A Web UI Modeling Approach Supporting Model-Driven Software Development

Jinkui Hou

Abstract Model-driven software development has become a tendency in software engineering. However, Web user interfaces have the characteristics of the customization but frequently renewing, which makes traditional software development approach not suitable for the design requirements of Web pages. To solve the problems of the development of Web user interface, and focusing on the characteristics of Web application, a user interface modeling approach is proposed on the basis of interface template and XML technology. Web user interfaces are described in a direct-viewing style with graphics at the model level. The approach can provide an effective support for model-driven software development.

Keywords Web user interface · Interface template · XML · Model-driven development

1 Introduction

With the rapid development in network technology, the requirements of the user continue to increase for the development efficiency and quality of Web applications, which leads to the increasing difficulty in development. Model-driven software development is becoming a trend in software engineering, in which user interface modeling is an important part. The core idea of the traditional conceptual model of user interface, such as PAC model and MVC model, is the separation between display and logic functions, but it is not well-supported model-driven software development. General interface design tools are simple and easy to use, but they only describe the static interface and do not support interaction. With the

J. Hou (✉)
School of Computer Engineering, Weifang University, Weifang 261061, China
e-mail: jkhou@163.com

progressive development in model-driven software development and the growing separation between software functions and interface presentation, it has become the trend of software design to generate the interface according to the interface model [1, 2]. There are many model-based methods and tools, such as UIDE [1], MASTERMIND [2], UMLi [3], but they can only provide some design descriptions and recommendations and not provide effective support to the final interface generation and UI layout. Furthermore, there is no detailed consideration on the special requirements of the user interface modeling in Web environment.

Web interface is centered by information display, and the user is a complexity. The requirement of Web UI not only includes high availability and robustness, but also includes more visual appeal. It plays a critical role for the layout and presentation form of user interface to the success of the whole software [3]. A good UI presentation model must be extracted from different domains, so as to provide a better guide to the automatic generation of Web user interface.

To solve the problems mentioned above, a Web user interface modeling approach is proposed on the basis of interface template and XML technology, which can provide an effective support for model-driven software development.

2 FMP Model

The function, model, presentation (FMP) [4] modeling approach is on the basis of the traditional application modeling approach by adding a description view of user interface, which is shown in Fig. 1. The model of this approach is not a description of the specific forms of interface elements and their attributes, but the description of the abstract data and behavioral elements of user interface. At the same time, the corresponding relationships between interface elements and presentation objects are also shown. Thereby, the interface elements are independent with any specific application platforms, and the data elements, and behavior elements, are separated from specific interface elements. As the source of model-driven development, the FMP modeling approach can be used to build platform-independent Web application model.

The extended use case diagram in UML is used in function model, which determines the UI needs to the internal model and the relationship between various UI pages through the analysis of the requirements of users.

Internal model includes object model and interaction model, in which object model describes the system from the static view, and it describes the necessary objects that constitute UI interface and the relationship between them. Interaction model describes the system from the dynamic view, which describes the composition of UI and the relationship between interface elements on the basis of object model and function model. The abstract form of the user interface can be fully expressed through the internal model. Presentation model shows the layout and presentation style according to the internal model and the display requirement of data from the users. It is a full description of the user interface in direct-viewing

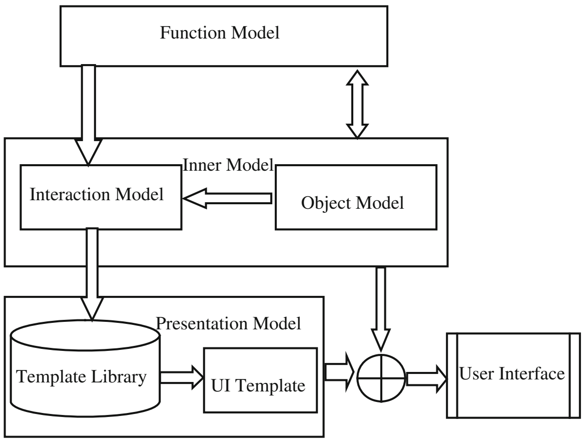


Fig. 1 The structure of FMP model

style, so that the interface elements are independent on any specific application platform, and data elements and behavioral elements are separated from the specific interface elements.

