Design of the Instrument Soft Panel using Windows Presentation Foundation

Feng Lei
Automatic test and control institute
Harbin institute of technology
Harbin, China
hitfenglei@hit.edu.cn

Song Xingzhe
Automatic test and control institute
Harbin institute of technology
Harbin, China
songxingzhe1993@163.com

Abstract—**In order to meet new demands on software designing in the field of testing instrument, this paper implements an instrument soft panel by WPF programming method using Microsoft Visual Studio. The appearance of the soft panel embodies WPF applications' advantages and the soft panel has the utility of controlling HITPE101 data acquisition module through hardware drivers based on VISA. Finally an auto-test system is built to verify the soft panel operates correctly in the design.**

Keywords-WPF; soft panel; hardware drivers; instrument

I. INTRODUCTION

A soft panel is software that people use to replace the operation panel on a real instrument. There are many software development tools for people in the field of test and measurement to design an instrument soft panel. Nowadays, the fact that touching tablet devices appear one after another may lead to a trend of tablet instruments. Also with users' higher requirements on beautifully designed UI (user interface), it is significant for soft panel designers to seek for a new method, which is suitable for developing soft panels with touching operations.

Windows Presentation Foundation (WPF) is a software technology contains the developments of desktop applications, web applications and mobile applications. It was introduced by Microsoft and based on Microsoft .NET Framework. WPF provides a new programming model in common, which separates the task of UI designers and software programmers[1]. Benefit from the new pattern, using WPF's unique XAML(eXtensible Application Markup Language) codes to describe the UI can make soft panel designers customized the controls as their wishes.

II. ANALYSIS OF WPF SOFT PANEL STRUCTURE

According to the WPF programming model, we can use XAML code to implement the appearance of an application while using managed programming languages (code-behind) to implement its behavior[2]. In the structure of soft panel, Microsoft Expression Blend is employed to complete the UI layout and generate the XAML code at the same time, C# is used to describe its process and behavior. Data can be delivered from C# to XAML or otherwise.

In addition, WPF exists as subset of .NET Framework, we must use .NET Framework programming language, such as C# and Visual Basic, to instantiate classes, set properties, call methods, and handle events[3]. However, the hardware drivers are usually programmed by C, which is classified as unmanaged language. Differences between managed and unmanaged language mainly reflect in their compiling mechanisms, the former runs on the .NET Framework that includes a virtual execution system called the common language runtime (CLR) and is compiled into an intermediate language while the latter is compiled into machine language[4]. So, codes written in managed language cannot call a function in unmanaged codes directly as usual. In the structure of WPF soft panel, a DLL (dynamic link library) with hardware drivers generated by C language meets the demand of calling unmanaged codes in managed codes.

The overall structure of a WPF soft panel is illustrated in the following Fig. 1[5].

The part in the dotted box shows the hardware drivers calling relation, for HITPE101 data acquisition module, drivers are programmed based on VISA (Virtual Instrument Software Architecture). As a result, the DLL must cover the content of VISA I/O library.

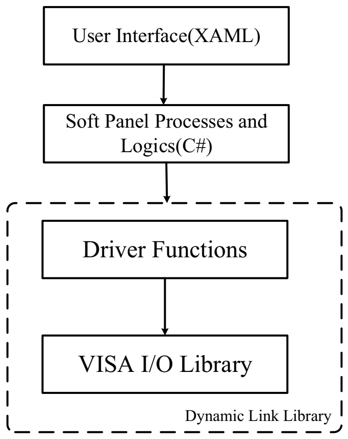

Figure 1. Structure of WPF soft panel

III. Design of Soft Panel Controls

Obviously, the controls that we used before, such as knobs and radio buttons, cannot make users comfortable to do some touching operations. So the first thing needed is changing these controls into a new form that we can easily handle with our fingers. Knobs are replaced by sliders, and radio buttons by toggle buttons. Some of the parameters hardly described with a control, are all determined by a virtual keyboard. The keyboard is available only when users want to change these parameters and click the 'edit' button.

Moreover, in WPF programming model, XAML simplifies creating a UI for an application. WPF provides Style objects and template objects as a way to define the visual appearance of an element in resources. We can design appearance-customized controls with their primary functions. Thus, there are much more choices for designers to extend their UIs. Here is a XAML tree view of a simple customized button Style shown in Fig. 2.

The Style contains the definitions of property, template and trigger.

Property defines the basic features of the control. Template defines the template control Style taken as the base class. Using Setter method, the template can be re-arranged and re-designed as programmer's like. If the control need to be used like a container, it is necessary to add a Content presenter to the template. Trigger defines the behaviors when some specific events happen. Using Setter method as well, the controls may change their appearance when the button is pressed or other operations are taken.

All of the Style contents in figure 2 are defined in a Resource Dictionary included in the WPF soft panel solution. Attribute syntax or property element syntax can be used to reference the Style.

IV. Controlling an Instrument Module

On the basis of WPF soft panel structure mentioned in part Ⅱ, soft panel controls the hardware module through the functions defined in the DLL of the HITPE101 data acquisition module drivers. We can call these functions (or methods) to accomplish operations, such as module initialized, reading or setting sample parameters and getting data acquired by the analog channel. To call these functions, a DllImportAttribute class must be instantiated. By analyzing how to use DllImportAttribute class, a general solution of calling unmanaged codes in managed codes will be found.

