EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Název projektu	Rozvoj vzdělávání na Slezské univerzitě v Opavě
Registrační číslo projektu	CZ.02.2.69/0.0./0.0/16_015/0002400

Objektové metody modelování v příkladech

Distanční studijní text

Zdeněk Franěk

Karviná 2018

SLEZSKÁ UNIVERZITA
OBCHODNĚ PODNIKATELSKÁ FAKULTA V KARVINÉ

Obor: Informační a komunikační technologie (ICT), Vývoj a analýzy softwaru a aplikací

Klíčová slova: Analýza, software, diagramy UML, metodika RUP, objekty, třídy, polymorfismus, dědičnost.

Anotace: Tento učební text se zabývá analýzou návrhu software s využitím UML diagramů a metodiky RUP. Je určen pro studenty 1. ročníku studijního programu Manažerská informatika v navazujícím stupni studia, zejména v kombinované formě výuky. Z tohoto důvodu je forma učebního textu koncipována tak, aby studenti měli k dispozici text postačující k ovládnutí dané problematiky a zároveň si mohli své znalosti po každé kapitole ověřit pomocí testových otázek nebo úkolů.

Hlavním tématem studijní opory je popis analytických postupů při návrhu informačních systémů - software s využitím jazyka Unified Modeling Language (UML) a metodiky RUP. Modelovací jazyk UML je základní nástroj pro objektovou analýzu při návrhu software a informačních systémů. Tato studijní opora navazuje na předchozí studijní oporu z roku 2014 s názvem „Objektové metody modelování – Výklad jazyka UML“. Tato studijní opora je určena zejména pro studenty v distanční formě studia a tomu odpovídá jiná struktura studijní opory dle předepsané wordovské šablony. V nové studijní distanční opoře je kladen důraz na praktické předvedení možností objektových technik modelování, a proto je výklad provázen řadou praktických příkladů. Po každé kapitole jsou zařazeny testové otázky tak, aby si student osvojené poznatky mohl sám otestovat.

Obě studijní opory se vzájemně doplňují a jsou dostupné v e-learningovém kurzu pro výuku povinně volitelného předmětu „Objektové metody modelování“ magisterského studijního programu „Manažerská informatika“

Objektově orientované modelovací techniky a jazyk UML jsou ústředním tématem této opory.

Autor: **RNDr. Zdeněk Franěk, Ph.D.**

Obsah

ÚVODEM

Návrh informačních systémů (dále jen IS), metodika jejich vytváření, programování systémů a aplikací, vytváření software, objektové metody modelování s využitím jazyka Unified Modeling Language (dále jen UML) jsou na vrcholu témat tvůrčí práce v oblasti informačních technologií (dále jen IT) technologií.

Učební text, který je z výše uvedených oblastí návrhu informačních systému zaměřen především na objektové metody modelování s využitím jazyka UML a metodiku vytváření software, je určen studentům 1. ročníků studijního programu Manažerská informatika na Obchodně podnikatelské fakultě v Karviné, Slezské univerzity v Opavě (dále jen SU OPF). Je zaměřen na kombinovanou formu studia a tomu je určena forma textu. Text je provázen ikonami, zdůrazňujícími hlavní rysy IT technik z výše uvedenou tématikou. Na začátku každé kapitoly je shrnutí čemu se studenti naučí, následuje výklad a příklad pokud to umožňuje charakter tématu. Vždy jsou graficky zdůrazněny hlavní myšlenky, resp. poznatky. Na konci každé kapitoly jsou uvedeny testové otázky k procvičení probrané látky. Správné odpovědi pro samokontrolu jsou přiloženy hned za otázkami.

Pochopení textu této distanční studijní opory předpokládá u studentů základní znalosti z oblasti informačních technologií. Je určen studentům navazujícímu stupni studia, především v kombinované formě studia. Tyto předpokládané znalosti odpovídají znalostem získaných absolvováním bakalářského stupně studia studijního programu Manažerská informatika na OPF. Jedná se zejména o zvládnutí základních poznatků z předmětu databázové systémy, algoritmy a datové a procesní modelování. Všechna tato témata jsou součástí studia v bakalářském stupni, studijního programu Manažerská informatika. Přesto lze k textu přistoupit bez předchozího studia a v případě neporozumění textu si znalosti doplnit z příslušných částí příslušných studijních opor.

Text je doprovázen případovou studií a látka je ilustrovaná příklady, na kterých lze porozumět předchozí teorii. Doporučený postup při práci s textem je následující: Přečíst si důkladně teorii a hned si ji ověřit na příkladu. Poté se znovu vrátit k teorii a pomocí příkladu objasnit sporná nebo těžká místa. Zpětným pročítáním textu lze tzv. iterační metodou přírůstků znalostí dospět k pochopení probíraných témat.

Zvláštní pozornost doporučujeme věnovat rovněž metodice návrhu informačních systémů. Je nutno zdůraznit, že bez dobré metodiky nelze dosáhnout úspěchu při návrhu IS. Metodika Unified Process (dále jen UP), objektové metody modelování a UML diagramy spolu úzce souvisí.

Je třeba zdůraznit, že tento učební text se věnuje výhradně fázi analýzy před zahájením programování informačního systému, resp. software obecně. Cílem autora bylo popsat všechny techniky tak, aby po jejich aplikaci bylo možno zahájit programátorské práce a aby

si zúčastněné strany (zadavatel, uživatel, analytik a programátor) plně rozuměli. Samozřejmě to nevylučuje to, že se UML diagramy často používají i pro nasazení a dokumentaci programových systémů.

Tato studijní opora je součástí e-learningového kurzu na http://elearning.opf.slu.cz, který je dostupný pro studenty předmětu „Objektové metody modelování“.

Text byl sestaven na základě využití literatury, zkušeností autora a na základě přednášek a seminářů během výuky předmětu.

V práci jsou využity poznatky při zpracování seminárních prací studenty ve spolupráci s lektorem předmětu.

V předmětu je využíván software, při jednoduchých úlohách je to MS VISIO a šablony pro UML. Pro složitější úlohy byly v předmětu využívány CASE nástroje. Nejdříve to byl CASE firmy IBM Rational IBM Architect. Později velmi kvalitní a nejvíce rozšířený CASE nástroj firmy SPARX Enterprise Architect. Tento software se při rozvoji předmětu plánuje využívat jako číslo 1.

Poděkování za spolupráci na této učební opoře patří samozřejmě autorům literatury, studentům-účastníkům přednášek a seminářů, dále pak spolupracovníkům z katedry matematiky a informatiky SU OPF a za podporu mým blízkým.

RYCHLÝ NÁHLED STUDIJNÍ OPORY

Cílem studijní opory je srozumitelnou formou s využitím ikon a struktury používané pro kombinovanou formu výuky seznámit studenty s teorií objektového modelování systémů a jejím významem pro projektování informačních systémů. Předmět seznámí studenty s historickým vývojem objektového přístupu a používanými standardy v dané oblasti. V textu jsou studenti seznámeni s metodikou RUP (Rational Unified Process). Hlavní náplň opory je věnována jazyku UML (Unified Modeling Language). Jazyk UML byl koncipován pro návrh, analýzu, tvorbu a dokumentaci informačních systémů s objektově orientovaným přístupem. V rámci studia učební opory studenti získají znalosti standardního způsobu zápisu – koncepce návrhu systému, jako jsou business procesy a systémové funkce. Dále se naučí tvořit konkrétní prvky, jako jsou například znovupoužitelné programové komponenty a databázová schémata. Na příkladech je demonstrována praktická práce s jazykem UML. V textu jsou procvičeny základní postupy při vývoji software s využitím Enterprise Architect (označuje se zkratkou EA) firmy SPARX, který patří do rodiny Computer Aided Software Engineering (ve zkratce CASE) a šablony pro UML diagramy produktu VISIO firmy Microsoft. Studenti jsou seznámeni se softwarem IBM Rational Enterprise Architect, průkopníkem mezi CASE softwarovými nástroji na poli vývoje informačních systémů.

Učební text se dělí do těchto témat:

1. Úvod do objektového modelování
2. Shrnutí základních pojmů objektově orientované analýzy a návrhu software. Pojem objekt. Základní koncepty: abstrakce, zapouzdření, skrývání informací, třídy, dědičnost, interface.
3. Popis jazyka UML
4. Co je UML, objekty a jazyk UML, struktura jazyka, stavební bloky, vyjádření tříd, atributů a operací. Obecný přehled diagramů UML. Diagramy UML: případy užití, stavové diagramy, diagramy sekvencí, diagramy spolupráce, diagramy tříd, diagramy činností, diagramy architektury.
5. Software produkty pro práci v UML
6. Přehled softwarových nástrojů pro objektové modelování. Představení a zpřístupnění software EA SPARX a MS VISIO - část pro kreslení UML diagramů. Případy užití.
7. Metodika Unified Process (UP) a Rational Unified Process (RUP)
8. Hlavní principy moderního iterativního vývoje softwaru metodikou RUP.
9. Jednotlivé fáze životního cyklu projektu, který vyvíjí software - účel, možnosti, rizika vývoje.
10. Případové studie návrhu informačního systému s využitím UML a metodiky RUP.

1 ÚVOD DO OBJEKTOVĚ ORIENTOVANÉHO MODELOVÁNÍ

RYCHLÝ NÁHLED KAPITOLY

Úvodní kapitola učebního textu seznamuje studenty s principy objektově orientovaného návrhu software. Objektově orientované modelování ve zkratce OOM pochází z anglického Object-Oriented Modeling. V první kapitole je popsána historie vzniku objektově orientovaného přístupu vývoje software a srovnání s předchozím procedurálním přístupem. V kapitole jsou popsány základní pojmy jako je třída, objekt, zapouzdření, dědičnost a polymorfismus a některé další vlastnosti OOM. Objektově orientované modelování pokrývá analýzu, programování a nasazení software.

CÍLE KAPITOLY

Cílem úvodní kapitoly je seznámit čtenáře se základními pojmy objektově orientovaného modelování při vytváření software, obecněji informačních systémů. Základní pojmy popsat a vysvětlit, a také doložit na příkladech.

ČAS POTŘEBNÝ KE STUDIU

100 minut.

KLÍČOVÁ SLOVA KAPITOLY

Objekt, třída, polymorfismus, dědičnost, zapouzdření, překrývání metod.

1.1 Od strukturálního pojetí k objektovému

Strukturální pojetí programování

Strukturální pojetí programování je charakteristické tím, že naprogramovaný kód pracuje přímo s daty. Vývoj programového kódu aplikace byl sestaven strukturovaně. Hlavním rysem analýzy při vývoji aplikace bylo to, že se aplikace dělila na dynamickou funkční část a statickou datovou část. Data byla ukládána v jednotlivých souborech nebo později v relačních databázích. Programový kód byl psán shora dolů s využitím volání funkcí. To na jedné straně zpřehledňovalo programování a na straně druhé zavádělo prvky opakované použitelnosti, tzv. re-use. Při tomto přístupu k programování narůstala složitost programů a používané analytické postupy při návrhu softwaru narážely na velké problémy propojení vrstvy datové a funkční. Výpočty a zacházení se stejnými daty se prováděla na mnoha místech programového kódu a bylo hodně komplikované provádět změny a další přidávání funkcionalit systému. Po zavedení architektury klient - server nastaly další potíže jak správně vyvíjet informační systémy.

Z výše uvedených důvodů se v programátorské komunitě ve světě začal používat objektově orientovaný přístup. Jedním z hlavních cílů objektově orientované analýzy, návrhu a programování je další zvýšení produktivity vývojářských prací. [Kan2004]

Objektově orientovaný přístup vývoje software

Objektově orientovaný přístup vývoje software je mladší technika navazující na strukturovaný přístup. Tento přístup je založen na objektech. Objekty jsou struktury, které mají definované vlastnosti (atributy) a své chování (operace), které daný projekt může provádět. Informační systém (IS), resp. software takto vyvinutý je chápán jako množina spolupracujících objektů. Tento přístup umožňují programovací jazyky jako je Java, C#, Smaltalk, atd. Návrh – analýza při tvorbě informačních systémů je reprezentována CASE nástroji (Computer Aided Software Engineering) a Unified Modeling Language. Modelovací jazyk UML je naprosto v souladu s objektovým přístupem. V systémech napsaných pomocí OOM se daleko více uplatňuje znovu-použitelnost. Zavádí se pojmy třída, dědičnost, zapouzdření, komponenty, distribuované objekty a jiné, které oproti strukturálnímu přístupu výrazně zvyšují produktivitu analýzy a programování systémů. [Kan2004]

CHARAKTERISTIKA KÓDU PROGRAMU

Df

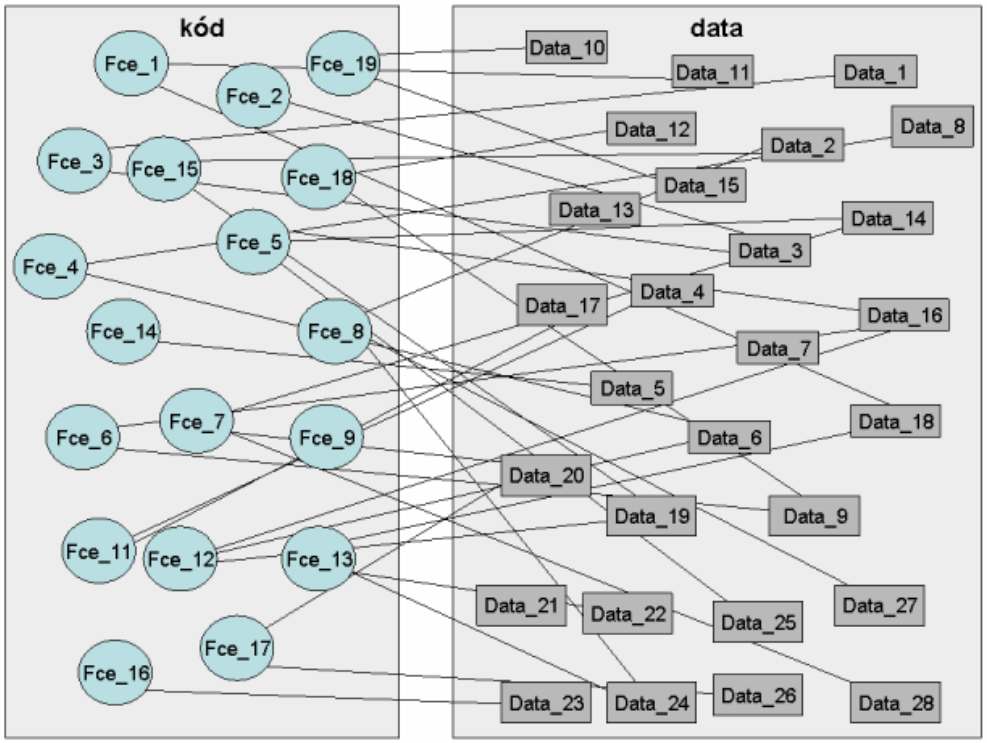
Definujme si kód: tyto prvky čtou a mění data; jsou to výkonné části informačního systému, provádějí nějakou činnost; mohou to být např. binární programy (či jejich části - moduly, funkce, procedury,...), skripty, triggery.

CHARAKTERISTIKA DAT

Definujme si data: jak proměnné (lokální či globální) držené jen v paměti (po vypnutí počítače se jejich obsah, ale vlastně i jejich samotná existence "ztratí"), tak perzistentní data (soubory, řádky databázových tabulek, apod.)

Kód pracuje s daty

Charakteristický obrázek 1 - část systému, kde kód pracuje s daty ve strukturovaném přístupu návrhu a naprogramování software:



Obrázek 1: Strukturální pojetí - kód a data, zdroj: http://mpavus.wz.cz

Tato modelová situace, kdy kód pracuje s daty, přináší různé problémy. Ukážeme si některé z těchto problémů, s jejichž řešením nám může pomoci objektový přístup.

Při vývoji IS musíme provádět transformaci požadavků do software, resp. do zdrojového kódu, který pracuje s daty. Nestačí nám prostě namodelovat realitu, ale navíc ji musíme rozdělit na oblast dat a na oblast kódu. Tato transformace nás stojí jisté úsilí, a důsledky této transformace se "nesou" celým dalším vývojem a údržbou IS.

Problémy ve strukturovaném přístupu programování

Tento přístup a postup nám ve fázi údržby a následného rozvoje může způsobit daleko větší obtíže, než které jsme si prožili při prvotním vytvoření IS.

Vznikají problémy při údržbě a dalším rozvoji IS, pokud chceme přidat/změnit funkcionalitu. Nelze často prakticky zjistit, která data jsou aktualizována kterou funkcí. Navíc daná funkce může vyvolávat další funkce a ty mohou spouštět další funkce, a až v těchto volaných funkcích může být pracováno s určitou oblastí dat. Toto čtení/měnění se může provádět jen za určitých podmínek, jejichž vyhodnocení může být velmi komplikované. Vzniká tak problém při údržbě a dalším rozvoji IS, jak jednoduše zajistit, aby obsah dat byl validní.

A další problémy vznikají při ladění a testování programového kódu s lokálními daty (předávanými jako parametr) a globálními daty (měnitelná i bez jejich předávání). V souvislosti s tím vzniká další otázka, jak dokonale ladit a testovat jednotlivé funkce systému, resp. jednotlivé situace v systému, které mohou nastat?

Další problémy přináší znovu-použitelnost (re-use).

K ZAPAMATOVÁNÍ

Na základě výše uvedených problémů při strukturálním přístupu byla vyvinuta IT technologie „Objektově orientovaná analýza a design“ pro tvorbu software/informačních systémů.

1.2 Objekt

DEFINICE

Objekt jako seskupení dat a funkcionality

Objekt je základní abstraktní jednotkou používanou v objektovém modelování. **Objekt je seskupením dat a funkcionality**, které jsou spolu spojeny za účelem plnění soudržné množiny zodpovědností. Objekt má svou identitu, vlastnosti, chování a zodpovědnost. [Kan2004]

Objekt je uzavřený, lokální data jsou obalena metodami, ale ani metody nejsou zvenčí přístupné - jediný způsob, jak použít objekt či jak s ním komunikovat, je poslat objektu

zprávu. Po zaslání zprávy objekt spustí svou metodu, která může použít atributy objektu a další metody objektu, může také poslat zprávu jinému objektu.

Základní pojmy charakterizující objekt byly podrobně popsány v elektronickém učebním textu, viz literatura [Fra2014], proto zde uvádíme jen jejich přehled s odkazem na elearningový kurz k předmětu Objektově orientované modelování:

Základní pojmy charakterizující objekt

- Zapouzdření
- Metody (chování)
- Atributy (lokální data)
- Odkaz na jiný objekt: pokud objekt zná odkaz na jiný objekt, může mu poslat zprávu.
- Zasílání zpráv

SAMOSTATNÝ ÚKOL

Prostudujte důkladně definici objektu a jeho základní vlastnosti a porovnejte, jakým způsobem popisují různí autoři, viz literatura tohoto učebního textu.

ÚKOL K ZAMYŠLENÍ

Dva příklady k objektovému myšlení: Objekt je černá skříňka a objekt jako HW, Komponenty, upraveno a inspirováno podle: http://mpavus.wz.cz/oo/

OBJEKT JE BLACK-BOX:

Mějme „black-box“ s několika tlačítky a kontrolkami. Může to být mobil, televize, automobil, počítač, mikrovlnka. Pro použití některého z objektů potřebujeme vědět, jaké má tento objekt chování a jaké tlačítko mám zmáčknout, aby se objekt zachoval tak, jak právě potřebujeme.

V případě mobilu: abych ho uměl zapnout, abych dovedl vytočit číslo, přijmout hovor, uložit své kontakty, abych dovedl zesílit či ztlumit hlasitost. Naprosto mě nezajímá, co je uvnitř. To znamená, že pokud požádám například zmáčknutím příslušného tlačítka o zobrazení kontaktů, mobil to provede. Mobil použije po obdržení zprávy (zmáčknutím tlačítka)

jednu nebo více metod, případně požádá o služby další objekty, které jsou připraveny, tak zvaně „zabaleny“, uvnitř v mobilu.

OBJEKT JAKO HW:

Komponenty personálních počítačů mají výše uvedené objektové vlastnosti. Jednotlivé komponenty (např. paměť, grafická karta, pevný disk, procesor ...) mají chování jako objekty:

Po správném složení a zapojení personálního počítače tyto objekty spolu spolupracují: Jeden objekt požaduje po jiném objektu, resp. jedna komponenta žádá jinou komponentu, o provedení služeb, které tento objekt poskytuje. Výsledkem spolupráce všech objektů je zajištění správné fungování systému. Pokud komponenty počítače správně spolupracují, je to funkční PC.

KOMPONENTY

Komponenty mají vnitřní paměť, vnitřní metody a lze je skládat. Komponenty jsou podrobně popsány v předchozím elektronickém skriptu [FRA2014], které je nedílnou součástí e-learningového kurzu předmětu „Objektové metody modelování“.

1.2.1 TŘÍDA

DEFINICE

Df

Třída reprezentuje šablonu pro objekty a popisuje interní strukturu těchto objektů.

Třída objektů je definována svými atributy a metodami

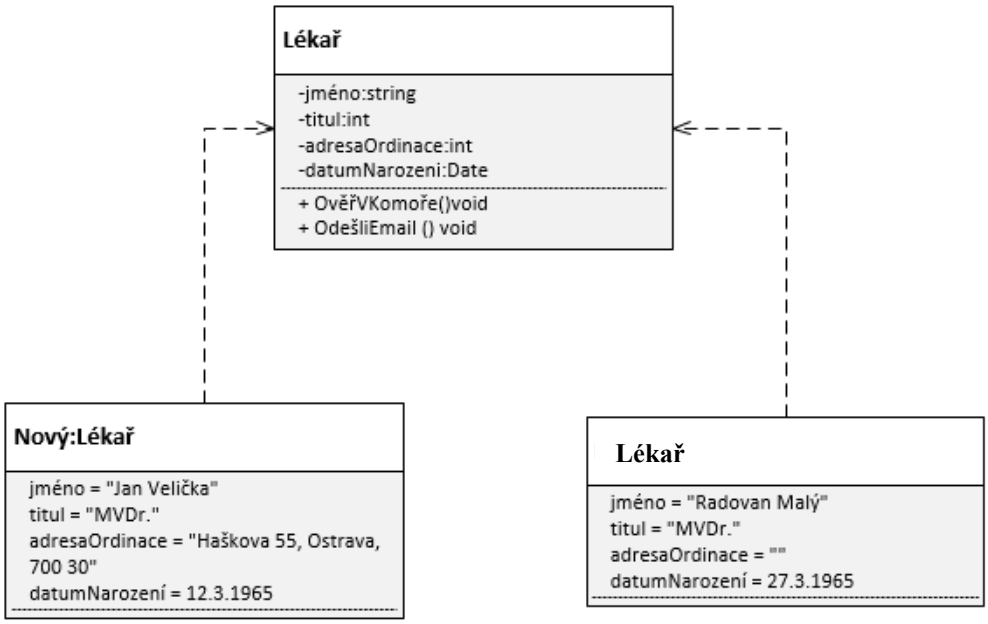
Pro návrh objektových systémů se předpokládá, že je třeba navrhnout model tříd objektů (Class model), který v podstatě nezobrazuje jednotlivé objekty, ale šablonu (předpis) pro vytvoření objektů - to je třída objektů. Třída objektů je definována svými atributy a metodami. Při návrhu třídy neuvažujeme o konkrétním naplnění atributů, pouze definujeme jejich název a typ. Při vzniku instance objektu (skutečný objekt) se atributům přiřadí skutečné hodnoty.

Diagramy tříd zobrazují statickou stránku systému

Diagramy tříd zobrazují statickou stránku systému, především vztahy mezi třídami. Vztahy, které jednotlivé třídy navzájem pojí, jsou asociace, agregace, kompozice, specializace/generalizace. Podřízenost jednoho objektu vůči druhému je v analýze chápána dvojím způsobem, buď jako agregace, nebo jako kompozice. V obou případech se však jedná o vztah dvou objektů, z nichž podřízený objekt má svou objektovou referenci vloženu do prvního objektu. K detailnímu vysvětlení jednotlivých typů vazeb se dostaneme v následujícím textu.[Kan2004]

ŘEŠENÁ ÚLOHA

Příklad: V informačním systému evidujícím lékaře máme třídu Lékař s atributy uchovávajícími informace o lékaři (jméno, adresa, atd.) a se dvěma metodami - ty umí ověřit členství lékaře v lékařské komoře a odeslat lékaři e-mail. Na následujícím obrázku 2 je znázorněna třída lékař se dvěma vytvořenými objekty nový lékař (právě zaváděný do systému) a hlavní lékař ČR.



Obrázek 2 Třída s dvěma vytvořenými objekty (instancemi třídy), zdroj vlastní

1.2.2 STRUKTURA TŘÍD

Dva principy, na kterých je třída založena

Struktura tříd je založena na dvou principech, na zodpovědnosti třídy a na zapouzdření třídy. Zopakujme si, co si pod těmito pojmy představujeme. Zodpovědnost třídy je jedním z klíčových faktorů objektově orientované analýzy a návrhu a znamená, že námi definovaný objekt nese zodpovědnost za danou problematiku (tj. „to o čem uchovává informace a chování") a žádný jiný objekt se nesmí „plést do této zodpovědnosti".

1.2.3 ZOBECNĚNÍ (GENERALIZACE-SPECIALIZACE), DĚDĚNÍ (INHERITANCE)

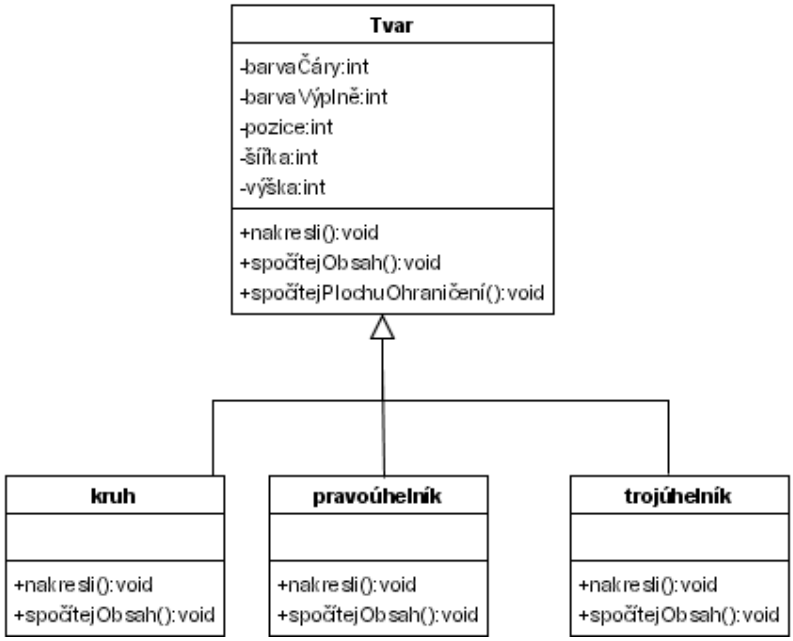
Zobecnění nám umožňuje další stupeň re-use, resp. znovu-použitelnost nikoliv ve vztahu třída a několik jejích instancí, ale třída a od ní několik odvozených tříd.

Pokud namodelujeme několik tříd, které jsou si velmi podobné, mohlo by se jednat o situaci, kde je vhodné použít zobecnění.

ŘEŠENÁ ÚLOHA

Vysvětlení pojmu zobecnění na příkladu

Klasický příklad pro znázornění zobecnění: obecnější třída (předek, nadtřída, bázová třída, předchůdce) se jmenuje tvar a zastupuje nějaký, blíže neurčený dvourozměrný tvar. Tvar má svou barvu čáry, barvu výplně, pozici (pozice těžiště), šířku a výšku, dále má metody nakresli (nakreslí tvar), spočítejObsah (spočítá plošný obsah) a spočítejPlochuOhraničení (spočítá plochu pravoúhelníku ohraničujícího tvar), viz obrázek 3.



Obrázek 3 Třída a podtřídy, zdroj http://mpavus.wz.cz/oo/

Vysvětlení pojmů dědění a specializace na příkladu

Podtřídy (potomci) jsou specializacemi nadtřídy -zde máme kruh, pravoúhelník, trojúhelník. Všude (tj. v jiných diagramech, v kódu,....), kde použijeme instanci (objekt) třídy tvar můžeme použít instanci libovolného jeho potomka. Říkáme, že "potomek je druhem předka" tj. kruh je druhem tvaru, trojúhelník je druhem tvaru - pokud si tuto větu říct nemůžeme, pravděpodobně je ve stromu zobecnění nějaká chyba (opakovaná chyba v učebnicích: předek je bod, potomek je úsečka, přímka, čtverec, kruh,... - zde si nemůžeme říct čtverec je druhem bodu).

Vztahy generalizace-specializace se v konkrétním prostředí realizují pomocí techniky dědění (inheritance). Potomci dědí:

- atributy,
- metody,
- relace,
- omezení.

Potomci mohou ke zděděnému přidávat to svoje (tj. atributy, metody, relace a omezení), dokonce můžou zděděné metody předefinovat (viz níže výklad pro abstraktní metody). Tj. kruh, pravoúhelník a čtyřúhelník zdědí od předka tvar všechny atributy (barvu čáry, šířku a výšku, atd.), dále zdědí metodu a její kód, např. „Spočítej plochu ohraničení“ a předefinují metody „Nakresli“ a spočítej obsah.

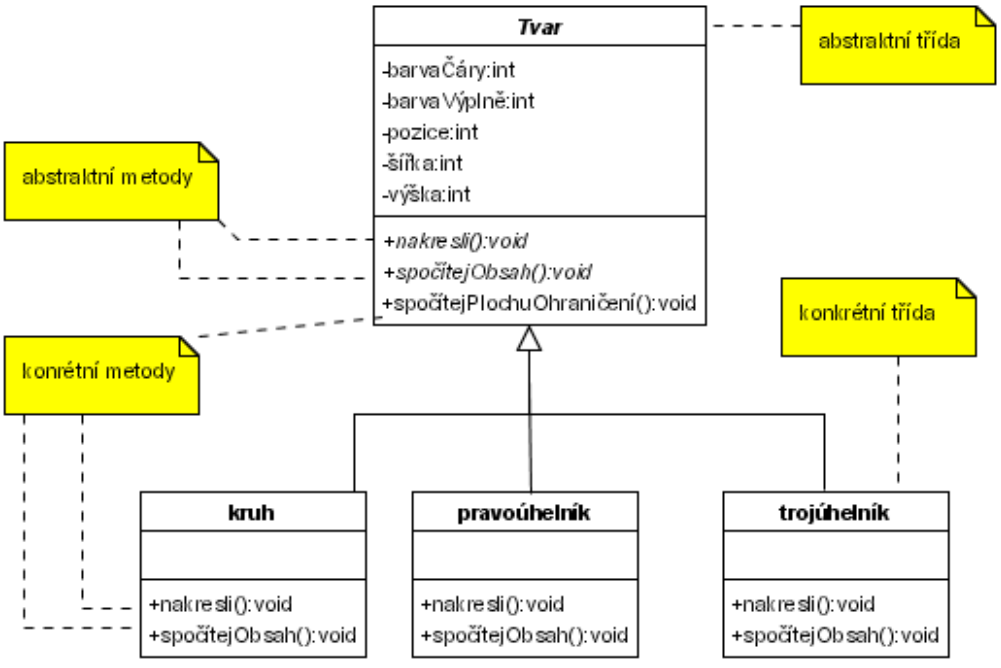
1.2.4 ABSTRAKTNÍ METODY A TŘÍDY

DEFINICE Df

Metodě, jejíž implementace v předkovi úplně chybí, říkáme abstraktní metoda. Třídě, která má alespoň jednu abstraktní metodu, říkáme abstraktní třída.

Vysvětlení pojmu abstraktní třída na příkladu

V předchozím příkladu jsme si ukázali, že metody předka „spočítejObsah()“ a „nakresli()“ musí být v potomcích předefinovány. Pokud všichni potomci překrývají metodu po svém, tak ani nemá smysl v předkovi tuto metodu implementovat - chceme ji ale přesto v předkovi uvést (můžou nás k tomu vést například důvody uvedené na konci předchozího odstavce). Takovýmto metodám bez implementace říkáme abstraktní metody. Ta třída, která má alespoň jednu abstraktní metodu, není nadefinována do té míry, abychom mohli vytvořit instanci této třídy (abstraktní metoda této třídy je prázdná). Takovéto třídě říkáme abstraktní třída. Metodě, která není abstraktní, pak říkáme konkrétní metoda, třídě, která není abstraktní, pak říkáme konkrétní třída. Předchozí obrázek tedy můžeme zakreslit s využitím notace UML, kde se abstraktní metody a třídy označují kurzívou jako obrázek 4.



