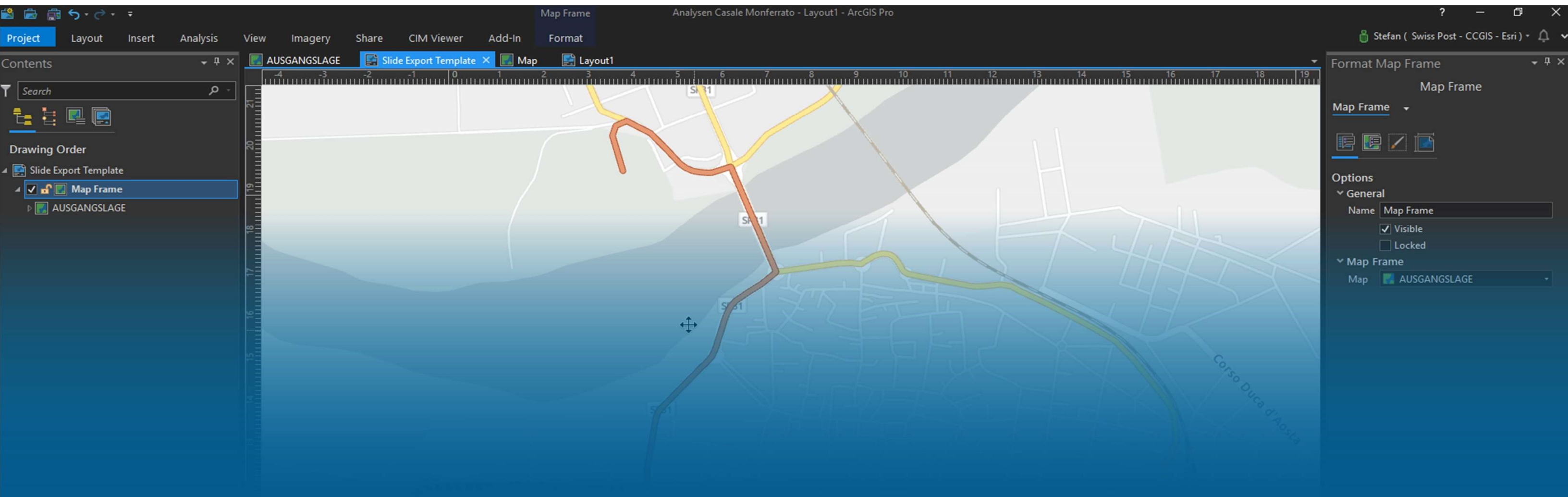


AddIn Migration Lessons learned

ArcMap → ArcGIS Pro

Stefan Schläfli

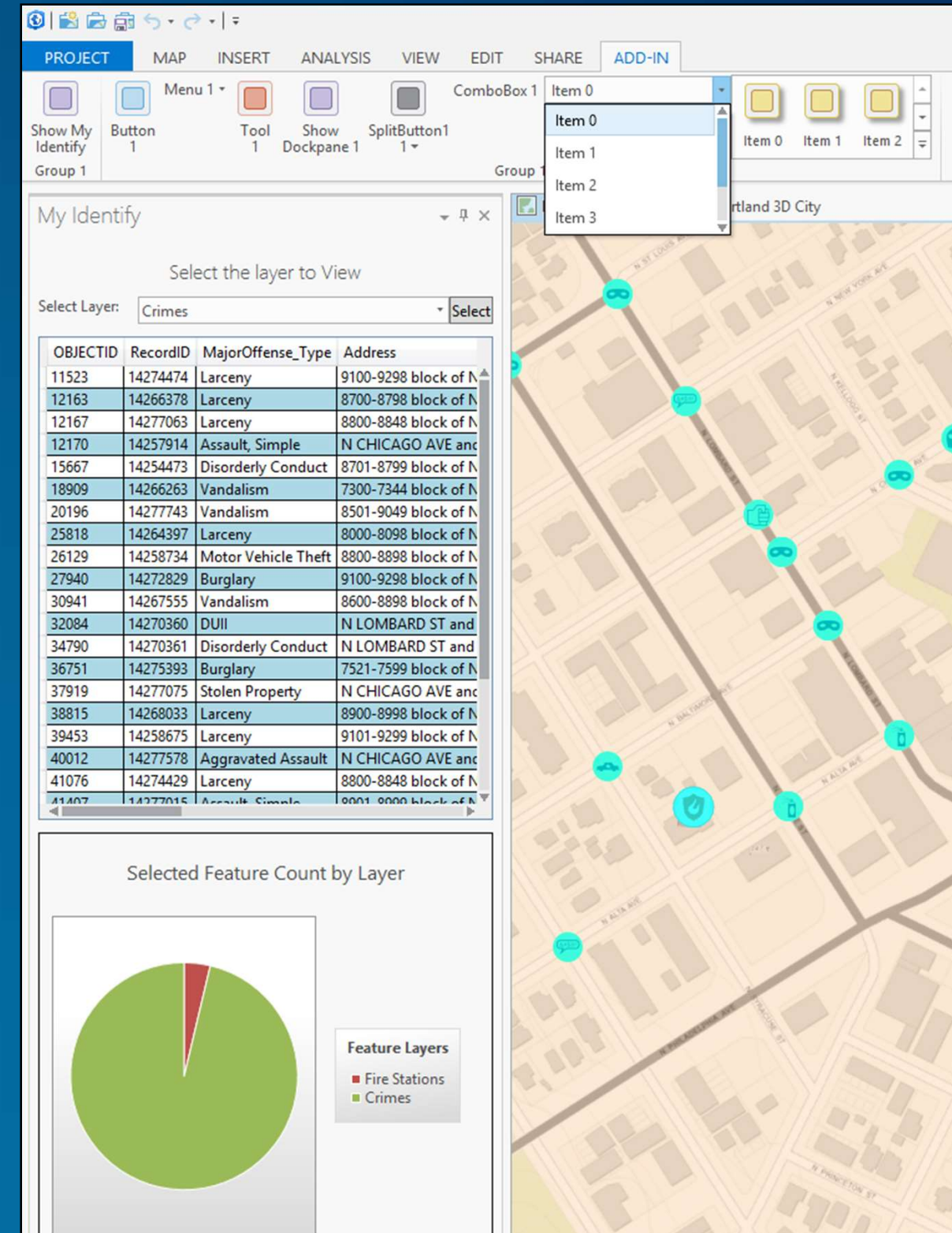


Considerations for AddIn Migration

- Changes in AddIn Framework (.daml, MVVM Integration)
- ArcObjects → ProSDK
- Windows Forms → WPF
- ProArchitecture: #maps/layouts ≥ 0 , multithreading

What is an ArcGIS Pro add-in ?

- **Extends ArcGIS Pro through:**
 - Buttons
 - Tools
 - Dockpanes
 - Embeddable control
 - ..
- **Packaged within a single, compressed file with an .esriaddinX file extension**
 - c:%Homepath%\Documents\ArcGIS\AddIns\ArcGISPro
- **AddIn Assembly Cache Folder:**
 - C:\Users\<user>\AppData\Local\ESRI\ArcGISPro\AssemblyCache



What are the ArcGIS Pro Add-in core components?

- **Declarative-based framework to define the UI elements**
 - Declarative framework is defined in a config.daml file
 - XML formatted, contains ArcGIS Pro framework elements (buttons, dockpane, galleries) and Add-in UI elements
- **The Module class**
 - Hub and central access point for each add-in
 - Similar to the Extension object used in the ArcObjects 10.x framework
 - Singletons instantiated automatically by the Framework

DAML

```
