Visual Programming for Web Applications that Use HTML Frame Facilities

Takao Shimomura
University of Tokushima
Dept. of Info Sci. & Intel. Syst.
Tokushima
JAPAN

Kenji Ikeda
University of Tokushima
Dept. of Info Sci. & Intel. Syst.
Tokushima
JAPAN

Quan Liang Chen
University of Tokushima
Course of Info Sci. & Intel. Syst.
Tokushima
JAPAN

Nhor Sok Lang
University of Tokushima
Course of Info Sci. & Intel. Syst.
Tokushima
JAPAN

Muneo Takahashi
Toin Univ. of Yokohama
Dept. of Control & SE
Yokohama
JAPAN

Abstract: This paper presents a method that makes it possible to visually design and program Web applications that use frame facilities. The BioPro system that implements this method newly provides Frameset design pages with which we can design Web pages that represent a frameset consisting of several frames and other framesets. We design the contents of ordinary Web pages using Web design pages. Each frame of a Frameset design page has a link to the Web page that is displayed in the frame. The system generates Web application program code from these Frameset design pages and the Web design pages linked to them. Frameset design pages provide several editing facilities such as adding, deleting, changing, and nesting of framesets to make it easier to develop Web applications that use frame facilities.

Key–Words: Automatic generation, Frame, Frameset, Visual programming, Web applications

1 Introduction

Web applications that need to show a lot of information in one Web page at a time often use some inline frames, exchange displayed elements using their tabs, or enable users to see a hidden part of the page using the scroll bars of a Web browser. On the other hand, if Web applications use a HTML frame facility, they can display a lot of information in one Web page at a time and show it to users [1]. The users can easily catch the outline of the provided information, and at the same time, they can see the contents of the frame of interest in detail by extending the frame to the whole page if they need to. Therefore, some Web applications that need to display a lot of information at a time such as computer-assisted instruction systems, Web-based chat systems, and the Help windows of various kinds of Web applications often use the HTML frame facility [2], [3].

For the development of Web applications, various kinds of Web application frameworks [4], [5], [6], [7], [8] and integrated development environments [9] have been proposed and utilized. Among them, visual programming for Web applications has been the object of programmers' attention as a tool that makes it easier to design and debug Web applications [10], [11], [12], [13]. We have developed the BioPro system that enables programmers to visually design the contents of Web pages, database tables, program tables, actions, and Web page transfers [13]. Most of existing development environments assist programmers to design the <body> parts of HTML documents for Web applications, which mainly receive form data from Web browsers, process database tables using those data, and display the results in Web pages. However, in the existing development environments, it is difficult to visually represent the design of Web pages that use frame facilities because <frame> elements are not written in the <body> parts of HTML documents.

This paper presents a method that makes it possible to visually design and program Web applications that use frame facilities. The BioPro system that implements this method newly provides Frameset design pages with which we can design Web pages that represent a frameset consisting of several frames and

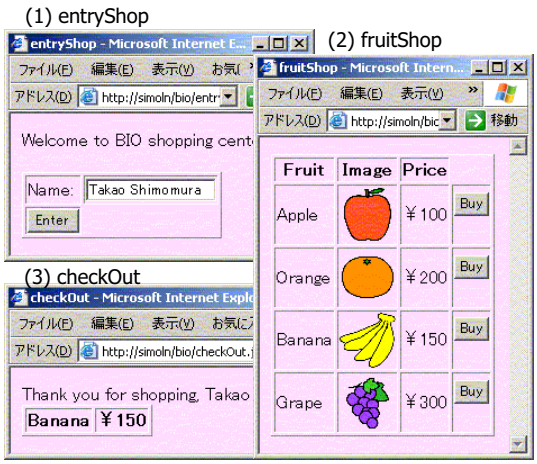

Figure 1: Web application example fruitShop

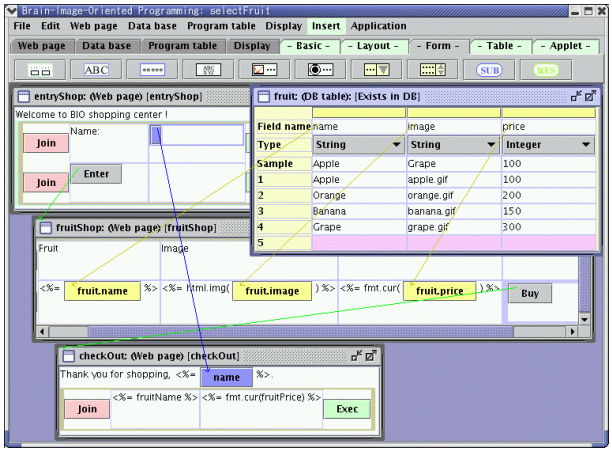

Figure 2: Visual programming for Web application fruitShop

other framesets. We design the contents of ordinary Web pages using Web design pages. Each frame of a Frameset design page has a link to the Web page that is displayed in the frame. The system generates Web application program code from these Frameset design pages and the Web design pages linked to them. Frameset design pages provide several editing facilities such as adding, deleting, changing, and nesting of framesets to make it easier to develop Web applications that use frame facilities.

