

Autor: Lukáš Zdechovan
Školiteľ: Mgr. Ondrej Svačina

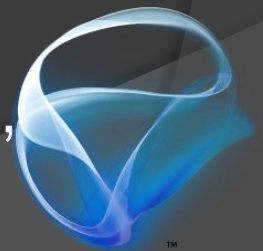
RIA V PODNIKOVÝCH APLIKÁCIÁCH A .NET RIA SERVICES

**Katedra aplikovanej informatiky,
Univerzita Komenského v Bratislave,
Fakulta matematiky, fyziky a informatiky**



Cieľ práce

- Analyzovať, ako je možné použiť RIA pri budovaní podnikových aplikácií.
- Naštudovať architektonické a návrhové vzory a navrhnúť, ktoré je vhodné použiť pri architektúre a návrhu, aby aplikácia spĺňala čo najviac bežných požiadaviek na vysokej úrovni.
- Popísať životný cyklus Silverlight RIA aplikácie založenej na zvolených vzoroch s využitím frameworku WCF(.NET) RIA Services.
- Implementovať ukážkovú (prototypovú) aplikáciu, ktorou overím zmysel predošlých analýz.



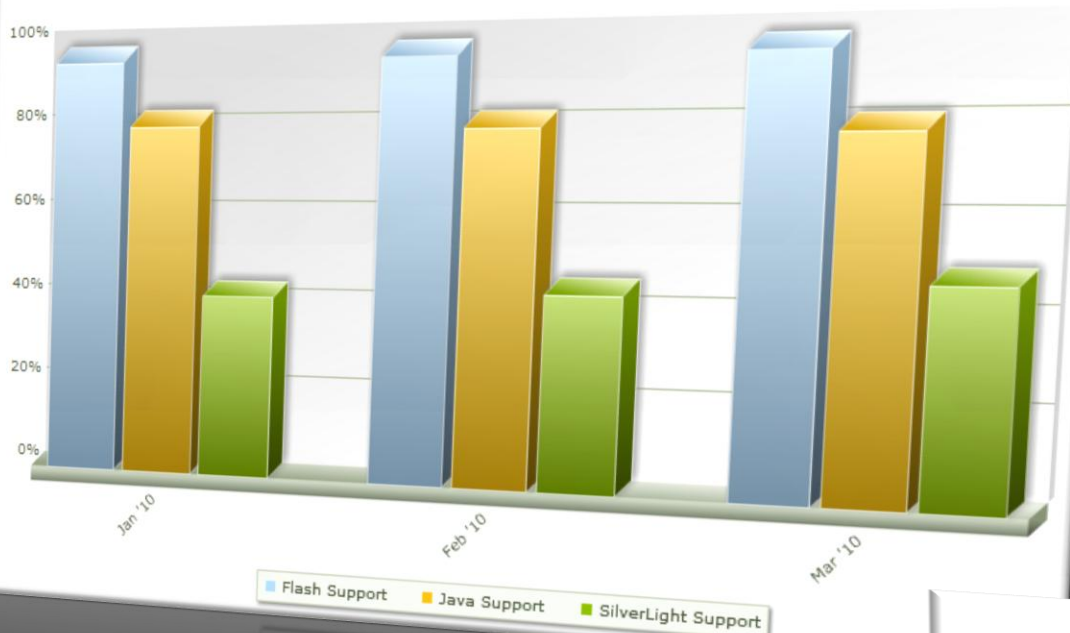
Štruktúra práce

- ⦿ Prehľad RIA technológií (Flash/Flex, JavaFX, Silverlight, AJAX+HTML5) a možnosti ich využitia v našom kontexte (budovanie podnikových aplikácií).
- ⦿ Náhľad na obvyklé požiadavky na podnikové aplikácie z technického pohľadu (bezpečnosť, rozšíriteľnosť + spravovateľnosť > modulárnosť..)
- ⦿ Analýza a popis použitia architektúr, ktoré môžeme uvažovať v našom kontexte.
- ⦿ Popísanie návrhových vzorov, ktoré odporúčam využiť pri návrhu RIA podnikových aplikácií.
- ⦿ Využitie analýz pri návrhu a implementácii aplikácie.

