DOI 10.5162/etc2014/3.3

Advanced Monitoring Techniques

Gonzalez-Martin, Moises, Rubio-Alvarez, Pedro,
Roses-Sanchez, Diego Lopez-Parra, Rodrigo, Guillermo Reillo-Morales
moises.gonzalez@military.airbus.com, pedro.rubio@military.airbus.com,
diego.roses@military.airbus.com, rodrigo.lopez@military.airbus.com,guillermo.reillo@military.airbus.com

Airbus Defence and Space - Flight Test - Spain

1. Abstract

The State of The Art in Operating Systems and new human machine interfaces are moving forward quickly. Flight Test Data Processing Department has developed new tools for monitoring Flight Tests using new computer technologies like .NET virtual machines, “on-the-fly” compilation, intelligent behavior, multi-touch capabilities and high performance vector graphics libraries.

All these new techniques allow to the user to optimize the Flight Tests reducing the time for take decisions, helping to make complex calculations in real time and adapting the visualization displays to FTE requirements in real time.

Key words: Flight Test, Data Processing, Monitoring, Expert Systems, Automatic Maneuver Detection.

2. Introduction

25 years ago only a few hundred parameters were monitored during the Flight Test Campaign. Today, the A400M Flight Test Program monitors over 200,000 parameters in real time. With the evolution of computers, flight tests should improve productivity per flight hour. With the improvement of processors performances and the parallel computing and the new human-machine interfaces it’s possible to regenerate and develop graphics representations previously unthinkable.

Flight Test Data Processing Department in Airbus Defence and Space establish to use the top of the technologies for the test optimizations. For that reason, software developers inside Flight test have attempted to establish alternative forms of visualization and interaction with the information in real time.

Through this document new ways of monitoring and the basis of the development will be explained.

2.1. New ways of Monitoring

Since last years it is possible to manage tactile portable devices like mobile phones, tablets and laptops with multi-touch capabilities. Besides, operating systems support these new interfaces and ways of interaction. The best known examples are Android, Microsoft Windows 8 and iOS platforms.

Flight Test Spain in Airbus Defence and Space uses Microsoft Windows technologies based in the .NET Framework and High Performance graphics libraries like vgdotnet[1] and Lightning charts[2].

For that platform, following new techniques has been developed:

- Behaviour Code. Designed to give the user the capability of changing the behavior of a display in real time.
- Smart Objects. New kind of monitoring tools that give the user the capability of make complex calculations and aid to take decisions.
- Document Oriented Monitoring and Analysis Tool (DOMA) have been designed to give capability to fulfill digitalized documents in real time. Acceptance Manuals, check-list and

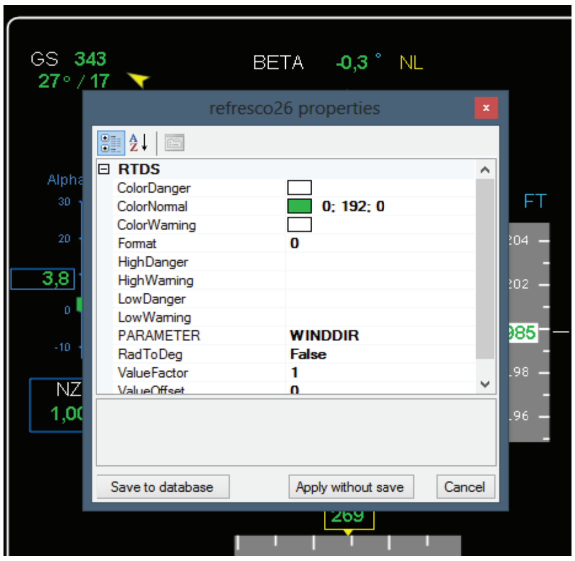
DOI 10.5162/etc2014/3.3

other crosscheck procedures are examples of documents digitalized.

2.2. Behaviour Code

In many cases Flight Test Engineer (FTE) requires little changes in the bevaiour of a monitoring screen because a misunderstand, a last minute modification or because it is more clear for him to have different behaviour. In a standard way of work, Flight Test Enginners must require to Software developer a change in the implementation.

Behaviour code allows to change the behaviour of the display on the execution environment by means of a property of the object. That property can change others properties of the object (color, HighDanger, scales.....) using a compilable C# code.



