XBoard: A Framework for Integrating and Enhancing Collaborative Work Practices

Ted Shab
NASA Ames Research Center/QSS
tshab@mail.arc.nasa.gov

Abstract

Teams typically collaborate in different modes including face-to-face meetings, meetings that are synchronous (i.e. require parties to participate at the same time) but distributed geographically, and meetings involving asynchronously working on common tasks at different times. The XBoard platform was designed to create an integrated environment for creating applications that enhance collaborative work practices. Specifically, it takes large, touch-screen enabled displays as the starting point for enhancing face-to-face meetings by providing common facilities such as whiteboarding/electronic flipcharts, laptop projection, web access, screen capture and content distribution. These capabilities are built upon by making these functions inherently distributed by allowing these sessions to be easily connected between two or more systems at different locations. Finally, an information repository is integrated into the functionality to provide facilities for work practices that involve work being done at different times, such as reports that span different shifts.

The XBoard is designed to be extendible allowing customization of both the general functionality and by adding new functionality to the core facilities by means of a plugin architecture. This, in essence, makes it a collaborative framework for extending or integrating work practices for different mission scenarios. XBoard relies heavily on standards such as Web Services and SVG, and is built using predominately Java and well-known open-source products such as Apache and Postgres.

Increasingly, organizations are geographically dispersed, and rely on "virtual teams" that are assembled from a pool of various partner organizations. These organizations often have different infrastructures of applications and workflows. The XBoard has been designed to be a good partner in these situations, providing the flexibility to integrate with typical legacy applications while providing a standards-based infrastructure that is readily accepted by most organizations.

The XBoard has been used on the Mars Exploration Rovers mission at JPL, and is currently being used or considered for use in pilot projects at Johnson Space Center (JSC) Mission Control, the University of Arizona Lunar and Planetary Laboratory (Phoenix Mars Lander), and MBARI (Monterey Bay Aquarium Research Institute).

1. Introduction

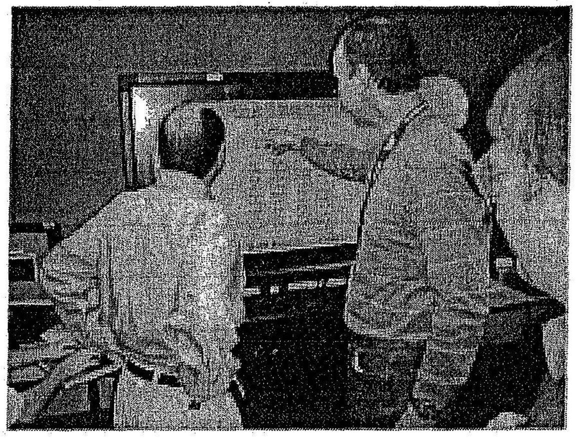

Figure 1 Sol Tree tool use at MER

The XBoard began as the result of ethnographic observations made during the 2001 FIDO Test at JPL. Many ad-hoc collaborative activities were found to be unsupported in the mission suite of tools. Although none of these were mission critical, solving them seemed to useful, especially in two main areas, enhancing the ability for teams to collaborate about science planning, and enhancing collaboration to support engineers in Tiger Team work in the event of unforeseen engineering issues.

Additionally, the MER Mission was slated to have two mostly discrete teams working in several buildings. Even in the primary building, the work was scattered across different floors and rooms.

A decision was made to utilize large, touch-screen plasma devices for several reasons. First, these would prove to be ideal for the small science team collaboration, as they provided the benefits of a projected with added interaction.

Specific work practices were observed that were non-optimal, and requirements were generated for creating tools to address these items. The general areas of work we looked into supporting included group collaboration, specifically in the context of the different science team groups (i.e. Geology, Atmospheric, etc.). Additionally, because the work spanned different rooms and floors in the facility (and sometimes different buildings), we took on creating functionality that would allow the distributed collaboration functionality to be integrated into a generalized mechanism that could be used by all the tools created for this application.

