2019 年第 2 期　总 133 期
ISSN 1674-9804　CN 31-1614/V

DOI：10.19416/j.cnki.1674－9804.2019.02.004

基于 WPF 的民机试飞数据实时监控画面开发系统设计与研究

Real-time Monitoring Program Development System for Civil Aircraft Flight Test Data Based on WPF

冯　灿　刘　涛　毛　为　汪　峰　梁嘉羿/FENG Can　LIU Tao　MAO Wei　WANG Feng　LIANG Jiayi
（中国商飞民用飞机试飞中心，上海 201323）
（COMAC Flight Test Center，Shanghai 201323，China）

摘　要：
试飞数据实时监控程序是大型民机试飞过程中不可或缺的重要支持性工具。随着机载测试系统网络化进程的不断深入，数量庞大、类型复杂的试飞参数给监控程序的开发带来了巨大的挑战。基于 WPF（Windows Presentation Foundation）平台，利用 XAML 文件的序列化和反序列化、反射、DataBinding 等技术设计了一套用于开发试飞数据实时监控程序的系统。在设计端，系统以拖拽显示控件的方式实现监控程序的快速集成和管理，将以往数周的准备周期缩短为数小时；在运行端，系统利用三层架构可实时驱动数百台监控终端，为某型客机的首飞及取证试飞提供了技术支持。

关键词：WPF；XAML；反射；DataBinding
中图分类号：V217⁺.2　　**文献标识码：**A　　**OSID：**

[**Abstract**] The real-time monitoring program of flight test data is an indispensable and important supporting tool in the test flight of large civil aircraft. With the continuous deepening of the networked test system, a large number of complex flight test parameters poses a huge challenge for the development of monitoring programs. Based on the WPF(Windows Presentation Foundation) platform, this paper designs a system for developing real-time monitoring programs for flight test data by making use of XAML files, reflection and DataBinding. The system realizes the rapid integration and management of the monitoring program by dragging and displaying the control, shortening the preparation cycle from several weeks in the past to several hours; when the system is running, the three-layer architecture can drive hundreds of monitoring terminals in real time. The system has provided technical support for the the domestic large-scale passenger aircraft in the maiden flight and forensic flight test.

[**Keywords**] WPF; XAML; reflection; DataBinding

0　引言

试飞数据实时监控的主要功能是利用可视化控件实时显示试飞参数和机载音、视频等信息，反映飞机的各系统状态和试验效果，为指挥人员、保障人员等提供决策性依据[1]。因此，构建能够快速开发、集成和管理监控画面的专业系统对于增强试飞安全、改进试飞方法、提高试飞效率等具有重要意义。

在传统的试飞模式下，国内主流试飞机构的实时监控画面一般由专业人员在 Labview、VC ++ 等环境下开发和维护，需要编写大量代码，准备周期十分冗长，同时由于程序相对独立，也不利用维护及移植，难以满

[基金项目]大型民机试飞数据高速实时处理及异常状态检测分析平台(17511105000)。

足当前试飞参数众多、数据类型复杂的监控需求[2]。

本文针对以上不足,提出并设计了一套试飞数据实时监控画面开发系统。该系统以 WPF(Windows Presentation Foundation,以下简称 WPF)为平台,立足于 SQLite 数据库,利用反射、XAML 语言等技术,以拖拽显示控件的方式实现监控画面的快速集成和管理,同时配合 DataBinding 模型完成显示控件的驱动和试飞数据的实时监控。

1　系统架构

试飞数据实时监控画面开发系统的整体架构如图 1 所示,分为设计端和运行端两部分,包含 DataServer、Config、DataRelay、DataDerivation、Viewer 等五大组件。

