ArabiaGIS

Introduction to Design pattern MVVM

Jaffal Hasan

November 2009

AGENDA

- Design Patterns Overview
- Do ArabiaGIS needs a pattern?
- Patterns examples
- Overview of MVC-MVP and MVVM design patterns
- How To choose the appropriate pattern?
- MVC, MVP or MVVM with WPF ?
- MVVM
  - View Concept
  - ViewModel Concept
  - Model Concept
  - How it works?
- Next Step !

AGENDA

- Design Patterns Overview
- Do ArabiaGIS needs a pattern?
- Patterns examples
- Overview of MVC-MVP and MVVM design patterns
- How To choose the appropriate pattern?
- MVC, MVP or MVVM with WPF ?
- MVVM
  - View Concept
  - ViewModel Concept
  - Model Concept
  - How it works?
- Next Step !

Design Pattern Overview

- Set of guidelines
- Provide solutions to common software design problems
- Consists of one or several software design elements such as modules, interfaces, classes, objects, methods, functions, processes, threads, etc.,
- Relationships among the elements, and a behavioral description
- Example design patterns: Model/View/Controller

ArabiaGIS

Design Pattern Overview

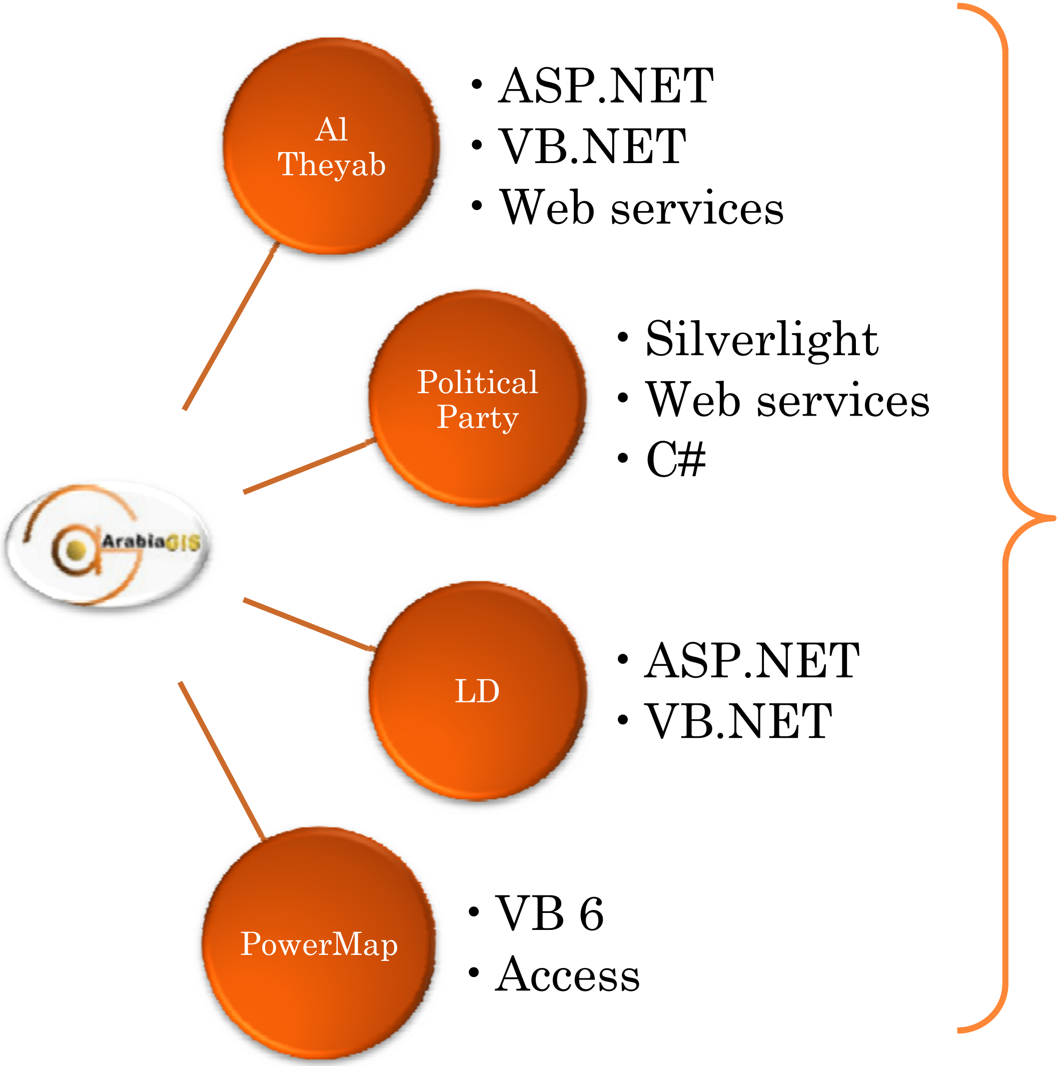
- Advantages:
  - Improve the structure of software
  - Simplify maintenance
  - Shared language for communicating
  - Separation of concerns
  - Minimize logic needed in views
  - Enhance testability
  - Reduce development time
  - Easy to customize applications

- Disadvantages:
  - Design pattern can be overkill in Simple UI

AGENDA

- Design Patterns Overview
- Do ArabiaGIS needs a pattern?
- Patterns examples
- Overview of MVC-MVP and MVVM design patterns
- How To choose the appropriate pattern?
- MVC, MVP or MVVM with WPF ?
- MVVM
  - View Concept
  - ViewModel Concept
  - Model Concept
  - How it works?
- Next Step !

Do we need a pattern?



Al Theyab

- ASP.NET
- VB.NET
- Web services

Political Party

- Silverlight
- Web services
- C#

LD

- ASP.NET
- VB.NET

PowerMap

- VB 6
- Access

CHAOTIC

DO WE NEED A PATTERN?

- Multiple projects
  - Big size projects – Long time projects
- Everyone speaks a different language
- No code maintainability
- Resources
  - Human resources: Developers – analysts – QA
  - Technical resources: Servers
  - Resource moving from project to project
  - Resource Sharing
  - Resource troubleshooting an application
    - Time loss in understanding
      - The structure
      - The technology

DO WE NEED A PATTERN?

- Why don’t use the same concepts, guidelines and rules?

UI, BUSINESS LOGIC AND DATA

Business applications consist of user interface (UI), business logic, and data models.

- When UI, business logic and data are collapsed into one object in rich users interface, it can lead to some of the following problems:
  - Difficult to use the data outside that object
  - Hard to change the UI, when UI and data are locked in the same object.
  - Hard to use multiple views of the same data.
  - Difficult to synchronize multiple view of the same data.

