INKBOARD – TABLET PC ENABLED DESIGN-ORIENTED LEARNING

INKBOARD – TABLET PC ENABLED

DESIGN-ORIENTED LEARNING

Hai Ning (ninghai@mit.edu)

John R. Williams (jrw@mit.edu)

Alexander H. Slocum (slocum@mit.edu)

Abel Sanchez (doval@mit.edu)

Intelligence Engineering Systems Lab (MIT 1-250)

Civil and Environmental Engineering Department

Massachusetts Institute of Technology

77 Massachusetts Avenue,

Cambridge,MA02138,U.S.A.

ABSTRACT

This research presents a software tool – InkBoard – that is built on the Tablet PC platform and supports interactive design sessions in the context of design-oriented education. It provides designers, within a small team, a rich set of communication tools including network-shared ink strokes and audio/video conferencing capabilities. It is our belief that this tool will greatly enhance the design-oriented education experience of the students and teachers alike.

Keywords: E-education, design, design-oriented learning, Tablet PC, sketch.

1 Introduction

The advance of computer technology has radically changed the higher-education landscape since the late 1990s. Educational institutes are aggressively exploring the opportunities of leveraging the power of the Internet to facilitate learning.

However, in the design-oriented education field, including architectural design, urban planning, and particularly mechanical engineering design, existing software packages hardly fulfill the need of supporting interactive design sessions with rich user interface and convenient communication tools.

One of the most powerful and yet often overlooked designer’s tools is freehand sketching. Despite the fact that CAD workstations have generally replaced drawing board in design studios, freehand sketch remains to be the indispensable visual thinking tool. It not only serves as communication medium for getting ideas across fellow designers, it also plays important roles in the self-critic conversation designers constantly have with themselves.

With the introduction of Tablet PC, we are able to create a software tool that communicates design ideas over the network in the most natural fashion – freehand sketching. With InkBoard, designers can send and receive ink strokes instantly, store the collaborative drawing in a central database, or as a local file. Every single stroke can be identified by creator, and has timestamp attached so that a sequential replay of the drawing can be performed to give designer a clear idea of the thought generation process. We will discuss many of this functionalities and the architecture of this tool in the following sections. But before getting into technical details, some aspects of design theory are discussed.

2 Design-oriented learning

2.1 What Is Design

Design was always related to certain professions such as architects, engineers, industrial designers and others. And design activity was naturally regarded as what these professions did in order to produce the drawings needed by their clients and by manufactures. However, after the global industrialization, attempts have been made by professional designers to redefine design activity and isolate the essence of designing as a standard method, or recipe, that can be applied to all situations. Numerous literatures dating as early as nineteen fifties and sixties try to labelthe seemingly obvious and yet elusive definition of design activity.

One thing that is common to most of these definitions is that most of them emphasize not the outcome of the design, but the ingredients of design. If we look at design from an end-to-end stand point, the whole chain events – beginning with the initial motivation, and moving through the actions of designers, manufactures, distributors and consumers – leads to the ultimate result of something new, something different from what we have already. In that sense, Jones argues that the "definition of designing is the initiation of change in man-made things"[1]. This definition expands the design activity from traditional design professions such as engineers and architects, to other professions including economic planners, legislators, managers, publicists, applied researchers, politicians etc.

However, the objective of this research is to explore the better approaches for design education practiced in university environment, focusing on fields such as mechanical engineering and architectural design. Therefore, for the sake of this research work, we limit the definition of design to the traditional sense of producing recorded ideas, 2-D drawings or 3-D models that are representations of physical products that serve pre-defined purposes.

2.2 Design Disciplines

The common design disciplines taught in school can include the following areas and combinations thereof:

Architectural design. This includes interior design, urban planning, structural and civil engineering, commercial and residential construction, business services, etc.

Electronic design. This includes microprocessor computer hardware, software engineering, consumer electronics, telecommunications, robotics, etc.

Industrial design. This includes mechanical engineering, manufacturing, fashion design, consumer design, biotechnology, etc.

Graphic design. This includes communications technology, environmental signage, broadcasting arts, animation, advertising, marketing, etc.

