshallSelf method.

A simple example of a class that contains the marshallSelf and unmarshallSelf methods follows:

```
import ncsa.habanero.*;
import ncsa.habanero.streams.*;
import java.awt.*;

public class Habanero_Example extends Panel implements Wrapped
{
        public Habanero_Example()
        {
                …..
        }
        … Code that reads or calculates values for variables …

        public void marshallSelf(MarshallOutput out) throws IOException
        {
                out.writeUTF(var1);
                out.writeUTF(var2);
                out.writeUTF(var3);
        }

        public void unmarshallSelf(MarshallInput in) throws IOException
        {
                var1 = in.readUTF();
                var2 = in.readUTF();
                var3 = in.readUTF();
        }
}
```

The ncsa.habanero and ncsa.habanero.streams packages must be imported into the code above because they contain the Habanero classes that are used. The MarshallOutput interface, in turn, inherits from the DataOutput interface that is included in the java.io package. The writeUTF() method is a method in the DataOutput interface and uses the Unicode Text Format (UTF) to write the variables. The MarshallInput interface inherits from the DataInput interface (also included in the java.io package). The readUTF method reads the variables that were written by the writeUTF method.

Variables that were assigned values through a graphical interface, or a computation, on a participant's computer are shared with all of the computers in the session. The marshallSelf and unmarshallSelf methods are called when a new participant joins a collaborative session or when information is distributed to other participants. These methods normally are not explicitly called in the application's code, but rather are called according to the requirements of the Habanero framework.

Habanero allows a user to create a session or to join an existing session. In the Habanero User Environment frame, presented in Figure 3, a user can click on the create button to create a session. If a user desires to join a session that has already been created then the user must enter the Session name, Server Address (the address of the user that created the session), and the Server Port Number that was specified when the session was created.

After the session has been created, Habanero's collaborative tools are displayed in the Session Control frame. Chat is a communication application that was developed at the National Center for Supercomputing Applications (NCSA). The Chat tool allows the sharing of text messages with the other participants in a session by using the Habanero object sharing mechanism rather than IRC. Habanero provides science and engineering tools including the Kansas Engineering and Science Interface (KESI), the Java Virtual Wind Tunnel, and several others. KESI is being developed at The University of Kansas to enable projects in collaborative reactor design. David Oh of the Computational Aerospace Sciences Laboratory, MIT, wrote the Java Virtual Wind Tunnel application, which uses computational fluid dynamics (CFD) methods to simulate the flow of air around two-dimensional bodies. The solution to the Euler equation is displayed as it is produced rather than as an animation. Habanero also includes entertainment and communication prototypes such as a game of checkers that enables two participants to compete over the Internet.

<u>The Kansas Engineering and Science Interface</u>:

The Kansas Engineering and Science Interface (KESI) uses a combination of Java and Habanero capabilities to enable session participants to plan, execute, and display the results from a simulation of the reactor's transient performance. These functions are implemented using button controls on the KESI frame. Figure 4 presents a view of the KESI frame in which the planning capability is exercised and a single frame taken from the animation file.

Planning for a simulation is done in KESI by establishing the operating conditions for the simulation, using a combination of button and text entry field inputs via the graphical user interface. As a further aid to collaborative planning, KESI allows consideration of multiple suggestions for operating conditions from the participants followed by a group decision procedure that resembles a vote.

Each participant may enter proposed values for the feed temperature, pressure, and the display time step as well as other pertinent information. Once submitted, the participant's candidate set of operating conditions is wrapped, sent to all computers in the session, and assigned to an object that is saved in Java's Vector construct. After all of the participants have either submitted conditions or passed on the opportunity, all sets are shown in a new frame on each computer. The participants are then prompted to select their preference from the summary frame. The participants choose by selecting a button corresponding to a participant's submission. The vote is processed and the results are displayed, both as a count and as a three-dimensional pie chart. The session controller, acting with veto power in reserve, sends the operating conditions to the compute server by making the final selection from all of the submissions. The participants can also use Habanero's chat box in order to support group discussions.

Once the operating conditions for the simulation have been transmitted to the compute server, the simulation calculation is initiated by selection of an option on the KESI menu bar. Selection of the simulation option causes the Java program to make a call to the underlying operating system of the session controller's computer using the Runtime class in the java.lang package. The call to the operating system runs a script that remotely triggers a script on the compute server to initiate the simulation. Upon completion of the simulation run, a dialogue box indicating simulation completion is sent to all participants in the collaboration.