RTDS	
ColorDanger	⬜
ColorNormal	🟩 0; 192; 0
ColorWarning	⬜
Format	0
HighDanger	
HighWarning	
LowDanger	
LowWarning	
PARAMETER	WINDDIR
RadToDeg	False
ValueFactor	1
ValueOffset	0

Fig. 1. List of properties for a monitoring object

Dynamic code execution is a powerful feature that allows applications to be extended with code that is not compiled into the application. Users can customize applications and developers can dynamically update code easily. The Behaviour Code Property is possible through Dynamic Code Using CodeDOM[3] for .NET framework.

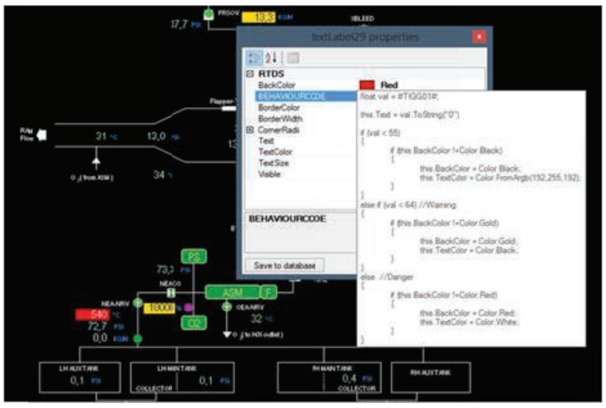
CodeDOM is Code Document Object Model. This provides a graphical (in the sense of graph nodes, not of a picture) representation of code in the form of a tree. This represents the hierarchy of nested code elements: namespaces contain classes, classes contain methods, and methods contain variable declarations.

2.2.1. Compiling code on-the-fly

.NET provides powerful access to the CIL (CIL is a CPU- and platform-independent instruction set that can be executed in any environment supporting the Common Language Infrastructure[4] such as the .NET runtime on Windows, or the cross-platform Mono runtime) code generation process through the System.CodeDom.Compiler and Microsoft.CSharp and Microsoft.VisualBasic namespaces. In these namespaces is possible find the tools that allow to compile an assembly either to disk or into memory. There is also need the Reflection namespace as it contains the tools to invoke an object and its methods once you've compiled the object.

The process to generate and execute code dynamically involves the following steps:

1. Create or read in the code that is to be executed as a string
2. Wrap the code into fully functional assembly source code, which includes namespace references (using commands), a namespace and a class that is to be invoked
3. Compile the source code into an assembly
4. Check for errors on compilation
5. Use the assembly reference to create an instance of the object
6. Call the specified method on the instance reference returned using Reflection
7. Handle any return value from the method call by casting into the proper type



```
float val = #TIGG01#;

this.Text = val.ToString("0");

if (val < 55)
{
        if (this.BackColor !=Color.Black)
        {
                this.BackColor = Color.Black;
                this.TextColor = Color.FromArgb(192,255,192);
        }
}
else if (val < 64) //Warning
{
        if (this.BackColor !=Color.Gold)
        {
                this.BackColor = Color.Gold;
                this.TextColor = Color.Black;
        }
}
else  //Danger
{
        if (this.BackColor !=Color.Red)
        {
                this.BackColor = Color.Red;
                this.TextColor = Color.White;
        }
}
```

DOI 10.5162/etc2014/3.3

Fig. 2. BehaviourCode property allows to generate code dynamically

Besides, if it would be necessary to get even more low level it's possible to use the System.Reflection.Emit namespace to generate CIL level code directly.

2.3. Smart Objects

For the Production Aircraft Test it's very significant the number of flights assigned to verify the proper operation of the systems. For that task the Air Crew must follow a Production Aircraft Test Manual (PATM). This document contains all the test points, operations procedures and maneuvers needed for complete the test.

In a general case, is necessary to make complex calculations in order to identify the flight phase, an aircraft condition or calculate a repetitive computation and generate evidences of the correct aircraft operation.

