Návrh Databáze české moderní architektury

FIT ČVUT Praha

ing. arch. Jaromír Srba

2015

Obsah

Úvod

PRVNÍ ITERACE - Analýzy

1. **Analýza business procesů**

   1.1. Analýza business procesů reálného systému

   1.1.1. Popis procesů
   1.2.1. Diagram business procesů

   1.2. Analýza business procesů aplikace

   1.2.1. Popis procesů aplikace
   1.2.2. Diagram business procesů aplikace
   1.2.3. Možné průchody aplikací

2. **Analýza požadavků**

   2.1. Soupis požadavků

3. **Analýza užití**

   3.1. Use Case Model
   3.2. Účastníci systému
   3.3. Popis cílů případů užití
   3.4. Scénáře případů užití
   3.5. Splnění požadavků v případech užití

4. **Analytický - doménový model tří**

   4.1. Popis tříd

DRUHÁ ITERACE - Návrh

5. **Návrh architektury**

6. **Návrhový model tříd**

7. **Relační datový model**

Prameny a literatura

Úvod

Úkol vytvořit formou doktorské práce Databázi české moderní architektury (DCMA) byl zadán v roce 2012 Ústavem teorie a dějin architektury Fakulty architektury ČVUT v Praze. Databáze zahrnuje architekturu v České republice v období od konce 19. století do současnosti a má za úkol vědeckým způsobem zpracovat informace nejen o jednotlivých architektonických dílech, ale také široké spektrum dalších důležitých souvisejících informací.

Jeví se jako vhodné zaměřit se zejména na tři základní skupiny a v nich níže uvedené podskupiny prvků zkoumaného systému:

1) Díla (architektonická i urbanistická, a to díla realizovaná i projekty děl nerealizovaných),
2) Osoby (fyzické i právnické - autoři, architekti, projektanti, investoři, stavebníci, badatelé, ...),
3) Dokumenty o dílech a osobách a jejich zdroje (dokumentace fotografická, obrazová, výkresová, textová, ze všech druhů zdrojů (primárních i sekundárních) - sbírky, archivy, články, časopisy, sborníky, knihy). Doplňkovou skupinou zkoumaného systému jsou také místa (např. městské části, města...), která lze zkoumat samostatně, či je přiřadit k první skupině systému a zkoumat je jako specifický druh děl.

Je nezbytné hledat nejvhodnější řešení databáze v postupných krocích. Jednotlivé plánované kroky byly rozděleny do čtyřech roků studia takto:

1. rok studia - 2012/2013 - Seznámení s problematikou, studium, rešerše, přípravné práce.
2. rok studia - 2013/2014 - Návrhová fáze databáze. Analýzy, formulování požadavků na databázi, modelování procesů, návrh teoretického modelu databáze, prvotní skica databáze.
3. rok studia - 2014/2015 - Realizace konceptu databáze dle hotového teoretického modelu. Naplnění databáze modelovými daty. Zkušební provoz. Sběr materiálu pro obsah databáze.
4. rok studia - 2015/2016 - Příprava výsledného obsahu databáze. Realizace výsledné podoby databáze. Odladění hotové databáze. Vypracování a obhajoba disertační práce.

Vzhledem k velké šíři zkoumané problematiky (velké množství děl i tvůrců) a limitovaným možnostem jediného řešitele se jeví jako vhodné redukovat dle předem stanoveného klíče množství dat zpracovávaných v databázi v rámci doktorské práce. Například zapracovat pouze díla určitých autorů či díla na určitém menším území. Po obhájení doktorské práce formou disertace by se do databáze další data dle potřeby doplnila.

V letním semestru školního roku 2013/2014 navštěvoval řešitel databáze přednášky a cvičení předmětu Softwarové inženýrství I na Fakultě informačních technologií ČVUT Praha. Účelem studia tohoto předmětu pro řešitele databáze je osvojit si správný postup analytických a návrhových prací. Cílem je zpracovat analýzy procesů a návrh databázové aplikace. Samotná realizace aplikace bude provedena v následujícím školním roce 2014/2015.

První iterace

1. Analýza business procesů

1.1. Analýza business procesů reálného systému architektury

1.1.1. Popis procesů

Pro správný návrh aplikace je nezbytné správně identifikovat a následně analyzovat procesy, které se ve zkomaném systému odehrávají a aktivity, které jsou součástí těchto procesů. Zároveň je vhodné identifikovat objekty (skupiny objektů), které se v systému vyskytují. Protože systém architektury je značně složitý, budeme nyní hledat ty nejvýznamnější procesy, aktivity a objekty systému. K nelezení procesů a jejich aktivit a pro nalezení objektů zvolíme slovní popis systému. V tomto popisu identifikujeme jednotlivé hledané prvky jejich zvýrazněním a podtržením takto:
Procesy systému = zelená barva a podtržení čárkovanou čárou
Aktivity procesů systému = modrá barva a podtržení tečkovanou čarou
Objekty systému = žlutá barva a podtržení plnou čarou

Pro nalezení procesů, aktivit a objektů, které se v systému architektury vyskytují jsou určující následující výroky:

Vývoj architektury je reprezentován jak vývojem (tvorbou) architektonických děl tak i vývojem tvůrců těchto děl.

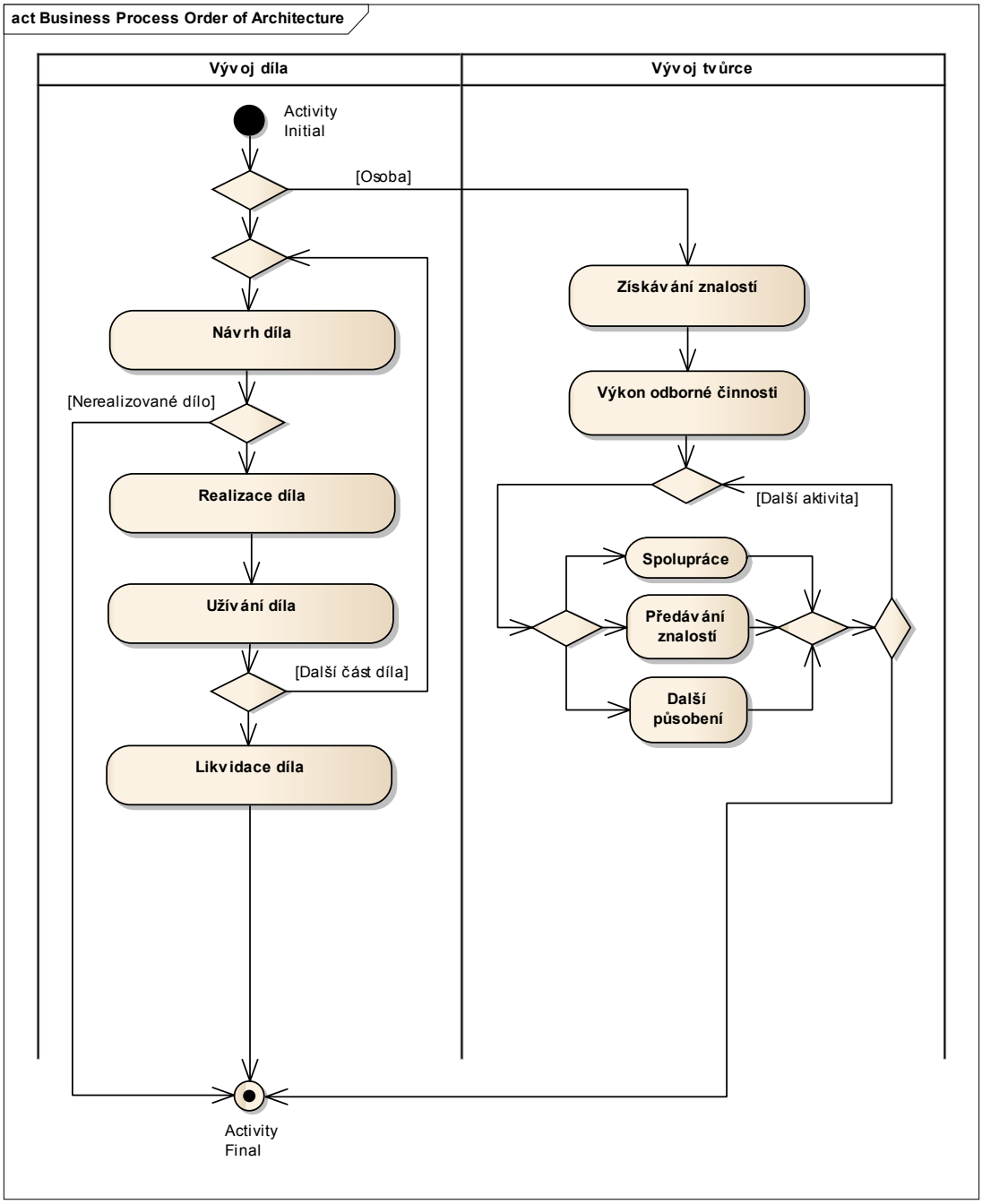
Za **architektonická díla** se považují jak realizované stavby tak i projekty nerealizovaných staveb. Mezi **tvůrce děl** patří osoby fyzické i právnické - architekti, projektanti, investoři i zhotovitelé.