AGENDA

- Design Patterns Overview
- Do ArabiaGIS needs a pattern?
- Patterns examples
- Overview of MVC-MVP and MVVM design patterns
- How To choose the appropriate pattern?
- MVC, MVP or MVVM with WPF ?
- MVVM
  - View Concept
  - ViewModel Concept
  - Model Concept
  - How it works?
- Next Step !

Design Pattern Classifications

- Creational patterns
- Structural patterns
- **Behavioral patterns**

DESIGN PATTERNS EXAMPLES

- MVC
  - Model – View – Controller
- MVP
  - Model – View – Presenter
  - Introduced by Martin Fowler in 2004
- MVVM
  - Model – View – ViewModel
  - Originated from Microsoft as a specialization of the MVP
- MVC# and ASP.NET MVC
  - For web application (ASP.NET / VB.NET-C#.NET)

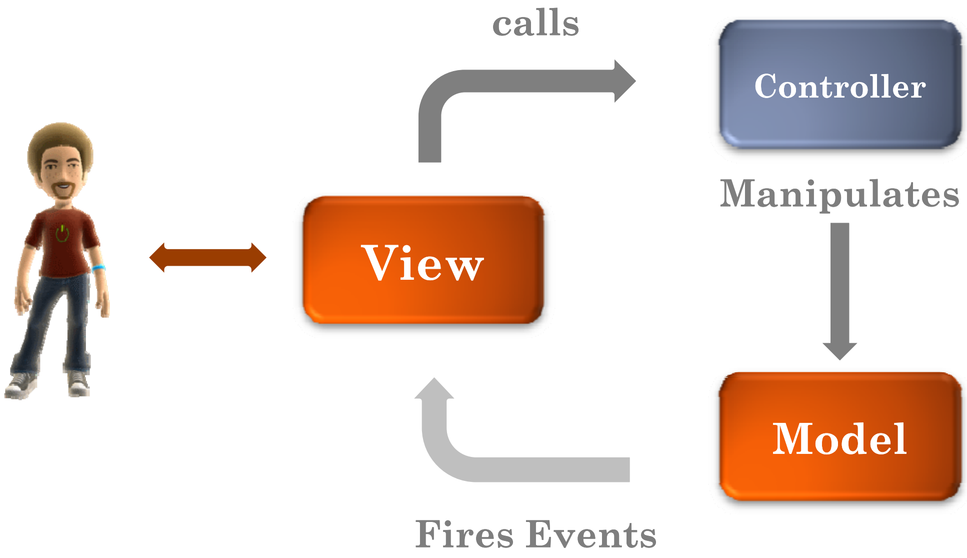
AGENDA

- Design Patterns Overview
- Do ArabiaGIS needs a pattern?
- Patterns examples
- Overview of MVC-MVP and MVVM design patterns
- How To choose the appropriate pattern?
- MVC, MVP or MVVM with WPF ?
- MVVM
  - View Concept
  - ViewModel Concept
  - Model Concept
  - How it works?
- Next Step !

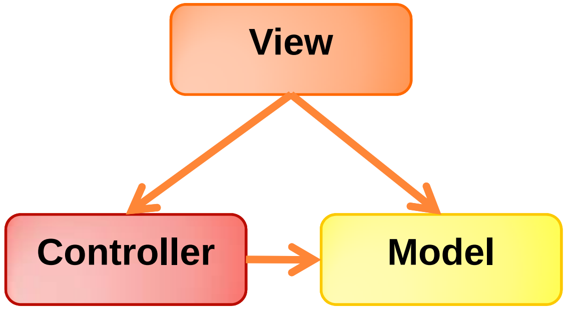
GOALS

- Separation of concerns
  - Decoupling the layers and components
  - Reducing development time (multi processes at time)
- Testability
- Flexibility
  - Minimal Code in UI
  - Changes in layers: DBMS
  - Change or use multiple platform to present data (mobile – web ..)

MVC : Model – View - Controller



MVC References Map



MVC

MVC :Model – View - Controller

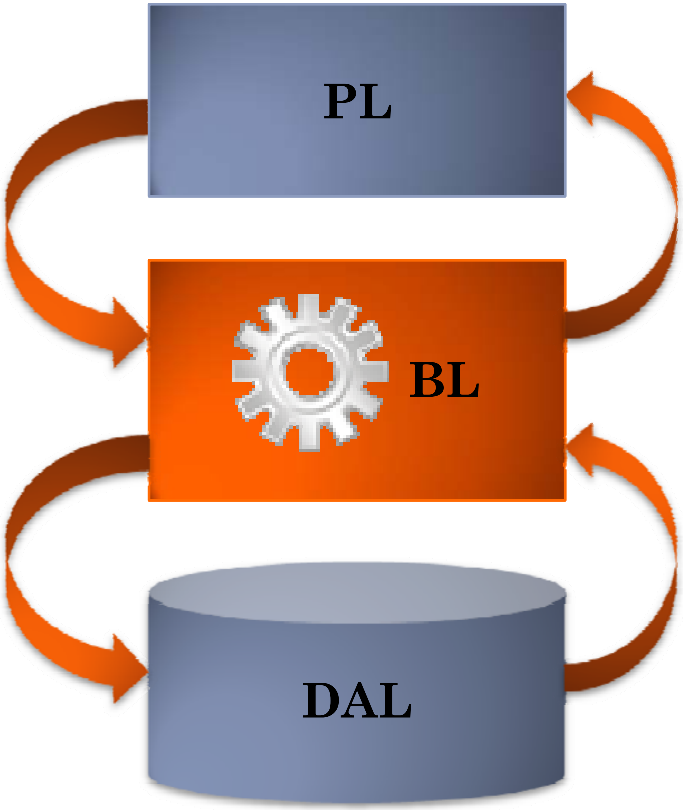
- Elements
  - **Model**: represents data and business rules/state
  - **View:** renders the data or state; visible layer
  - **Controller:** manages Views & user interaction; coordinates with one or more Models
- Guidelines
  - Controller doesn’t know anything about the View
  - Views can switch Controllers
  - Single Controller usable by multiple Views
  - View subscribes to Model change events

