2017 2nd International Conference on Computer Science and Technology (CST 2017)
ISBN: 978-1-60595-461-5

NextUI Technique-Based Internet Financial Business Client

Liou YUAN[a]*, Jin-yuan ZHANG and Gang YU

yuanliou@cmbc.com.cn, Beijing, China

zhangjinyuan@cmbc.com.cn, Beijing, China

yugang@cmbc.com.cn, Beijing, China

[a]yuanliou@cmbc.com.cn

*Corresponding author

Keywords: XML, NextUI, Declarative interface language, Internet financial business client, UI

Abstract. This paper proposes a NextUI model through XML graphical interface description language upon the basis of MyGUI open-source model, it achieves the separation of user interface design and business logic, and solves the problem of cross-platform problem as well, which dramatically improves the efficiency of maintenance and update for financial application system. The practical project reveal that our NextUI framework can make the complex financial business UI requirement become simple and fast to complete, resulting in decline on the risk and cost of system development. Also, it has good portability and reproducibility.

Introduction

With the innovation of financial business, the graphical user interface of financial front-end business system must satisfy higher demand, it might be flexible and on-demanded. During the entire process of development on online financial client, the time taken by transaction interface development costs a larger proportion. Traditionally, The development model based on a variety of instructional languages, e.g. C++, Java, etc., not only requires developers to be familiar with the programming language graphics library and focus on the process of interface generation, but also makes the interface code and business logic code mixed, leading to difficulties in system testing, maintenance, inadequateness of rapid changes in business[1,2]. Traditional C++ interface models, such that MFC is built on the Windows API, it can make the programmer work easily, generate smaller program, bring higher programming efficiency, as well as provide object-oriented programming. Nevertheless, the generated interface looks ugly, the executable program requires MFC runtime to run properly and does not support cross-platform; QT can support cross-platform, object-oriented completely and expanding easily, allowing component programming, however, its library is relatively strong and needs to cut off when necessary, hence the compilation is slow and widget efficiency is small; GTk + is an open-source framework model with the properties of good design, flexibility, scalability and comfort international and local supporting, including supporting on XML interface language description, however, GTK model is very sophisticated and hard to employ. For traditional Java interface models, although AWT is fast and stable with less resources, it is not extensible, failing to provide important components such as tables and trees, also the buttons cannot support images; SWT has a wealth of component types and component features, as well as responses fast with less memory consumption, but it does not support cross-platform, portability and extensibility, function cannot guarantee to be stable as well, non-windows platform performance is poor; Swing has a wealth of component types, as well as good flexibility, scalability and compatibility, it follows the MVC model, enabling the separation of business logic and interface design, and supports cross-platform with adequate compatibility, but Swing occupies more memory resources to degrade its performance[4].

In contrast, the declarative interface language only requires the programmer to describe the interface as a separate document, then it makes the runtime engine interpreting document into an interface. XML is a markup-based structured language with good scalability, structuration and format consistency of data description[5,6]. As a meta-language, it provides general syntax and structure to describe other domain-related languages. Since XML has the neutrality of language and implementation, it keeps in line with the cross-platform, and makes the interface description statement adequately separate from the logic implementation code. In recent years, XML-defined declarative interface language-XML UI technology has become a hot issue, XML-based interface description languages (UIDL) have emerged in a large number. The UIML proposed by the Virginia Human-Computer Interaction Technology Center implements a cross-platform interface description mechanism. The International Information Technology Standards Committee presents AAIML aiming at equipment independence. Microsoft has proposed XAML and applied it to the Vista interface and Silverlight development. Adobe's Flex technology also makes full use of XML declarative interface description and development solution. Mozilla has developed XUL and utilized by the fireFox browser[7].

At present the open source platform has emerged a lot of XML-based GUI model, such that, CEGUI is powerful but too large, thus it is complicated to use. MYGUI [3] is an efficient, lightweight, flexible model, but it is vulnerable to launch multiple languages. To address the problems confronted with financial network transaction client in development and maintenance, upon the basis of MYGUI open-source model, we study the NextUI model and successfully apply it to the development of a bank's online banking assistant client project. NextUI model realizes the separation between interface and logic, and supports cross-platform simultaneously.