Obrázek 4 Abstraktní třídy a metody, zdroj http://mpavus.wz.cz/oo/

1.2.5 POLYMORFISMUS

Pojem polymorfizmu se někdy zbytečně zdá být komplikovaný a těžko pochopitelný, ale jestliže uvažujeme objektově a máme určitou praxi s objektovým modelováním, není tento pojem složitý k pochopení. Polymorfismus znamená mnohotvárnost.

DEFINICE

Podle [Arl2007] jsou polymorfní operace takové operace, které mají mnoho implementací.

Polymorfní operace a metody

Polymorfní metody jsou takové metody, které se vyskytují u více objektů (tj. mají u různých objektů stejnou signaturu), ale mají odlišnou implementaci, dle mpavus.wz.cz/oo/oo-trida-4.php.

ŘEŠENÁ ÚLOHA

Vysvětlení pojmu polymorfní metody na příkladu

Příklad: Máme metody „spočítejObsah()“ a „nakresli()“. V konkrétních třídách mají tyto metody stejnou signaturu, avšak v každé třídě je jiná implementace: metoda „nakresli()“ spuštěná v objektu třídy kruh bude kreslit kruh, metoda „nakresli()“ spuštěná v objektu třídy pravoúhelník bude kreslit čtverec nebo obdélník. Tyto metody jsou tedy polymorfní - v různých třídách mají různé chování.

1.2.6 ZAPOUZDŘENÍ

Softwarové objekty sdílí stejný koncept jako objekty z reálného světa: taktéž obsahují stavy a příbuzné akce. Objekt ukládá ve vlastnostech své stavy a přes metody nabízí příslušné akce. Metody operují nad interními stavy objektu a jsou určeny k prvotnímu způsobu komunikace mezi dvěma objekty.

DEFINICE

Základní kámen OOP je zapouzdření

Skrývání interních stavů a jejich nabízení prostřednictvím metod se nazývá zapouzdření dat – jeden ze základních kamenů objektově orientovaného programování. Viz http://programujte.com/clanek/2007043001-java-tutorial-objektove-orientovane-programovani-3-dil/

SHRNUTÍ KAPITOLY

V této kapitole je čtenář seznámen se základními pojmy OOM, je to stručný úvod do objektově orientovaného modelování. Výklad, vymezení a pochopení těchto pojmů je nezbytnou podmínkou pro porozumění dalších kapitol tohoto učebního textu.

OTÁZKY

Co je to třída?

Vyberte jednu z nabízených možností:

a. Třída je totéž co objekt

b. Třída je to, co vzniká z objektu při běhu aplikace

c. Třída je šablona - předpis pro vytvoření objektů

Jaká je správná definice objektu?

Vyberte jednu z nabízených možností:

a. Objekt zajišťuje jen funkcionalitu prvku v informačním systému

b. Objekt je seskupení dat a funkcionality

c. Objekt reprezentuje data pro subjekt v databázi

Při dědění tříd se dědí:

Vyberte jednu z nabízených možností:

a. Atributy i metody

b. Jen atributy

c. Jen metody

Abstraktní metoda u třídy je:

Vyberte jednu z nabízených možností:

a. Metoda, jejíž implementace je v předkovi úplně obsažena

b. Metoda, jejíž implementace v předkovi úplně chybí

c. Metoda, jejíž implementace je v předkovi obsažena částečně

Polymorfní metoda je:

Vyberte jednu z nabízených možností:

a. Taková metoda, která se vyskytuje u více objektů, ale má stejnou implementaci

b. Taková metoda, která se vyskytuje u nadřízeného objektu, ale má odlišnou implementaci u podřízeného objektu

c. Taková metoda, která se vyskytuje u více objektů, ale má odlišnou implementaci

ODPOVĚDI

c. b. a. b. c.

2 ZÁKLADNÍ ELEMENTY JAZYKA UML

RYCHLÝ NÁHLED KAPITOLY

V této kapitole si objasníme principy jazyka UML. Modelovací jazyk UML je souhrnem grafických notací k vyjádření analytických a návrhových modelů. UML je jazyk, který dává předpoklady a možnosti pro modelování jednoduchých i složitých aplikací s využitím stejné formální syntaxe.

CÍLE KAPITOLY

Cílem kapitoly je objasnit základní pojmy jazyka UML a k čemu tyto diagramy slouží.

ČAS POTŘEBNÝ KE STUDIU

60 minut.

KLÍČOVÁ SLOVA KAPITOLY

UML, Diagramy, metodika RUP, modelování.

2.1 Princip jazyka UML

Charakteristika jazyka UML

Modelovací jazyk UML je souborem grafických notací k vytváření analytických a návrhových modelů informačních systémů. UML dobře poslouží při vytváření jak jednoduchých, tak i složitých aplikací, přitom využívá stejnou formální syntaxi. Je ideálním prostředkem pro sdílení pracovních postupů pro vývojáře, testery, ale i zadavatele a uživatele systému. Slouží rovněž k objasňování požadavků uživatelů na vytvářený systém. Diagramy UML se soustředí vždy na právě jeden pohled na vyvíjený systém. Navrhovaný systém zachycuje celek komplexem jednotlivých diagramů UML.

UML je jazyk pro vizualizaci, specifikaci, stavbu a dokumentaci software zajišťující chod informačních systémů. Používané modelovací techniky s využitím jazyka UML podporují nejznámějšími metodiku UP (Unified Process) a RUP (Rational Unified Process) firmy Rational (nyní ve vlastnictví IBM) a Select Business Solution, viz např. [Kan004].

DEFINICE

UML = Unified Modeling Language (tj. unifikovaný modelovací jazyk)

Unifikovaný: unifikuje Booch, OMT a Objectory modelovací jazyky

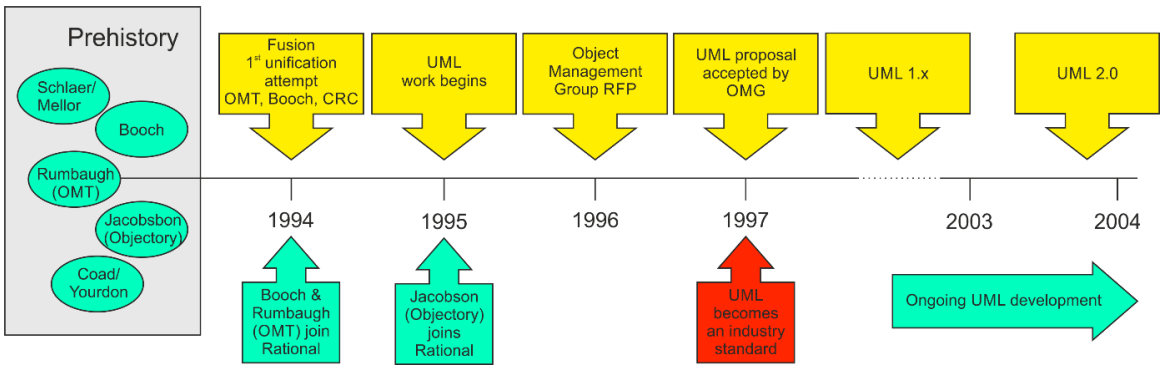
Modelovací: UML je jazyk pro specifikaci, vizualizaci, konstrukci a dokumentaci artefaktů SW systémů.

UML je využitelný i pro business modelování i pro modelování nesoftwarových systémů. V UML lze modelovat jakýkoliv typ aplikace běžící na jakémkoliv typu a kombinaci HW, OS, programovacím jazyku a sítě. Lze modelovat distribuované aplikace.

UML není programovací jazyk, ale je to jazyk, neboť má jasně definovanou syntaxi a sémantiku.

2.2 Historie vzniku jazyka UML

Historie vzniku UML je přehledně zachycena na obrázku 5.



Obrázek 5: Historie vzniku UML, zdroj: [Arl2008)]

Přehled milníků vzniku a tvůrců jazyka UML je uveden a podrobně popsán v [FRA2014].

2.3 UML – skladba

Skladba definuje z jakých *základních prvků* se UML skládá.

Tyto základní prvky jsou tři:

1) **Předměty (things)**, tj. elementy modelu definující:

- **strukturní abstrakce (structural things)** : podstatná jména (modelu UML), tj. elementy: třída (class), rozhraní (interface), spolupráce (collaboration), případ užití (use-case), aktivní třída (active class), komponenta (component), uzel (node),
- **chování (behavioral things)** : slovesa (modelu UML), tj.: interakce (vztahují se ke komunikaci mezi objekty - např. zprávy (messages), spoje (links)) a stavový stroj (specifikuje sekvenci stavů objektu pomocí stavů, přechodů, událostí a aktivit),
- **seskupení (grouping things)** : balíčky (packages) seskupující (dle potřeby) prvky modelu,
- **poznámky (anotational things)** : poznámky s přidanými informacemi.

2) **Relace (relationships)**, tj. elementy modelu spojující spolu dva (či více) předmětů - **strukturních abstrakcí** (také mohou spojovat **seskupení**); jejich typy jsou:

- **závislost (dependency)** : znázornění vztahu, kdy změnou v jednom elementu je ovlivněn jiný (závislý) element.
- **asociace (association)** : spojení definující vztah mezi elementy,
- **zobecnění (generalization)** : vztahy generalizace-specializace, tj. jeden element je specializací jiného elementu,
- **realizace (realization)** : jeden element je realizací jiného elementu.

3) **Diagramy**: model je souhrn všech předmětů a relací, jejichž znázornění v jednom obrázku by bylo nepřehledné, mnohdy zcela nemožné. Naproti tomu diagram je jeden *pohled, okénko*, kterým se díváme na model.

   Rozdělení základních diagramů do dvou skupin:

   Statický model (zaměřený na systémovou strukturu) :

   - diagram tříd
   - diagram komponent
   - diagram nasazení

Dynamický model (zaměřený na chování systému) :

- diagram případu použití
- sekvenční diagram
- diagram spolupráce
- stavový diagram
- diagram aktivit
- objektový diagram

Zpracováno podle [Arl2007] a http://mpavus.wz.cz/uml/uml-skladba-2.php

2.4 UML – mechanismy

UML má čtyři mechanismy, které se prolínají celým jazykem. Tyto mechanismy jsou čtyři.

1) Specifikace: každý element může (či měl by) být specifikován textem, který popisuje sémantiku tohoto elementu. Tato specifikace upřesňuje, blíže popisuje, udává smysl modelovaného elementu. Popisuje business pravidla elementů (tudíž má největší význam u elementů popisujících problémovou doménu).

2) Ozdoby (adornments): další informace známé o elementu modelu. Každý element může být zadán jednoduchým tvarem, ale je možno přidávat k němu i další informace - ozdoby. Proč je těchto ozdob u elementu zobrazeno někdy více a někdy méně: postupně vytváříme model: zpočátku máme málo informací, které postupně doplňujeme

3) Podskupiny (common division) : udávají, jak je možno rozdělovat (seskupovat) jednotlivé elementy; první způsob dělení:

- **klasifikátor a instance**: pro dva elementy UML objekt a třída platí, že objekt je **instance**, kdežto třída je **klasifikátor**. Podobný vztah klasifikátor-instance lze nalézt pro další elementy UML. Každý element je buď klasifikátor anebo instance. Toto rozlišení je velmi důležité. Osvojením tohoto dělení si usnadníte komunikace mezi členy týmu. Můžeme se s tímto setkat i v CASE nástrojích a v literatuře,
- **rozhraní a implementace**: v pasáži o objektu objekt jsme se zmínili o **protokolu zpráv**, což je rozhraní objektu. Implementace pak jsou metody, které *řeší*, implementují, toto rozhraní.

4) Mechanismy rozšiřitelnosti: jazyk UML sám v sobě obsahuje připravené mechanismy umožňující rozšířit jazyk tak, aby vyhovoval momentálním potřebám. Máme k dispozici tři mechanismy rozšiřitelnosti:

- **omezení (constraints)**
- **stereotypy (stereotypes)**
- **označené hodnoty (tagged values)**

Zpracováno podle [Arl2008].

SAMOSTATNÝ ÚKOL

Důkladně promyslete a zapamatujte si výše uvedené základní prvky a mechanizmy jazyka UML. Prostudujte podrobnější popisy těchto pojmů v [Arl2007] a v [Fra2014]. Konfrontujte s popisy na Internetu.

SHRNUTÍ KAPITOLY

Tato kapitola se věnovala základním pojmům jazyka UML. Naučili jste se k čemu UML slouží a z jakých předpokladů vychází. Tyto základní pojmy je nutno se naučit a pochopit jejich význam. Tyto pojmy se pochopí lépe při druhém čtení po prostudování dalších kapitol a zejména příkladů využití jazyka UML. Proto autor doporučuje se k této kapitole znovu vrátit.

OTÁZKY

Základní rozděleni UML diagramů je:

Vyberte jednu z nabízených možností:

a. Komponenty a správa modulů

b. Statická struktura a dynamické chování

c. Aktivity a nasazení

Co znamená zkratka UML

Vyberte jednu z nabízených možností:

a. Universal Modeling Language

b. Unified Modeling Language

c. Unified Modal Language

Co je to UML?

Vyberte jednu z nabízených možností:

a. UML je souhrnem především grafických notací k vyjádření analytických a návrhových modelů software

b. UML umožňuje především programovat SW

c. UML je jazyk především pro vytváření dokumentace SW

Co umožňuje UML?

Vyberte jednu z nabízených možností:

a. UML umožňuje popsat objektovou analýzu a návrh SW

b. UML umožňuje modelovat testování SW

c. UML umožňuje navrhovat a spravovat uživatelské požadavky v SW

Je správa požadavků součástí UML?

Vyberte jednu z nabízených možností:

a. Částečně

b. ANO

c. NE

Do diagramů pro modelování struktury patří

Vyberte jednu z nabízených možností:

a. Sekvenční diagram

b. Diagram tříd

c. Diagram případu užití

Do diagramů pro modelování chování patří

Vyberte jednu z nabízených možností:

a. Diagram nasazení

b. Diagram aktivit

c. Diagram balíčků

ODPOVĚDI

b. b. a. a. c. b. b.

---

3 POPIS JAZYKA UML – USE CASE

RYCHLÝ NÁHLED KAPITOLY

Tato kapitola se zabývá tzv. případy užití. Případy užití, typové úlohy nebo chcete-li užitné případy. Se všemi těmito překlady originálu „**Use Case“** jsou v odborné literatuře používány. V následujícím textu se přikloníme k terminologii „případ užití“, který odpovídá anglickému originálu.

CÍLE KAPITOLY

Cílem kapitoly je objasnit k čemu případy užití přesně slouží. Případy užití popisují přesně funkcionalitu vytvářeného informačního systému a vymezují tím jednoznačně rozsah prací, které je nutno vykonat od zjištění požadavků, přes analýzu, programování. Testování a uvedení do provozu. Každý případ užití popisuje jeden ze způsobů chování systému, jednu jeho požadovanou funkčnost.

ČAS POTŘEBNÝ KE STUDIU

60 minut

KLÍČOVÁ SLOVA KAPITOLY

Use Case, Aktér, Případ užití, Scénář.

3.1 Úvod do problematiky USE CASE

Případy užití, typové úlohy nebo také užitné případy. Se všemi těmito překlady originálu „Use Case“ se setkáváme v odborné literatuře setkat. V následujícím textu se přikloníme k terminologii „případ užití“, který odpovídá anglickému originálu.

Vývojáři bez ohledu na svou orientaci na vývojové prostředí se již dlouhou dobu snaží modelovat typické interakce uživatelů se systémy, aby co nejlépe pochopili skutečné požadavky uživatelů na budoucí systém a zároveň aby vymezili rozsah navrhované aplikace.

Případy užití popisují funkčnost a vymezují rozsah prací IS

Případy užití zachycují přesně funkčnost, která bude budoucím informačním systémem pokryta a vymezují tak jednoznačně rozsah prací. Každý případ popisuje jeden ze způsobů systému, popisuje tedy jednu jeho požadovanou funkčnost. Pro co nejnázornější výklad pojmu USE CASE a pro výklad UML diagramů v dalších kapitolách si popíšeme konkrétní modelový případ návrhu části informačního systému „Objednávka na e-shopu“. Vytvoříme tzv. případovou studii, kdy budeme navrhovat část elektronického objednávání zboží, zkráceně e-shop“.

3.2 Případová studie USE CASE

Text tohoto výukového materiálu provází případová studie „Objednávka na e-Shopu“. V této kapitole si objasníme celou problémovou oblast vytváření „USE CASE“ diagramu a k němu příslušné scénáře.

Cílem bylo demonstrovat praktické použití UML na konkrétním projektu s cílem zjednodušit a zprůhlednit ukázky v této případové studii. Tato studie řeší pouze část funkcionality nakupování po internetu – e-shopu. Neřeší modelování kompletního e-shopu pro složitost a rozsáhlost problematiky, která by způsobila nepřehlednost analýzy. Z tohoto důvodu byla ponechána stranou i důležitá část informačního systému a to odstraňování a archivace dat.

Ukázky by měly demonstrovat základní metody UML a tomu je přizpůsobena složitost případové studie i uvedených příkladů.

Cílem není dokonalá analýza dané problematiky, ale demonstrace technik UML, a proto na některých místech učebního textu jsou uvedeny i ukázky, které nepochází z modelové studie.

PŘÍPADOVÁ STUDIE

Příklad z praxe případové studie

Případová studie předpokládá modelovou situaci, kdy softwarová firma získala zakázku na analýzu, návrh a vývoj informačního systému internetové aplikace e-shop. Představme si tedy situaci, kdy existuje poptávka po vytvoření aplikace e-shop a softwarová firma má za úkol tuto aplikaci navrhnout a vytvořit. To sebou nese celou řadu problémů týkající se oblasti oborů marketingu a projektového managementu. My se zaměříme pouze na část e-

shopu, a to na návrh objednávkového systému na internetu z pohledu analýzy při návrhu software. Pro jednoduchost budeme v popisované modelové studií uvažovat pouze o jednom modulu informačního systému, a to modulu e-shopu pro objednávání zboží zákazníkem. Záležitosti ekonomické (účetnictví, fakturace) a další funkce e-shopu jako jsou například reklamní kampaně, hodnocení efektivity a další funkce, nejsou předmětem případové studie.

Vedoucí pracovníci softwarové firmy absolvovali několik úvodních jednání a porad s analytiky, kteří mají navrhnout aplikaci e-shop a připravit ji k programování. Výsledky těchto jednání byly sumarizovány do podoby uživatelských požadavků a představy o fungování budoucí aplikace.

Pro pochopení souvislostí uvedeme nyní seznam požadavků na funkce e-shopu:

1) Zadávání a evidence zboží s funkcemi

Základními údaji jsou název, kód a cena a popis zboží pomocí HTML editoru. Definice cen poskytuje možnost vytváření až 30 cenových skupin, možnosti vytváření variant zboží, zboží je možné vložit do různých kategorií, samozřejmě lze vkládat neomezený počet fotografií, či jiných příloh, vestavěný export a import pomocí CSV, XML).

2) Definice způsobů dopravy a platby

V e-shopu budou pokryty všechny obvyklé způsoby platby – platba bankovní kartou, hotovostní platba při odběru zboží, platba převodním příkazem a splátkový prodej.

3) Zákazníci, zvýhodněné ceny, cenové skupiny

Uživatelé nakupující v e-shopu mají možnost volby, zda se zaregistrují nebo ne. Lze nakoupit i bez registrace. Každému zaregistrovanému uživateli lze nastavit slevu v procentech i cenovou skupinu, podle obratu zákazníka.

4) Statistiky

Interní statistiky návštěvnosti, historie pageranku, + napojení na Google Analytics, Google pro Webmastery, Toplist, Navrcholu, statistiky objednávek, obratů, statistiky prodeje a návštěvnosti pro jednotlivé produkty, napojení pro měření konverzí pro libovolné systémy.

5) Vytvoření rozhraní pro napojení ekonomického systému

V pěti bodech je shrnuta celá funkcionalita e-shopu. Pro naše účely vysvětlení pojmů a diagramů modelovacího jazyka UML se dále budeme zabývat jen objednáváním zboží na e-shopu z pohledu zákazníka.

ROLE V PŘÍPADOVÉ STUDII E-SHOP OBJEDNÁVKY ZÁKAZNÍKA

1) Zákazník (registrovaný a neregistrovaný)

Tři role v případové studii

Neregistrovaný zákazník má volný vstup na stránky e-shopu a není omezen z pohledu tvorby objednávky, ale pouze její modifikací. Tento typ uživatele může zboží vyhledávat, prohlížet, popřípadě přidávat do košíku, vyplnit a následně odeslat objednávku na zboží. Nemá nárok na většinu slev.

Registrovaný zákazník má možnost spravovat svůj účet, kde si může prohlížet svou historii objednávek, modifikovat objednávku před její expedicí a hlavně možnost uplatňovat slevy dle obratu.

Systém pro elektronickou platbu: příjme zákazníkův požadavek platit a prostřednictvím vybrané služby (kreditní karta, paypal) mu to umožní.

Systém pro registraci: slouží pro registraci, Přihlášeni a ověření totožnosti zákazníka.

2) E-shop (systém)

Provádí případnou registraci návštěvníka. Zajišťuje přístup ke katalogům zboží. Slouží pro zobrazování požadavků zákazníka, provází jej nákupem a odkáže ho v případě zájmu na platbu. Bude vytvořeno uživatelské prostředí, které napomáhá k jednoduššímu nákupu zboží.

3) Platba (systém)

příjme zákazníkův požadavek platit a prostřednictvím vybrané služby (kreditní karta, hotovost, paypal).

DIAGRAM PŘÍPADU UŽITÍ V PŘÍPADOVÉ STUDII

V diagramu případu užití jsou zobrazeny 3 aktéři: Neregistrovaný, Registrovaný a E-shop (systém).

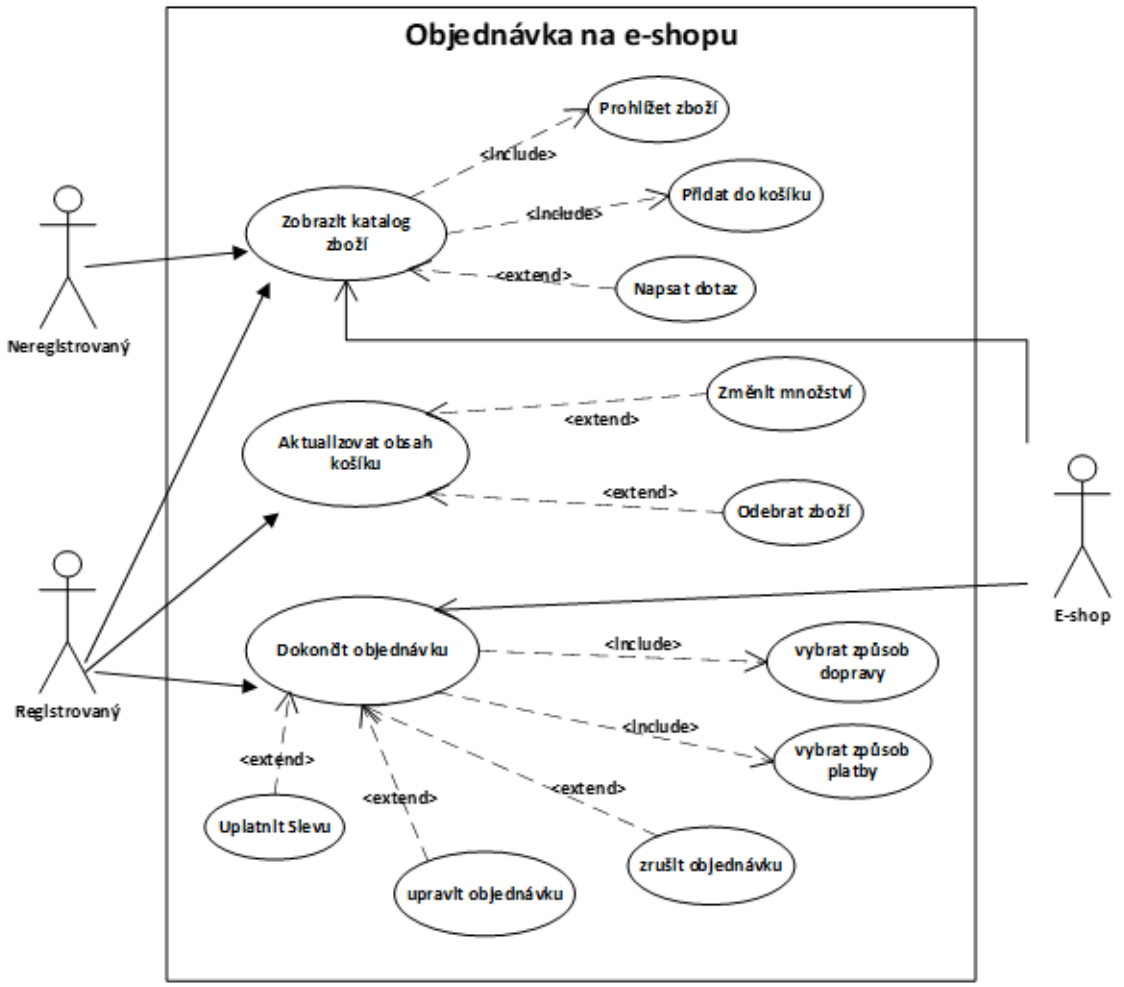
Neregistrovaný:

Nemá povinnost se registrovat do systému, ale může objednávat zboží. Jeho nevýhodou je nemožnost uplatnění slevového programu a při každém vstupu na portál e-shopu vyplnit kontaktní údaje.

Registrovaný:

Objednává zboží, na které může uplatnit slevový program dle obratu, může sledovat svou historii nákupů.

E-shop (systém): Zajišťuje vstup zákazníků na portál e-shopu, registraci zákazníka a přihlášení zákazníka. Provází zákazníka při tvorbě objednávky, zajišťuje požadavek na způsob placení.



Obrázek 6: Objednávka na e-shopu, příklad případu užití, zdroj vlastní

3.3 Scénáře v případech užití

Scénář se skládá z postupných kroků, které zachycují vzájemné akce mezi uživatelem (aktér-em) a obrazovkou (systémem)

Případy užití jsou základním elementem při plánování, vývoji a realizaci projektu – vytváření informačního systému. Co znamená pojem **scénář případu užití**? Scénář se skládá z postupných kroků, které zachycují vzájemné akce mezi uživatelem (aktérem) a obrazovkou (systémem). V systému z případové studie „Objednávky v e-shopu“, můžeme vytvořit scénář, viz tabulka 1:

Krok	Role	Akce
1	Zákazník	Zadání www adresy e-shopu vstoupí na úvodní stránku e-shopu.
2	Zákazník	Zadáním přihlašovacích údajů se přihlásí se do systému.
3	Systém	Zkontroluje údaje a zobrazí přehledové informace o zákazníkovi a nabídne katalog.
4	Zákazník	Pomocí rolování nebo funkce „Vyhledat“ vyhledává vhodné zboží.
5	Systém	Ke zvolenému zboží zobrazí základní informace, alternativně podrobné informace, údaje o možné slevě, zobrazí obrázky ke zboží.
6	Zákazník	Studuje popisy a údaje ke zboží.
7	Zákazník	Ke zvolenému zboží zadá množství.
8	Systém	Zvolené zboží vloží do košíku a dotáže se se, zda chce dále pokračovat ve výběru nebo přejít k zaplacení.
9	Zákazník	Nakupující buď pokračuje ve výběru zboží (body 4 – 8) nebo se rozhodne zaplatit.
10	Systém	Zkontroluje košík, provede výpočty, vypočítá slevu dle obratu, sestaví návrh objednávky a požádá zákazníka o odsouhlasení objednávky
12	Zákazník	Potvrdí objednávku, případně provede úpravy v nákupním košíku a znovu požádá systém o kompletaci objednávky.
13	Systém	Nabídne zákazníkovi na obrazovce způsob platby a dopravy.
14	Zákazník	Zvolí si z příslušných možností metodu platby kartou a dopravu PPL.
15	Systém	Zaeviduje objednávku a požádá zákazníka o kontrolu a potvrzení
16	Zákazník	Potvrdí nebo opraví dodací údaje a potvrdí objednávku.
17	Systém	Objednávku zaeviduje a zašle e-mail s objednávkou a dalšími informacemi.
18	Systém	Zboží je zasláno zákazníkovi.

Tabulka 1: Scénáře případu užití registrovaný uživatel, který uplatňuje slevu a platí kreditní kartou, zdroj: vlastní

Krok	Role	Akce
1	Zákazník	Zadání www adresy e-shopu vstoupí na úvodní stránku e-shopu.
2	Zákazník	Hledá v katalogu zboží na e-shopu.
3	Systém	Ke zvolenému zboží zobrazí základní informace, alternativně podrobné informace, a zobrazí obrázky ke zboží.
4	Zákazník	Prohlíží údaje o zboží.

5	Zákazník	Zvolí si zboží a zadá požadované množství.
6	Systém	Vybranou položku s množstvím vloží do košíku a dotáže se, zda chce pokračovat ve výběru dalšího zboží nebo přejít k zaplacení..
7	Zákazník	Zákazník buď pokračuje ve výběru zboží (body 4 – 8) nebo se rozhodne zaplatit.
8	Systém	Zkontroluje košík, provede výpočty, sestaví návrh objednávky a požádá zákazníka o odsouhlasení objednávky.
9	Zákazník	Potvrdí objednávku, případně provede úpravy v nákupním košíku a znovu požádá systém o kompletaci objednávky..
10	Systém	Nabídne zákazníkovi na obrazovce způsob dopravy a platby.
11	Zákazník	Zvolí si metodu platby (například z účtu) a jak zboží dopravit.
12	Systém	Zaeviduje objednávku a požádá zákazníka o potvrzení a pokud došlo ke změně dodacích údajů o jejich opravu.
13	Zákazník	Potvrdí nebo opraví dodací údaje a potvrdí objednávku.
14	Systém	Zavede objednávku do databáze a potvrdí na e-mail zákazníka včetně údajů k platbě z účtu, tj. účet dodavatele a variabilní symbol.
15	Zákazník	Zaplatí za objednávku převodem na účet.
16	Systém	Počká na potvrzení došlé platby z ekonomického systému a pak odešle zboží zákazníkovi a informuje zákazníka mailem.

Tabulka 2 Scénář případu užití „neregistrovaný uživatel“, který platí přes bankovní účet, zdroj: vlastní

Stojí za povšimnutí, že takto zapsaný scénář případu užití je podobný se skutečnými scénáři filmů nebo divadelních her.

Dostáváme se k odpovědi na otázku „Co je to definici případu užití“. **Případ užití je soubor scénářů, které dohromady spojuje společný cíl.**

3.4 Popis případu užití

Jak má vypadat popis případu užití? Modelovací jazyk UML pro psaní scénářů nedefinuje žádný standard, existují však doporučení osvědčených v praxi:

- Název případu užití by měl být tvořen pomocí slovesné vazby (představuje nějakou akci v systému, tedy např. *nákup na e-shopu z pohledu zákazníka*.
- Pro zápis hlavního úspěšného scénáře použijte jednoduchou sekvenci číslovaných kroků, viz například tabulka 3.
- Pro zápis případných rozšíření v podobě alternativních scénářů použijte opět číslovanou sekvenci odkazující se na hlavní sekvenci.
- Používáme stručné a srozumitelné věty.

Použití těchto konstruktů ilustruje tabulka 3.

Krok	Role	Akce
1	Zákazník	ZAČÁTEK CYKLU Vybere zboží z nabízeného seznamu
2	Systém	Zobrazí formulář nabízeného zboží s podrobnou specifikací
3	Zákazník	POKUD Zákazník souhlasí s nabízenými parametry zboží, vloží zboží do nákupního košíku JINAK Přejde zpět na seznam zboží KONEC - POKUD
	Systém	Zobrazí seznam nabízeného zboží
5	Zákazník	Pokračuje ve výběru zboží KONEC CYKLU

Tabulka 3 Případ užití: Výběr zboží v e-shopu s uplatněním podmínky, zdroj: vlastní

3.5 Vstupní a výstupní podmínky

Pre-Condition a Post_Condition

Kromě uvedeného popisu případu užití formou scénářů je možné popis dále upřesnit pomocí přídavných sekcí. Takovými sekcemi mohou být **vstupní podmínky případu užití** (Pre-Conditions), definující předpoklady, které musí být splněny, aby případ užití mohl být zahájen, nebo naopak **výstupní podmínky** (Post-Conditions), určující kritéria, která musí být splněna po skončení případu užití.

Podrobnější popis vstupních a výstupních podmínek byl zpracován v předchozím elektronickém skriptu [Fra2014], které je k dispozici studentům předmětu „Objektové metody modelování“ na e-learning portále SU v rámci připraveného kurzu.