Vývoj architektonického díla prochází následujícími fázemi: **návrh**, **realizace**, **užívání** a **likvidace**. **Vývoj tvůrce** pak může být charakterizován zejména těmito aktivitami: **získávání znalostí**, **výkon odborné činnosti**, **spolupráce s dalšími tvůrci**, **předávání znalostí**, **další působení**.

Informace o architektuře jsou sdělovány prostřednictvím různých dokumentů pojednávajících o dílech a jejich tvůrcích. Tyto **dokumenty** mají různou podobu a formu (textové, obrazové, fotografické, zvukové, audiovizuální...)

Prvky nalezené ve slovním popisu systému nyní využijeme pro modelování systému pomocí diagramů. Procesy a jejich aktivity lze prezentovat pomocí následujícího Diagramu business procesů.

1.1.2. Diagram business procesů



1.2. Analýza business procesů aplikace Databáze české moderní architektury

1.2.1. Popis aplikace a procesů

Účel
Systém bude sloužit pro shromažďování, zpracování a správu dat o české moderní architektuře.

Předmět zájmu (informace o jakých objektech systém shromažďuje data)

Hlavní předmět zájmu
Díla (stavby, projekty) české moderní architektury
Tvůrci (architekti, stavitelé) děl české moderní architektury

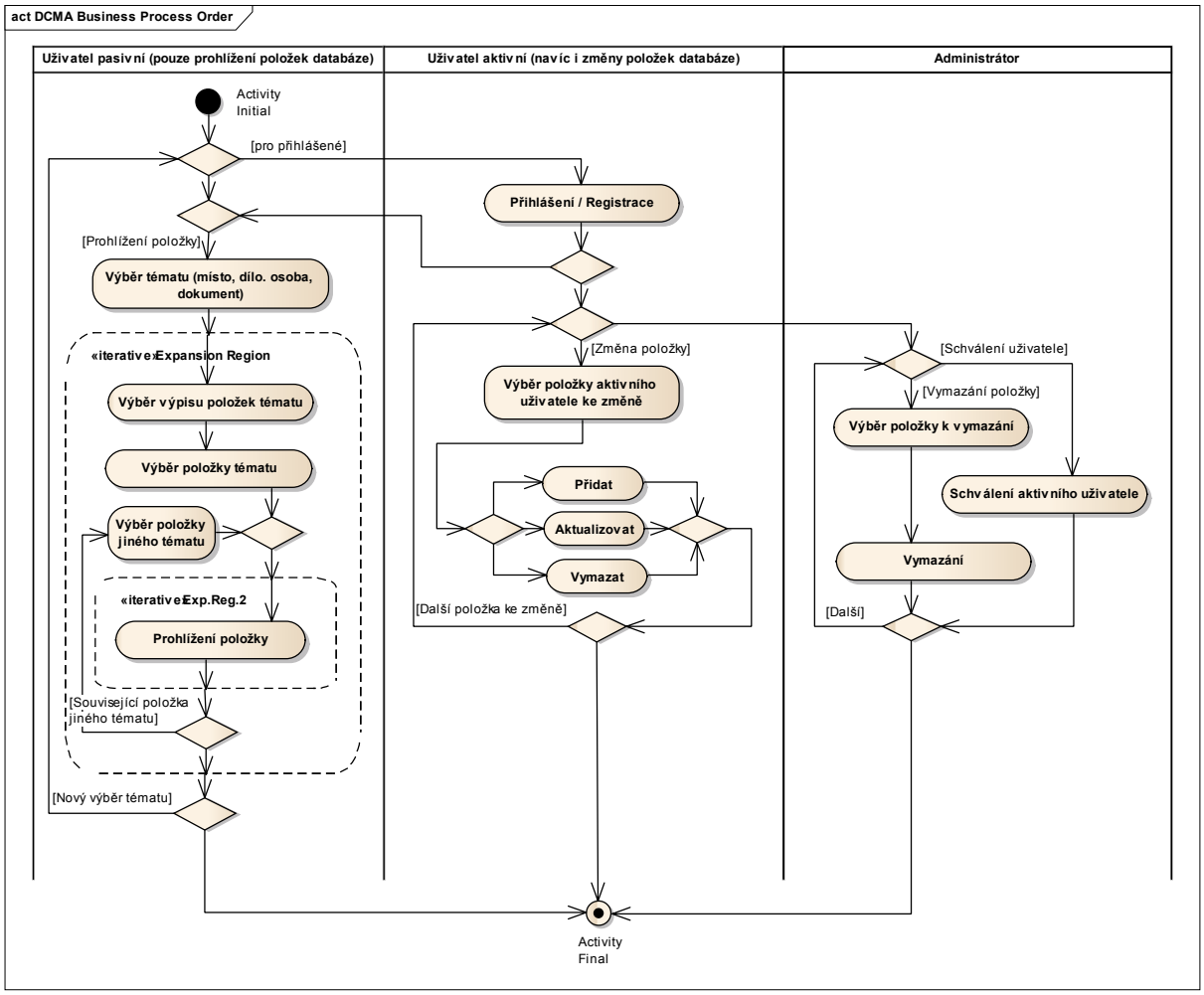
Doplňkový předmět zájmu
Místa (regiony, města, části měst) související s díly a s jejich tvůrci
Dokumenty (články, knihy) pojednávající o dílech a o jejich tvůrcích

Účastníci systému a jejich aktivity
Pasivní uživatel (zájemce o architekturu, student...) - prohlíží v systému uložená data
Aktivní uživatel (badatel, učitel..) - kromě prohlížení dat může do systému vkládat nová data a tato vložená data aktualizovat či mazat
Administrátor - může vymazat kterákoli data uložená v systému, dohlíží na provoz systému

Business procesy v systému
Prohlížení dat o předmětu zájmu - vybírání předmětu zájmu a hledání a prohlížení dostupných položek
Vkládání dat o předmětu zájmu a jejich aktualizace a mazání
Evidence uživatelů

Systém můžou badatelé využívat k ukládání informací o dílech architektury a jejich tvůrcích. Běžní uživatelé mohou systém využít pro získávání informací o moderní architektuře, jejích dílech a tvůrcích.

1.2. Diagram business procesů aplikace



Aktivity procesů jsou v diagramu rozkresleny v podrobnějším členění

1.3. Možné průchody aplikací

Po příchodu na stránky aplikace uživatel, který si chce prohlížet obsah databáze, volí téma svého zájmu (volí zda chce prohlížet místa, díla, osoby nebo dokumenty). Může využít políčko hledat nebo využít menu pro získání výpisu míst (osob, děl či dokumentů) a po zvolení konkrétního místa (osoby, díla či dokumentu) se dostane na konkrétní stránku:

- **stránka místa** s podrobnými informacemi o místu (např. městu nebo části města) a se základními informacemi o dílech, osobách a dokumentech, které s místem souvisejí. Součástí stránky je interaktivní mapa zobrazující umístění děl souvisejících s místem. Ze stránky místa se lze dostat na stránku objektu, osoby či dokumentu, který s místem souvisí.

- **stránka osoby** s podrobnými informacemi o této osobě a se základními informacemi o objektech, osobách a dokumentech souvisejících s osobou, Ze stránky osoby se lze dostat na stránku objektu, osoby či dokumentu, který s osobou souvisí.

- **stránka díla** s podrobnými informacemi o tomto díle a se základními informacemi o osobách, dokumentech a ostatních dílech, které s dílem souvisejí. Ze stránky díla se lze dostat na stránku souvisejícího dokumentu, díla či osoby.

- **stránka dokumentu** obsahuje podrobné informace o tomto dokumentu a základní informace o osobách, dílech a dokumentech s tímto dokumentem souvisejících. Pokud je dokument dostupný online, lze jej z této stránky otevřít a zpřístupnit jej tak uživateli. Ze stránky se lze dostat na stránku každého souvisejícího díla, dokumentu či osoby.

Z každé stránky aplikace se lze dostat na stránku pro přihlášení uživatele a na stránku pro registraci uživatele.

Registrovaný a přihlášený uživatel může (na stránce k tomu určené) přidávat do aplikace další místa (osoby, díla, dokumenty) a tato jím přidaná místa (osoby, díla, dokumenty) aktualizovat či mazat.

Administrátor aplikace může vymazat kterákoli data uložená v systému. Administrátor dohlíží na provoz systému.

2. Analýza požadavků

2.1. Soupis požadavků

Požadavky funkční

F1 - Evidování uživatelů
F2 - Přihlašování uživatelů
F3 - Evidování osob (tvůrců)
F4 - Evidování děl
F5 - Evidování dokumentů
F6 - Evidování míst
F7 - Zobrazování položek (míst, děl, osob, dokumentů)

Požadavky obecné

N1 - Forma - webová aplikace
N2 - Bezpečnost - zálohování dvojnásobné (server + archiv), denně
N3 - Dostupnost - 365 dnů v roce, 7 dnů v týdnu, v případě poruchy zprovoznit systém do 24 hodin

3. Analýza užití

3.1. Use Case Model

3.2. Účastníci systému a jejich aktivity

Pasivní uživatel (zájemce o architekturu, student...)
- prohlíží v systému uložená data
- registrací v systému a následných schválením administrátorem se z pasivního uživatele může stát uživatel aktivní

Aktivní uživatel (badatel, učitel..)
- prohlíží data a navíc může do systému vkládat nová data a tato svá vložená data aktualizovat či mazat

Administrátor
- může vymazat kterákoli data uložená v systému

- dohlíží na provoz systému
- schvaluje nové aktivní uživatele

3.3. Popis cílů případů užití

UC1 - Přidat uživatele
Přidá do systému (zaregistruje) nového uživatele. Registrací v systému a následných schválením administrátorem se z pasivního uživatele, který smí obsah databáze pouze prohlížet může stát uživatel aktivní, který kromě prohlížení, smí obsah databáze také spoluvytvářet.