3 User Interface Template

Rich experience has been accumulated in the process of software development. In this paper, the FMP model is extended in order to guide later software development. The organizational structure of interface templates and template libraries is improved, which can provide an intuitive presentation at the modeling stage for the previous development experience and interface design patterns. Thereby, the efficiency and quality of software development are increased.

3.1 The Structure of Interface Template

As shown in Fig. 2, the presentation view is composed by interface templates, interface area, template interactive object, and abstract description controls.

An interface template is divided into multiple interface areas according to the layout, while a template interactive object consists of one or more abstract description controls. A template interactive object is composed of one or more abstract description controls.

At interface template layer, the interface style and macrolayout should be determined. In interface regional layer, main consideration should be given to the display form of the interactive object. In the abstract description controls, layer's main consideration should be given to intuitive interface presentation.

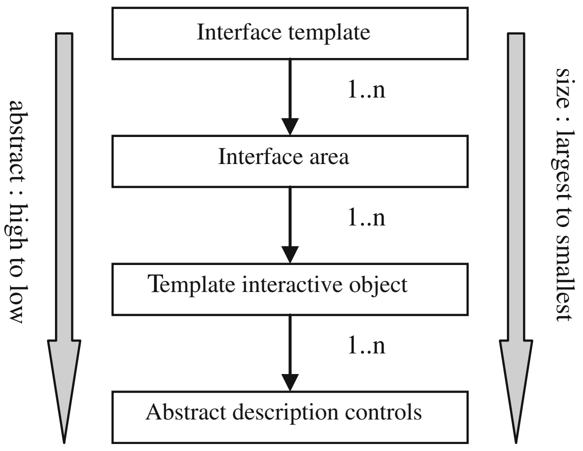

Interface template
1..n
Interface area
1..n
Template interactive object
1..n
Abstract description controls
abstract : high to low
size : largest to smallest

Fig. 2 The structure of UI presentation model

The four-layer structure of interface design makes it easier to control parameters of the template, which separates concerns at different abstract levels.

3.2 The Layout Tree of Interface Template

The layout tree of interface template describes the interface layout strategy. A Web page is divided into different zones. There are two adjacent relationships between these zones: horizontal adjacent and vertical adjacent.

As shown in Fig. 3, the whole page is divided into left and right parts through vertical division.

The left part is a tree style, and the right part is divided into upper and lower sections through horizontal division. The upper part is the graphics and the lower part, a table.

The leaf nodes of the layout tree represent abstract description components, and the non-leaf nodes represent the division mode of the interface template. According to this layout style, the layout of the structure tree can be easily described and its XML representation is shown as in Fig. 4.

4 Library of Interface Templates

According to the classification criterion of interaction model, a variety of templates and common abstract description widgets are stored in the library of interface templates in the form of XML files. The template library consists of three parts: storage subsystem of template library, query subsystem, and maintenance subsystem. Storage subsystem can be a professional relational database and also

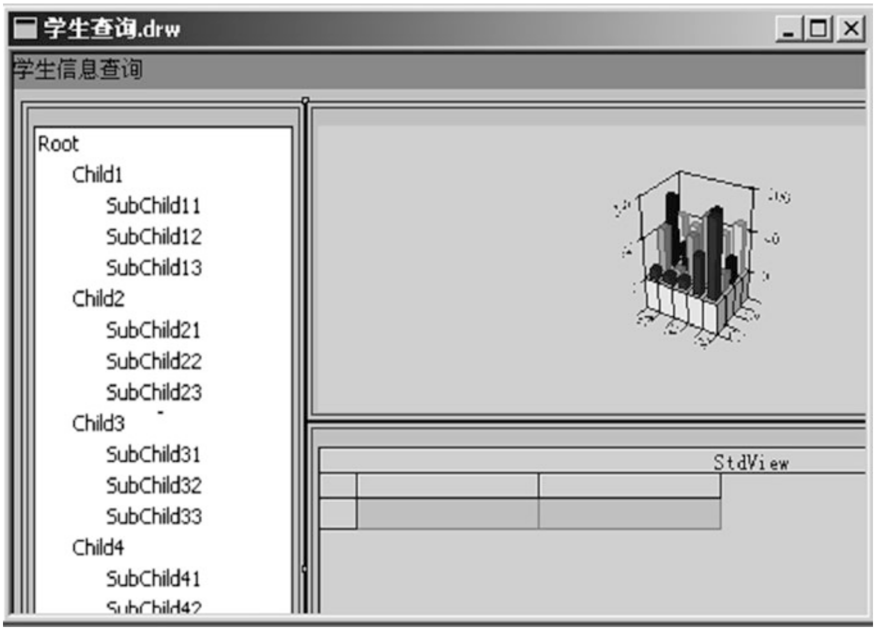
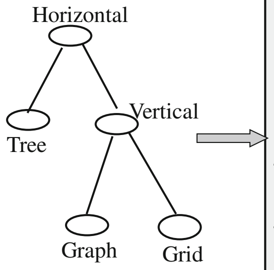