3.6 Postup tvorby popisu případu užití

K postupu tvorby popisu případu existuje v literatuře celá řada doporučení, doplnění a upřesnění.

SAMOSTATNÝ ÚKOL

Prostudujte si tato doporučení v literatuře, zejména v [Arl2007] a [Fra2014]. Konfrontujte nabyté poznatky se zdroji na Internetu.

SHRNUTÍ KAPITOLY

Kapitola se zabývá 1. diagramem UML, a to use case diagramem. Popisuje jeho prvky a vazby mezi nimi. Na příkladu vysvětluje jeho možnosti využití. Kapitola dává návod jakým způsobem postupovat při vytváření use-case a jak se vytváří vstupní a výstupní podmínky.

OTÁZKY

Který výrok o případy užití - USE CASE není pravdivý

a. Případy užití zachycují přesně funkčnost, která bude informačním systémem pokryta a vymezují tak jednoznačně rozsah prací

b. Případ užití popisuje více způsobů užití funkčnosti systému

c. USE CASE je součástí UML

Které dva prvky jsou součástí USE CASE diagramu

a. Actor a use case

b. Actor a činnost

c. Činnost a rozšíření

Které dvě spojnice se používají v USE CASE

Vyberte jednu z nabízených možností:

a. Extend a Include

b. Extend a Output

c. Output a Input

Diagram případu užití se používají k modelování

Vyberte jednu z nabízených možností:

a. Posloupnosti zpráv předávaných mezi objekty

b. Základních struktur v systému

c. Interakce uživatele se systémem

Dva základní prvky „USE CASE“ diagramu jsou

Vyberte jednu z nabízených možností:

a. Aktér (znázorňovaný jako postavička s názvem) a případ užití (znázorňovaný jako elipsa s názvem uvnitř)

b. Aktér (znázorňovaný jako kruh s názvem uvnitř) a případ užití (znázorňovaný jako elipsa s názvem uvnitř)

c. Aktér (znázorňovaný jako postavička s názvem pod) a případ užití (znázorňovaný jako kruh s názvem uvnitř)

ODPOVĚDI

b. a. a. c. a.

---

4 POPIS JAZYKA UML – MODELOVÁNÍ TŘÍD A OBJEKTŮ

RYCHLÝ NÁHLED KAPITOLY

Tato kapitola se zabývá a popisuje základní diagram statické struktury IS, tj. modelování tříd a objektů. Dále se zabývá vztahy mezi objekty. Toto téma řeší diagram UML tříd a diagram UML objektové spolupráce.

CÍLE KAPITOLY

Cílem kapitoly je vysvětlit pojmy třída a objekt a naučit se porozumět a vytvořit model objektové spolupráce.

ČAS POTŘEBNÝ KE STUDIU

90 minut

KLÍČOVÁ SLOVA KAPITOLY

Model, IS, třída, objekt, objektová spolupráce.

4.1 Základní popis diagramu tříd

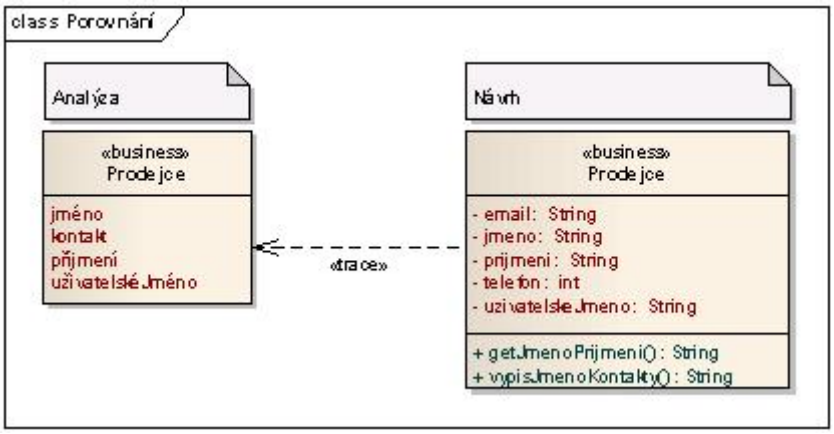
DEFINICE

Diagram tříd (Class Diagram) představuje „statický pohled na modelovaný systém“ [Buch2007] a jeho úkolem je znázornit typy objektů v systému a jejich vztahy. [Fow2003] Návrh tříd, jejich odpovědností a následné vytvoření tohoto diagramu je jedním z prvních a základních kroků analýzy navrhovaného programového systému. [Buch2007]

Při tvorbě diagramu tříd je nutné vzít v úvahu jeho účel a rozlišit, zda potřebujeme vyjádřit požadavky na modelovaný software nebo získat podrobný popis designu, atd. Z tohoto důvodu se dle [Buch2007] rozeznávají tři úrovně modelu tříd – konceptuální, designová a implementační.

Konceptuální model je vytvářen za účelem analýzy požadavků na software

Konceptuální (doménový, analytický) model (viz. Obrázek 7) je vytvářen za účelem analýzy požadavků na software. Obsahuje pouze tzv. byznys třídy (business classes), které modelují problémovou oblast a jsou tedy součástí slovníku problémové domény (slovníčku pojmů). [Buch2007] U jednotlivých tříd se uvádí obvykle jen názvy klíčových atributů a některých klíčových metod. Pokud je diagram vytvářen pouze za účelem znázornění relací mezi třídami, je možné atributy i metody vynechat. [Arl2008].



Obrázek 7 Porovnání třídy analytického a návrhového modelu tříd, zdroj: http://uml.czweb.org/diagram_trid.htm

Model návrhu

Designový model (model návrhu) vychází z modelu konceptuálního, který rozšiřuje a zpřesňuje například o viditelnosti atributů a metod, datové typy apod. Dále do modelu přidává třídy uživatelského rozhraní (presentation classes) a třídy obsluhující systémové události (control classes), [Buch2007]. Z jedné třídy v analytickém modelu se tedy může

stát v designovém modelu více návrhových tříd. Mezi analytickými třídami a třídami návrhu existuje relace typu trace, viz obrázek 7, [Arl2008].

Implementační model

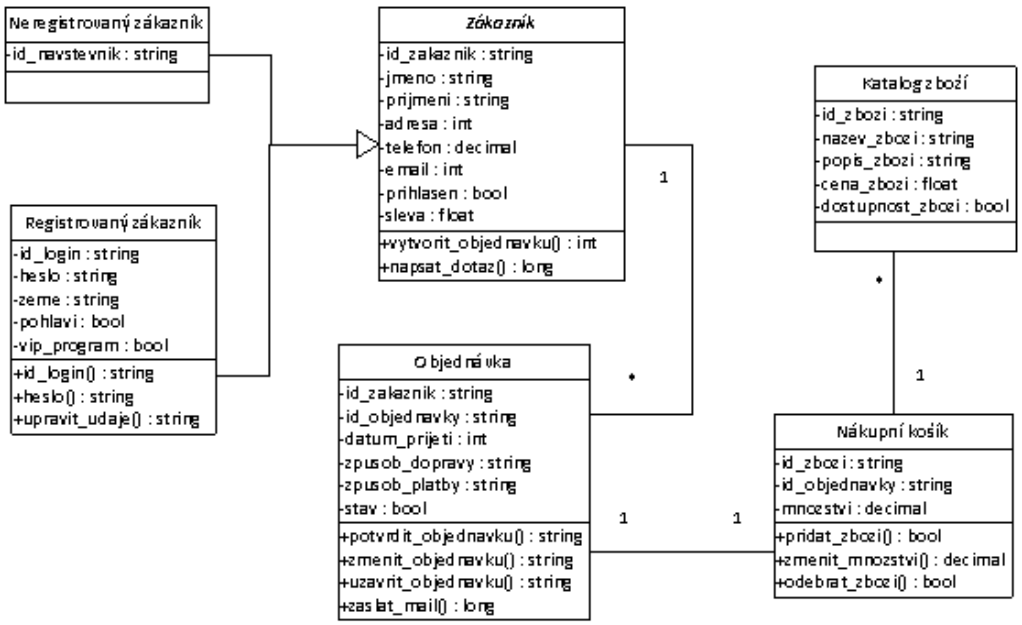
Implementační model se zaměřuje na „grafické zobrazení implementovaného kódu“, [Buch2007].

4.2 Prvky diagramu tříd a doporučení k vytváření modelu tříd

Mezi prvky používané v diagramu tříd lze zařadit třídy (classes), asociace (associations), rozhraní (interfaces) a popřípadě balíčky (packages). [Buch2007] Třída je „abstrakcí objektů se stejnými vlastnostmi, stejným chováním a stejnými vztahy k ostatním objektům.“ [Buch2007] Každá třída má popsány vlastnosti a operace (souhrnně označovány jako features), které může provádět, a omezení definující, jak mohou být jednotlivé třídy propojeny. [Fow2003] Vlastnosti tříd jsou v UML označovány jako atributy, operace ve fázi návrhu jako metody (V rámci analýzy se používá označení operace). Třídy jsou vzájemně propojeny pomocí asociací. [Buch2007]; [Arl2008]

Pro vytváření modelu tříd pomocí diagramu UML platí celá řada doporučení. Nejdůležitějším pravidlem je, že odpovědnosti tříd, které se zaznamenávají jako atributy, by měly být identifikovány již v analytickém (doménovém) modelu. Pro pojmenování tříd, metod, atributů platí další sada pravidel, které souvisí s konvencemi použitého programovacího jazyka. Další pravidla platí pro vazby mezi třídami, důraz na jejich minimalizaci, atd. Tato pravidla jsou důležitá pro pochopení problematiky a jsou podrobně rozpracována v literatuře, viz např. [Clas2006b], [Arl2008] a [Fra2014]. Návrh a vykreslení tohoto diagram je velmi obtížný proces, vyžadující zkušenosti a v praxi se postupuje tak, že se diagram diskutuje v týmu a postupně (iteračně) se diagram tříd vylepšuje, než se dosáhne konečné verze.

4.3 Příklad modelu tříd objednávka na e-shopu



Obrázek 8: Doménový model tříd pro objednávku na e-shopu, zdroj: vlastní

4.4 Model objektové spolupráce

Cílem této podkapitoly je naučit se porozumět a vytvořit model objektové spolupráce. Výše jsme ukázali, jak pomocí analytických tříd můžeme modelovat statickou strukturu systému. Pro identifikaci tříd v systému posloužily vytvořené případy užití. Pro vytváření modelu objektové spolupráce využijeme opět případy užití k tomu, abychom pro ně hledali jejich realizace, tedy takové množiny tříd, které provádějí chování případu užití pomocí vzájemné spolupráce. Jinak řečeno se jedná o převod slovního popisu scénáře případu užití na model interakce identifikovaných tříd.

Proces hledání realizace případů užití je soustavným (iteračním) upřesňováním. Přitom procházíme specifikace jednotlivých případů užití a modelujeme způsob, jak požadované chování zajistit pomocí nalezené množiny analytických tříd a jimi poskytovaných operací. Volání těchto operací je zajišťováno systémem předávání zpráv mezi objekty.

K modelování spolupráce objektů používáme zvláštní typy diagramů

Pro modelování spolupráce objektů používáme zvláštní typy diagramů, které mají vyjadřovací schopnosti k tomu, aby znázornily, jak mezi sebou objekty spolupracují. Právě interakční diagramy jsou tím nástrojem, který nám pomůže odhalit většinu operací spolupracujících tříd. [Kan004]

4.5 Základní charakteristika a prvky objektového diagramu

Objektový diagram (Object Diagram) je snímkem objektů a jejich vztahů v systému v určitém časovém okamžiku. [Buch2007] Je také nazýván diagramem instancí z důvodu, že zobrazuje instance tříd. Používá se především pro znázornění určité konfigurace objektů či zobrazení vzájemně propojených objektů ve speciálních situacích, kdy je diagram tříd či sekvenční diagram nepostačující. [Buch2007]; [Fow2003] Objektový diagram může být chápán jako speciální případ diagramu tříd vytvářený za účelem zdůraznit vazby mezi instancemi.

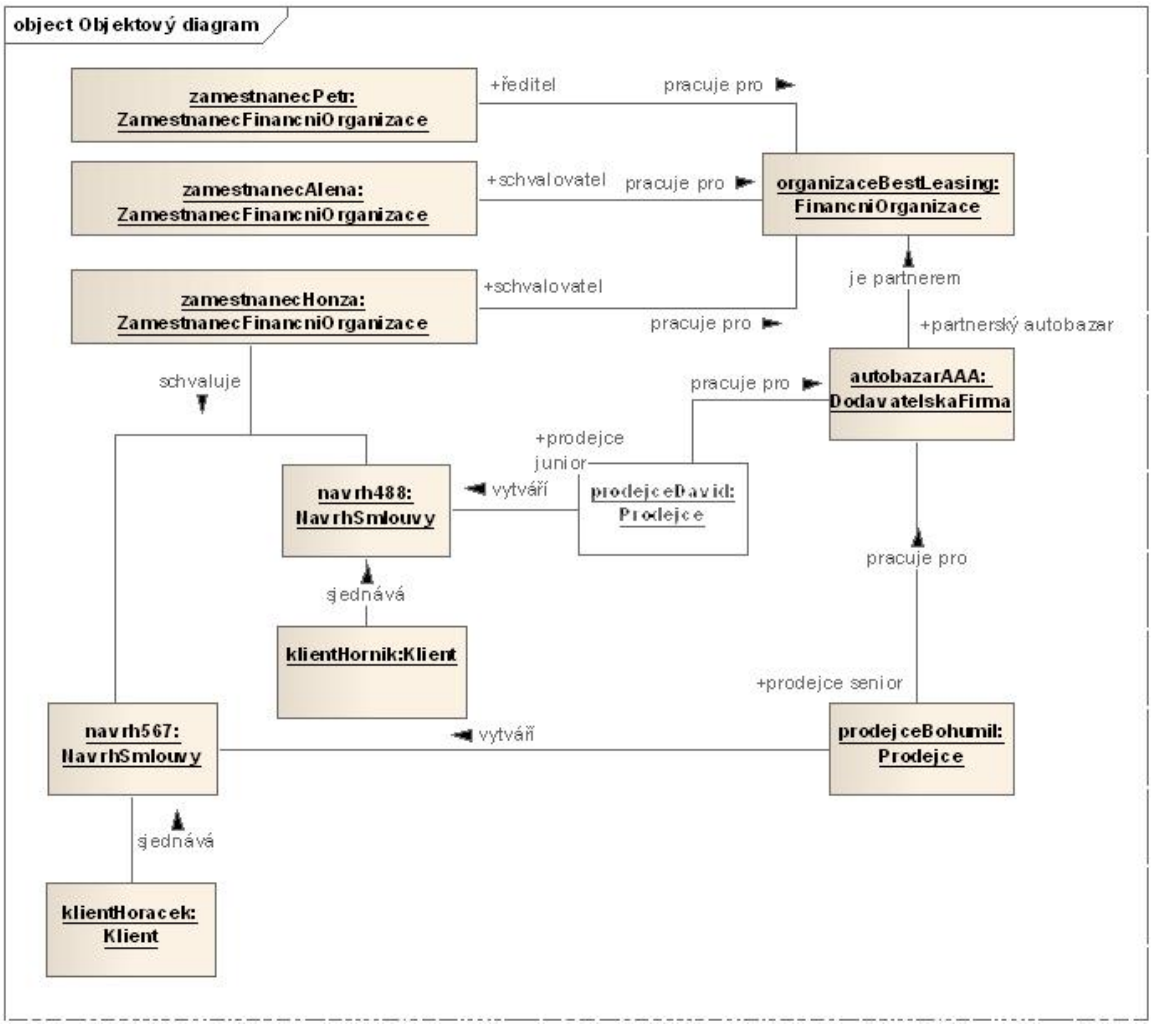
Objektový diagram je užitečný také v počátečních fázích projektu pro modelování ukázek problémové domény, které odhalují objekty a jejich vztahy (viz. Obrázek 9). Často se také používá pro modelování testovacích případů (test cases) pro ověření správnosti diagramu tříd. [Pen2003]

Objektový diagram se svou notací velmi podobá diagramu tříd či komunikace a obvykle obsahuje pouze objekty (objects) a spojení (connections) mezi nimi. Atributy se u objektů vyznačují pouze v případě, že je to nutné pro jejich jednoznačnou identifikaci, metody se neuvádějí vůbec, viz [Obj2006].

SAMOSTATNÝ ÚKOL

Vyzkoušejte si tvorbu objektového diagramu na tématu v rámci své seminární práce, využijte k tomu sadu podrobných návodů a doporučení zpracovaných v publikacích [Fra2013], [Arl2008] a [Pen2003].

4.6 Příklad objektového diagram



Obrázek 9: Objektový diagram, zdroj: http://uml.czweb.org/diagram_trid.htm

SHRNUTÍ KAPITOLY

Tato kapitola vysvětluje pojmy objekty a třídy. Jsou v ní obsaženy doporučení pro jejich vytváření. V této souvislosti byla probrána metodika vytváření objektového diagramu.

OTÁZKY

Na kterých dvou principech je založena struktura tříd?

Vyberte jednu z nabízených možností:

a. Zodpovědnost a zapouzdření

b. Struktura a metodika

c. Název atributu a viditelnost

Který z pojmů nevyjadřuje vztah mezi třídami

Vyberte jednu z nabízených možností:

a. Agregace

b. Rekurze

c. Asociace

Třída je standardní konstrukce UML používaná pro

Vyberte jednu z nabízených možností:

a. Definování vzorů, z nichž během běhu systému (software) vzniká programový kód

b. Definování vzorů, z nichž během běhu systému (software) vznikají objekty

c. Definování vzorů, z nichž během běhu systému (software) vznikají události v systému

Diagram objektové spolupráce klade důraz především na:

Vyberte jednu z nabízených možností:

a. Zjednodušený přehled jednotlivých funkcí systému

b. Na pořadí jednotlivých událostí

c. Kontext a uspořádání spolupracujících objektů

Objektový diagram úzce souvisí s:

Vyberte jednu z nabízených možností:

a. S diagramem nasazení

b. Se stavovým diagramem

c. S diagramem tříd

ODPOVĚDI

a. b. b. c. c.

---

5 POPIS JAZYKA UML – STAVOVÉ DIAGRAMY

5.1 Základní charakteristika

Stavový diagram (State Machine Diagram) „zachycuje jednotlivé stavy objektu a přechody mezi nimi.“ [Buch2007] Stavové diagramy se používají především pro popis chování určitého objektu napříč více případy užití a jejich vznik je spojen už s prvními objektově orientovanými technikami. [Fow2003]

Stavové diagramy jsou jednou ze známých technik pro znázornění chování systému

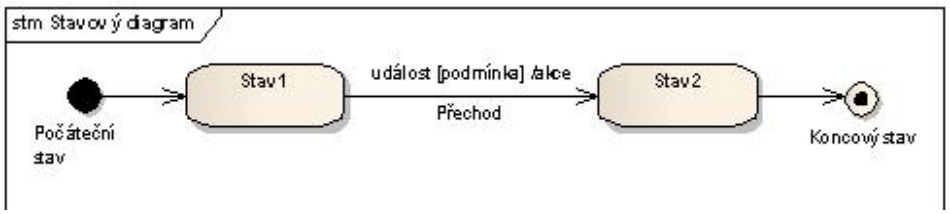
Stavové diagramy jsou jednou ze známých technik pro znázornění chování systému. Popisují všechny možné stavy, které může nabývat konkrétní objekt systému, jinými slovy modelují chování objektu napříč všemi případy užití. Zároveň znázorňují, jak se stavy objektu mění v závislosti na událostech, které se ho dotýkají. [Kan2004]

V mnoha objektově orientovaných technikách se stavové diagramy kreslí pro konkrétní třídu s cílem zachytit životní cyklus konkrétního objektu.

5.2 Prvky stavového diagramu

Základními prvky stavového diagramu jsou stavy, přechody a události, viz obrázek 10. Pokud to CASE nástroj umožňuje, může být diagram ohraničen rámem s názvem objektu. V případě, že se stavy nepohybují v cyklu, měl by diagram obsahovat počáteční (initial state) a koncový stav (final state). [Arl2008]

Stav (state) je dle [Rum2004] „situace v životě objektu, během níž objekt splňuje nějakou podmínku, provádí nějakou operaci nebo čeká na událost.“



Obrázek 10: Prvky stavového diagramu, zdroj: http://uml.czweb.org/diagram_trid.htm

Přechody (transitions) představují podmínky pro přechod objektu z jednoho stavu do druhého. V diagramu jsou značeny linií vedoucí od jednoho stavu k druhému. Jejich popis se skládá ze tří základních částí: událost [podmínka]/ akce. K vykonání uvedené akce a přechodu do dalšího stavu může dojít pouze v případě, pokud je při vzniku události uvedená podmínka (guard) pravdivá. Událost (trigger signature) je „specifikací určitého výskytu něčeho v čase a prostoru.“ [Arl2008] Pokud není v diagramu uvedena, znamená to, že přechod

do dalšího stavu probíhá automaticky. Na obrázku 11 jsou přechody označeny pouze událostmi (např. posuzování zahájeno). [Arl2008]; [Fow2003]

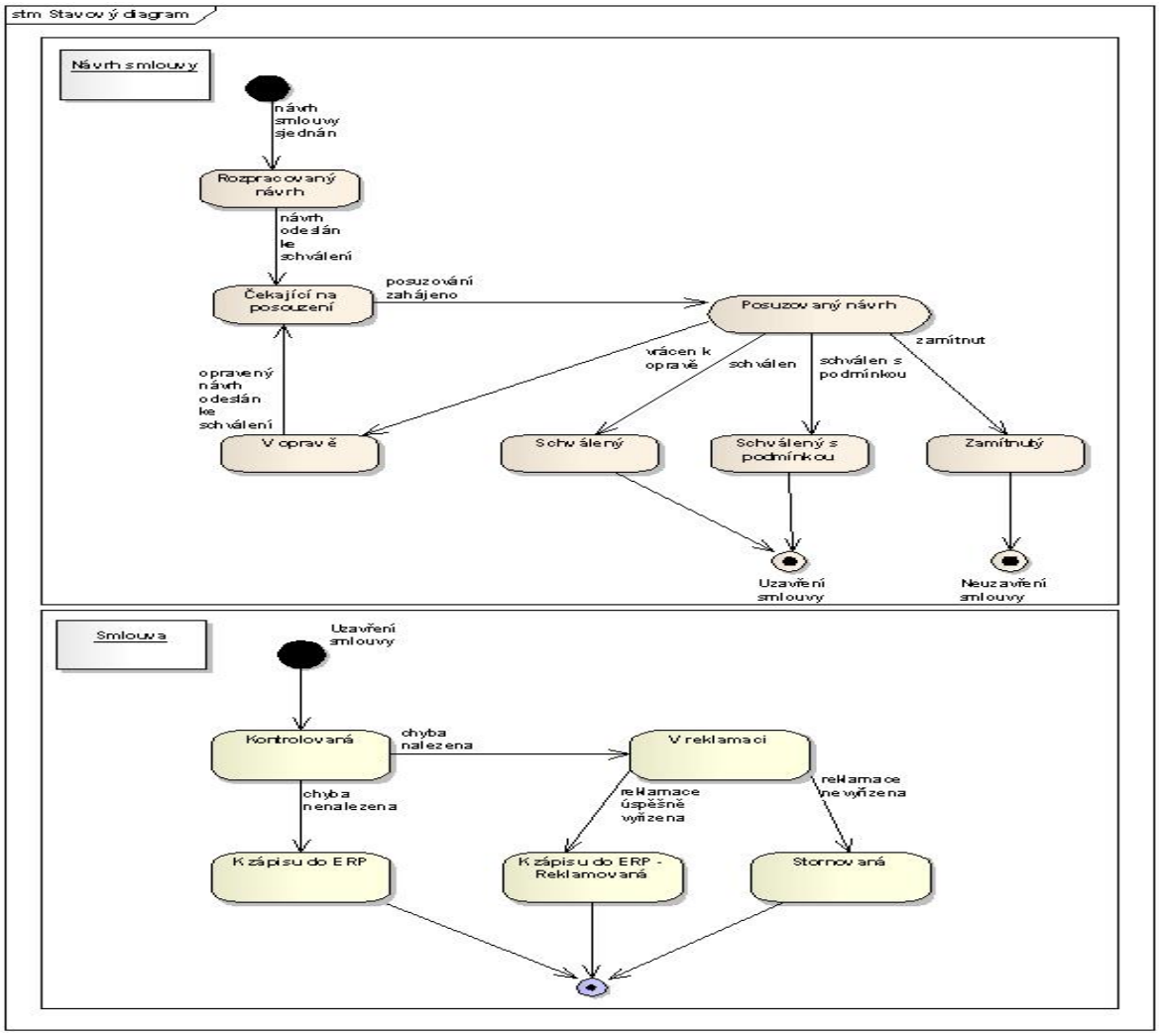
5.3 Doporučení k vytváření stavového diagramu

Podrobnější popis doporučení k vytváření stavového diagramu je zpracován v předchozím elektronickém skriptu [Fra2014], které je k dispozici studentům předmětu „Objektové metody modelování“ na e-learning portále SU v rámci připraveného kurzu.

SAMOSTATNÝ ÚKOL

Podle literatury [Fra2014], [Sta2006] a [Arl2008] vypracujte návod a doporučení pro vytváření stavových diagramů. Získané poznatky aplikujte na svůj příklad stavového diagramu v rámci seminární práce.

5.4 Příkladový stavový diagram



Obrázek 11: Stavový diagram, zdroj: http://uml.czweb.org/

SHRNUTÍ KAPITOLY

V této kapitole byl popsán stavový diagram. Na příkladově je vysvětlen princip a smysl stavového diagramu.

OTÁZKY

K čemu slouží stavový diagram

Vyberte jednu z nabízených možností:

a. Modeluje chování objektu pro některé případy užití

b. Popisují některé stavy, které může nabývat konkrétní objekt systému

c. Znázorňují, jak se mění stavy objektů v závislosti na událostech

Stavový diagram obsahuje prvky

Vyberte jednu z nabízených možností:

a. Start, stop, stav

b. Start, stop, příkaz

c. Start, stav, rozhodnutí

Kolik může být symbolů Start ve stavovém diagramu

Vyberte jednu z nabízených možností:

a. Více

b. Právě jeden

c. Dva

ODPOVĚDI

c. a. b.

6 POPIS JAZYKA UML – DIAGRAMY AKTIVIT

RYCHLÝ NÁHLED KAPITOLY

Tato kapitola objasňuje princip diagramu aktivit.

CÍLE KAPITOLY

Cílem je porozumět a naučit vytvářet diagram aktivit

ČAS POTŘEBNÝ KE STUDIU

90 minut

KLÍČOVÁ SLOVA KAPITOLY

UML, diagram aktivit,

6.1 Základní charakteristika diagramu aktivit

Diagram aktivit (Activity Diagram) je typem diagramu interakcí, který se používá pro popis procedurální logiky, byznys procesů či pracovních postupů. Umožňuje také graficky modelovat jednotlivé případy užití jako posloupnost akcí. [Fow2003] [Arl2008]

Diagramy aktivit představují pravděpodobně nejobtížnější část jazyka UML. Jsou užitečné zejména tím, že dovolují popisovat chování, které má charakter paralelního zpracování.

Activity diagram zobrazuje sekvenci činností

Diagram aktivit (Activity diagram) zobrazuje sekvenci činností, které podporují jak sekvenční, tak paralelní chování. Diagram aktivit je v podstatě variantou stavového diagramu.

Diagramy aktivit modelují procesy jako kolekce aktivit a přechodů mezi nimi. Vzhledem k tomu, že diagramy aktivit jsou určeny zejména pro komunikaci s lidmi se znalostí struktury obchodních a podnikových procesů, měla by být dostatečně přehledné. [Kan2004]

6.2 Prvky diagramu aktivit

DEFINICE

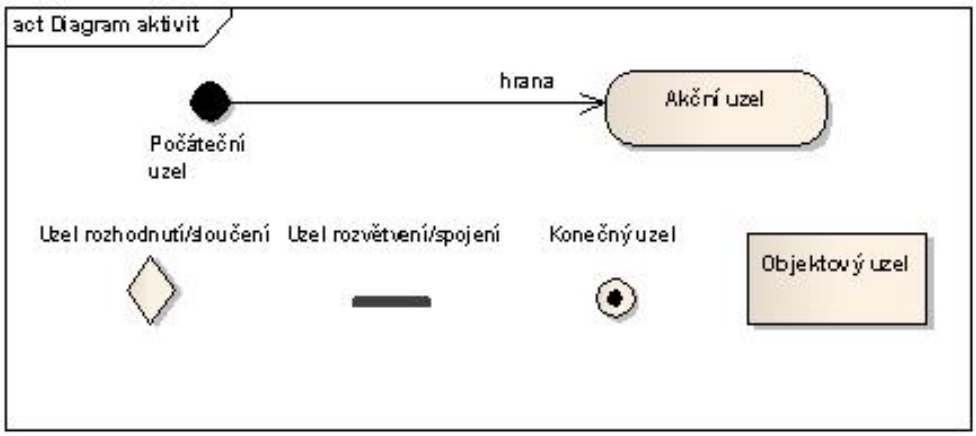
Diagram aktivit modeluje procesy jako aktivity, které se skládají z uzlů (nodes) vzájemně propojených hranami (edges), viz obrázek 12. [ARL2007]

Tři typy uzlů diagramu aktivit

Existují tři typy uzlů – akční uzly, které reprezentují samostatné a v rámci aktivity nedělitelné jednotky, řídící uzly, jejichž úkolem je řídit cestu uvnitř aktivity a uzly objektové, které zastupují objekty. Nejpoužívanějším akčním uzlem je tzv. call action node, který inicializuje aktivitu, chování či operaci. Příkladem řídících uzlů jsou počáteční (initial nodes), konečné uzly (final nodes) nebo uzly rozhodnutí (decision nodes). [Arl2008]

Uzel rozhodnutí má jednu vstupní hranu a několik konkurujících si hran výstupních. Daný výstup bude zvolen podle toho, která ze vzájemně se vylučujících kontrolních podmínek bude splněna. K označení hrany, která bude použita, pokud nebude splněna žádná kontrolní podmínka, se používá klíčové slovo jinak (else) [Fow2003], [Arl2008].

Uzel sloučení (merge) může mít několik vstupních, ale pouze jednu výstupní hranu. Používá se především pro sjednocení větví diagramu aktivit, které byly předtím rozděleny uzlem rozhodnutí [Arl2008].



Obrázek 12: Prvky diagramu aktivit, zdroj [ARL2007]

Procesy, které diagram popisuje, mohou probíhat paralelně, což je umožněno řídícími uzly rozvětvení (fork) a spojení (join) [Fow2003]. Pro zpřehlednění se může diagram rozdělit například dle rolí či organizačních jednotek do tzv. zón odpovědnosti či plaveckých drah (swimlanes) [Arl2008].

6.3 Doporučení k vytváření diagramu aktivit

Podrobnější popis doporučení k vytváření stavového diagramu je zpracován v předchozím elektronickém skriptu [Fra2014], které je k dispozici studentům předmětu „Objektové metody modelování“ na e-learning portále SU v rámci připraveného kurzu.