The script on the compute server also triggers post-processing of the simulation output to prepare it for viewing and moves the output files to a specific directory that is accessible by all the participants through http. The participants can choose to review the temperature and concentration of each species. Each participant's computer is given the command to retrieve the set of files from the URL of the compute server. The color bit maps are assigned to the Java Vector construct and are animated asynchronously with respect to time on each computer.

In addition to providing information on the entire transient response of the reactor by means of the animation capability, KESI also has the ability to focus on a single frame from the animation file. This permits detailed study of minor trends and/or anomalies in the computed results.

The collaborative nature of the session is emphasized by the inclusion, in KESI, of the ability for participants to request control of the session by becoming the session manger. The current session manager may accede to or refuse the request.

<u>A Collaborative Session</u>:

A portion of the activities carried out in a collaborative session on reactor operating condition selection is presented in Figure 5. The authors were the participants in the session. Brent Driggers used an SGI Indy located at the University of Kansas, Ken Bishop used an Intel Pentium-based Windows NT machine, located in his office at the University. Jay Alameda used an Intel Pentium Pro-based Windows NT machine, located at the National Center for Supercomputing Applications at the University of Illinois. The activities that occurred during the example session were triggered by an arbitrary set of Habanero and KESI commands issued by Brent, who acted as session controller. The screen display is common for all participants and is synchronized in order to provide collaborative cues. The frame in the upper left portion of the screen, labeled "Simulation" was displayed as the computation was completed. The KESI frame contains the dimensionless o-xylene composition as a function of dimensionless position in the reactor at a particular instant (sixth of ten produced in the simulation). The frame located in the lower left quadrant of the screen, labeled "Opvar In" lists the parameters associated with the simulation run. The second KESI frame shown in the lower right quadrant shows a diagram of the subject reactor. The chat box shown in the upper right quadrant contains a discussion among the participants on the merits of the particular set of operating conditions chosen.

The chat box capability serves well as a documenting and archiving tools but is less than satisfactory as a communication device. The developers of Habanero are working on an "audio chat" function to augment and improve communications among collaborators. We had found that we needed to supplement Habanero's communications schemes with a telephone call, so the audio chat function will be a welcome addition.

Conclusions:

The Distributed Collaboration project was undertaken for the purposes of 1.) Identifying the requirements and coding issues associated with managing a reasonably sized third party code using a contemporary Java based environment and 2.) Suggesting prototypical tools that support such distributed collaboration. Not only do the issues appear surmountable and the tools appear both powerful and convenient, but the Kansas Engineering and Science Interface project has enjoyed a uniformly enthusiastic reception from the engineers and scientists who have seen preliminary demonstrations.

Although it must be considered a work in progress, the KESI simulation management tool provides great convenience and utility. The manually entered catechisms that traditionally accompanied simulation planning, execution, post-processing and output display have been eliminated in favor of graphical interfaces and UNIX scripts. Specific details are entered using buttons and text entry field boxes. KESI is rapidly becoming our preferred simulation management interface, regardless of the collaborative requirements of a particular simulation.

The Habanero environment is also a maturing product. Even so, its ability to seamlessly manage small groups of participants in a collaborative session is excellent. Its use with the large groups (classes) has yet to be tested. The power of Habanero lies, not in the tools that are available, but rather in the fact that virtually any tool can be made collaborative with only moderate changes. In our experience, the only real hardware dependency involves fielding a consistent JDK for the entire suite of computers being used.

Development of the Kansas Engineering and Science Interface is continuing at a rapid pace. In addition to its use in the reactor simulation management application described here, analyses that will lead to applications in the areas of groundwater supply management and the simulation of the transient behavior of fractured, tight, natural gas reservoirs is underway. Unless or until some truly significant and unexpected barrier is encountered, the future for Habanero and KESI appears to be full of promise. The single caveat is that until the tool set offers a rich selection of very easy to use elements which provide for efficient communication of text, graphics, information analyses, and flexible freeform drawings as well as for efficient archiving, documentation, and communications, personal contacts will be difficult to replace completely. A great deal of progress has been made; a great deal more is necessary.

References:

1. Eggeman, T. J., Snyder, J. D., Subramaniam, B. and K. A. Bishop, "Color Raster Display- A New Tool for Reactor Design," Annual Meeting of the American Institute of Chemical Engineers, November, 1989.