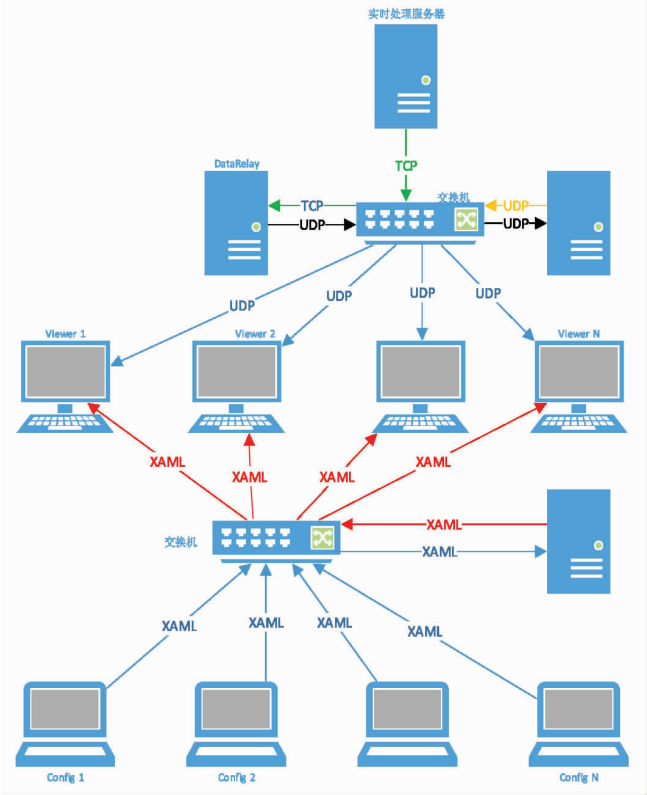

图 1　系统架构

DataServer:数据服务组件,采用 EF6.0 架构,以 SQLite 数据库为核心管理各种配置信息,包括监控参数、监控程序界面、参数绑定关系、数据计算方法、通信规则等。

Config:监控设计组件,主要用于创建试飞监控参数列表、以拖拽控件的方式编辑监控程序界面以及为控件绑定对应的试飞参数,是监控画面程序设计阶段的核心组件。

DataDerivation:数据合成组件,主要在实时监控时完成试飞参数的各种逻辑及数学运算,同时相应用户操作实现试飞数据的实时回放。

DataRelay:数据转发组件,其功能主要是从前端系统获取试飞数据,并重构、组播分发至各个监控终端。

Viewer:监控程序运行组件,该组件运行于各个监控终端,是实时监控程序的运行容器。

整个系统只包含一个 DataServer 和一个 DataRelay,而 Config 作为监控程序的设计器,可多个同时工作,在离线非实时状态下完成监控程序的开发;同时,任意数量的 Viewer 可实时接收 DataRelay 的 UDP/IP 组播数据实现试飞参数的显示。系统分为设计和运行两个阶段,其步骤如下:

设计阶段,在 Config 端完成监控程序的设计,并将相似系统或相同试飞科目的程序编辑为子系统,譬如飞控、航电、动力燃油等,同时创建监控试飞参数表,完成各个参数与对应显示控件的绑定,形成以 XAML 格式表示的监控画面,最后完成 XAML 文件的序列化并实现 DataServer 端的数据库更新;

运行阶段,启动 Viewer 程序,请求数据库,从 DataServer 下载监控程序信息,通过 XAML 信息的反序列化得到监控画面程序的所有信息,并最终通过 WPF 平台解析为监控界面;DataRelay 从实时处理服务器订阅试飞参数,完成数据的压缩和组播转发,DataDerivation 实现参数的计算,Viewer 端则按照既定的应用层协议显示相应参数的值。

2　DataServer

根据系统的整体架构,DataServer 是监控程序设计以及运行阶段必不可少的核心组件。DataServer 使用 SQLite 维护所有画面程序的信息,在设计阶段,DataServer 接收并存储来自各个 Config 端的设计信息,包括显示控件的类型、各种属性(如尺寸、相对位置、背景颜色等)以及用于控件驱动的试飞参数。DataServer 数据库结构如图 2 所示。

数据库中涉及业务内容的核心表包括 Frame、FrameHist、InstParam 以及 Client 和 Control。

Frame:存储了所有实时监控程序信息,其中界面布局以 XAML 格式保存在 Context 字段中,bool 量 Published 表示监控程序是否发布,只有发布的监控程序才能在 Viewer 端加载显示。

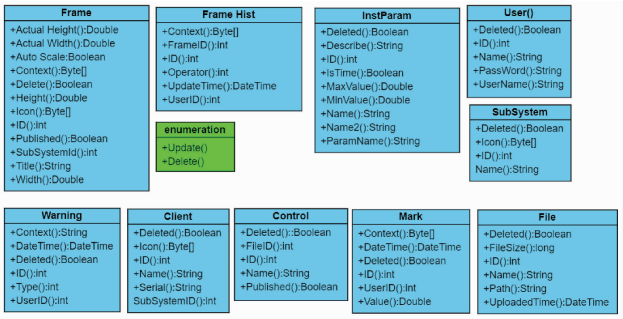

图 2　数据库结构

FrameHist：画面操作记录表，如画面的更新、发布、删除等。

InstParam：该表中维护了所有需要实时监控的参数，包括试飞参数的名称、上下限阈值、是否为时间参数等属性。

Client：主要维护各个终端的信息，即向 DataServer 发送请求的各个 Viewer 的 ID、IP 地址、名称等信息。

Control：控件库，用于管理显示控件的添加、删除、更新等。

3　Config

Config 是用户进行实时监控程序开发和集成的环境。用户通过选择控件库中的显示控件，以拖拽的方式实现监控画面的整体布局，为控件绑定试飞参数后生成最终的 XAML 文件。

