VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

# Informační systém pro vyhodnocování aktivit pracovníků ve stromové hierarchii

Semestrální projekt

**2006** **Jaroslav Burget**

# Informační systém pro vyhodnocování aktivit pracovníků ve stromové hierarchii

© Jaroslav Burget, 2005.

*Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů..*

## Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Jaroslava Rába. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Jaroslav Burget
29.12.2006

# Abstrakt

Tématem tohoto semestrálního projektu je Informační systém pro vyhodnocování aktivit pracovníků ve stromové hierarchii. V první části obsahuje seznámení s problematikou řízení zaměstnanců ve stromové hierarchii ve finančním sektoru a ve druhé konkrétní případ modelu systému v jazyce UML pro prostředí databáze postrelačního typu - Caché.

.



# Klíčová slova

Informační systém, Caché, stromová hierarchie, UML,



# Poděkování

Děkuji svému vedoucímu Ing. Jaroslavu Rábovi za vedení mého projektu a poskytování  užitečných rad.

# Abstract

The topic of this semestral project is Information system for plotting activity of staff in the tree hierarchy. The first part includes preacquaintance with problems of direction of staff in the tree hierarchy in the financial sector and the second part includes the concrete case of model of the system in UML language for postrelation database environment – Caché.



# Keywords

Information system, Caché, tree hierarchy, UML

# Obsah

1 Úvod ... 6
2 Řízení zaměstnanců ve stromové hierarchii ... 7
2.1 Síťový marketing versus „pyramidové struktury“ ... 7
2.2 Stromové hierarchie ve finančním sektoru ... 8
3 Návrh informačních systémů ... 9
3.1 Úvodem ... 9
3.2 UML ... 9
3.2.1 Základní elementy jazyk UML ... 9
3.2.2 Postupné kroky modelování ... 10
4 Návrh systému pro vyhodnocování aktivit pracovníků ve stromové hierarchii ... 13
4.1 Neformální specifikace ... 13
4.2 Rozdělení iterací ... 14
4.3 Případy použití ... 15
4.3.1 Diagram případů použití ... 15
4.3.2 Popis případů použití ... 17
4.3.3 Výjimky v případech použití ... 20
4.4 Konceptuální diagram tříd ... 20
4.5 Návrh schématu databáze ... 22
4.6 Diagramy interakce ... 24
4.7 Diagram návaznosti obrazovek ... 24
4.8 Návrh přejímacích testů validující případy použití ... 25
5 Závěr ... 28

# 1 Úvod

Na žádost několika pracovníků nejmenované finanční firmy jsem se rozhodl vytvořit informační systém dle jejich požadavků. Hlavním kritériem bylo zefektivnění a zrychlení evidence jejich klientů a smluv které jim poskytují. Finanční firma je založena na stromové hierarchii, jak je podrobněji popsáno v druhé kapitole a pracuje již s dlouholetou tradicí. Vyvíjený informační systém bude využíván a testován úzkou skupinou uživatelů kteří tvoří jednu „větev“ toho stromu.

V tomto semestrální projektu jsem navrhl model informačního systému v jazyce UML s hlediska pozdější implementace v rámci diplomové práce. Vybral jsem si objektově-relační databázi v prostředí Caché.

# 2 Řízení zaměstnanců ve stromové hierarchii

Základním principem těchto subjektů je přijímaní nových pracovníků a vytváření stromové struktury jak do šířky, tak do hloubky. Každý stávající zaměstnanec počínaje zakladatelem má právo přivézt do systému libovolný počet svých podřízených, kteří tvoří jeho podstrom. Rekurzivně i tito podřízení přivádějí další zaměstnance.

## 2.1 Síťový marketing versus „pyramidové struktury“

Firmy řízené stromovou hierarchií jsou světovým fenoménem již několik desetiletí. Široká veřejnost je zná pod pojmy MLM – multilevel marketing, siťový marketing ale někteří si je pletou s pyramidami (letadly) což jsou nelegální víceúrovňové struktury lidí, jejichž cílem je vydělat peníze na lidech, kteří se připojí po nich. Letadlo je založené na tzv. duplikačním efektu. Kdokoliv přivede někoho nového, vybere od něj vstupní poplatek, jehož část si nechá a část pošle na přerozdělení člověku, který ho do systému přivedl. Ten si část nechá a část pošle výš. A tak to jde až k jedinci na vrcholku pyramidy, který daný systém založil. Vzhledem k tomu, že trh je omezený, nemůže tento systém fungovat do nekonečna a po poměrně krátké době se zhroutí. Problémem je, že letadlo nevytváří žádnou přidanou hodnotu, pouze přerozděluje peněžní prostředky. Pyramidové hry jsou sice ve všech vyspělých zemích zakázány, stále je však obtížné systém rozpoznat a organizátory usvědčit.

Oproti tomu MLM je systém, který dává dohromady dvě marketingové strategie - přímý marketing a franchising. Subjekty, které tento systém využívají, zabývající se především přímým prodejem svých produktů (se kterým je multilevel marketing velmi úzce spjat) prostřednictvím svých obchodních zástupců, mají více či méně podobné systémy přerozdělování provizí. Hlavním rozdílem je, že tyto firmy na rozdíl od typických pyramid a letadel opravdu prodávají nějaké produkty - vytvářejí určitou přidanou hodnotu. Veškeré peníze (nebo jejich drtivá většina), které se v systému pohybují, jsou vygenerovány prodejem firemních produktů. To, že firma fungující na bázi MLM prodává určité produkty, její smysl na trhu nejen moralizuje, ale především legalizuje. Prodáváním produktů (zprostředkováním služeb) získávají provize sami zaměstnanci, kteří produkt poskytli a také zaměstnanci ve struktuře nad nimi.

V kosmetice, spotřebním průmyslu a zejména finanční sféře zaujímají velký podíl trhu. V české republice téměř všechny finanční podniky a ústavy spolupracují MLM firmami, jelikož se jedná o nejpřímější oslovení koncového zákazníka. Díky síti zaměstnanců pracujících nezávisle na ústředí firmy může tato síť pokrýt obrovské spektrum zákazníků.

## 2.2 Stromové hierarchie ve finančním sektoru

Finanční sektor je na české trhu zastoupen několika MLM společnostmi, které pokrývají většinu dostupných finančních produktů. Rozdíl mezi MLM společnostmi prodávajícími produkty je v tom, že zaměstnanci finančních společností svým klientům nenabízí přímé produkty ale zprostředkovávají služby.

Zaměstnanci jsou zprostředkovateli mezi finančními institucemi (bankami, pojišťovnami, spořitelnami atd.). Všechny nabízené služby by klient sice mohl získat sám v konkrétní finanční instituci, ale pro širokou veřejnost není jednoduché se objektivně orientovat v množství nabízených služeb různých konkurujících si institucí. Zaměstnanci MLM společností jsou speciálně školeni pro přehled a aktuální orientaci ve finančním sektoru. Jsou schopni svým klientům nejen nestranně nabídnout kompletní seznam nabízených služeb různými finančními institucemi, ale jsou také oprávněni tyto služby zprostředkovat. Tím je koncový zákazník ušetřen problémů týkajících se samostatného vyřizování všech náležitostí.

Za takto zprostředkované služby získávají poté zaměstnanci provize. A stejně jako v ostatních MLM společnostech tyto provize získávají zaměstnanci ve struktuře nad nimi.

# 3 Návrh informačních systémů

## 3.1 Úvodem

Návrh informačních systémů je obtížnou záležitostí pokrývající mnoho různorodých oblastí. Kvalitní objektově orientovaný návrh aplikace je nezbytným předpokladem jeho úspěšné implementace" Existuje několik metod a způsobů, jakým se dají informační systémy modelovat. Pro svůj projekt jsem si vybral jazyk UML.

## 3.2 UML

UML – Unified Modeling Language je unifikovaný modelovací jazyk, který má, na rozdíl od převážně textově orientovaných programovacích jazyků, vlastní grafickou syntaxi (pravidla pro sestavování jednotlivých elementů jazyka do větších objektů) a sémantiku (jednoznačná pravidla určující jednotlivým syntaktickým výrazům jejich význam).