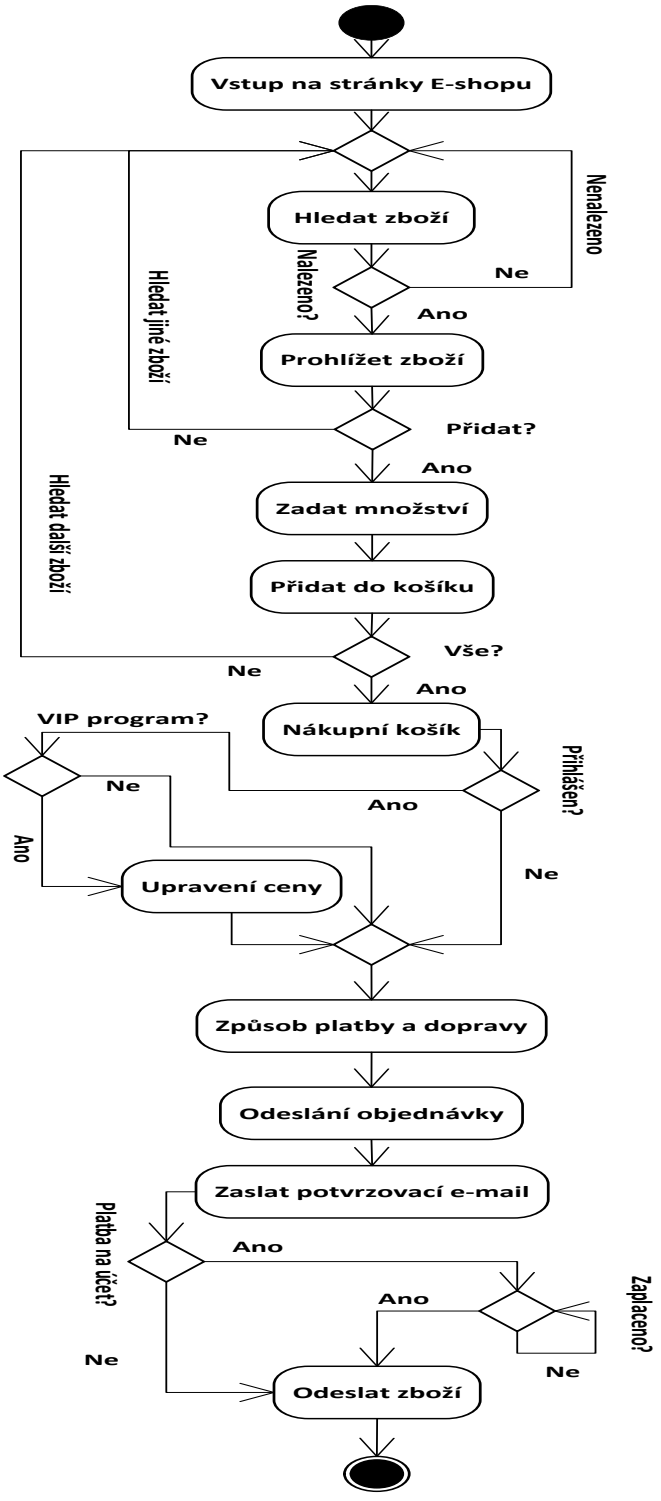
SAMOSTATNÝ ÚKOL

Podle literatury [Fra2014], [Act2006] a případně i další literatury si sestavte pravidla pro vytváření diagramů aktivit. Tento návod aplikujte při kreslení diagramu aktivit ve své seminární práci.

6.4 Příklady diagramů aktivit

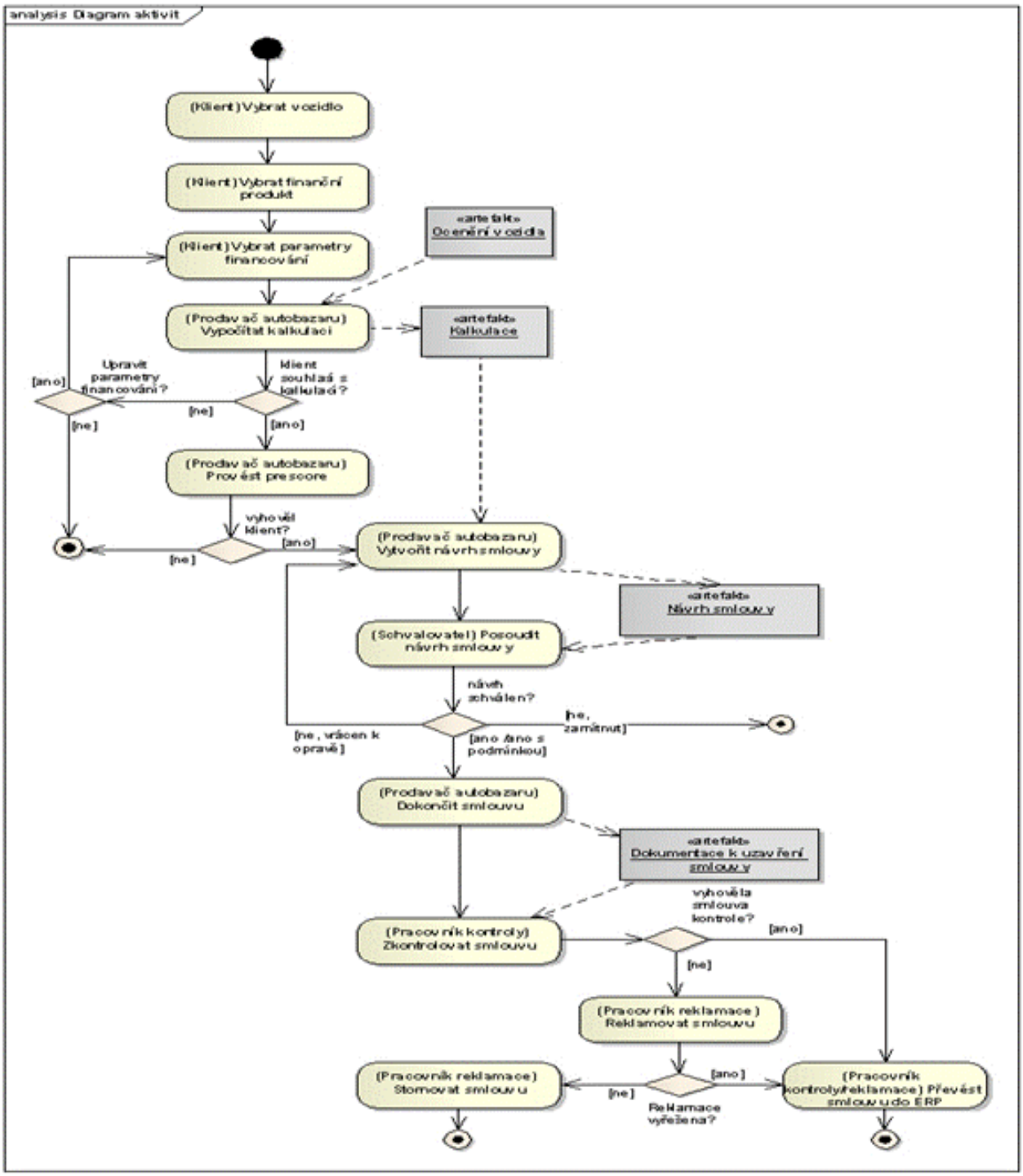
Příkladový diagram v rámci případové studie znázorňuje chování systému e-shopu při tvorbě objednávky zákazníkem. Diagram prezentuje činnost zákazníka e-shopu od vstupu na stránky při tvorbě objednávky. Proces objednávání obsahuje vyhledání zboží, prohlížení popisu zboží a postupy při jeho zakoupení. Zákazník – uživatel e-shopu si zboží prohlíží, a pokud si ho vybere, přidá ho do nákupního košíku. Tento proces přidávání zboží opakuje uživatel tak dlouho, dokud nemá v košíku všechno zboží. Samozřejmě při procesu vkládání do košíku může volit množství, přednastavená hodnota počtu kusů je 1. Systém si před přechodem k dokončení objednávky zkontroluje, zda zákazník je přihlášen a má aktivovaný

slevový VIP program. Pokud ano, tak systém upraví cenu. Následně je vyzván k vyplnění (v případě neregistrovaného zákazníka) nebo potvrzení (přihlášený zákazník) platebních, dopravních a doručovacích údajů. Následně je objednávka odeslána a systém zašle potvrzovací e-mail a na základě zvolené metody platby odešle zboží zákazníkovi.



Obrázek 13: Diagram aktivit - zákazník nakupuje na e-shopu, zdroj vlastní zpracování

Druhým příkladem složitějšího diagramu aktivit je případ ocenění vozidla, viz obr. 14.



Obrázek 14: Diagram aktivit ocenění vozidla, zdroj: vlastní

SHRNUTÍ KAPITOLY

V této kapitole byla probrána metodika vytváření diagramu aktivit. Na příkladech byly vysvětleny základní pojmy a způsob využití diagramu aktivit.

OTÁZKY

Diagramy činností se používají pro

Vyberte jednu z nabízených možností:

a. Vyjádření dynamického chování modelu

b. Vyjádření statické struktury modelu

c. Vyjádření vztahu mezi uživateli a systémem

Které tvrzení o diagramech činností je pravdivé?

Vyberte jednu z nabízených možností:

a. Diagram činností je není vybaven pro větvení dle podmínek a zachycuje jen paralelní zpracování

b. Diagram činností je vybaven pro větvení dle podmínek a zachycuje paralelní zpracování

c. Diagram činností je vybaven pro větvení dle podmínek, ale nezachycuje paralelní zpracování

Diagramy činností patří mezi diagramy

Vyberte jednu z nabízených možností:

a. Statické struktury systému

b. Dynamického chování systému

c. Žádná z uvedených dvou možností

Diagramy činností podporují

Vyberte jednu z nabízených možností:

a. Jen paralelní chování

b. Jen sekvenční chování

c. Jak sekvenční, tak i paralelní chování

ODPOVĚDI

a. b. b. c.

7 SEKVENČNÍ DIAGRAM

RYCHLÝ NÁHLED KAPITOLY

Tato kapitola objasňuje princip sekvenčního diagramu.

CÍLE KAPITOLY

Cílem je porozumět a naučit vytvářet sekvenční diagram.

ČAS POTŘEBNÝ KE STUDIU

90 minut

KLÍČOVÁ SLOVA KAPITOLY

Sekvenční diagram, UML,

7.1 Základní charakteristika a prvky sekvenčního diagramu

Sekvenční diagram popisuje, jak spolu objekty komunikují v čase

Stavové diagramy pracují se stavy objektů. Popis stavů ovšem neřeší všechno. Proto v jazyku UML existuje další prostředek - diagram sekvencí, který popisuje, jak spolu objekty v čase komunikují. [Sch2001]

Sekvenční diagram nejčastěji zobrazuje chování a spolupráci jednotlivých objektů v rámci jednoho případu užití. Pro popis chování jednoho objektu napříč více případy užití se používá stavový diagram. [Fow2003]

Sekvenční diagram (Sequence Diagram) je nejvíce používaným diagramem interakcí. „Zachycuje grafický průběh zpracování v systému v podobě zasílání zpráv.“ [Buch2007]

PRVKY DIAGRAMU

Diagram sekvencí se skládá z objektů, zpráv a znázornění času

Diagram sekvencí (sekvenční diagram) se skládá z objektů zakreslených běžným způsobem (jako obdélníčky s podtrženým jménem), ze zpráv zakreslených jako plné šipky a z času, který je znázorněn vertikálním postupem v diagramu. [Kan004] Zprávy mohou být v sekvenčním diagramu posílány jak mezi jednotlivými objekty, tak i třídami či dokonce aktéry. Proto se prvky, které mezi sebou v diagramu komunikují, nazývají souhrnně klasifikátory (classifiers). [Buch2007] Z každého klasifikátoru vede tzv. čára života (lifeline), která reprezentuje, jakým způsoben se instance určitého klasifikátoru účastní interakce. [Arl2008]

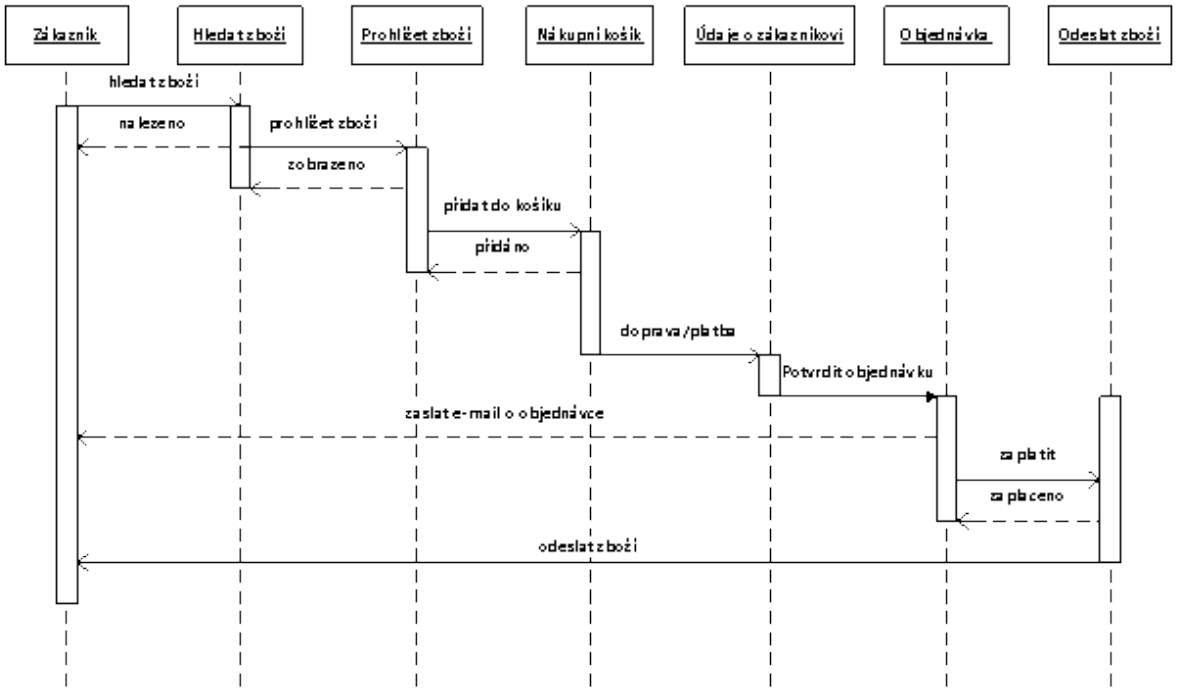
7.2 Doporučení k vytváření sekvenčního diagramu

Dle literatury [Seq2006], [Arl2008], [Buch2007] platí pro sekvenční diagram celá doporučení a návodů, například že aktéři by měli být pojmenování stejnými názvy jako v use-case diagramu. Podrobně byla pravidla vytváření sekvenčních diagramů zpracována v učebním textu [Fra2013], který je k dispozici na e-learning portálu v rámci kurzu „Objektové metody modelování“.

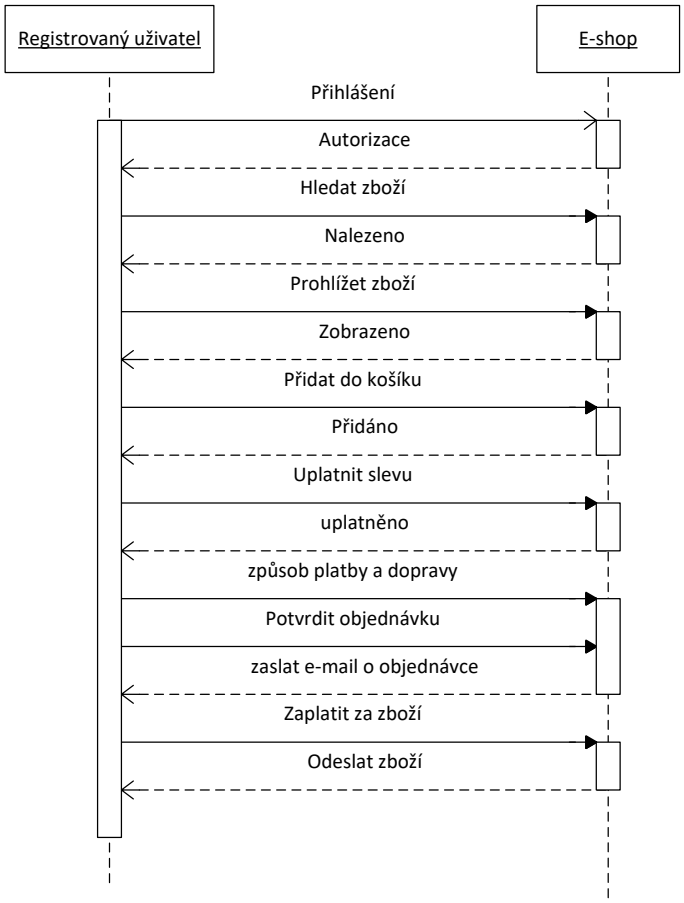
SAMOSTATNÝ ÚKOL

Podle literatury [Seq2006], [Arl2008], [Buch2007] a [Fra2014] si sestavte pravidla pro vytváření sekvenčního diagramu. Tato pravidla aplikujte při kreslení diagramu aktivit ve své seminární práci.

7.3 Příkladový sekvenční diagram



Obrázek 15: Sekvenční diagram popisující nákup zboží na e-shopu neregistrovaným uživatelem, zdroj: vlastní



Obrázek 16: Sekvenční diagram popisující nákup zboží na e-shopu registrovaným uživatelem s uplatněním slevy, zdroj: vlastní

SHRNUTÍ KAPITOLY

V této kapitole byla probrána metodika vytváření sekvenčního diagramu. Na příkladech byly ukázány základní prvky diagramu tak, aby studenti byli schopni za využití literatury sestavit sekvenční diagram.

OTÁZKY

Sekvenční diagram a diagram objektové spolupráce znázorňují

Vyberte jednu z nabízených možností:

a. USE CASE - případ užití

b. Statickou strukturu systému

c. Jak spolu objekty spolupracují

Jak se v sekvenčním diagramu vyznačuje zpráva?

Vyberte jednu z nabízených možností:

a. Šipkou s vyplněným hrotem

b. Šipkou s polovinou hrotu

c. Jednoduchou šipkou

Jak se v sekvenčním diagramu zobrazuje čas?

Vyberte jednu z nabízených možností:

a. Vertikálně

b. Nezobrazuje se

c. Horizontálně zleva doprava

Sekvenční diagram vystihuje tvrzení.

Vyberte jednu z nabízených možností:

a. Zobrazení funkcionality systému

b. Zobrazení objektů v čase

c. Zobrazení vzájemných vztahů objektů

ODPOVĚDI

c. a. a. b.

---

8 PŘEHLED SOFTWARE PRODUKTŮ PRO PRÁCI

RYCHLÝ NÁHLED KAPITOLY

Kapitola dává přehled software, který se používá, resp. je dostupný na trhu. Jistě existuje celá řada software produktů, které umožňuji konstrukci UML diagramů. Zde jsou vytipovány tři základní produkty s jejich popisem funkčnosti.

CÍLE KAPITOLY

Cíl kapitoly dát přehled o nejčastěji používaných sw produktech a v případě SW Enterprise Architect dát podrobnější návod k jeho ovládání.

ČAS POTŘEBNÝ KE STUDIU

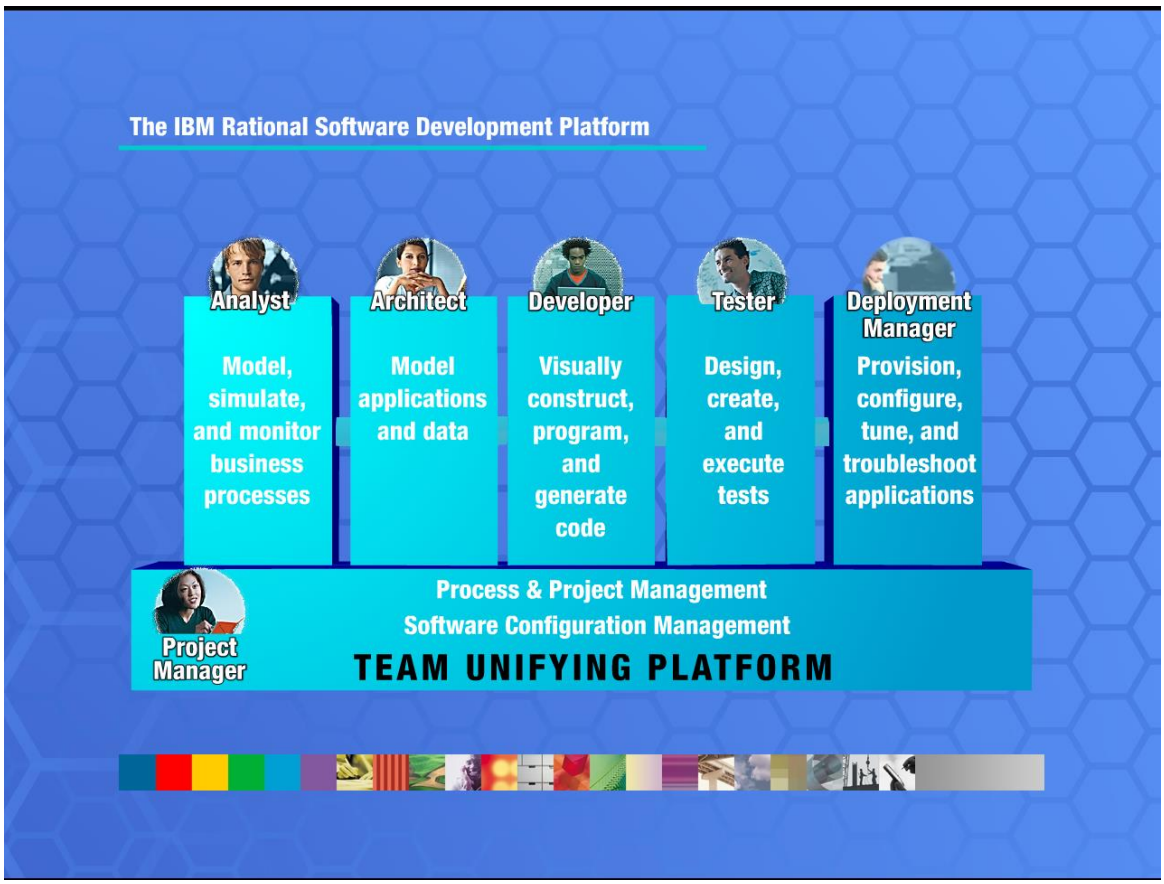
180 minut

KLÍČOVÁ SLOVA KAPITOLY

Software, UML diagramy, objektový orientovaný návrh SW.

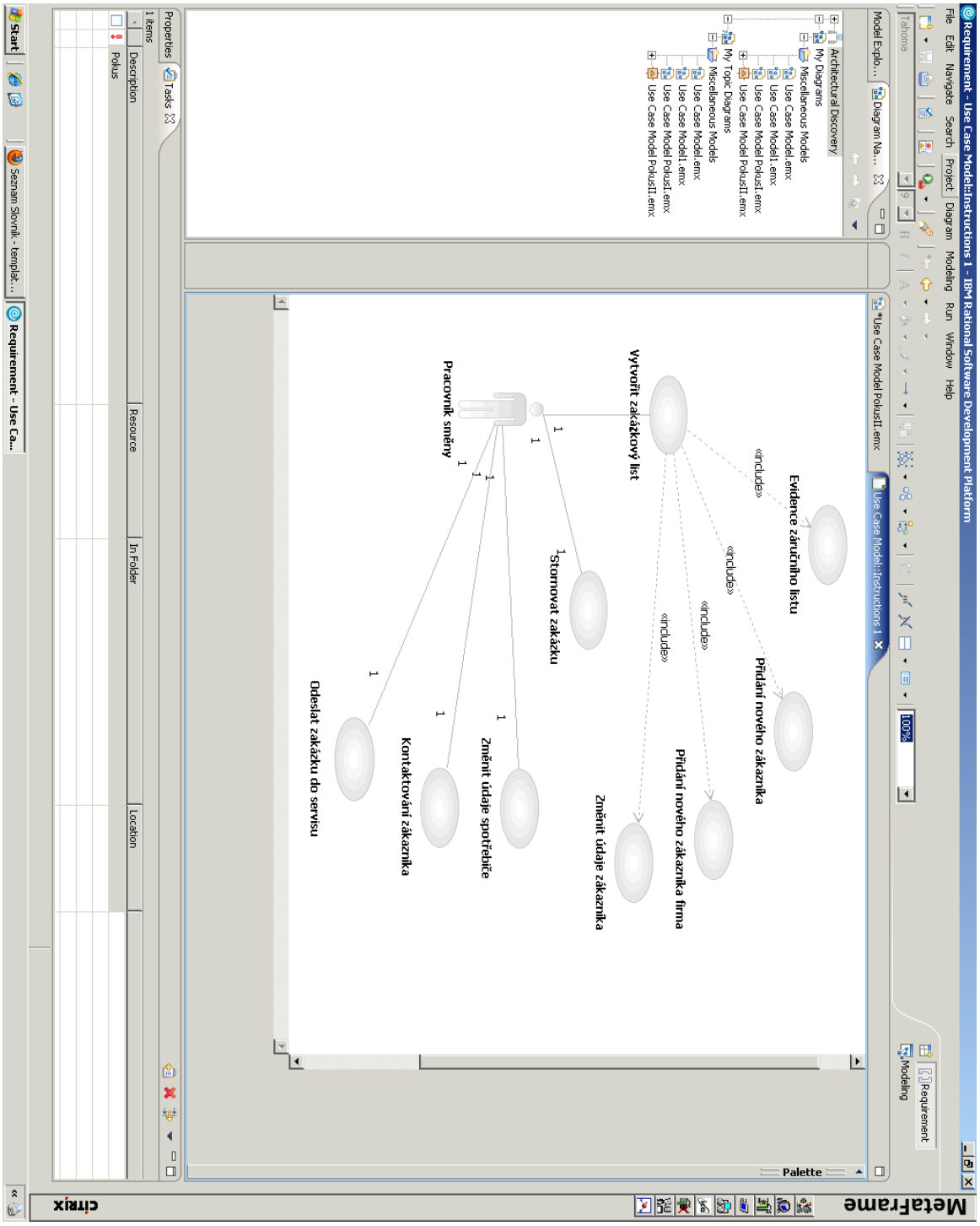
8.1 IBM Rational Software Development Platform

Case nástroj firmy IBM pro podporu vývoje aplikací zahrnuje všechny možné nástroje od popisu firemních procesů až po kreslení diagramů UML. Je určen pro týmovou práci při řešení rozsáhlých projektů vyvíjejících informační systém.



Obrázek 17: IBM Rational Development Platform Týmová práce při vývoji software, zdroj: vlastní (printscreen)

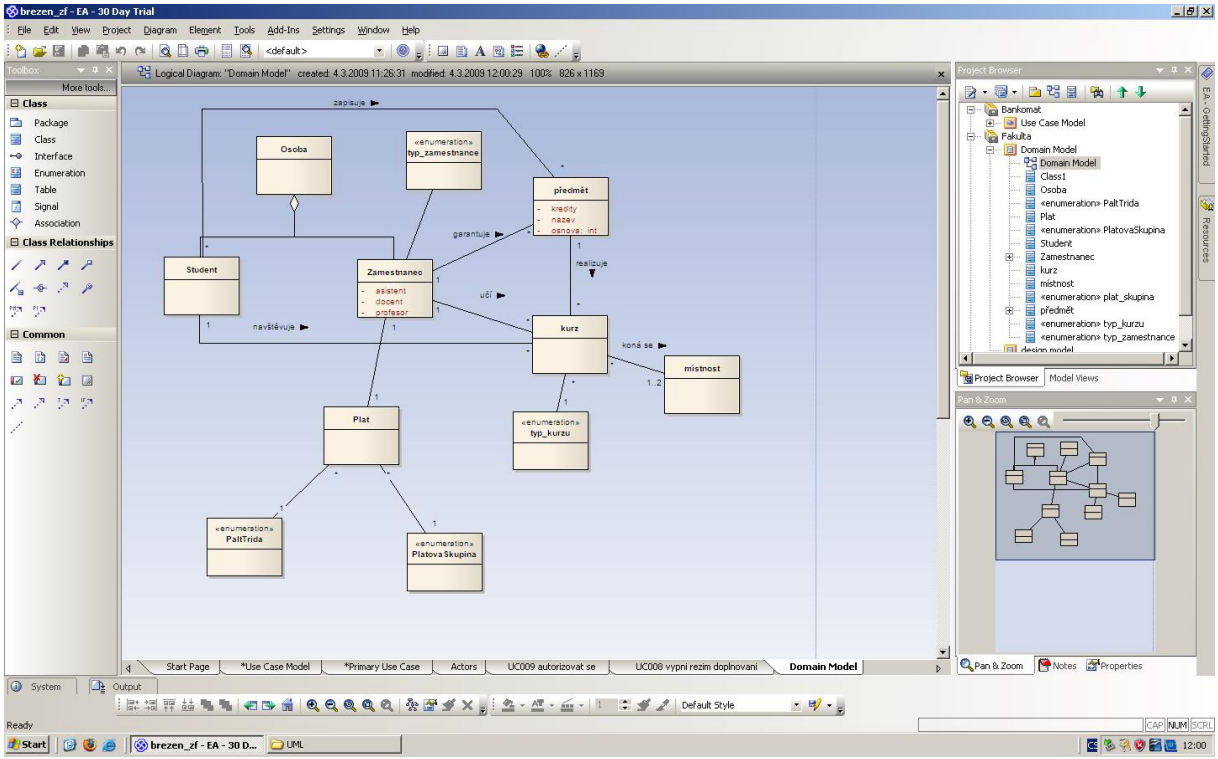
Tento software od firmy IBM byl nainstalován na serveru počítačové sítě SU OPF. Vzhledem k jeho vysoké ceně fakulta vlastní pouze jednu licenci pro demonstrativní účely ve výuce.



Obrázek 18: Příklad vývojového prostředí IBM Developer Platform – USE CASE Model, zdroj: Vlastní s využitím software IBM Rational Development Platform

8.2 ENTERPRISE ARCHITECT firmy SPARX

Velmi používaný program pro kreslení UML diagramů je Enterprise Architect firmy Sparx. Program je běžně využíván pro výuku pro transparentnost, přehlednost a poměrně jednoduché ovládání.



Obrázek 19: Vývojové prostředí Enterprise Architect firmy Sparx – CLASS DIAGRAM, zdroj: vlastní s využitím software Enterprise Architect

NÁVOD K OVLÁDÁNÍ PROGRAMU

Firma Sparx umožňuje klientům a tedy i studentů stáhnout 100 denní Trial verzi.

EA firmy Sparx je ideálním CASE nástrojem, ve srovnání s Rational IBM Architect je nepoměrně levnější a ve srovnání s šablonami na podporu UML v MS VISIO je to komplexní CASE nástroj.

VYTVOŘENÍ PROJEKTU

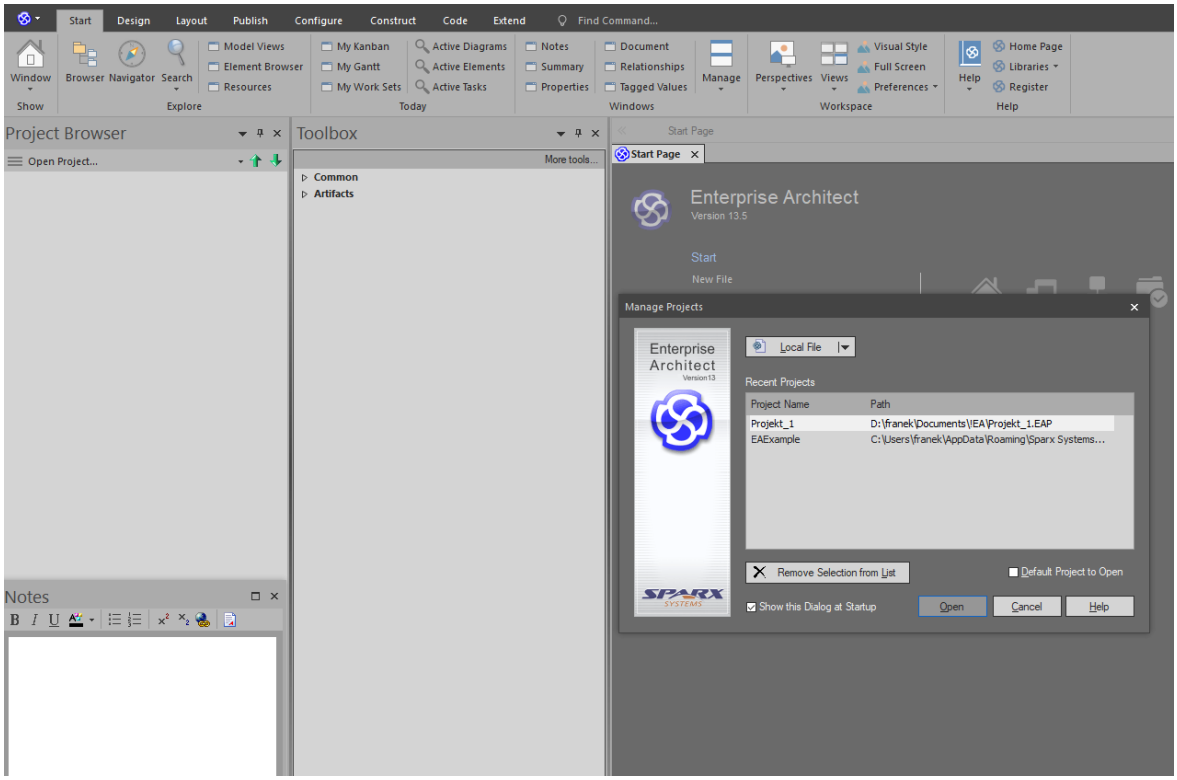
Projekt je tvořen jedním souborem nebo slouží jako úložiště pro jeden nebo více modelů. Prvním krokem je buď otevřít existující projekt, nebo vytvořit nový projekt. V tomto příkladu vytvoříme projekt založený na jednom souboru a přidáme některé modely založené na šablonách. Vytvoříme jednoduchý Use Case diagram, který budeme dále přizpůsobovat našim požadavkům. Kdykoliv můžeme znovu otevřít projekt, když na něj poklepneme v prohlížeči souborů. Objeví se také v našem seznamu Současné projekty (Recent Projects) na úvodní stránce (Start Page).