XAML 是一种用于实例化.NET 对象的标记语言，其文件定义了在 WPF 应用程序中组成程序窗口的面板、按钮以及各种控件的布局[4]，真正意义上将图形显示部分从底层代码中分离出来，是本系统实现监控程序快速集成的关键。

3.1 画面布局

Config 中的控件库中集成了两类控件：布局控件（LayoutControl）和可视控件（UIControl）。为了简化操作，本系统的布局控件均为 Canvas 类型如图 3 所示，即使用精确的绝对坐标为 Canvas. Left 和 Canvas. Top 这两个属性赋值来定位控件元素，同时为布局控件中的各个元素添加鼠标事件，实现显示控件的平移、旋转及尺寸修改。

本系统的可视控包括波形图表（Graph）、滑块（Slider）、仪表（Angular Gauge）、图片（Image）等。图 4 是利用该系统开发的监控程序，图 5 是相应的 Angular Gauge、Irig Display 以及静态文本等控件布局的 XAML 文本信息。

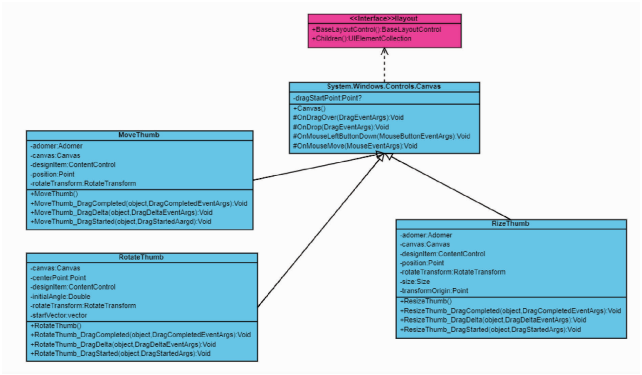

图 3　布局控件

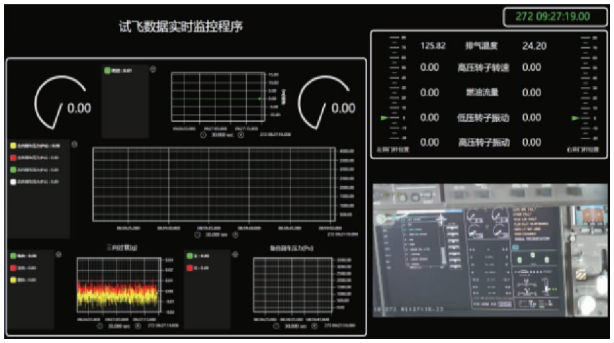

图 4　实时监控程序

```
<?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="192.3219178082l9" x:Name="Gauge1"\>
    <cgl:AngularGauge Canvas.Left="881.647260273973"Canvas.Top="192.3219178082l9" x:Name = "Gauge2"\>
  </Border>
</Canvas>
```

图 5　画面的 XAML 信息

3.2 试飞参数绑定

Config 除了完成实时监控程序的界面布局设计，另一个重要的功能是实现试飞参数与显示控件的一一映射。这种映射关系是通过 WPF 平台的 DataBinding 来建立的，DataBinding 的结构如图 6 所示[3]。

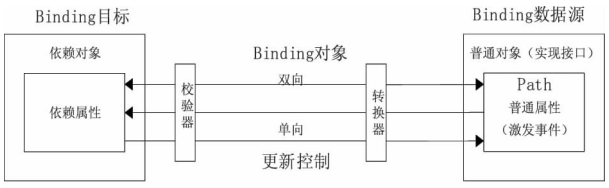

图 6　DataBinding 数据模型

Binding 的数据源就是试飞参数，Binding 的目标则是各个显示控件。系统为每一个试飞参数添加

“包装器”,将其包装为一个属性,并在属性的 set 方法中激发 PropertyChanged 事件,从而实现 INotifyPropertyChanged 接口(图 7 是参数 NX 的包装过程),Binding 自动侦听来自这个接口的事件,当 Binding 的目标(显示控件)订阅该事件时,试飞参数值则会在控件上更新显示。显示控件可同时订阅多个参数的 PropertyChanged 事件(图 8 为波形图表控件 graph “绑定”三向过载参数 NX、NY、NZ 的 XAML 信息)。

```
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"));}
                     }}
}
```