Fig. 3 An instance of interface template

Fig. 4 Layout tree and its XML representation



```
<Uitemplate>
  <Horizontal>
     <APU>Tree</APU>
   <Vertical>
<APU>Graph</APU>
<APU>Grid</APU>
   </Vertical>
  </Horizontal>
</Uitemplate>
```

can be a series of XML files. Query subsystem is responsible for the communication with interaction model and interface template, which provides the support for query. The maintenance subsystem is responsible for the adding, editing, and deleting UI templates in the template library.

4.1 The Composition of Interface Template Library

In the process of constructing the interface template library, the first is the analysis of the appearance, characteristics, and behavior of the overall interface and its elements and analysis. Then, we extract their common features and form general interface templates and abstract description model. They are divided in accordance with the types and levels. First of all, these elements are classified by types, and then, the common characteristics of the same type of elements are taken out and are putted on the upper layer, and so on. Finally, this approach comes down to the root node.

The basic elements of Web interface can be abstracted into five categories: data objects, data collection, querying object, object for control parameters, and object for use case collection. Data object is mainly used for data input and output. Different attributes of the object use different types of control components to display, and this presentation format is called free form. The properties belonging to the same group are located within the same framework to improve the readability of the interface logic. Data collection is a presentation form which is used for concentrated displaying and operating a collection of data objects, which mainly includes free form, tabular form, tree form, graphic form, and the form of object groups. Querying object is used to generate a query condition. When the querying object is a data object, its presentation form is usually a simple control group, which is generally associated with buttons or hyperlinks. When the querying object is a data collection, its presentation style is tables or trees. Control parameter object is used to generate the control parameters, and its presentation forms usually are drop-down list box, checkbox, and other simple control group. The use cases represent related operations, and their presentation form can be ordinary buttons, image buttons, menu items, hyperlinks, and so on. Use case group is a collection of use cases applied to the same data object, of which general forms can be ordinary button group, picture button group, and hyperlink group.

4.2 Matching of Interface Template

In the interface template library, there are many interface templates with similar functions. The one with the highest degree of matching should be filtrated for the user. According to the component querying methods proposed in [5], we take two steps to retrieve the best one.

Firstly, according to the number and type of abstract description controls and of the overall style of the template, the overall framework of the interface template is determined, and a set of eligible templates is identified.

Then, the highest matching interface template is determined by defining the degree of structure matching (SM) and the degree of attribute matching (AM). Herein, SM represents the matching degree of the similar templates in the aspects of layout, etc. AM refers to the matching degree of the similar templates in the aspects of overall properties, associated semantics, etc.

The weight of all leaf nodes of the matching model (noted as MV) is defined as 1, and the weight of the parent node is the sum of the weight of all its child nodes. QR and TR, respectively, represent the structure of the required template and the template structure in the database. QA and TA, respectively, represent the attributes of the required template and the template attributes in the database. M(QR,TR) represents the matching degree between the querying condition and the atomic structure of templates in database, while M(QA,TA) represents the matching degree between the querying condition and the properties of templates in database. The value is set to 1 if match and 0 if not.

The formula for SM degree is shown as follows:

$$\mathrm{SM} = \frac{\sum_{i=1}^{n} M(\mathrm{QR}_i, \mathrm{TR}_i) * \mathrm{MV}_i}{\text{Number of }(\mathrm{TR})} \tag{1}$$

The formula for AM degree is shown as follows:

$$\mathrm{AM} = \frac{\sum_{j=1}^{m} M(\mathrm{QA}_j, \mathrm{TA}_j) * \mathrm{MV}_j}{\text{Number of }(\mathrm{TA})} \tag{2}$$

Thereby, the formula for the whole template matching degree is shown as follows:

$$\mathrm{Match}(T) = \mathrm{SM} + \mathrm{AM} \tag{3}$$

Based on the above formulae, the most qualified template can be calculated relatively easily.

