Client-Side Centric Model for Generating One-Page Modern Web Applications Dealing with Databases

Hiroki KUMAMOTO, Tomoya NORO and Takehiro TOKUDA
Department of Computer Science, Tokyo Institute of Technology
2-12-1-W8-71, O-okayama, Meguro, Tokyo 152-8552, Japan
{kumamoto,noro,tokuda}@tt.cs.titech.ac.jp

Abstract. Nowadays modern Web applications provide desktop-application-like flexible user experiences without using explicit requests. Modern Web applications need complex components: server-side logic programs, server-side communication programs, output Web pages, client-side logic programs, and client-side communication programs. We present a new generation model called client-side centric model. This model allows us to define one-page modern Web applications dealing with databases easily. From GUI based declarative definitions, we can generate COMET technology based modern Web applications such as chat applications and calendar applications.

Keywords. modern Web application, generator, AJAX, COMET

1. Introduction

The nature of Web applications have dramatically changed these years. Classical Web applications used to be very simple. They execute server-side programs according to explicit submission of requests from client-side and return responses to Web browsers each time.

Today modern Web applications can provide desktop-application-like user experiences without using explicit requests. Modern Web applications can respond to various user events and server-side events, and perform partial update of Web page screen. Examples of modern Web applications include Google Docs, Google Calendar, and Google Maps. These Web applications behave just like desktop applications except that we are using Web browsers only.

From a viewpoint of generation of Web applications, modern Web applications are much harder than classical Web applications. For modern Web applications, we need to generate following five components as shown in Fig. 1:

1. sever-side logic programs
2. server-side communication programs
3. output Web pages
4. client-side logic programs
5. client-side communication programs

The complexity of modern Web applications seems almost greater than or equal to the complexity of desktop applications. In typical manual implementations, components from 1 to 3 are described using Java servlets and components from 4 to 5 are described using JavaScript. In addition, we need to use advanced communication technology between client side and server side such as AJAX (Asynchronous JavaScript and XML) technology [1] and COMET (or reverse AJAX) technology [2]. AJAX technology allows client side to send asynchronous requests to server side. COMET technology allows to have long-held connection so that server-side events are known to client side without using frequent asynchronous requests.

Historically we have two approaches to generate classical and modern Web applications: server-side centric model and server-side with client-side model [3–5]. Server-side centric model generates sever-side logic programs and output Web pages only. No client-side logic program is generated. Explicit submission of requests from client-side is needed to go on Web applications.

Server-side with client-side model generates sever-side logic programs, output Web pages, and client-side logic programs. Logic programs for sever side and client side are both generated. In this generation model it is not possible to have tight communication between client side and server side. Usually client-side logic programs perform independent tasks such as input data checking.

So far there exists no generation model to deal with full-scale modern Web applications consisting of the above five components. The purpose of this paper is to present a new model to generate modern Web applications. We refer to this model as client-side centric model. Our basic target Web applications are one-page modern Web applications dealing with databases. Unlike previous generation models, our model allows client side and server side to have tight communication and cooperation. Also one-page Web applications may have pseudo multi pages which are all created at initialization time of Web applications and one of them is selected to be shown at a time.

The organization of the rest of this paper is as follows. In Chapter 2, we explain our client-side centric model for generation of modern Web applications. In Chapter 3 and 4, we give one example of generation and evaluation of our proposed model respectively. In Chapter 5, we give conclusion.
2. Client-Side Centric Model

Our client-side centric model focuses on client side and we describe the system as if client side and server side are on the same side together as shown in Fig. 2. We describe the system as if the target system is a single standalone system. This reduces the complexity of the description of the system.

Our description consists of three definitions: data model, UI model and event model. Data model defines databases and data. UI model defines user interface of one-page Web application, namely UI component arrangement on the page. Event model defines event handling methods to associate change of user interface with input/output of databases. We describe both user events and database events as if server side and client side are on the same side together.

For UI model definition, we use following UI components: label, button, textbox, passwordbox, datetextbox, timetextbox, selectbox, table, panel and frame. A frame is a container of panels and a panel shows arrangement of UI components and information. For UI components and databases, we have predefined events. For example, in the case of databases we have following events: onInsert, onDelete and onUpdate. For buttons, we have events onDisplay and onClick.

