DESIGN AND RESEARCH OF REAL-TIME MONITORING PROGRAM DEVELOPMENT SYSTEM FOR CIVIL AIRCRAFT FLIGHT TEST DATA BASED ON WPF

Wei Mao, Can Feng, Tao Liu, Feng Wang, Jiayi Liang

Instrumentatuion Department
Flight Test Center of the COMAC
No.108, Jinwen Road, Pudong, Shanghai, 201323
maowei@comac.cc

ABSTRACT

The real-time monitoring program of flight test data is an indispensable and important supporting tool in the flight test of large civil aircraft. With the continuous deepening of the networked test system, a large number of complex flight test parameters pose a huge challenge for the development of monitoring programs. Based on the WPF platform, this paper uses XAML files, reflection and DataBinding to design a system for developing real-time monitoring programs for flight test data. The system realizes the rapid integration and management of the monitoring program by dragging and displaying the control, shortening the preparation cycle in the past few weeks to several hours; when the system is running, the three-layer architecture can drive hundreds of monitoring terminals in real time, which is the domestic large-scale passenger aircraft C919 aircraft. The first flight and forensic flight test provided technical support.
Key words: WPF; XAML; reflection; DataBinding

INTRODUCTION

The real-time monitoring image of flight test data is an important monitoring method. Its main function is to display flight test parameters and airborne audio and video information in real-time with visual controls, reflect the status of various systems and test results of the aircraft, and provide decision-making basis for commanders and security personnel [1].Therefore, it is of great significance for enhancing flight test safety, improving flight test methods and

improving flight test efficiency to construct a professional system that can rapidly develop, integrate and manage the monitoring image.

In the traditional flight test mode, the real-time monitoring screen of domestic mainstream flight test institutions is generally developed and maintained by professionals in Labview, VC++ and other environments, requiring a large amount of code and lengthy preparation period. At the same time, due to the relatively independent procedure, it is not conducive to maintenance and migration, so it is difficult to meet the current flight test with many parameters and complex data type monitoring requirements [2].

In view of the above shortcomings, this paper proposes and designs a flight test data real-time monitoring program development system. This system is able to realize the fast integration and management of the monitoring screen by dragging and dropping the display control and completes the real-time monitoring of the driving and flight test data of the display control together with the DataBinding model, using reflection, XAML language and other technologies based on WPF (Windows Presentation Foundation) platform and SQLite database.

THE BODY

A. System framework

The overall framework of the flight test data real-time monitoring program development system is shown in Fig. 1, which is divided into two parts: the design mode and the operation mode, including five components, i.e. DataServer, Config, DataRelay, DataDerivation and Viewer, etc.

DataServer: data service component, using EF6.0 framework, and taking SQLite database as the core manages various configuration information, including monitoring parameters, monitoring program interface, parameter binding relationship, data calculation method, communication rules, etc; Config: monitoring design component, which is mainly used to create the flight test monitoring parameter list, edit the monitor program interface by dragging and dropping the control and bind the corresponding flight test parameters for the control, is the core component of monitoring screen programming stage; DataDerivation: data synthesis component mainly completes various logic and mathematical operations of flight test parameters during real-time monitoring, and realizes real-time playback of flight test data through corresponding user operations; DataRelay: data forwarding component, whose function is to obtain flight test data from the front-end system, reconstruct, multicast and distribute to various monitoring terminals; Viewer: monitor running component, which runs in each monitor terminal, is the running container for the real-time monitor.