UC2 - Přihlásit uživatele
Umožňuje registrovanému uživateli přihlášení do systému.

UC3 - Aktualizovat uživatele
Umožňuje registrovanému a přihlášenému uživateli aktualizovat údaje o své osobě uložené v systému

UC4 - Přidat položku
Umožňuje registrovanému a přihlášenému aktivnímu uživateli přidat do databáze nové místo (dílo, osobu nebo dokument)

UC5 - Aktualizovat položku
Umožňuje registrovanému a přihlášenému aktivnímu uživateli aktualizovat údaje o libovolné položce (místu, díle, osobě nebo o dokumentu) uložené jím v databázi.

UC6 - Odstranit položku
Umožňuje registrovanému a přihlášenému aktivnímu uživateli odstranit libovolnou jím dříve vloženou položku (místo, dílo, osobu nebo o dokument) z databáze a odstranit z databáze všechna data o odstraňované položce.

UC7 - Zobrazit položku
Zobrazí podrobné informace uložené v databázi o libovolné položce (o místu, díle, osobě nebo o dokumentu)

UC8 - Odstranit libovolnou položku
Umožňuje přihlášenému administrátorovi databáze odstranit libovolnou položku (místo, dílo, osobu nebo o dokument) z databáze a odstranit z databáze všechna data o odstraňované položce.

UC9 - Schválit nového aktivního uživatele
Umožňuje přihlášenému administrátorovi databáze schválit nově registrovaného uživatele. Schválením se z pasivního uživatele stává uživatel aktivní, který kromě prohlížení obsahu databáze může také do databáze přidávat nové položky a aktualizovat je a může položky, které přidal odstraňovat.

3.4. Scénáře případů užití

UC1 - Přidat uživatele
1.1. Systém zobrazí příslušný formulář
1.2. Uživatel vyplní a odešle formulář
1.3. Systém ověří správnost vyplněných údajů
1.3.a) Pokud systém nalezne chybné údaje požádá uživatele o jejich opravu
1.3.b) Uživatel provede opravu a údaje následně odešle systému, který opět ověří jejich správnost 1.3.
1.4. Systém vloží údaje do databáze, zobrazí oznámení o vložení a odešle uživateli potvrzovací mail

UC2 - Přihlásit uživatele
2.1. Systém zobrazí příslušný formulář
2.2. Uživatel vyplní a odešle formulář
2.3. Systém ověří správnost vyplněných údajů
2.3.a) Pokud systém nalezne chybné údaje, požádá uživatele o jejich opravu
2.3.b) Uživatel provede opravu a údaje následně odešle systému, který opět ověří jejich správnost
2.4. Systém zobrazí oznámení úspěšném přihlášení a přejde ke zobrazení příslušné stránky

UC3 - Aktualizovat uživatele
3.1. Systém zobrazí příslušný formulář s údaji o uživateli
3.2. Uživatel aktualizuje data ve formuláři a poté formulář odešle
3.3. Systém ověří správnost údajů
3.3.a) Pokud systém nalezne chybné údaje, požádá uživatele o jejich opravu
3.3.b) Uživatel provede opravu a údaje znovu odešle systému, který opětovně ověří jejich správnost
3.4. Systém vloží údaje do databáze, zobrazí oznámení o vložení a odešle uživateli potvrzovací mail

UC4 - Přidat položku obsahu databáze (místo, dílo, osobu, dokument)
4.1. Přihlášený uživatel volí, který druh položky chce do systému přidat (zda místo, dílo, osobu, či dokument)
4.2. Systém zobrazí příslušný formulář
4.3. Uživatel vyplní a odešle formulář
4.4. Systém ověří správnost vyplněných údajů
4.4.a) Pokud systém nalezne chybné údaje požádá uživatele o jejich opravu
4.4.b) Uživatel provede opravu a údaje následně odešle systému, který opět ověří jejich správnost
4.5. Systém vloží údaje do databáze, zobrazí oznámení o vložení a odešle uživateli potvrzovací mail

UC5 - Aktualizovat položku
5.1. Systém zobrazí výpis položek vložených dosud uživatelem
5.2. Přihlášený uživatel vybere konkrétní místo (dílo, osobu či dokument), které chce aktualizovat
5.3. Systém zobrazí příslušný formulář s údaji o vybrané položce
5.4. Uživatel aktualizuje data ve formuláři a poté formulář odešle
5.5. Systém ověří správnost údajů
5.5.a) Pokud systém nalezne chybné údaje, požádá uživatele o jejich opravu
5.5.b) Uživatel provede opravu a údaje znovu odešle systému, který opětovně ověří jejich správnost
5.6. Systém vloží údaje do databáze, zobrazí oznámení o vložení a odešle uživateli potvrzovací mail

UC6 - Odstranit položku
6.1. Systém zobrazí výpis položek vložených dosud uživatelem
6.2. Přihlášený uživatel vybere konkrétní místo (dílo, osobu či dokument), které chce vymazat
6.3. Systém zobrazí žádost o opětovné potvrzení výmazu
6.3.a) Pokud si uživatel nepřeje pokračovat ve výmazu, kliknutím přejde na počáteční zobrazení 6.1.
6.4. Uživatel opětovně potvrdí výmaz
6.5. Systém vymaže data položky ze všech tabulek databáze, zobrazí oznámení o výmazu a odešle uživateli potvrzovací mail

UC7 - Zobrazit položku
7.1. Uživatel volí téma, které chce zobrazit (zda místo, dílo, osobu, či dokument)
7.2. Systém zobrazí výpis položek zvoleného tématu
7.3. Uživatel zvolí konkrétní položku tématu ke zobrazení

7.4. Systém zobrazí stránku s podrobnými informacemi o zvolené položce a s výpisem položek (míst, děl, osob či dokumentů), souvisejících se zvolenou položkou, Ze stránky zvolené položky se lze dostat na stránku položky (místa, objektu, osoby či dokumentu), která se zvolenou položkou souvisí.

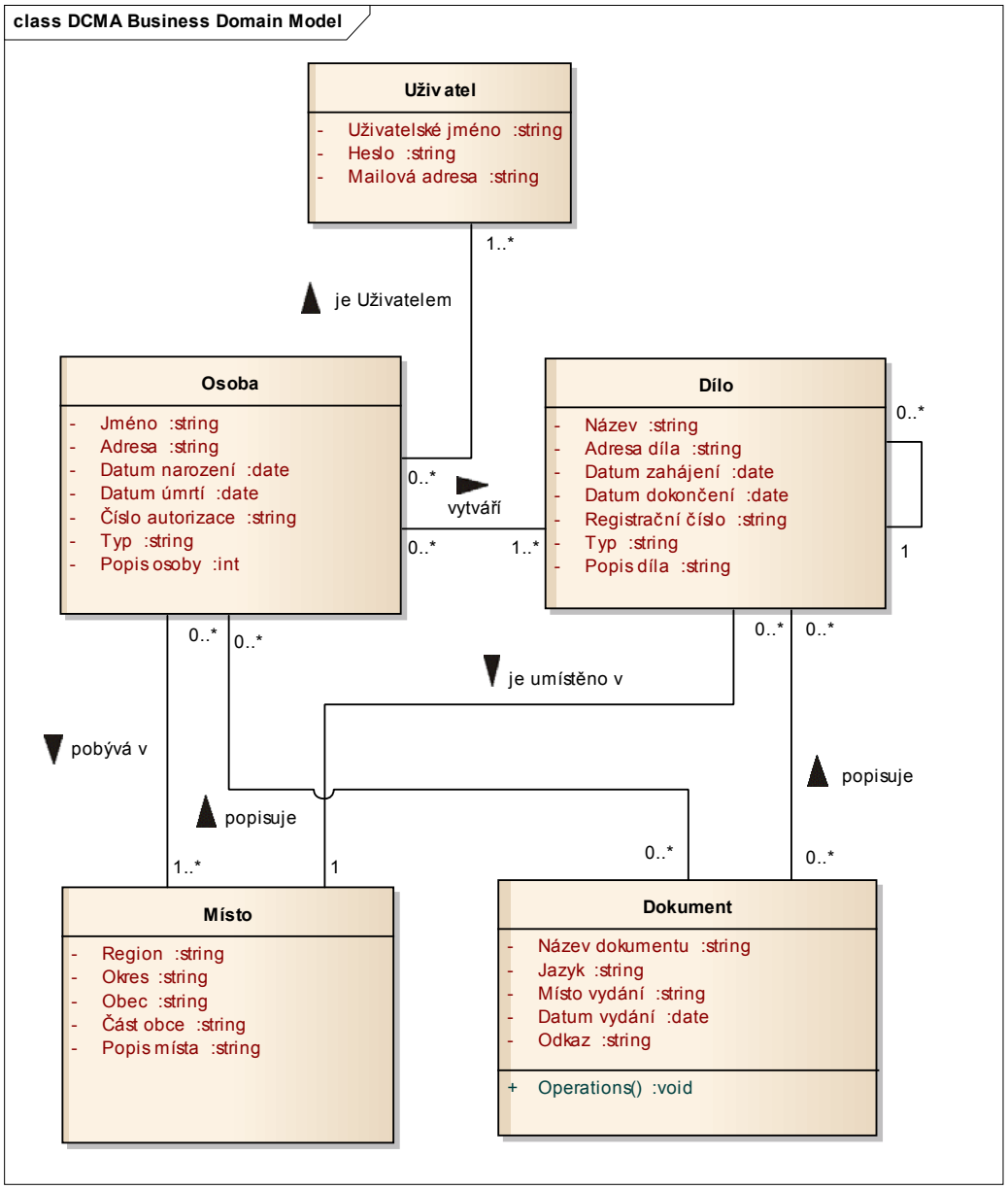
UC8 - Odstranit libovolnou položku (místo, dílo, osobu, dokument)
8.1. Administrátor vybere konkrétní položku (místo, dílo, osobu či dokument), kterou chce vymazat
8.2. Systém zobrazí žádost o opětovné potvrzení výmazu
8.3.a) Pokud si administrátor nepřeje pokračovat ve výmazu kliknutím přejde na počáteční zobr. 8.1.
8.4. Administrátor opětovně potvrdí výmaz
8.5. Systém vymaže data položky ze všech tabulek databáze, zobrazí oznámení o výmazu a odešle administrátorovi potvrzovací mail