V současné době má jazyk UML největší význam převážně při návrhu softwarových systémů. Pro objektově orientovaný návrh je samozřejmě možné použít různé podpůrné prostředky, zejména další odlišné typy diagramů, avšak UML přesně specifikuje, co má daný diagram obsahovat, což nejpodstatnější zejména při sdílení informací mezi jednotlivými analytiky a vývojáři. Je nutné, aby vytvářené grafy měly vnitřní konzistenci a přesně danou sémantiku, což u jiných typů grafů nemusí být obecně zaručeno. Existuje několik typů UML diagramů které se liší podle toho, jaké se pomocí nich plánují či zpracovávají úlohy. Tyto diagramy se od sebe odlišují především repertoárem použitých značek, způsobem jejich vzájemného propojení a také s nimi související sémantikou.

Jazyk UML je standardizován a podporován celou řadou vývojových nástrojů, což mohou být samotné aplikace určené pouze pro práci s UML, nebo také integrované vývojové prostředí, které v mnoha případech umožňují převod informací mezi UML diagramem a algoritmem již zapsaným v programovacím jazyce.

### 3.2.1 Základní elementy jazyk UML

UML je založen na třech elementech, které jsou z uživatelského hlediska reprezentovány grafickými značkami. Podle své funkce se nazývají:

- Předměty
- Digramy
- Relace

**Předměty** jsou elementy zpracovávaného modelu, které se dále člení do různých podkategorií. Tyto podkategorie jsou: strukturní abstrakce UML modelu (programové třídy, aplikační rozhranní, objektové rozhranní, případy použití, komponenty atd.), chování (komunikace mezi jednotlivými objekty), seskupení (seskupuje čísti diagramu na nižší úrovni, balíčky) a poznámky (pro bližší specifikaci vlastnosti a chování dalších elementů UML digramu)

**Relace** slouží k vzájemnému propojení, určování vztahů mezi různými předměty. V UML existují tyto typy relací: asociace (modelují obecnou, přesně definovanou závislost předmětů), závislost (jsou používány pokud aby změna v jednom předmětu způsobila změnu v předmětu jiném), generalizace (modeluje vztah, kdy je jeden předmět specializací jiného) a realizace (jedná se o druh vztahu, ve kterém jeden předmět představuje dohodu, za jejíž splnění je odpovědný jiný předmět)

**Diagramy** zachycují různé aspekty modelovaného systému, který nemusí být vyjádřen pouze jedním UML diagramem, protože je možné například pomocí balíčků sdružovat více diagramů do jednoho hierarchicky organizovaného celku. Typů diagramů existuje velké množství, uvedu například: diagram případů použití**,** diagram tříd**,** diagram objektů**,** diagram komponent**,** diagram nasazení**,** diagram aktivit**,** stavový diagram**,** diagram spolupráce a sekvenční diagram**.** Každý z těchto výše vyjmenovaných typů diagramů obsahuje poněkud odlišný repertoár dostupných grafických značek.

### 3.2.2 Postupné kroky modelování

Pro modelování informačního systému je vhodné požití modelovacích kroků sdružených do takzvaných iterací. Výsledkem každé iterace je přírůstek funkčnosti systému. Je proto vhodné jednotlivé iterace volit tak, aby bylo možné přehledně rozfázovat jednotlivé kroky návrhu. Samotná každá iterace se potom sestává z několika kroků, jejichž počet si můžeme sami volit dle potřeby návrhu. Níže uvedené jednotlivé kroky uvádím podle návrhu mého informačního systému. Každý

**Neformální specifikace** je prvním krokem, který je iniciován převážně od zákazníka, který si systém objednává. Jedná se slovní zadání problému, který by měl systém řešit, co si představuje zákazník že by měl systém umět. Jací budou jednotliví uživatelé systému a jaká budou mít práva a možnosti. Během samotného návrhu systému je možné tuto specifikaci modifikovat. Z pohledu UML nemá neformální specifikace žádný diagram, ale na jejím základě jsou následně digramy vytvářeny.

**Model případů použití** (Use case) zachycuje hlavní funkční požadavky na systém. Tyto modely jsou psány z pohledu zákazníka a podávají první představu o rozsahu projektu. Využívá se základních diagramů a přirozeného jazyka aby se pro zákazníka co nejpřehledněji načrtla funkční kostra systému. Správně navržené modely použití rozdělují odpovědnost mezi systémem a okolím a informují jaké jsou vazby systému k okolí. Druhým důležitým aspektem je, že model použití ukazuje kdo systém bude používat. Jsou namodelováni uživatelé systému tzv. aktoři. Název aktora vystihuje roli uživatele nebo jiného systému při interakci s naším systémem

**Detaily případů použití** se úzce vztahují k modelům případu použití. Jedná se o formálněji specifikované jednotlivé případy použití přirozených jazykem. Je vhodné blíže specifikovat složitější případy pro pozdější návrhy. Neexistuje přesně daná šablona popisu detailů, ale uvádějí se: název a popis případu použití, primární respektive sekundární aktéři, předpoklady a vstupní podmínky, hlavní tok – cože je základní posloupnost kroků případu použití, alternativní toky atd. dle potřeby popisu.

**Konceptuální diagram tříd** ukazuje strukturu systému. Třída popisuje množinu objektů, které sdílejí identickou sadu vlastností, omezení a mají stejnou sémantiku. Z laického hlediska můžeme objektem rozumět libovolný předmět, jenž má většinou svůj korelát v realitě (například faktura nebo objednávka). Tak většinou chápe pojem objekt zákazník, se kterým se analýza systému vytváří a který popisuje, jaké objekty jsou použity v jeho problémové doméně. Tyto objekty je třeba převést na konceptuální objekty. Diagram tříd potom modeluje tyto třídy a vztahy mezi nimi.

**Návrh schématu databáze** ukazuje jak budou samotná data v databázi uložena. Pro případ mého informačního systému se jedná o objektově-relační databázi Caché, proto je nejvhodnějším modelem ER diagram. Tento diagram ukazuje jednotlivé entity a vztahy mezi nimi. Zároveň modeluje integritní omezení zajišťující správnost a konzistenci ukládaných dat.

**Diagramy interakce** jsou hlavní technikou úrovně návrhu poskytovanou jazykem UML. Modely objektových i objektově-relačních databází se již nedají dostatečně postihnout pouze diagramy případů použití a diagramy tříd (respektive ER-diagramy). Diagramy interakce popisují spolupráci skupin objektů pro dosažení určitého chování. Typicky zachycují chování jednoho případu použití. Poslední verze UML 2.0 nabízí čtyři typy diagramů interakce: diagram sekvence (pohled na komunikaci jako posloupnost zpráv, které si účastníci interakce posílají, uspořádaném v čase), diagram komunikace (zdůrazňuje strukturní vztahy mezi objekty), přehledové diagramy interakce (ukazují, jak množina jednodušších interakcí realizuje složité chování) a digramy časování (zdůrazňuje aspekt reálného času, má charakter časových průběhů ukazujících doby trvání určitých stavů objektu)

**Stavové diagramy** nejsou specifické pro objektově orientované modelování, ale jazyk UML je nabízí pro strukturovanou analýzu. Jejich základem je koncept modelů automatů. Při objektovém modelování je využitelný pro zachycení stavu objektu a aktivit objektu v těchto stavech.

**Diagramy aktivit** někdy označované jako objektově orientované vývojové diagramy se většinou používají pro modelovaní business procesů, workflow, datových toků a operací. Oproti jiných diagramům modelujícím chování bez potřeby specifikovat statickou strukturu tříd a objektů, které tuto aktivitu realizují.

**Diagram komponent** ukazuje strukturu systému tvořenou komponentami a závislosti mezi softwarovými komponentami včetně klasifikátorů a artefaktů, které je implementují.

**Diagram nasazení** ukazuje nasazení softwarových prvků na fyzickou architekturu a také komunikaci mezi fyzickými prvky.

**Diagram objektů** ukazuje objekty a jejich vztahy v jistém časovém okamžiku.

**Diagram balíčků** ukazuje balíčky a vztahy závislosti a generalizace mezi nimi. Balíček je v UML chápán jako prostředek pro seskupování prvků modelu.