Rich Internet Application

- ◉ **Flex, AIR**
dobrá podpora na klientských staniciach (Flash)
open-source (MPL), MXML (GUI), CSS
ActionScript 3 (Specification ECMA-262)
- ◉ **JavaFX**
stačí bežný JRE, nezáujem vývojárov
viazanosť na Javu, JavaFX Script (GNU GPL), CSS
- ◉ **Silverlight**
nová platforma, brat WPF (Rich Desktop Application)
XAML (podobný MXML), štýly ako XAML resource
C#, VB, IronRuby, IronPython
- ◉ **HTML5 + JavaScript**
aké zmeny prinesie z pohľadu RIA?

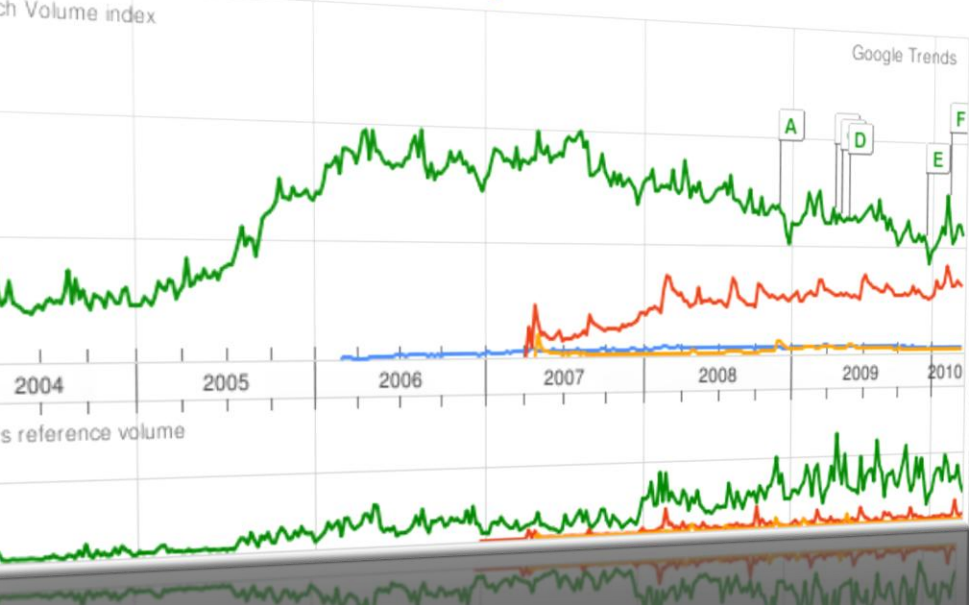
Rich Internet Application Usage