VYTVOŘENÍ NOVÉHO PROJEKTU:

1. Nastartujeme Enterprise Architect

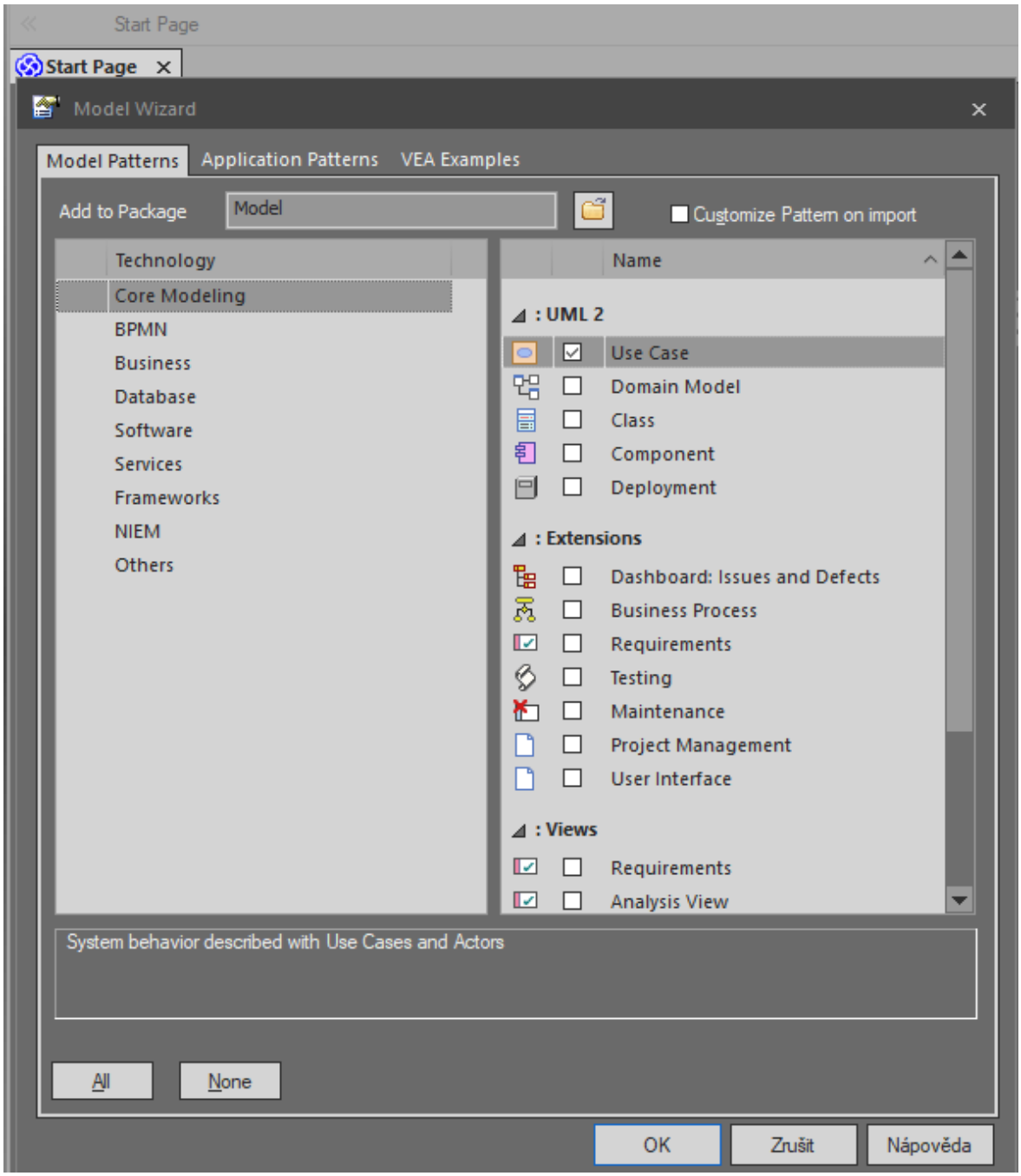
Zobrazí se pracovní prostředí EA a dialog „Manage Projects“ správy projektů se seznamem dříve použitých projektů, viz obrázek 20. Můžeme tak pokračovat v práci na již vytvořených projektech nebo si otevřít ukázkový projekt EAExample. Okno zavřeme.

2. Klikneme na tlačítko nový project (New File) a vybereme vhodné jméno a umístění nového projektu.



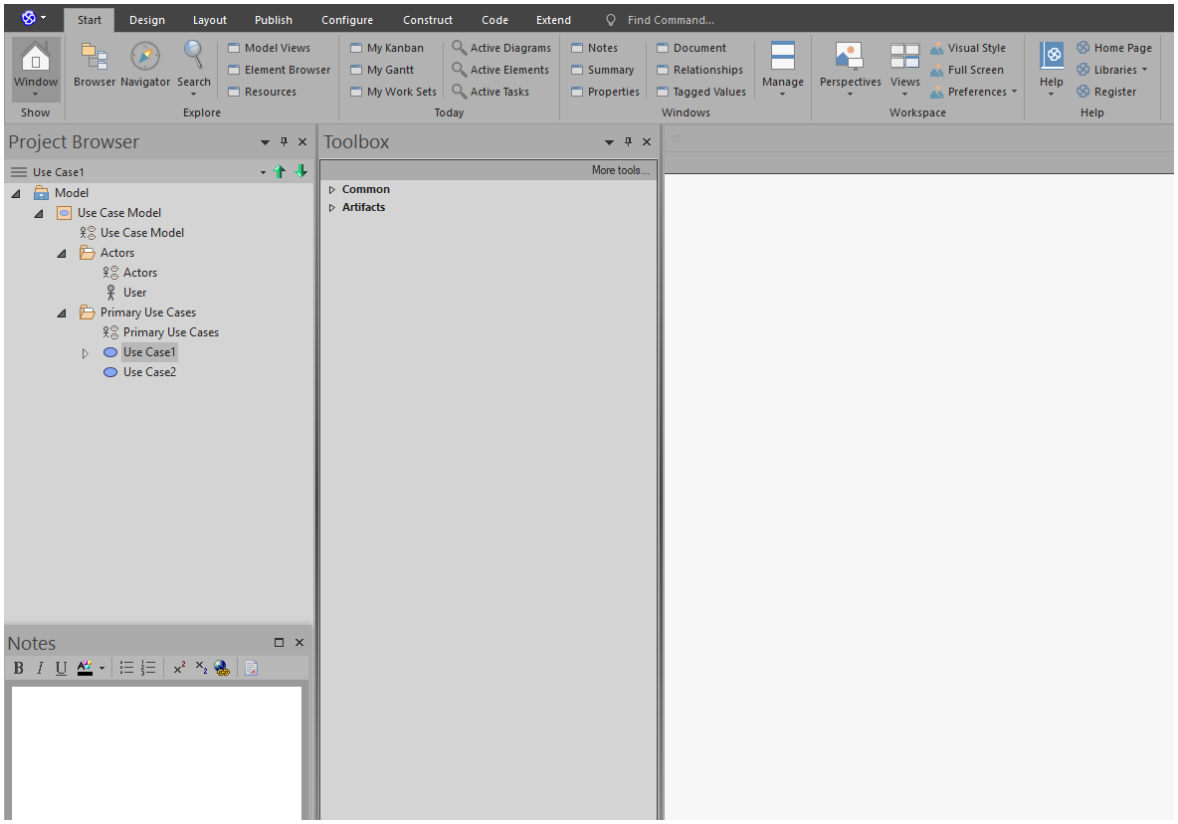
Obrázek 20 Vytvoření nového projektu v Enterprise Architect, zdroj vlastní

Otevře se standardní dialogové okno prohlížeče souborů Windows, soubory mají příponu .eap Projekt pojmenuje například newproject a soubor uložíme. Enterprise Architect vytvoří nový soubor projektu a uloží ho na určeném místě. Od této chvíle se celý projekt bude ukládat do souboru newproject.eap. Po uložení se nám otevře průvodce Model Wizard a zašktrtneme políčko Use Case, viz obrázek 21.



Obrázek 21 Model Wizard EA s volbou Use Case, zdroj vlastní

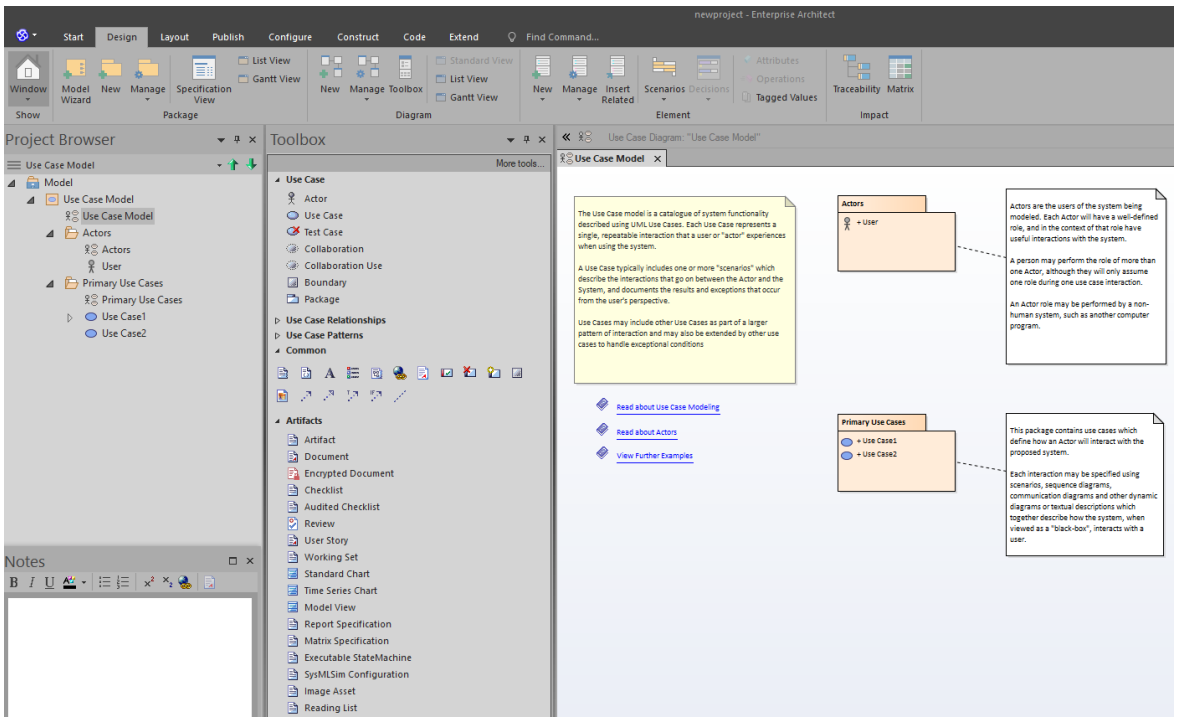
Průvodce (Model Wizard) automaticky vytvoří pro nás nový „Use Case Model“, který vidíme v Project Browser na levé straně vývojového prostředí, viz obrázek 22. Rozbalíme pomocí šipek model a dostáváme Use Case Model, Actors a Primary Use Cases.



Obrázek 22 Vytvoření Use Case Modelu, zdroj vlastní zpracování

NAKRESLENÍ USE CASE

Doubleclick na Use Case Model – otevře se nám ukázkový diagram Use Case, viz obrázek 23. Na obrázku je připraven důkladný návod a popis Use Case diagramu, balíčky Actors a Primary Use Case, viz též Project Browser.



Obrázek 23 Ukázkový Use Case Model, vlastní zpracování

PŘIDÁNÍ PRVKŮ DIAGRAMU POMOCÍ PANELU NÁSTROJŮ

V podokně Toolbox máme připravené kreslící nástroje pro kreslení diagramu. Pokud Toolbox není vidět, aktivujeme ho kombinací kláves Alt+5. Metodou drag and drop přetáhneme ikonu Actor a dvakrát Use Case. Po přetažení se nám na pracovní ploše EA objeví příslušné obrázky Actor1 a Use Case1 a Use Case2. Při každém přetažení objektu se nám vysvítí příslušné okno s vlastnostmi. Ty mohu dle potřeby vyplnit, tedy pojmenovat Actory a Use a zejména u Use Case ve vlastnostech je textové pole pro zadání scénáře. Podobně přetažením myši vytvoříme asociaci mezi Actor1 a Use Case1 a Use Case2,viz obrázek 24.

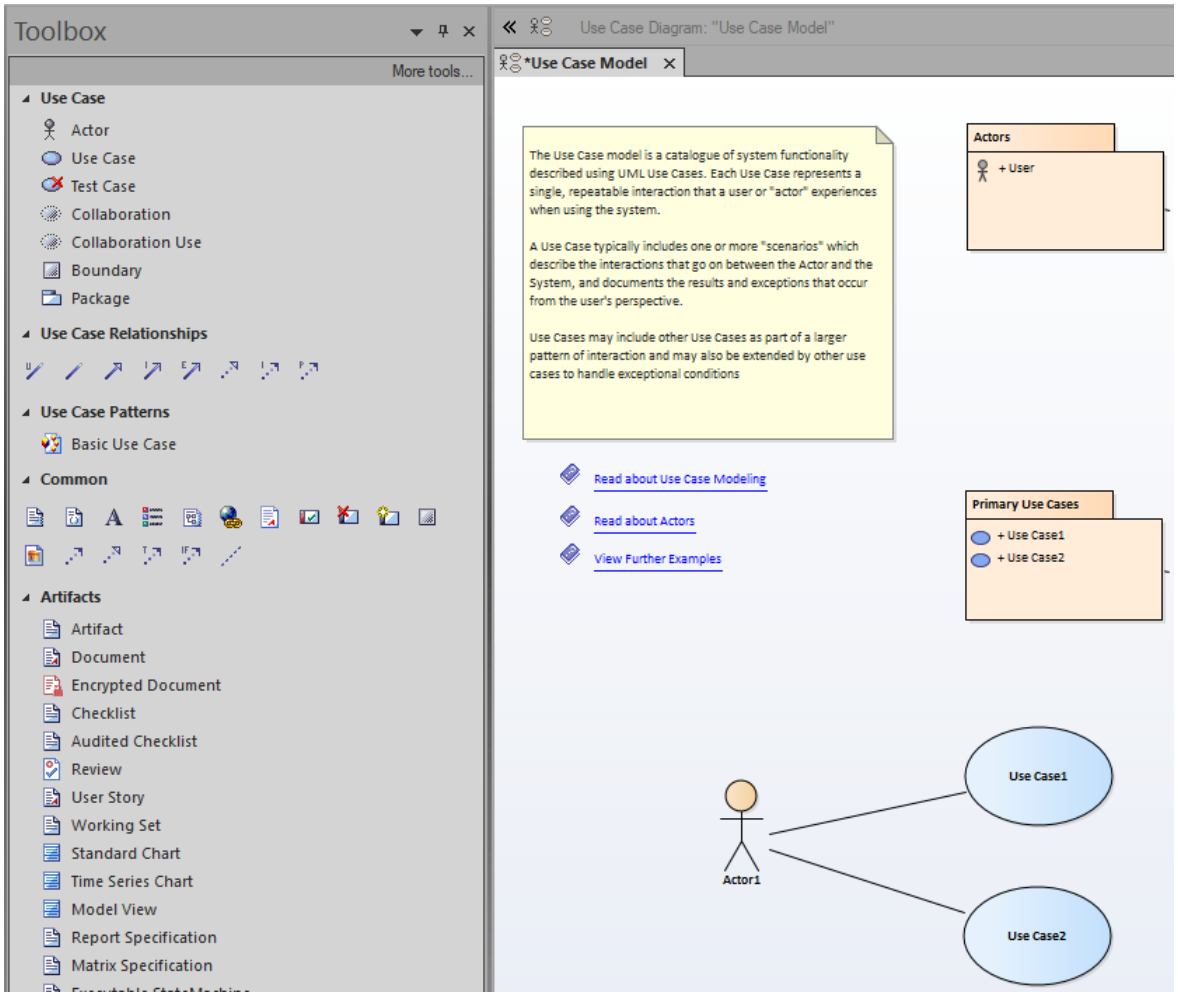
Pro vytvoření asociace můžeme využít i jiný postup. Po kliknutí na Actor1 se nám objeví pomocné navigační symboly, viz obrázek 25. Zatím jsme ponechali vzorové návody a balíčky na obrazovce. Ty můžeme jednoduše odstranit.

ODSTRANĚNÍ PRVKU

Odstranění prvku s pracovní plochy EA provedeme jednoduše tak, že prvek označíme myší a zmáčkneme klávesu DEL.

Prvek v Prohlížeči projektu můžeme mít ve více diagramech v rámci celého projektu. Pokud prvek odstraníme z diagramu pomocí klávesy Delete, nebude automaticky odstraněn odpovídající prvek v celém Prohlížeči projektu. Toho dosáhneme kombinací kláves CTRL + DEL.

Poznámka: Po stisknutí Ctrl + Delete budeme vyzváni k potvrzení odstranění a budeme varováni, že operaci nelze vrátit zpět.

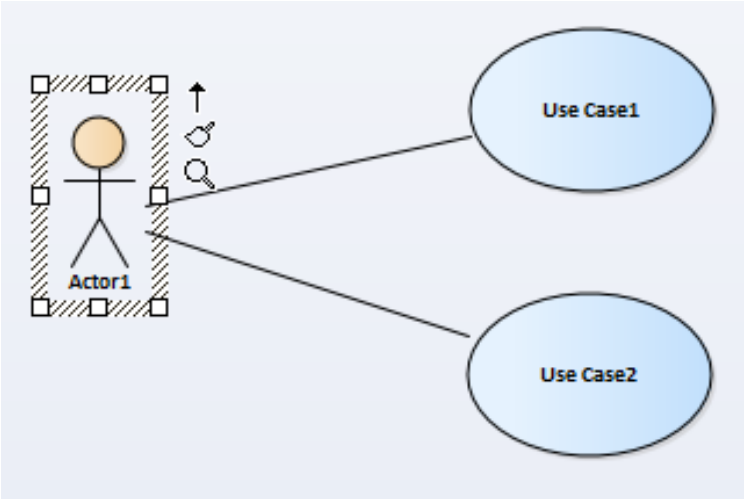


Obrázek 24 Kontextový Toolbox a přidávání prvků Use Case diagramu, zdroj vlastní

POUŽITÍ QUICKLINKERU

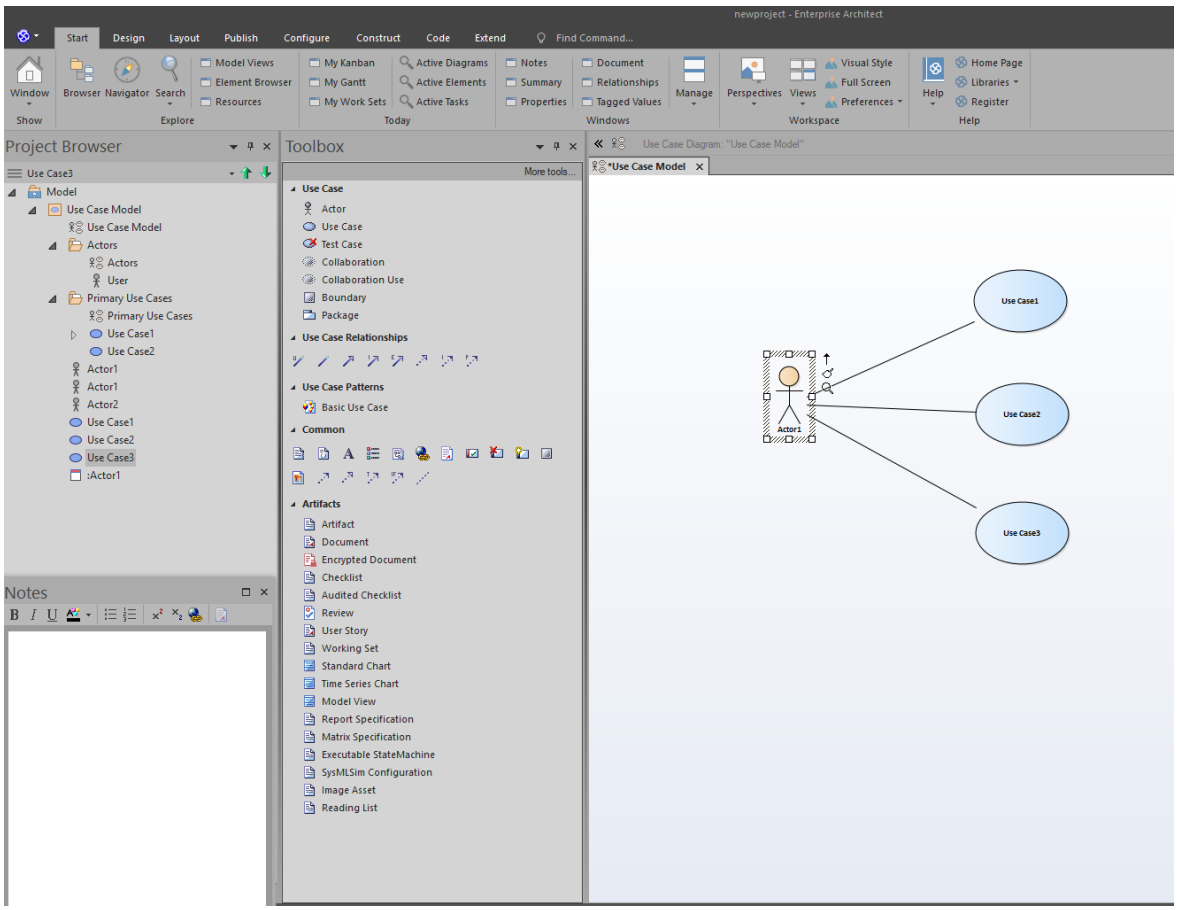
Použitím Quicklinkeru můžeme rychle vytvářet vztahy mezi stávajícími prvky bez použití Toolboxu. Po kliknutí na Actér1 se nám objeví příslušné konektory, viz obrázek 25.

Jednoduše přetáhneme šipku Quicklinkeru na požadované místo a pomocí menu vytvoříme prvek nebo konektor. Kontextové menu obsahuje nejběžnější prvky a konektory pro daný diagram.



Obrázek 25 Quicklinker – šipka, zdroj vlastní zpracování EA, zdroj vlastní

V našem případě přidáme „Use Case 3“ a smažeme všechny návody a pomocné balíčky, viz obrázek 26.



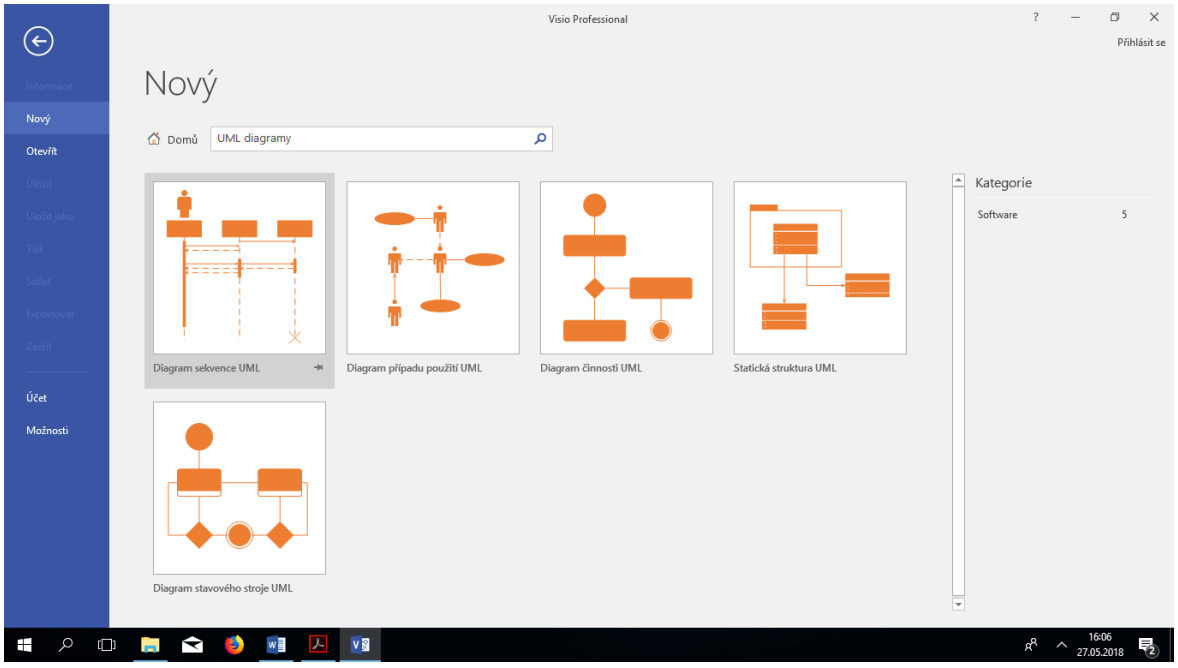
Obrázek 26 Vytvoření asociace pomocí Quicklinkeru, zdroj vlastní

8.3 UML A VISIO firmy MICROSOFT

Šablona diagramu modelu UML aplikace Microsoft Office Visio nabízí podporu vytváření objektově orientovaných modelů.

- Diagram případu použití
- Diagram statické struktury – diagram tříd
- Diagram činnosti
- Stavový diagram
- Sekvenční diagram
- Diagram komponent
- Diagram zavedení

Šablony MS VISIO pro UML vyhledáme zadání hesla UML diagramy do vyhledávání šablon a poté vybereme příslušný diagram, viz obrázek 27.



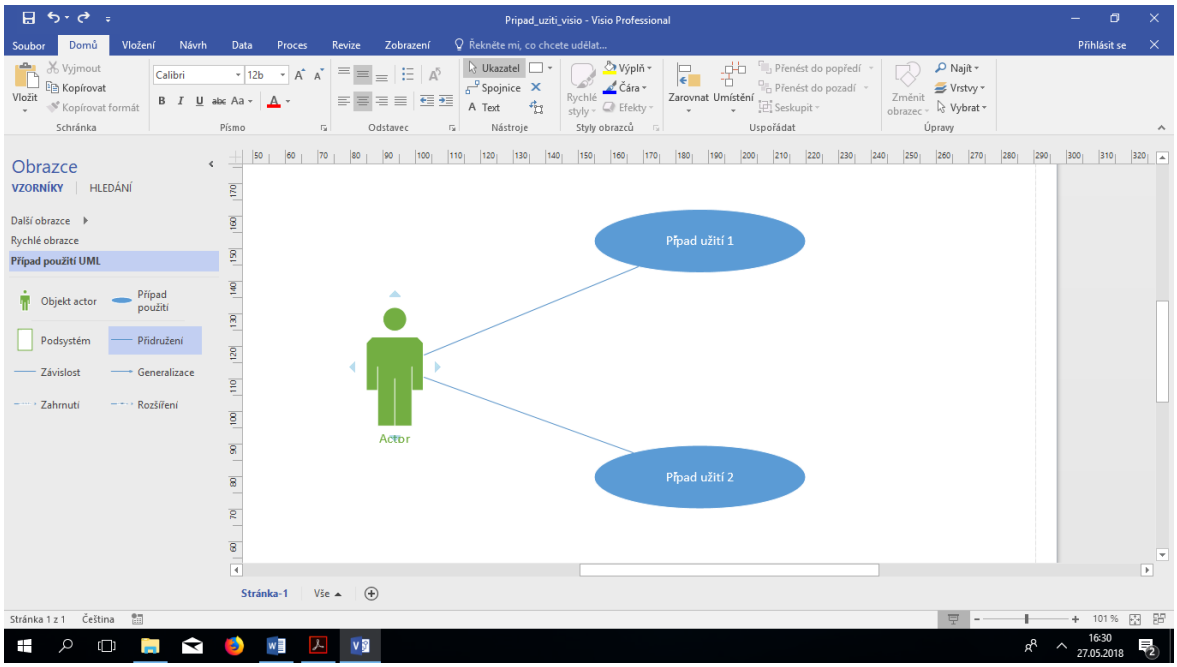
Obrázek 27 Vyhledání šablon na podporu UML diagramů, zdroj vlastní v MS VISIO

Program VISIO je dostupný studentům Obchodně podnikatelské fakulty na počítačových učebnách, s šablonou diagramů UML na PC učebně, kde probíhá výuka předmětu „Objektové metody modelování.

Pro řešení seminární práce, kterou studenti dle sylabu předmětu „Objektové metody modelování“ musí vytvořit je podpora UML v MS VISIO postačující. Ideální prostředek pro řešení pokročilejších projektů jsou však výše uvedené CASE nástroje.

Návody na použití těchto šablon je nad možnosti rozsahu tohoto textu, ovládání je intuitivní, s využitím znalostí ovládacích postupů v MS OFFICE.

Prostředí a vzhled obrazovky šablony pro USE Case je na následujícím obrázku 28:



Obrázek 28 Printscreen prostředí kreslení USE CASE v MS Visio, zdoj vlastní zpracování v MS VISIO

SHRNUTÍ KAPITOLY

V této kapitole je uveden přehled software pro vytváření diagramů UML a využití metodiky RUP. Zvláštní pozornost je věnována software ENTERPRISE ARCHITECT firmy Sparx. S tímto software počítáme do budoucna jako nejdůležitější a počítá se pořízením minimálně dvaceti plovoucích licencí. Software IBM Rational Architect je dostupný ve verzi z roku 2008 přes „tenký klient“ CITRIX. Tento software bude pouze demonstrativní s využitím postupů RUP.

Software VISIO obsahuje několik základních šablon pro kreslení UML diagramů. Je postačující pro psaní seminárních prací, pro psaní bakalářských a diplomových prací bude nutno použít EA.

SAMOSTATNÝ ÚKOL

1) Nainstalujte si software Enterprise Architect firmy Sparx, trial verzi na 10 dnů a vyzkoušejte si nakreslit diagramy uvedené v tomto skriptu.

2) Vyzkoušejte vzdálený přístup přes VM Ware software k programu Visio a vyzkoušejte si nakreslit diagramy v tomto skriptu

KONTROLNÍ OTÁZKA

Který ze software Visio a Enterprise Architect firmy Sparx umožňuje z diagramů generovat programový kód?

ODPOVĚDI

Enterprise Architect firmy Sparx.

9 RUP - RATIONAL UNIFIED PROCESS

RYCHLÝ NÁHLED KAPITOLY

Tato kapitola seznamuje s metodikou vytváření informačních systémů s názvem Rational Unified Process (dále RUP).

CÍLE KAPITOLY

Cílem kapitoly je vysvětlit metodiku RUP.

ČAS POTŘEBNÝ KE STUDIU

120 minut

KLÍČOVÁ SLOVA KAPITOLY

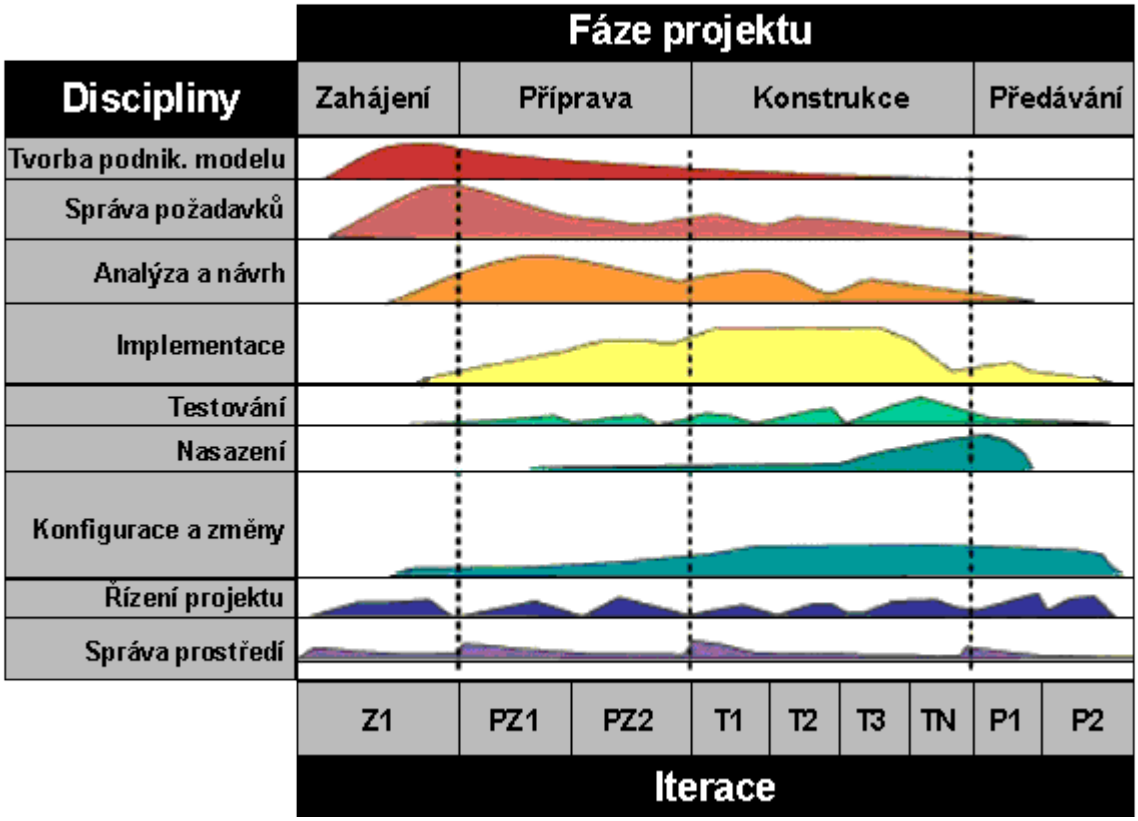
RUP, metodika, informační systém, vývoj sw, iterace, správa požadavků, komponenta, zahájeni, inception, příprava, elaboration, konstrukce, construction, předávání, transition

PRŮVODCE STUDIEM

Metodiku Rational Unified Process Rup (RUP) vytvořila společnost „Rational Software Corporation“. Tato metodika byla distribuována formou softwarového produktu, který převzala i se společností v roce 2003 firma IBM. Nynější podobu metodiky a její realizaci pomocí software najdete na stránkách http//www.ibm.com - do vyhledání na stránkách nutno zadat slovo „rational“. Nutno poznamenat, že RUP je komerční produkt na rozdíl od obecné metodiky Unified Process.