XML - Based Graphical User Interface Description

Graphical user interface, as a human-computer interaction mechanism, has become an essential component of present software, and the development of graphical user interface language can be divided into two categories: instructional language and declarative language. The instructional language asks the programmer to explicitly indicate how the interface is generated, such as C++ and Java interface code that require the programmer to be familiar with the programming language's graphics library, which often causes the interface code to be mixed with the business logic code, as well as makes software maintenance difficult. The declarative language only requires the programmer to specify what is generated on the interface and describes the interface as a separate document, then it interprets the intermediate language as an interface presented to the user by the parsing engine at runtime.[9] In such a way, the interface code is independent of the business logic code and maintenance costs decline validly.

XML is a markup-based structured language with good scalability, structuration and format consistency of data description[10]. As a meta-language, it provides general syntax and structure to describe other domain-related languages[8]. The interface elements of the data structure and display are described by XML so that to separate the data structure and display phase, this greatly improves the performance efficiency; The use of XML to describe the interface elements also defines the self-description interface of the interface elements, making it is possible to reuse the code and interface and to construct the interface. Figure 1 shows the relationship among user interface, business logic and controller in declarative interface description language. The user interface only describes the interface elements, business logic is responsible for the realization of background business, while the controller is responsible for responding to user requests and mapping user action into business logic.

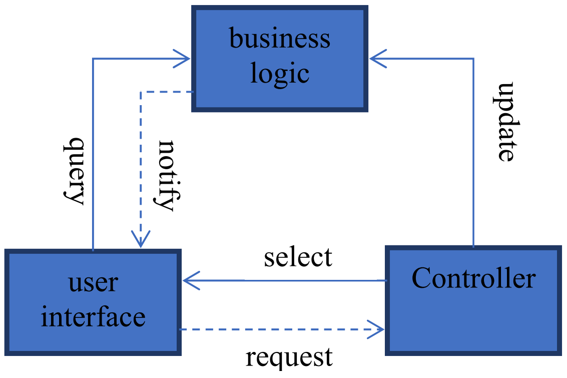

Figure 1. User interface, business logic, controller diagram.

Figure 2 states the implementation of graphical interface description model based on XML, XML description of the interface elements and XML business data are parsed by parser or rendering engine, and displayed in the interface.

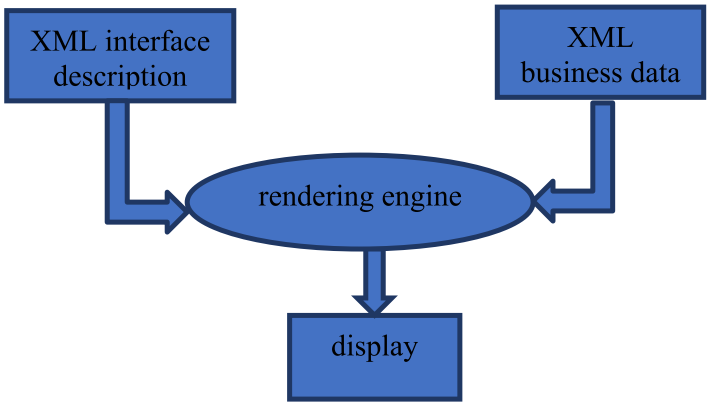

Figure 2. Graphical interface description model.

XML Graphical User Interface Description Method Based NextUI model

The NextUI model is a graphical user interface based on the open-source UI model MyGUI. It concludes following characteristics: i) interface and logic are separated, layout is flexible and diverse; ii) built-in commonly used controls; iii) support container properties and custom controls; iv) support png / gif / jpg / bmp and other image formats; v) support for alpha blending and image transparency; vi) support the dynamic hue transformation; vii) support plug-in system; viii) support a variety of animation effects; ix) possess powerful layout, resources and other management system; x) have powerful event handling mechanism; xi) support hardware acceleration and rectangle Drawing technology; xii) support for dual engines (DirectX and OpenGL); xiii) support cross-platform.