3 TIERS AND 3 LAYERS

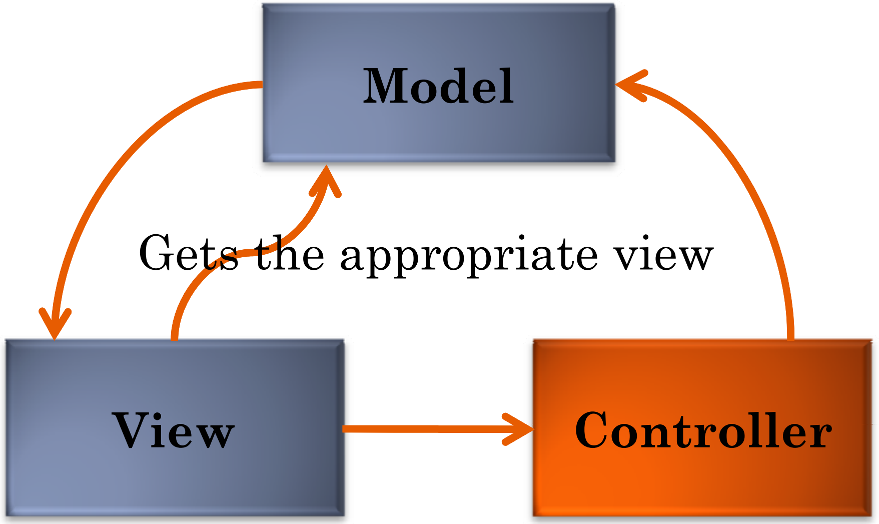
- N tier architecture is about splitting up an application in different (logical and or physical) layers, UI on a machine (or set of machines), Business logic and services on another…
- Layered architecture Development model where presentation, business logic and data are separated.

MVC VS. 3-LAYERS

- 3-Layers architecture



- MVC architecture

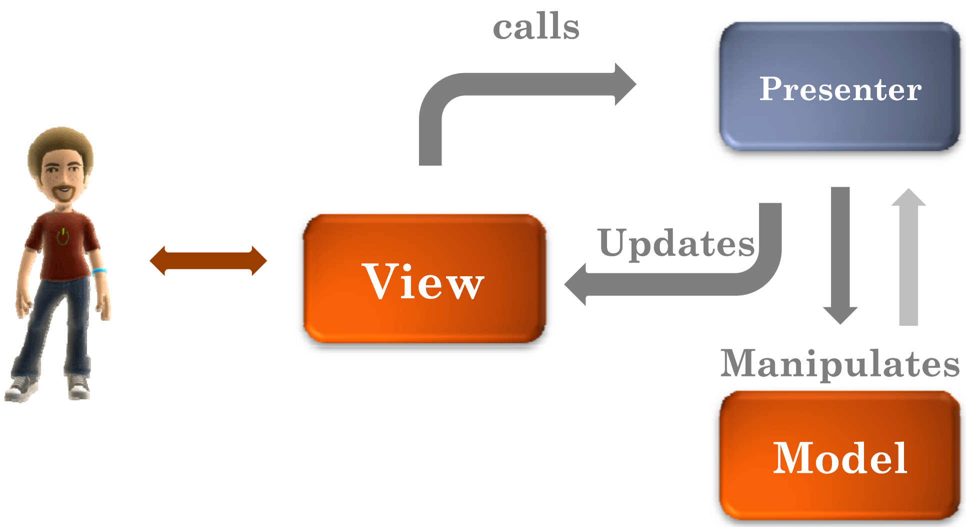


- The controller controls and Model presents – BL presents the data
- Linear vs. triangular
- Can be implemented together
- MVC has guidelines – PL no guidelines

MVC

- MVC comes in different flavors
  - ***MVC active model***
    - the model must notify the views to refresh the display
  - ***MVC passive model***
    - The controller modifies the model and then informs the view that the model has changed and should be refreshed

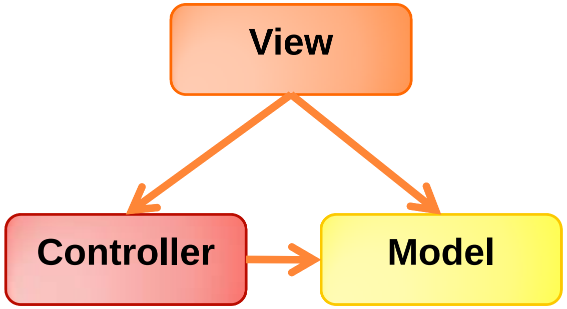
MODEL – VIEW - PRESENTER



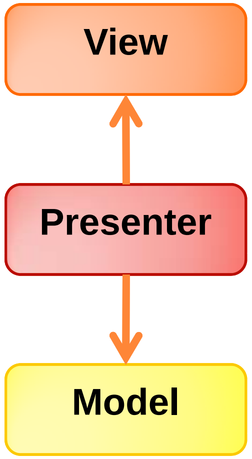
MODEL – VIEW - PRESENTER

- How does it differ from MVC? Presenter refers back to View but Controller does not
- Elements
  - **Model:** represents data and business rules/state
  - **View:** renders the data or state; visible layer
  - **Presenter:** manages Views & user interaction; coordinates with one or more Models; can update the View directly
- Guidelines
  - Presenter refers to an abstraction (interface or abstract base-class) of the View for testability
  - Presenter updates Model and the View
  - More testable than MVC
  - Less code behind than MVC
  - More separation of concepts than MVC

