WEB 2.0 in Support of Sketching in Architectural Practice

Alastair Weakley(1,3), Keith Deverell (2,3), Jeremy Yuille (2,3)

1. Creativity & Cognition Studios, University of Technology, Sydney. 2. Spatial Information Architecture Laboratory, Royal Melbourne Institute of Technology, Melbourne. 3. The Australasian CRC for Interaction Design, Level 3, Room 312, Musk Avenue, Kelvin Grove, QLD 4072, Australia.

Abstract. This position paper describes Scribblr, a web-based system to support remote, shared annotation of architectural images and plans. The paper describes the current work practice of our clients, a large engineering firm in collaboration with an architect in another country. In particular we are concerned with sketching activities which are important in their work. We describe how, by employing Web-2.0 technologies in the development of Scribblr, we have been able to meet the requirements of a traditional paper-based sketching environment, and indeed to add new capabilities. The paper describes the rationale behind our application of modern Web technologies in this prototype.

Background

Sketching is a very important part of architectural work practice. As Schön [1] describes, in addition to being used to communicate with others, sketches may also be used by creative professionals as a way of exploring some thought or idea. In making the sketch, the designer may discover new insights into the problem they are working on. The sketch as Schön says, "talks back" to the designer rather than being an inert artefact. Historically, computer based drawing tools have not been effective at supporting the representation of ideas when they are at this nebulous or informal stage [2].

Principal designers, for example architects, have little time and commonly communicate complex design ideas via personally identifiable sketches or scribbles. These thoughts and ideas typically relate to specific projects that develop over a long period of time. The scribbles form part of an ongoing conversation centred on representations of a site or a plan. The office then functions as a filtering and translation engine to turn these scribbles into increasingly refined plans and, ultimately, a building.

The start of the project described here was a discussion with a firm of engineers who routinely work with an eminent architect based overseas who spends a significant portion of his time travelling. Typical projects involve the preparation of a proposal document, detailing a plan for a new site. The work progresses as staff at the engineering firm prepare material for the proposal document, usually graphical, and usually one page at a time. Each page is passed to the architect for comment and this feedback is incorporated into the next iteration of the page in question. From the architect's point of view, material is passed to and from the engineering firm normally by fax. His annotations are typically made directly on a fax that he has received, and he then returns this to the office the next time he is near a fax machine.

This mode of working is effective given that the architect spends much of his time travelling because fax machines are available in most hotels. The situation could be improved, however, if there could be some central repository of information that was accessible to both the architect and his collaborators at the engineering firm. Whereas in face-to-face working this central repository might consist of pieces of paper, we can see that problems are introduced when an attempt is made to translate this working style to the remote condition. By faxing material, a new separate copy of it is introduced and with this comes problems of versioning and so on: a shared central repository would simplify the business of tracking comments and versions. Specific to this domain, there is an issue to do with annotating a facsimile of the original image that is that at some point a person has to translate the annotation and to relate it to the original image. Again, if shared, remote access to the original image was available, then this translation step would be removed and possible confusion avoided. In addition, annotations made electronically could be recorded together with metadata about when and by whom, they were made. Rather than having all the annotations drawn on a sheet of paper grouped together, it might be possible to view them independently and, with reference for example to the order in which they were made, to understand more clearly what was in the mind of the annotator.

When considering the development of systems that should be easy to access from any location, using the Web has always been an attractive proposition. Browser-based systems promise operating-

system-agnostic, remote, shared access to data of almost any sort. In addition, Web-based systems can be relatively painless to update because they typically reside on the server and not on the user's machine.

Historically, a number of difficulties have prevented widespread development of Web-based systems, and many of these related to user interaction. One major breakthrough in this area has become known as AJAX (Asynchronous Javascript and XML [3]), a technique that allows a Web-based system to be in continual regular contact with a server behind the scenes, without interrupting the user. Thus, one can work with a single Web-page which evolves and changes as required, much like a desktop application, rather than a series of discrete Web pages that must load in to the browser one by one [4]. There are increasing numbers of Web-based systems that use these sorts of technologies. Google Docs (http://www.google.com), a Web-based word processor, and Google Maps, an online highly interactive map viewer, are just two examples of this new breed of Web-based application. We describe below our Scribblr system which uses modern Web technologies to support remote, shared annotation of images in architectural practice.