**Diagram složené struktury** ukazuje sestavu propojených prvků, které reprezentují instance existující v sobě běhu programu a které pracují pro dosažení nějakého společného cíle.

Výše uvedené diagramy nejsou konečný výčet všech možností jazyka UML. Také zdaleka ne všechny jsem modeloval v další kapitole, která se zabývá samotným návrhem konkrétního systému, ale jedná se o letmý souhrn možností jazyka UML pro modelování různých systémů a úhlu pohledu na něj.

# 4 Návrh systému pro vyhodnocování aktivit pracovníků ve stromové hierarchii

Celá tato kapitola se zabývá samotným návrhem vybraného systému převážně v jazyce UML. Je rozčleněna dle jednotlivých kroků návrhu a doplněna patřičnými diagramy vhodnými pro pozdější implementaci v prostředí objektově relační databáze Caché.

## 4.1 Neformální specifikace

Konzultací se zaměstnanci firmy jsme dospěli k této neformální specifikaci požadavků.

Je zájem vytvořit informační systém finanční společnosti, která je řízena stromovou hierarchií. Systém stromové struktury: jedná se o MLM společnost, kde každý zaměstnanec má právo do společnosti přivézt nové zaměstnance, kteří se ve stromové hierarchii zařadí pod něj. Rekurzivně každý takový nový zaměstnanec může přivézt do společností další zaměstnance a vytvořit si tak svůj vlastní podstrom. V případě výpovědi zaměstnance a jeho odebrání se systému se všichni jeho přímí „podzaměstnanci“ stávají přímými „podzaměstnanci“ jeho přímého nadřízeného.

Hlavním správcem systému je osoba(nadále uváděn jako správce), která má na starost udržování aktuálního stavu zaměstnanců, vztahů mezi nimi a také zpravuje seznamy služeb (tzn. smluv) které mohou zaměstnanci poskytovat svým klientům. Správce má právo zaměstnance do systému vkládat, odebírat nebo editovat jejich parametry tak aby zůstala zachována stromová struktura společnosti uvedená výše. Správce přidává, odebírá a edituje typy služeb(smluv) podle aktuální nabídky finančních ústavů, jejichž služby zaměstnanci nabízejí.

Druhou a několikanásobně frekventovanější skupinou uživatelů, kteří budou systém používat jsou zaměstnanci společnosti. Každý z těchto zaměstnanců musí mít svého nadzaměstnance a může mít libovolný počet podzaměstnanců o kterých mu systém musí nabízet aktuální přehled. Každý zaměstnanec:

- Má společností přiděleno jedinečné identifikační číslo pod kterým vystupuje a které je uváděno na každé jeho zprostředkované smlouvě.
- Má svoje klienty (fyzické a právnické osoby) o kterých potřebuje uchovávat informace. Tito klienti si skrze zaměstnance objednávají služby (stavební spoření, pojištění, povinné ručení atd…aktuální výčet služeb je proměnný podle toho, co momentálně nabízejí pojišťovny, banky …)

- zaměstnanec za každou poskytnutou službu klientovi (konkrétně za úspěšně podepsanou a vyřízenou smlouvu) dostává určité bodové hodnocení (dle tabulek podle typu smluv, pojistné částky atd.) a za tyto body má odpovídající finanční prémie. Díky stromové hierarchii také všichni zaměstnanci v hierarchii nad ním získají tento toto bodové hodnocení.
- Každý bod má svoji finanční hodnotu podle pozice na které se zaměstnanec nachází.
- Tato pozice se určuje podle celkového počtu bodů, které zaměstnanec získal od svého nástupu ke společnosti. (např: 0-100bodů = pozice 1 a 1bod = 100kč, 101-500bodů = pozice 2 a 1bod = 110kč….)
- Pokud některý ze zaměstnanců z firmy odejde, všichni jeho „podřízení“ jsou přiřazeni nadřízenému tohoto zaměstnance a nárok na klienty tohoto odcházejícího zaměstnance získává také jeho nadřízený.

Systém každému zaměstnanci bude umožňovat:

- uchovávat databázi jeho klientů s možností přidání odebrání klientů a editace jejich osobních dat.
- přidávat, odebírat a editovat klienty dle potřeby
- u každého klienta uchovávat typy smluv které má uzavřeny, data sjednání a respektive ukončení smluv a také základní důležité informace na těchto smlouvách jako je např. výše stavebního spoření a měsíční platba, výše pojistky, peněžní ústav atd..
- přidávat, odebírat a editovat smlouvy dle potřeby
- vypsání jednotlivých klientů a smluv, různě způsoby seřazení a také vyhledávání podle parametrů.
- za každou smlouvu ukazovat přidělený počet bodů a také celkový počet bodů a pozici zaměstnance
- vidět svoje přímé podřízené (ty které sám do firmy přivedl) a kolik bodů jde od každého z nich (respektive od podvětve stromu každého z nich). A také musí zobrazovat jeho přímého nadřízeného. Nemá již ale právo vidět klienty svých podřízených(nadřízeného) dokud mu nejsou nabídnuty po odchodu svého podřízeného z firmy.
- vidět aktuální tabulku smluv a případného bodového hodnocení za tyto smlouvy

## 4.2 Rozdělení iterací

Obsáhlost systému je vhodné rozdělit do několika iterací. Z hlediska jednotlivých typů aktérů tohoto systému jsem se rozhodl použít dvě iterace.

První iterace je zaměřena na návrh systému z pohledu správce (administrátora) a postihuje všechny jeho případy použití zobrazené v diagramu č.1. V rámci první iterace ní bude vytvořena stromová struktura systému. Což zahrnuje způsoby přidání, odebrání a editace jednotlivých zaměstnanců a vazby mezi nimi. Dále budou během první iterace navrženy tabulky smluv a jejich náležitosti.

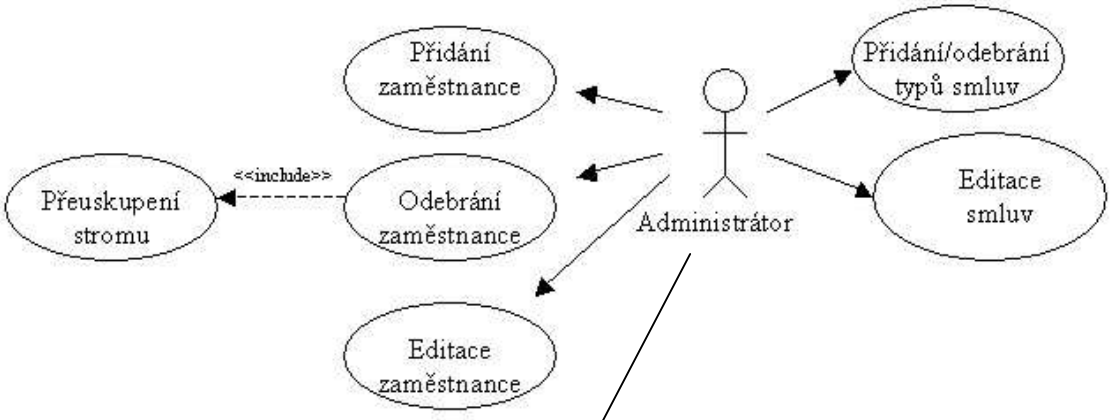
Druhá iterace zahrnuje návrh systému z hlediska zaměstnance a jeho případů použití, které jsou zobrazeny v diagramu č. 2. Jedná se o přidaní, editaci a mazání klientů, smluv, vyhledávací a řadící metody a možnosti systému.