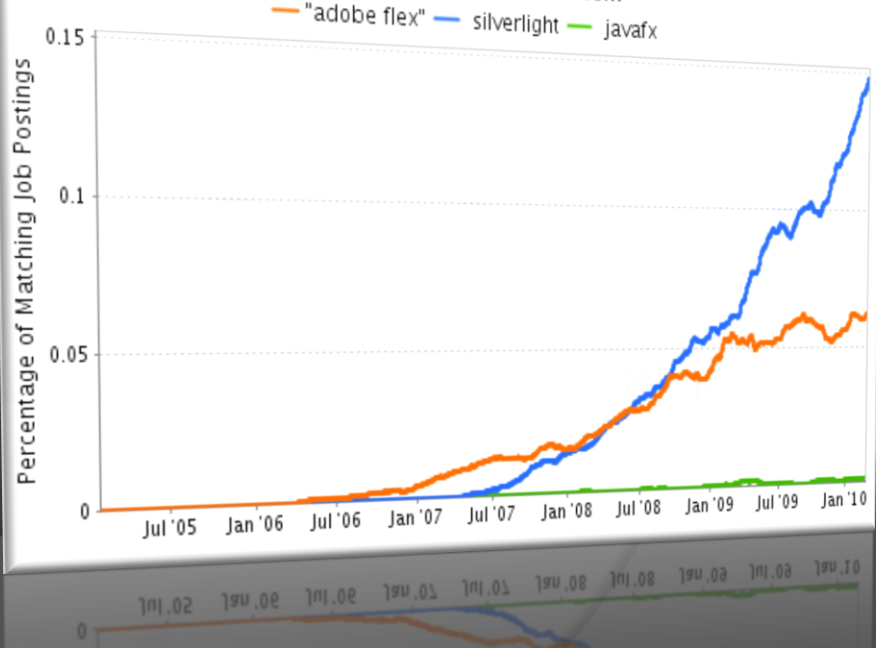
Market Share

Flash 96%
 Java 80%
 Silverlight 47%

adobe flex" ● silverlight ● javafx ● ajax

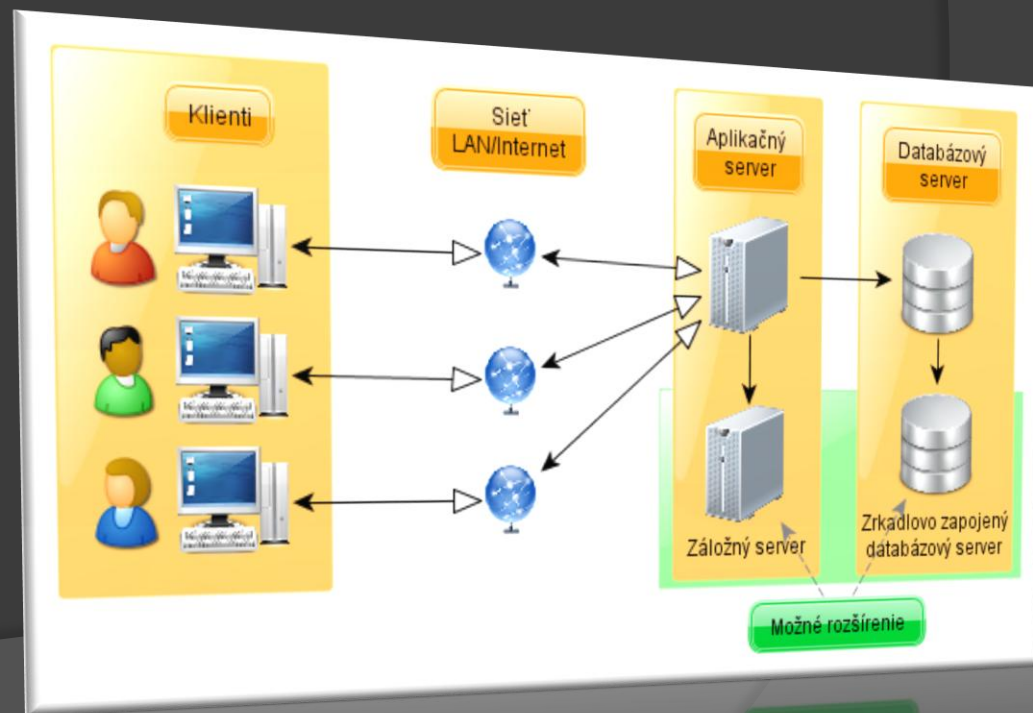
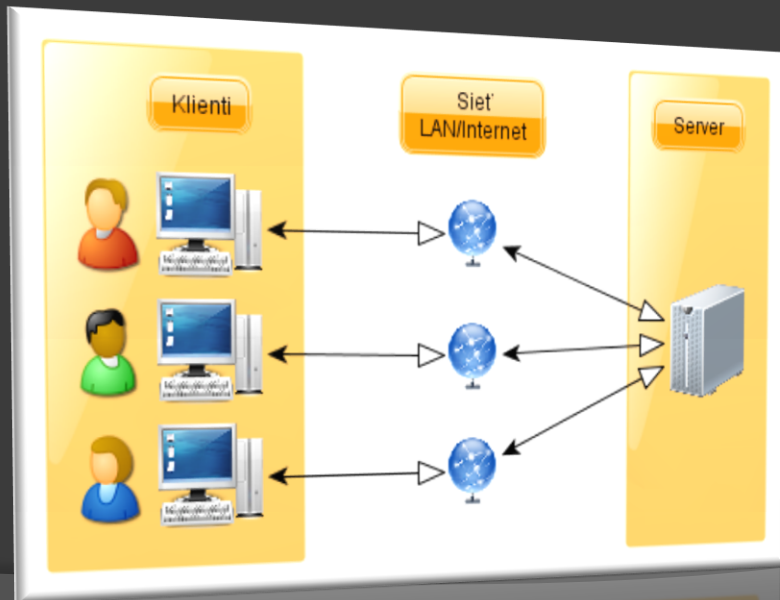


Job Trends from Indeed.com



Architektúra

- Klient/Server, N-vrstvová architektúra
- Kompozitné aplikácie, Modulárna/Komponentová arch.
- Doménový model (ORM na databázu, Vrstva služieb/Fasáda), ~~Transakčný skript~~, ~~Tabuľkový modul~~
- RIA > Tenký klient / ~~Hrubý klient~~
- Model-View-ViewModel