For user events and database events, we can perform predefined actions. Predefined actions perform side effects of UI interface and databases. Predefined actions are basic actions and compound actions. Basic actions are change actions of databases such as insert, delete, and update, and change actions of user interface such as call of value setter functions. Compound actions are sequential composition or conditional composition of basic actions. From client-side centric model definition, we can generate components of modern Web applications as shown in Fig. 3. Database event detection is performed on the server side. User event detection is performed on the client side. Change actions of databases are done by Java servlet programs and change actions of user interface are done by JavaScript programs with the help of client-side and server-side communication programs.

3. An Example

We explain our model using one example of modern chat Web application. In this chat Web application you can login, select one of chat rooms, chat with other users in the chat room, and leave the chat room. This Web application has three pseudo-multi pages, which are panels. Three panels are login panel, room selection panel and chat room panel. These three panels are stored in a container called root frame.

In login panel, you give your user information to enter the system. In room selection panel, you pick up one of chat rooms. In chat room panel, you type your user name first.
Then you type your messages and watch all the available messages independently. We constructed a GUI-based definition system called model editor, as shown in Fig. 4, to generate XML definitions of modern Web applications.

As shown in Fig. 5, 6 and 7, we can define data model, UI model, and event model using our model editor. In Fig. 5, we define the structure of three databases called users database, message database, and room database. We define a database by specifying field names and data types. In Fig. 6, we define UI components of chat room screen. The outermost rectangle shows the panel of chat room screen. Inside the panel we define one frame and one table for showing the list of messages. This frame is a container of two panels. They are user name panel with one textbox and two buttons, and also message panel with one textbox and two buttons. In Fig. 7, we define the event handling method.
for submission of the message to insert the message into the database by clicking the button.

GUI-based definition is transformed into XML based declarative definition. For example, event handling method for sending messages is defined as follows. Here, when a messageSubmitButton is clicked, a message given in the messageTextbox is inserted into message database.

```xml
<!-- Sending Message-->
<event target="messageSubmitButton" type="click">
  <action>
    <insert id="insert_message" database="message">
      <param name="message">
        <value element="messageTextbox" func="getValue"/>
      </param>
      <param name="user_name">
        <value element="userNameTextbox" func="getValue"/>
      </param>
    </insert>
  </action>
</event>
```
Event handling method for message update is defined as follows. Here, when a new message is inserted into the message database and the message is belonging to the chat room you are using, your message table is updated on the screen.

From the XML definition we generate five components of modern Web applications. Each action in the XML definition is transformed into combination of actions on the client side and server side with the help of communication programs. For example, de-
Figure 8. Generated chat Web application

tection of arrival of a new message in the message database on the server side causes the update of message tables on the client side. Also submission of a new message on the client side causes the insert of the message into the message database. The screenshots of the generated system are shown in Fig. 8. We show four screenshots at different time of the generated chat application.

Our system generates code using COMET technology. Long-held connections between client side and server side are used for updates of messages of client-side message table using newly arrived messages of server-side message database. Ordinary HTTP requests are also used for display of initial client-side message table and submission of messages into server-side message database. We use Jetty Web server which allows us to run our modern Web applications on the computer if we have Java execution environment. Jetty Web server accepts both COMET technology and AJAX technology.

4. Evaluation

Manual programming of modern Web applications is a complex task for creating flexible and efficient source code. The size of our client-side centric definition of modern Web applications is much smaller than that of manual programming definition without requiring knowledge of Java or JavaScript programming.

Manual programming related approaches include S-Web generator [6] and Google Web toolkit [7]. Using S-Web generator we can generate classical Web applications from standalone Java applications. Using Google Web toolkit, we can generate AJAX applica-
tions/JavaScript front-end applications from Java applications. These approaches require knowledge of Java programming.

Previous generation models, such as server-side centric model and server-side with client-side model, cannot produce modern Web applications with tight communication between client-side and server-side.

Our generated COMET technology based chat application works well using the long-held connection time of one minute. We can watch updates of message tables naturally. We can also modify our generator system to generate AJAX technology based chat application. If we use polling interval of every 3 seconds, then AJAX technology based chat application can show natural updates of message tables.

Our approach may have the limitation in defining Web applications of very complex server-side logic programs or of truly multiple output Web pages with complex user event handling.

5. Conclusion

We have presented client-side centric model for generation of modern Web applications. This model allows us to define one-page modern Web applications dealing with databases easily. We describe the system as if client side and server side are on the same side together. This approach reduces the complexity of modern Web applications consisting of complex five components. Typical modern Web applications such as chat applications and calendar applications can be generated automatically from the GUI based declarative definitions without using knowledge of Java or JavaScript programming.

References