# 4.3 Případy použití

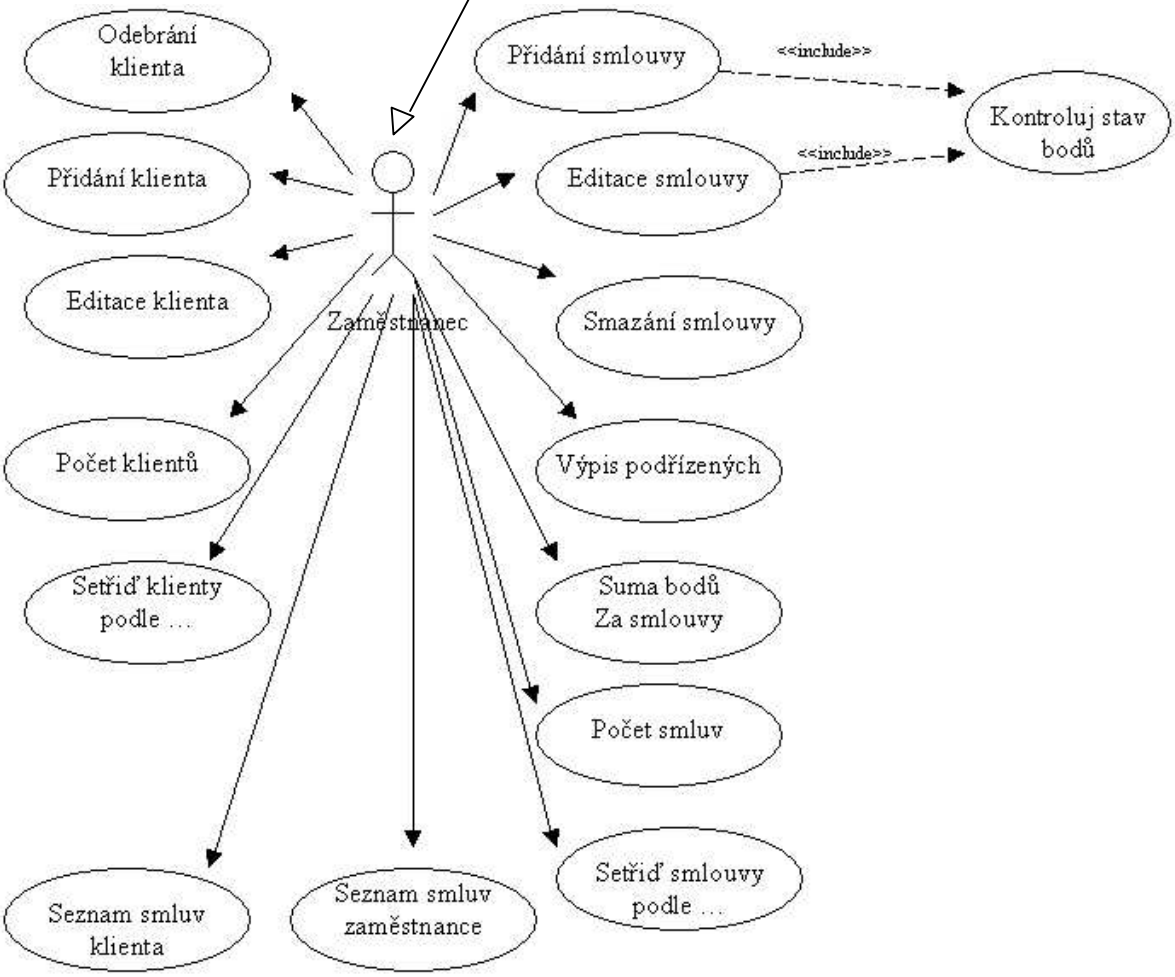
Jak bylo uvedeno v kapitole 4.2 jsou případy použití systému rozděleny do dvou iterací podle role aktérů, kteří budou v systému vystupovat. Případy použití (Use Case) jsou rozděleny do tří částí. První z nich jsou samotné diagramy obou iterací a vazba mezi nimi. Druhou je slovní strukturovaný popis vybraných případů použití první i druhé iterace a ve třetí části se zabývám potencionálními výjimkami v případech použití.

## 4.3.1 Diagram případů použití

První iterace zahrnuje případy použití administrátora. Z důvodu návaznosti na diagram případů použití druhé iterace jsou oba diagramy na následující stránce.




obr. 1 - diagram případů použití první iterace




obr.2 - diagram případů použití druhé iterace

### 4.3.2 Popis případů použití

Tato kapitola popisuje některé vybrané případy použití z diagramu v předchozí kapitole. Nejsou uvedeny konkrétně všechny případy použití, jelikož některé neuvedené jsou jen variací předchozích nebo ještě budou blíže specifikovány a upraveny po další konzultaci se zaměstnanci firmy.

| |
|---|
| Případ použití: **Přidání zaměstnance** |
| ID: **1** |
| Stručný popis: **přidá do systému nového zaměstnance na konec stromu** |
| Primární aktéři: **Administrátor** |
| Sekundární aktéři: |
| Předpoklady: **administrátor je přihlášen do systému** |
| Hlavní tok: **případ použití se spustí, když administrátor vybere „přidání nového zaměstnance“.**<br>**- dokud jsou zadané osobní údaje neplatné**<br>**- systém požaduje aby administrátor vyplnil povinné údaje jméno příjmení, rodné číslo, adresu, kontakt**<br>**- systém ověří jestli již zaměstnanec není evidován**<br>**- pokud je vše ověřeno korektně přidělí systém zaměstnanci osobní číslo (pokud tak neučinil administrátor)**<br>**- a přidá jej na „konec stromu“ tím že se zadá hodnota jeho přímého nadřízeného.**<br>**- vytvoří se nový zaměstnanec** |
| Následné podmínky: **systém musí nabídnout vytvořit strom i s novým zaměstnancem** |
| Alternativní toky: **zrušení přidávání** |

| |
|---|
| Případ použití: **Odebrání zaměstnance** |
| ID: **2** |
| Stručný popis: **odebere ze systému zaměstnance a upraví strom** |
| Primární aktéři: **Administrátor** |
| Sekundární aktéři: |
| Předpoklady: **administrátor je přihlášen do systému, zaměstnanec existuje** |
| Hlavní tok: **případ použití se spustí, když administrátor vybere „Odebrání zaměstnance“.**<br>**- když je zadáno platné osobního čísla zaměstnance a potvrzeno smazání**<br>**- nalezne zaměstnance s daným osobním číslem a vypíše jeho klienty smlouvy.**<br>**- Administrátor potvrdí smazání vybraného zaměstnance**<br>**- nalezne všechny jeho podřízené**<br>**- přepíše všem podřízeným nového nadřízeného, to je toho kterého měl mazaný zaměstnanec.**<br>**Když mazaný zaměstnanec má číslo nadřízeného 0,** |

| |
|---|
| - **systém vyhledá zaměstnance s nejnižším osobním číslem a přiřadí klienty a smlouvy jemu.**<br>- **přiřadí klienty a smlouvy mazaného zaměstnance jeho nadřízenému**<br>- **vymaže zaměstnance ze systému** |
| Následné podmínky: **systém musí umět vytvořit strom se změnami** |
| Alternativní toky: **žádné** |

| |
|---|
| Případ použití: **Editace zaměstnance** |
| ID: **3** |
| Stručný popis: **umí editovat zaměstnance a případně upraví strom** |
| Primární aktéři:**Administrátor** |
| Sekundární aktéři: |
| Předpoklady: **administrátor je přihlášen do systému, zaměstnanec existuje** |
| Hlavní tok: **případ použití se spustí, když administrátor vybere „Editace zaměstnance“.**<br>**- administrátor zadá platné osobního čísla zaměstnance**<br>**- když takový zaměstnanec existuje, systém:**<br>- **nalezne zaměstnance s daným osobním číslem**<br>- **vypíše všechny jeho parametry**<br>- **vygeneruje formulář pro jejich změnu.**<br>- **zažádá o uložení změn a uloží je**<br>- **když byla změněna hodnota nadřízeného nabídne vygenerovat upravený strom**<br>**- pokud neexistuje, nabídne znovu zadat číslo nebo storno** |
| Následné podmínky: **systém musí umět vytvořit strom se změnami, pokud k nim došlo** |
| Alternativní toky: **žádné** |