2 Requirements for Frame Design

2.1 Visual programming for Web applications

The BioPro system is a visual programming environment for Web applications based on the model-view-controller architecture [14] (See Fig. 5). We first describe a procedure of development for Web applications using the BioPro system. Let's consider a simple Web application fruitShop shown in Fig. 1 as an example. In this Web application, we first enter our name in the entryShop page and then click the "Enter" button. The next fruitShop page will show the images and prices of several kinds of fruits. When we click the "Buy" button to select one fruit, the checkOut page shows the name and price of the selected fruit.

To design this Web application, as shown in Fig. 2, we first design the contents of these Web pages by choosing appropriate Web components (forms, text fields, buttons, etc.) from menus and pasting them in each of the Web design pages. We next create a DB design table fruit to create a real DB table fruit, which stores several kinds of fruits to be displayed in the fruitShop page. We drag the name, image, and price fields of DB design table fruit and drop them in the fruitShop Web design page so that the fruitShop page will display these fruits. We also drag the name text field of the entryShop Web design page and drop it in the checkOut Web design page so that the checkOut page will display the name that is entered in the entryShop page. We finally choose the target Web page control transfers to when we click each of submit buttons in these Web design pages to complete the development of this Web application.

2.2 Requirements

To make it easier to develop Web applications that use frame facilities, we take into account the following requirements:

1. We can use the existing visual programming procedure of Web applications that do not use frame facilities as it is.

2. We can nest frames in a frameset any number of times, and we can easily understand the relationship between the frames and the Web pages that are displayed inside them.

3. We can easily change the structure of framesets, easily add and delete frames, and easily change the Web pages that are displayed inside frames by using the mouse operation.

3 System configuration

3.1 Introduction of frameset design pages

To make it possible to use the existing visual programming procedure as it is, we separate the design

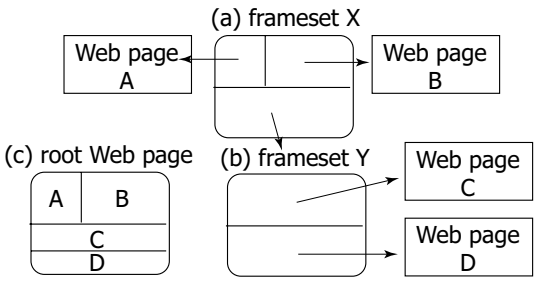

Figure 3: Design page for framesets

of frameset pages from the design of ordinary Web pages (Requirement 1). We introduce Frameset design pages as root Web pages that represent framesets, and link each frame included in a Frameset design page to an ordinary Web design page. For example, in Fig. 3 (b), Frameset design page Y has two frames, and its upper frame is linked to Web design page C, and its lower frame is linked to Web design page D.

We make it possible to link each frame included in a Frameset design page to not only an ordinary Web design page but also another Frameset design page. This enables frames to be nested in a frameset any number of times (Requirement 2). As shown in Fig. 3 (a), the Frameset design page X has two frames, where its upper frame is divided into two other frames, left and right frames, each of which is linked to Web design page A and B, respectively. Its lower frame is linked to another Frameset design page Y. When we execute the Web application that contains these pages, the Frameset design page X is displayed as a root Web page that includes four Web pages A, B, C and D as shown in Fig. 3 (c). In a Frameset design page, we can easily divide, delete, exchange frames by clicking or drag-and-dropping the mouse (Requirement 3).

Figure 4 shows two Frameset design pages and four Web design pages created by a programmer using the BioPro system, as described in Fig. 3. Each frame of the Frameset design page is linked with a pink line to the Web design page that is displayed in it.

