## COMMUNICATING WITH APPLICATIONS

The applet is a script that uses a chat component. However, since such a chat component is not a standard component in every browser, we modified our browser to setup communication with other applications so it can delegate commands for unknown components. An alternative would be to compile such components into the browser but that would lead to a large monolithic browser what we consider as rather inflexible.

Instead, our browser communicates with remote applications to delegate functionality. In this paper we will use the term remote applications for applications that are running on some host (possibly other than the local host) with which a browser can communicate.

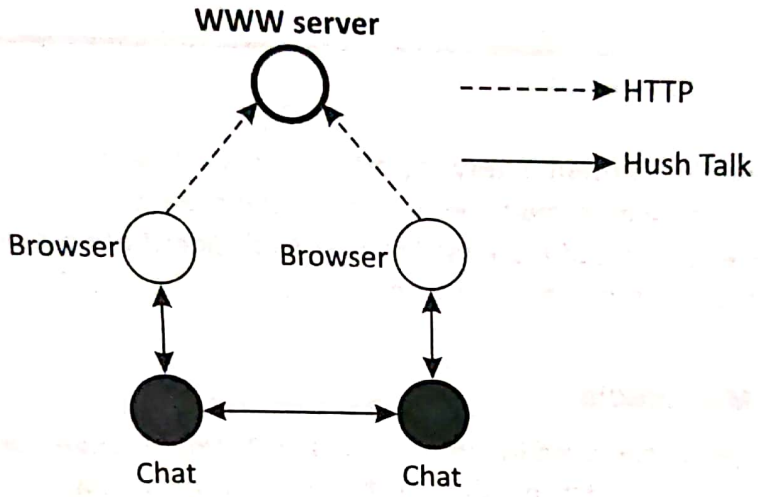


**Fig. 4. After Requesting the chat page it looks**

In our Hush browser, a connection with a remote application is set-up after the page is loaded and the browser only serves as a viewer for the interface of the remote application.

The WWW server has no active role except for initiation, see fig. If two users have requested the chat page, two chat applications will be running and the interface will be visible embedded in an HTML page. This example illustrates the use of new functionality that the browser nor the WWW server itself support. The WWW is used only as a general access-point for using applications on the network and to have the web browser act as a viewer for the application interface.

### Hush Talk — communication on top of HTTP

For communication between a browser and a remote application , an additional communication protocol is needed. Such a protocol should at least have the following properties ; it should be fast, have two way communication with flexible message passing and some form of session management.

The first two properties are needed for efficiency reasons and to keep up the performance of the system. The last property is important for cooperative work so applications that participate in cooperative work can reach and be aware of each other throughout a session. The stateless HTTP protocol is not suited for this kind of continuous communication, nor does it offer any session management. Therefore we developed a new communication protocol called Hush Talk, currently based on Sun's Tool Talk.

Tool Talk is a message based system for inter-application communication which is available for UNIX platforms as part of the standard Solaris 2.x distribution. The actual communication in Tool Talk is done by RPCs which make it faster than the HTTP protocol since connections are not opened and closed all the time. It also provides the notion of a session that manages connected applications and message delivery. Choosing another protocol than HTTP also has the advantage that the HTTP server load is not increased.

With Hush Talk we tried to handle communication in an object-oriented style *i.e.,* by calling handler objects with events rather than calling callback functions. In our example we use Hush Talk to set up communication between chat applications and for the communication between the browser and remote application.