2. NCSA, http://www.ncsa.uiuc.edu/SDG/Software/Habanero/ , 1996

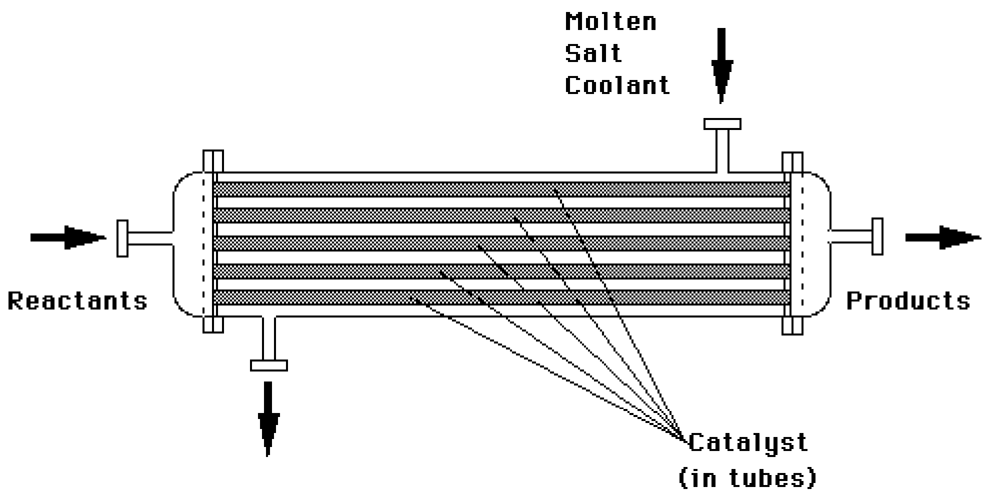

Figure 1. Schematic Diagram of a