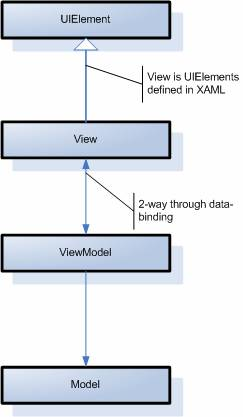
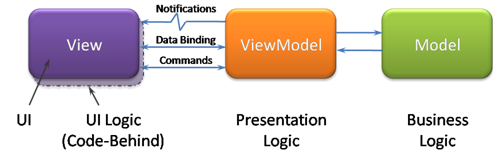
**Model View ViewModel (MVVM)**

 O MVVM é um pattern, que seria um padrão, que foi criado em 2005, por John Gossman, um dos arquitetos do WPF e Silverlight na Microsoft. O MVVM assemelha-se em alguns aspectos o MVC (Model View Controller) e ao MVP (Model View Presenter), podemos até dizer que o MVVM é uma especialização do MVP adaptado para a arquitetura do WPF E Silverlight. O MVVM, visa estabelecer uma clara separação de responsabilidades em uma aplicação WPF e Silverlight, mantendo uma espécie de faça de entre o Modelo de objetos e a View que é a interface, com a qual o usuário interage. Entenda essa divisão na imagem a seguir:

Como podemos perceber na imagem a cima, a camada Model (Modelo) não conhece a View( Camada de apresentação ) e vice-versa, na verdade a View conhece a ViewModel e se comunica com ela através do mecanismo de binding. A comunicação da View e ViewModel ocorre da seguinte maneira, apresentada na imagem abaixo:



A View através do databinding interage com a ViewModel notificando a ocorrência de eventos e o disparo de comandos. A ViewModel por sua vez, responde a esta notificação realizando alguma ação no modelo; seja obtendo alguma dado, atualizando ou inserindo informações no modelo.

A responsabilidade da **View** é definir a aparência ou estrutura que o usuário vê na tela. O ideal é que o codebehind da view, contenha apenas a chamada ao método InitializeComponent dentro do construtor, ou em alguns casos, código que manipule os controles visuais, ou crie animações; algo que é mais difícil de fazer em XAML. A View se liga ao ViewModel, através da propriedade DataContext que é setada para a classe ViewModel correspondente à aquela View.

A responsabilidade da **ViewModel** no contexto do MVVM, é disponibilizar para a View uma lógica de apresentação. A View Model não tem nenhum conhecimento específico sobre a view, ou como ela implementada, nem o seu tipo. A ViewModel implementa propriedades e comandos, para que a View possa preencher seus controles e notifica a mesma, caso haja alteração de estado; seja através de eventos ou notificação de alteração.

O **Model** no MVVM, encapsula a lógica de negócios e os dados. O Modelo nada mais é do que o Modelo de domínio de uma aplicação, ou seja, as classes de negócio que serão utilizadas em uma determinada aplicação. O Modelo também contém os papéis e também a validação dos dados de acordo com o negócio, cuja aplicação em questão visa atender.

O MVVM permite a você ter uma visão, da clara separação da Interface com o usuário( View ), sua lógica de apresentação (ViewModel) e os seus Dados(Model).

Abaixo um link para melhor entendimento do assunto:

Curso de MVVM: O que é MVVM? – Aula Demonstrativa:

<https://www.youtube.com/watch?v=EEc18BUrl7o>