Description of the Prototype

Scribblr is a Web-based system that allows users from any location to annotate images. The fact that the system is accessible through a browser means that it can be used from virtually any location. Collaborators working on projects such as that described above can access the system either using their own machine or, indeed, any other machine connected to the Internet that happens to be available. The user can interact with and annotate the images directly in the browser window without having to use a separate application. The system is in continual contact with the remote server in order to save and retrieve data as required. Base images are loaded into the Scribblr window, and annotations may then be added, or previously made annotations reviewed. Figure 1 shows the system as a user has opened an image, displayed the annotation "This Section' made by the user Keith, and is now in the process of composing a reply.

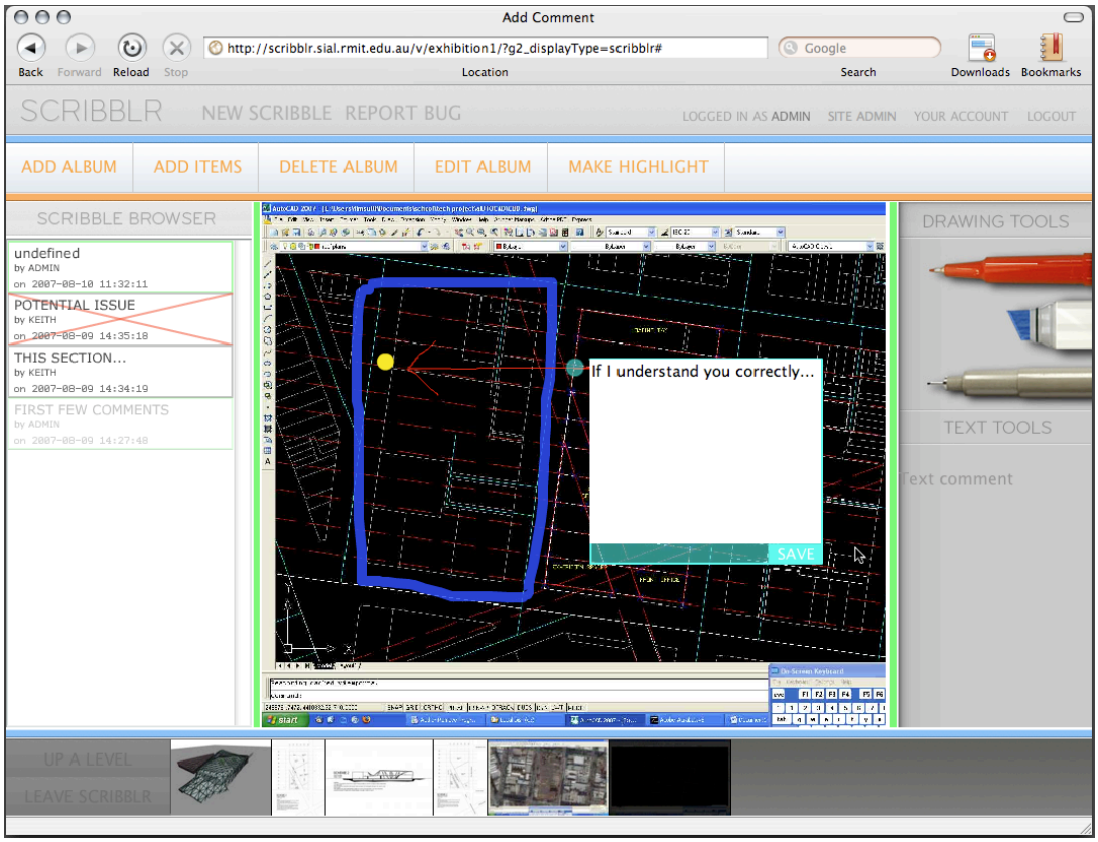
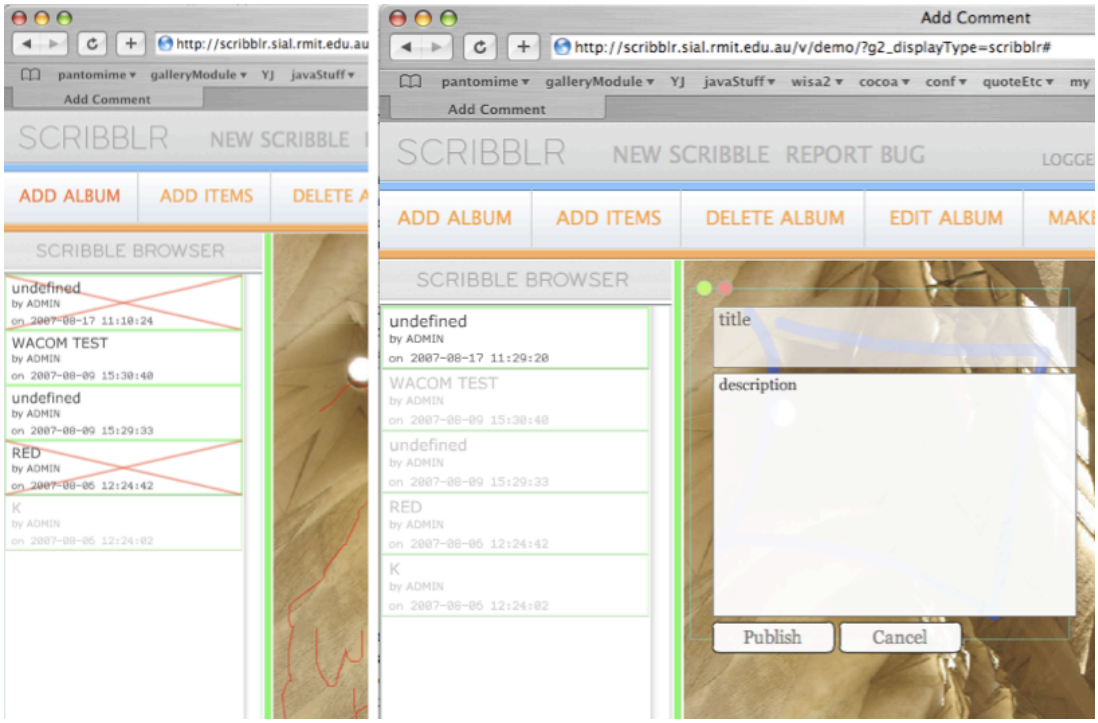