UC9 - Schválit nového aktivního uživatele
9.1. Systém zobrazí výpis nových aktivních uživatelů čekajících na schválení administrátorem
9.2. Administrátor vybere ve výpisu aktivního uživatele, kterého chce schválit
9.3. Systém zobrazí žádost o opětovné potvrzení schválení
9.3.a) Pokud si administrátor nepřeje pokračovat ve schválení, kliknutím přejde na počátek 9.1.
9.4. Administrátor opětovně potvrdí schválení
9.5. Systém vloží údaj o schválení do databáze, zobrazí oznámení o schválení a odešle administrátorovi potvrzovací mail

3.5. Tabulka splnění požadavků v případech užití

	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8	UC9
F1	+		+						+
F2		+							
F3				+	+	+			
F4				+	+	+			
F5				+	+	+			
F6				+	+	+		+	
F7							+		

4. Doménový model tříd



4.1. Popis tříd

Uživatel
Data o registrovaných (aktivních) uživatelích

Osoba (tvůrce)
Data o tvůrcích evidovaných děl (osoby fyzické i právnické - autoři, architekti, projektanti, investoři, stavitelé...)

Dílo
Data o dílech (díla architektonická i urbanistická a to jak díla realizovaná, tak i projekty nerealizovaných děl)

Dokument
Data o dokumentech souvisejících s evidovanými díly a jejich tvůrci (články, knihy, dokumentace obrazová, fotografická, výkresová...).

Místo
Data o místech souvisejících s evidovanými díly a jejich tvůrci (regiony, okresy, města, části měst)

Druhá iterace

5. Návrh architektury

Volba technologie

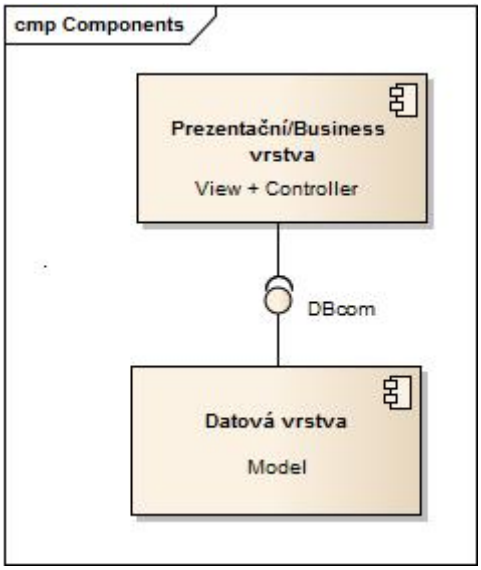
Projekt Databáze české moderní architektury bude realizován jako webová aplikace a implementován v jazyce PHP s využitím databázového systému MySQL. Skriptovací jazyk PHP je určený především k vytváření dynamických webových stránek. Tyto dynamické stránky jsou generovány na straně serveru - k uživateli je přenášen až výsledek interpretace kódu. Velkou výhodou je také fakt, že jazyk PHP není zavislý na platformě. PHP podporuje mnoho knihoven pro různé účely - např. zpracování textu, grafiky, práci se soubory, přístup k většině databázových systémů a podobně.

Volba a popis architektury

Aplikace je nyní navžena tak, aby využívala dvouvrstvou architekturu (MVC). Jedná se o softwarovou architekturu (jednodušší vzor MVC - Model 1), která rozděluje datový model aplikace, uživatelské rozhraní a řídicí logiku do dvou nezávislých komponent tak, že modifikace jedné z nich má minimální vliv na ostatní. Tato architektura také usnadňuje týmovou spolupráci, kdy každý její člen implementuje svoji část aplikace. Toto řešení poskytuje možnost v případě potřeby v budoucnosti oddělením business vrstvy od prezentační vrstvy přejít na na třívrstvou architekturu.
Jednotlivé softwarové třídy aplikace jsou rozděleny do následujících čtyř balíčků: Core, View, Controller, Model.

Prezentační vrstva (View & Controller)
Prezentační vrstva obsahuje třídy a komponenty, za jejíž pomoci dochází ke zobrazování adekvátních informací uživateli a zároveň od uživatele přebírá jeho požadavky a odstraní závislost na použité technologii.

Datová vrstva (Model)
Datová vrstva se stará o práci s daty v databázi. Přistupuje k nim a zpracovává je na základě vstupních požadavků (editace, uložení, výpis…).



Návrh GUI

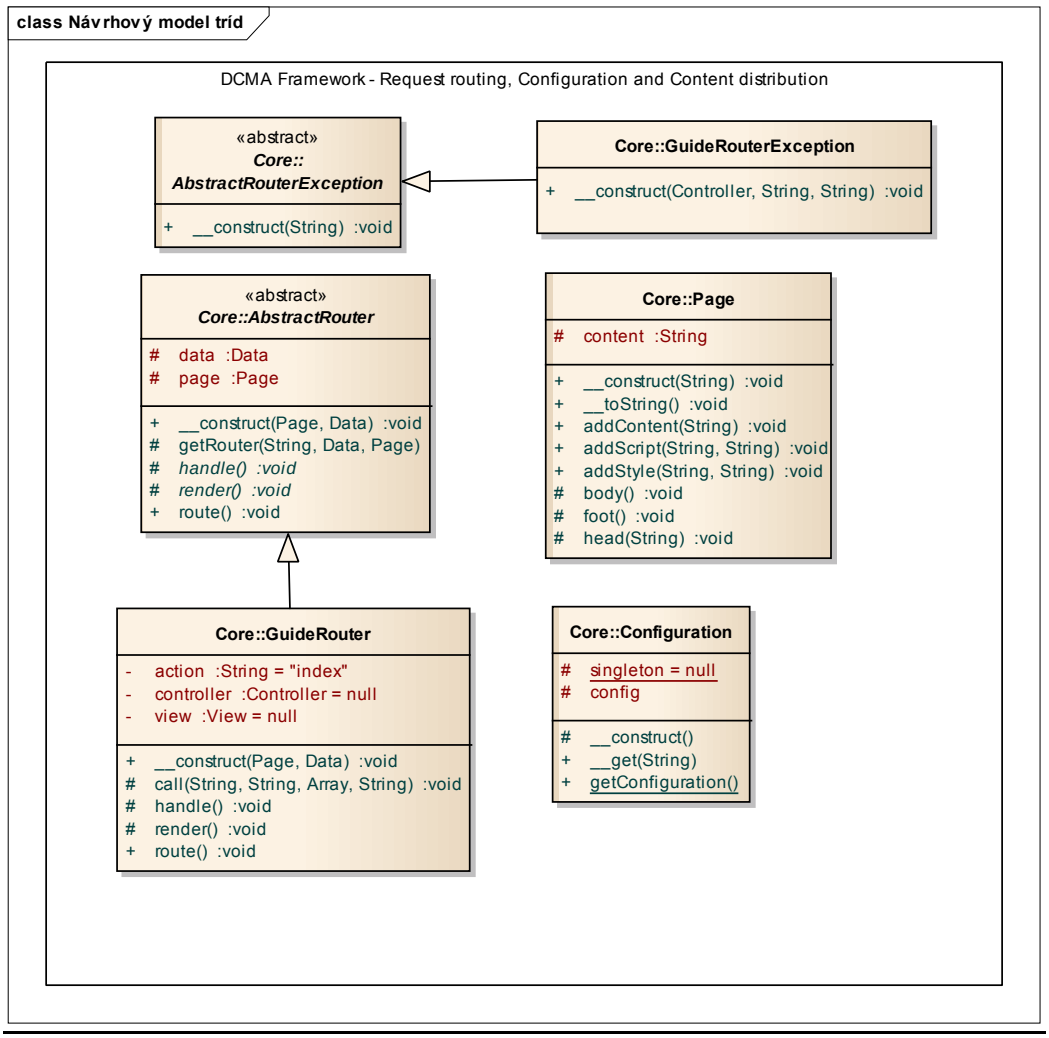
Grafické rozhraní bude navrženo z důrazem na jednoduchou orienataci a snadnou navigaci v celé aplikaci. Po straně hlavního pole stránky je sidebar, který obsahuje hlavní navigaci rozdělenou do jednotlivých záložek (místa, míla, msoby, mokumenty). Navigace bude doplněna kontextovým vyhledáváním.