3.2 Framework for generating Web applications

The system generates Web application program code from the visual design of a Web application. As shown in Fig. 5, we first visually design the contents of each of Web pages in their Web design pages. When we use database tables, we first visually design those database tables in DB design table windows. Instead, we may specify the names of existing database tables to display them in DB design table windows. We drag and drop the fields of a database table from

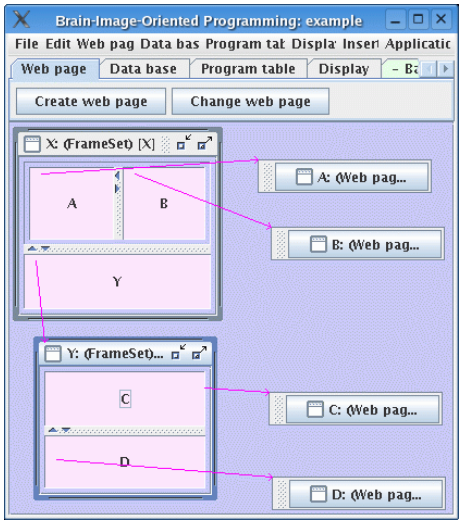

Figure 4: Frameset design pages created by the BioPro system

a DB design table window to a Web design page so that these fields will be displayed in the corresponding Web page. When we use a table in the program that exists only during the execution of the program, we visually design the contents of this table in a Program table design window. For example, we design a table of an online-shop cart in this Program table design window. The system automatically generates JavaBeans code from these Program tables. We next write the actions that are executed when control transfers to a Web page in the Web source window that corresponds to the Web page. Control may transfer to one Web page from multiple Web pages. In the Web source window of a Web design page, for each Web page control transfers from, we can write a necessary action, which is executed when control transfers from the Web page to this Web page. From these resources, the BioPro system automatically generates Servlets, JSP pages [15], and Java classes that compose a Web application in a client side, and uploads them to the Web server together with other resources such as image files, and customized Java classes. To start the Web application, the system then runs a Web browser to make it send a request to the first Web page of the Web application.

Figure 6 shows how the program code that implements a Web application will be generated from various pieces of information designed using the BioPro system. From a Frameset design page X, the system generates JSP page X.jsp that displays the frameset in a Web page. The frame of Frameset design page X is linked to Web design page A that is displayed in this frame. Therefore, JSP page X.jsp

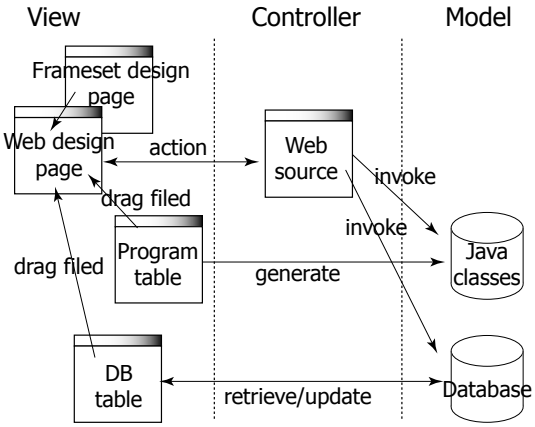

Figure 5: Automatic generation of Web application programs

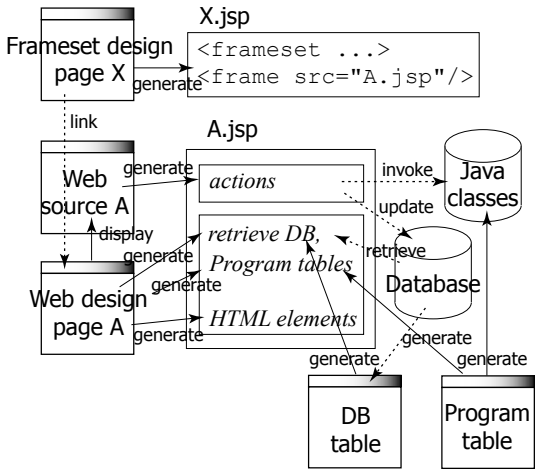

Figure 6: Automatic synthesis of program code from design

contains the HTML code ""<frame src="<%= response.encodeURL("A.jsp") %> .../>"", which displays Web page A in it.

From Web design page A, the system generates JSP page A.jsp that displays its Web page. In the Web source window that corresponds to Web design page A, the system generates some variables that store values such as the values of form data to be sent to Web page A, the names of Program tables, and the names of DB tables that Web design page A refers to. Moreover, it generates some methods that process those Program tables. Using these predefined variables and methods, programmers can write actions to be executed when control transfers to Web page A for each Web page control transfers from. JSP page A.jsp contains the program code that performs these actions, HTML elements designed in Web design page A, and the program code that displays the values of the fields of DB tables drag-and-dropped to Web design page A.