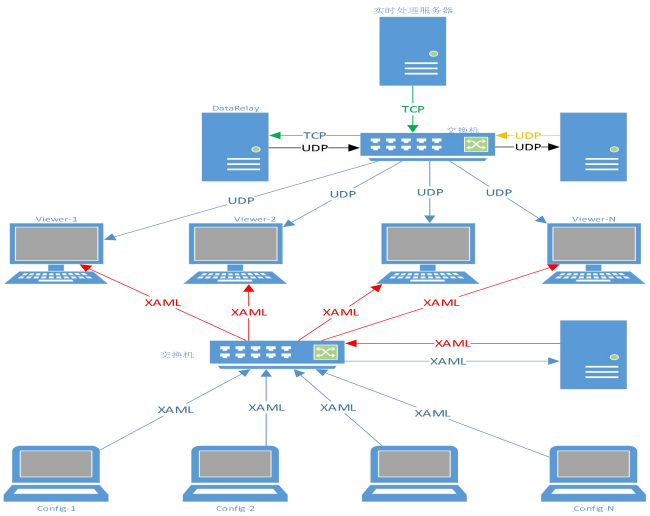

Figure 1: System framework

The whole system only contains one DataServer and one DataRelay, while Config, as the designer of the monitor program, is able to work multiple times at the same time and complete the development of the monitor program in off-line non-real-time state. What't more, any number of viewers are able to receive the UDP/IP multicast data of DataRelay in real time to display flight test parameters. The system is divided into two stages: design and operation. Design stage: complete monitoring monitoring program design in the Config end, and will be the same flight course program editing system or similar for subsystems, such as flight control, avionics, power fuel, at the same time create a monitoring test parameter table, complete the various parameters and corresponding to the binding of display controls formation in XAML format said the monitor screen, finally complete the XAML file database updates of serialization and implement DataServer end; Run stage: start Viewer program, request database, download monitor program information from DataServer, get all information of monitoring program through XAML information deserialization, and finally resolve to monitor interface through WPF platform; DataRelay subscribes to flight test parameters from the real-time processing server, completes data compression and multicast forwarding, calculates parameters of DataDerivation, and displays values of corresponding parameters on the Viewer side according to the established application layer protocol.

B. DataServer

According to the overall framework of the system, DataServer is an essential core component of monitoring program design and runtime. DataServer uses SQLite to maintain the information of all monitoring programs. In the design stage, DataServer receives and stores the design information from each Config, including the type of display control, various properties (such as size, relative position, background color, etc.) and flight test parameters for control-driven. DataServer database structure is shown in Fig. 2.