Úvodní stránka
obsahuje poslední přidané položky. Z úvodní stránky uživatel volí téma svého zájmu (volí zda chce prohlížet místa, díla, osoby nebo dokumenty). Může využít políčko hledat nebo využít menu pro získání výpisu míst (oosob, děl či dokumentů).

Detailní stránka
Po zvolení konkrétního místa (osoby, díla či dokumentu) se uživatel dostane na stránku s detailními informacemi o místě (dílu, osobě nebo dokumentu).

6. Návrhový model tříd

6.1. Balíček "Core"



Třída “AbstractRouter”
Abstraktní třída, reprezentující router.

Název atributu	Datový typ	Popis
data	Data	datová výměna MVC
page	Page	sestavení web. kódu

Metoda	Návratový typ	Popis
getRouter(String, Data, Page)	AbstractRouter	získání routeru
handle()	void	zpracování požadavku
route()	void	směrování k cíli
render()	void	vykreslení

Třída "GuideRouter"

Název atributu	Datový typ	Popis
Action	String	akce MVC
Controller	Controller	použitý controller
View	View	použitý view

Metoda	Návratový typ	Popis
call(String, String, Array, String)	void	získání routeru
handle()	void	zpracování požadavku
route()	void	směrování k cíli
render()	void	vykreslení

Třída "Page"

Název atributu	Datový typ	Popis
Data	Data	datový registr
Page	Page	struktura stránky

Metoda	Návratový typ	Popis
addContent(String)	void	přidá obsah
addScript(String, String)	void	přidá script do obsahu
addStyle(String, String)	void	přidá stylopis do obsahu
body()	void	vykreslí tělo stránky
foot()	void	vykreslí patičku stránky
head(String)	void	vykreslí hlavičku stránky
__toString()	String	vypíše obsah celé stránky

Třída “Configuration”

Název atributu	**Datový typ**	**Popis**
config		
singleton	Configuration	struktura stránky

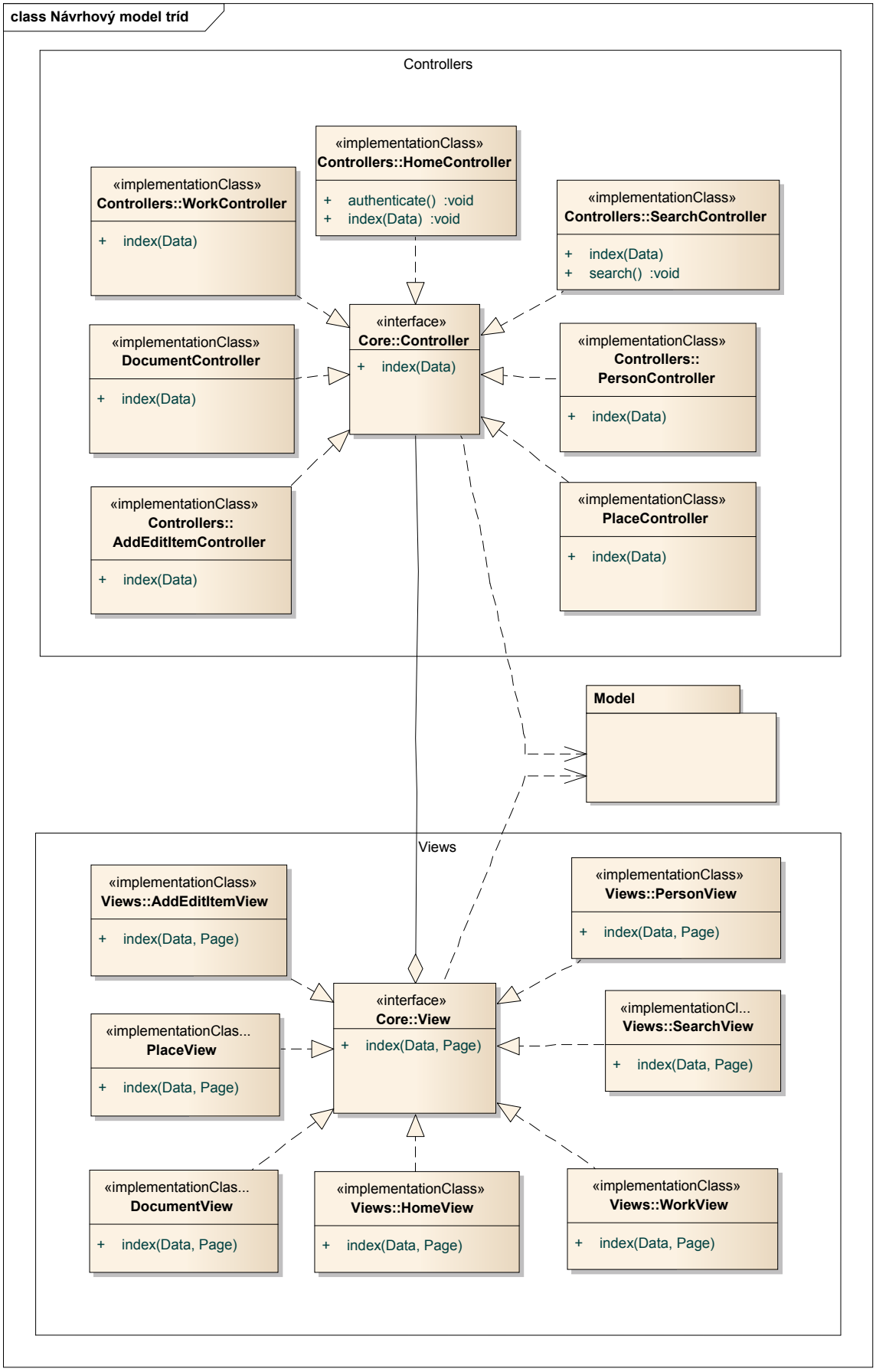
Metoda	**Návratový typ**	**Popis**
getConfiguration()	Configuration	vrací singleton instanci
__get(String)	String	vrací hodnotu konfigurace

<<interface>> “View”

Metoda	**Návratový typ**	**Popis**
index(Data, Page)	void	povinná metoda pro iml. view

<<interface>> “Controller”

Metoda	**Návratový typ**	**Popis**
index(Data)	void	povinná metoda pro iml. cont.



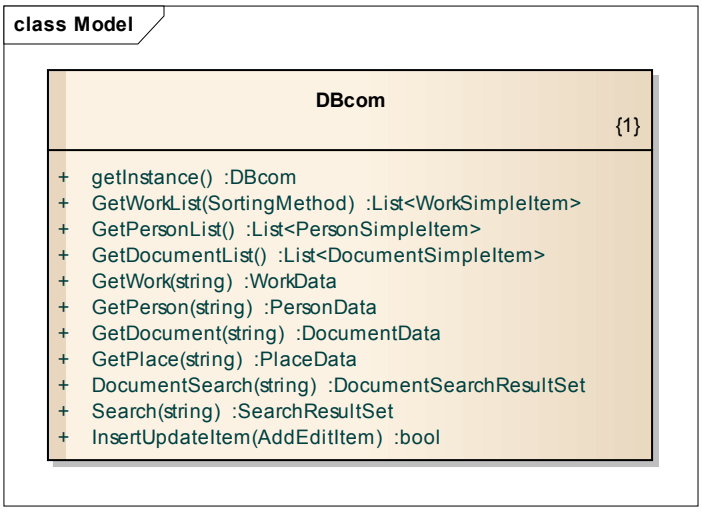
Model architektury MCV = balíčky Model, View a Controller

6.2. Balíček "Model"

Balíček obsahuje základní třídu DBcom (Database component), která aplikaci umožňuje pracovat s daty uloženými v databázi a obsahuje také třídy transformující data z relační vrstvy do objektů se kterými pracuje apikace.

Třída "DBcom"

Základní třída Modelu, jejíž metody jsou volány z Business/Prezentační vrstvy. Jedná se o Singleton třídu. Stará se o komunikaci s databází a o “transformaci” dat z relační vrstvy do objektů, se kterými aplikace pracuje.



Metoda "GetWorkList(SortingMethod)" : Návratový typ "bool" (úspěch/neúspěch)
Metoda sloužící pro získání seřaditelného seznamu děl.

GetPersonList() : List<PersonSimpleItem>
Metoda sloužící pro získání seznamu osob.