Metodika RUP (komerční produkt) a UP (obecná metodika)

Metodiku RUP názorně ilustruje grafické znázornění disciplín a fází projektu na obrázku 29:



Obrázek 29: Schéma projektu dle metodiky RUP, zdroj: [Arl2008]

Pro modelování procesů v životním cyklu projektu se využívá UML

Pro modelování procesů v životním cyklu projektu – vývoje informačního systému se využívá diagramů jazyka Unified Modeling Language (UML).

Metodika RUP byla popsána v předchozím učebním textu [FRA2014]. Tento učební text obsahuje komplexní popis metodiky RUP, resp. UP a bude nedílnou součástí e-learningového kurzu k výuce předmětu „Objektové metody modelování“.

SHRNUTÍ KAPITOLY

Tato kapitola je věnována popisu metodiky tvorby informačních systémů. Samotné UML se svými diagramy nejsou metodikou. Ty metodice pomáhají při tvorbě software. Veškeré studijní materiály o metodice RUP a srovnání s UP jsou obsaženy v [FRA2014].

OTÁZKY

Zkratka UP znamená

Vyberte jednu z nabízených možností:

a. Unified process

b. Unified program

c. Universal process

Co je UP?

Vyberte jednu z nabízených možností:

a. Soubor typových řešení iterativní metodologie vývoje SW resp. IS

b. Objektově orientovaná iterativní metodologie vývoje SW resp. IS

c. Jiný název pro UML diagramy

Axiomem metodiky UP není

Vyberte jednu z nabízených možností:

a. Zásada řízení případem užití a rizikem

b. Zásada soustředění na programování

c. Zásada iterace a přírůstku

Pět základních pracovních postupů (workflow)

Vyberte jednu z nabízených možností:

a. Plánování, Analýza, Návrh, Implementace, Testování

b. Analýza, Návrh, Programování, Implementace a Testování

c. Požadavky, Analýza, Návrh, Implementace a Testování

Fáze metodiky UP jsou

Vyberte jednu z nabízených možností:

a. Milník, Fáze, Iterace a Zavedení

b. Zahájení, Konstrukce, Zavedení a Testování

c. Zahájení, Rozpracování, Konstrukce a Zavedení

Software je v metodice UP vytvářen v iteracích

Vyberte jednu z nabízených možností:

a. Každá iterace je mini projekt

b. Iterace jsou skládány jedna za druhou, ale nemají vliv na konečnou podobu systému

c. Iterace jsou aplikovány jen na celý projekt

Každá iterace generuje

Vyberte jednu z nabízených možností:

a. Baseline

b. Přírůstek programového kódu

c. Helpline

Tvrzení "Pro každý nový projekt tvorby SW je třeba vytvořit novou instanci metodiky UP" je pravdivé

Vyberte jednu z nabízených možností:

a. ANO

b. Někdy ANO někdy NE - záleží na použití metodiky UP

c. NE

Metodika UP a RUP se liší

Vyberte jednu z nabízených možností:

a. Sémantikou elementů jednotlivých metod

b. Jsou to úplně odlišné metodiky

c. RUP vykazuje určité terminologické a syntaktické odlišnosti

Metodika UP a RUP jsou rigorózní nebo agilní?

Vyberte jednu z nabízených možností:

a. agilní i rigorózní

b. agilní

c. rigorózní

ODPOVĚDI

a. b. b. c. c. a. a. a. c. c.

---

10 PRAKTICKÉ PŘÍKLADY VYUŽITÍ UML

PŘÍPADOVÉ STUDIE

Tato kapitola přináší případové studie – čerpající příklady ze seminárních prací, které vznikly při výuce předmětu „Úvod do objektového modelování“ v letech 2012 - 2017. Kapitola je zařazena proto, aby studenti inovovaného předmětu „Metody objektového modelování“ měli inspiraci při tvorbě seminárních prací. Vznikla tak sbírka řešených příkladů. Zde uvedené řešené příklady – případové studie poslouží pro praktické procvičení uplatnění UML diagramů při návrhu IS. Jsou zde obsažena různá témata vesměs ze zkušenosti studentů, ať již z tvorby IS nebo jejich provozování. Každý student si vyzkoušel modelovat část IS s využitím UML diagramů. Nejčastěji studenti využívají MS VISIO, méně často pak Enterprise Architect firmy Sparx.

10.1 Užití skladového informačního systému

Úvod

Tématem práce je navrhnout modul informačního systému pro sklad. Základní požadavky jsou, aby mohl jakýkoliv libovolný pracovník pomocí IS zažádat o vyskladnění, naskladnění výrobku a skladník, pracující ve skladu mohl zboží připravit a oznámit, že je zboží připraveno.

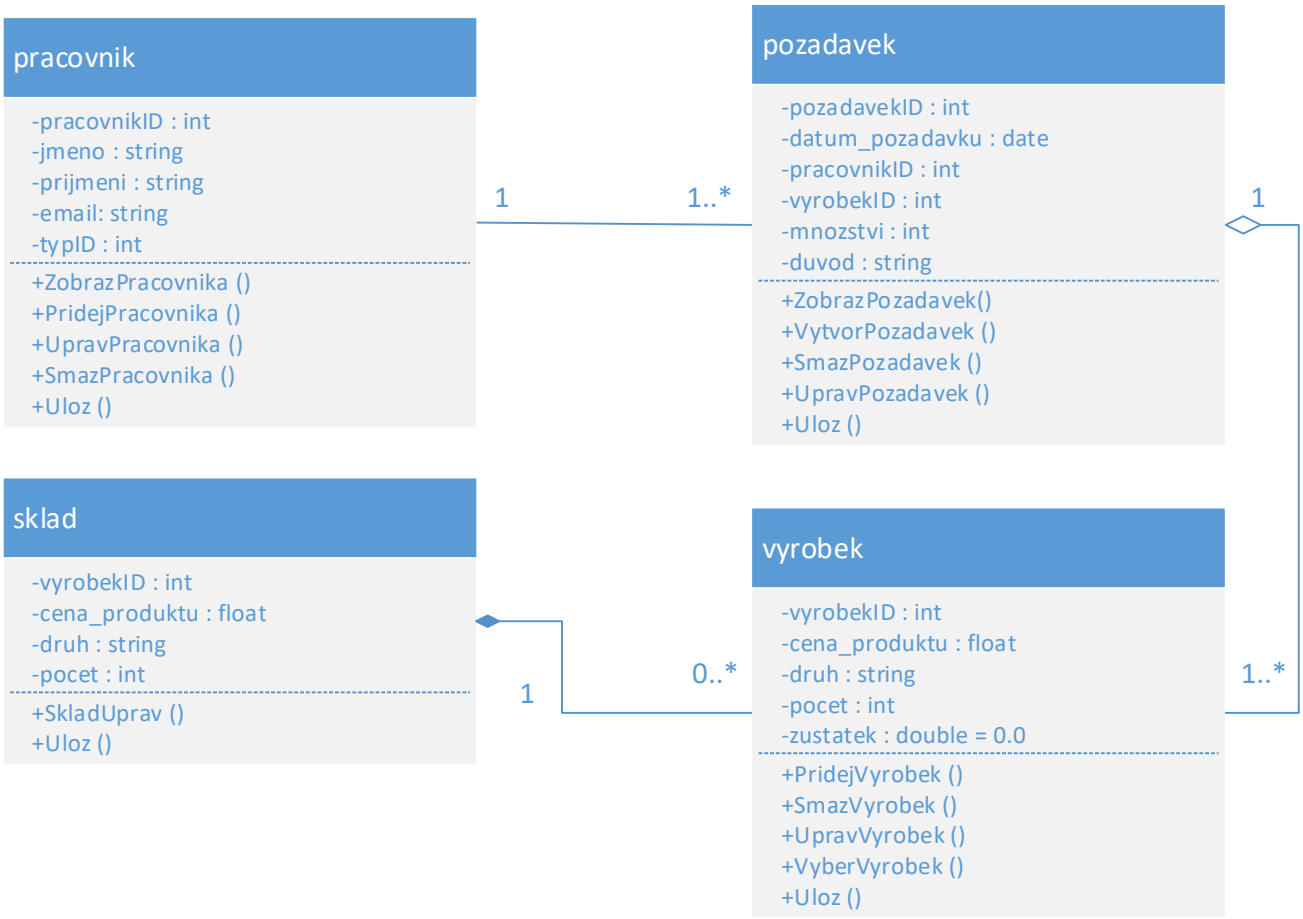
Modul by měl být zaveden především z důvodu, že sklad a ostatní pracovitě se nachází v odlišných lokalitách. Tím společnost XY ušetří náklady a čas nutný ke zbytečným výjezdům.

Cílem práce není za napsání kódu modulu, nýbrž vytvoření jeho stručného konceptu za pomocí diagramu tříd, use case diagramu, diagramu aktivit a pro doplnění sekvenčním diagramem. Diagramy byly vytvářeny za pomocí software Microsoft Office 2013, Visio 2013.

DIAGRAM TŘÍD

První část práce je věnována diagramu tříd.

Diagram č. 1: Diagram tříd modulu skladu



Zdroj: vlastní zpracování

Diagram pro modul skladu obsahuje základní čtyři třídy. Každá třída má vlastní název, který je napsán v modré hlavičce. Ve střední části jsou uvedeny atributy třídy a spodní část obsahuje metody.

TŘÍDY

Třída **pracovník** představuje jak pracovníka, který zadává požadavky, tak i skladníka, který požadavky vyřizuje. Pro jejich odlišení se využívají odlišná práva, která zajišťuje atribut typID. Třída umožňuje pracovníka za pomocí metod zobrazit, upravit smazat a ukládat změny.

Třída **pozadavek** představuje katalog požadavků, kam pracovníci ukládají své žádosti ohledně skladu. Tyto požadavky je možno díky metodám zobrazovat, vytvářet, mazat, upravovat a ukládat.

Třída **vyrobek** představuje výrobky, které jsou uloženy ve skladu a jsou požadovaný pracovníky za pomocí katalogu požadavků. Výrobky lze opět zobrazovat, upravovat, přidávat a mazat, jednotlivé kroky lze ukládat.

Třída **sklad** představuje reálný sklad, v němž jsou uloženy výrobky. Ten lze za pomocí metod upravovat a změny ukládat.

Vztahy

V diagramu tříd jsem využil 3 základní vztahy mezi třídami. Jsou to kompozice, agregace asociace.

Asociace je vztah mezi třídami, který specifikuje spojení mezi jejich instancemi. Tento vztah jsem využil mezi třídami pracovnik a pozadavek.

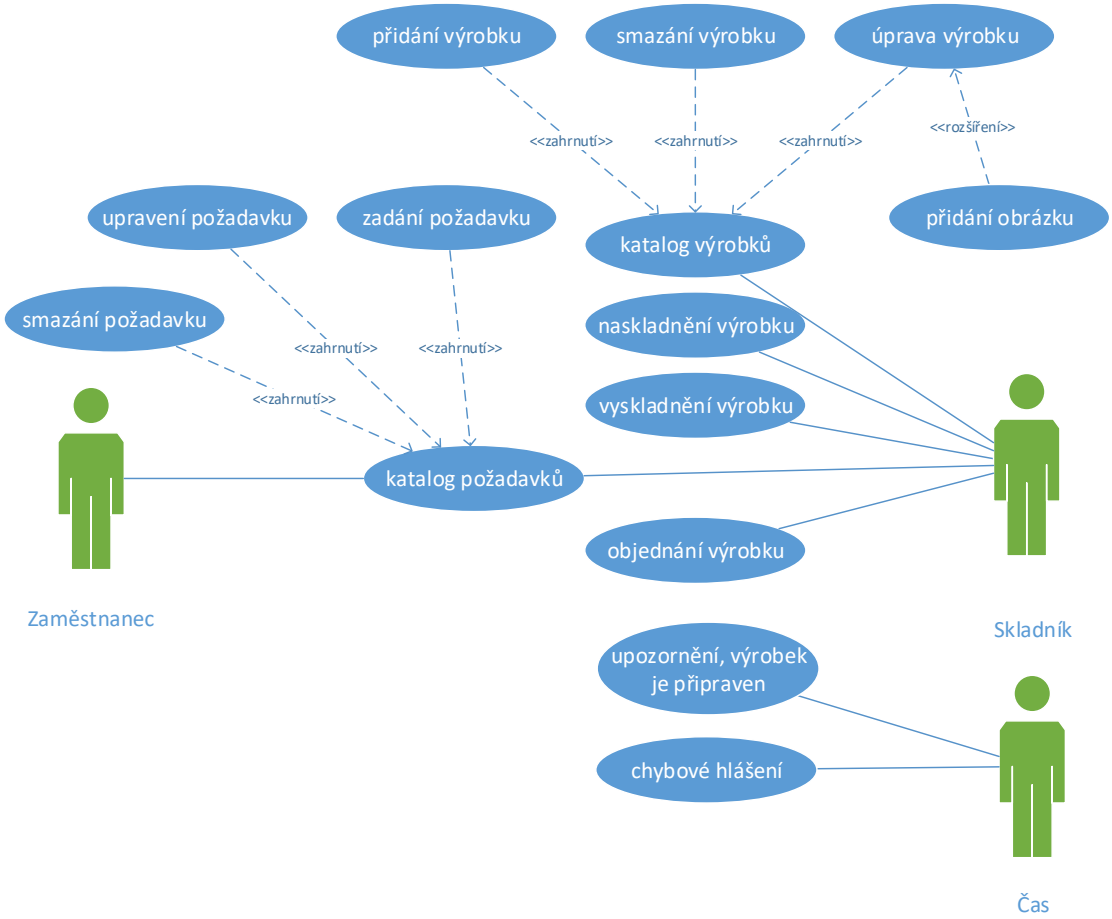
Kompozice je určitá forma asociace, kde je vyjádřen vztah celek část. Tento vztah je využit mezi vyrobkem a pozadavkem.

Agregace je silnější vztah než asociace, *„při zániku kontejneru automaticky rušíme i daný element.“* V diagramu jej nalezneme mezi výrobkem a skladem.

PŘÍPADY UŽITÍ – USE CASE DIAGRAM

„Use case diagram“ zobrazuje informační systém jako celek a ukazuje, v jakých případech jej lze použít. Nejdůležitějšími prvky diagramu jsou hranice systému, aktéři, činnosti a vazby mezi nimi.

Diagram č. 2: Use case diagram modulu skladu



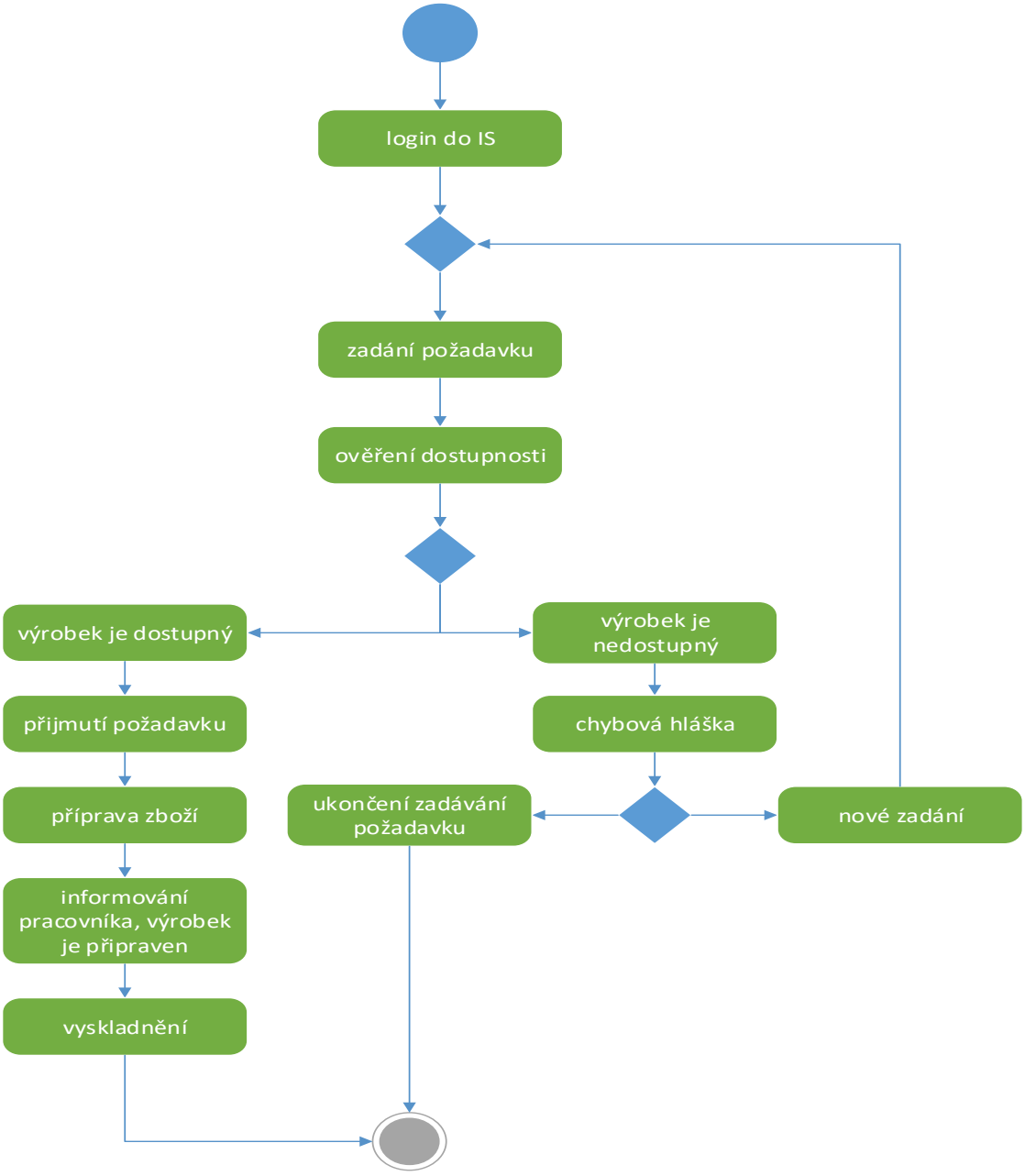
Zdroj: vlastní zpracování

Aktéři – v modulu sklad se vyskytují tři aktéři. Prvním je zaměstnanec, ten může zadávat požadavky do katalogu požadavků, dále je mazat a upravovat. Do katalogu požadavků má dále přístup i druhý aktér, skladník. Ten dále může výrobky naskladňovat, vyskladňovat, objednávat a má přístup do katalogu výrobků, kde může výrobky měnit, mazat, přidávat. Posledním aktérem je čas, který vykonává činnosti v určitém časovém bodě, např. poskytuje, že je výrobek připraven k vyzvednutí nebo podává chybová hlášení.

V „Use case diagramu“ byly použity relace zahrnutí (include) a rozšíření (extend). *„Relace <<include>> vyčleňuje kroky společníka několika případům užití do samostatného případu užití, který je následně do příslušných užití zahrnut“. „Relace <<extend>> je způsobem, jímž lze do existujícího případu užití vložit nové chování“*

DIAGRAM AKTIVIT

Diagram aktivit popisuje chování systému. Diagram popisuje vyskladnění výrobku.



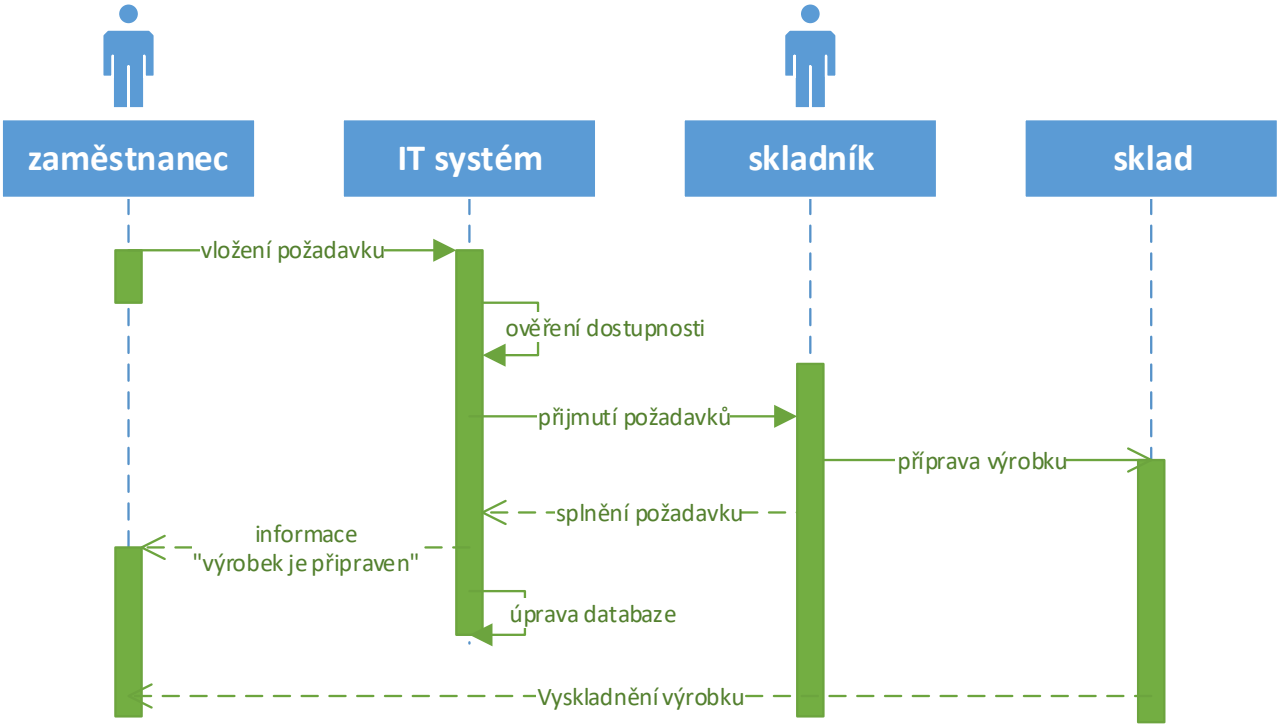
Zdroj: vlastní zpracování

Diagram se skládá z akcí, označených zelenými políčky, počátečního a koncového uzlu. Tok akcí je zobrazen za pomocí čar se šipkami. Symbol modrého kosočtverce označuje rozhodnutí nebo sloučení.

SEKVENČNÍ DIAGRAM

Sekvenční diagram ukazuje posloupnost daných akcí, jak následují postupně v čase. Diagram je navržen tak, aby byl dodržen základní logický koncept. Vynechání jedná části má ve většině případů zhroucení celého systému.

Diagram č. 4. ukazuje, jak vypadá úspěšné vyskladnění výrobku ze skladu.



Zdroj: vlastní zpracování

Modré pole s panáčkem vyjadřují životnost aktéra, modré políčka bez panáčka vyjadřují životnost objektů. Zelená svislá pole ukazují, kdy je objekt aktivní. Dané aktivity (sekvence) jsou pak značeny čárami se šipkami, které udávají směr. Plná čára označuje klasickou zprávu, přerušovaná čára pak zprávu návratovou a šipka zalomená zpět označuje zpětnou zprávu (např. IT systém sám zjistí, jestli je výrobek dostupný a sám upraví data v databázi).

ZÁVĚR

V této práci byl navržen modul skladu pro informační systém za pomoci programovacího jazyka UML. Zdárně byly vytvořeny všechny diagramy, které byly zpracovány v softwaru Microsoft Office 2013, Visio 2013. Cíle práce tedy byly splněny.

Práce přinesla mnoho nových zkušeností a poznatků. Projevila se zde také silná stránka UML, jakou jsou například standardizace a veliká šíře záběru. Šíří záběru je myšleno to, že v sobě ukrývá celý softwarový cyklus od návrhu až po fyzické nasazení. Velkou výhodou je také snadná úprava a rozšiřitelnost. Výhodou jazyka UML je, že podporuje i průmyslovou oblast přes nástroje CASE.

Jako slabou stránku lze považovat slabší důraz na grafické rozlišení. Některé prvky mají stejné grafické zobrazení, ale rozdílné vysvětlení. Mnoho uživatelů si taktéž stěžuje, že nelze na první pohled někdy z diagramu vyčíst, co přesně vyjadřuje, či jestli představuje třídy či objekty, jestli jsou vztahy dynamické nebo statické apod.

Celkově převládají dobré vlastnosti a lze jej využívat i v reálném prostředí pro dobře odvedenou práci. UML má stále co nabídnout a stojí za zvážení, zda se tento jazyk naučit podrobně a komplexně. Odměna v podobě dosažených výsledků bude jistě stát za to.

10.2 Analytická úloha pro společnost DERS

Úvod

Cílem této práce je opravit chyby, které způsobil bývalý analytik společnosti DERS při sestavování analýzy a návrhu informačního systému (dále jen IS) pro jednu nejmenovanou leteckou společnost. Úkol opravit tuto analýzu se skládá ze dvou dílčích úloh:

- opravit diagram tříd, který udává nepřesné informace, a
- pokud možno opravit a rozšířit Use Case diagram o scénář a upravit diagram případů užití.

Vzhledem k náročnosti tohoto úkolu se tato práce nebude zabývat částí teoretickou, teorie bude vysvětlena v rámci praktické části.

V této práci byly použity dva druhy vývojových prostředí. DERS používá vývojové prostředí Enterprise Architect 2012, tato práce využívá MS Visio 2016.

Představení společnosti DERS

Firma DERS je společností zabývající se tvorbou aplikací pro různá odvětví, od aplikací sloužících v sociálních službách, až po aplikace, které jsou „šité“ na míru zadavatelům. Vznikla v roce 1991 a její sídlo je v Hradci Králové.

V rámci **podpory vedení akademických pracovníků** se společnost zabývá např. správou kapacit výzkumné infrastruktury, správou zakázek a evidencí a vykazování publikační činnosti.

V rámci **oblasti koloběhů** je to například správa procesu nákupu, procesu pořízení majetku, ale i například procesu pracovních cest a jiných personálních činností.

V oblasti **sociálních služeb** se DERS zabývá hlavně podporou a řízení sítě sociálních služeb.

A konečně v **zakázkové činnosti** se firma zabývá specifickými problémy a žádostmi klientů.

DERS k úspěšnému vyřešení problémů a k dosažení cílů využívá hlavně těchto tří prvků:

- UX[1] – silná orientace na zákazníka;

[1] UX - dojem, který zůstane v naší paměti po interakci s lidmi, produkty a událostmi (pozn. autora).

- SimplifyWorks – framework na tvorbu aplikací;
- a konečně, jak sami v DERSu říkají, zdravý selský rozum a otevřený přístup.

Ve zkratce se dá napsat, že firma DERS se „zabývá jednoduchým softwarem, který odstraňuje papírové koloběhy a umožňuje se soustředit na odbornou práci“ (DERS 2016).

UVEDENÍ DO PROBLEMATIKY ÚKOLU

Obrázky skrývající se za odkazem u jednotlivých otázek zjednodušeně popisují systém fungování letiště z pohledu prvků (Class model) a jejich chování (Use case model):

Základní prvky systému:

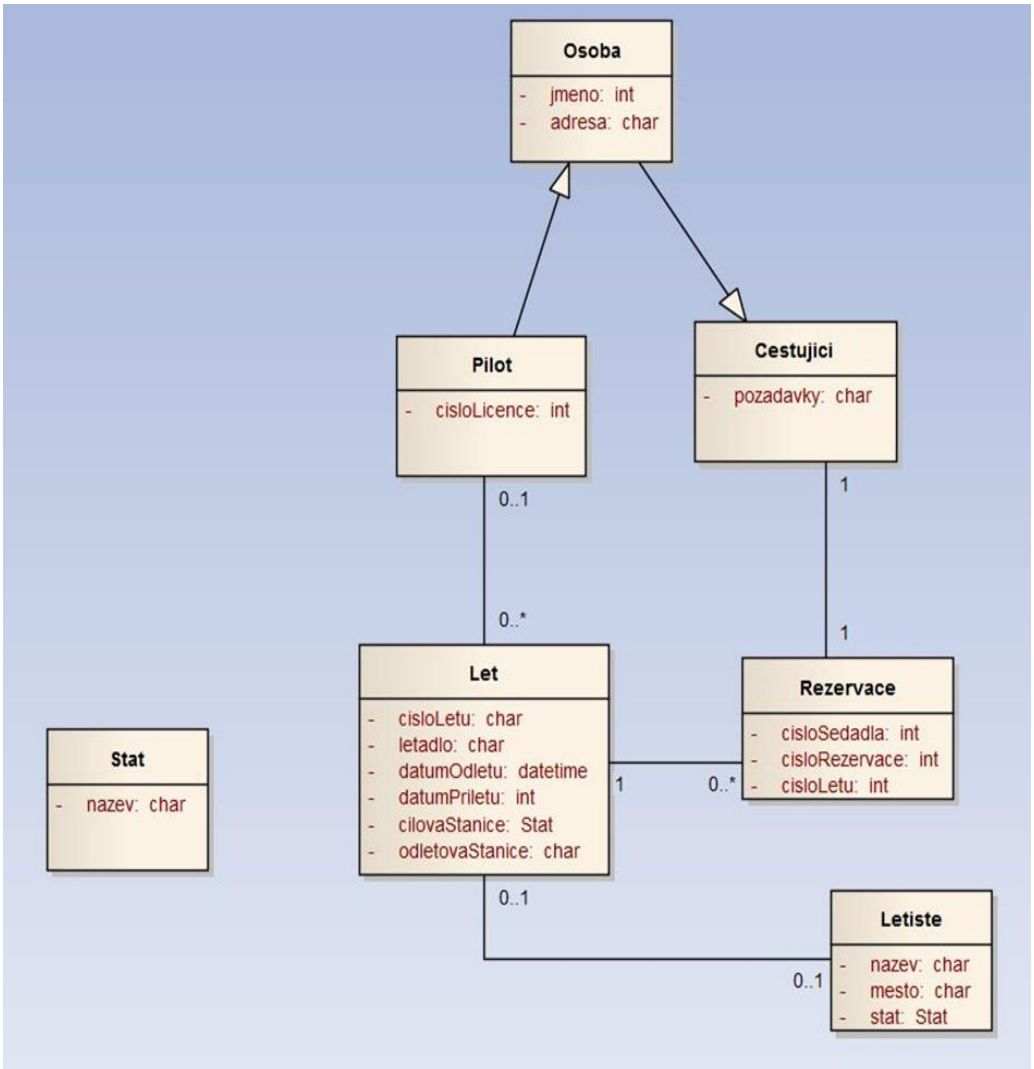
- Osoby (pilot, cestující)
- Let
- Rezervace
- Letiště
- Stát

Do každého letu se vždy musí přihlásit dva piloti, jinak se neletí. Systém jim následně odešle potvrzení o přihlášení. Cestující jsou povinni si zarezervovat místa v letadle – mohou mít na své jméno i více rezervací. Systém jim následně odešle potvrzení rezervace. U každého letu musí být kromě základních informací vždy uvedeno také odletové a cílové letiště + datum odletu. Letušky potřebují palubní seznam cestujících pro daný let, který obsluhují. Management letiště potřebuje 1x měsíčně získat přehled všech pilotů a jejich letů.