Finally, because MER consisted of two missions, running on diametrically opposed shifts due the rovers placement on Mars, information dissemination back and forth between the two missions was expected to be somewhat complicated. People were expected to work in shifts consisting of four days on, three days off, tracking Martian time, which changed daily. We wanted to provide repositories where people could put non-critical information that would still be useful to the teams involved (i.e. documents, notes, images), in a consistent manner.

These modes of collaboration turned out to be generalizable, as organizations increasing becoming geographically dispersed, and rely on "virtual teams" that are assembled from a pool of various partner organizations. These organizations often have different infrastructures of applications and workflows. The XBoard was designed to be a good partner in these situations, providing the flexibility to integrate with typical legacy applications while providing a standards-based infrastructure that is readily accepted by most organizations.

The XBoard design is based on a human centered computing (HCC) methodology. Requirements and design definition are based interaction with, and observation of, users. The XBoard is viewed as part of a total work system involving people, processes, procedures and tools. The different software plugins are designed to provide an integrated experience to the user, allowing them to concentrate on the actual work being performed rather than ancillary details such as IP addresses or designed file share location.

The general types of work practices supported include:

Support for co-located collaboration for groups of people working in close proximity to each other.

Support for synchronous group collaboration for groups in related work areas, such as planners at different centers working on the same plan

Support for synchronous collaboration in rooms up to the size of a conference or control room.

Support for asynchronous collaboration for Information Consistency, shift handover, Communication, and Tiger Teams.

2. Integrated Environment

One of the key features of the XBoard was that the experience would be simple and integrated without requiring a lot of training. The phrase "Palm Pilot" simple was the rallying call. In essence, many of the work that was done at the board, for example whiteboarding, could be done with paper-based systems. In fact, there were commercial products that would allow remote based collaboration with electronic whiteboarding software. Finally, things like shared drives could be used to provide much of the functionality of our shared "XSpace" ubiquitous file system.

However, our system differed in having this presented to a user in a cohesive manner so as not to interfere with the work practice at hand. Whiteboards could be authored, and with the same mechanism that all applications use on the XBoard, collaboratively shared with other users. The results of this work will be saved in an integrated manner to a common location that is self-obvious, reducing cognitive overload.

We leveraged this kind of integration in other areas. For example, we had XBoard-wide screen capture, which meant information could be grabbed and exported to the whiteboard from any of the integrated applications (including the web browser, images from the data product explorer, other systems being remotely shared, etc.), then from the same integrated mechanism, distributed to other users using Email or the shared filespace.

3. COTS Alternatives

XBoard is designed to run on COTS (commercial, off-the shelf) hardware. However, COTS software alternatives existed for bits and pieces of the functionality, but always with caveats. First, licensing was often tricky and the time and effort required to deal with investigating, finalizing and tracking the licensing was often greater than the product costs themselves.

COTS products typically did not allow the level of integration we provided (as discussed above), or the level of customizability. Creating security that met the standards of the mission but still provided the flexibility and applicability that best corresponded to the applications used required significant tailoring of our software.

Our system is designed to be very user friendly and to provide an environment for accomplishing the tasks with as little cognitive overhead for the mechanism as possible. A simple example of this is that to connect between computers did not require knowledge of IP addresses and the authentication was provided for several kinds of access control.

Finally, our system integrated the collaborative functionality with a whole class of software, including Java applications, various web-based apps, local C++ applications and virtualized software running on a server. This flexibility allowed us to rapidly bring the functionality of our system to other applications.

4. Multi-user applications

One of the features of the XBoards that was prevalent on the Mars Exploration Rover was the user of large, plasma displays with touch-screens. This enabled the use of the board in a true concurrent multi-user fashion. Online, role-playing games enable users to be dealing with events both autonomously and collaboratively in parallel. Most PC applications, however, require serial usage, even when multiple users are collaborating on something. Examples of this are the baton used in some distributed, multi-user whiteboard applications, which passes control of the application. Often times this is just done through work practice, such as how multiple users would concurrently edit SAP information on MER, with a roomful of people viewing SAP on a large projector, discussing, deciding and then passing on the specifics of that decision to a single SAP operator. MERBoard had an application called Sol Tree Tool, which enabled Strategic Planners to describe and consider high-level, multi-sol scenarios using a graphical interface [Huang, 2006]. Although the work here was still done in a serial fashion, there wasn't the strongly enforced sense of a single application "driver".

