Research on the windows interactive operating system in view of WEB simulation

Qing Cui, Gang Shi, Shuxian Liu, Chongguo Wang
School of Information Science and Engineering, Xinjiang University, Urumqi 830046, China
luanuq2003@163.com

Abstract. The system uses Web simulation to simulate the interaction of the Windows operating system, and establishes a virtual Windows interactive operating environment in the browser. This system can not only be used as an auxiliary means of teaching, but also can establish a Web-based simulation examination system to enhance the comprehensiveness of the assessment content. This article uses API functions such as Find Window and hook functions to obtain the properties and sub-objects of dialog objects for the Windows operating system, and to establish an inter dialog call diagram, and then apply the method of Creating Objects dynamically by Silverlight. The characteristic of this study is that different versions of Windows can be generated with dialog properties, and different versions of the simulation system can be generated automatically, with strong reusability and extensibility.

1. Introduction
Software simulation has been an important direction in the research of virtualization and visualization in recent years. In the process of teaching and evaluating Windows operating system, practical operation is very important, such as setting the network environment or changing system attribute parameters. However, these operations often change the experimental environment, and the wrong system configuration can cause the device not to function properly. A restore card is usually installed on an experimental device in the school's computer room in order to protect the computer from returning to its original state. However, in assessing the actual operation of students, this method often can not play a correct evaluation effect. The interactive interface system of Windows operating system based on Web simulation can solve the problem of evaluating students' operational ability by realizing the simulation of the operating system through the browser on the one hand, and obtaining the process and results of the student's operation. In practice, we have developed the "Web-based simulation of The Windows Operating System Application Examination System", and applied it to the basic examination of computer culture in universities, and achieved good results.

The user interface of the Windows operating system consists of four main parts: the desktop, which includes menus, icons, and taskbars; resource manager; browser; Interactive dialog form, which is the most used in. However, to simulate the Windows operating system is a very difficult thing, on the one hand, the Microsoft Windows system updates relatively fast, and versions needing to be tested are very more; the second, the substance is more that being related to the configuration in Windows operating system, and all dialogue forms and character are complicated. These cause it too heavy that realizes the Windows operating system by the directly using the hard coding, and with the release of the update will have to be re-implemented, but the reusable parts are relatively small [1-2].
2. Getting Windows' Interface Information

2.1. The interface information needed to be obtained

Using the API function in Windows, automate to obtain the Windows dialog box and its properties, analyze the call relationship between the objects, and dynamically generate these objects on the page by the graphical display technology of the Web, and realize the Interactive Simulation system to dialogue form base the Windows Operating System in view of WEB. The Silver Light technology is an important way to realize graphics functions efficiently and quickly on the Web.

It is possible to directly realize the interface of Windows using encoding. But the reusability of its code is usually inappropriately because of many versions of Windows, and constant updates. The Windows interactive interface automated reconstructed obtains information of Windows interactive dialogue form completely depending on user’s interactive operation when there no Windows source code. Simulating Windows interaction dialog slots in the Web requires the following basic information:

1) The type of object: to create different simulation objects based on different types.
2) The handle of the object: This is the identity of the current object in memory.
3) Information of the object size and position: Used to restore the size and position of the object.
4) The title of the object (caption): Used to restore the object, and distinguish the object within a certain range.

2.2. Process acquiring properties of dialog element

First, using the function named GetDesktopWindow() to get the handle of the desktop window; Second, recursing to obtain the handle to all Windows objects and the containing controls in current using the GetWindow() and FindWindowEx(), and record the dependency, and then obtain the metadata of the window object using the functions of GetWindowText(), GetClassName(), GetWindowRect() and other functions.

A partial metadata result of the objects actually obtained by studying as shown in Table. 1.

Table 1: Gets partial results of Windows Interface

| Handle   | ParentHandle | ClassName | Caption     | Left | ... |
|----------|--------------|-----------|-------------|------|-----|
| 65552    | 0            | #32769    | 0           |      |     |
| 65636    | 65552        | Button    | Begin       | 0    |     |
| 3016488  | 65552        | #32770    | Grab window | 676  |     |
| 3148038  | 3016488      | SysListView32 | List1  | 697  |     |
| 1049312  | 3016488      | Button    | Close       | 960  |     |

As you can see from Table 1, all interface elements start with the desktop and the parent handle for the desktop is 0. The dependency relationship between window objects can be seen in Parent Handle. In Table 1, the "grab window" is a dialog box for a window class with a list and a close button in its sub-objects. The Class Name of the window class in the table is an integer that begins with #.

The dependent relationship of the window object obtained by the acquisition tool can be represented as a dependency containing tree as shown in Fig. 1.