GetDocumentList() : List<DocumentSimpleItem>
Metoda sloužící pro získání seznamu dokumentů.

GetWork(string) : WorkData
Metoda umožňující získat v objektu WorkData informace o díle včetně soupisů souvisejících děl, osob a dokumentů.

GetPerson(string) : PersonData
Metoda umožňující získat v objektu PersonData informace o osobě včetně soupisů souvisejících osob, děl, a dokumentů.

GetDocument(string) : DocumentData
Metoda umožňující získat v objektu DocumentData informace o dokumentu, případně včetně soupisů souvisejících dokumentů.

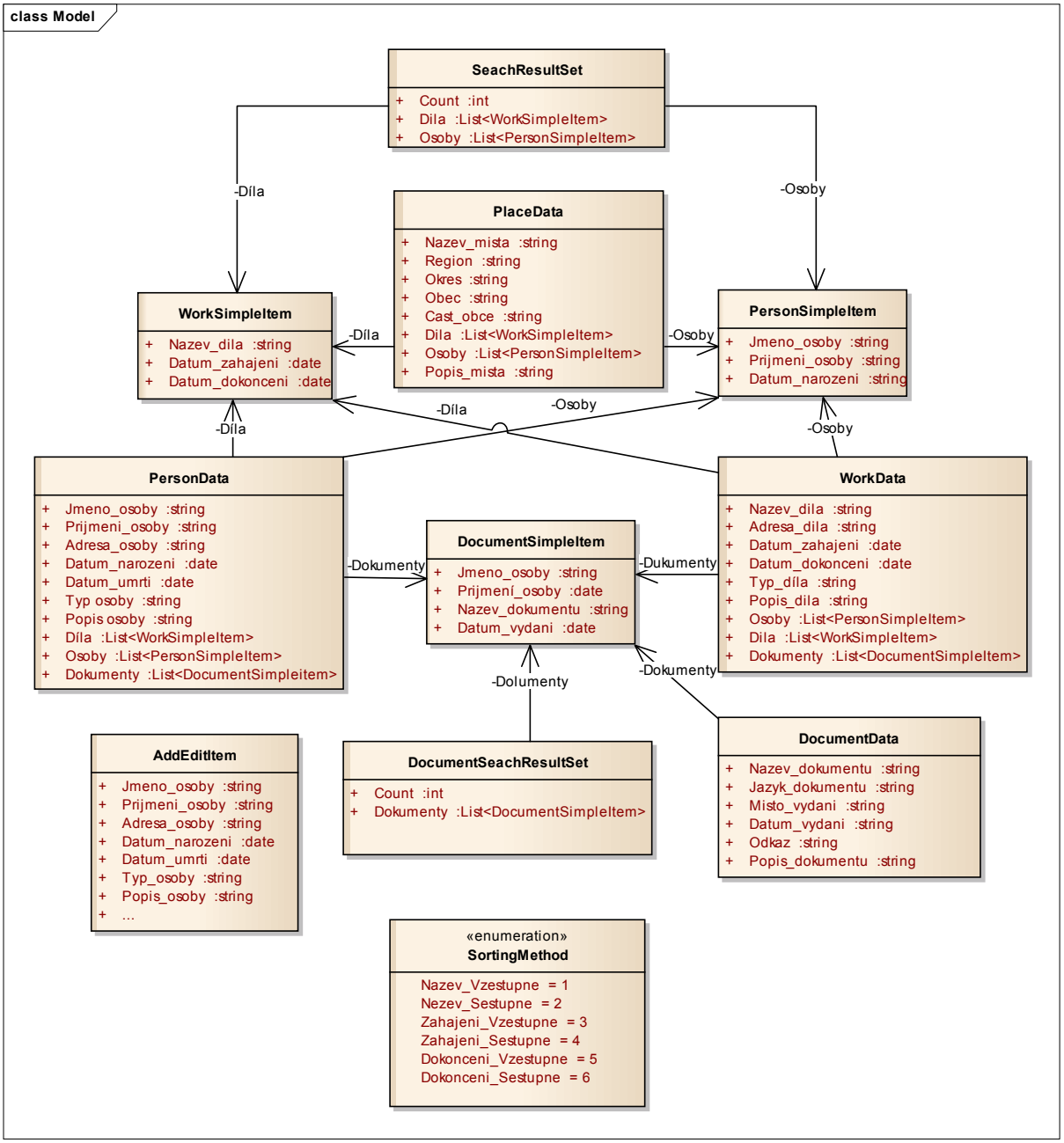
GetPlace(string) : PlaceData
Metoda umožňující získat v objektu PlaceData informace o místu včetně soupisů souvisejících děl a osob.

DocumentSearch(string) : DucumentSearchResultSet
Metoda sloužící pro vyhledání dokumentu dle zadaného výrazu.

Search(string) : SearchResultSet
Metoda sloužící pro získání výsledků hledání dle zadaného výrazu.

InsertUpdateItem(AddEditItem) : bool (úspěch/neúspěch)
Metoda InsertEditItel slouží pro přidání/editaci záznamu v databázi. Jako vstupní parametr přijímá objekt AddEditItem reprezentující vkládaný/editovaný záznam (místo, dílo, osoba, dokument).

Ostatní třídy balíčku "Model"



Třída "WorkSimpleItem"
Reprezentuje jednu položku ve výpisu děl WorkList.

Třída "DocumentSimpleItem"
Jedna položka ve výpisu dokumentů DocumentList.

Třída "PersonSimpleItem"
Jedna položka ve výpisu osob PersonList.

Třída "PersonData"
Objekt obsahující data potřebná pro zobrazení stránky osoby. Kromě dat o osobě obsahuje také výpisy základních informací o dílech, dokumentech a osobách souvisejících se zobrazovanou osobou.

Třída "DocumentData"
Objekt obsahující data potřebná pro zobrazení stránky dokumentu.

Třída "WorkData"
Objekt obsahující data potřebná pro zobrazení stránky díla. Kromě dat o díle obsahuje také výpisy základních informací o osobách, dokumentech a dílech souvisejících se zobrazovaným dílem.

Třída "PlaceData"
Objekt obsahující data potřebná pro zobrazení stránky místa. Kromě dat o místu obsahuje také výpisy základních informací o dílech a osobách souvisejících se zobrazovaným dílem.

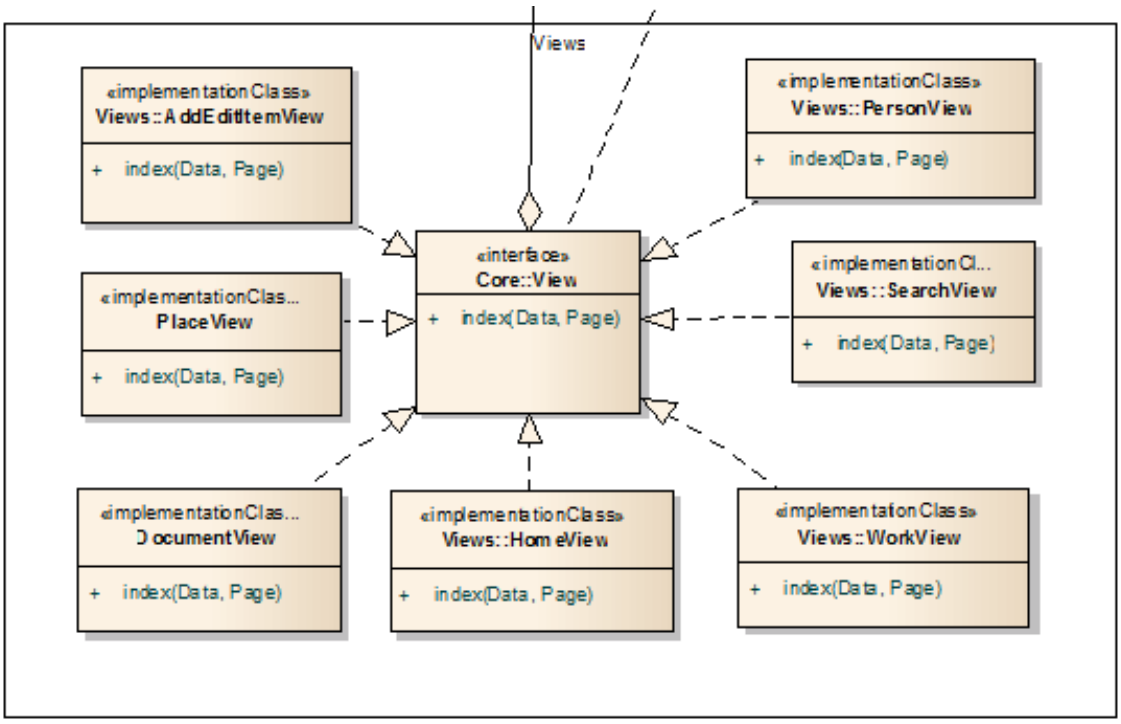
Třída "SearchResultSet"
Objekt obsahující výsledky hledání - výpis nalezených osob a děl.

Třída "DocumentResultSearchSet"
Objekt obsahující výsledky hledání dokumentů

Třída "AddEditItem"
Objekt obsahující data jednotlivého objektu (místo, dílo, osoba, dokument) vkládaná do databáze nebo aktualizovaná v databázi.