5 Verifying with Code Generator

Code generator generates a specific Web page according to the layout information, presentation style, and description parameters of interface template. The detailed process is shown in Fig. 5.

First of all, presentation model gets the name of an abstract description control widget from interaction model, and then, it is placed into the template query subsystem; On the basis of input information, the system begins to query in the library of interface templates. The result will be stored in the form of XML text block. At the same time, the user creates a new interface template and sets the parameter and layout according to the user's requirements. The parameters and layout set are saved in the XML file describing the interface template. And then, the XML text chunk inquired is integrated into the template XML file according to

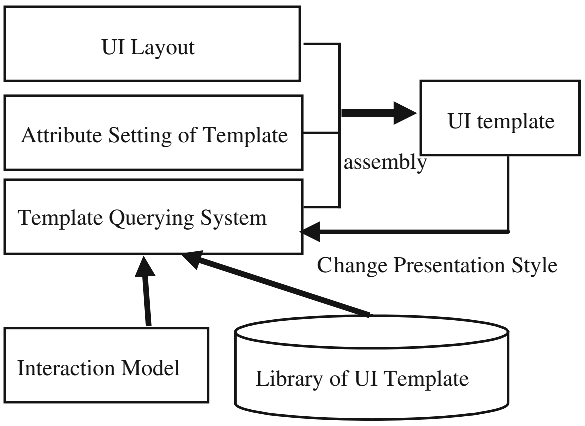


Fig. 5 Assembly process of UI template

the layout mode, and the abstract description control is displayed in the template. And the user modifies the relevant parameters and presence according to the circumstances of each case. Parameters modified are saved in XML files describing the interface template.

In addition, users also can directly match the eligible common template in the template library according to the number and type of abstract interaction control widgets from interaction model, thus without having to manually set the layout and property.

The approach allows the user to reuse the interface in the process of the modification. That is to say, the user can choose a ready-defined template according to the situation and insert it into a zone of another template, or copy an existing template when creating the new interface template in order to improve the reusability of interface template and the degree of automatic code generation.

In code generation, XML files of the template will be passed to the code generator to provide the required parameters for generating Web interfaces. The finally generated codes are separated from the model. If the generated user interface is not satisfied, it can be redesigned by modifying the parameters of the XML files or by directly adjusting the model in the model editing environment.

Model designer can use a professional XML editor to describe the new controls or the interface template and store these new abstract description controls and interface templates into template library according to their classification. By using editing tool, the designer can preview the abstract description controls and UI template, thus to make sure if they meets the requirements.

We have designed a Web application development system based on FMP, all of the programs are achieved within the environment of Microsoft^{TM} Visual C++.NET. At present, automatic code generation has been completed from high-level model to J2EE and ASP.NET. Figure 6 shows an example of interface template of adding employees in a human resource management system, of which the display area is formed by the three subregions. The running page of the generated Java codes on the Struts framework is shown in Fig. 7.