Fig. 1 The relationship containing Windows interface element
In Fig. 1 the container window is some container that can contain other windows, and it can include other sub windows. Each window consists of some basic element controls. The desktop is the parent window of all windows, which also contains the basic elements of controls such as the Start button.

However, from the result of a single fetch we can only directly analyze the object of the current running state, and can not get all the objects. It is difficult how to get all objects this process. Since can't starting with the source code, and may turning to different dialog objects because of different inputs when running, some dialog boxes can only be displayed in special input situations, such as the prompt box of the IP address conflict, how to deal with this?

In order to get the object information for all the dialog boxes, we use the method of manual multi-time and to merge it. Although this process is not complete, we basically get the property information of all dialog objects and their sub-objects in the Windows Control Panel through many large operations.

It is difficult to merge process after obtaining object information by multiple people, due to the difference of the running environment, and it is not consistent to obtain the process ID and handle of the same object. If each subordinate letter obtained is expressed to $T_i$, and Combining $T_i$ and $T_j$ of two trees, it is requires determining whether the leaf nodes in the two trees are the same object, which must be determined by the parent Handle, Handle, Caption, and Class properties of each node of the two trees. To compare starting with the root node layer by layer and down, and eliminating the same object. By merging multiple times, you will finally obtain the dependent diagram of the a basically complete Windows object with.

3. Get the calling relationship among windows object

3.1. Importance and method of the calling relationship

The interaction function of the simulation windows under Web page, in addition to being able to display dialog boxes and their sub-objects, it is more important to be able to navigate among dialog boxes.

It is possible to manually establish navigation relationships among dialog boxes. But in practice we take another approach: saving the calling relationship among dialog objects when grabbing an object. The call procedure for acquiring an object is realized through a hook function, based on the mouse event that captures the object, establishes an event log, records the object handle and the new window object handle displaying that is clicked on each mouse event.

This process is more cumbersome, in order to confirm whether the new window, it is needed to start again to get the object tree based on desktop, but because of the consistency of the before and after environment in this time, the existing window handle is the same, can directly skipping the existing dialog box, directly to find out whether there is a new dialog box.

3.2. Hook functions related and the grabbing process

Hook of Windows is a better method that Windows capture the events occurring in own process or others. The hook function is a callback function installed in the specified Windows event through hooking.

The procedure for getting window objects and calling relationships through callback functions. Saving calling relationship refers to a new window object that is newly established by the mouse clicking the navigation control (button or hyper-chain) in the calling window. This allows to get the calling relationship among windows while getting the window.

Getting an object each time, can get a calling relationship diagram. This diagram includes two relationships: the one is a parent-child inclusion relationship, Which is navigation controls of the window and internal buttons, hyper chains and so on. The second is the call relationship, to another window from one window navigate controls. In a navigation relationship, sometimes another dialog box is not started, and this is the call from itself to itself. Such as clicking the Save or Apply button. Sometimes may reduce dialog boxes, through cancelling or closing buttons.
We mark the call relationship graph generated by different users as $G_i$, which is obviously not complete. How to get a complete entire call diagram? This requires a merge of the diagrams.

Represented by the $G_i$ as an adjacency matrix. Can directly using being added matrix to merge call diagram, there are:

$$G = \sum_{i=1}^{n} G_i$$

(1)

The $G_i$ here is the adjacent matrix for each call.

Manual acquisition of a few call diagram $G_i$, is not complete, but in the course of many experiments can be considered basic complete. This is also helpless without obtaining the Windows source code.

4. Simulating Windows Interface in Web

Microsoft Silverlight is a Web technology based on Net Framework-based. For developers, Silverlight is a Web rendering technology of combining Microsoft's multi-technology. Silverlight provides a development framework based on Net Framework-based. And support seamless integration of images of any size by using vector layer technology Web based on ASP.net, AJAX, etc. It is realized that the seamless connectivity of the Web development environment [3-4].

Silverlight enables development designers to collaborate better and effectively create web applications of the rich and magnificent interface that run in browsers and have a brilliant interface.

It can realize that the process generating dynamically the Windows window dialog box based on the data of the first two stages, However, dynamic generation can greatly burden the server when a user requests a page. In the application of the examination system, concurrent requests are more demanding because the candidates almost all start at the same time. So we use Silverlight code that generates the objects, instead of dynamically generating objects at runtime, then locally optimize them, and compile the deployment to run. On the one hand, this scheme can improve the parallel efficiency of the program and ensure the high-speed response of user requests. On the other hand, you can debug and add local decorations, such as animated pictures [5]. Process of code generation is below:

(1) A hard-coded login and desktop.

System includes a virtual login page, that can leave a inputting interface to candidate login for future candidates. The desktop is the root node of all Windows interactive objects, including too many elements. To simplify the program, we implement these two parts with direct coding. As this is not the focus of this article, there is no discussion here.