2.3 The design-oriented learning approach

Since design is a mix of understanding of explicit design parameters and conducting conscious and yet implicit creative activity, design-oriented learning takes a unique approach of a combination of objectivism/behaviorism and constructivism, with the emphasis on the following aspects.

2.3.1 Creative problem solving

The definition of design by Alexander is to “create something out of nothing and make it useful.”[2] Theoretical developments in cognitive science has provided us tools to analyze the information handling procedures that can be identified within the designer’s operations[3]. These tools, despite different angles of view, all point out the characteristic of design is the creative problem-solving process under the conditions of bounded rationality[4].

Of course, here “problem” refers to a much wider definition of the literal meaning of the word itself, and they can be well-defined[5], ill-defined[6], or they can be wicked[7].

Commonly practiced procedures for problem solving usually include the sequential steps to reach the final design. Since “solving the problem creatively” is such an abstract concept, it is natural for design instructors to resort to a somewhat concrete and linear process that can be more tangible and easier for students to grasp. Constant feed-back loop check during each step ahead ensures the design decisions are made towards the right direction. Classroom experience tells us that the “creation” step usually is the hardest one for students, simply because that is the step where “something comes out of nothing” as pointed out by Alexander.

2.3.2 Team collaboration

With the advancement of the social and technological environment, the design tasks become invariably more and more complicated and require teams of designers to accomplish. Achieving effective collaboration in a design team is never an easy task. The design education students acquired should release them from the tyranny of imposed ideas and enable each to contribute to, and to act upon, the best that he/she is capable of imagining and doing. This is not an easy task, and it can only be achieved by creating a collaborative environment where students can freely express themselves individually and yet the individual efforts can still be unified towards the common goal.

2.3.3 Project management

A complimentary and yet invaluable lesson students should take away with from design-oriented courses is project management skills. Because of the hands-on feature of design courses, design project is often the central if not sole task designed by instructors and handed out to students. With a given deadline, sometimes limited physical resources, students, usually divided into design teams, are expected to turn in a physical design product, or 2D or 3D representation of in the product in the end. The whole project management would likely include but not limited to identifying key players in respective fields, allocating resources, setting up realistic short-time milestones and long-time goals, dividing task into independent modules, designing clean interface between modules, establishing clear boundary of responsibilities, analyzing risks and returns, and on top of all, managing to deliver project on time. These are essential skills that determine the ultimate success or failure of the design project. Incorporating these elements in a project-drive design course can help students build a sense of how design project is handled in the real world.

3 Sketch and Design

3.1 The relations of drawing to problem solving

Understanding the importance of drawing during all the developmental stages of the design process helps to produce better software for supporting design. Here, based on a study in OregonStateUniversitywhere mechanical engineering students were videotaped during a design process, we summarize that drawings help design in the following aspects[8].

- To archive the geometric form of the design
- To communicate ideas
- To act as an analysis tool
- To simulate the design
- To serve as a completeness checker
- To act as an extension of designer’s short-term memory

3.2 Role of Sketching

The capability of making “back-of-the-envelope” or “cock-tail napkin” sketches to aid in solving a given design problem is greatly emphasized in architectural education. In mechanical engineering field where, perhaps to the outcome of the design, form is not as important as function, engineers are still expected to be able to capture and express ideas going through his/her mind with impromptu sketches. This type of quick sketches allow for clearer thinking, stabilizing generated ideas during the conceptual stage of design, and facilitating spatial and geometric reasoning.

Freehand sketch stands out among other types of drawing activity as a much more than a means of communicating design information. Its function includes but not limited to the presentation of spatial information and relationships. Furthermore, it often acts as the link that bridges the gap between the cognitive process of design and the physical world in which the designed artifact eventually will exist. It is the first stage to realize and assess the design ideas.

3.3 Cognitive analysis

Sketching plays the most important role in visual communication because of its low barrier of effort and high expressiveness with trained hands. Study shows that a stunning 67% of all drawings done over the course of a typical design project are freehand sketches. Designers are constantly having a conversation with themselves through the cycle of sketching, inspecting, and revising.

Following is a widely recognized model[9] in which an initial designer generating ideas is called an ORIGINATOR. The designer's mind searches mentally through a broad (DOMAIN) knowledge base and through more specific base related to the problem (CONTEXT). The mental representation is transformed into a physical representation. Depending on the quality and preference on the language skill of the designer the idea is transformed into some graphical solution (sketches) or textual solution (sentences).