One example of complex calculations is when the FTE needs maneuvers with previous stabilization. The criteria of validation are base on the vertical difference between the values of the linear regression (least squares) in the first and in the last sample of the time slice for the following parameters:

- MACH (Speed)
- SAT (Ambient Temperature)
- ALT (Altitude)
- EGT (Engine Temperature)
- Others

A Manual detection if stabilization delegates the responsibility of the quality of the stabilization in the intuition and expertise of the FTE. By the aid of a smart object there is an automatic detection based in the rule of calculating the criteria of validation every 10 seconds using the data from the previous 3 minute

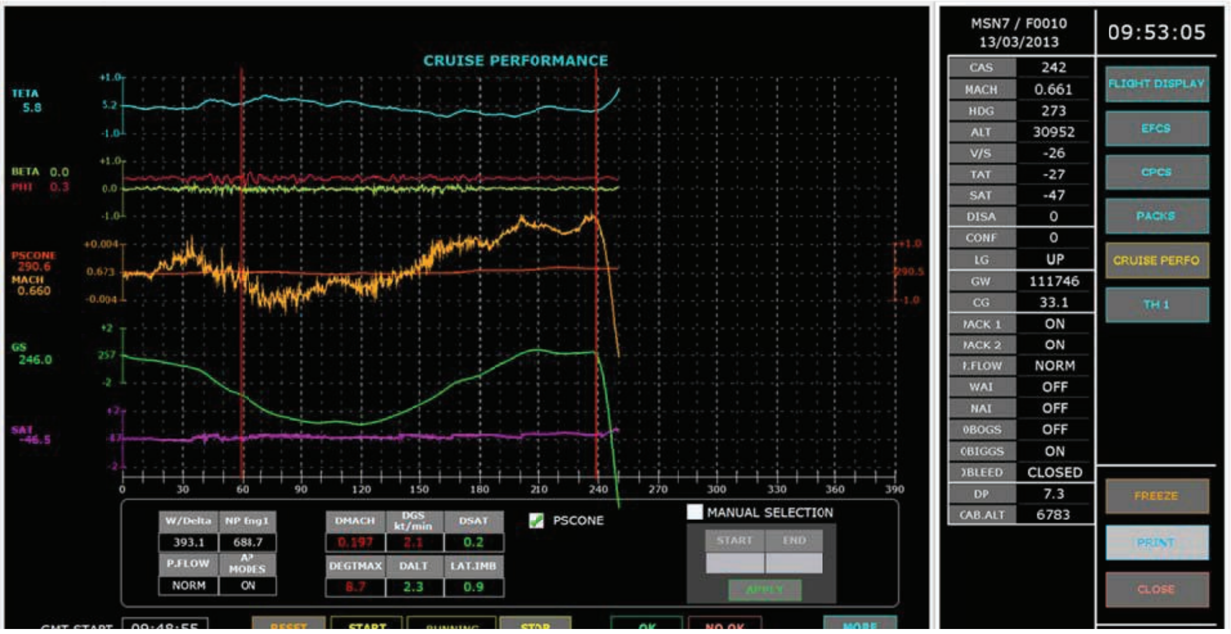
Every single object contains a set of buttons, status and warnings that manage the smart object operations in real time.

2.3.1. Smart Objects Buttons

- START Automatic tasks are started.
- STOP Automatic tasks are stopped and data are not deleted.
- RESET Reset the component to initial status, deleting all data.

2.3.2. Smart Objects Status

- RUNNING Indicates that the component is activated and running automatic tasks.



MSN7 / F0010
13/03/2013
09:53:05

CAS	242
MACH	0.661
HDG	273
ALT	30952
V/S	-26
TAT	-27
SAT	-47
DISA	0
CONF	0
LG	UP
GW	111746
CG	33.1
PACK 1	ON
PACK 2	ON
P.FLOW	NORM
WAI	OFF
NAI	OFF
OBOGS	OFF
OBIGGS	ON
BLEED	CLOSED
DP	7.3
CAB.ALT	6783

W/Delta	NP Eng1
393.1	688.7
P.FLOW	AP MODES
NORM	ON

DMACH	DGS kt/min	DSAT
0.197	2.1	0.2
DEGTMAX	DALT	LAT.IMB
8.7	2.3	0.9

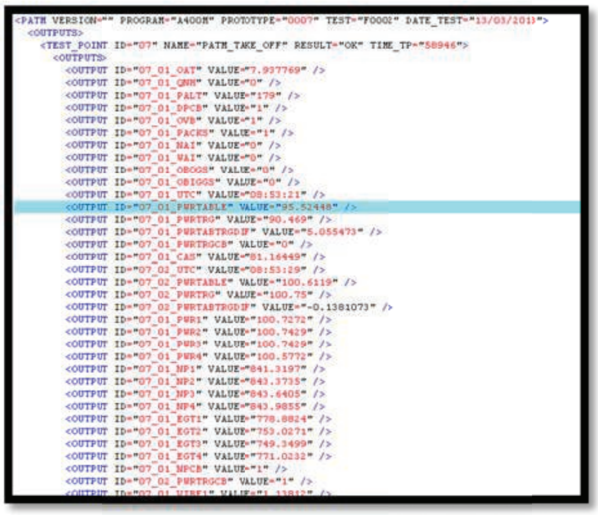
- FINISHED Indicates that the component is finished and suggests a result.
- STOPPED Indicates that the component has been stopped and all tasks are stopped.