(2) Create a standard window object.

Standard window object is a container that hosts elements inside the simulation dialog box. Because the user controls of all dialog boxes need to be generated based on the data of the first two stages, which have many common features, can be dragged, sized, closed, and so on, you actually define a real user control (User Control) and write code to implement the above functionality. This is used as a template for a standard window.

(3) Generates window object code in standard window object according to The Windows interactive object data.

The user control of Silver Light is divided into two files: one is an xaml file, which stores the user interface information in the form of XML text.

Depending on the size of the dialog box and the properties of the internal controls, you can replace the text content of the standard window above. Replace the properties of the User Control tag with the properties of the window, and the controls contained in the window are included in the Grid tag, generating control code based on information such as control type, size, location, and so on. Text is inputted with a Text Block tag, and pictures are displayed as Canvas-contained Image tags.

When building control objects, you should not only pay attention to the consistency of object properties, but also name each resulting object control. Because you must ensure that the resulting object cannot be renamed, the control is automatically named in the form of a "type prefix plus flow number".
When you generate an xaml description of a control object, you must add an event-handling tag to the navigation control, this is determined by the data in the object call diagram.

The event handler, on the other hand, is in the cs file (C# The source program file) corresponding to the user control.

(4) Manual correction and compilation testing.

The automatically generated Silver Light code, which is not perfect in some places, can be modified manually. For example, some dialog boxes in Windows will display some dynamic results, such as pictures. This requires manual modifications, even adding new resources, and so on.

All generated window object files are added together to the project, details are modified, and conciliation and testing can be performed.

5. Experimental Results
The entire project was developed and implemented using Visual Studio 2015. The object crawler is implemented in the form of a Windows application.

First it can be completed that entering parameters, click to start the crawl based on the actual needs of work before grabbing object. Because the mouse event is intercepted using the hook function and the crawl and comparison function is called, the operation should be carried out at a slower than usual input rhythm. In practice, there is little impact on existing functionality.

Silverlight's editing modification tool is designed with Blend for Visual Studio 2015. Build Silver Light code compiles and runs under Visual Studio 2015, implementing the generation of all dialog boxes in the Windows Control Panel, and the login and desktop are implemented in a hard-coded manner. Throughout the process, the display effect of dialog box is consistent with the native effect, and result generated is very well.

6. Conclusion
This article obtains the dialog box and sub-object properties of the Windows interactive operation by constructing a Windows interactive interface metadata, establishes the call graph and dependent diagram among the objects, and then through the experiment of multiple multi-operation channels by multi-people, and effectively merges the experimental results. The call diagram and sub-object property sheet of the complete Windows interaction object are eventually formed. Using this obtained data, the code of the SilverLight emulator is automated, and then the simulated Windows interactive operating system in the Web is compiled.

The method proposed in this paper not only greatly reduces the difficulty of writing the Windows interactive interface emulator, but also applies to different versions of Windows system, which is completely similar to the real Windows operating system in the interactive operation simulation effect, and can change the properties of the object directly by changing the display properties. The resulting interface element code can also be used directly in other systems. This provides a very good basis for the development of Web-based Windows exam system or Web-based Windows teaching practice system.

The method proposed in this paper can also be applied to many simulation program generation sprees based on C/S structure. This will greatly reduce the development cycle of application simulation teaching or demo programs, with good promotional value.

Acknowledgments
This work was financially supported by In 2017 (the second batch) Google supported the Western Computer Education Promotion Program (201702004002) and National Natural Science Foundation of China (61762085).

References
[1] Huang Conghui, Chen Jing, Yu Shuiqing, Chen Minghua. 64 Windows ABI Virtualization Methodology Research[J]. Computer Science, 2014, 41 (1): 39-42(in Chinese).
[2] Hong Wenzhen, Zhou Jinxuan, Liang Huina. Application of open source virtualized desktop in experimental environment deployment[J]. computer education, 2015 (2): 94-98(in Chinese).

[3] Yu Minbin, Pan Li, Yao Lihong. Virtual Machine Monitor System Policy Integrity Security Mechanism[J]. Computer Engineering and Design, 2014, 35 (1): 17-20(in Chinese).

[4] Bi Yimo, Hong Zhenzhen, Lu Chao, Wang Hua. Silverlight-based Red Tide Monitoring WebGIS Technology Research and Applications[J]. Computer Applications and Software, 2014, 31 (9): 60-62, 113(in Chinese).

[5] Li Shaoping. NET Platform Coupling Silverlight Image System Development[J]. Shandong Agricultural University Journal (Natural Science Edition), 2015,46 (1): 106-109(in Chinese).