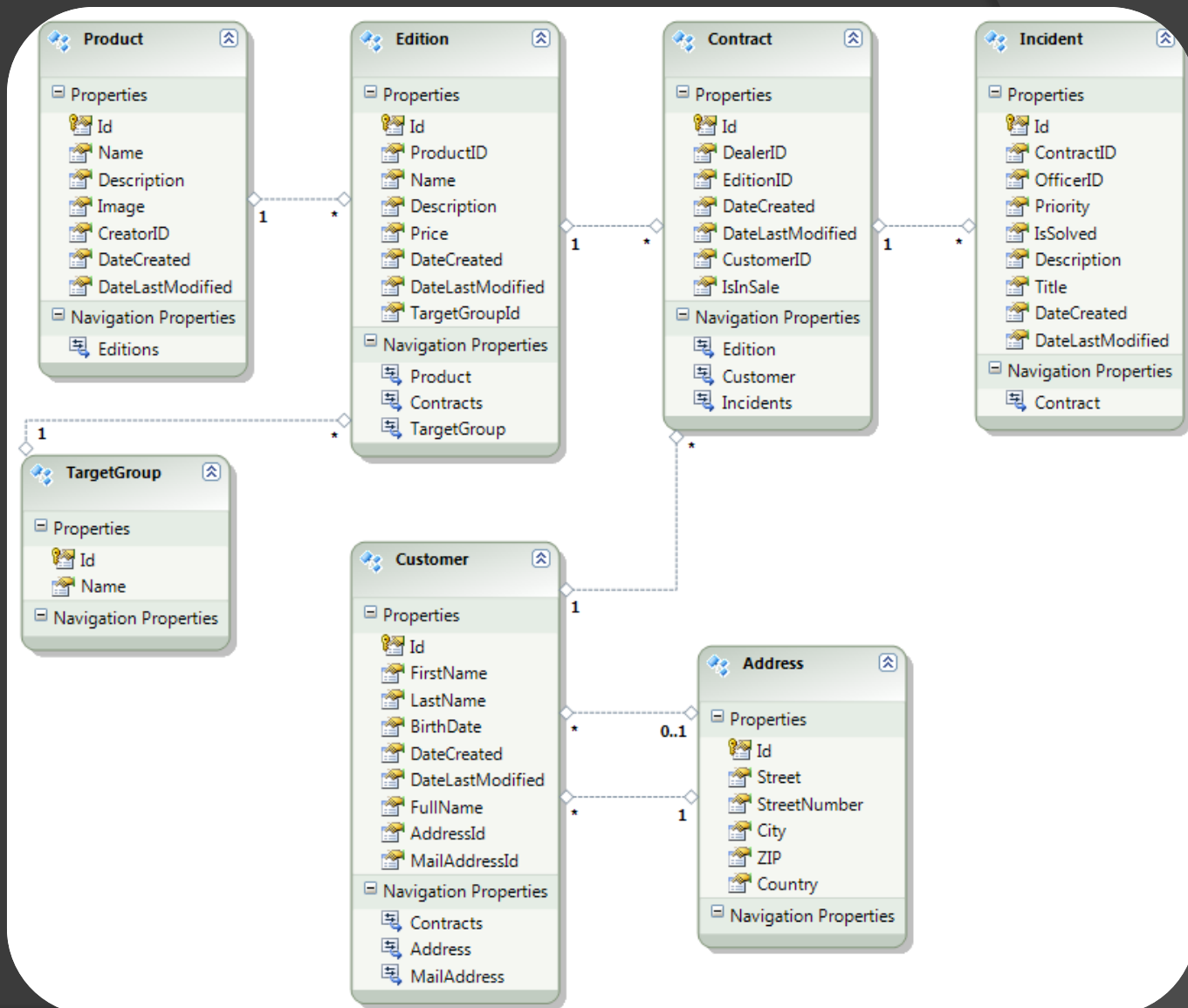


Doménový model

Objektovo orientovaný model problémovej domény.

Entity – samostatné objekty reprezentujúce objekty z domény.

Databáza – ORM alebo využitie objektovej databázy.