MVP References Map

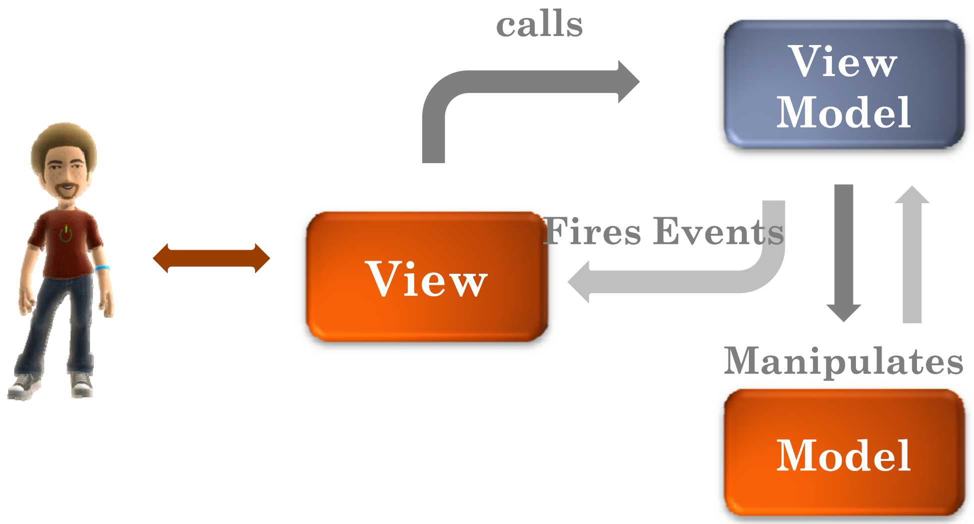


MVC



MVP

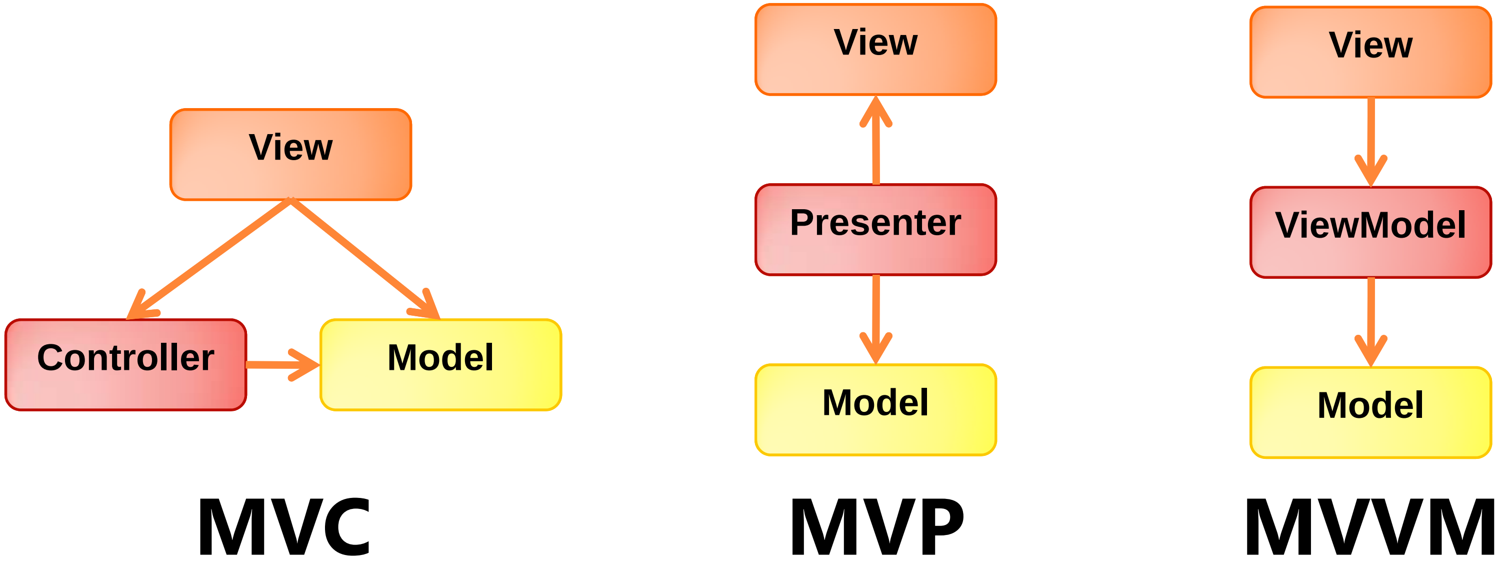
MODEL – VIEW - VIEWMODEL



MODEL – VIEW - VIEWMODEL

- How does it differ from MVP? ViewModel does not need a reference to a View
- Elements
  - **Model**: represents data and business rules/state
  - **View:** renders the data or state; visible layer
  - **ViewModel:** Coordinates with one or more Models; exposes properties for the View to bind to
- Guidelines
  - View knows about the ViewModel but not the Model
  - ViewModel knows about the Model but not the View
  - Model only knows about itself
  - View binds to properties in the ViewModel
  - ViewModel can combine state info and/or data from multiple Models
  - .Net XAML classes expose a DataContext property to which the ViewModel can be bound either declaratively or in code behind
  - Changes to properties in ViewModel automatically propagate to the View – no additional wiring needed!
  - Data changes made in the ViewModel, never the View
- **More testable than MVC than either MVC or MVP**

MVVM References Map



BENEFITS

- Modularity
  - **decoupling components**
  - allows each component to be versioned independently
  - worked on by individuals on team (UI person, DB person, etc)
- Flexibility
  - multiple Views for one Model (web frontend, desktop frontend, mobile frontend, etc)
  - replace one component (replace data storage from flat file to database)
- Maintainability
  - only change one component where bug exists, less risk in late changes
- **Testability**
  - each component communicates through contract so each component can be unit-tested independently

AGENDA

- Design Patterns Overview
- Do ArabiaGIS needs a pattern?
- Patterns examples
- Overview of MVC-MVP and MVVM design patterns
- How To choose the appropriate pattern?
- MVC, MVP or MVVM with WPF ?
- MVVM
  - View Concept
  - ViewModel Concept
  - Model Concept
  - How it works?
- Next Step !

MVC, MVP OR MVVM ?

- How to choose?
  - Based on the used technologies.