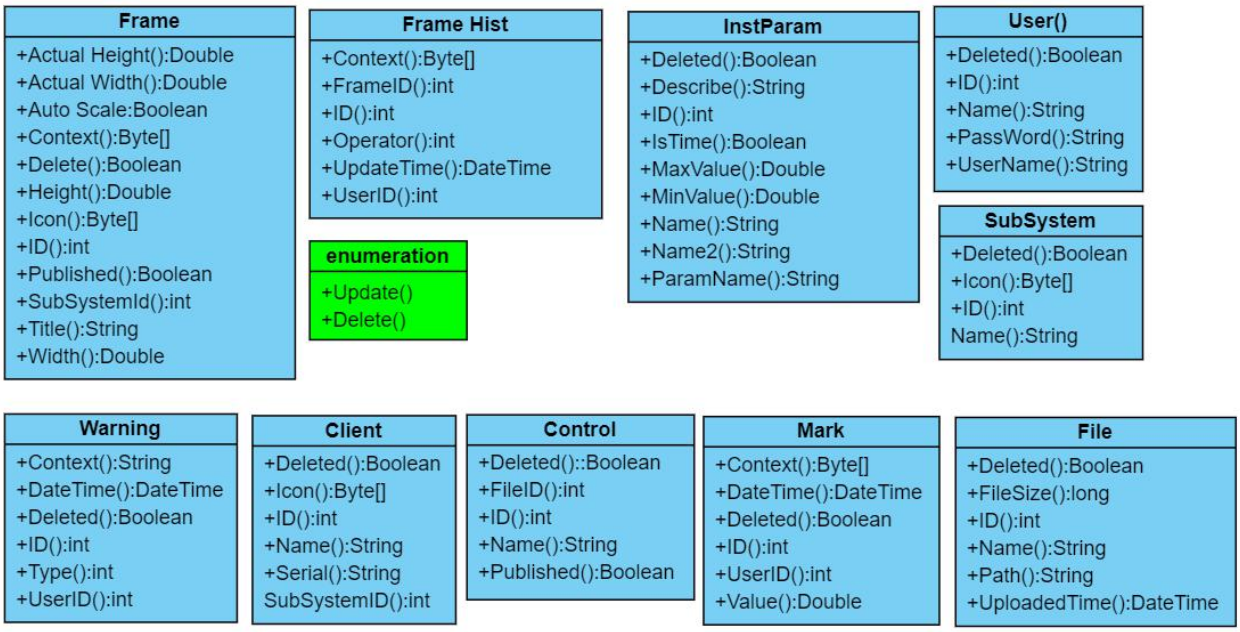

Figure 2: Database structure

The core tables in the database that deal with business content contain Frame, FrameHist, InstParam, and Client and Control. Frame: all real-time monitor information is stored. The interface layout is saved in the Context field in XAML format. Bool volume Published indicates whether the monitor is published or not. Only the Published monitor can be loaded and displayed in the Viewer. FrameHist: screen operation record table, i.e. screen surface update, release, delete, etc. InstParam: all parameters that need to be monitored in real time are maintained in this table, including the name of flight test parameters, upper and lower limit threshold, whether they are time parameters and other attributes; Client: mainly maintains the information of each terminal, that is, the ID, IP address, name and other information of each Viewer that sent requests to DataServer; Control: control library, manages the addition, deletion, update, and so on of display controls.

C. Config

Config is an environment for users to develop and integrate real-time monitoring programs. The user selects the display control in the control library to achieve the overall layout of the monitoring screen by dragging and dropping, and generates the final XAML file after binding flight test parameters for the control.

XAML is a kind of markup language used for instantiating .NET objects. Its files define the layout of the panels, buttons and various controls that make up the program window in the WPF application program[4]. Truly separating the graphic display part from the underlying code is the key to the rapid integration of the monitor program in this system.

1) Screen layout

Control libraries of Config integrate two types of controls: LayoutControl and UIControl. In order to simplify the operation, the layout controls are all Canvas type in this system, that is, the precise absolute coordinates are assigned to Canvas.Left and Canvas.Top to locate the control elements. Meanwhile, mouse events are added to each element of the layout control to realize translation, rotation and size modification of the display control.

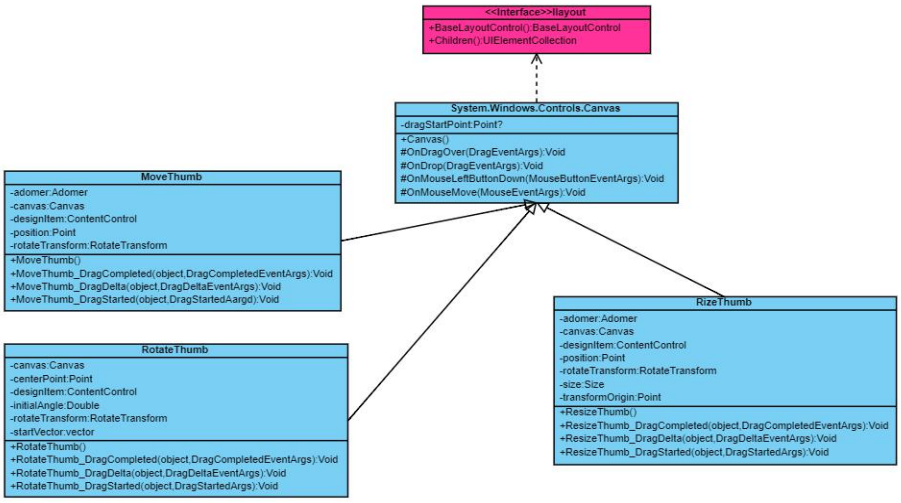

Figure 3: LayoutControl

The visual control of the system includes Graph, Slider, Angular Gauge and Image. Fig. 4 shows the monitoring program developed using the system, and table 1 shows the XAML text information for the corresponding Angular Gauge, Irig Display, static text and other control layouts.