Some of the modeling elements and XML descriptions in NextUI are as follows:

(1) Layout: that UI interface elements of the typesetting information, XML is described as follows:

```
<NEXTUI type="Layout">
<Widget type="StaticImage" skin="StaticImage" position="215 300 371 98"
align="Center" layer="Main">
<Property key="Image_Resource" value="pic_AccountsBack"/>
<Property key="Widget_NeedMouse" value="true"/>
</Widget>
......
</NEXTUI>
```

(2) Skin: Description of the graphical interface elements how to show, XML description is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<NEXTUI type="Resource" version="1.1">
   <Resource type="ResourceSkin" name="CheckBoxSkin" size="23 21"
texture="NEXTUI_BlueWhiteSkins.png">
      <BasisSkin type="SubSkin" offset="0 0 21 21" align="Left Top">
         <State name="disabled" offset="2 2 21 21"/>
         <State name="normal" offset="2 24 21 21"/>
         <State name="highlighted" offset="2 46 21 21"/>
         <State name="pushed" offset="2 68 21 21"/>
         <State name="highlighted_checked" offset="2 134 21 21"/>
         <State name="pushed_checked" offset="2 156 21 21"/>
      </BasisSkin>
      ……
   </Resource>
   ……
</NEXTUI>
```

(3)Theme : used to set the theme of the use of the skin, XML is described below

```
 <?xml version="1.0" encoding="UTF-8"?>
  <NEXTUI>
  <Tag name="NEXTUI_Theme_Texture">core.png</Tag>
  <Tag name="NEXTUI_Theme_Button_ColourNormal">#000000</Tag>
  <Tag name="NEXTUI_Theme_Button_ColourPushed">#000000</Tag>
  </NEXTUI>
```

(4)Template: the essence is UI layout, it describes a set of control layout information. The template abstracts the repetitive things, making the reuse more convenient.

```
<?xml version="1.0" encoding="UTF-8"?>
<NEXTUI type="Resource">
   <Resource type="ResourceLayout" name="CheckBox" version="3.2.0">
      <Widget type="Widget" skin="CheckBoxSkin" position="20 20 23 21"
name="Root">
         <Property key="TextAlign" value="Default"/>
         <Property key="FontName" value="Default"/>
         <UserString key="LE_TargetWidgetType" value="Button"/>
      </Widget>
   </Resource>
   <Resource type="ResourceLayout" name="RadioButton" version="3.2.0">
      <Widget type="Widget" skin="RadioButtonSkin" position="20 20 21 20"
name="Root">
         <Property key="TextAlign" value="Default"/>
         <Property key="FontName" value="Default"/>
         <UserString key="LE_TargetWidgetType" value="Button"/>
      </Widget>
     </Resource>
      ……
</NEXTUI>
```

(5)Layer: Used to set the interface vertical layout information. The XML representation of the layer is:

```
 <?xml version="1.0" encoding="UTF-8"?>
 <NEXTUI type="Layer" version="1.2">
   <Layer type="SharedLayer" name="Wallpaper">
   <Property key="Pick" value="false"/>
   </Layer>
   <Layer type="OverlappedLayer" name="Overlapped">
   <Property key="Pick" value="true"/>
   </Layer>
```

(6)Font: Used to define the font used by the interface, the XML representation of the font is as follows, describes the shared layer, the overlapping layer font type:

```
 <?xml version="1.0" encoding="UTF-8"?>
  <NEXTUI type="Resource" version="1.1">
  <Resource type="ResourceTrueTypeFont" name="font_simhei.14">
  <Property key="Source" value="simhei.ttf"/>
  <Property key="Size" value="16"/>
  <Property key="Resolution" value="50"/>
  <Property key="Antialias" value="false"/>
  <Property key="OffsetHeight" value="0"/>
     <Codes>
    <Code range="33 126"/>
    <Code range="8216 8217"/>
     <Code range="8230"/>
     <Code range="12289 12290"/>
     <Code hide="128"/>
     <Code hide="1026 1039"/>
     <Code hide="1104"/>
     </Codes>
     </Resource>
     </NEXTUI>
```

Each interface element in the NextUI model is a container, and the container is composed of a basic control or other container. The above key modeling elements can be expressed as two independent modules, spatial information and visual information. Layer, layout, template are representation of space information of the interface: Layer is a vertical layout, the layout can use the template, the template can also contain layout, templates and layouts are included; Themes and fonts provide material for the skin.

Skin control visual information, layout control spatial information, and the two together form the final visual effects.

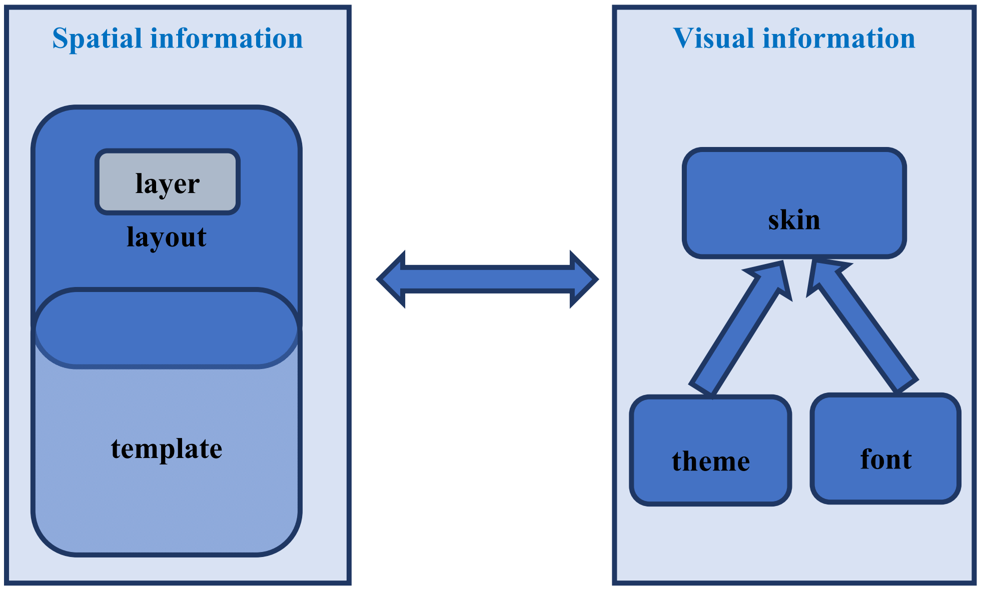

Figure 3. Interface modeling key elements of the relationship diagram.

NextUI Framework Adopted in Network Financial Transaction Client Development

Most of the network financial clients utilize the traditional UI model. Currently, the traditional UI model performs poorly with respect to security, interface flexibility and stability. The interface designed by the model is user window hierarchy, Figure 4 describes the traditional UI design framework, This framework has many limitations: UI library control depends on the system resources, resulting in cross-platform cannot be achieved; Thanks to the scenario that interface and logic are not separated, when the UI interface is adjusted, the application needs to be re-encoded, compiled, installed, leading to high maintenance overhead; Interface display is not flexible enough, and security is poor.

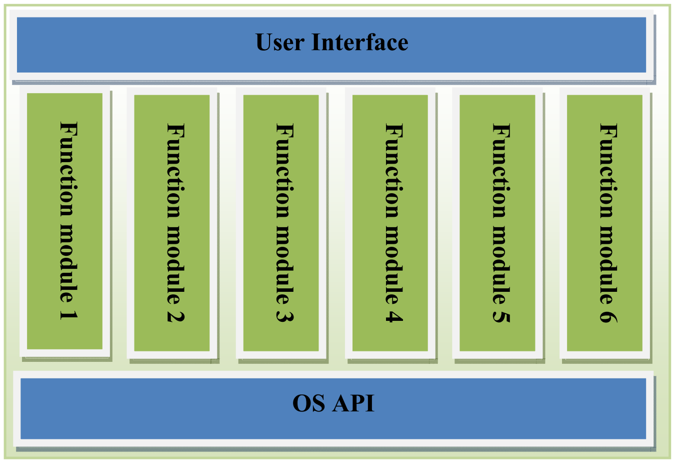

Figure 4. Traditional UI design model.

The NextUI model solves the problems in the traditional UI model above. First, NextUI uses XML-based graphical interface description to completely separate the interface from business logic; NextUI supports container features and custom UI controls without relying on system resources, thus achieves cross-platform; While adjusting the UI interface, without re-encoding and other steps, we just need update the interface UI configuration file, and maintenance costs can also be reduced correspondingly.

Figure 5 presents the design framework for financial network client based on NextUI, which is a stable, powerful bottom built on the Windows system API and NextUI engine; Multi-task manager is used in application layer to coordinate the different work services to ensure the orderly and smooth operation of the program; Resources manager is used to manage the GUI and other resources; Functional modules are designed independently to reduce the coupling as much as possible, so as to ensure the follow-up extension and scalability; The interface and logic are designed to separate in GUI part, guaranteeing layout flexible and diverse, easy to modify and to expand; This GUI interactive system can be ported to Mac, iOS, Android, Linux platform, and achieve cross-platform features; The clients support all versions of Windows OSs from XP to 10, and compatibility works well.