This is an area where future research may create mechanisms that more naturally allow fully parallel application interaction.

5. Extensibility through a plug-in architecture

XBoard became seen as a place to put applications that could then take advantage of the collaborative capabilities that we were providing. We came up the concept of a plug-in architecture that would allow for existing applications to be integrated into the collaborative environment, or new applications to be built and then add the collaborative functionality in a straightforward manner.

Plug-ins would be available as icons in the integrated environment and could be customized to address the needs of specific groups.

6. Passive and Active Interaction

Large displays that can be readily customizable to provide situational awareness ends up being a very nice thing on a mission, especially one that has largely unnatural overlays like running on Mars Time. People tended to be very aware that they could find the current Mars times of the two missions commonly displayed on the MERBoards. Additionally, there was often schedule information displayed on other MERBoards.

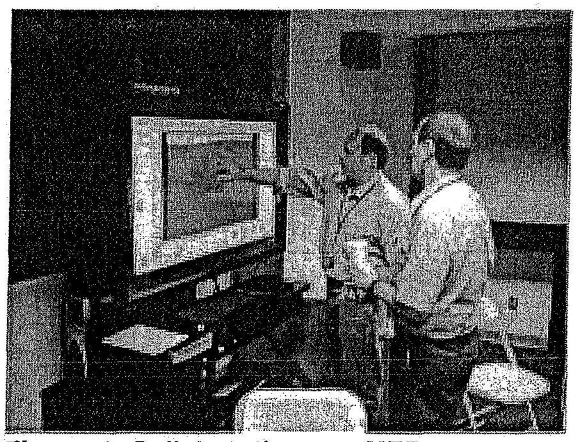
Figure 2 Collaboration on MER

One thing that wasn't always obvious to the users was when it was appropriate to move between the passive mode of situational awareness displays and using them for active interaction. Part of the reason is inferred from the "Design for multi-user interaction" discussion above. Additionally, however, there was a general sense that there were on-going sessions even with the board was in a passive mode, in other words being used for situational awareness because it was not being used actively for another reason. A challenge for taking this work forward is providing the feedback needed to allow users to be aware of whether board is in an active session or being used to provide situational awareness in a passive manner.

7. Virtualization of Applications

The XBoard was designed to be platform neutral, but in some cases needed to run platform specific software. This was especially the case in a recurrent request to be able to run Microsoft Office applications, which often contain important mission information in the form of spreadsheets, documents, and presentations.

We decided to "virtualize" that aspect of the application by running the software under Linux Windows emulation sessions and having the application connect, behind-the-scenes, to these sessions. All of this was transparent to the user, and allowed our software to retain it's portability without losing the benefits of having important platform specific software running in its native environment.

This virtualization functionality can be extended to provide connections from other applications running in a server environment to the XBoard. For example, we've prototyped incorporating image editing and X-Windows applications using the same basic mechanism.

Another advantage to this virtualization is the application is running in a controlled server environment which reduces the risk of contamination through interaction with users' non-standard desktop applications.

8. XBoard Architecture

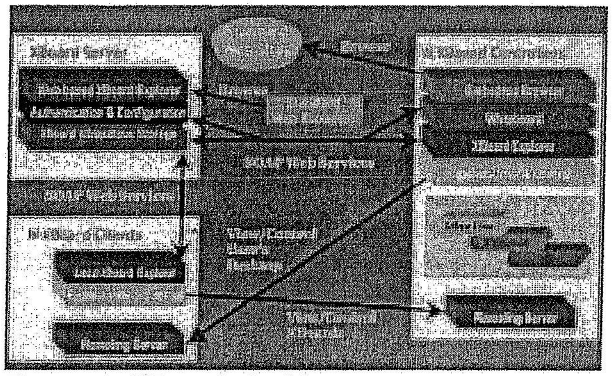

Figure 3 XBoard Architecture

The XBoard Software architecture consists of a Server, a Display Application (or Controller) and a Client. Server functionality consists of providing Web Server functionality (httpd), providing a hierarchical file store, and providing login security through a database.