4 Implementation

4.1 Visual programming for Web applications using framesets

As an example of a Web application that uses frames, let's consider a simple Web-based chat system. As shown in Fig. 7, this Web page consists of two frames. The upper frame has a text field to enter a name, and buttons to enter and exit a chat room. Below them, it has an area to display the contents of chatting and its status. The lower frame has a text field to enter a chat message, and a button to send the message. To update the contents of chatting that vary any minute, the upper frame keeps being refreshed at a certain period of time. Because this Web page is divided into these two frames, the lower frame is not affected by refreshing the upper frame even when a user is entering a message.

Figure 8 shows an example of visual programming of this Web-based chat system. The window in the top left corner of Fig. 8 (a) is a Frameset design page, which corresponds to the Frameset design page Y in Fig. 3 (b). The upper frame is linked to chatFrame Web design page to display chatFrame Web page. The lower frame is linked to utterFrame Web design page to display utterFrame Web page. Figures 8 (b) and 8 (c) show the Web design pages of chatFrame and chatFrame Web pages, respectively. In the Web source window of chatFrame Web design page, we can refer to variables user, enter, exit, and textarea as predefined variables, which are generated from the visual design of the Web pages by the system. For example, variable user has the value of the name text field. Using these predefined variables, we can easily write necessary actions for the processes of entering and exiting the room. In the Web source window of utterFrame Web design page, we can easily write the process for adding the submitted message by referring to the contents of the message as a predefined variable.

4.2 Automatic generation of Web application programs

Figure 9 shows a method to automatically generate the program code of the Web-based chat system from the visual design created in Section 4.1. We first designed a Frameset page chat in the Frameset design page window, and designed two Web pages that are displayed inside its frames in the Web page design windows. In the Web source windows of the corresponding Web

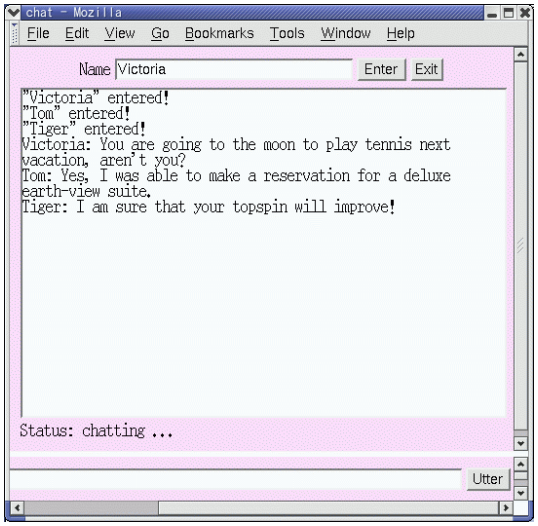

Figure 7: Example of Web-based chat system that uses frames

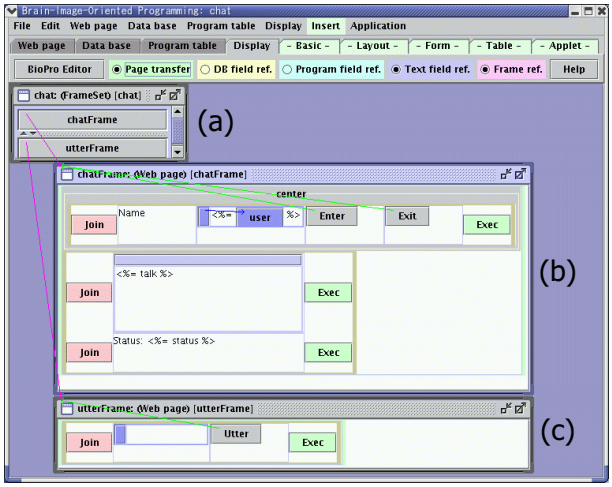

Figure 8: Visual programming using frameset design pages

pages, we then wrote the necessary actions that would be executed when control transfers to each of the Web pages by referring to predefined variables that were automatically generated by the BioPro system.

The BioPro system generates the program code that composes the Web application from these pieces of design information. It generates a JSP page chat.jsp from the Frameset design page chat, and as shown in Fig. 9, it generates a JSP page utterFrame.jsp by synthesizing the utterFrame Web design page and its Web source. The part of the program code printed in italics in Fig. 9 represents the code that was automatically generated by the BioPro system.

5 Observation

Using visual programming, we were able to develop a Web application that uses frame facilities. We were also able to nest frames and framesets in a frameset. In the conventional development, it required 272 lines of code (10 for chat.jsp, 178 for chatFrame.jsp, and 84 for utterFrame.jsp) to develop the Web-based chat system we considered as an example. This method reduced it to 63 lines of code, which is about one-forth. As shown in Fig. 8, the structure of the Web application is displayed visually, and this makes it easier to understand the program and efficient to modify and debug the program.