2.3.3. Smart Objects Suggestions

- OK indicates that all tasks have been performed and all checks are OK.
- NO OK indicates that all tasks have been performed and some of them have any problem.

When a component is activated (RUNNING) it starts to perform the programmed computation, and when all the computations are finished, it suggests user OK or NO OK depending on the results.

Another feature implemented in the Smart Object architecture is the capability of generates results. This facilitates the FTE to generate evidences and automatic reports even on board.



```
<PATH VERSION="" PROGRAM="A400M" PROTOTYPE="0007" TEST="F0002" DATE_TEST="13/03/2013">
  <OUTPUTS>
    <TEST_POINT ID="07" NAME="PATH_TAKE_OFF" RESULT="OK" TIME_TP="58946">
      <OUTPUTS>
        <OUTPUT ID="07_01_OAT" VALUE="7.937769" />
        <OUTPUT ID="07_01_QNH" VALUE="0" />
        <OUTPUT ID="07_01_PALT" VALUE="179" />
        <OUTPUT ID="07_01_DPCB" VALUE="1" />
        <OUTPUT ID="07_01_OVB" VALUE="1" />
        <OUTPUT ID="07_01_PACKS" VALUE="1" />
        <OUTPUT ID="07_01_NAI" VALUE="0" />
        <OUTPUT ID="07_01_WAI" VALUE="0" />
        <OUTPUT ID="07_01_OBOGS" VALUE="0" />
        <OUTPUT ID="07_01_OBIGGS" VALUE="0" />
        <OUTPUT ID="07_01_UTC" VALUE="08:53:21" />
        <OUTPUT ID="07_01_PWRTABLE" VALUE="95.52448" />
        <OUTPUT ID="07_01_PWRTRG" VALUE="90.469" />
        <OUTPUT ID="07_01_PWRTABTRGDIF" VALUE="5.055473" />
        <OUTPUT ID="07_01_PWRTRGCB" VALUE="0" />
        <OUTPUT ID="07_01_CAS" VALUE="81.16449" />
        <OUTPUT ID="07_02_UTC" VALUE="08:53:29" />
        <OUTPUT ID="07_02_PWRTABLE" VALUE="100.6119" />
        <OUTPUT ID="07_02_PWRTRG" VALUE="100.75" />
        <OUTPUT ID="07_02_PWRTABTRGDIF" VALUE="-0.1381073" />
        <OUTPUT ID="07_01_PWR1" VALUE="100.7272" />
        <OUTPUT ID="07_01_PWR2" VALUE="100.7429" />
        <OUTPUT ID="07_01_PWR3" VALUE="100.7429" />
        <OUTPUT ID="07_01_PWR4" VALUE="100.5772" />
        <OUTPUT ID="07_01_NP1" VALUE="841.3197" />
        <OUTPUT ID="07_01_NP2" VALUE="843.3735" />
        <OUTPUT ID="07_01_NP3" VALUE="843.6405" />
        <OUTPUT ID="07_01_NP4" VALUE="843.9855" />
        <OUTPUT ID="07_01_EGT1" VALUE="778.8824" />
        <OUTPUT ID="07_01_EGT2" VALUE="753.0271" />
        <OUTPUT ID="07_01_EGT3" VALUE="749.3499" />
        <OUTPUT ID="07_01_EGT4" VALUE="771.0232" />
        <OUTPUT ID="07_01_NPCB" VALUE="1" />
        <OUTPUT ID="07_02_PWRTRGCB" VALUE="1" />
        <OUTPUT ID="07_01_VIBE1" VALUE="1.13812" />
```

Fig. 4. Smart Object Output File

Additionally, Flight Test Means has developed and architecture for gathering results and store in a Database. From this database is possible to generate a PATM document fulfilled using the DOMA Application Framework.