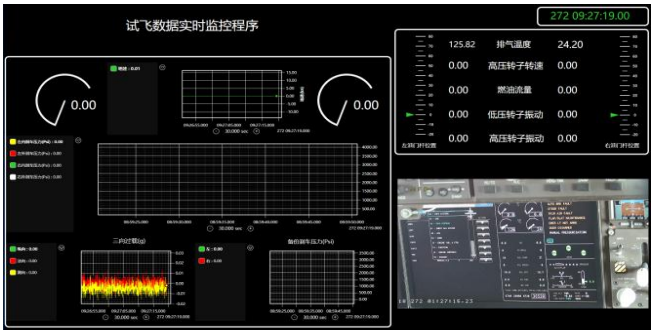

Figure 4: Real-time monitoring program

Table 1 XAML information of controls

<?xml version="1.0" encoding="utf-8"?>
<Canvas

```
    <tadsdc:IrigDisplay Canvas.Left="1589.1198630137"Canvas.Top="18.7808219178083"/>
    <tadsdc:TextDisplay Text="试飞数据实时监控程序" Canvas.Left="363.483888139684"
      Canvas.Top="14.1984526935908"/>
    <Border>
      <cgl:AngularGauge Canvas.Left="46.5102739726027" Canvas.Top="192.321917808219" x:Name=”Gauge1”\>
      <cgl:AngularGauge Canvas.Left="881.647260273973"Canvas.Top="192.321917808219" x:Name = “Gauge2”\>
    </Border>
  </Canvas>
```

2) Flight test parameter binding

In addition to completing the interface layout design of the real-time monitor program, another crucial function of Config is to realize the one-to-one mapping between flight test parameters and display controls. The mapping relationship is established through the DataBinding of the WPF platform. The structure of the DataBinding is shown in Fig. 5 [3].

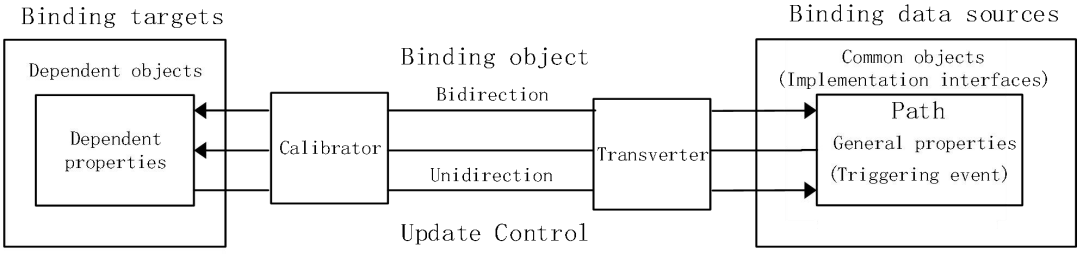

Figure 5: DataBinding data model

The data source of Binding is the flight test parameter, and the target of Binding is each display control. The system adds a "wrapper" to each flight test parameter, wraps it as a property, and stimulate the PropertyChanged event in the attribute set methods, so as to realize an INotifyPropertyChanged interface (The wrapper process for the parameter NX is shown in Table 2). Binding automatically listening for events from this interface, and when the Binding target (display control) subscribes to this event, the flight test parameter values update the display on the control. Display the PropertyChanged event that the control can subscribe to multiple parameters at the same time (The XAML information of the graph "binding" three-way overload parameters NX, NY and NZ of the waveform chart control is shown in Table 3).

Table 2 Packaging Information of NX Parameters

```
public class FlightTestParameters:INotifyPropertyChanged
{
    public event PropertyChangedEventHandler    PropertyChanged;
    private string nX;
    public string NX{get => nX;
```

```
                    set{nX= value;
                        if (this.PropertyChanged != null){
this.PropertyChanged.Invoke(this,new PropertyChangedEventArgs("NX"));}
                    }}
}
```

Table 3 Parameter Bindings

```
this.graph.SetBinding(Graph.DataSourceProperty, new Binding() { Source =
flightTestParameters, Path = new PropertyPath("NX") });
this.graph.SetBinding(Graph.DataSourceProperty, new Binding() { Source =
flightTestParameters, Path = new PropertyPath("NY") });
this.graph.SetBinding(Graph.DataSourceProperty, new Binding() { Source =
flightTestParameters, Path = new PropertyPath("NZ") });
```

The WPF platform provides two classes, which are XAMLWriter and XAMLReader, to realize the serialization and deserialization of XAML. The serialized information is stored in the Context field of the Frame table in the database and Shared to the Viewer on each monitoring terminal.