Vrstva služieb (Service Layer)

- ⦿ Doménová model a logika na serveri je dostupná pre klienta pomocou **služieb** (REST, SOAP).
- ⦿ Služby tvoria teda **fasádu** nad doménový modelom.
- ⦿ Synchronizácia modelu na serveri a u klienta → **WCF(.NET) RIA Services**.
- ⦿ **LINQ** – možnosť dopytovania sa na prístupové metódy, ktorých návratová hodnota implementuje **Iqueryable**.
- ⦿ Služba na serveri / Kontext u klienta
Každá entita si sleduje zmeny (Tracking changes) a na server sa posiela množina zmien na vykonanie (Change Set).

DomainService

Metadata (valid.)

DomainContext

```

[RequiresAuthentication]
[EnableClientAccess()]
public class ProductsDomainService : LinqToEntitiesDomainService<TelecoSystemsContain
{
    public IQueryable<Product> GetProducts()
    {
        return this.ObjectContext.Products;
    }

    [RequiresRole("Manager")]
    public void InsertProduct(Product product) ...

    [RequiresRole("Manager")]
    public void UpdateProduct(Product currentProduct)
    {
        currentProduct.DateLastModified = DateTime.Now;

        this.ObjectContext.Products.AttachAsModified
    }
}

```

```

[Editable(false)]
public int Id { get; set; }

[Required]
[StringLength(20)]
[Display(Name = "Product name", Description = "The n
public string Name { get; set; }

[Required]
[StringLength(200)]
[Display(Description = "Short description of the pro
public string Description { get; set; }

[Display(AutoGenerateField = false)]
public Guid CreatorID { get; set; }

[Editable(false)]
[Display(Name = "Created", Description = "Date when
public DateTime DateCreated { get; set; }

```

```

var q = _productsCtx.GetProductsQuery();
if (Products.SortDescriptions.Count > 0)
{
    bool isFirst = true;
    foreach (SortDescription sd in Products.SortDescriptions)
    {
        q = OrderBy(q, isFirst, sd.PropertyName, sd.Direction == L
        isFirst = false;
    }
}
else
{
    q = q.OrderBy(p => p.Id);
}

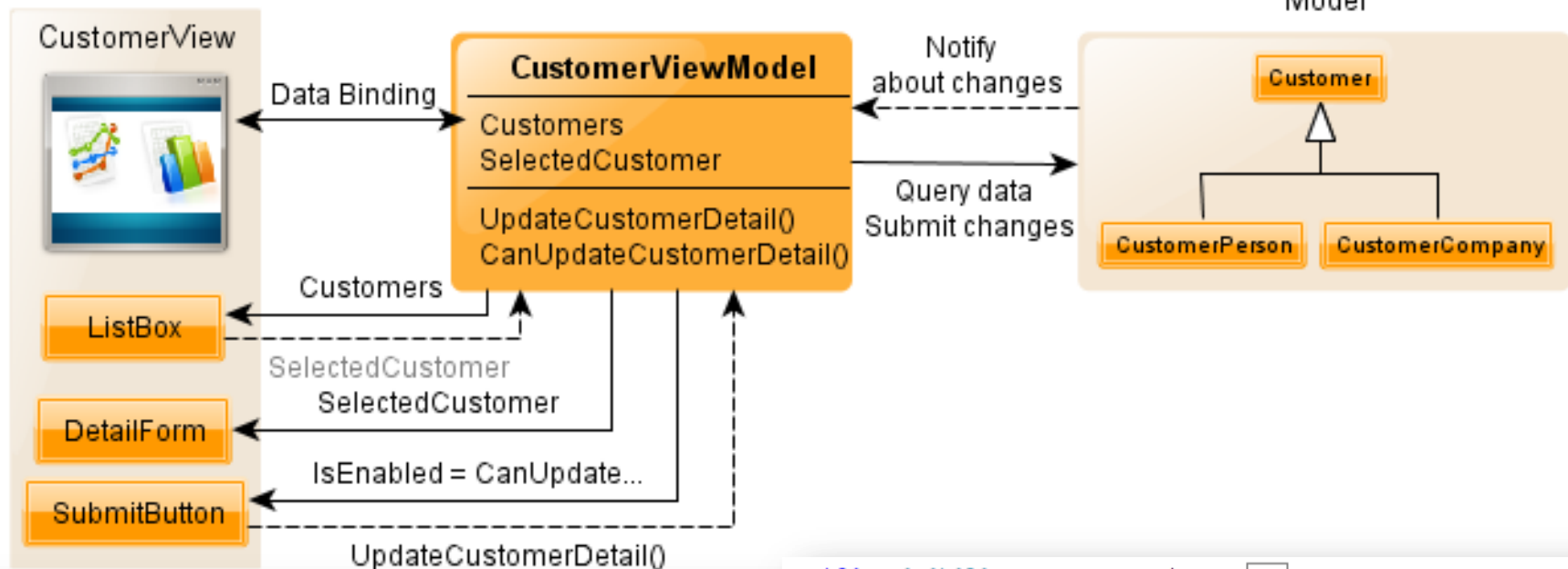
if (PageSize > 0)
{
    q = q.Skip(PageSize * PageIndex);
    q = q.Take(PageSize);
}

_productsCtx.Load(q, ProductsLoaded, null);