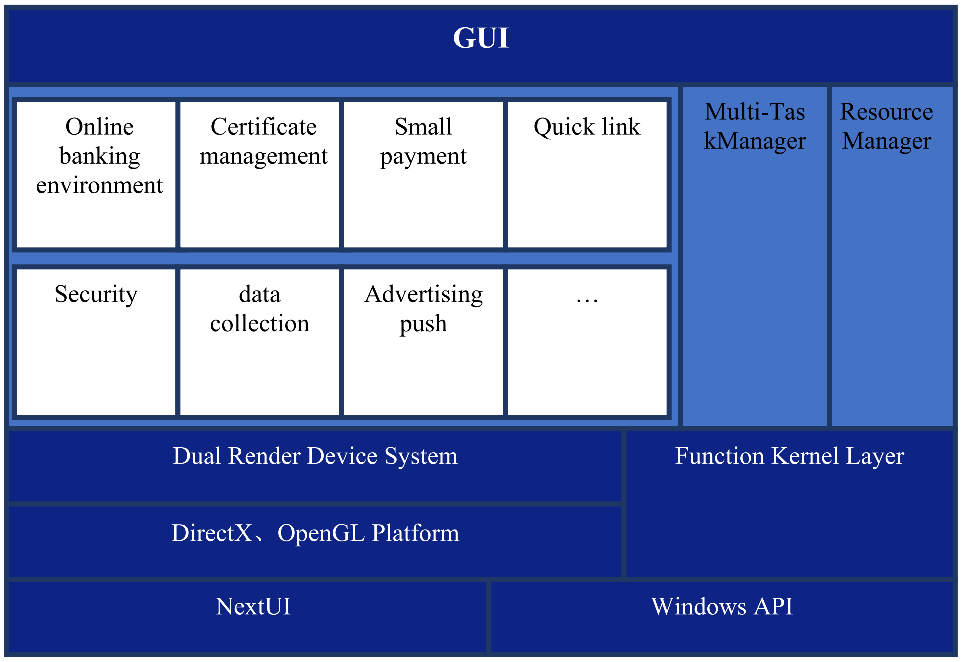

Figure 5. NextUI design framework.

The rendering engine-based UI model has undergone three generations of change. The first generation of UI model uses browser kernel as render engine, it suffers from vulnerability. The second-generation model adopts custom GUI kernel; however, it still relies on system resources. The third generations of UI model customize controls and engines, and achieves the separation of business and logic. Table 1 compares the three generations of UI models, which assumes that the first-generation UI model maintains one interface change as one cycle. If the first-generation UI model maintains one interface change for two weeks, furthermore, the efficiency of the maintenance of the second-generation UI model can be doubled, only 1 week, additionally, the NextUI model can be doubled than the second-generation UI model, just 0.5 weeks.

Table 1. UI model comparison.

	The First Generation UI model	The Second generation UI model	NextUI model
Interface maintenance cycle	1 unit cycle	0.5 unit cycle	0.25 unit cycle
Interface maintenance costs	High cost, need to re-design interface, code, compile, install	High cost, need to re-code, compile, install	Low cost, just modify the interface configuration file
Rendering engine	With the browser kernel, cannot combine well with the control, business and interface is not separated	Customize GUI kernel, rely on system resources, business and interface is not separated	Out of system restrictions, the components are all containers, business and interface is separated
safety	Use js script to implement control operations, source code can be easily obtained, unsafe	Security is Slightly improved, but can be cracked by capturing the form properties	High security; the form properties cannot be captured if you are not familiar with the layout of the form; difficult to crack
performance	Do not support multi-threaded, response slowly, not support animation	Performance is slightly improved, support multi-threaded, support simple animation	Better performance, support multi-threaded, use hardware acceleration technology, support complicated animation
stability	Unstable, running is not smooth, interface style is inconvenient unified	More fluid, Restricted to system resource, the display is not flexible	Stable, not limited by system resources, display flexible, support translucent and other special effects
portability	Do not support cross-platform	Do not support cross-platform	support cross-platform