| |
|---|
| Případ použití: **Přidání typu smluv** |
| ID:**4** |
| Stručný popis: **přidá nový druh smlouvy do systému** |
| Primární aktéři: **Administrátor** |
| Sekundární aktéři: |
| Předpoklady: **administrátor je přihlášen do systému** |
| Hlavní tok: **případ použití se spustí, když administrátor vybere „přidání typu smluv“.**<br>**- systém vypíše všechny typy smluv které již existují**<br>**- administrátor si vybere zda chce založit úplně novou, nebo použít formulář některé z exitujících.**<br>**- Když vybere použít existující:**<br>- **systém vygeneruje formulář dle vybraného a nabídne jej editovat**<br>- **po upravení patřičných částí formuláře jej může administrátor uložit pod nových názvem**<br>**- jinak musí vytvořit celý sám:** |

| |
|---|
| - **vybere počet polí formuláře a jejich typy**<br>- **zadá, která pole jsou povinná a která ne**<br>**- po stisku „uložit“ se přidá nový druh smlouvy** |
| Následné podmínky: |
| Alternativní toky: |

| |
|---|
| Případ použití: **Odebrání typu smluv** |
| ID:**5** |
| Stručný popis: **odebere nepoužívanou smlouvu ze systému** |
| Primární aktéři: **Administrátor** |
| Sekundární aktéři: |
| Předpoklady: **smlouvu již aktivně nevyužívá žádný klient systému** |
| Hlavní tok: **případ použití se spustí, když administrátor vybere „odebrání typu smluv“.**<br>**- systém vypíše všechny typy smluv které již existují**<br>**- když administrátor vybere který typ chce vymazat**<br>- **systém zkontroluje, zda některý klient nemá aktivní tento typ smlouvy**<br>- **když je aktivní, upozorní administrátora a vrátí se k výběru typu smluv**<br>- **jinak systém zažádá o potvrzení smazání**<br>**- systém vymaže smlouvu ze systému** |
| Následné podmínky: |
| Alternativní toky: |

| |
|---|
| Případ použití: **Editace typu smluv** |
| ID:**6** |
| Stručný popis:**umožní změnit některé parametry existujícího typu smluv** |
| Primární aktéři:**Administrátor** |
| Sekundární aktéři: |
| Předpoklady:**smlouva již existuje** |
| Hlavní tok: **případ použití se spustí, když administrátor vybere „editace typu smluv“.**<br>**- systém vypíše všechny typy smluv které již existují**<br>**- když administrátor vybere který typ chce editovat**<br>- **systém nabídne parametry, které je možné editovat**<br>- **po změnění daných parametrů je administrátor vyzván k uložení**<br>**- systém uloží změněnou smlouvu** |
| Následné podmínky: |
| Alternativní toky: |

| |
|---|
| Případ použití**: Seznam smluv klienta** |
| ID: **7** |

| |
|---|
| Stručný popis: **vypíše seznam smluv konkrétního vybraného klienta** |
| Primární aktéři: **zaměstnanec** |
| Sekundární aktéři: |
| Předpoklady: **zaměstnanec přihlášen do systému** |
| Hlavní tok: **případ použití se spustí, když zaměstnanec vybere „vypiš seznam smluv klienta“.**<br>**- systém vypíše všechny klienty, které „obsluhuje“ zaměstnanec**<br>**- zaměstnanec vybere konkrétního z nich a potvrdí volbu**<br>**- systém vypíše na obrazovku jméno a osobní informace klienta a seznam všech smluv které má uzavřeny. Zároveň na obrazovce nabídne způsoby podle jakých je bude řadit.** |
| Následné podmínky: |
| Alternativní toky: |

### 4.3.3 Výjimky v případech použití

Seznam výjimek bude postupně upravován při zjištění nových.

**Vyjímka č. 1: Odebrání prvního zaměstnance stromu.**

- první zaměstnanec v hierarchii nemá uloženo v atributu „osobní číslo nadřízeného“ číslo nějakého nadřízeného, ale NULU. Pokud dojde k jeho zrušení, systém automaticky jeho přímým podřízeným zadá do tohoto atributu také nulu. Viz příklady přejímacích testů v kapitole 4.8.

## 4.4 Konceptuální diagram tříd

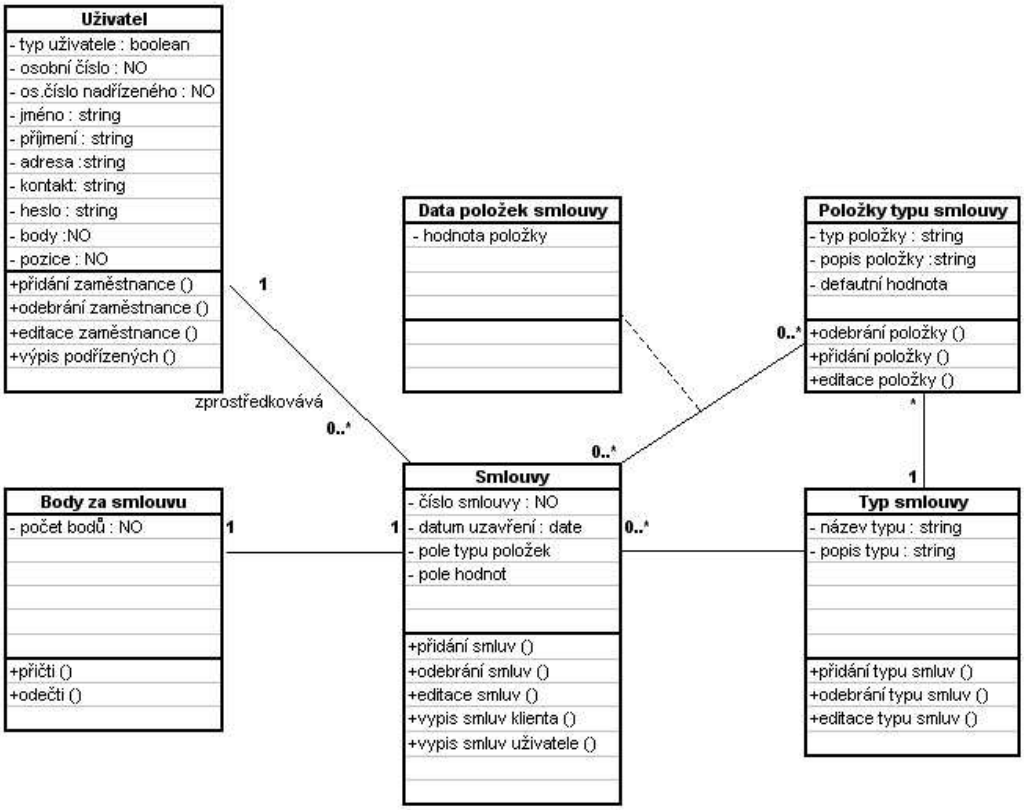
Diagram tříd je navržen opět pro první i druhou iteraci. V první iteraci jsou modelovány třídy týkající se uživatele (konkrétně objektů typu administrátor) a objekty pro práci se smlouvami. U každé třídy jsou zobrazeny atributy žádané zaměstnanci nebo potřebné pro implementaci. Jejich rozšíření respektive změna bude doplněna po konzultaci se zaměstnanci společnosti. Analogicky jsou každé třídě přiřazeny metody podle specifikace zaměstnanců a jejich rozšíření bude ještě konzultováno.

Vytvoření stromové struktury mezi jednotlivými zaměstnanci zajišťuje atribut os.číslo_nadřízeného:int který uchovává z každého zaměstnance osobní číslo zaměstnance, který jej do společnosti přivedl a tudíž je v hierarchii přímo nad ním.

Třída uživatel zahrnuje objekty typu administrátor nebo zaměstnanec rozlišené boolovskou proměnnou typ_uživatele, což vysvětluje generalizační vztah administrátora a zaměstnance mezi digramy použití první a druhé iterace.