6 Conclusion

This paper has presented a method that makes it possible to visually design and program Web applications that use frame facilities. As the next step, we are going to investigate how to visually incorporate rich components such as Flex, Flash, JavaScript, and Applet in the design phase. In addition to visual programming of Web applications, we will also investigate how to visually display the design information of Web applications during their execution that is required for enhancing and maintaining the program.

References:

[1] S. Pemberton, et al. *XHTML 1.0 The extensible hypertext markup language (Second Edition)*. http://www.w3.org/TR/xhtml1/, 2002.

[2] P. van Schaik and J. Ling. The effects of frame layout and differential background contrast on visual search performance in web pages. *Interacting with Computers*, Vol. 13, No. 5, pp. 513–525, 5 2001.

[3] T. Comber and J.R. Maltby. Layout complexity: does it measure usability? pp. 623–626, 1997.

[4] *The Apache Software Foundation : Struts*. http://jakarta.apache.org/struts/, 2004.

[5] A. S. Christensen, A. Moller, and M. I. Schwartzbach. Extending java for high-level web service construction. *ACM TOPLAS*, Vol. 25, No. 6, pp. 814–875, 2003.

[6] Steven John Metsker and William C. Wake. *Design Patterns in Java*. Addison-Wesley, 4 2006.

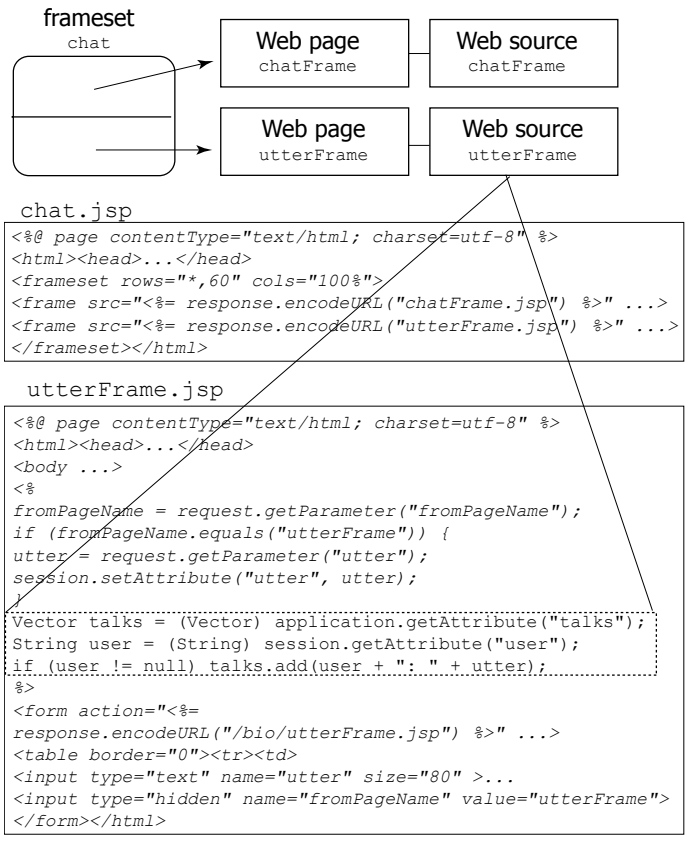

Figure 9: Automatic generation of Web application program code

[7] Seung C. Lee and Ashraf I. Shirani. A component based methodology for web application development. *Journal of Systems and Software*, Vol. 71, No. 1-2, pp. 177–187, 4 2004.

[8] A. Repenning, A. Ioannidou, M. Payton, W. Ye, and J. Roschelle. Using components for rapid distributed software development. *IEEE Software*, Vol. 18, No. 2, pp. 38–45, 2001.

[9] *The Eclipse Foundation : Eclipse*. http://www.eclipse.org/, 2006.

[10] T. Shimomura and S. Isoda. Linked-list visualization for debugger. *IEEE Software*, Vol. 8, No. 3, pp. 44–51, 1991.

[11] *Sun Microsystems, Inc. : Sun Java Studio Creator*. http://wwws.sun.com/software/products/jscreator/, 2004.

[12] *Iron Speed, Inc. : Iron Speed Designer*. http://www.ironspeed.com/, 2006.

[13] Takao Shimomura. Visual design and programming for web applications. *Journal of Visual Languages and Computing*, Vol. 16, No. 3, pp. 213–230, 6.

[14] A. Leff and J.T. Rayfield. Web-application development using the model/view/controller design pattern. pp. 118–127, 9 2001.

[15] *Sun Microsystems, Inc. : JavaServer Pages Technology*. http://java.sun.com/products/jsp, 2006.