D. DataRelay

The system runs in a typical three-layer framework, which is divided into data access layer, business logic layer and display layer. DataRelay is the core module of the data access layer, whose function is mainly to encapsulate the test parameters subscribed into broadcast packets, which are compressed by the data stream and forwarded to the business logic layer and the display layer. Meanwhile, the mapping relationship between flight test parameter names and data packet index is sent to each layer periodically using HTTP, so as to meet the dynamic change of the number of monitor programs in the display layer.

DataRelay converts all subscribed flight test parameter values to an 8-byte double-precision floating-point number, even if the parameter is a 1-bit Boolean, as shown in figure 6. The data packet encapsulated by N flight test parameters is composed of 32bytes packet head, N*8bytes valid data and 4bytes packet tail.

When the flight test parameters are too many, the larger length of the packet is not conducive to processing, so the encapsulated packet will be compressed. DataRelay can compress data streams to one-tenth of their original length via Delate compression algorithm integrated by .NET.

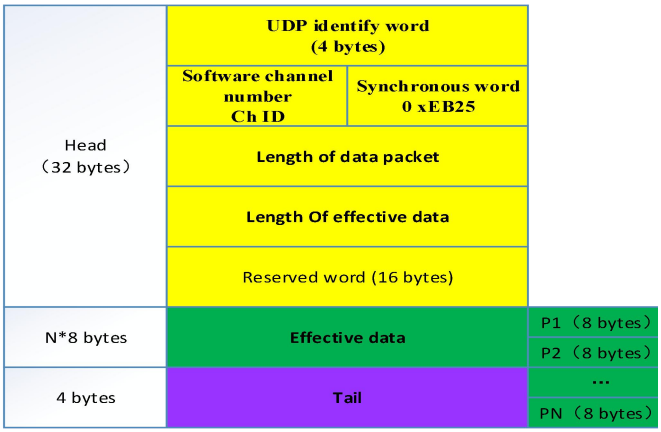

Figure 6: The data format of flight test parameter encapsulated

Table 4 Packet compression code snippet

```
byte[] flightTestParams = GetDataPacket(double[]params);
System.IO.MemoryStream memoryStream = new System.IO.MemoryStream();
System.IO.Compression.DeflateStream deflateStream = new
                                    System.IO.Compression.DeflateStream( memoryStream,
                                    System.IO.Compression.CompressionMode.Compress,true);
deflateStream.Write(flightTestParams,0,flightTestParams.Length);
deflateStream.Close();
```

In addition to encapsulating flight test parameters, DataRelay establishes an HTTP service in a separate thread, allowing each monitoring terminal to GET the JSON information of parameter name and real-time physical quantity's position index in UDP broadcast packet in a GET manner.

The Viewer creates the corresponding key-value pair based on the JSON information and parses the corresponding flight test parameter by offset it in 8-byte units according to the parameter index. The parsing procedure can be described as follows: first declare the parameter index variable as index, and create values, an intermediate array storing 8 bytes; and then Array.Copy (flightTestParams, index * 8, values, 0, 8) is used to extract the valid bytes of corresponding parameters from flightTestParams; Finally, BitConverter.ToDouble(values,0) is used to parse the physical quantity with double precision.

E. DataDerivation and Viewer

According to the system structure in figure 1, the flight test data for real-time monitoring have plenty of parameters that need to be calculated twice besides the direct parameters sent by the DataRelay broadcast, which are called derived parameters. The value of a derived parameter is the return value after logical or mathematical operation of one or multiple direct parameters as formal parameters. DataDerivation uses reflection technology to dynamically realize real-time

monitoring of derived parameters. Reflection is the ability to access and modify a program while it is running[5]. WPF integrates reflection information of variables, such as class name, method name, method parameter name, method return value, etc. into the executable file during program compilation, and provides interface access reflection information for the program. DataDerivation serves as the business logic layer in the three-layer structure of the runtime. The processing process is shown in figure 7.

