A Technology for Lightweight Web-Based Visual Applications

Donald Gordon, Robert Biddle, James Noble,
Ewan Tempero

Technical Report CS-TR-03/4
March 2003

A Technology for Lightweight Web-Based Visual Applications

Donald Gordon, Robert Biddle, James Noble,
Ewan Tempero

Technical Report CS-TR-03/4
March 2003

Abstract

Providing useful and usable visual interaction for web-based applications is a challenge without requiring client-side support such as Java applets. We describe Cliki, which provides web-based lightweight support for visual systems. Cliki makes minimal assumptions about the client used to view the web pages, meaning applications that use it are highly accessible.

Author Information

Robert Biddle and James Noble are members of the academic staff in the School of Mathematical and Computing Sciences and the School of Information Management at Victoria University of Wellington. Donald Gordon is a Bachelor of Honours student in Computer Science at Victoria University of Wellington. Ewan Tempero is a member of the academic staff of the Department of Computer Science at the University of Auckland.

A Technology for Lightweight Web-Based Visual Applications

Donald Gordon, Robert Biddle, James Noble
School of Mathematical and Computing Sciences
Victoria University of Wellington
New Zealand
{donald,robert,kjx}@mcs.vuw.ac.nz

Ewan Tempero
Department of Computer Science
University of Auckland
New Zealand
ewan@cs.auckland.ac.nz

Abstract

Providing useful and usable visual interaction for web-based applications is a challenge without requiring client-side support such as Java applets. We describe Cliki, which provides web-based lightweight support for visual systems. Cliki makes minimal assumptions about the client used to view the web pages, meaning applications that use it are highly accessible.

1 Introduction

The web browser is an ubiquitous user interface paradigm that provides a well-understood interaction model for a shared, distributed environment. This makes it a good candidate for collaboration tools, but providing useful and usable *visual* interaction is a challenge. In this paper, we describe Cliki, a Java framework that provides lightweight graphical interaction for web-based applications.

We have been exploring web applications for supporting various parts of the software development process. While it has been very successful for text-based applications [1, 3], it has been unclear how to support visual tools. One solution is to use technologies such as Java applets, however this allows arbitrary and complex interaction models that are not compatible with the basic web browser. There are also issues with download speeds and browser and Java library compatibilities that can make applets problematic. Our preference is to keep the client side as lightweight as possible, which means making minimal assumptions about the browser, and doing as much on the server as possible.

We have been exploring a idea we call Cliki. Cliki is based on images dynamically generated by the server, and use of HTML image form fields. Our experience has been that it has good potential, however, each application has had to be carefully hand-crafted. To reduce this cost, we are developing the Cliki framework. The intent is that this framework will allow Cliki-style interfaces to be provided for applications at minimal cost.

In this paper, we discuss the general idea of Cliki, demonstrate several Cliki applications, and discuss the current state of the Cliki framework.

2 Web-based tools

With the the rise of the web and scripting languages, it is now possible to find lightweight web alternatives to many traditional large applications. For example, there are many web-based email systems. These applications seldom have all the features of more traditional applications, yet they have some distinct advantages. The constraints of the web interface typically result in lightweight web-based applications being relatively simple and easy to use. Similarly, constraints of the web architecture mean these applications can be implemented simply, and the infrastructure of the web facilitate simple network communication.

Lightweight tools have another benefit — often as a side effect of their implementation technology — in that they are seldom as stylistically refined as traditional applications. Lightweight tools tend to be ugly. The benefit is similar to that in lo-fidelity prototypes, and it is simply that users tend to focus on the functionality. One of the gifts of the web is the general acceptance of low complexity interfaces for simple functionality.

Of course, web applications can also work by using applets or browser plug-ins, but this in essence bypasses the browser and the constraints, and these quickly become indistinguishable from more traditional applications.

Our initial project was to build lightweight web-based CASE tools. We began with a text based tool that supported use case capture. We then began to explore visual interaction. The obstacle was that web browser support for visual interaction is very poor.