```

Model-View-ViewModel

- ◉ Oddelenie prezentácie od modelu.
- ◉ Podobný MVC (starší vzor, dôležitý je controller, prezentačná logika u View).
- ◉ Martin Fowler navrhol Presentation Model.
 - vyžaduje tech. podporu vo forme dátových väzieb medzi View a Prezentačným modelom.
 - PM ponúka View verejné vlastnosti, príkazy (návrhový vzor Command)
- ◉ MVVM je realizáciou PM s aplikáciou vo WPF/Silverlight, ktoré podporujú Data Binding.
- ◉ Mladý, málo popísaný vzor, stále vo vývine, priestor pre môj „výskum“.



```

BusyIndicator" BusyContent="{Binding BusyContent}" IsBusy="{Binding
Width="700" d:LayoutOverrides="Height">
Binding Products}" IsReadOnly="True" SelectedItem="{Binding Product
ntal" Width="700">
Binding PageSize}" HorizontalAlignment="Stretch" Width="650" S
="{Binding PageSize, Mode=TwoWay}" HorizontalAlignment="Right"
Margin="20,0,0,0">
"{Binding Products}" CurrentItem="{Binding ProductDetail, Mode=TwoWay}"
ntal" Margin="0,10,0,0">
OrderBrush="#54CECECE" BorderThickness="1" Padding="3">
Image" Tag="{Binding ProductDetail.Image, Mode=TwoWay}" Source="{Binding

```

```

public Visibility ContextHasChanges[...];

private PagedEntityCollectionView<Product> _products;
public PagedEntityCollectionView<Product> Products[...];

private Product productDetail;
public Product ProductDetail[...];

private EntityCollectionView<Variant> _variants;
public EntityCollectionView<Variant> Variants[...];

private bool _isBusy = false;
public bool IsBusy[...];

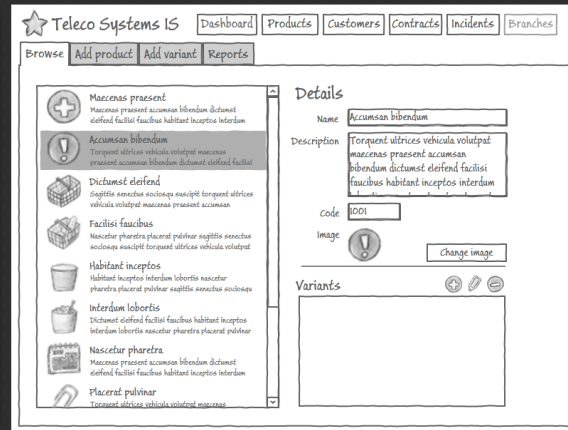
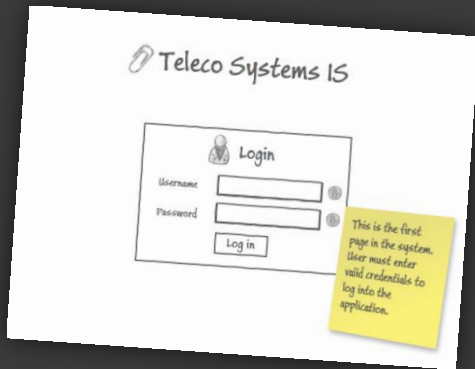
private string _busyContent = "Please wait...";
public string BusyContent[...];
#endregion

#region Commands
public DelegateCommand<object> SubmitChangesCommand { get; set; }
void SubmitChanges(object o)[...]
bool CanSubmitChanges(object o)[...]

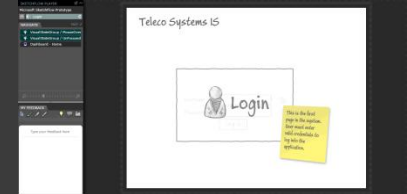
```