2.4. DOMA

Document Oriented Monitoring and Analysis is a Framework developed for digitalized a Tagged Word document. The idea is avoid the use of paper on board. That implies that the documents used for taking notes and write the aircraft parameter resul must be written by an automatic system.

Starting from a Word document, DOMA is able to read the document that has been tagged using a strict labeling method which aid to identify the kind of parameter that must be associated with each token. For example, check-box, boxes for values and tables are associated to parameters from FTI or results of the Smart Objects calculations.

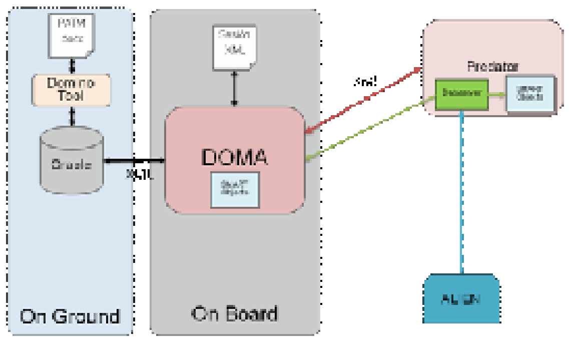
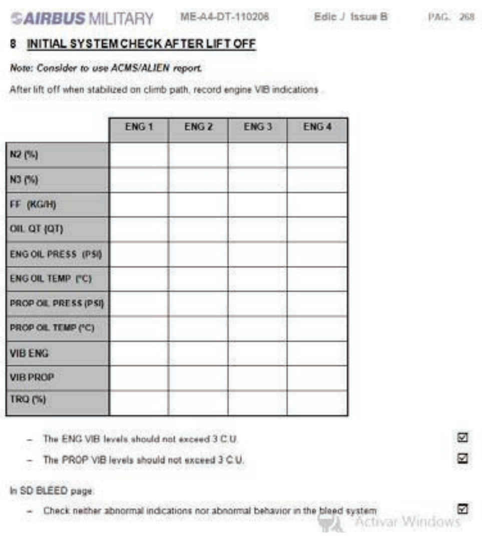


Fig. 5. DOMA Document management Process

The overall process consists of the following steps:

- The Word Document is generated according a Tagging system rules in order to know the parameter associated to each element of the document.
- DOMA on ground tool parsers the document and store the information in a Database.
- Before the test the FTE download the information in a Tablet and download the Digital Document. That allows to manage the document in a stand-alone mode. The document is stored in XML format
- During the Test, the data is gathered and stored in the tabled using XML file.
- Once the Test have finished, the data can be sent to Database.
- FTE can regenerate the Word Document with the information of the tests needed for fill the document.



AIRBUS MILITARY ME-A4-DT-110208 Edic J Issue B PAG. 268

8 INITIAL SYSTEM CHECK AFTER LIFT OFF

Note: Consider to use ACMS/ALIEN report.

After lift off when stabilized on climb path, record engine VIB indications .

	ENG 1	ENG 2	ENG 3	ENG 4
N2 (%)				
N3 (%)				
FF (KG/H)				
OIL QT (QT)				
ENG OIL PRESS (PSI)				
ENG OIL TEMP (°C)				
PROP OIL PRESS (PSI)				
PROP OIL TEMP (°C)				
VIB ENG				
VIB PROP				
TRQ (%)				

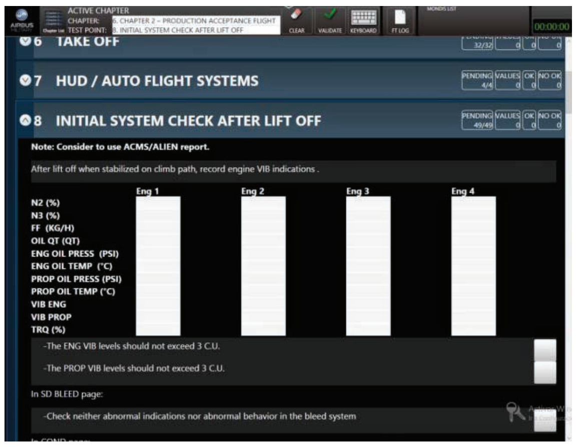
- The ENG VIB levels should not exceed 3 C.U. ☑
- The PROP VIB levels should not exceed 3 C.U. ☑