图 7　NX 参数包装过程

```
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") });
```

图 8　参数绑定

WPF 平台提供了 XAMLWriter 和 XAMLReader 两个类实现 XAML 的序列化和反序列化,序列化后的信息存储在数据库中 Frame 表的 Context 字段中,以此来共享给各个监控终端上的 Viewer。

4　DataRelay

系统以典型的三层架构来运行,分为数据访问层、业务逻辑层和显示层。其中 DataRelay 是数据访问层的核心模块,主要功能是将订阅的试飞参数封装成广播数据包,经数据流压缩后转发至业务逻辑层和显示层,同时利用 HTTP 周期性地向各层发送试飞参数名称与数据包索引的映射关系,以满足显示层监控程序数量的动态变化。

DataRelay 将所有订阅的试飞参数数值全部转换为 8 个字节的双精度浮点数,即使该参数为 1 个 bit 的布尔量,封装格式如图 9 所示。N 个试飞参数封装后的数据包由 32 字节包头、N×8 字节有效数据和 4 字节的包尾组成。

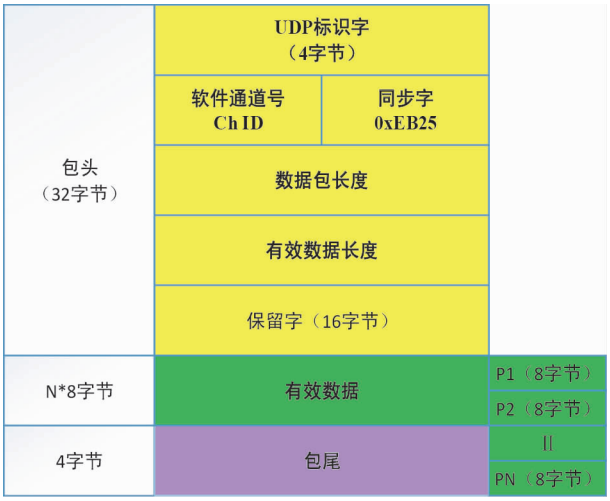

图 9　试飞参数封装数据格式

当试飞参数过多时,数据包的长度较大不利于处理,因此封装后的数据包会进行压缩。DataRelay 使用.NET 已经集成的 Delete 压缩算法,可将数据流压缩至原始长度的十分之一。

```
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();
```

图 10　数据包压缩过程

DataRelay 除了封装试飞参数外,会独立一个线程建立 HTTP 服务,允许各个监控终端以 GET 方式获取参数名称与实时物理量在 UDP 广播数据包中位置索引的 JSON 信息。

Viewer 根据 JSON 信息创建对应的键值对,并根据参数索引以 8 字节为单位进行相应的偏移从而解析对应的试飞参数。解析过程可描述为:首先声明参数索引变量为 index,同时创建存储 8 字节的中间数组 values;其次通过 Array. Copy (flightTestParams, index * 8, values, 0, 8)从 flightTestParams 中提取对应参数的有效字节;最后使用 BitConverter. ToDouble(values,0)解析为双精度物理量。

5　DataDerivation 和 Viewer

按照图 1 的系统结构,用于实时监控的试飞数据,除了由 DataRelay 广播发出的直接参数外,还有很多需要二次计算的参数,这类参数称之为派生参数。派生参数的值是以一个或多个直接参数作为形

参、经过逻辑或数学运算后的返回值。DataDerivation 利用反射技术动态的实现派生参数实时监控。所谓反射是指在程序运行期间对程序本身进行访问和修改的能力[5]。WPF 在程序编译期间，将变量的反射信息，譬如类名、方法名、方法的形参名、方法的返回值等整合到可执行文件中，并为程序提供接口访问反射信息。DataDerivation 作为运行端三层结构中的业务逻辑层，处理流程如图 11 所示。