<ArcGIS defaultAssembly="LucatAddIn.dll" defaultNamespace="LucatAddIn" xmlns="http://schemas.esri.com/DADF/Registry" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <AddInInfo id="{0d2eeedd-bba5-4b40-b0e8-000000000000}" version="1.0" desktopVersion="2.2">...</AddInInfo>
  <modules>
    <insertModule id="LucatAddIn_Module" className="LucatModule" autoLoad="false" caption="Module1">
      <!-- uncomment to have the control hosted on a separate tab-->
      <tabs>...</tabs>
      <groups>
        ...
        <group id="LucatAddIn_Group1" keytip="LUCAT" caption="LUCAT" appearsOnAddInTab="true">
          <!-- host controls within groups -->
          <button refID="LucatAddIn_DpLucat_ShowButton" size="large" />
        </group>
      </groups>
      <controls>
        <!-- add your controls here -->
        <button id="LucatAddIn_DpLucat_ShowButton" keytip="LUCAT öffnen" caption="LUCAT öffnen" className="DpLucat_ShowButton" loadOnClick="true" smallImage="...">
          <tooltip heading="Lucat AddIn öffnen">Lucat öffnen</tooltip>
        </button>
      </controls>
      <dockPanels>
        <dockPanel id="LucatAddIn_DpLucat" caption="LUCAT Themenkatalog" className="DpLucatViewModel" dock="group" dockWith="esri_core_contentsDockPanel">
          <content className="DpLucatView" />
        </dockPanel>
      </dockPanels>
    </insertModule>
  </modules>
</ArcGIS>
```



What is the ArcGIS Pro SDK for .NET?