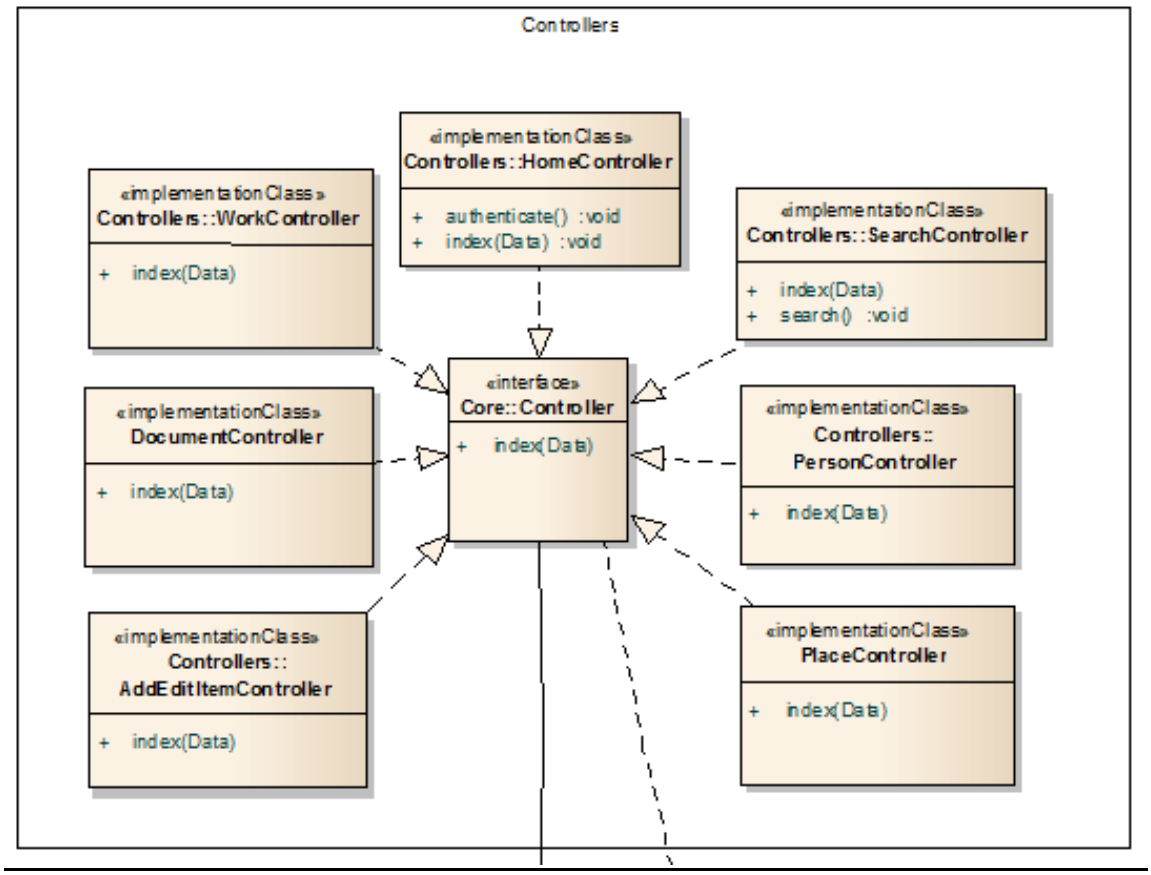
Třída "SortingMethod"
Obsahuje atributy pro možnost řazení výsledků metody GetWorkList.

Balíček "View"



Balíček "View" obsahuje následující třídy:
Třída "HomeView"
Třída "PlaceView"
Třída "WorkView"
Třída "PersonView"
Třída "DocumentView"
Třída "AddItemView"
Třída "SearchView"
Třídy obsahují metody potřebné pro zobrazení adekvátních informací uživateli.

Balíček "Controller"



Balíček "Controller" obsahuje následující třídy:
Třída "HomeController" -
Třída "PlaceController"
Třída "WorkController"
Třída "PersonController"
Třída "DocumentController"
Třída "AddItemController"
Třída "SearchController"
Třídy obsahují metody potřebné pro komunikaci mezi View a Modelem.

7. Model komunikace

V této části je popsán postup zavádění samotného MVC a několik dalších komunikací mezi objekty, které jsou detailněji popsané dále.

Zavádění aplikace
Klient (prohlížeč na straně uživatele) dostane od serveru soubor index.php (FrontController), zavolá se bootstrap.php, který obsluhuje konfiguraci, zároveň proběhne autoload všech tříd a zvolí se router. Router deleguje všechny požadavky na příslušné kontrolery.

Diagram Načteni Navigace
Tento diagram popisuje načtení hlavní navigace, tedy seznamu míst, děl, osob a dokumentů. Načtení probíhá tak, že se zavolají metody getPlaceList getWorkList, getPersonList a getDocumentList (tyto metody jsou pro jednoduchost nahrazeny v diagramu metodou getNavigation) třídy DBCom. Třída DBCom vrátí objekt, obsahující požadovaná data.

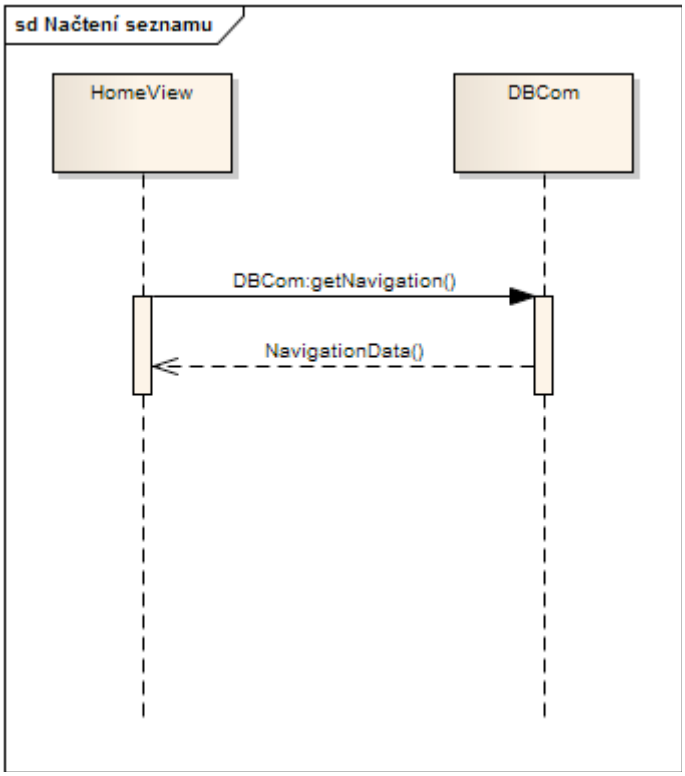


Diagram zobrazení stránky
Diakram znázorňuje zobrazení stránky osoby, tedy informaci o zvolené osobě spolu s výpisem základních informací o souvisejících osobách, dílech a dokumentech. Volá se metoda GetPerson třídy DBcom, vrací se objekt s daty osoby PersonData