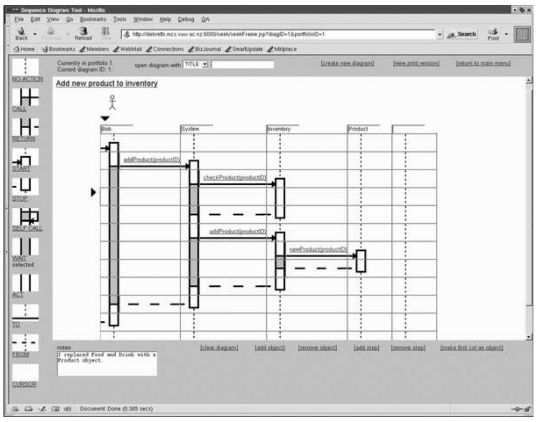

Figure 1. A sequence diagram being modified with Seek

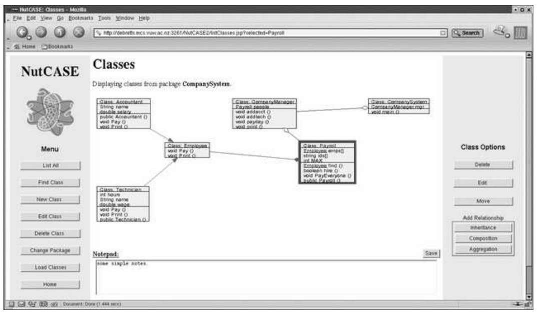

Figure 2. NutCASE displaying a class diagram

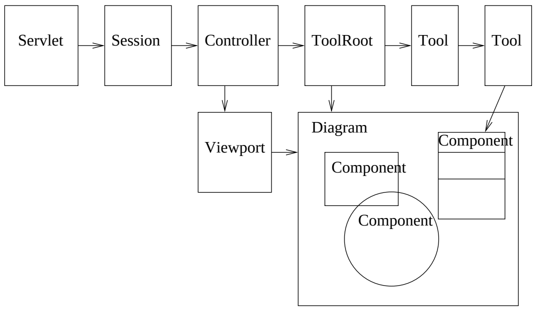

Figure 3. The major elements of the Cliki Architecture. The arrows indicate the usual flow of communication.

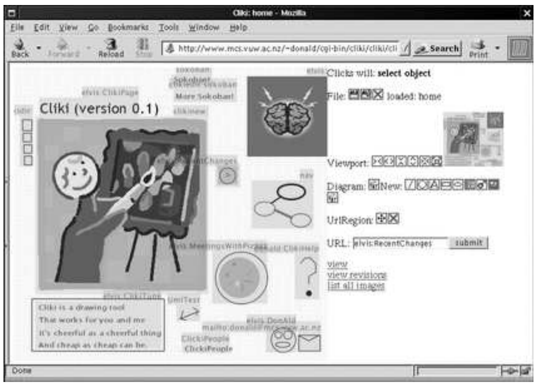

Figure 4. A Cliki diagram editor for drawing and image composition.

We first built Seek [2] (figure 1), a tool for building UML sequence diagrams that used small static images dynamically arranged within an HTML table. We then developed NutCASE [4] (figure 2), a tool for building UML class diagrams by dynamic generation of the whole image. This approach proved sufficiently versatile that we sought to generalise the technology.

3 Cliki Overview

Cliki is based on HTML image form fields. An image field is a input type for HTML forms. It consists of a graphic that is displayed on the web page, and, when clicked on, will send the coordinate of the click with respect to the image back to the server. Image fields are typically used to provide a user-choice mechanism that is more flexible and visually-appealing than HTML buttons or links. As a consequence, the images are static or pre-computed.

What Cliki does is to generate the images dynamically in response to the user clicks. This behaviour allows the image generated to provide direct feedback to the user's actions. For example, the image may show a set of icons. When the user clicks on one of them, the image that is generated may show that icon highlighted in some way. The user may then click in a different location on the image, and then the image that is generated shows the highlighted icon has moved to that location.

This simple idea is very flexible. It allows many of the kinds of things we want to allow in graphical user interfaces to be done, such as drawing packages or games. The limitation is that the user interaction is restricted to just the mouse click. One of our goals is to explore just how much of a limitation this is.