Fig. 1. Scribblr

The images that are to be annotated may, for example, be stored in a gallery-type web application ready to be loaded into the Scribblr page. At the bottom of Figure 1 we can see the other images in the current set (named an Album in this system) which are available for comment. One user might upload a series of images relating to a particular issue and group them in an album. It is possible to navigate through a hierarchy of albums, as well as to use Scribblr to create or modify albums, for instance by adding new images.

Annotations relating to a particular image are stored in a database, and the original image is left unchanged. This means that layers of different annotation, by different people, can be made on the same image without causing confusion. The interface allows the user to show or hide any existing annotation layer or layers. In this way it is possible, for example, to overlay annotations one by one in date order, or to study annotations made by a single person irrespective of when this was done. Figure 2 shows a list of available annotations for a particular image, two are hidden and the rest visible. Clicking on an entry in the list toggles the display of that annotation.



(a) List of Scribbles (b) Publishing

Fig. 2. Views of Scribblr

The annotations that are made on the image can be either sketches or text notes. As the user works with the system the annotations are saved automatically to a remote server. Once the user is satisfied with a particular annotation they can mark it as "published' (Figure 2) at which point it becomes available to other users of the system. At this point, RSS or e-mail could be used to alert other users of the system that a new annotation is available for their consideration.

One important aspect of the system is that, much like making a pencil sketch on a faxed piece of paper, it encourages a fast, sketchy style of working. This is in contrast to many existing computer-based drawing tools. We have not implemented a bezier-curve drawing facility for example, just as one would not use a French curve to assist with a quick pencil sketch. Such tools would delay the process and interrupt the flow of the work. Instead, users can set up a limited 'pencil case' of tools to draw with, a thick marker pen, a red pen and a black one by default. This limited set provides sufficient variety and personalisation without compromising the speed of the application.

Platform

After some discussion scribblr was built upon Gallery [12]. We choose this platform due to its focus on image presentation and organisation. We had discussed building modules or plugins for web based services, such as Flickr [13] but the need for data to be housed locally meant that we moved away from

this approach. Using a pre-existing system has allowed us to focus on the 'tool' rather than the content management system, though not without issues. The template system for Gallery has caused considerable time demands in implementing the interface design and the interactivity we require.

In the design and development of scribblr, we have aimed to use as many open source modules, libraries, and classes as possible. This has enabled actionscript to be able to use XPath, JQuery to handle javascript, and numerous PHP libraries.

At present we haven't used any pre-defined metadata for describing the annotations, wishing initially to study the language that emerged through sketching rather than needing abstract signifiers. With the amount of data that we are sharing between the client and the server our initial emphasis has been on developing a lean schema for our XML.

However, we are now currently looking at the form of mark-up that we use for data transfer. In our current model we have two extraneous processes, one on the server side that forms information from the database into XML, and one on the client side the parses this XML for use in the flash environment. We are currently looking at implementing some form of direct Object parsing using AMFPHP [10] or JSON [11] to remove these steps.

Annotation

Scribblr addresses issues of previously identified requirements for web annotations [5] such as, allowing further edits to the underlying page with out loss of annotations, support for personal and public annotations, providing facilities for organising, filtering, and searching annotations, allowing readers to view annotations in context, and provide readers with easy interactive control over viewing annotations.

At the core of scribblr is a system that provides a means of viewing, editing, and making annotations within the context of the image, that is, comments are located at the exact place on the image they are referring to. To keep annotations separate from the original image all annotations are stored as strings and numerical references, meaning the original image is never changed. By keeping annotations separate from the each other and from the underlying image, a user can select the annotations they wish to view, as well as being able to view the image in its original state. This enables a user to comment on top of another annotation without changing the original comment.

As well as viewing combinations of scribbles on screen, scribblr has the functionality for rendering the image and selected scribbles to PDF. The PDF created keeps the images and the scribbles in their respective formats (bitmap, text, and vector) allowing them to be imported into graphical editing programs, such as Adobe Illustrator, for further editing.

Scribblr utilises a publishing system that enables a user to set the published state of a scribble a either published or unpublished. Publishing a scribble labels the annotation public, making it viewable to anyone. Unpublished scribbles are labelled private and are viewable only by the person who made the scribble. The published state of a scribble can be changed at any given time through controls located in the scribble browser.

In the Scribble Browser the user currently has control over the published state of a scribble, the title and description of a scribble, and whether the scribble is visible (drawn onto the image). The list of scribbles shown in the scribble browser can be filtered based on certain criteria, currently by date, by last number of, and by published state.

Discussion

The Scribblr prototype demonstrates how a Web-2.0 system could support distributed working between creative professionals, in this case the work is set in the domain of architecture. The lightweight, easily accessible working style afforded by the prototype mimics the traditional act of paper-based sketching. In addition, the prototype extends a traditional sketching environment by facilitating sharing of the annotation information.

In the Scribblr system, the act of annotation does not change the original document. Instead, annotation data is stored along with just a reference to the original image. The system allows different annotations to be recalled from the database and overlaid onto the image in layers. It is possible to display as many or as few of the available annotations as is desired. One may exclude annotations that are not relevant to a particular discussion, or review annotations made over time and see how they developed. In both these ways, the system extends what is possible by traditional means.