Figure 1. System model for generation and interpretation of ideas

A "talk-back" process generates new mental images when the designer sees his or her idea physically represented and realizes changes to the original idea. The "talk-back" process is repeated until the ORIGINATOR is satisfied with the resulting design solution.

Subsequently, the design solution is then passed on to another designer (RECEIVER). The transformation reverses its course where the physical representation "talk-backs" to the RECEIVER and, depending upon language skill and quality, and DOMAIN and CONTEXT knowledge, the RECEIVER creates a mental representation. The transformation cycle continues until the designer is content with his/her interpretation or understanding of the original design.

3.4 Back to the Drawing Board

Today most of the architecture construction documents or mechanical design documents are drawn with CAD software. Unfortunately however, this revolution also brings less favorable impact on some other aspects of the design.

3.4.1 Lost of Public Forum

In the past, the large drawing board served as forums for fellow designers to share their ideas and communicate the design. It was a perfect setup for informal face-to-face meetings where designers gather spontaneously to examine and discuss design problems. Now, not only computer screens are perceived as private workspace and hence not appropriate to be studied by others for a prolonged period of time. The existing CAD packages have not been able to help designer to reclaim this lost public space that was proven, and still should have been, very helpful in a collaborative design environment.

3.4.2 Lack of abstraction, ambiguity, vagueness and imprecision

It is interesting that in the early conceptual, creative phases of designing, designers almost always consciously reject using computer tools. Rather, they prefer making rough sketches with pencil and paper in good old fashion. One of the main reasons for this seemingly peculiar behavior is because of the ambiguity, vagueness and imprecision that is tremendously helpful in early conceptual design, easily obtainable with freehand sketch, and yet unavailable in today’s popular CAD systems. Current design software, which restricts visual representations to precisely drawn geometric elements, stifles the graphical conversation that designer has with herself. CAD drawings eliminates the suggestive power of the sketch.

3.4.3 No incremental information storage

From plotted CAD drawing one can hardly follow the incremental progression of the design, since it is more than often a “cleaned up” version of the initial sketch. The ease of erasing, duplicating, modifying geometric shapes in CAD software often subconsciously encourage designer to constantly remove, refine, or replace shapes drawn during the formation process of design. The final product may be a very clear and clean representation of the design, but the thought process that could have been revealed by the comments, lines or shapes drawn along the way is lost forever.

4 The promise of Tablet PC

It is our hope that the emergence of new technology such as Tablet PC could mitigate, if not entirely eliminate, the aforementioned problems with traditional computers running CAD software packages, since most of these problems are caused by the awkwardness of the reining hardware interface, namely, mouse and keyboard. Keyboard is the natural input device for textual information processing. With the invention of windows-like graphical user interface and mouse, the archaic text command lines, now only cherished by die-hard geeks, are replaced by simply point and click. However, it is sufficient to say that freehand drawing was never a task made easy by these input devices. Tablet PC, with its natural user interface, is trying to change all this.

4.1 Pen-based natural user interface

Perhaps the most striking difference one feels when first time using a Tablet PC is the smoothness of the flowing ink appearing on the screen. Being able to use the stylus writing directly on the surface of the laptop computer screen and seeing the digital ink appearing from the tip of the stylus is a very satisfactory experience indeed. It removes the frustrating barrier of old input devices where users are forced to move their hands holding a digitizer on top of a pad on the side while trying to focus their eyes on the screen to see the outcome. The pressure-sensitive digitizer screen also adds the natural smooth feeling of real strokes of ink with varying width. Thus, designers will be able use Tablet PC just as if they are using pen and paper to make conceptual sketches. Combined with the ease of storing and retrieving digital sketch, perhaps even with shape recognition, at certain stage when the designer is confident enough about the idea, the sketch made can even be imported to CAD software for further detailed formal drawings.

4.2 Advanced operating system

Microsoft Windows XP Tablet PC Edition is a superset of Windows XP Professional so it provides the power of Windows XP with no sacrifices. Tablet PC Edition has the full capabilities of Windows XP Professional, plus additional features for tablet pen–based computing. And, because it uses the Windows operating system, Tablet PC will run Windows XP–compatible applications.