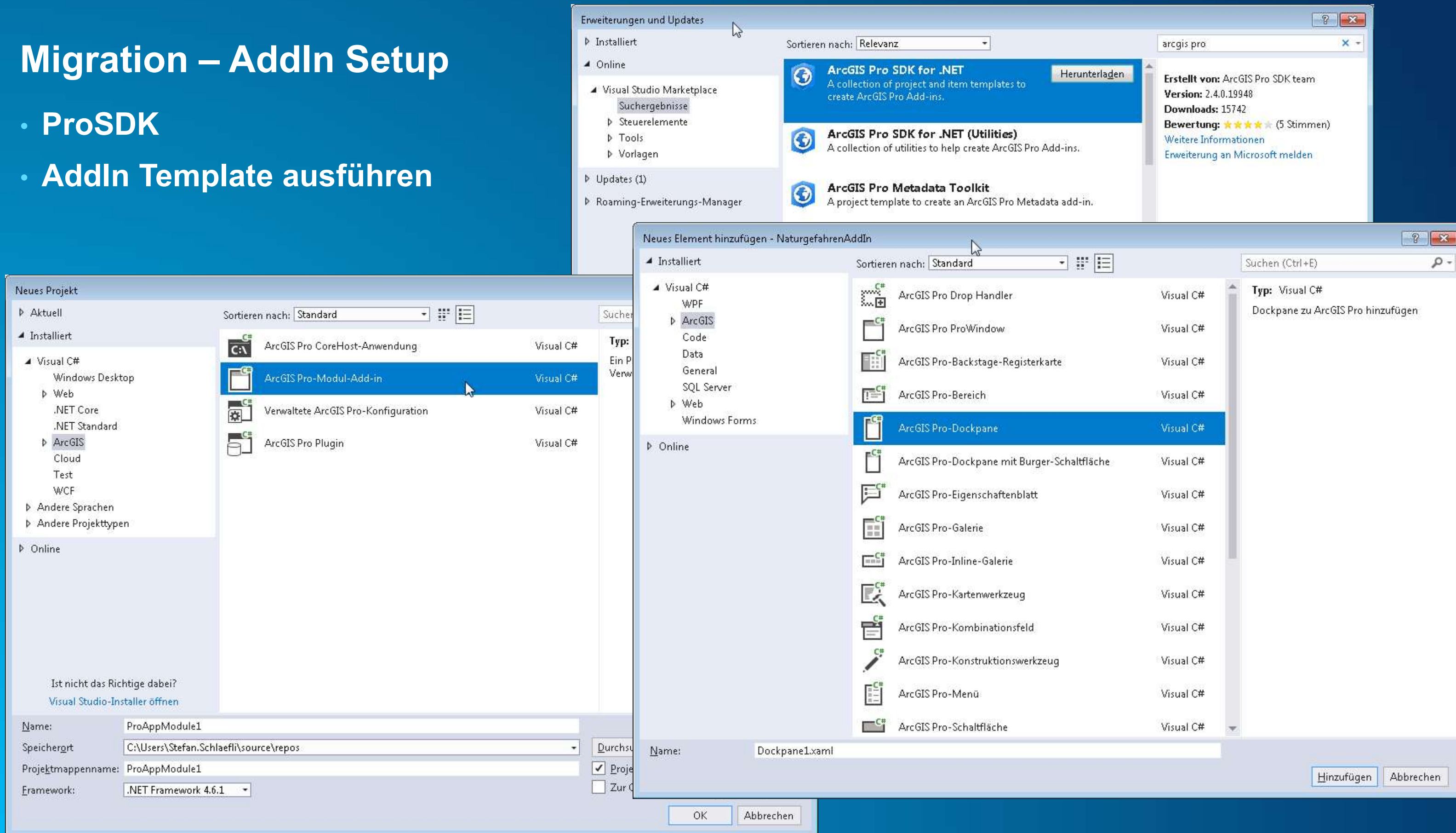
- ArcGIS Pro SDK features and patterns
- 64-bit platform
- UI based on WPF (Windows Presentation Foundation) / .NET 4.6.1 +
 - Note: .NET 4.8 at ArcGIS Pro release 2.5
- MVVM pattern (Model-View-ViewModel)
- Asynchronous Patterns: Multiple threads

Migration aspects

- **VS Extension: Pro SDK**
 - AddIn Template für AddIn Projekte
- **Esri ViewModel Integration with Dockpanes**
 - INotifyChanged Event derived in Pane() class
 - SetProperty() für Bindings
- **Events – Bindings**
- **Async Tasks**