The Client functionality consists of preparing and modifying files on the shared XSpace, as well as remotely viewing XBoards via VNC.

The next XBoard release will unify the Controller and Client into one Display Application that can function in either a client or controller mode, depending on configuration.

The Controller is the client to the actual touch screen, and is written in Java. It uses a plug-in architecture to support the following core functions:

Whiteboard. The whiteboard application allows users to take notes, create lists, annotate images, and save these files to the shared filespace. The whiteboard supports group, undo, multiple fonts and geometric objects such as circles, squares and ovals.

Embedded web browser. This allows use of standard web pages, included Java applets, to be displayed and interacted with on the large screen displays. It also allows for non-standard handling of web pages, for example loading an image from a web page automatically into the whiteboard.

Remote Control and Display. We are using open-source code from AT&T Research to support the VNC protocol. This allows users to remotely access peer computers to display and interact with programs running on these machines. Additionally, we are running open-source software from AT&T research to support the VNC Server functionality. This allows remote users to login and view an XBoard session from their own systems.

Office File Viewing and Editing. XBoard allows users to collaboratively create, edit and view Excel, PowerPoint and Word documents.

XSpace file explorer. XBoard contains a ubiquitous storage system allowing users and groups to share and store files.

A future plugin planned for MCCBoard will allow X-Window applications to be displayed in the integrated application.

The XBoard Sever Application is written in Java, and integrates several popular open-source products including:

Apache, an open-source web server (httpd) provides the web server functionality.

Tomcat, an open-source Java servlet environment that hooks into Apache, runs Java servlets and JSPs to handle various Server oriented functionality, such as supporting Webdav file transfers and Dynamic Web Page.

Postgres, an open-source relational database, for storing user permissions and configuration information.

mod_dav for certain server functionality relating to dav-based file storage.

We use a Redhat Linux-based server environment, although the Java and open-source tools can run on other platforms (OS X, Windows, Solaris).

9. XBoard Framework

Although the large, touch-screen interfaces worked well for MER, we decided to look into generalizing the framework to support the general issue of providing collaborative capabilities to general notes.

Essentially, XBoard is a collaborative work system designed to support different collaborative work activities.

On-going engagements are now piloting this framework to provide collaboration capabilities to existing applications, as well as using the integrated capability of the core suite of XBoard applications.

Over twenty plugins have been created for the XBoard by a diverse group of software teams. On-going pilots are taking place at MBARI (Monterey Bay Aquarium Research Institute) and LPL (Lunar Planetary Laboratory at the University of Arizona).

We also are entering into pilots with two NASA centers JSC (Johnson Space Center) and KSC (Kennedy Space Center) to enhance their ability to collaborate with other NASA centers and other national and international partners.

We plan to have an end-of year demo connecting JSC with several other NASA centers and an international partner by the end of Fiscal Year 2006.

10. MCCBoard

MCCBoard is an instantiation of the XBoard framework to support collaborative activities for planning activities for the International Space Station.

The system will provide the display of situational awareness data at multiple locations, including a Mission Cognizance webpage developed by JSC for walk-by viewing and interaction at MCC of various information relative to ISS.

Existing web-based systems that have been demonstrated to work in this environment, including Flight Notes, ChitS, Jets, JEDI, web-based systems for communicating Schedule Information, Inventory, Anomaly Reports, assignments and other communications.