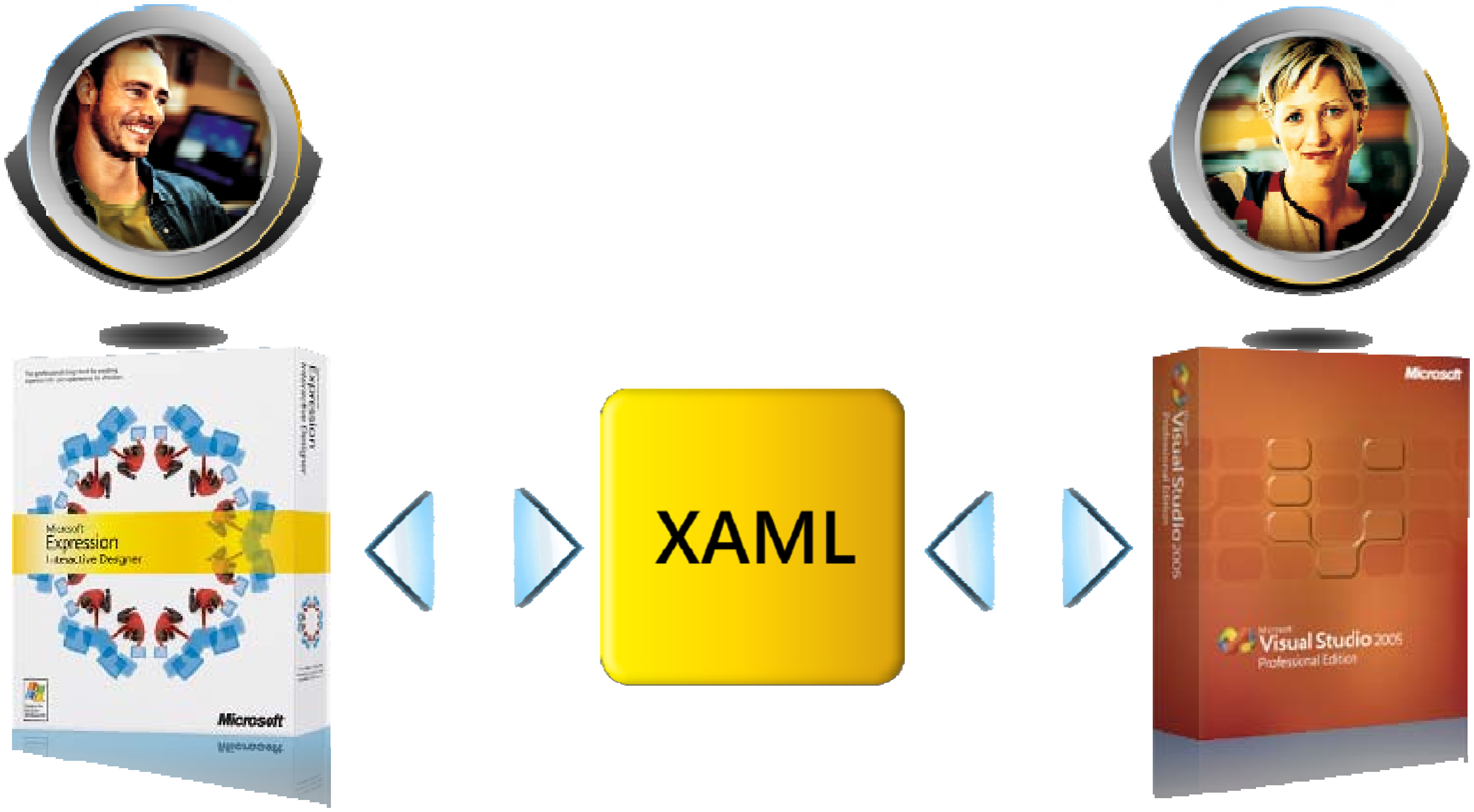
Agenda

- Design Patterns Overview
- Do ArabiaGIS needs a pattern?
- Patterns examples
- Overview of MVC-MVP and MVVM design patterns
- How To choose the appropriate pattern?
- MVC, MVP or MVVM with WPF ?
- MVVM
  - View Concept
  - ViewModel Concept
  - Model Concept
  - How it works?
- Next Step !

Windows Presentation Foundation WPF

MVVM pattern adaptation

WPF?


Microsoft
Expression
Interactive Designer
Microsoft
XAML
Microsoft
Visual Studio 2005
Professional Edition

WPF

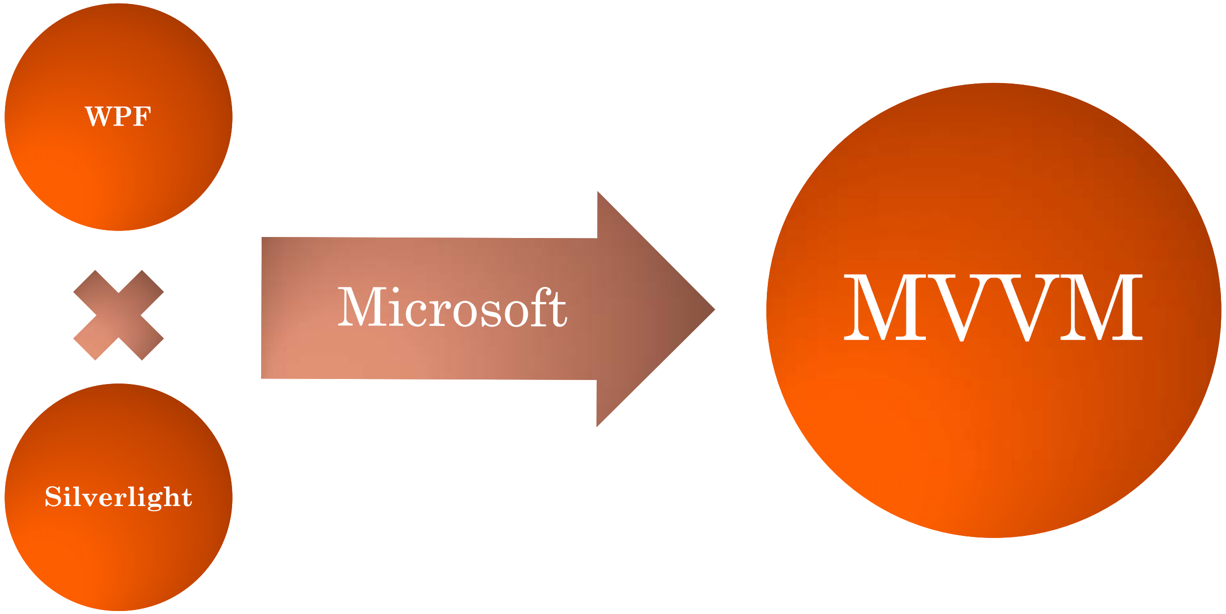
- WPF and Silverlight are UI development platforms
  - Developer specialized in user interface design and human-computer interaction
- Most powerful feature is the **two way binding**
- Most people's first attempts at WPF resemble their first forays into winforms, or even a VB Centric approach
  - Name all UI controls
  - Implement handlers for events coming from controls (i.e. Button "click, etc) directly in code behind
  - Store references to model objects in code behind
  - Write code directly populate named controls
- Results:
  - Coupled code-behind and XAML (View)
  - View has become storage for data: no unit testing
  - Not making use of two way binding.

MVC, MVP OR MVVM WITH WPF ?

- MVP can work with WPF
- But
  - Not taking full advantage of two way binding of WPF
  - You need to implement MVP style assessors for all your controls and write code to always callback to the view to set controls value.

MICROSOFT CHOICE !