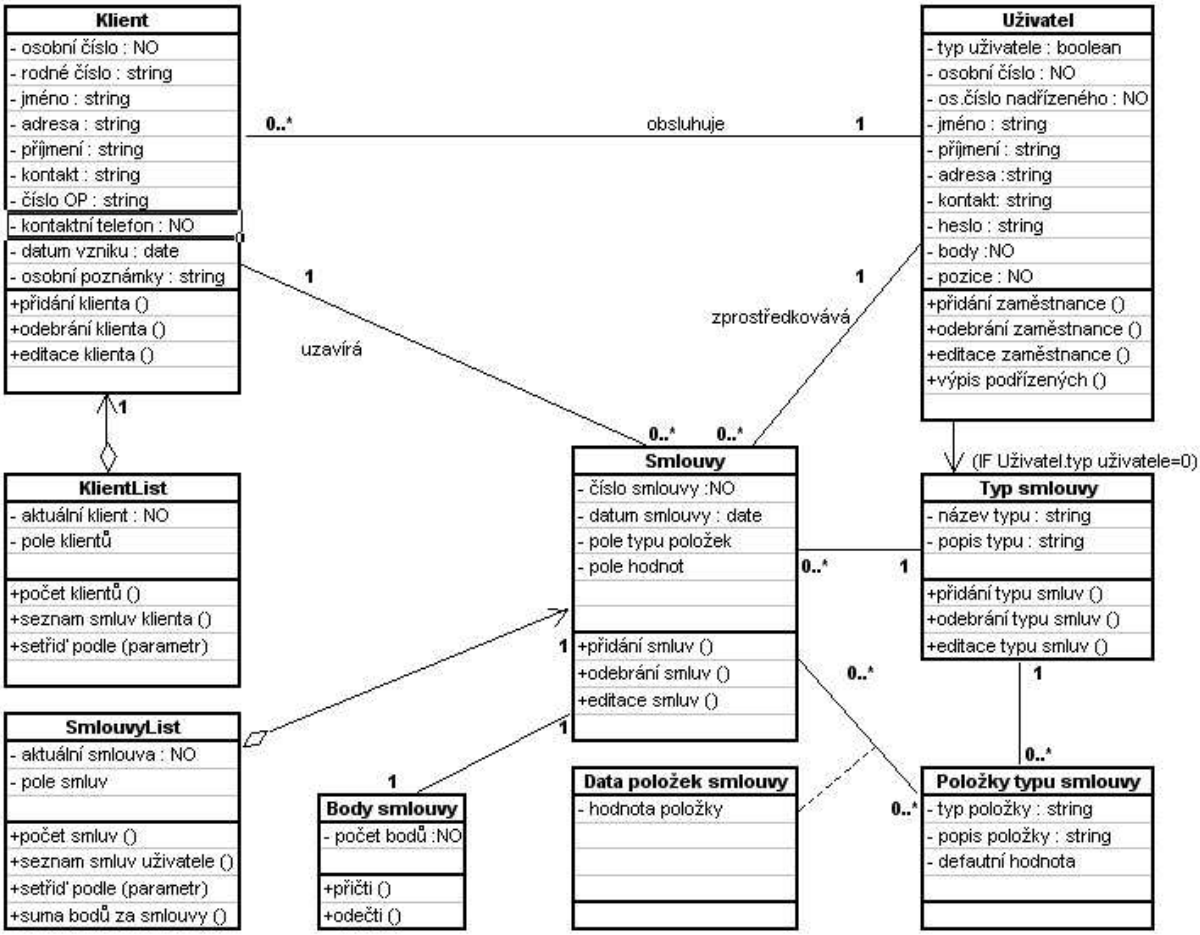
Třída Body za smlouvu umožňuje přidání respektive odebrání patřičného počtu bodů za každou smlouvu v systému.

Každý objekt třídy Smlouvy musí uchovávat povinné atributy a musí být určitého typu, což realizují třídy Typ smlouvy a Položky typu smlouvy. Toto rozdělení bude podrobněji vysvětleno v následující kapitole 4.5 při návrhu schématu databáze, jelikož každý typ smlouvy obsahuje uchování jiných parametrů a informací ze smlouvy. Třída Data položek smlouvy je externí asociační třídou, která doplňuje hodnoty jednotlivých položek.




obr.3 - konceptuální diagram tříd první iterace

Ve druhé iteraci je diagram tříd (diagram č.4) rozšířen o třídy modelující klienty zaměstnanců a vztahy k nich a také o třídy KlientList a SmlouvyList zpřístupňující metody pro práci nad seznamem klientů a smluv. Blíže popisuji vztahy mezi jednotlivými třídami z hlediska návrhu schématu databáze v kapitole 4.5.



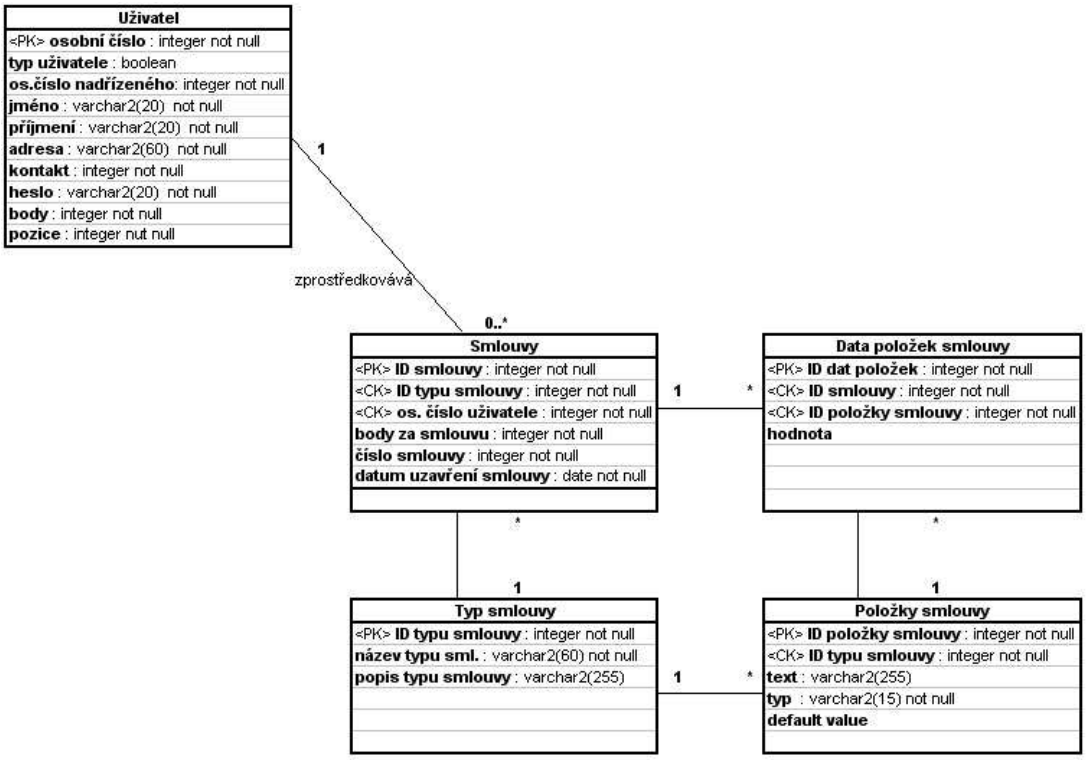

obr.4 - konceptuální diagram tříd druhé iterace

# 4.5 Návrh schématu databáze

Analogicky jako v kapitole o konceptuálním diagramu tříd je i návrh schématu databáze rozdělen do dvou iterací. Jedná se o dva ER diagramy modelující strukturu dat v databázi včetně integritních omezení.