Fixed Bed Catalytic Reactor

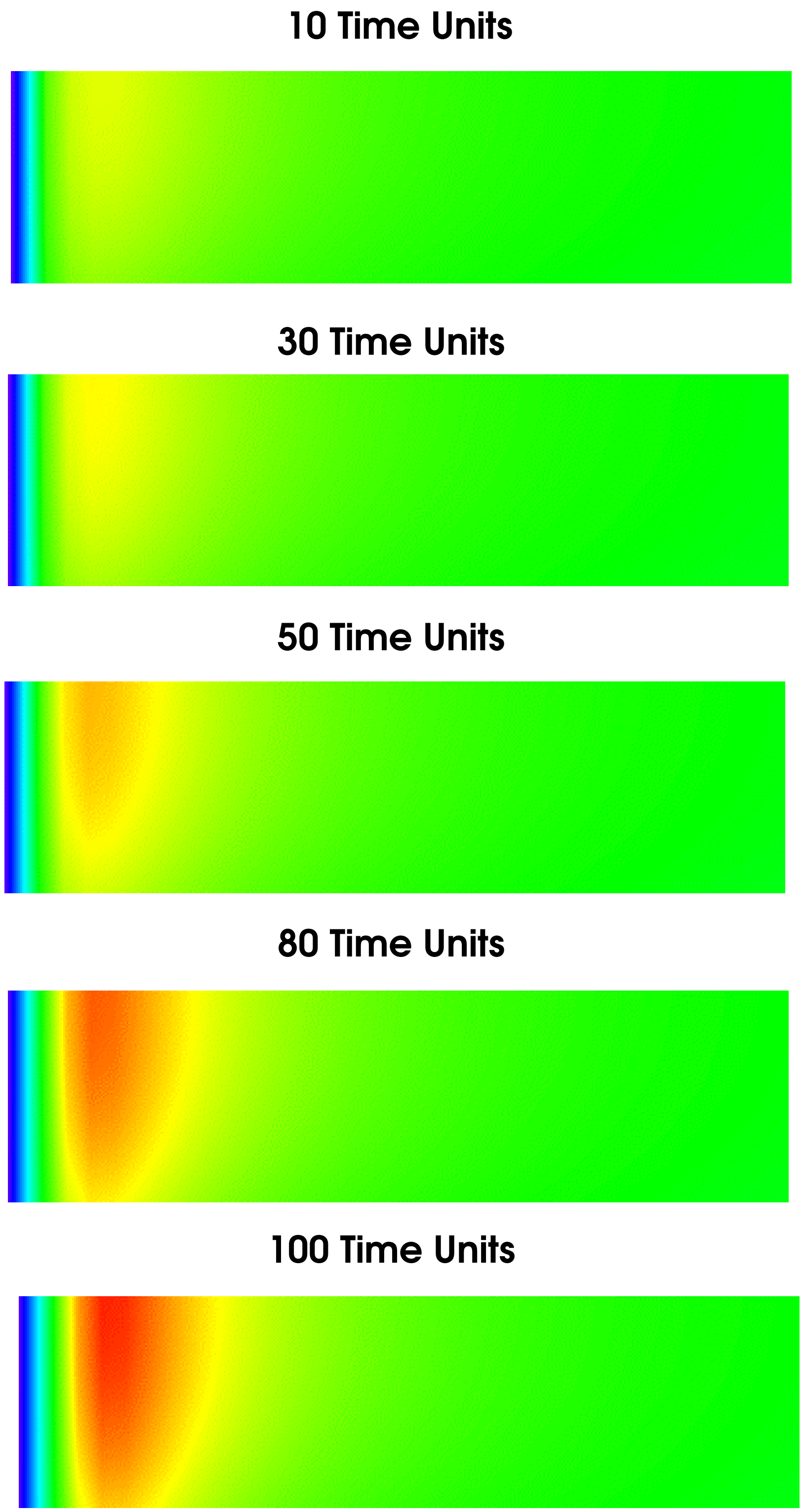

Figure 2. Display of Transient Temperature

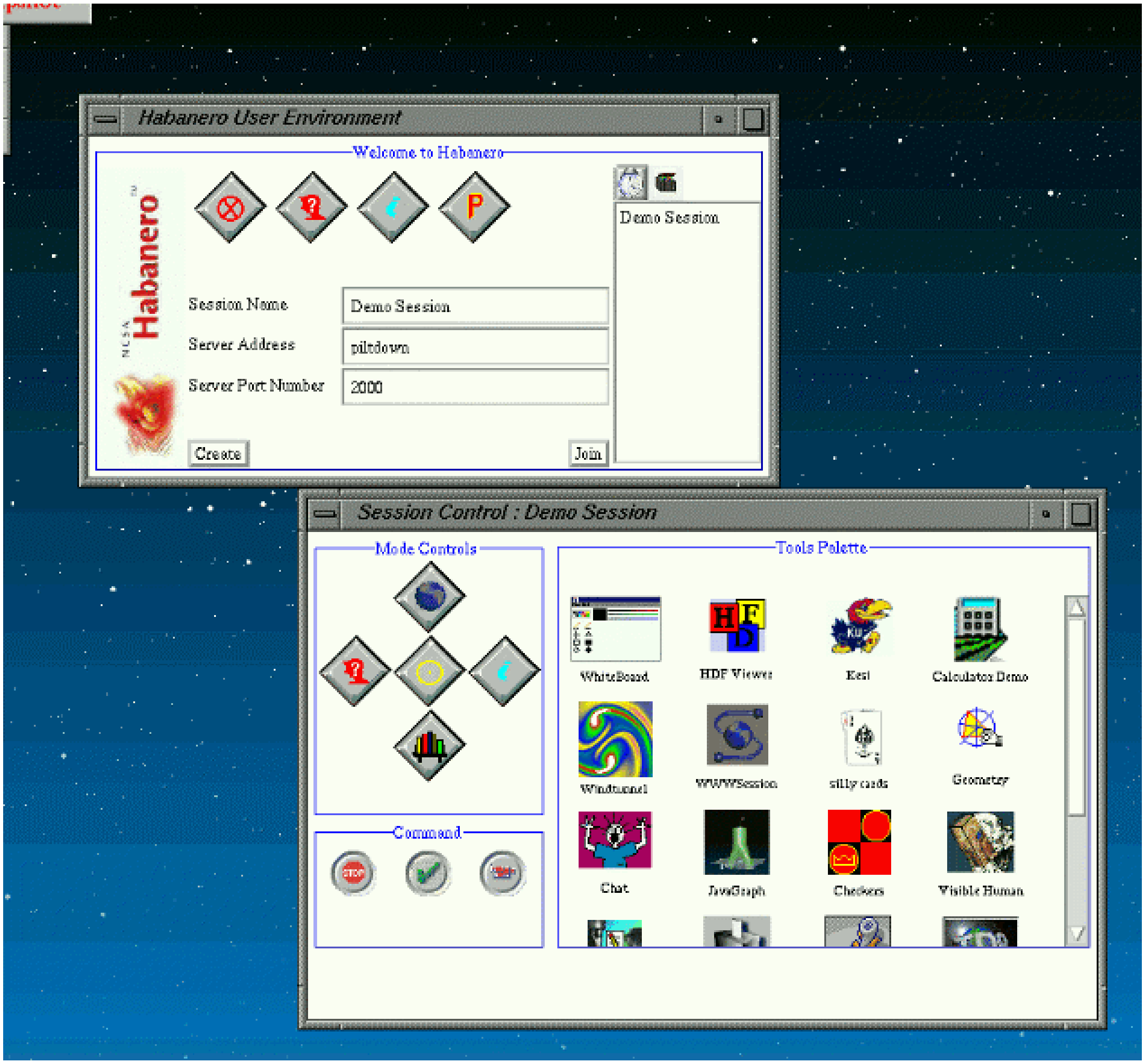