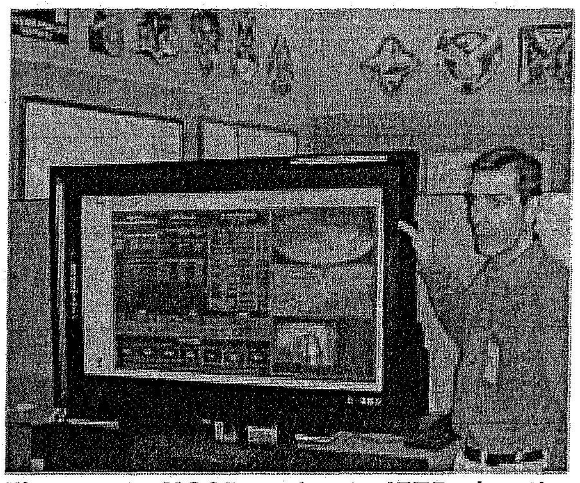
Figure 4 MCCBoard at IETP in the Mission Control Center, JSC

Users will be able to display and control web-based and X-Window based planning systems from multiple locations with synchronous viewing and manipulation.

Information Consistency will be provided across multiple sites/groups using the XBoard Document repository as a mechanism for moving draft versions of plans in Excel between different locations/team.

The system will provide Public/group access to shared information throughout MCC, as well as global access to a central information repository for access to synchronized planning information.

It will support distributed meetings with whiteboard capture and distribution.

It is anticipated that MCCBoard's software architecture will be utilized to provide a modular architecture for building mission operations tools.

The ISS consists of several important International Partners who coordinate closely with the MCC to plan various activities. Control Centers in Russia, the EU, Japan and Canada will all be involved in planning the on-going operation of the ISS once it is completed, based on their various contributions [LeBlanc, 2006].

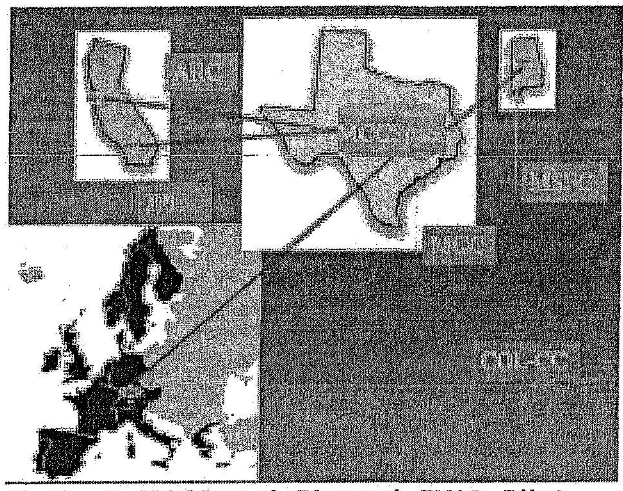

Figure 5 MCCBoard Planned FY06 Pilot Deployment

11. Future Directions

Increasingly, we would like XBoard to be a set of integrated collaboration services that allow enhance existing web application and distributed work practices without creating the necessity to install a "collaboration application" at every node.

Specifically, we envision MCCBoard to be a set of web services that provide a mechanism for adding collaborative functionality to existing web applications with a few web-service calls and a browser-based plugin.

"MCCBoard" would be a substrate that would not be visible to the user as a separate application, but instead would be integrated into a browser-based set of web applications, that now are inherently collaborative.

By leveraging standard web-based https-style secure authentication, users would not have to endure the cognitive shift of going from one network to a different network via VPN. Additionally, all collaboration would just be an extension of familiar, browser-based applications.

However, the applications would now be enhanced by having developer-customizable collaborative capabilities, such as screen sharing, screen capture and annotation, shared browsing, and shared file editing. Additionally, users will be able to save all work to a fundamentally shared, ubiquitous file space.

Existing MCCBoard server features would be refactored as web services. Where appropriate, functionality from the controller would be placed into a browser-based plugin.

We expect to demonstrate this functionality in early FY 07, for possible roll out in 3Q of FY07.

12. References

[1] Huang, E., "Patterns and Challenges for MERBoard", Georgia Tech, 1/2006
[2] LeBlanc, T, Shab, T., *MCCBoard: A Framework for Enhancing Collaboration between Control Centers*, Knowledge Management and Mission Operations Conference, *JSC, 3/2/2006.*