The fact that the annotations are not embedded in the image data means that they may be redrawn over a higher-resolution version of the same image. When using the system online, for example, it may be most convenient to work with a relatively low-resolution image. Back in the engineers' office, however, the same annotation data could be imported into a high-end graphics application, and its appearance would be the same. The one system therefore supports working at different levels of abstraction. While one person may annotate the image with a fairly high-level view, others may be working with the same data but in much closer detail.

The scribblr system is aimed at enhancing a process that is deeply embedded in creative practice. We do not suggest that input devices for computer-based systems are yet at the level where a designer would prefer to sketch on a computer rather than on a piece of paper. Where scribblr is so effective is in bringing together people who are separated. The system is better than the paper-and-fax approach because: it is more easily accessible, and in our increasingly networked world is likely to become more so; it allows greater flexibility and personalisation in terms of the annotations that can be made; it allows more than one person to work on the same image, either independently, or at the same time studying each other's work; it facilitates closer analysis of annotations, and annotations are largely self-documenting; and it is resolution-independent. an annotation drawn on a low resolution image can be applied to a high-res version of the same image.

Web 2.0

The important advantages of this system have come about largely because it is delivered as a web based service, rather than as software to be installed on a client machine. In developing scribblr we were particularly inspired by current web application practices, commonly described as web 2.0 [6]. Key development practices included concentrating on rich user experiences using collections of technologies like Flash [7], JQuery [8] and AJAX [3], agile prototyping with lightweight programming methods with frequent incremental updates, leveraging open source platforms and code libraries, hooking into existing platforms such as email and rss, and (most importantly) working closely with our users on the development.

We have been able to leverage existing image gallery software to manage the images and merely access these using Scribblr. In an effort to simplify the user experience and focus it on the act of annotation, the system is in regular contact with the remote database, behind the scenes; the user is not interrupted in their work. The annotation data is securely stored in the database as soon as each line is drawn. This was particularly important in this project because the time of the architect is so scarce and so valuable: it would be unacceptable to lose any of his work. Because we work from a central code repository that handles version control [9] we are able to deploy updates to the system as soon as they are ready for public use, often on an hourly basis.

The prototype demonstrates how modern techniques can be used to create highly interactive, browser-based systems to solve real-world problems. With our fast, lightweight annotation system, we can support remote working in architectural practice more effectively than previous methods.

Conclusion

This paper has described Scribblr, a Web-based system that supports remote sketching. We have described our development of the system in response to real-world architectural practice. We have explained how we adopted modern web 2.0 techniques in application service design to meet the requirements in a browser-based system. We believe that such techniques can be used in other areas to create flexible shared working environments that meet the needs or today's distributed workforce even in areas where closely coupled working is traditionally the norm.

Acknowledgements

This work was conducted within the Australasian CRC for Interaction Design, which is established and supported under the Australian Government's Co-operative Research Centres Programme. http://acid.net.au

References

SCHÖN, D. A. (1983) The Reflective Practitioner: How Professionals Think in Action, Aldershot, UK, Arena, Ashgate Publishing Limited.

GROSS, M. D. (1996) The Electronic Cocktail Napkin- a computational environment for working with design diagrams. Design Studies, 17, 53-69.

GARRETT, J. J. (2005) Ajax: A New Approach to Web Applications. Adaptive Path.

CRANE, D. et al. (2006) AJAX in Action, Greenwich, CT, USA, Manning Publications Co.

BOUVIN, N. O. et al. (2002) Fluid Annotations Through Open Hypermedia: Using and Extending Emerging Web Standards, WWW2002. http://www2002.org/CDROM/refereed/656/

O'Reilly, T. (2005) What is Web 2.0: Design Patterns and Business Models for the Next Generation of Software http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html

Flash http://en.wikipedia.org/wiki/Adobe_Flash

Jquery http://jquery.com

Subversion http://subversion.tigris.org/

AMFPHP http://amfphp.org

JSON http://json.orgGallery 2

Gallery 2 http://gallery.menalto.com

Flickr http://flickr.com