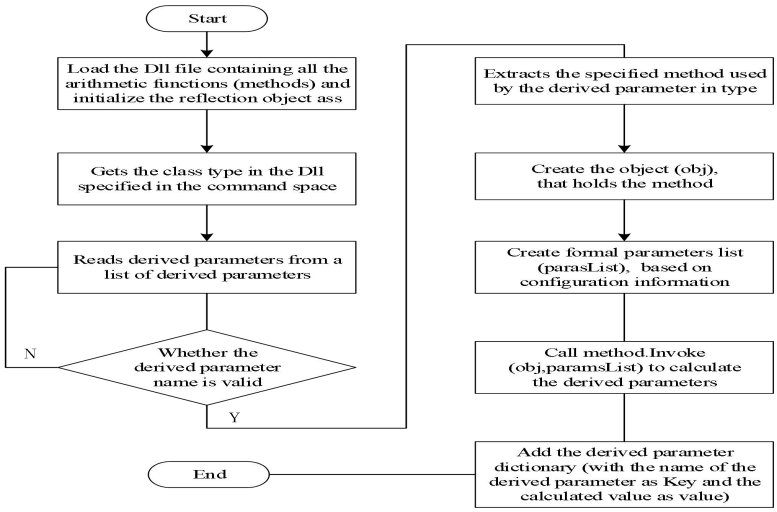

Figure 7: Reflection process

Viewer is the container of each monitoring screen, directly facing flight test engineers and aircraft designers.The operating mechanism of Viewer is in a "passive" way, relying entirely on DataServer and DataRelay, as shown in figure 8.

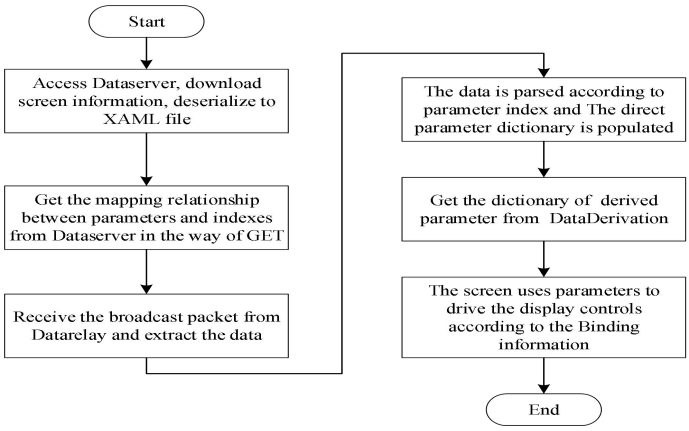

Figure 8: Viewer workflow

EXPERIMENTAL RESULTS

The flight test data real-time monitoring program development system was used to develop 40 monitoring programs, and the test results are shown in table 5.

Table 5 test results

Project	System proposed	Traditional mode
Development cycle	3 days	One month
Number of flight test parameters	8000	8000
Whether codes are needed	No	Yes
Whether professionals are needed	No	Yes
Number of driving terminals	Expansion unlimited	Expansion limited
Maximum number(single screen)	3000	200
Monitor delay	≤50ms	≥100ms

CONCLUSIONS

The flight test data real-time monitoring program development system designed in this paper has successfully guaranteed the first flight of domestic large passenger aircraft and the flight test of various subjects: (1) the rapid integration of the monitoring screen is completed by dragging and dropping the display control, greatly shortening the preparation cycle; (2) XAML file is used to realize the design, management and maintenance of monitoring screen program; (3) DataBinding model is used to realize the binding and real-time driving of flight test parameters.

REFERENCES

[1] Wang Peng, Guo Shiwei, Huo Zhaohui, etc. Conception and Practice of Air-Sea Integrated Test Flight Test Network[J].China Measurement ,2017,43(1): 1-5.
[2] Qi Chun. Telemetry real-time processing software system and its application[J].Science Technology and Engineering,2010(10): 7047-7050.
[3] Liu Tiemeng. Exploring WPF [M]. Beijing: China Water Resources and Hydropower Press, 2010: 35-50.
[4] Mathew MacDonald. WPF Programming Collection [M]. Wang Decai, translated. Beijing, Tsinghua University Press, 2013: 17-25.
[5] Xu Bo. Go language from entry to advanced practice [M]. Beijing: Mechanical Industry Press, 2018: 280-281.