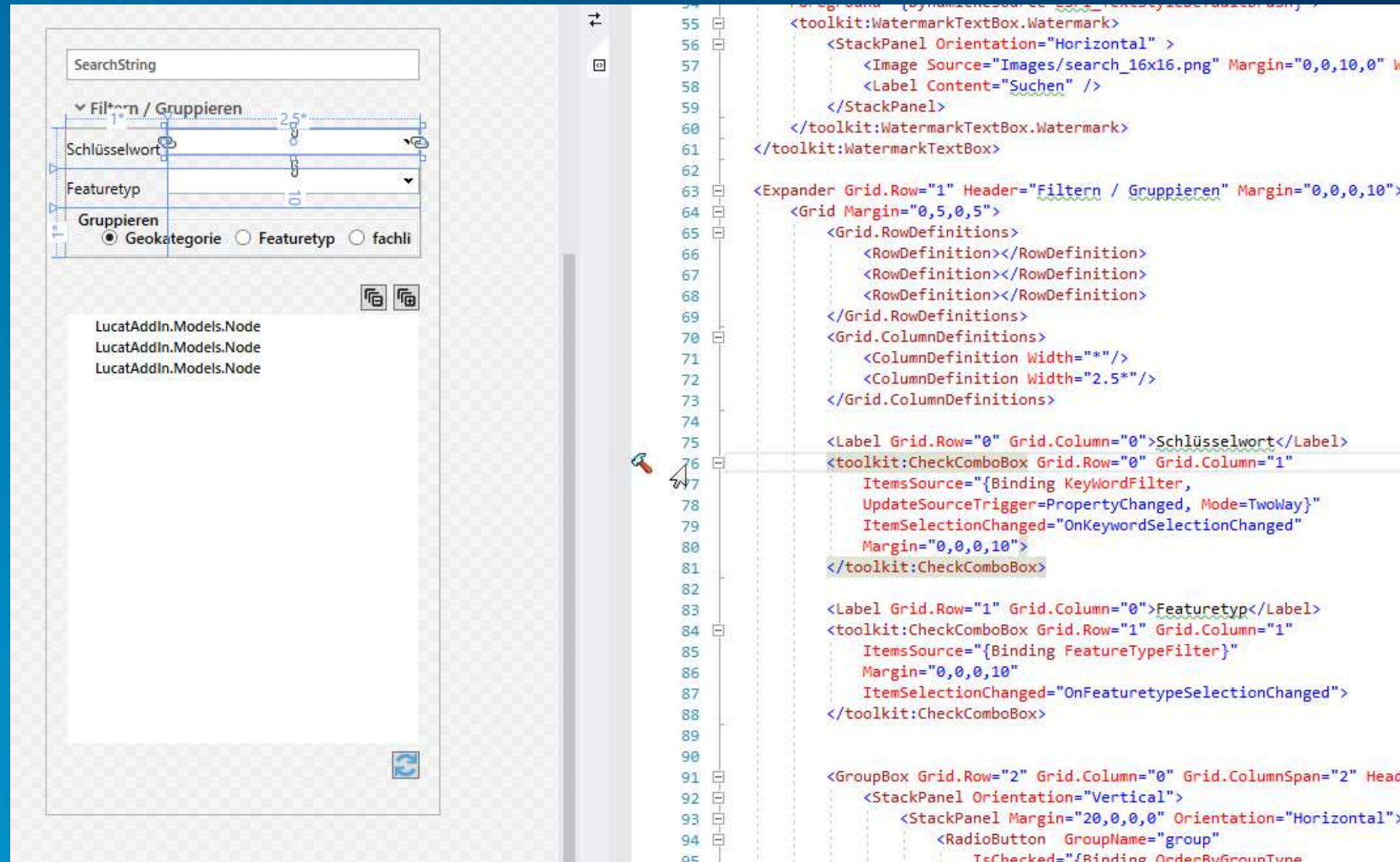
Migration – AddIn Setup

- ProSDK
- AddIn Template ausführen



Migration - View

- UI: Build View in .xaml
 - identify binding props



The image displays a WPF application in two parts: a visual design on the left and the underlying XAML code on the right.

Visual Design (Left): The UI features a search bar at the top labeled "SearchString". Below it is a section titled "Filtern / Gruppieren" (Filter / Group). This section contains a "Schlüsselwort" (Keyword) field, a "Featuretyp" (Feature type) dropdown menu, and a "Gruppieren" (Group) section with three radio buttons: "Geokategorie" (selected), "Featuretyp", and "fachli". Below these controls is a list of "LucatAddIn.Models.Node" items. The design includes dimension lines indicating spacing and alignment.