Figure 3. The Habanero Interface

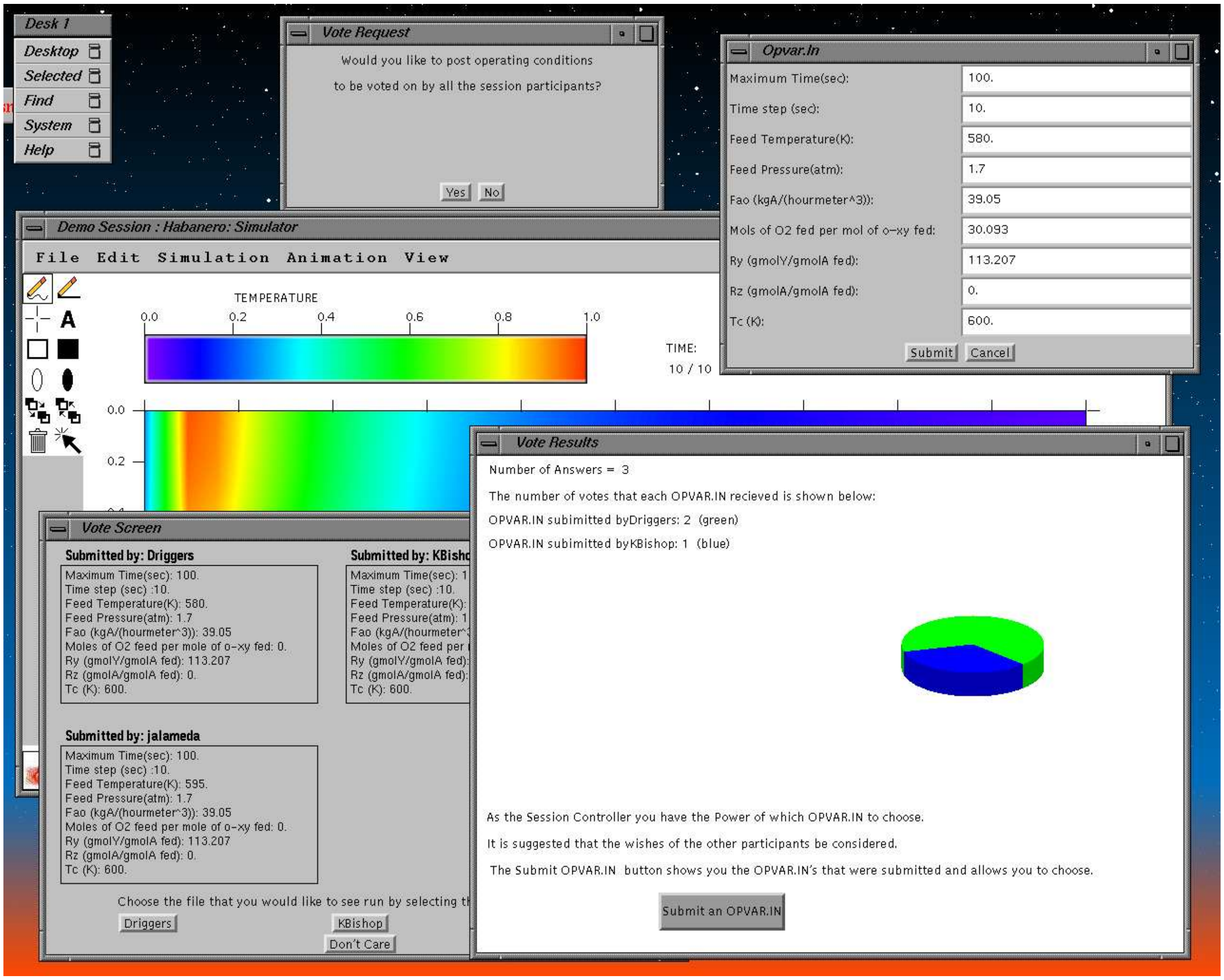

Figure 4. An illustration of the KESI voting tool.