5 InkBoard Software Design

5.1 Goals

The goal of this research effort is to create an Ink-enabled sketch pad application that connects multiple design team members simultaneously through network (preferably wireless network) and allows they to communicate their sketches in real time. With this overall objective in mind, there are three important goals to achieve.

5.1.1 Design-oriented education needs

The InkBoard should address the common needs of design-oriented education.

5.1.2 Scalability

The InkBoard software needs to be highly scalable.

5.1.3 Application usability

InkBoard needs to be Tablet-friendly. Because pen and mouse operate differently from one another, it is a big challenge to offer new UI experience that better fits pen-paper type of interaction but at the same time establishes subconscious connections with traditional windows GUI design so that users would not feel totally disoriented

5.2 Challenge: Ink-

enabled communication

The emphasis of InkBoard as we discussed before lies in the collaboration functions. At the very bottom of multiple client communication is the Ink networking support, which happens to be the one feature missing from the Tablet PC SDK. Other than providing a proprietary serialization method that turns ink stroke object into undecipherable byte arrays, much is left for the developers to figure out how to transfer ink object through TCP/IP network in real time between Tablets. And this is the main technical challenge of InkBoard.

5.3 System architecture

Client/server structure is adopted for InkBoard because of its ease to collect transferred data and the predicable and controllable network traffic pattern.

Figure 2. Client/server architecture of InkBoard

In the client/server architecture illustrated above, multiple Tablet PC clients running InkBoard client application connect to the central InkBoard server via TCP/IP network. They send every stroke the users make, or any other type of actions, over the InkBoard server in the form of InkBoard Message. InkBoard Server collects the messages sent from every client, figures out the meaning of the messages, stores them in the InkBoard Database Server and follows the recipient list of each message and sends them out again to the respective clients. The clients receive these messages, restore them back to strokes and take the appropriate action to display them. All these steps happen in the real time.

5.4 InkBoard Client

Features

InkBoard Client is an Ink-enabled Windows application that has very rich user interfaces and features. Inside the client Windows form class, there are 3 main components – InkBoard drawing area, other UI elements and the network manager.

Figure 3. InkBoard main user interface

5.4.1 InkBoard drawing area

The main drawing area is realized using a class called *InkBoardDrawingArea*. It includes a Windows panel that attaches itself to an *InkOverlay* object to collect user Ink strokes, horizontal as well as vertical *Scrollbars* for moving the panning and zooming, an image *ArrayList* object to hold images uploaded by local user or transmitted over the network from other users and a range of other elements that deals with the collecting and drawing of Ink as well as images.

5.4.2 Sign-in dialog box

InkBoard requires user to identify herself at the start of the program. Normally this is done with a text box presented to the user at the time of connecting. With the advance handwriting recognition algorithm, it is natural to create a sign-in dialog box that automatically converts user’s handwriting into username.

Figure 4. Sign-in dialog box

5.4.2.1 InkBoard toolbar

On the top of the main drawing area, ten different buttons lined up as the most common tasks performed by an InkBoard user. All of these functions can be activated with a touch of the stylus on the respective buttons so that users never have to resort to keyboards. It is also worth noting that the drop down menu appears on the left side of the root menu as it shows in the above screen capture. This is to accommodate right-handed users so that their hand will not block their eye-sight when operating the stylus. Of course this setting can be changed for left-handed users as well.

Figure 5. InkBoard toolbars

5.4.3 Timeline

Another interesting UI feature is the time line. Since we have recorded every message sent by every client to construct a particular drawing, we have the capability to describe each stroke made by each user using a time line whose length is proportional to the time intervals between the strokes. A VCR like interface is provided to the user so they can move the time marker along the time line. Any stroke made after the time marker becomes the “future strokes”, and thus grayed out. User gets a clear picture of how the drawing came into being by dragging the time marker and observing the drawing appearing. This greatly helps them to understand the design process, enhance the design experience immensely.