In SD BLEED page

- Check neither abnormal indications nor abnormal behavior in the bleed system ☑

Activar Windows

Fig. 6. Fragment of PATM Document in Word format



ACTIVE CHAPTER
CHAPTER: 6. CHAPTER 2 – PRODUCTION ACCEPTANCE FLIGHT
TEST POINT: 8. INITIAL SYSTEM CHECK AFTER LIFT OFF

00:00:00

6 TAKE OFF — PENDING 32/32, VALUES 0, OK 0, NO OK 0

7 HUD / AUTO FLIGHT SYSTEMS — PENDING 4/4, VALUES 0, OK 0, NO OK 0

8 INITIAL SYSTEM CHECK AFTER LIFT OFF — PENDING 49/49, VALUES 0, OK 0, NO OK 0

Note: Consider to use ACMS/ALIEN report.

After lift off when stabilized on climb path, record engine VIB indications .

	Eng 1	Eng 2	Eng 3	Eng 4
N2 (%)				
N3 (%)				
FF (KG/H)				
OIL QT (QT)				
ENG OIL PRESS (PSI)				
ENG OIL TEMP (°C)				
PROP OIL PRESS (PSI)				
PROP OIL TEMP (°C)				
VIB ENG				
VIB PROP				
TRQ (%)				

-The ENG VIB levels should not exceed 3 C.U.

-The PROP VIB levels should not exceed 3 C.U.

In SD BLEED page:

-Check neither abnormal indications nor abnormal behavior in the bleed system

Fig. 7. Fragment of Fig. 6. In the document digitalized in DOMA

DOMA has been developed using MVVM[5] pattern and WPF[6] as display technology. This technology is largely used on successful projects.[7]

2.4.1.MVVM and DOMA

The Model View ViewModel (MVVM) is an architectural pattern used in software engineering originated from Microsoft as a specialization of the Presentation Model design pattern introduced by Fowler. Largely based on the model–view–controller pattern (MVC) MVVM facilitates a clear separation of the development of the graphical user interface from the development of the business logic or back end logic known as the model The view model of MVVM is a value converter meaning that the view model is responsible for exposing the data objects from the model in such a way that those objects are easily managed and consumed. In this respect, the view model is more model than view, and handles most if not all of the view's display logic.

DOMA has been designed using MVVM pattern. This allows producing self contained modules consisting of Model and ViewModel parts for a certain object in the system, fully testing it using testing harnesses. ViewModel exposes properties and command objects a GUI (a View in the MVVM pattern) can display. GUI changes or modifications do not require any changes of the ViewModel.

2.4.2.WPF and DOMA

Windows Presentation Foundation (WPF) is a graphical subsystem for rendering user interfaces in Windows-based applications by Microsoft. WPF attempts to provide a consistent programming model for building applications and separates the user interface from business logic. WPF employs XAML[8], an XML-based language, to define and link various interface elements

DOMA has been designed using WPF for running in a Tablet running Windows 8.1. The View in the MVVM does not have any code behind. It is pure XAML having only GUI look design declaration and data binding specification. This feature could benefit for future implementations in another platforms like smart phones.

Conclusions

With the aid of the new technologies, Flight Test Data Processing in Airbus Defence and Space has developed a new set of tools which gives the user new possibilities for the real time monitoring. This new tools are the previous step to use expert systems in the near future.

References

[1] F. Hileman. “Real-Time Data Visualization Using Vector Graphics” http://www.vgdotnet.com/articles/real_time_vis.shtml

[2] Pasi Tuomainen, “High-performance WPF Charts - the Truth Revealed

http://www.arction.com/high_performance_wpf_charts_truth_revealed

[3] Microsoft “Dynamic Source Code Generation and Compilation” http://msdn.microsoft.com/en-us/library/650ax5cx(v=vs.110).aspx

[4] Standard ECMA-335
Common Language Infrastructure (CLI)
http://www.ecma-international.org/publications/standards/Ecma-335.htm

[5] Introduction to Model/View/ViewModel pattern for building WPF apps – John Gossman http://blogs.msdn.com/b/johngossman/archive/2005/10/08/478683.aspx

[6] Introduction to WPF .NET Framework 4

http://msdn.microsoft.com/en-us/library/aa970268.aspx

[7] Sameer Soni1, Pranali Dhete2, Shirish Patil3 and Dr B.B. Meshram4 Industrial Automation using Windows Presentation Foundation & Model View View-Model Pattern ISSN: 2278 – 1323

[8] XAML Overview (WPF) http://msdn.microsoft.com/en-us/library/ms752059.aspx