XAML Code (Right): The code defines the visual structure and data binding. It starts with a `toolkit:WatermarkTextBox` for the search bar, followed by an `Expander` for the "Filtern / Gruppieren" section. Inside the expander is a `Grid` with row and column definitions. The grid contains labels for "Schlüsselwort" and "Featuretyp", and `toolkit:CheckComboBox` controls for both, with `ItemsSource` and `ItemSelectionChanged` bindings. Below the grid is a `GroupBox` containing a `StackPanel` with a `RadioButton` for grouping, also with a binding for `IsChecked`.

```
55 <toolkit:WatermarkTextBox.Watermark>
56 <StackPanel Orientation="Horizontal" >
57 <Image Source="Images/search_16x16.png" Margin="0,0,10,0" W
58 <Label Content="Suchen" />
59 </StackPanel>
60 </toolkit:WatermarkTextBox.Watermark>
61 </toolkit:WatermarkTextBox>
62
63 <Expander Grid.Row="1" Header="Filtern / Gruppieren" Margin="0,0,0,10">
64 <Grid Margin="0,5,0,5">
65 <Grid.RowDefinitions>
66 <RowDefinition></RowDefinition>
67 <RowDefinition></RowDefinition>
68 <RowDefinition></RowDefinition>
69 </Grid.RowDefinitions>
70 <Grid.ColumnDefinitions>
71 <ColumnDefinition Width="*" />
72 <ColumnDefinition Width="2.5*" />
73 </Grid.ColumnDefinitions>
74
75 <Label Grid.Row="0" Grid.Column="0">Schlüsselwort</Label>
76 <toolkit:CheckComboBox Grid.Row="0" Grid.Column="1"
77 ItemsSource="{Binding KeywordFilter,
78 UpdateSourceTrigger=PropertyChanged, Mode=TwoWay}"
79 ItemSelectionChanged="OnKeywordSelectionChanged"
80 Margin="0,0,0,10">
81 </toolkit:CheckComboBox>
82
83 <Label Grid.Row="1" Grid.Column="0">Featuretyp</Label>
84 <toolkit:CheckComboBox Grid.Row="1" Grid.Column="1"
85 ItemsSource="{Binding FeatureTypeFilter}"
86 Margin="0,0,0,10"
87 ItemSelectionChanged="OnFeaturetypeSelectionChanged">
88 </toolkit:CheckComboBox>
89
90
91 <GroupBox Grid.Row="2" Grid.Column="0" Grid.ColumnSpan="2" Head
92 <StackPanel Orientation="Vertical">
93 <StackPanel Margin="20,0,0,0" Orientation="Horizontal">
94 <RadioButton GroupName="group"
95 IsChecked="{Binding OrderByGroupType,
```

Migration - Viewmodel

- refactor code behind stuff for binding architecture
- setup public properties in viewmodel
- apply SetProperty() in property setters

```
2 references | Stefan Schlaefli, 230 days ago | 1 author, 2 changes
188 public ObservableCollection<string> FeatureTypeFilter
189 {
190     get { return _FeatureTypeFilter; }
191     set
192     {
193         SetProperty(backingField: ref _FeatureTypeFilter, value, property: () => FeatureTypeFilter);
194     }
195 }
196
```

Migration – Models/Businesslogic

- **implement business logic in vm and model classes**



Good To Know - I

- **Refactoring:**
 - Resharper/VS refactoring does not touch config.daml
 - config.daml is not updated! Need to take care manually
- **Renaming Dockpanes**
 - generated partial xaml code behind classes e.g. (LocationFinderView.xaml.cs) are not correctly refactored.
 - config.daml is not updated! Need to take care manually
- **Moving Pro Controls to different folders:**
 - config.daml is not updated! Need to take care manually
- **Embedded Resources for AddIns**

Good to know - II

Styles

- <https://github.com/Esri/arcgis-pro-sdk/wiki/proguide-style-guide>
- esri styles ensure compatibility for dark/light mode
- some ui controls have default esri styles applied > apply esri styles if not

Community Samples

<https://github.com/Esri/arcgis-pro-sdk-community-samples>

Conclusion

- Migration of AddIns is not as bad as it might seem
- Effort proportional to grade of map/layer interaction
- Separation of Code and UI is a good thing
- ProSDK:
 - Lots of debuggable community samples around
 - Some classes like Geometry and Database still familiar
- Finally:
 - `if (codeQualityIsGood) { migrationCanBeSmooth == true }`
 `else { migrationIsUgly == true }`