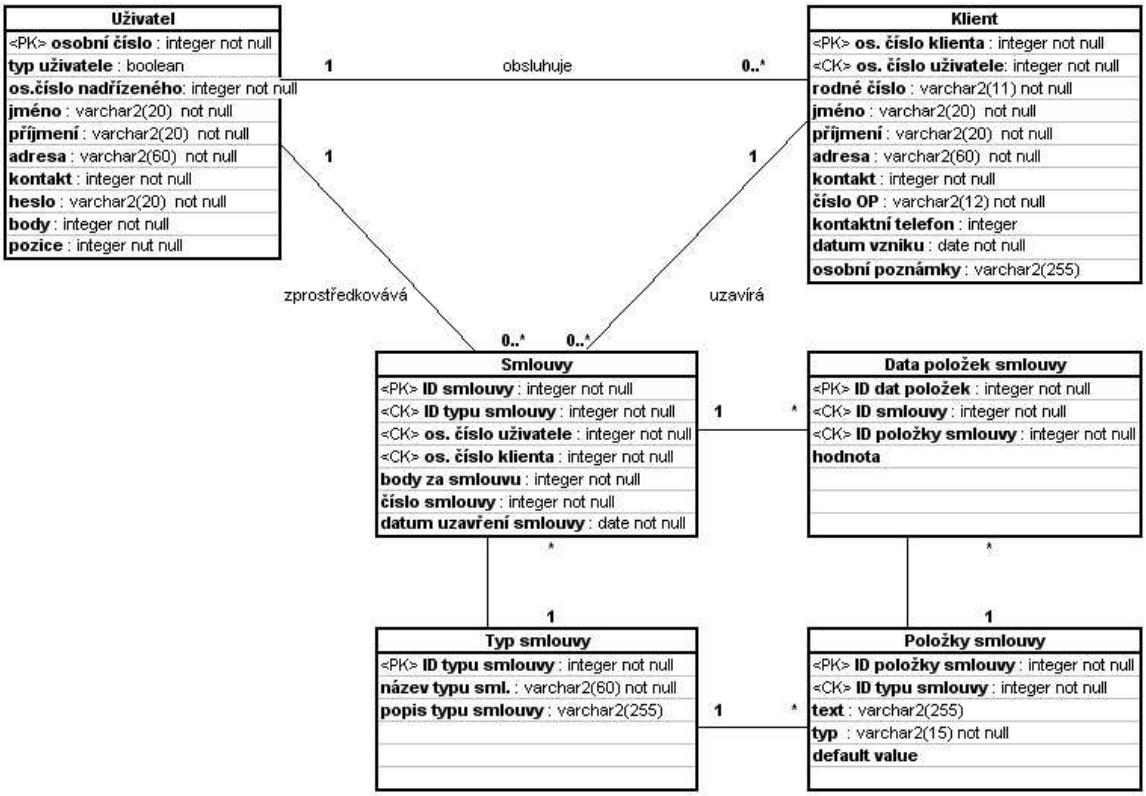
Entita uživatel má jako primární klíč uloženo svoje osobní číslo a také osobní číslo nadřízeného je přímo uloženo v databázi, čímž je zachována stromová hierarchie společnosti.

Komplikovanější je modelování smluv a jejich typů. Z finančního hlediska má každý typ smlouvy jiné náležitosti a je potřeba zaznamenávat a uchovávat jiné informace. Samotná entita smlouvy uchovává informace o bodech získaných za smlouvu, číslo smlouvy (což může být i číselná řada typů smluv) a datum uzavření smlouvy, které je také společné všem typům smluv. Entita typ smlouvy potom zajišťuje, že každá smlouva může být jen jednoho určitého typu a následně entita Položky smlouvy specifikují konkrétní náležitosti konkrétního typu smlouvy. Samotná data se ukládají v entitě Data položek smlouvy.




obr.5 - ER diagram první iterace

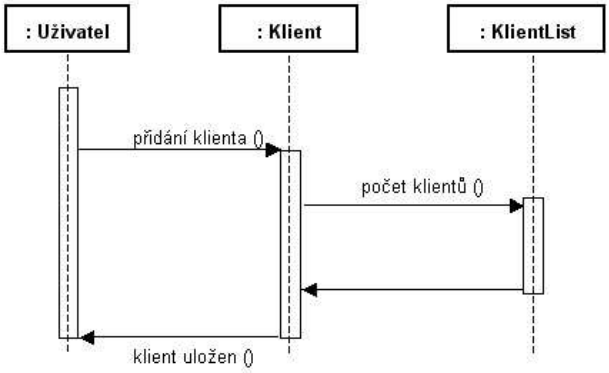
ER diagram druhé iterace již analogicky podle konceptuálního diagramu tříd modeluje návrh celé databáze a přidané vztahy po přidání entity klienta




obr.6 - ER diagram druhé iterace

## 4.6 Diagramy interakce

Diagramy interakce jak již bylo popsáno ve třetí kapitole popisují spolupráci skupin objektů pro dosažení určitého chování. Typicky zachycují chování jednoho případu použití. Je možné použít diagramy sekvence nebo komunikace. Diagramy sekvence ale mají z pohledu UML vyšší modelovací sílu. V této kapitole uvádím jen některé diagramy, zachycující modely případy použití. Časem mohou být opět rozšířeny po konzultaci se zaměstnanci.




obr.7 - sekvenční diagram modelující přidání nového klienta zaměstnancem

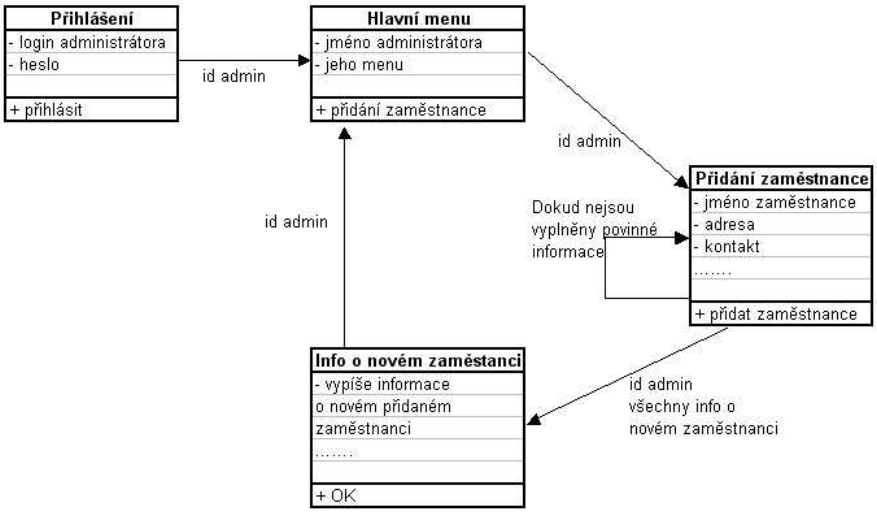
## 4.7 Diagram návaznosti obrazovek

Diagram návaznosti obrazovek není typickým UML diagramem. V tomto modelu jej uvádím jako ukázku jak budou z pohledu koncového uživatele systému mezi sebou jednotlivé obrazovky navazovat a komunikovat. Jaké hodnoty se mezi jednotlivými obrazovkami přenáší a co přibližně zobrazují. V pozdější části implementace systému mohou být takovéto diagramy nahrazeny screenshoty jednotlivých obrazovek pro vizuální prezentaci zákazníkovy.

**Popis diagramu obrazovek:**

1) obrazovka (Přihlášení)
   Formulář pro vyplnění login a hesla uživatele. Po stisku tlačítka přihlásit systém ověří přihlašovací informace a přejde k obrazovce č.2.
2) obrazovka (Hlavní menu)
   Zobrazuje jméno administrátora a jeho menu. Jednou z položek menu je i přidání uživatele, po jehož kliknutí se přejde k obrazovce č.3. ID administrátora se mezi jednotlivými obrazovkami neustále přenáší.
3) obrazovka (Přidání zaměstnance)
   Vygeneruje formulář pro vyplnění osobních informací zaměstnance. Po stisku tlačítka přidat zaměstnance se ověří platnost a vyplnění dat a dokud není vše platně vyplněno opakuje zobrazení této obrazovky.

4) obrazovka (Info o novém zaměstnanci)
Z předchozí obrazovky se přenese kromě id administrátora i všechny informace o novém uživateli a vypíší se pro informaci na obrazovku. Po stisku tlačítka OK se přejde opět na hlavní menu (obrazovka 2)

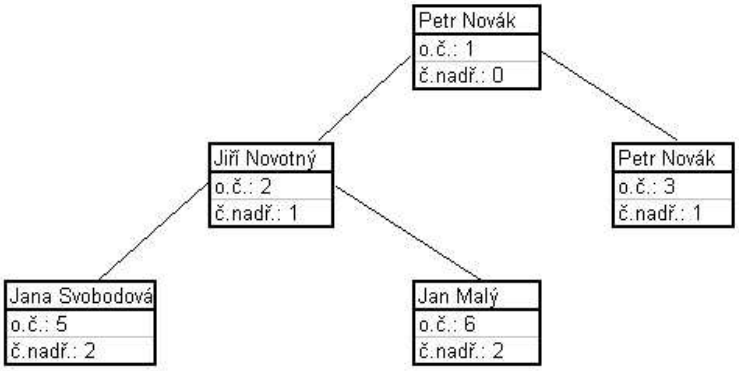



obr.8 - diagram návaznosti obrazovek případu použití přidání nového zaměstnance

## 4.8 Návrh přejímacích testů validující případy použití

Pro návrh těchto testů jsem vytvořil miniaturní model společnosti o několika zaměstnancích a minimálně klientech i smlouvách. Cílem je demonstrovat a otestovat chování systému pro konkrétní případ použití. Na přání zaměstnanců mohou být přidány další testy.