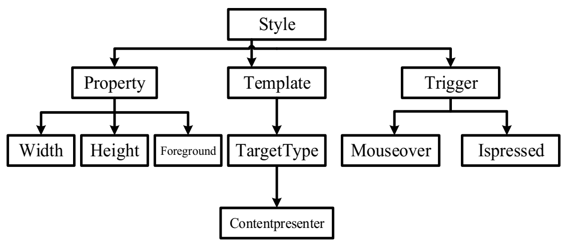

Figure 2. XAML tree view of a button control

DllImportAttribute class indicates that the attributed method is exposed by an unmanaged DLL as a static entry point. To use the class, a namespace (System.Runtime.InteropServices) must be imported into the C# solution. Then, using the constructor to initialize a new instance of the DllImportAttribute class with the name of the DLL containing the method to import. The compiler will search for the DLL in the order of executable file directory, System32 directory, environment variable directory. EntryPoint field indicates the name or ordinal of the DLL entry point to be called. That means, a function in the DLL can be exported by setting the EntryPoint as the function's exact name. After the construction of DllImportAttribute class, the method imported must be declared with the C# modifier of 'extern'.

Another main point of calling a DLL successfully is making sure that the data types between managed codes and unmanaged codes match each to each. Because the drivers of HITPE101 data acquisition module is based on VISA, data types in header file 'visatype.h' should be translated to data types in C#. Some of the transformational relations are listed in the Table. 1.

Besides, pointer in C is regard as unsafe codes in C# because of the .NET Framework running mechanism. When designing a WPF soft panel, we should avoid using pointer to programme C# codes. However, most functions of HITPE101 data acquisition drivers define formal parameters using pointers. To deal with this, C# provides the modifier 'out' and 'ref' to describe a data type of the pointer. If a parameter is declared as a pointer in C, after using DllImpointAttribute class, this parameter must be re-declared with one of these two modifiers. The difference between them is whether the variable modified need to be initialized.

When finishing the instantiation of DllImportAttribute classes and the transformation of data types, a driver function can be used freely in managed codes like C# as usual.

V. Soft Panel Functional Verification

Since the WPF soft panel was designed by taking HITPE101 data acquisition module as a hardware device. HITPE101 is a module which uses an high-speed A/D converter to acquire data from analog channel and transmit these data to the computer through PXI Express bus. To

TABLE I. TRANSFORMATIONAL RELATIONS OF DATA TYPES

Data types in VISA	Data types in C#
ViSession	uint
ViStatus	int
ViUInt16/32	ushort/uint
ViInt16/32	short/int
ViBoolean	bool
ViChar	char
ViChar[]	StringBuilder(or string)
ViReal64	double
ViUInt8	byte

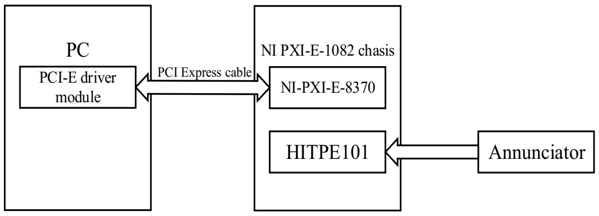

Figure 3. PXI Express auto-test system

complete the functional verification of the soft panel, a PXI Express system should be built according to the diagram shown in Fig. 3.

After finishing the PXI Express system, users can change the parameter of the data acquisition, such as input impedance and coupling mode of the analog channel, or sampling rate and sampling points of the hardware circuit. Done with all these settings, users can click the 'start' button to make the sampling start. The following Fig. 4 shows the display of WPF soft panel when the annunciator generates sinusoidal signals with frequency 50kHz and Vpp 1.00V.

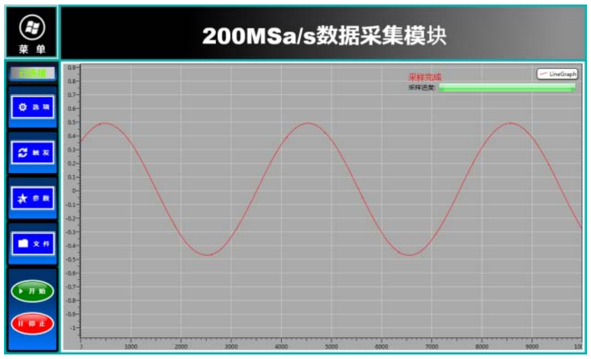

Figure 4. Soft panel functional verification

VI. CONCLUSION

New requirements contribute to innovations in technology, the paper carries out a WPF soft panel designing method to realize the ideals of tablet devices in the field of testing instruments. Not only giving the brief structure of a WPF soft panel, the XAML tree view of designing appearance-customized controls and the way of calling unmanaged functions in managed codes are both explained as well. The verified WPF soft panel may be a new orientation in the development of instrument modules.

REFERENCES

[1] Erik Sorensen. Model-View-ViewModel(MVVM) design pattern using windows presentation foundation(WPF) Technology[J/OL].MegaByte Journal, 2010 : 2-4.

[2] Charles Petzold. Applications=Code + Markup: A Guide to the Microsoft Windows Presentation Foundation[M]. Microsoft Press, 2006: 1-5.

[3] Daniel M. Solis. Illustrated WPF[M]. New Edition. New York : Apress, 2010: 28-29.

[4] Shirish Patil SS. Application Development Using WPF[J].International journal of advanced research in computer engineering and technology. 2012, 1 : 480-483.

[5] Pan, Haihong, Jiang, Jingjie, Chen, Lin, Sun, Hongtao, Tan, Huaqin. A scalable graphics user interface architecture for CNC application based on WPF and MVVM[J]. Advanced Materials Research, 2011 : 317-319.