Figure 6 is the main interface of an online banking assistant client using NextUI model. The interface layout of this client is more flexible, and interface adjustment can be conducted without modifying the main program. The usage of Modular interface layout can present the most concerned information to user. It is easy to provide users with Internet-like operating experience. A separate advertising module is provided to facilitate the bank's business promotion; Interface layout can be flexibly expanded to facilitate the follow-up new features; interface effect is more abundant, which includes translucent display, state disk of various dynamic effects, Page, function blocks, buttons and other progressive movement to show and hide the effects of animation, the effect of transparency gradient and page transparency asymptotic hiddenness and appearance, etc. Basic functions are fully upgraded. Maintenance of USBKey driver and online banking control are comprehensively upgraded, control is more Flexible; Client online banking environment and other strategies are configured and released by server, which effectively reduces the maintenance costs.

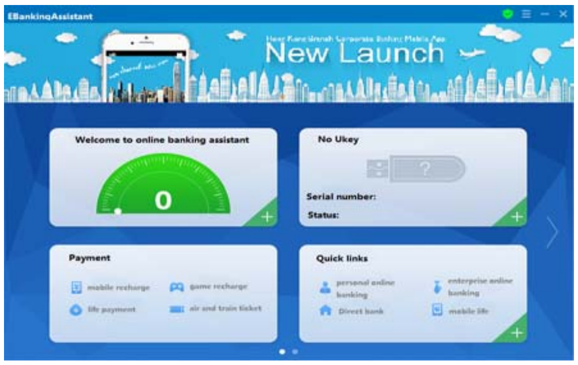

Figure 6. main interface of online banking assistant client implemented with NextUI.

Conclusion

Taking into account XML is a self-explanatory and self-descriptive language, based on the XML graphical user interface description, this paper developed the NextUI model upon the basis of MyGUI open-source model, and achieved good application effect in a bank online banking assistant client project. The practice scenario shows that the NextUI model realizes the separation of transaction interface design and business logic to support the parallel development fashion of interface and logic. The developed user interface is independent on the system wherein it is located, and it is completely free from the background system, correspondingly, the cross-platform problem is solved as well. Therefore, rapid changes and sophisticated financial business UI requirement can be simplified and fast for achievement, reducing the risk and cost of system development, enabling good portability and reproducibility.

References

[1] QH Meng. 2009. Research on financial transaction client framework based on RCP and XML UI technology [D]: [Master] Hefei: University of Science and Technology of China.

[2] QH Meng. 2008. Transaction-based language and run-time engine based on XML [J]. Journal of Computer Applications, (11): 57-61. http://MYGUI.info/

[3] H Huang, H Lin, B Wang. 2011. A Graphical User Interface XML Description Method and Tool Development [J]. Journal of Computer Applications and Software, 28 (10): 198-202.

[4] W Wen, SJ Cao. 2008. A description of the expression of GUI - based user interface GUI without coupling[J]. China Water Transport, 6 (1): 189-191.

[5] LK Wang. 2011. Design and Implementation of Dynamic Graphic Interface Based on Automatic Generation of XML [D]: [Master]. Chengdu: University of Electronic Science and Technology of China.

[6] YJ Qi, JL Li. 2007. Research on Application of Graphic Interface Development Language XUL [J]. Computer Technology & Development, 17 (3): 233-235.

[7] T Channonthawat, Y Limpiyakorn. 2016. Model Driven Development of Android Application Prototypes from Windows Navigation Diagrams[C]. International Conference on Software Networking, 2016:1-4.

[8] NA Almonte, WR Stubbs. 2014. Multi-monitor, multi-JVM java GUI infrastructure with layout via XML. Free Patents Online.

[9] N Souchon, J Vanderdonck. 2003. A Review of XML—Compliant User Interface Description Languages[J], Lecture Notes In Computer Science,2844,39 1-40 1.