Figure 6. InkBoard timeline

5.4.4 Presence and ink layers

InkBoard adopted an UI element that imitates the popular instant messenger buddy list. When a user starts the client application, he will see an icon representing himself in the list box. After he joins a collaborative sketch session, everyone who is working, or has worked on the same drawing appears too as icons. A tiny flashing red pen alongside the icon means this person is right now making strokes. This last state has been particular useful since it indicates the actions being taken from the other end of the network. This gives the user a very satisfying sense that someone is working simultaneously with her, and it encourages responsive actions from her own end too.

Figure 7. InkBoard layer and buddy list

The drawing area of InkBoard stores ink stroke objects in layers according to their owner. By clicking on the checkbox in front of the user name, user can turn on or off the layer of – and thus all the strokes made by – a particular user. This is very helpful for identifying individual contributions of the drawing.

6 InkBoard Message Protocol

The technical challenge in developing InkBoard application mainly lies in dealing with networking capability of ink strokes. A basic message structure and a messaging protocol have to be put in place.

6.1 The anatomy of IMP

In order to handle various different type of ink related messages, the sensible thing to do is to implement a base class structure that captures the fundamental elements of similarities among these messages, and then build specific type of messages using class inheritance from the object-oriented programming concept. The advantage of doing this is that we would only need to implement one set of network mechanism to be able to handle the base class message type, instead of having to write different networking code for each of these different message types. Since all other message types inherit from the base message class, the networking code would be able to handle all of them.

6.2 Transferring

InkBoardMessage

After the *InkBoardMessage* type object is serialized into byte array in the form of *MemoryStream* object, the next step is to send it over the network to the server or client through TCP/IP socket.

6.2.1 Asynchronous client socket

After a network connection is established through TCP/IP socket, a *NetworkStream* object is created attached to the connected *Socket* object. By default, the *NetworkStream* class sends and receives data over *Stream* socket in blocking mode, which means the application suspends while waiting for network operation to complete. This is not acceptable for InkBoard client application. To maintain the responsiveness to the user interface, we need to use an asynchronous client socket to send and receive data in non-blocking mode. This way, the application will process the network operation on one thread while the application itself continues to run on its own original thread. *NetworkStream* object provides *BeingWrite()* and *BeginRead()* functions for asynchronous reading and writing. By using these methods, we are able to spin off new threads when network operation is executed. And consequently, a callback function is required to notify the original application thread that the network sending and receiving activities are finished and the data writing is finished, or received data is ready to be further processed.

6.2.2 Preserve message boundary

Unlike UDP, TCP/IP is a connection-oriented protocol in that Windows OS TCP subsystem uses buffers to aggregate packets before sending them out, and thus does not preserve data message boundaries. In other words, the remote device won’t necessarily receive the data the same number of message units.

Figure 8. InkBoard messaging protocol

InkBoard message protocol implements a message marker to solve this problem. Each message is prefixed with a serialized *Integer* that indicates the length of the *InkBoardMessage* object. The *BinaryFormatter* object for serializing object always serializes a four-byte *Integer* into a 56-element long byte array. Thus, from the receiving end, we can always chop off the first 56 bytes or received binary array, and de-serialize it into an *Integer* L, which tells us the length of the immediately following *InkBoardMessage* object. Then we will continue to receive the binary data and store them in a buffer array, and keep a incremental counter until the counter reaches the number L, at which point we know for sure that this is the last byte of the serialized current message object.

7 Integrating Conference xp

7.1 Conference XP & InkBoard

Microsoft Research's Learning Sciences and Technology Group10 started Conference XP as an initiative of the Learning Experience Project, aiming to bring high-quality audio/video conferencing and messaging capability to multicast-enabled broad-band network. InkBoard, being a graphical communication tool through networked sketches, integrates Conference XP technology to provide A/V conference functions on top of the ink sharing interactions among designers. When peers can see each other's facial expression, hear each other's voice while studying each other's freehand sketch on a Tablet PC, this increases the effectiveness of the collaboration immensely.

However, we also have to bear in mind that by using Conference XP technology, we limit the A/V conference functions to users that are on multicast-enabled networks because of the RTP mechanism it adopts. Generally it will not work across different LAN, although the ink sharing part will still continue to function because of the InkBoard Messaging Protocol adopts the unicast approach.