4 Cliki in Action

As mentioned previously, we have several applications built with the Cliki framework. There is a diagram editor (figure 4), a UML class diagram editor (figure 5), and the

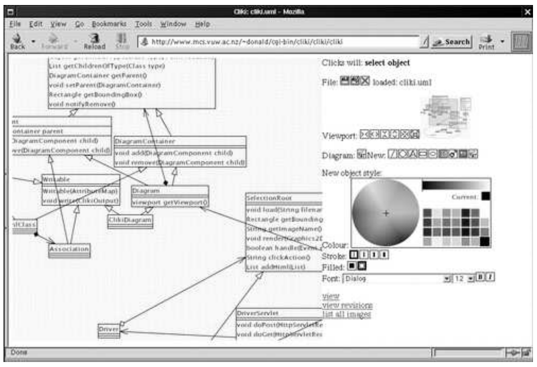

Figure 5. A Cliki tool for drawing and working with UML class diagrams.

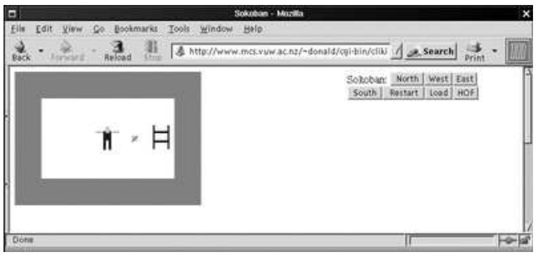

Figure 6. A Cliki Sokoban game.

Sokoban game (figure 6).

The diagram editor offers the usual functionality of such applications. Users can draw different shapes, with different colours, and so on, on a canvas. Shapes on canvas can then be selected, have their properties altered, or be moved or removed.

The UML class diagram editor is effectively an extension of the diagram editor, with specialised shapes (representing a class) and restricted connectivity (different kinds of associations between classes). The UML diagram shown is in fact for the Cliki framework.

The Sokoban game is a re-implementation of an existing Java application. The non-display code is unchanged, with only the display code modified to fit the Cliki framework, so that the graphical part of the user interface looks identical to the original Java application.

5 Architecture

Figure 3 shows the major elements of the architecture, which are described further below.

Controller The controller handles all input. It is passed HTTP requests, which it converts to events that are then processed by the current set of Tools. It is also responsible for generating the HTML form and diagram image that comprise the user interface from the Tools.

Viewport The viewport represents the view of the diagram; it draws the diagram upon request, and translates between image and diagram co-ordinates (e.g. to decide which Component was selected by a click on the image).

Tool Tools, displayed to the right of the Diagram, are used to allow the user to manipulate the image. They are arranged in a chain; the most specific tool (with the greatest relevance to what is currently selected) is at the end. Tools can draw on the diagram (e.g. to signify that a component is selected), process user interface events (to input new points, handle text input, etc) and attach and detach themselves and their children from the chain.

ToolRoot The ToolRoot is a special tool in that it starts the application, and keeps track of the Diagram.

Diagram The Diagram typically stores the set of Components that comprise the image being edited. It provides (as all components do) facilities to render the diagram onto a Graphics2D, and ascertain which component was selected.

Component Components represent parts of the Diagram; they can draw themselves, return Tools that allow the user to manipulate them, and contain other Components.

6 Conclusion

We have outlined the motivation for lightweight web-based applications, and introduced our Cliki technology.

Cliki makes minimal assumptions about the browser used to view the web pages, and so supports visual interaction without any need for augmentation with applets or plug-ins. This in turn allows applications built using Cliki to work on any modern browser anywhere there is web connectivity. We are now evaluating the usability of a range of visual applications built using Cliki technology.

References

[1] R. Biddle, J. Noble, and E. Tempero. Supporting reusable use cases. In *Seventh International Conference on Software Reuse*, pages 210–226, 2002.

[2] R. Khaled, D. McKay, R. Biddle, J. Noble, and E. Tempero. A lightweight web-based case tool for sequence diagrams. In *Proceedings of SIGCHI-NZ Symposium On Computer-Human Interaction (CHINZ 2002)*, 2002.

[3] B. Leuf and W. Cunningham. *The Wiki way: quick collaboration on the Web*. Addison-Wesley Longman Publishing Co., Inc., 2001.

[4] D. Mackay, R. Biddle, and J. Noble. A lightweight web based case tool for UML class diagrams. In R. Biddle and B. Thomas, editors, *Proceedings of the 4th Australasian User Interface Conference*, Adelaide, South Australia, 2003. Australian Computer Society.