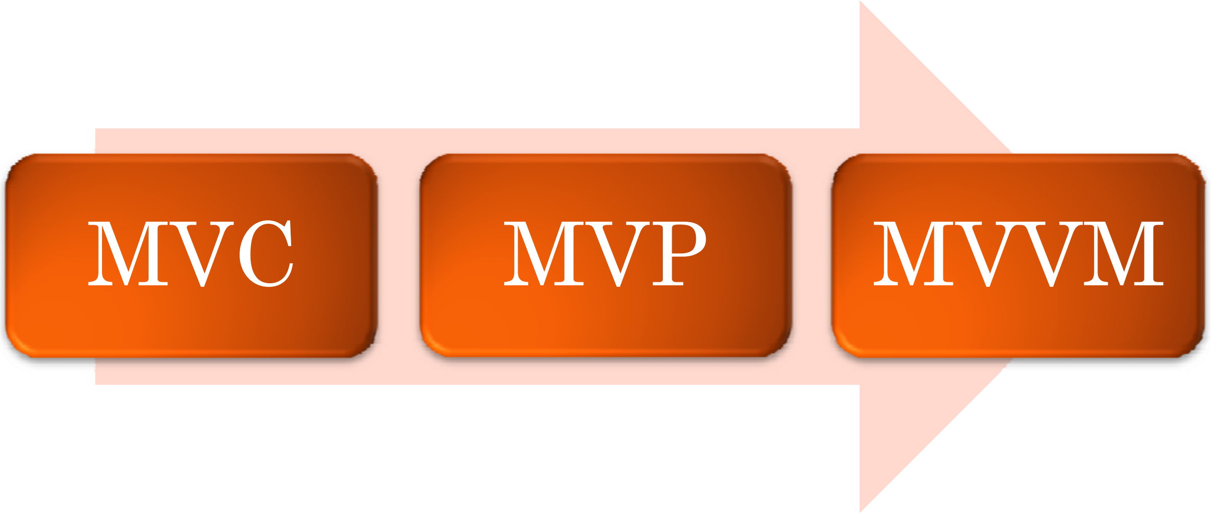
- Microsoft was using MVVM internally to develop WPF applications, such as Microsoft Expression Blend.
- MVVM is targeted at modern UI development platforms (WPF and Silverlight) [Wikipedia]
- MVVM was designed to make use of specific functions in WPF [Wikipedia]



AGENDA

- Design Patterns Overview
- Do ArabiaGIS needs a pattern?
- Patterns examples
- Overview of MVC-MVP and MVVM design patterns
- How To choose the appropriate pattern?
- MVC, MVP or MVVM with WPF ?
- MVVM
  - View Concept
  - ViewModel Concept
  - Model Concept
  - How it works?
- Next Step !

MVVM Origin

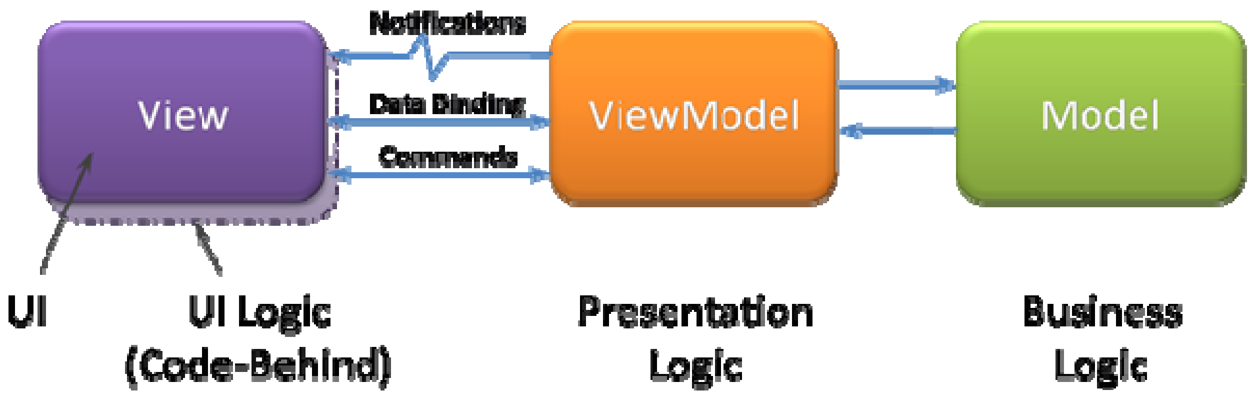

MVC

MVP

MVVM

- Originated from Microsoft as a specialization of the MVP design pattern introduced by Martin Fowler
- Specific for the Windows Presentation Foundation (WPF).
- Largely based on the Model-view-controller pattern (MVC)

Model – View - ViewModel



"ViewModel acts as a complete mirror of the view but it's a stand alone C# class – you can think of it as an adapter for the view"

Jason Dolinger
WPF Architect

Model – View - ViewModel

View

- UserControl based
- Xaml
- Minimal code behind
- Datacontext set to the associated VM
- No event handlers
- Data binding to VM (datas & commands)

ViewModel

- Implements INotifyPropertyChanged
- Expose Icommand
- Handle validation
- Adapter class between the View and the Model
- Listen to Model's events
- Testable

Model

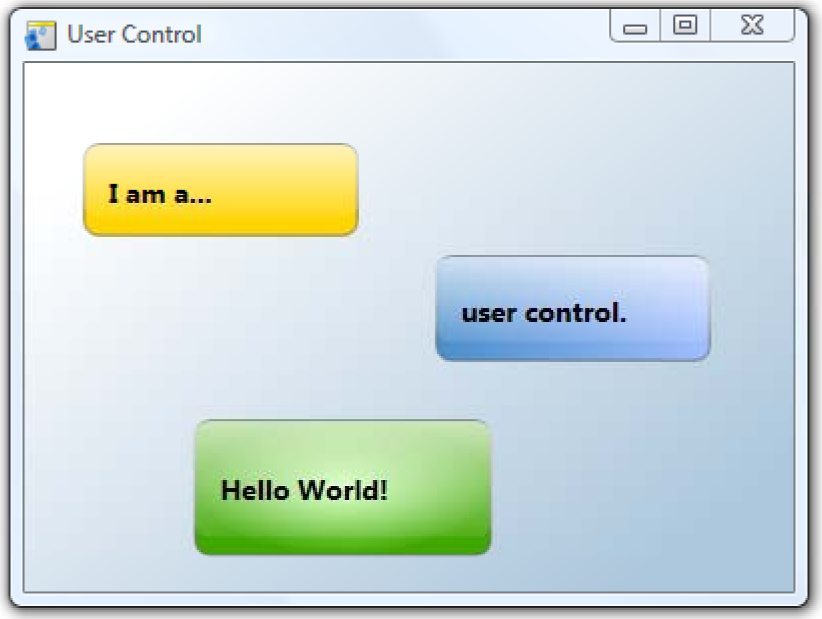
- No WPF related concepts
- Event based mechanism to signal changes to the ViewModel
- May already exists before the introduction of WPF mechanisms

VIEW

- UserControl based
- Xaml
- Minimal code behind
- No event handlers
- Datacontext set to the associated VM
- Data binding to VM (datas & commands)