Existuje stromová struktura firmy o několika zaměstnancích (viz ukázkový strom níže), nad kterou test poběží:




obr.9 - demonstrační stromová hierarchie společnosti pro přejímací testy

Klienti a smlouvy jednotlivých pracovníků:

**Petr Novák**

1) Josef Nikdo (Stavební spoření s číslem 1)

**Jiří Novotný**

1) Alexandra Němcová (Stavební spoření s číslem 2)

**Jana Svobodová**

Martin Svoboda (Penzijní připojištění s číslem 1)

**Test číslo 1 – vymazání zaměstnance s osobním číslem 6.**

1) Administrátor zadá vymazání zaměstnance s osobním číslem 6.
2) Systém v databázi nalezne zaměstnance Jana Malého, poté nalezne všechny klienty které obsluhoval a všechny smlouvy které zprostředkoval.
3) Systém vypíše všechny osobní informace o Janu Malém a seznam jeho klientů a smluv.

   **Jan Malý**

   **osobní číslo: 6**

   **číslo nadřízeného:2**

   Klienti: -----

   Smlouvy: -----
4) Administrátor potvrdí smazání vybraného zaměstnance
5) Systém se pokusí najít v databázi zaměstnance, kteří mají číslo nadřízeného 6. Jelikož žádné nenajde, přejde k bodu 6)
6) Jelikož nenašel žádné smlouvy ani klienty, přejde k bodu 7)
7) Vymaže zaměstnance ze systému

***Testuje, zda vymaže zaměstnance bez klientů, smluv a podřízených.***

**Test číslo 2 – vymazání zaměstnance s osobním číslem 4.**

1) Administrátor zadá vymazání zaměstnance s osobním číslem 4.
2) Systém v databázi nenalezne žádného zaměstnance s tím osobním číslem a nabídne administrátorovi opětovné zadání osobního čísla nebo návrat do hlavního menu .

***Testuje chování systému v situaci, kdy administrátor zadá špatné osobní číslo.***

**Test číslo 3 – vymazání zaměstnance s osobním číslem 2.**

1) Administrátor zadá vymazání zaměstnance s osobním číslem 2.
2) Systém v databázi nalezne zaměstnance Jiřího Novotného, poté nalezne všechny klienty které obsluhoval a všechny smlouvy které zprostředkoval.
3) Systém vypíše všechny osobní informace o Jiřím Novotném a seznam jeho klientů a smluv.

**Jiří Novotný**

**osobní číslo: 2**

**číslo nadřízeného:1**

Klienti: Alexandra Němcová

Smlouvy: Stavební Spoření s číslem 2

4) Administrátor potvrdí smazání vybraného zaměstnance
5) Systém se pokusí najít v databázi zaměstnance, kteří mají číslo nadřízeného 2. Nalezne Zaměstnance s osobním číslem 5 a 6. Oběma změní hodnotu čísla nadřízeného na 1.
6) Klienta Alexandru Němcovou systém přiřadí nadřízenému Jiřího Novotného, tzn: přepíše hodnotu osobního čísla uživatele z dvojky na 1.

   Stavební spoření s číslem 2 přepíše na nadřízeného: tzn: osobní číslo uživatele u smlouvy změní z 2 na 1.
7) Vymaže zaměstnance ze systému

***Testuje, zda vymaže zaměstnance který není na vrcholu stromu a má smlouvy i podřízené.***

**Test číslo 4 – vymazání zaměstnance s osobním číslem 1.**

1) Administrátor zadá vymazání zaměstnance s osobním číslem 1.
2) Systém v databázi nalezne zaměstnance Petra Nováka, poté nalezne všechny klienty které obsluhoval a všechny smlouvy které zprostředkoval.
3) Systém vypíše všechny osobní informace o Petru Novákovi a seznam jeho klientů a smluv.

   **Petr Novák**

   **osobní číslo: 1**

   **číslo nadřízeného:0**

   Klienti: Josef Nikdo

   Smlouvy: Stavební Spoření s číslem 1
4) Administrátor potvrdí smazání vybraného zaměstnance
5) Systém se pokusí najít v databázi zaměstnance, kteří mají číslo nadřízeného 1. Nalezne Zaměstnance s osobním číslem 2 a 3. Oběma změní hodnotu čísla nadřízeného na 0.
6) Jelikož je splněna podmínka, že zaměstnanec má číslo nadřízeného 0, tak systém nalezne zaměstnance s nejnižším osobním číslem což je č.2 (Jiří Novotný) a klienta Josefa Nikda systém přiřadí Jiřímu Novotnému, tzn: přepíše hodnotu osobního čísla uživatele z 1 na 2. Stejným způsobem i stavební spoření s číslem 1 přepíše na Jiřího Nováka: tzn: osobní číslo uživatele u smlouvy změní z 1 na 2.
7) Vymaže zaměstnance ze systému

***Testuje, zda vymaže zaměstnance který je na vrcholu stromu a má smlouvy i podřízené***

# 5 Závěr

Cílem tohoto semestrálního projektu byl návrh modelu informačního systému. Ve stavu v jakém se nachází vychází z první neformální specifikace a dvou konzultacích se zaměstnanci společnosti, kteří jej budou používat.

Tento návrh bude s nimi v nejbližší době opětovně konzultován a eventuelně poupraven dle konkrétních požadavků. Samotná struktura systému vycházející z případů použití a diagramů tříd a ER diagramu pravděpodobně nebude zásadně modifikována, spíše se předpokládá její potencionální rozšíření o další případy použití a z toho plynoucí operace tříd. Na přání zaměstnanců budou domodelovány konkrétní další diagramy eventuelně přejímací testy.

V další části je plánována implementace v prostředí postrelační databáze Caché. Kompletní konečný model systému a jeho implementace v Caché budou tvořit celou diplomovou práci. Systém by ve své finální podobě měl sloužit pro malou skupinu „testovacích“ uživatelů pro soukromou evidenci jejich zákazníků a smluv. Měl by být dostatečně variabilní pro stávající i plánované typy služeb na našem finančním trhu.

# Literatura


[1] Ivana Rábová. Diagramy UML a Rational Rose, Pef MZLU, 2000.

[2] Hana Kanisová, Miroslav Müller. UML srozumitelně. Praha,Computer Press 2006.

[3] Jim Arlow,Ila Neustadt. UML a unifikovaný proces vývoje aplikací, Praha, Computer Press 2003.

[4] Ilja Kraval. Nejčastější chyby při modelování 1, URL: <http://www.dbsvet.cz/view.php?cisloclanku=2004080401> [cit. 2006-12-26].

[5] René Stein. Návrh aplikací v jazyce UML, URL: <http://interval.cz/clanky/navrh-aplikaci-v-jazyce-uml-unified-modeling-language/> [cit. 2006-12-26].

[6] Pavel Tišnovský. Nástroje pro tvorbu UML diagramů, URL: <http://www.root.cz/clanky/nastroje-pro-tvorbu-uml-diagramu/> [cit. 2006-12-26].

[7] Jan Spilka. Multilevel – cesta k bohatství či do pekel?, URL: <http://www.mesec.cz/clanky/multilevel-cesta-k-bohatstvi-ci-do-pekel/> [cit. 2006-12-26].

[8] Síťový marketing versus "pyramidové struktury, URL: <http://aktualne.centrum.cz/finance/investice/clanek.phtml?id=137950> [cit. 2006-12-26].