7.2 Integration with InkBoard

To make Conference XP components an isolated module so that it can be easily plugged in to any program that needs A/V conference support, we wrapped the video components and audio components together with their user interface into two Windows ActiveX control objects. The lower-level RTP APIs were retained with minimal changes made to the Managed DirectShow Layer. Using the Microsoft Visual Studio.NET integrated development environment (IDE), these two Windows ActiveX controls can be readily dragged from the control library, and dropped on to a standard Windows form object. The integration code is then automatically generated by the IDE and all the necessary API calls are exposed as public methods of the objects, making it very easy to wire them with the user interface event handling mechanisms of the host application, in this case, InkBoard.

The video conference control that can contain up to four small video windows is placed directly under the participant list box, making it easier for the local user to recognize remote users’ faces and sketch strokes. Each video window has its user name shown as well to make easy connections. The audio control is placed below the video control. There is no limit to the number of users that can talk simultaneously as far as the UI is concerned. It is only restricted by the network bandwidth and the Tablet PC’s hardware configuration. A slider bar is provided for adjusting the audio volume using the digital pen.

Figure 9. InkBoard Audio/Video Conferencing

When user connects to the InkBoard Server, they can choose to have A/C conference enabled by selecting two checkboxes. The InkBoard client will spin off two new threads at that moment. One goes out to make the socket connection with InkBoard Server; the other starts the services to subscribe to a multicast IP, send out and receive the A/V packets.

8 Conclusion

InkBoard, as the Tablet PC software tool built for addressing the synchronous interactive needs between design team members, tries to fulfill the requirements of the design-oriented pedagogy. It is the very first sketch-sharing application built on the Tablet PC, leveraging the powerful Microsoft Tablet PC Platform SDK.

InkBoard is recognized to be a very useful tool in teaching design courses, and was also showcased in Microsoft Faculty Summit 2003, along with other major research effort of Learning Experience Project conducted by Microsoft Research,UniversityofWashingtonandBrownUniversity. InkBoard is also featured as one of the “Partner Downloads” from the Conference XP project website.

InkBoard software is freely available from the Internet (http://iesl.mit.edu), along with part of the source code in the hope of promoting collaboration and to making improvements with the collaborative effort from the academic community. So far, we have got very positive responses, with openly expressed interest for collaboration from various parties such as the Computer Science Department of Rice University, the Computer Science Department of University ofIndiana, and universities fromNorway.

9 References

---

[1] Jones, J. Christopher. (1980). *Design Methods.*New York: John Wiley & Sons.

[2] Alexander, Christopher. (1963). "The Determination of Components for an Indian Village"*.* In *Conference on Design Methods*.New York: The MacMillan Company.

[3] Hayes, J. R. (1978). *Cognitive Psychology: Thinking and Creating.* Homewood,IL: Dorsey.

[4] Rowe, Peter G. (1995). *The Design Thinking*.Cambridge,MA: MIT Press.

[5] Newell, Alan, and Herbert A. Simon. (1972). *Human Problem Solving.*Englewood Cliffs, NJ: Prentice-Hall.

[6] Bazjanac, Vladimir. (1974). "Architectural Design Theory: Models of the Design Process." In William R. Spillers, ed., *Basic Questions of Design Theory*, pp. 8-16.New York: North-Holland.

[7] Churchman, C. West. (1967). "Wicked Problems", *Management Science, 4*, no. 14, pp. B-141, and B-142.

[8] Ullman, David G, et al. (1990). "The Importance of Drawing in the Mechanical Design Process", *Computer & Graphics*, Vol. 14, No. 2. pp. 263-274.Pergamon Press,Great Britain.

[9] Shah, J., Vargas-Hernandez, N., Kulkarni, S., and Summers, J. (2001). "Collaborative Sketching (S-Sketch) – An Idea Generation Technique for Engineering Design", Accepted to

appear in *Joural of Creative Behavior*.

10 Anderson, R., Beavers, J., VanDeGrif, T., Videon, F. (2001). “Videoconferencing and Presentation Support for Synchronous Distance Education”, 33rd ASEE/IEEE Frontiers in Education Conference, Boulder, CO,Nov 5-8, 2003

Acknowledgements

The team was funded by grants from Microsoft I-Campus Program, Kajima Corporation and the Center for Innovation in Product Development at Massachusetts Institute of Technology.