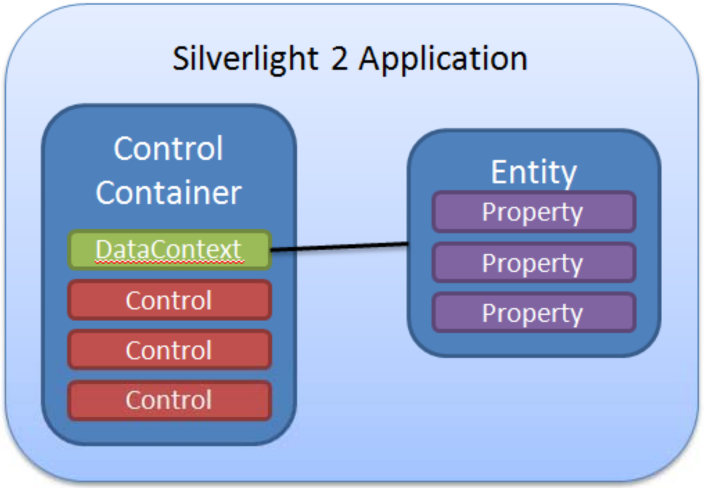
VIEW– USER CONTROL

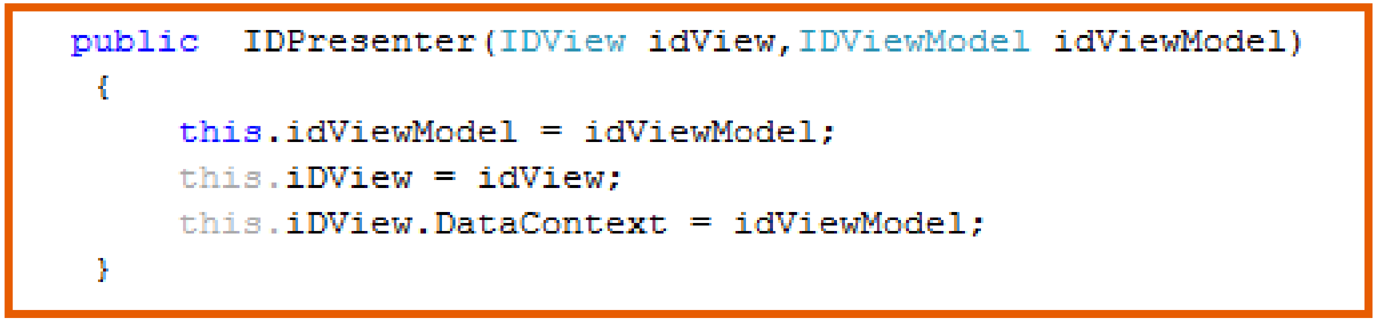
- A control created by a developer, usually by combining other controls, often intended for use in a specific application



VIEW– DATA CONTEXT

- The entity is set to the **DataContext** for a control.
- The **DataContext** refers to a source of data that can be bound to a target.
- The **DataContext** often is set to an instance of an entity.



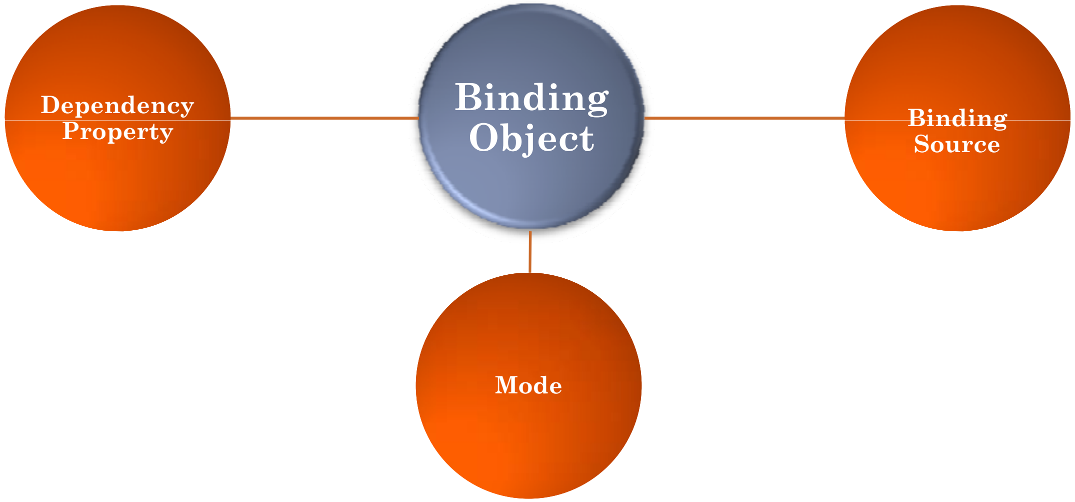


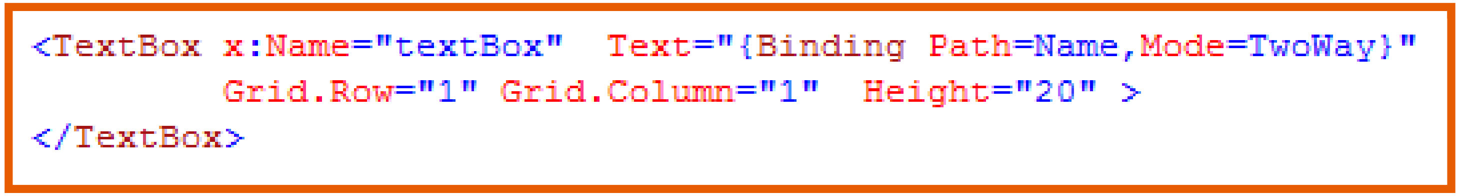
```
public  IDPresenter(IDView idView,IDViewModel idViewModel)
 {
     this.idViewModel = idViewModel;
     this.iDView = idView;
     this.iDView.DataContext = idViewModel;
 }
```

VIEW– DATA BINDING

- Simple way to display and interact with data
- A connection between the UI and a data object
- Allows data to flow between the two
- Automated
- **Databing push the changes to from ViewModel to View and From View To ViewModel**
- Path: Bind a Dependency property to a binding source
- Mode:
  - One Way
  - Two Way
  - One Time

VIEW– DATA BINDING





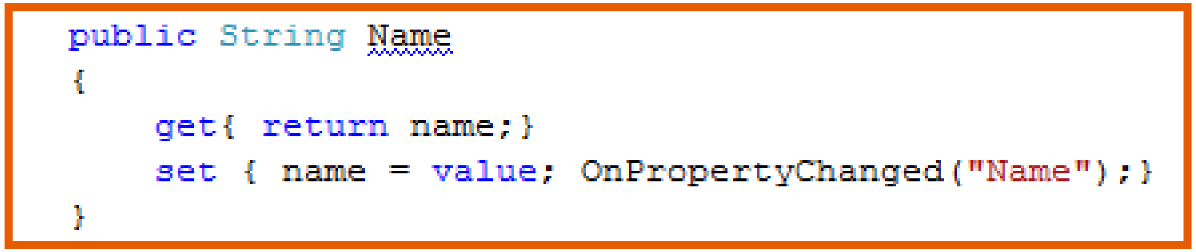
```
<TextBox x:Name="textBox"  Text="{Binding Path=Name,Mode=TwoWay}"
         Grid.Row="1" Grid.Column="1"  Height="20" >
</TextBox>
```