Ukážková aplikácia

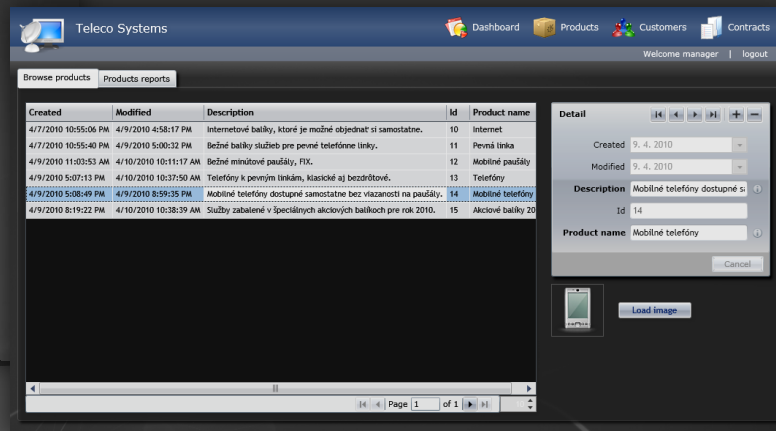
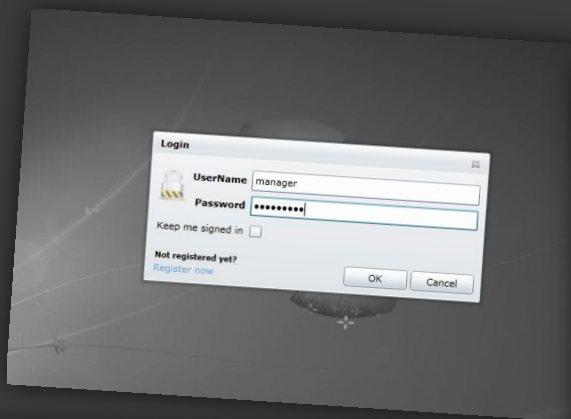
Prototyp UI



Video



Reálny UI



Video



Text bakalárskej práce

- ⦿ V súčasnosti cca 30 strán textu.
- ⦿ Čaká ma
 - rozsiahle okresanie úvodných kapitol
 - prehĺbenie kapitol venujúcich sa vzorom a WCF RIA Services
 - dodatočné úpravy po konzultácii so školiteľom
- ⦿ Text obohatený o diagramy a schémy

Home | Thesis' Diary | Thesis' Mission | About

Search here... Search

Silverlight & WCF RIA Services

Bachelor Thesis About RIA Enterprise Applications (written by Lukáš Zdechovan, led by Ondrej Svačina)

Localization of Silverlight Business Application (sample)

★ Author: Lukáš Zdechovan | Posted: 23-01-2010 | Category: Samples



Although it looks that localization of **Silverlight Business Applications** is not worth of writing a post, there are some really inconspicuous problems which you have to resolve.

If you want to localize the application by creating resource files of strings for a particular language (AppStrings.sk-SK.resx, AppString.de-DE.resx) you should definitely read this post (with sample source-code) to not get into trouble.

Illustration At first I create a New project with Business Application Template.

	Silverlight Business Application	Visual C#
	WCF RIA Services Class Library	Visual C#

The strings resource files are in the template located in both Client and Server (Web) projects.



- LocalizedSilverlightApp
 - Properties
 - References
 - Assets
 - Resources
 - ApplicationStrings.resx
 - SecurityQuestions.resx
 - Styles.xaml
 - Controls
 - Helpers
 - Libs

The most important resource is the ApplicationStrings.resx file, in which almost all the strings from the client side are stored.

Strings which are used in the login and registration form are stored on the server side in RegistrationDataResources.resx.

To localize the strings you have to create copies of desired resource file with an appropriate suffix according to the culture, so slovak copy of AppStrings will be ApplicationStrings.sk-SK.resx. Do the same to all resource files in the project and translate them to your language.

Categories

- Introduction (2)
- Patterns (3)
- Resources (6)
- Samples (3)
- User Experience (1)

Author



Tag Cloud

NET RIA Services about architecture bachelor thesis Blogs Diagrams domain logic domain model domain-driven design Enterprise Applications Entity Framework Eric Evans facebook Google Search API Introduction Jimmy Nilsson Localization martin fowler MIX'09 MVVM Patterns RIA RIA Services webcast

Ďakujem za pozornosť.

Priestor pre vaše otázky.