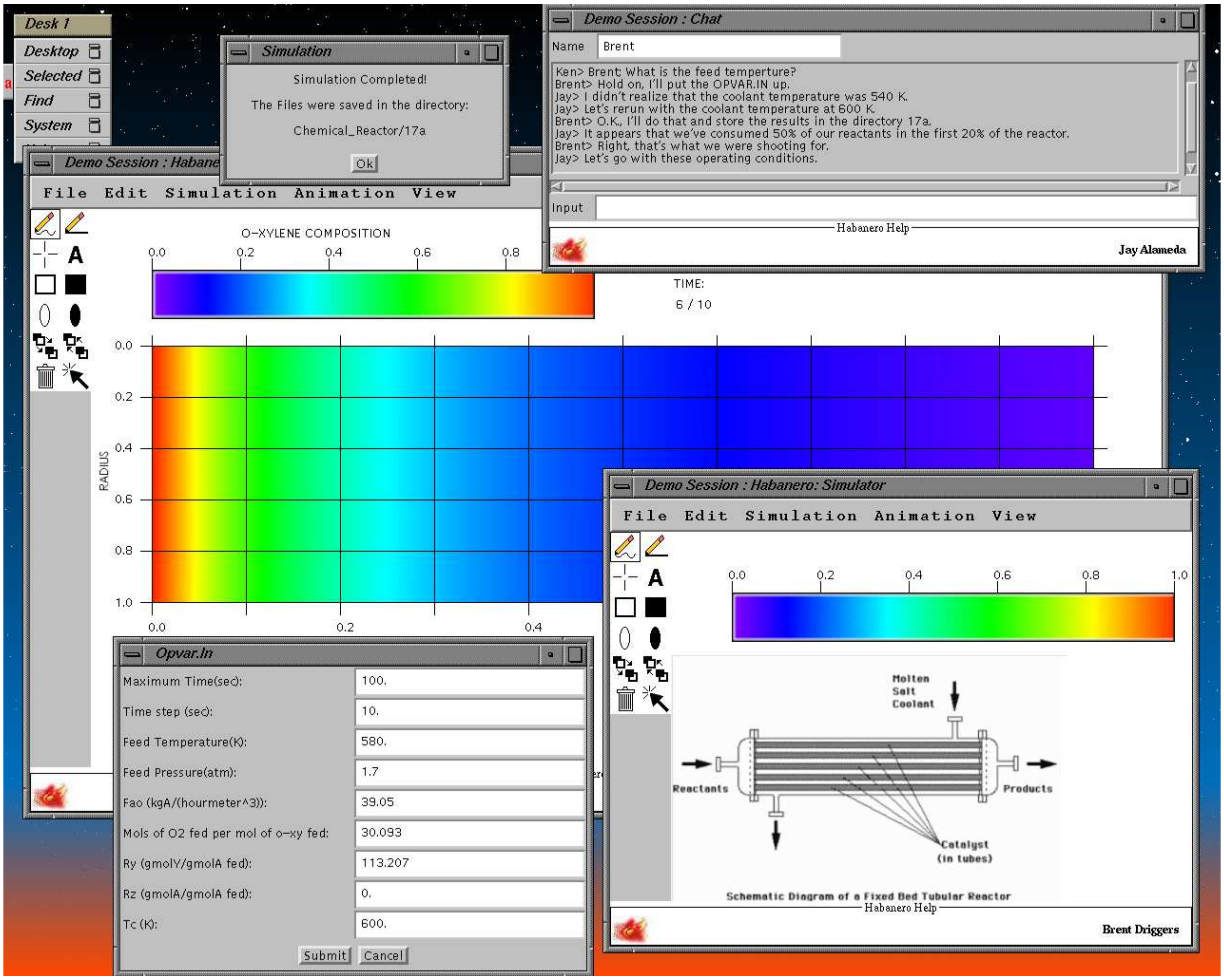

Figure 5. An illustration of a collaborative KESI session.