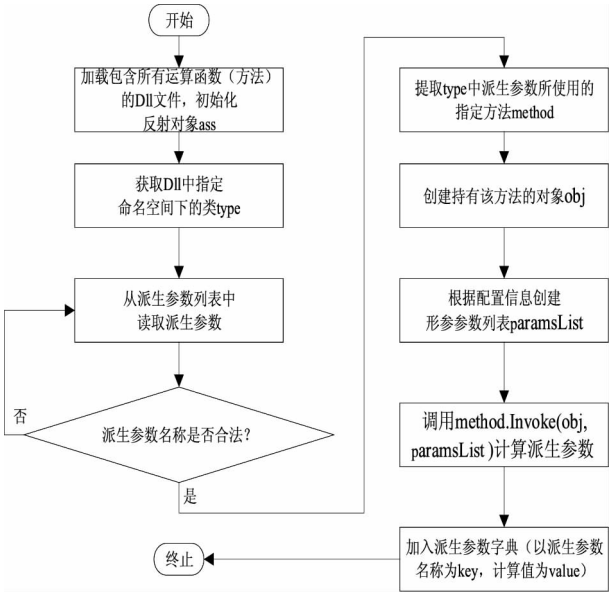

图 11　反射处理过程

Viewer 是各个监控画面的容器，直接面向试飞工程师、飞机设计人员。Viewer 的运行机制是“被动”的，完全依赖于 DataServer 和 DataRelay，处理流程如图 12 所示。

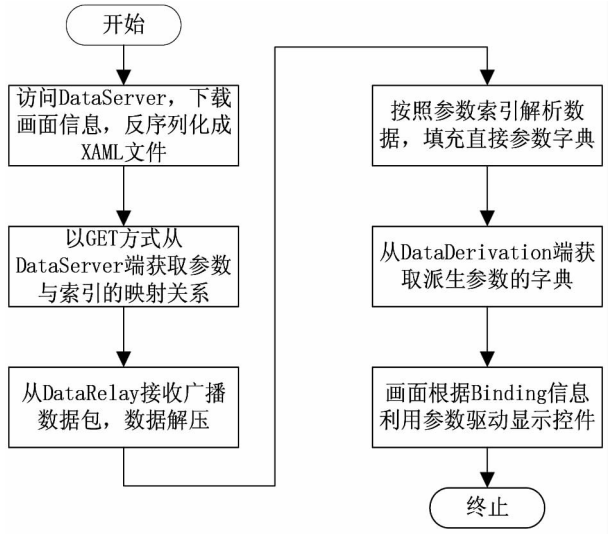

图 12　Viewer 工作流程

6　试验与结论

利用试飞数据实时监控画面开发系统开发了 40 个画面程序，试验结果如表 1 所示。

表 1　试验结果

项目	本文系统	传统模式
开发周期	3 天	一个月
试飞参数数量	8 000	8 000
是否需要表写代码	否	是
是否需要专业人员	否	是
驱动终端数量	随网络可无限扩展	扩展有限制
单个画面最大参数量	3 000	200
监控延时	≤50ms	≥100ms
重构和移植	更新配置文件	需重构代码

本文所设计的试飞数据实时监控画面开发系统，成功保障了某型客机首飞及各项科目的试飞：

（1）以拖拽显示控件的方式完成对监控画面的快速集成，大幅缩短了准备周期；

（2）利用 XAML 文件实现了对监控画面程序的设计、管理和维护；

（3）基于 DataBinding 模型实现了试飞参数的绑定和实时驱动。

参考文献：

[1] 王鹏，郭世伟，霍朝晖，等. 空天地一体化试飞测试网络构想与实践[J]. 中国飞行试验研究院，中国测试，2017，43(1)：1-5.

[2] 祁春，段宝元. 遥测数据实时处理软件系统及其应用[J]. 科学技术与工程，2010，10(28)：7047-7050.

[3] 刘铁猛. 深入浅出 WPF[M]. 北京：中国水利水电出版社，2010：35-50.

[4] Matthew MacDonald. WPF 编程宝典[M]. 王德才，译. 北京：清华大学出版社，2013：17-25.

[5] 徐波. Go 语言从入门到进阶实践[M]. 北京：机械工业出版社，2018：280-281.

作者简介：

冯　灿　男，博士，高级工程师。主要研究方向：试飞与测试。E-mail：fengcan@comac.cc

刘　涛　男，硕士，工程师。主要研究方向：试飞数据实时处理。E-mail：liutao2@comac.cc

毛　为　男，副总工程师。主要研究方向：试飞与测试。E-mail：maowei@comac.cc

汪　峰　男，硕士，助理工程师。主要研究方向：试飞数据实时监控。E-mail：wangfeng4@comac.cc

梁嘉羿　男，硕士，助理工程师。主要研究方向：试飞数据实时处理。E-mail：liangjiayi@comac.cc