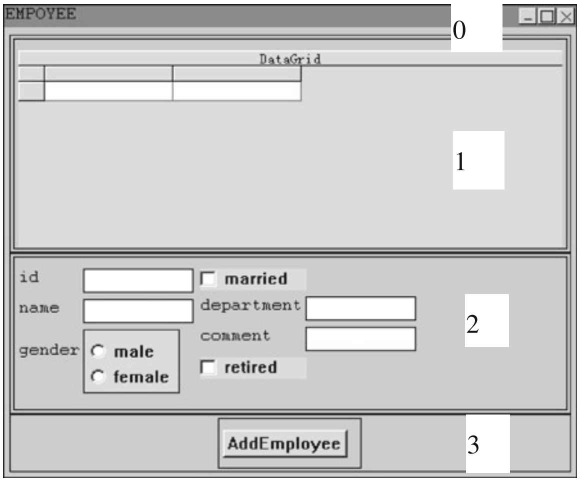
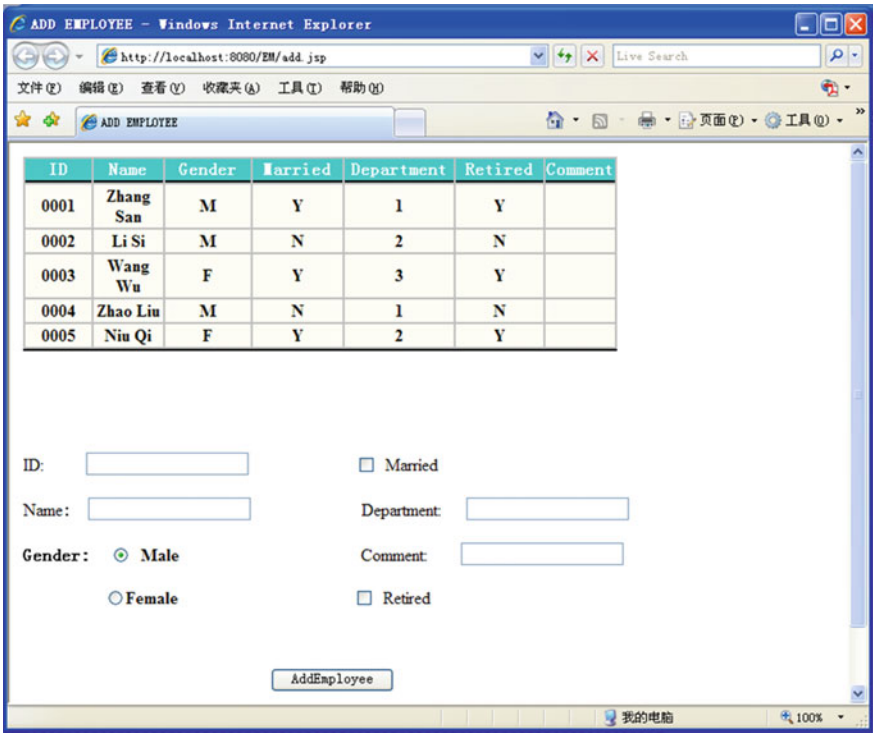


Fig. 6 An example of interface template—adding employees


ADD EMPLOYEE - Windows Internet Explorer
http://localhost:8080/EM/add.jsp
Live Search
文件(F) 编辑(E) 查看(V) 收藏夹(A) 工具(T) 帮助(H)
ADD EMPLOYEE
页面(P) 工具(O)
ID:
Name:
Gender:
Male
Female
Married
Department
Comment
Retired
AddEmployee
我的电脑
100%

ID	Name	Gender	Married	Department	Retired	Comment
0001	Zhang San	M	Y	1	Y	
0002	Li Si	M	N	2	N	
0003	Wang Wu	F	Y	3	Y	
0004	Zhao Liu	M	N	1	N	
0005	Niu Qi	F	Y	2	Y	

Fig. 7 A running instances—adding employees

It can be seen from this example that the approach can provide supports for automatic generation of Web user interfaces.

6 Conclusion and Future Work

In this paper, a user interface modeling approach is provided on the basis of XML and interface template, and the reusing of interface design patterns is achieved. The approach can provide an effective support for model-driven software engineering [6]. As far as our future work is concerned, a comprehensive extraction and description of Web interface model will be conducted to further enhance the visual appeal of the generated Web pages.

Acknowledgments The author is most grateful to the anonymous referees for their constructive and helpful comments on the earlier version of the manuscript that helped to improve the presentation of the paper considerably. This research was supported by the foundation of science–technology development project of Shandong Province of China under Grant No. 2011YD01042 and Grant No. 2011YD01043.

References

1. Won K, Don JF (2010) User interface presentation design assistant. In: Hudson S (ed) Proceedings of UIST'10, New York. ACM Press, New York, pp 10–20
2. Pedro S, Piyawadee S, Palbo C et al (2008) Declarative interface models for user interface construction tools: the MASTERMIND approach. In: Proceedings of engineering for human–computer interaction. Chapman & Hall, London. IEEE Press, New York, pp 120–150
3. Silvapp DA, Paton NW (2009) User interface modeling in UMLi. IEEE Softw 20(4):62–69
4. Wan J, Sun B (2011) Interface model to support automated generation of user interface. Comput Eng Appl 39(18):730–736
5. Yan L, Feng Y, Song Y (2010) Reusable component classification and querying method. Comput Eng Appl 39(6):85–87
6. Kleppe A, Warmer J, Bast W (2009) MDA explained, the model driven architecture: practice and promise. Addison-Wesley, Boston