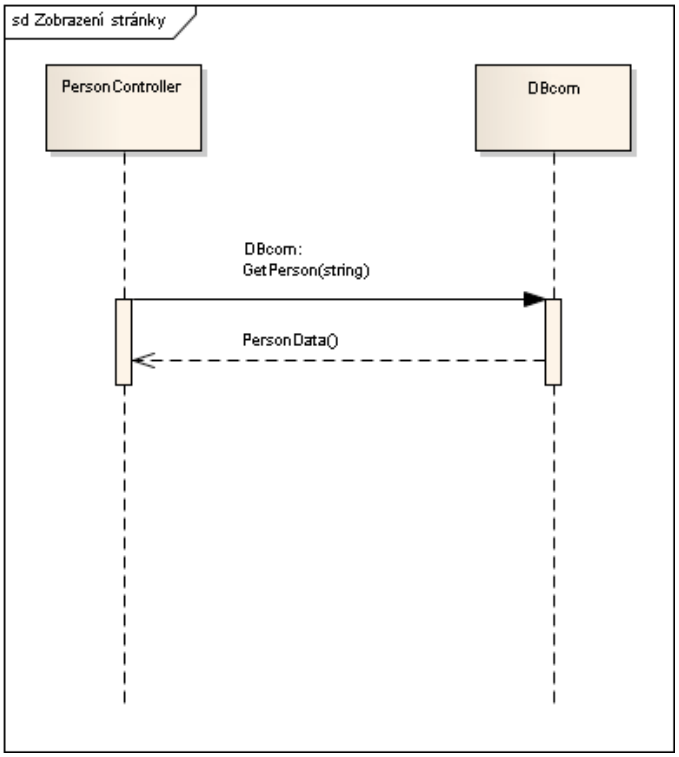
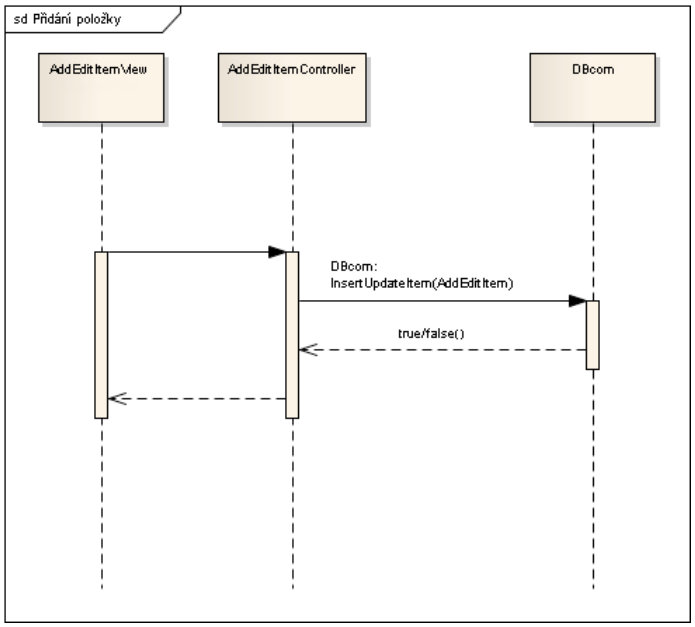
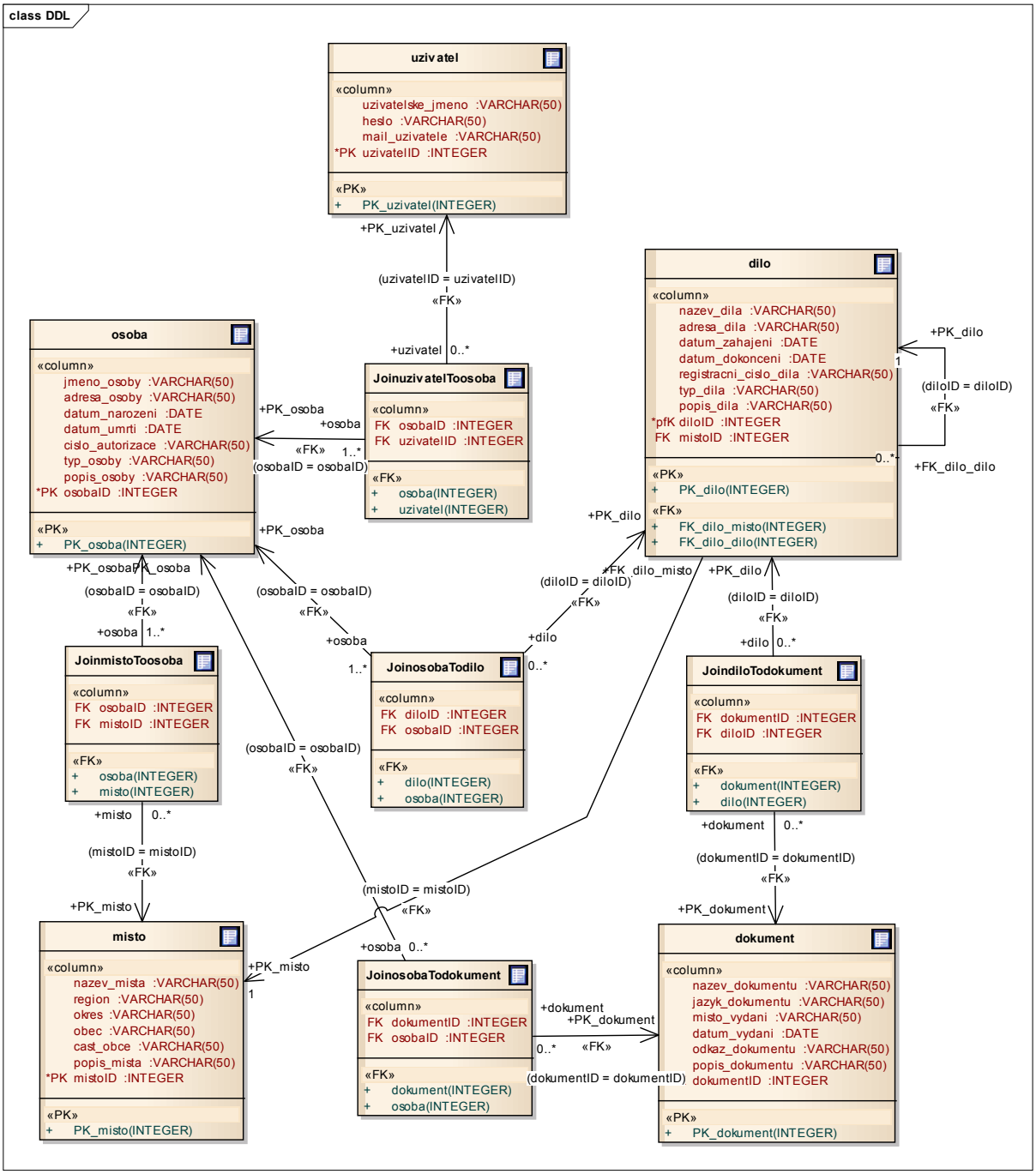


Diagram přidání položky (místo, dílo, osoba, dokument) do databáze
Položka se přidává z příslušné stránky AddEditItemView. Zavolá se abstraktní metoda třídy AddEditItemController, která pošle objekt třídě DBcom a ta data uloží do relační databáze. Vrací se true/false dle úspěchu.



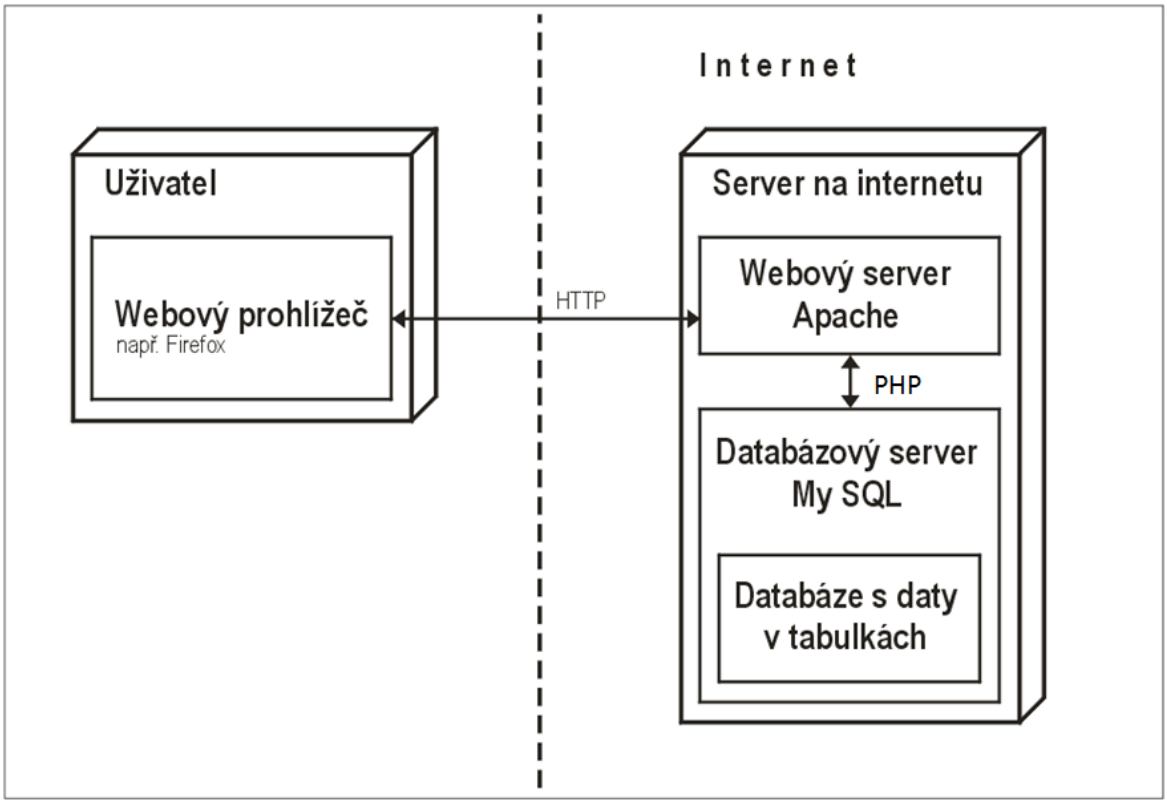
8. Relační datový model



9. Model nasazení

Model nasazení popisuje způsob nasazení aplikace a umístění jednotlivých částí na fyzická zařízení. Aplikace DCMA je tvořena jako webová aplikace, takže bude přístupná z jakéhokoliv počítače s webovým prohlížečem a připojením na internet. Spolu s touto aplikací může být distribuován script pro

vytvoření MySQL databáze, se kterou aplikace pracuje. Aplikace je nainstalována na internetovém serveru s běžícím webovým serverem (například Apache) .Pro běh samotné aplikace je potřeba mít na webovém serveru nainstalovanou technologii PHP 5 (a novější) a databázový systém MySQL.



Prameny a literatura

Mlejnek, Jiří. Softwarové inženýrství I - přednášky a cvičení. Praha: FIT ČVUT, 2014.
Fowler, Martin. Destilované UML. Praha: Grada, 2009.
Achten, Henri. Inženýrská informatika - přednášky a cvičení. Praha: FA ČVUT, 2013.