VIEWMODEL

- Implements INotifyPropertyChanged
- Expose Icommand
- Handle validation
- Adapter class between the View and the Model
- Listen to Model's events
- Testable

VIEWMODEL– INOTIFYPROPERTYCHANGED

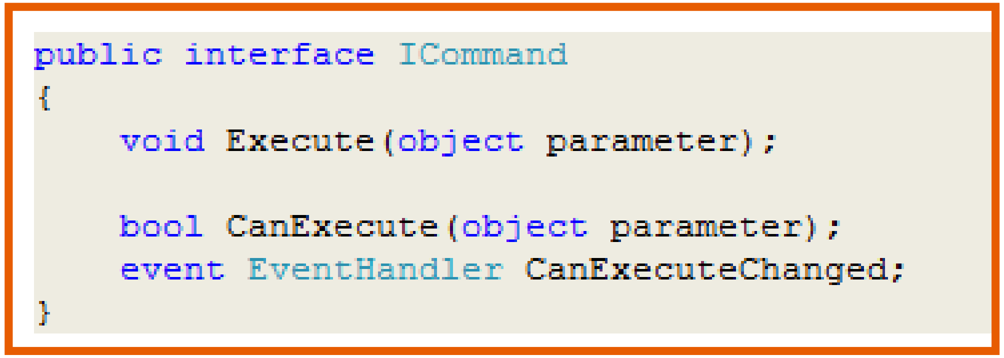
- The INotifyPropertyChanged interface is used to notify clients, typically binding clients, that a property value has changed.
- INotifyCollectionChanged for collections (Observable collections): will fire NotifyCollectionChanged event when items added/removed



```
public String Name
{
     get{ return name;}
     set { name = value; OnPropertyChanged("Name");}
}
```

ViewModel– ICommand

- Allow Multiple source to invoke it



```
public interface ICommand
{
    void Execute(object parameter);

    bool CanExecute(object parameter);
    event EventHandler CanExecuteChanged;
}
```

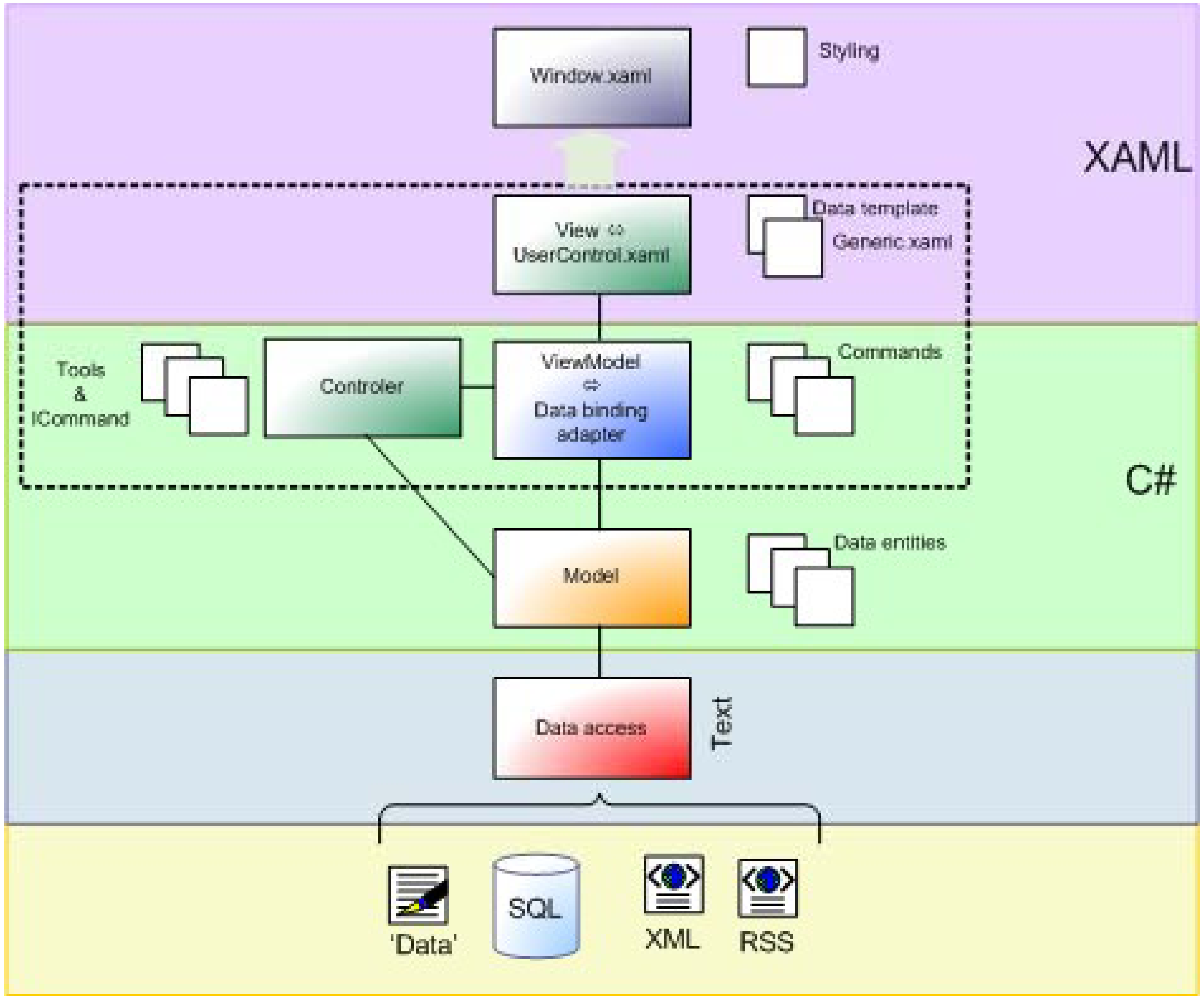
“you are on the right track when you almost NEVER have to name a control with x:Name”

Jason Dolinger
WPF Architect

MODEL

- No WPF related concepts
- Event based mechanism to signal changes to the ViewModel
- May already exists before the introduction of WPF mechanisms

The Full Image



DEMO

NEXT STEP?

- Web Applications appropriate design pattern?
  - MVC Sharp
  - ASP.NET MVC

Happy Coding !

THANKS FOR LISTENING.

Jaffal Hasan
hjaffal@arabiagis.com
009613926130