DIAGRAM TŘÍD

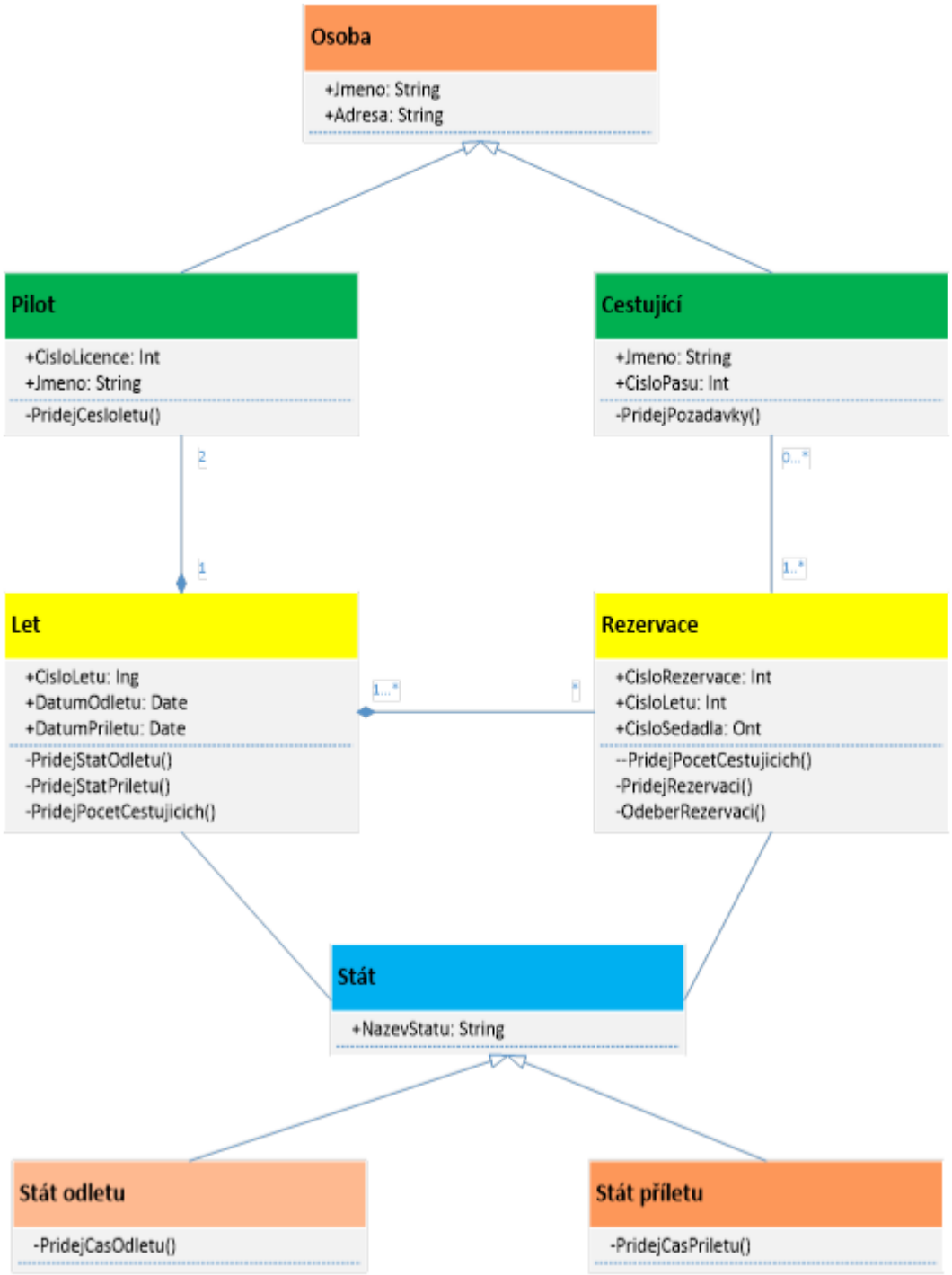
Firma DERS zaslala analytikům tento nepovedený diagram tříd:

Obrázek 1: Diagram tříd před úpravami



Jak je již z obrázku patrné, jsou zde chyby jak v atributech, tak v asociacích, Stát dokonce nemá žádnou relaci. Po důkladném rozboru tohoto diagramu tříd jsem vytvořil diagram tříd svůj, který vypadá takto:

Obrázek 2: Diagram tříd



Jak již je vidět z diagramu po úpravách, změnily se jak relace, tak i některé atributy a metody volání. První věcí, která se změnila, je dědičnost, která je dána mezi osobou, pilotem a cestujícím. Samozřejmě že je zde jasná dědičnost ve vztahu pilota a cestujícího vzhledem k osobě, která předává pilotovi a cestujícímu údaje o adrese a jméně. Vzhledem k tomu, že

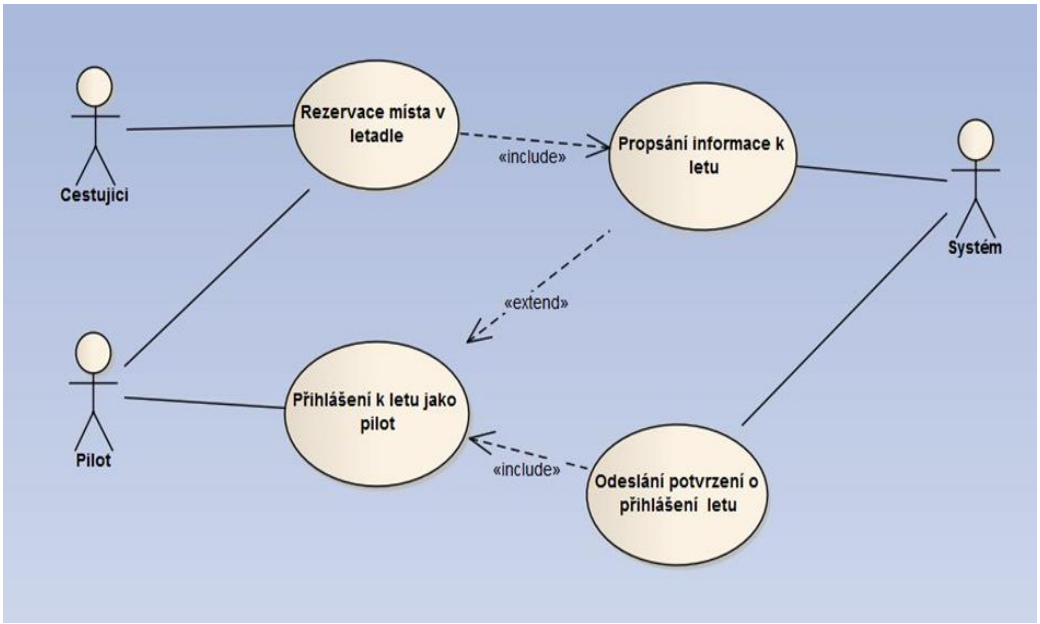
může dojít k omylům, nechává se u cestujícího i pilota jméno i v atributech těchto tříd. Dále se rozšířil, co se týká dědičnosti stát, který se dále rozdělil na stát odletu a příletu.

Vazby mezi třídami se vyřešili složenou agregací, vzalo se v potaz, že let je omezen nejen počtem cestujících, ale i kapacitou letu a je ovlivněn právě počtem pilotů, který musí být roven či větší než dva, což se týká i násobnosti v relaci pilot – let.

DIAGRAM PŘÍPADŮ UŽITÍ A SCÉNÁŘ PŘÍPADU UŽITÍ

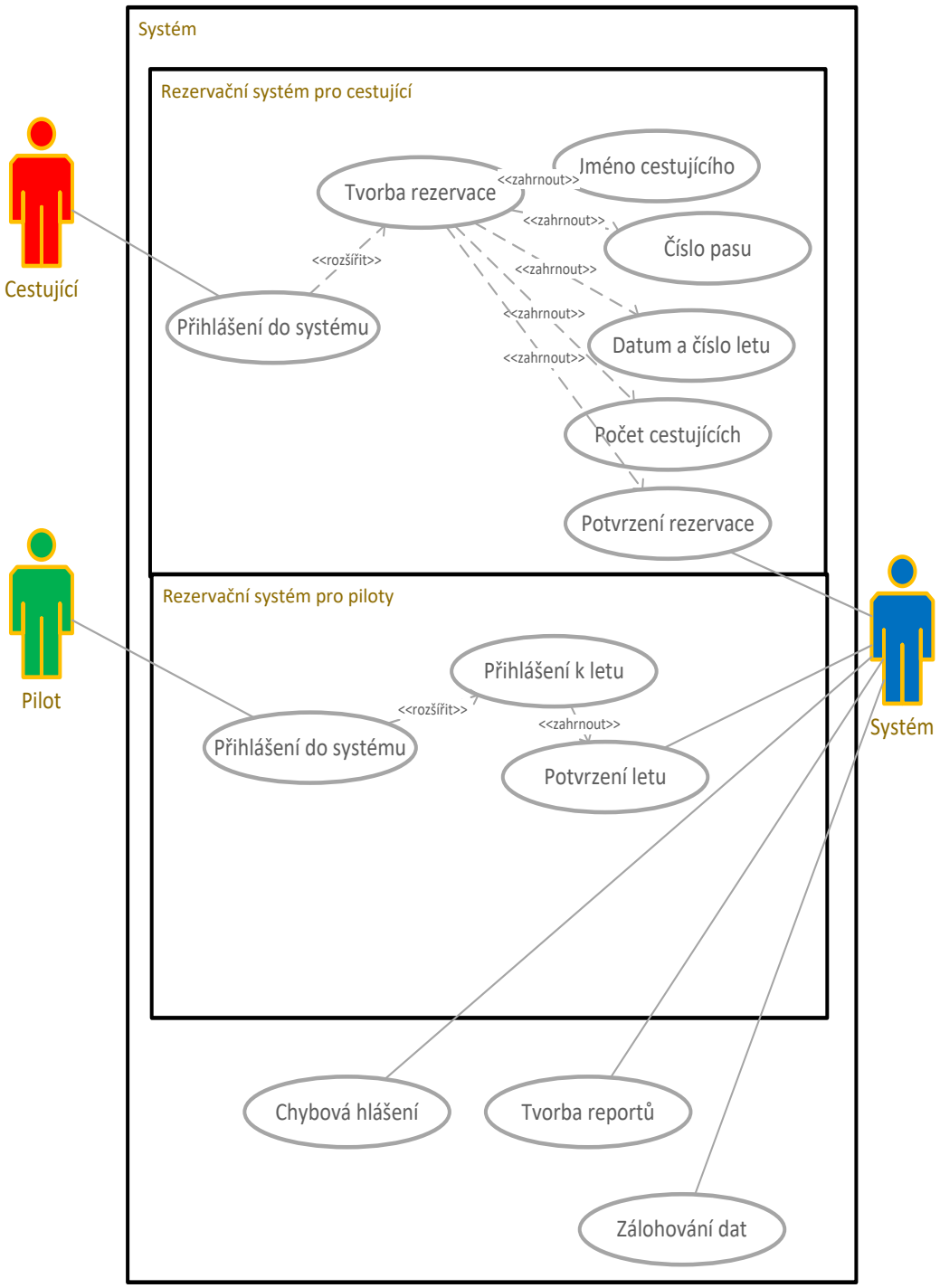
Analytik nechal v DERSu tento diagram případu užití:

Obrázek 3: Diagram případu užití před úpravami



Jak je již z diagramu vidět, analytik vůbec nedbal zadání a nevložil do diagramu hlášení systému o počtu pilotů a počtu letů minimálně 1x za měsíc. Po úpravách tohoto diagramu vzešel diagram tento:

Obrázek 4: Diagram případu užití po úpravách



Tento diagram případů užití se kompletně přeměnil, jelikož předešlý diagram byl neúplný, špatně zkonstruovaný a chyběly mu klíčové prvky, jako jsou např. subsystémy. Proto se rozhodlo o tom, že se vytvoří úplně nový, který je rozdělen na dva subsystémy, které na

sobě pracují nezávisle. Jen hlavní systém dokáže tyto dva subsystémy řídit. Z tohoto diagramu se také mohou vytvořit právě dva nezávisle na sobě pracující scénáře, jeden pro subsystém vytvářející rezervace pro cestující, druhý pro zaměstnance aerolinek, respektive v tomto případě pro piloty společnosti. Scénáře by mohly vypadat takto:

1. ***Scénář pro rezervaci letenek:***

Tabulka 1: Scénář rezervace letenek

1	Uživatel se přihlásí do systému
2	Systém potvrdí přihlášení
3	Uživatel vybere datum a číslo letu a zašle požadavek
4	Systém potvrdí požadavek na číslo letu a datum letu
5	Uživatel určí počet cestujících a zašle požadavek
6	Systém potvrdí počet cestujících
7	Uživatel potvrdí objednávku a uzavře rezervaci
8	Systém potvrdí rezervaci
9	Systém zašle potvrzující mail s platebními údaji

Mohla by nastat i situace, kdy je tento let již plně obsazen. Poté by mohl nastat alternativní scénář v bodě 4, kdy systém nahlásí již plnou obsazenost. V tomto případě by se scénář musel vrátit k bodu číslo 3 a uživatel by si musel najít jiné číslo a datum letu. Tento scénář je jen rychlý nástin do problematiky scénářů, samozřejmě kdyby se mělo jít do detailu, některé body tohoto scénáře by se daly rozklíčovat na více pod bodů (např. bod 3, bod 5, dále rozšíření bodu 7 apod.)

2. ***Scénář pro přihlášení pilotů na let***

Tabulka 2: Scénář pro přihlášení pilota k letu

1	Pilot se přihlásí do systému
2	Systém potvrdí přihlášení
3	Pilot vybere datum a číslo letu a zašle požadavek
4	Systém potvrdí požadavek na číslo letu a datum letu
5	Pilot potvrdí datum a číslo letu
6	Systém vloží pilota do databáze letů
7	Systém zašle zprávu pilotovi o přiděleném letu
8	Pilot potvrdí systému přidělený let
9	Systém znovu zašle potvrzující zprávu pilotovi o přiděleném letu

I zde je patrné, že systém bude generovat přidělování letu podle časové a pracovní náročnosti pilota. V reálu by to měl být ovšem systém, který bude přidělovat lety pilotům právě

podle časového a pracovního vytížení pilotů. Tato problematika ovšem vyžaduje více informací, které pro tuto práci ovšem nejsou dostupné, či nejsou v kompetenci píšícího tuto práci je použít.

Mohlo by se nyní podle scénářů postupovat ve tvorbě sekvenčních diagramů a diagramů aktivit, znovu ovšem tato práce naráží na své limity, které jsou ovlivněny firmou DERs.

Závěr

Problematika analýzy a návrhu IS je velmi široká a zahrnuje nejen znalosti z prostředí UML 2.0, popřípadě UML 2.5, je nutno znát také metodiky a metody, znát prostředí společnosti či firmy zadavatele, umět aplikovat nástroje Business procesů jako například BPMN 2.0 aj. Je třeba také spolupracovat s týmem kodérů, testerů, s lidmi, kteří tento systém budou do společnosti zavádět a hlavně je třeba komunikovat se zadavatelem, s lidmi, kteří tento systém budou používat, protože tyto informace jsou pro analytika klíčové. Když analytik tyto informace nemá nebo je dokonce nezná, dopadne to takovým způsobem, jako bylo vidět na diagramech, které byly použity v této práci.,

10.3 Podnikový prodej - vytvoření nové objednávky zákazníkem

SLOVNÍ POPIS PODNIKOVÉHO PROCESU

Jedním z nejdůležitějších podnikových procesů je Prodej. V našem případě se firma zabývá nákupem a následným prodejem zboží skrze E-commerce systém. Celý tento podnikový proces začíná tím, že vedení firmy zadá do informačního systému pokyn k zahájení marketingové kampaně. Informační systém informuje manažera prodeje o pokynu vedení firmy (např. současně skrz notifikační lištu v systému, sms, emailem, atd.). Manažer následně zadává do informačního systému pokyn, který je určený pro oddělení prodeje. Systém informuje oddělení prodeje o pověření k zahájení marketingové kampaně. Oddělení tedy zahájí marketingovou kampaň a kontaktuje potencionální zákazníky s nabídkou produktů (může se jednat o emailový kontakt, telefonický kontakt či jinou formu). Pokud má zákazník o zboží zájem, oddělení prodeje vloží do systému žádost o odeslání odkazu na katalog produktů, resp. webové rozhraní E-commerce systému. Systém následně odešle zákazníkovi zprávu obsahující odkaz na E-commerce systém. Později se zákazník naviguje na webové rozhraní E-commerce. V tom momentě E-commerce navyšuje počítadla návštěvnosti a informuje systém. Systém si to poznamená a v požadovaných intervalech (měsíčně, týdně, atd.) generuje sestavy návštěvnosti, které jsou určené vedení firmy. E-commerce po připojení zákazníka předkládá zákazníkovi žádost o registraci. Zákazník se registruje a E-commerce registraci potvrzuje. Poté je zákazníkovi předložen katalog zboží. Při výběru zboží zákazníka kontroluje E-commerce skrz dotaz na sklad, zda je zboží dostupné. E-commerce poté odesílá systému výběr zákazníka, který je určen k analýze potenciálně chtěného zboží, na které se později mohou vztahovat slevy. Systém opět generuje sestavy přehledu potencionálně chtěných výrobků a ty předkládá vedení firmy. Současně informuje manažera prodeje o potencionálně velké objednávce. Manažer prodeje konzultuje s vedením firmy případné slevy. Vedení firmy dává pokyn manažerovi k zadání hromadné slevy pro objednávku zákazníka. Manažer vkládá hromadnou slevu do E-commerce systému ke konkrétní objednávce. E-commerce generuje konečnou cenu, způsob dopravy a platby. Zákazník vybírá údaje. E-Commerce přesměrovává zákazníka na platební bránu. Zákazník provádí platbu. Platební brána informuje jak prodejce, tak zákazníka o provedené platbě. E-commerce generuje fakturu zákazníkovi a informuje systém o nové objednávce. Systém informuje oddělení prodeje o nové objednávce ke zpracování. Pro přehlednost je uvedený proces přepsán do tabulky:

Pořadí činnosti	**Kdo** (počáteční aktér)	**Akce**	**Cíl** (cílový aktér)
1.	**Vedení firmy**	*Zadá do informačního systému pokyn k zahájení marketingové kampaně*	**Systém**
2.	**Systém**	*Informuje manažera prodeje o pokynu vedení firmy*	**Manažer prodeje**
3.	**Manažer prodeje**	*Zadává do informačního systému pokyn určený pro oddělení prodeje*	**Systém**
4.	**Systém**	*Informuje oddělení prodeje o pověření k zahájení marketingové kampaně*	**Oddělení prodeje**
5.	**Oddělení prodeje**	*Kontaktuje potenciálního zákazníka ohledně katalogu produktů*	**Zákazník**
6.	**Zákazník**	*Projevuje zájem o zboží*	**Oddělení prodeje**
7.	**Oddělení prodeje**	*Vkládá do systému žádost o odeslání odkazu na katalog produktů*	**Systém**
8.	**Systém**	*Odesílá odkaz na katalog produktů*	**Zákazník**
9.	**Zákazník**	*Navigace na webové rozhraní*	**E-commerce**
10.	**E-commerce**	*Navyšuje počítadla návštěvnosti*	**Systém**
11.	**Systém**	*Generuje sestavu návštěvnosti webu*	**Vedení firmy**
12.	**E-commerce**	*Předkládá zákazníkovi žádost o registraci*	**Zákazník**
13.	**Zákazník**	*Provádí registraci*	**E-commerce**
14.	**E-commerce**	*Potvrzuje registraci*	**Zákazník**
15.	**E-commerce**	*Předkládá nabídku zboží*	**Zákazník**
16.	**Zákazník**	*Vybírá zboží*	**E-commerce**
17.	**E-commerce**	*Kontroluje dostupnost zboží na skladě*	**Sklad**
18.	**Sklad**	*Potvrzuje dostupnost zboží*	**E-commerce**
19.	**E-commerce**	*Odesílá výběr zákazníka k analýze*	**Systém**
20.	**Systém**	*Generuje sestavu potencionálně chtěného zboží*	**Vedení firmy**
21.	**Systém**	*Informuje manažera o potenciálně velké objednávce*	**Manažer prodeje**
22.	**Manažer prodeje**	*Konzultuje s vedením potenciální slevy objednávky*	**Vedení firmy**
23.	**Vedení firmy**	*Zadává pokyn k úpravě smluvních cen v objednávce*	**Manažer prodeje**
24.	**Manažer prodeje**	*Zadává hromadnou slevu pro objednávku*	**E-commerce**
25.	**E-commerce**	*Generuje konečné ceny, způsob dopravy a platby*	**Zákazník**
26.	**Zákazník**	*Vybírá možnosti*	**E-commerce**
27.	**E-commerce**	*Přesměrovává zákazníka na platební bránu*	**Zákazník**
28.	**Zákazník**	*Uskutečňuje platbu*	**Platební brána**
29.	**Platební brána**	*Informuje zákazníka o uskutečněné platbě*	**Zákazník**
30.	**Platební brána**	*Informuje E-Commerce o platbě*	**E-Commerce**
31.	**E-commerce**	*Generuje Fakturu*	**Zákazník**
32.	**E-commerce**	*Informuje systém o nové objednávce*	**Systém**
33.	**Systém**	*Informuje oddělení prodeje o objednávce ke zpracování*	**Oddělení prodeje**

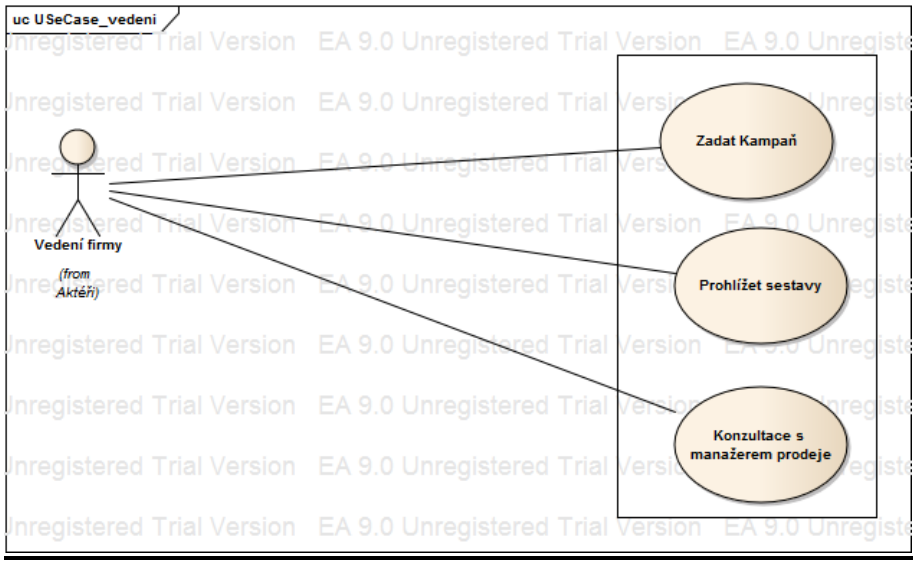
USE CASE DIAGRAMY

Celkově tedy máme v modulu vytvoření objednávky tyto aktéry, kteří vykonávají následující operace:

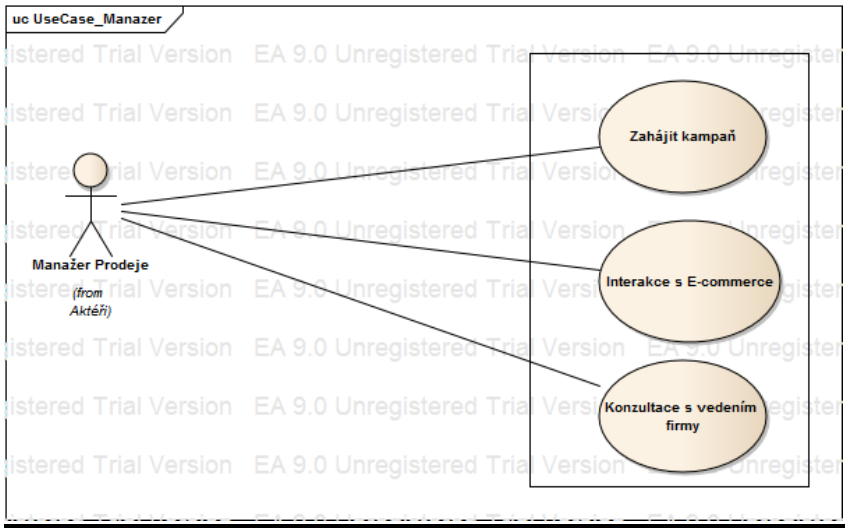
Aktér	Akce
Vedení firmy	*Zadat kampaň* *Prohlížet sestavy* *Konzultace s manažerem prodeje*
Manažer prodeje	*Zahájit kampaň* *Interakce s E-commerce* *Konzultace s vedením firmy*
Oddělení prodeje	*Spustit kampaň* *Zadat požadavek na odeslání produktu* *Řešit nové objednávky*
Systém	*Informovat o kampani* *Analyzovat výběr zákazníka* *Generovat sestavy*
E-commerce	*Interakce se zákazníkem* *Interakce se systémem* *Interakce s platební bránou* *Interakce s manažerem prodeje* *Interakce se skladem*
Zákazník	*Interakce s oddělením prodeje* *Interakce s E-commerce* *Interakce s platební bránou*
Platební brána	*Interakce se zákazníkem* *Interakce s E-commerce*
Sklad	*Interakce s E-commerce*

PŘEHLED USE CASE DIAGRAMŮ:

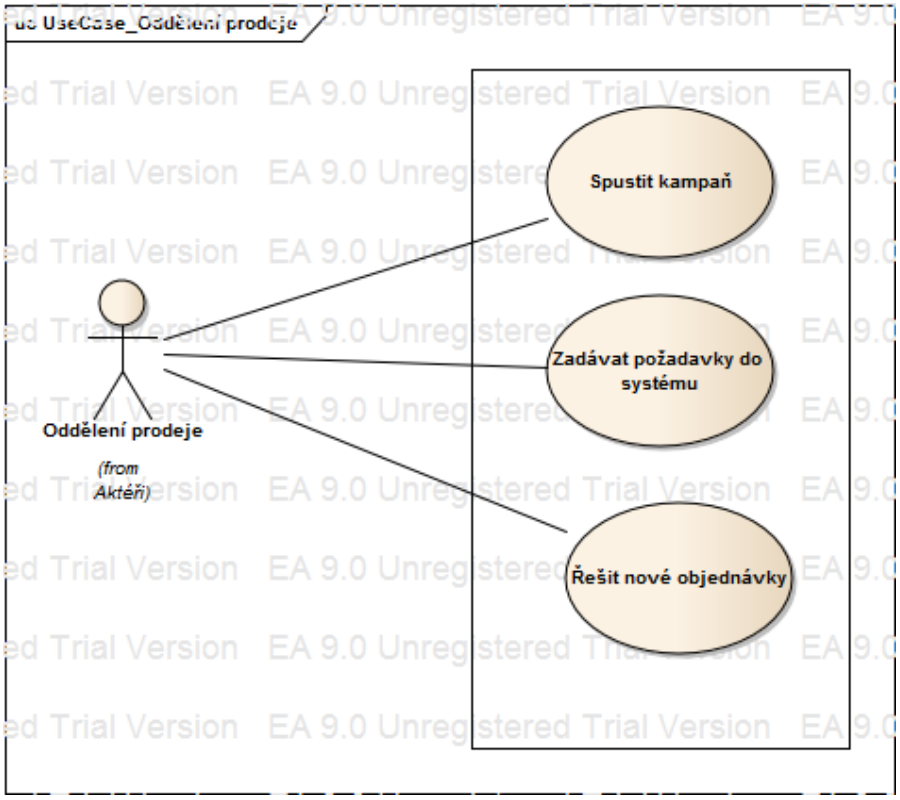
Vedení firmy:



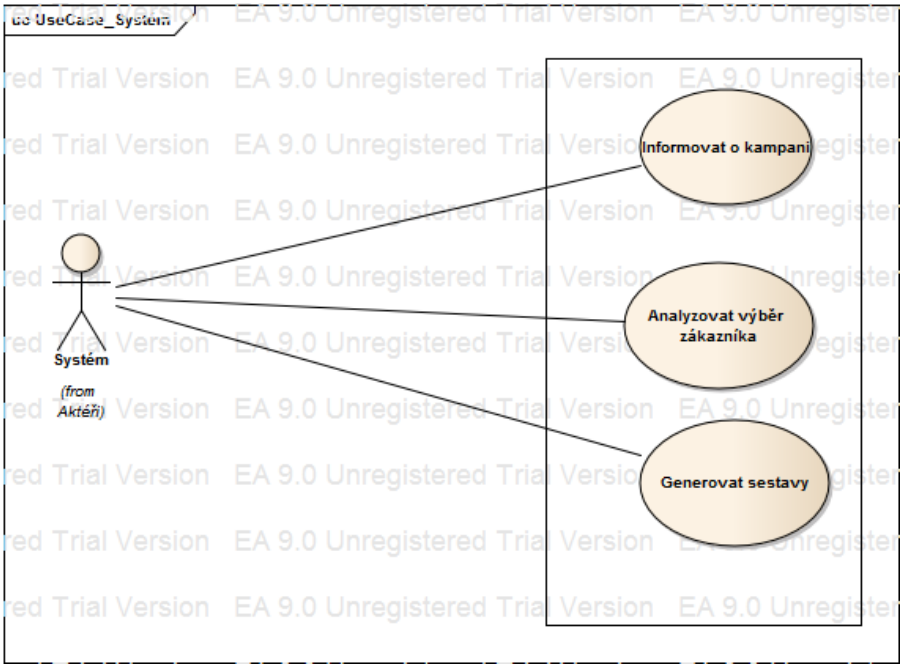
Manažer prodeje:



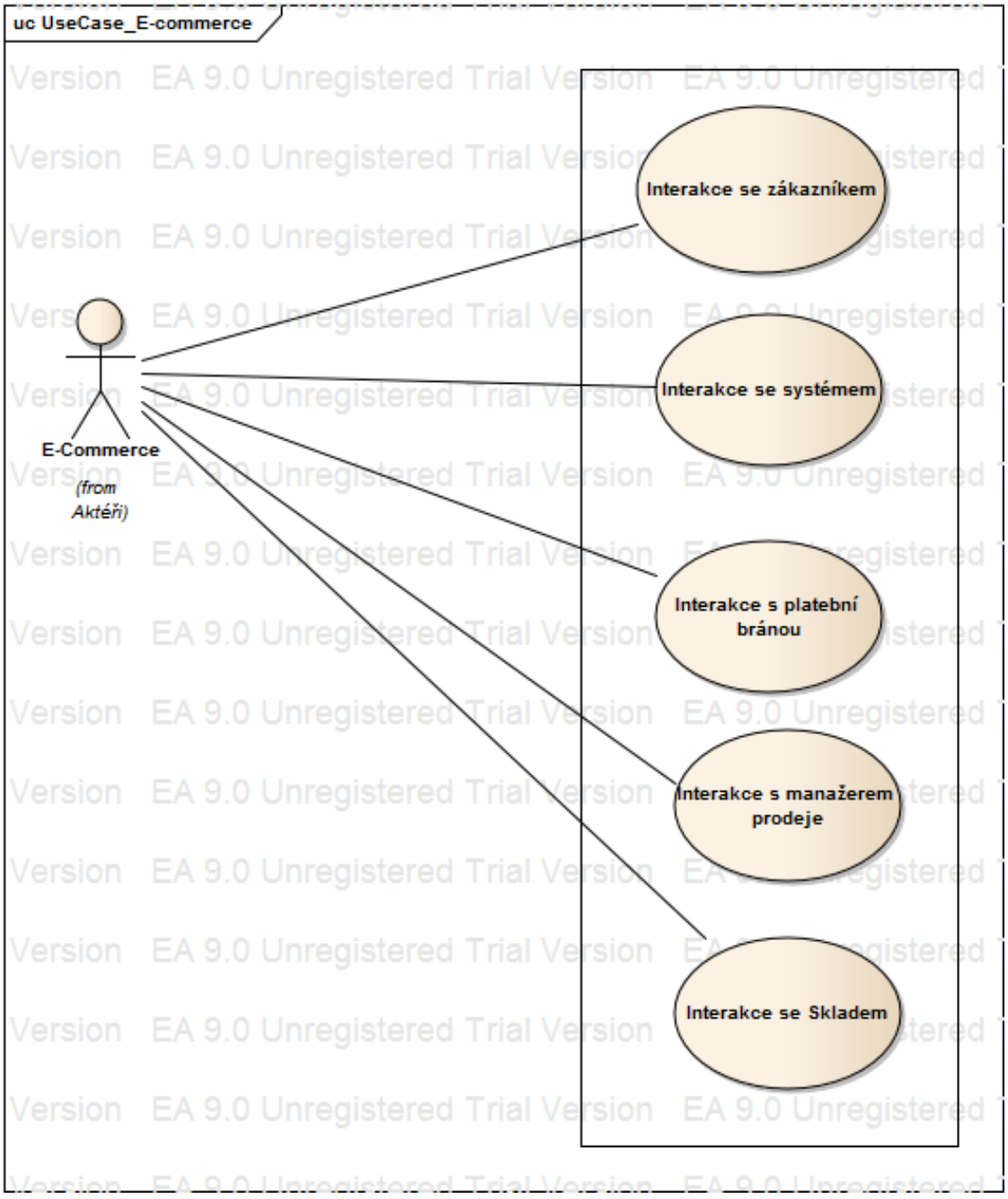
Oddělení prodeje:



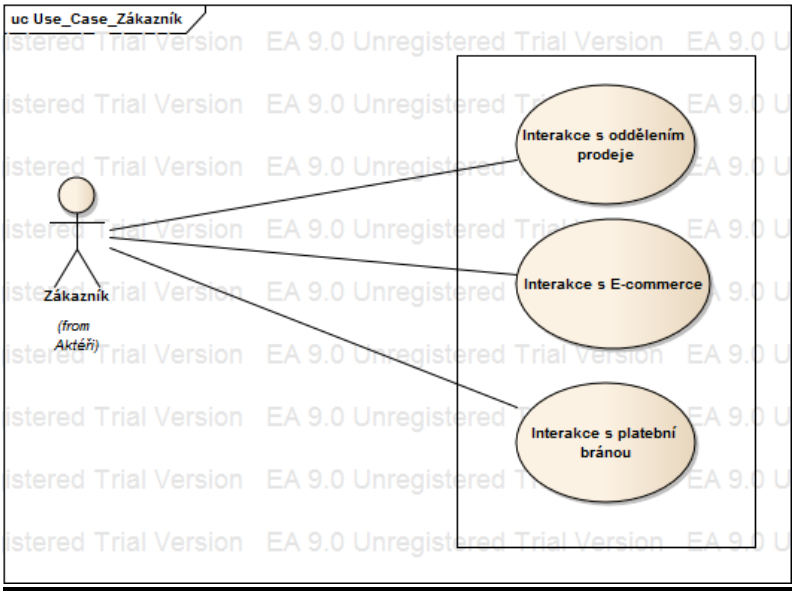
Systém:



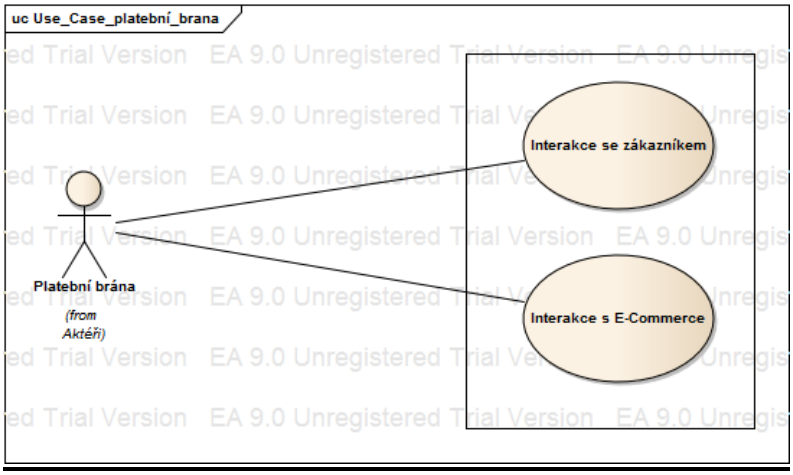
E-commerce


uc UseCase_E-commerce
Interakce se zákazníkem
Interakce se systémem
Interakce s platební bránou
Interakce s manažerem prodeje
Interakce se Skladem
E-Commerce
(from Aktéři)

Zákazník:



Platební brána:



Sklad:

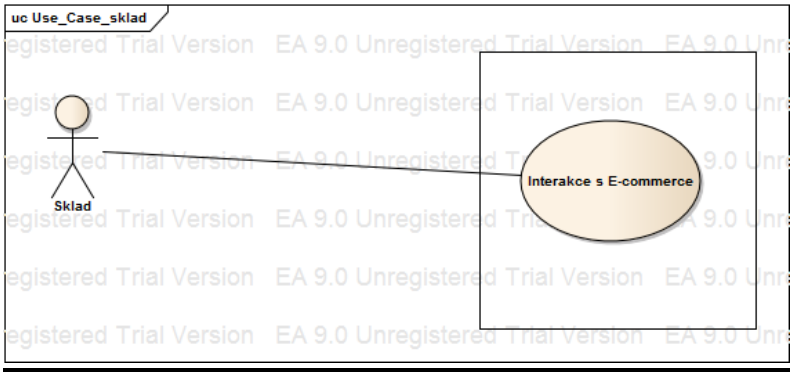
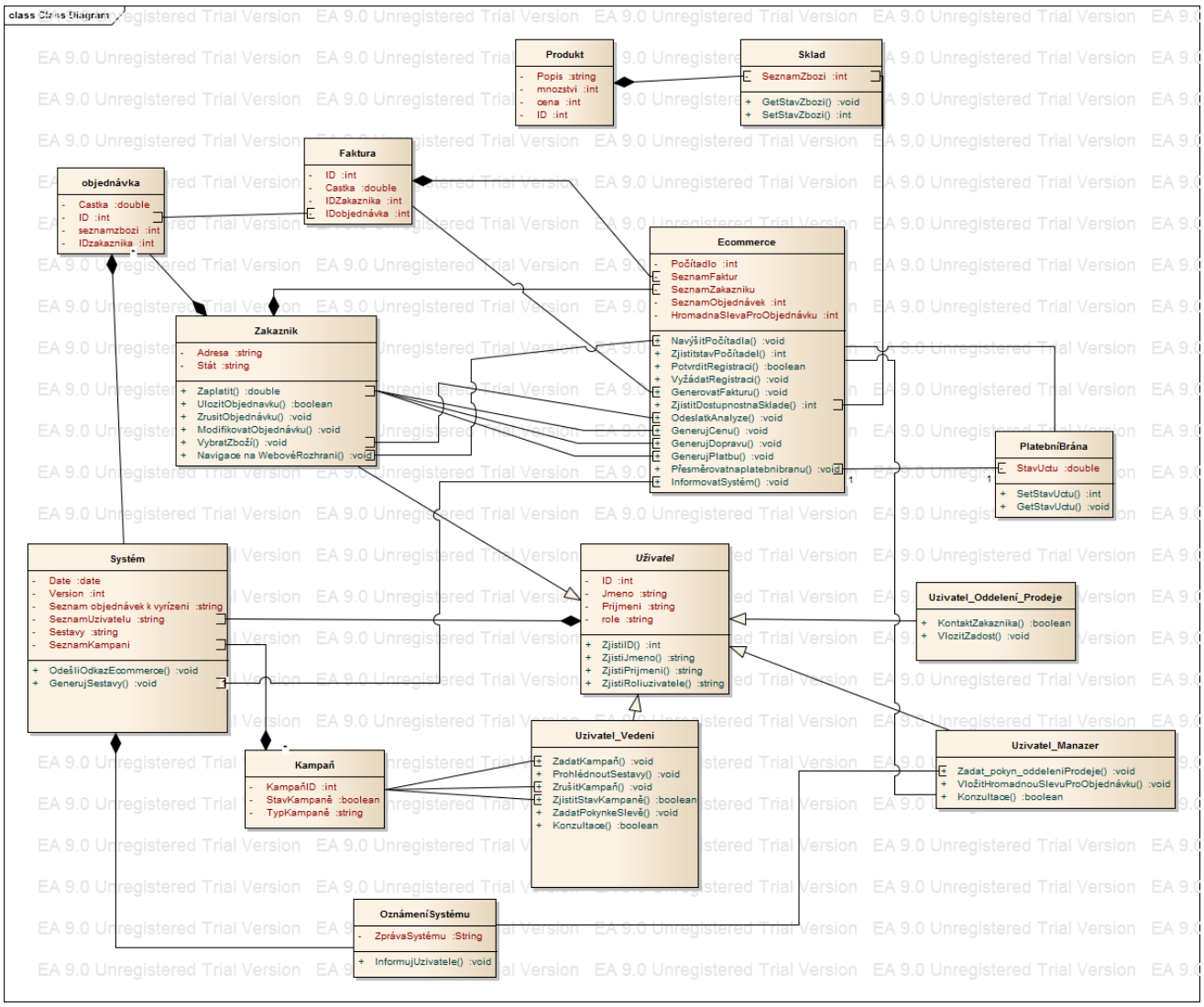


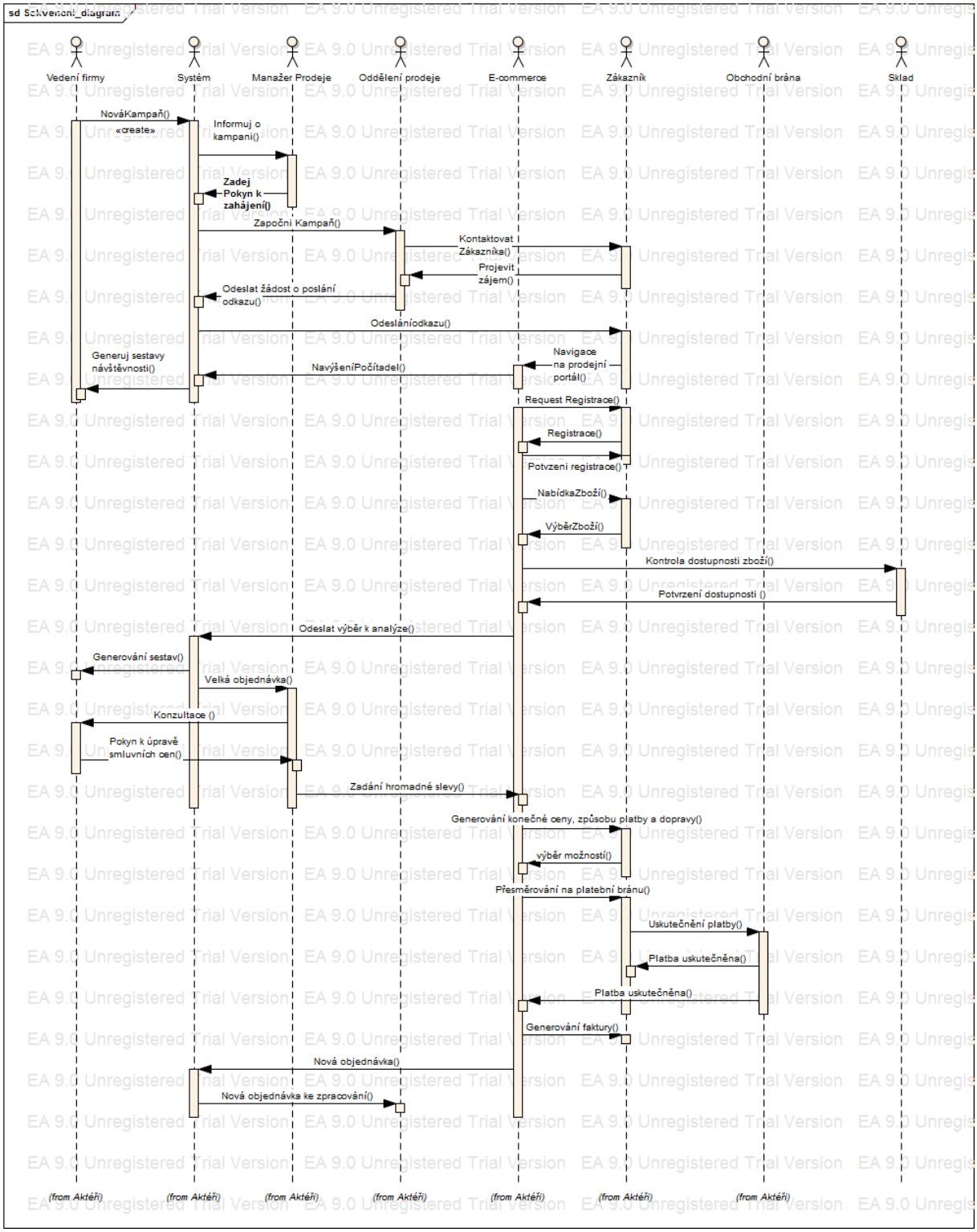
Diagram Tříd



Co se týče diagramu tříd, Výše zachycené vztahy mezi třídami, zahrnující asociaci, agregaci a kompozici, jedná se o jakýsi předběžný model, neboť ve finální verzi musí být mnohem propracovanější.

Sekvenční Diagram

Sekvenční diagram pokrývá za sebou jdoucí činnosti v podniku. V tomto diagramu mají jednotliví aktéři přiděleny plovoucí dráhy tzv. swimlines. Mezi těmito aktéry probíhají činnosti formou zpráv. Na rozdíl od předchozích diagramů, jedná se o diagram dynamický.


sd Sekvenční_diagram
Vedení firmy
Systém
Manažer Prodeje
Oddělení prodeje
E-commerce
Zákazník
Obchodní brána
Sklad
NováKampaň()
«create»
Informuj o kampani()
Zadej Pokyn k zahájení()
Započni Kampaň()
Kontaktovat Zákazníka()
Projevit zájem()
Odeslat žádost o poslání odkazu()
Odeslánímodkazu()
Generuj sestavy návštěvnosti()
NavýšeníPočítadel()
Navigace na prodejní portál()
Request Registrace()
Registrace()
Potvzení registrace()
NabídkaZboží()
VýběrZboží()
Kontrola dostupnosti zboží()
Potvrzení dostupnosti ()
Odeslat výběr k analýze()
Generování sestav()
Velká objednávka()
Konzultace ()
Pokyn k úpravě smluvních cen()
Zadání hromadné slevy()
Generování konečné ceny, způsobu platby a dopravy()
výběr možností()
Přesměrování na platební bránu()
Uskutečnění platby()
Platba uskutečněna()
Platba uskutečněna()
Generování faktury()
Nová objednávka()
Nová objednávka ke zpracování()
(from Aktéři)
(from Aktéři)
(from Aktéři)
(from Aktéři)
(from Aktéři)
(from Aktéři)
(from Aktéři)

Závěr

Cíl práce byl splněn, všechny předem vytyčené diagramy byly zkonstruovány. Je nutno podotknout, že v případě budoucí modifikace těchto diagramů, bude třeba snížit míru abstrakce a zaměřit se na danou doménu podrobněji.

10.4 Rezervační systém kulturních akcí

Úvod

V seminární práci popisuji, jak funguje systém pro objednávání vstupenek na kulturní akce pomocí internetového portálu. Proces zahrnuje dvě zákaznické role. Jednu jako neregistrovaný uživatel a druhou jako uživatel registrovaný. Od závislosti na registraci či ne registraci zákazníka se pak odvíjí možnosti činností, které daný zákazník může v systému vykonávat. Dále zde máme roli administrátora, který potvrzuje, mění a ruší objednávky, spravuje nabídku akcí a odpovídá na dotazy.

Tyto procesy vizuálně modeluji pomocí UML diagramů, které jsem vytvořila pomoci programu Microsoft Visio 2010.

Volba tématu z prostředí kultury byla pro mne nasnadě, jelikož ráda navštěvuji všelijaké kulturní akce. Často proto využívám rezervaci a objednávání vstupenek na různá divadelní představení, koncerty a plesy přes internet. Jedná se o velmi mobilní, přehlednou a pohodlnou variantu, která se těší stále větší oblibě.

Diagram tříd

Diagram tříd zachycuje proces evidence dat o zákaznících a objednávkách. U zákazníků, kteří se chtějí registrovat, se evidují údaje o registraci a přihlášení. U objednávek se evidují bližší informace o dané kulturní akci, ke které se vztahují informace o sálu, ve kterém se akce pořádá.

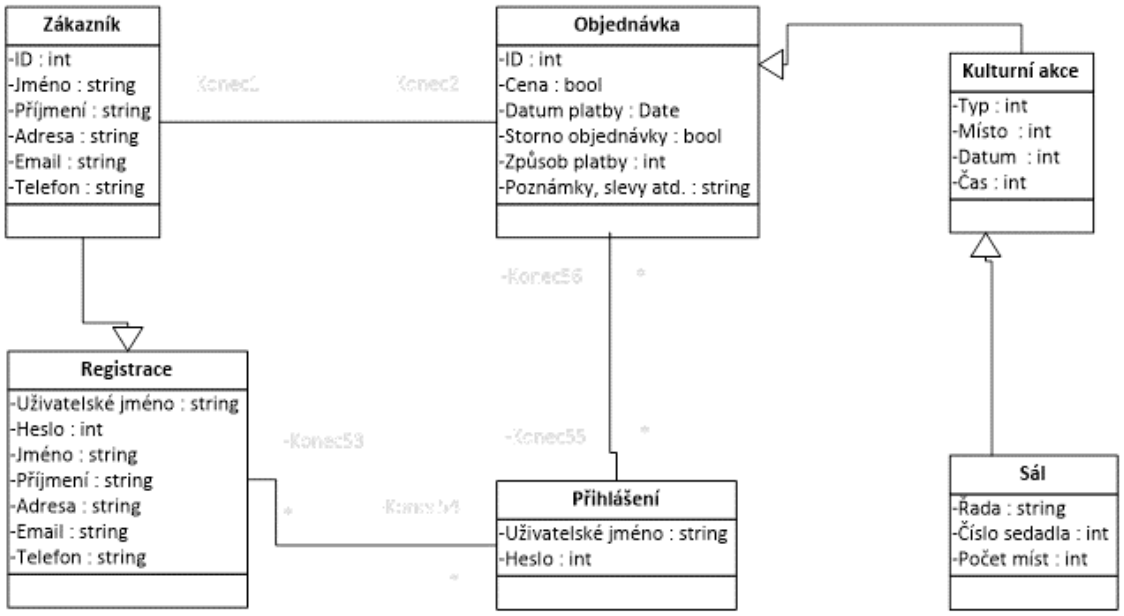


DIAGRAM PŘÍPADU UŽITÍ

Digram případu užití zobrazuje proces vytváření, rušení a potvrzení objednávek vstupenek na kulturní akce v rámci portálu. V diagramu jsou zobrazeny tři role lidí a systém. Jsou zde dvě uživatelské role, přičemž dané role mají různé možnosti činností v systému a to v závislosti na přihlášení nebo nepřihlášení uživatele.

Běžný uživatel

Běžný uživatel má možnost prohlížení akcí, jejich objednání a také registraci.

Registrovaný uživatel

Registrovaný uživatel má kromě možnosti prohlížení akcí a jejich objednání, také k dispozici přehled svých objednávek, s možností měnit je a stornovat. Dále může ze svého rozhraní položit dotaz administrátorovi.

Administrátor

Administrátor vytváří přehled kulturních akcí, provádí jejich změny, rušení a aktualizace. Dále registruje uživatele, potvrzuje a ruší objednávky, generuje pro uživatele rozhraní přehledu vlastních objednávek a zodpovídá dotazy. Je třeba říci, že role administrátora se prolíná s rolí systému. Typy činností jako potvrzení a rušení objednávek, správa uživatelova rozhraní, jeho registrace a podobně jsou řešeny automaticky pomocí systému. Člověka je

pak potřeba na správu celého systému, zadávání a změnu programu akcí, odpovídání na dotazy atd.

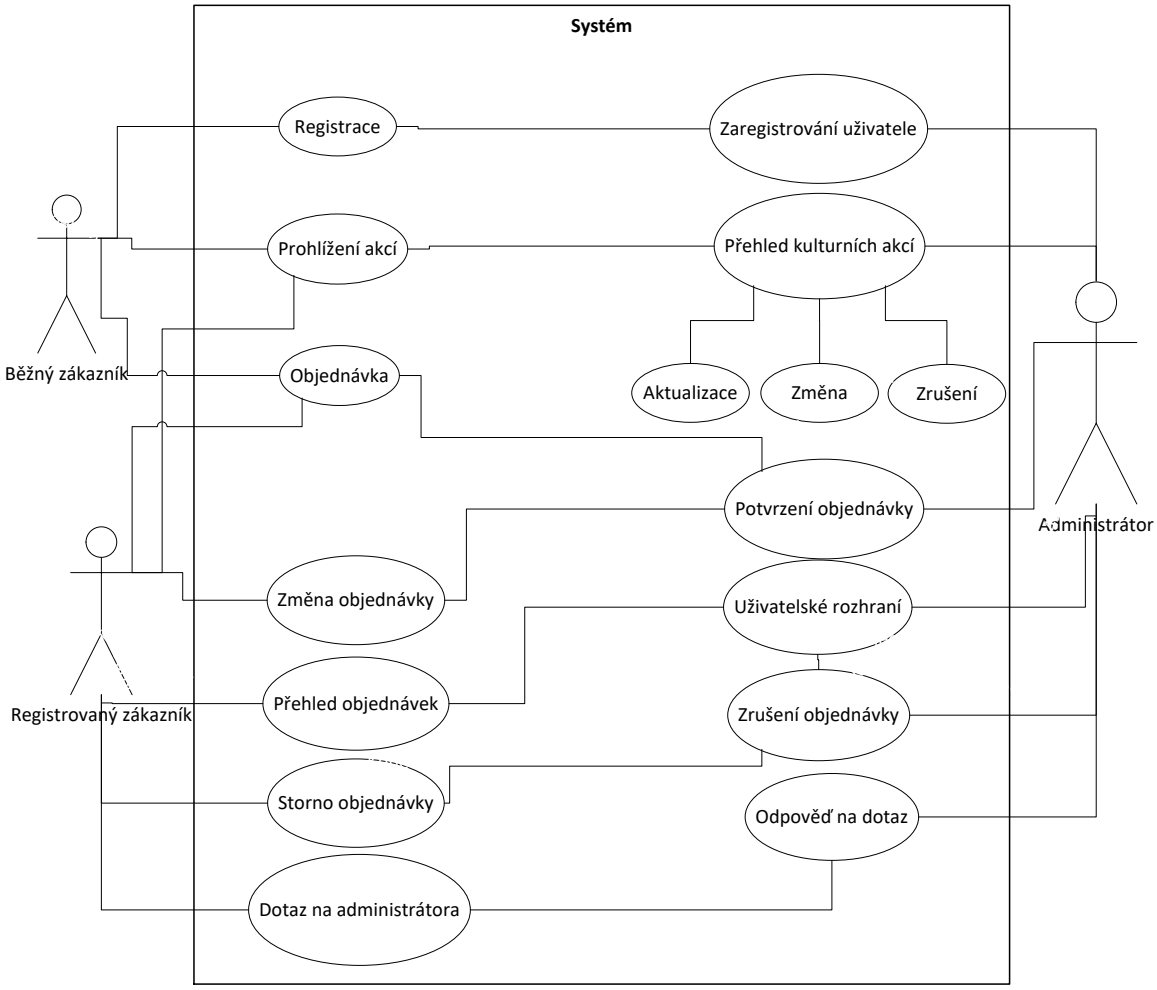
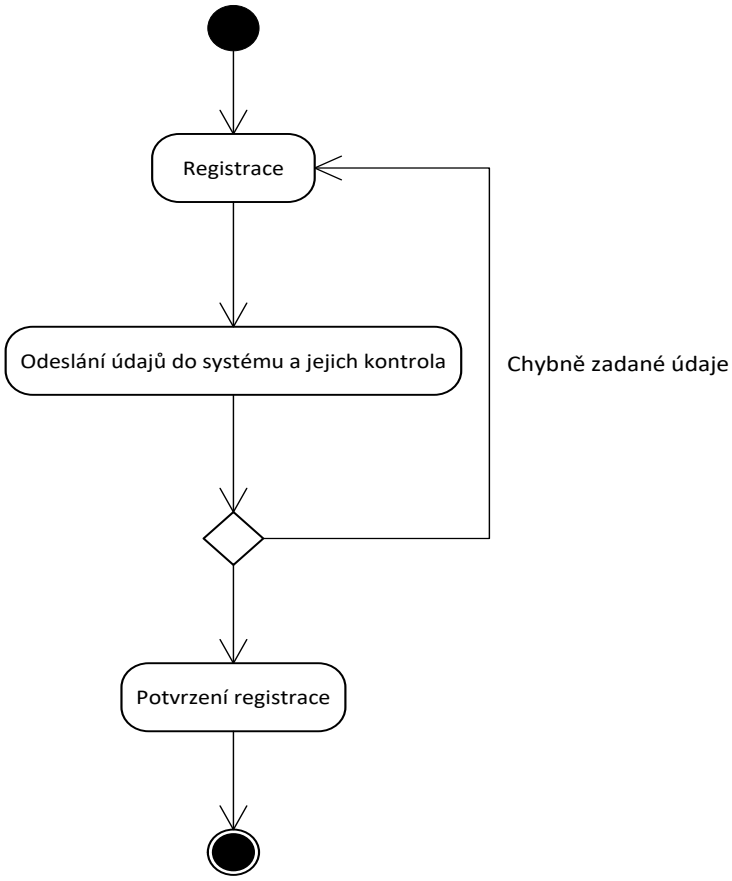


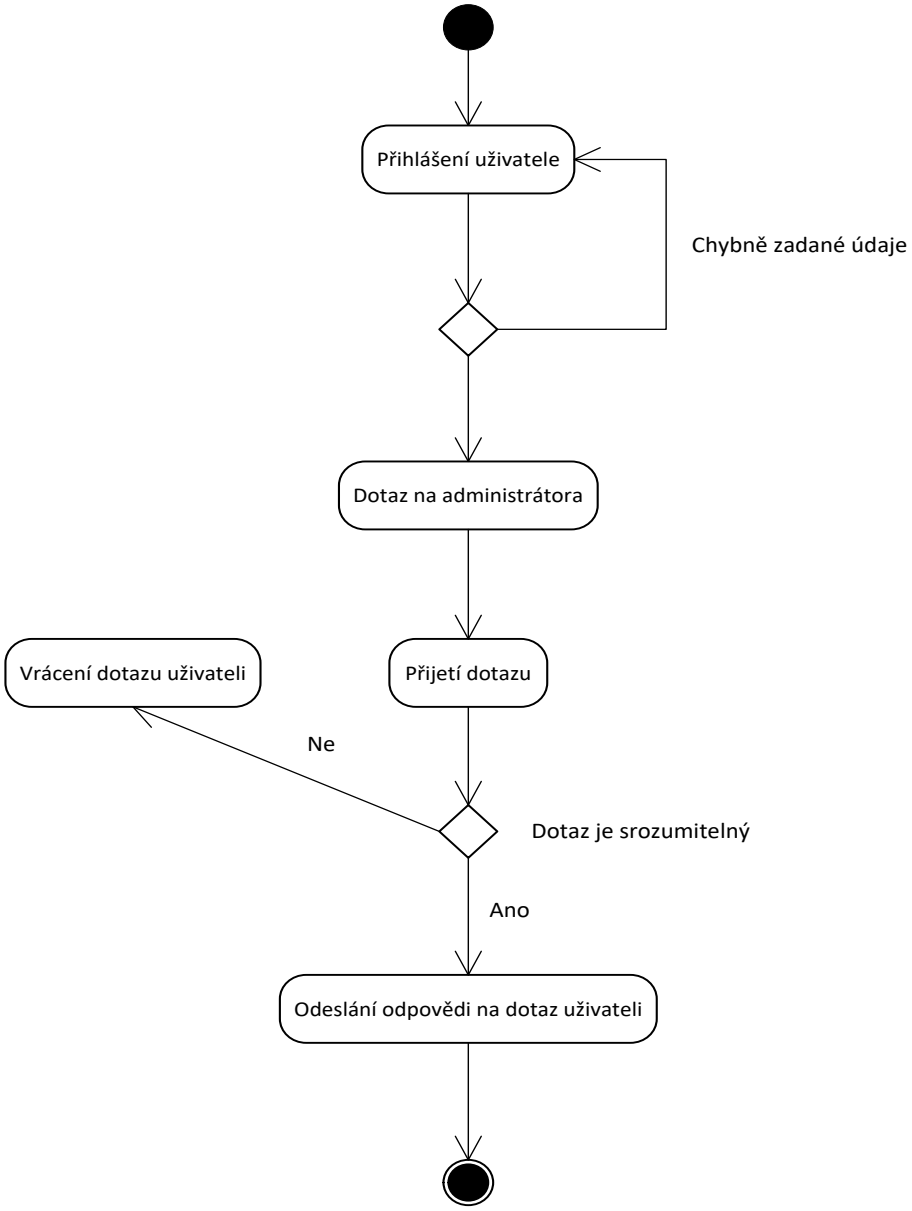
DIAGRAM AKTIVIT

Prostřednictvím diagramu aktivit zobrazuji dva typy činností: registraci nového zákazníka a dotaz registrovaného zákazníka na administrátora.

1) Diagram registrace nového zákazníka


Registrace
Odeslání údajů do systému a jejich kontrola
Chybně zadané údaje
Potvrzení registrace

2) Diagram dotazu registrovaného zákazníka na administrátora



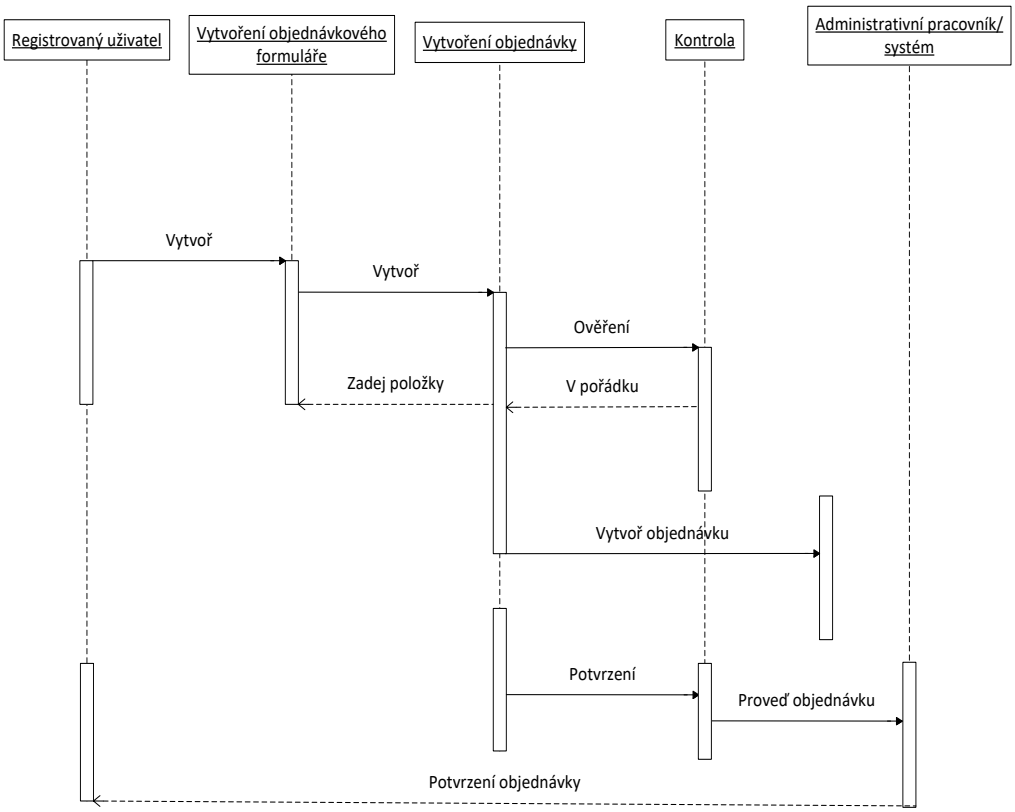
SCÉNÁŘ PŘÍPADU UŽITÍ

Scénář případu užití popisuje proces událostí, které popisují průběh rezervace přes rezervační systém. Komunikace probíhá na základě událostí vykonaných registrovaným uživatelem, které následně rezervační systém vyhodnocuje a na jejich základě provádí nezbytné operace pro úspěšné dokončení rezervace.

1	Uživatel	Vstoupí do portálu.
2	Systém	Zobrazí dialogové okno pro přihlášení.
3	Uživatel	Zadá své přihlašovací údaje nebo se nově zaregistruje.
4	Systém	Provede kontrolu přihlašovacích údajů a potvrdí přihlášení uživatele nebo potvrdí registraci a nově registrovaného uživatele přihlásí.
5	Uživatel	Vybere kulturní akci, termín a místa v sále.
6	Systém	Provede předběžnou rezervaci pro daná místa v sále.
7	Systém	Vygeneruje formulář pro způsob platby.
8	Uživatel	Vybere způsob platby z možností osobně nebo přes internet banking.
9	Systém	Vygeneruje potvrzení objednávky a údaje uloží.
10	Systém	V případě platby přes internet banking systém generuje formulář pro zaplacení.
11	Uživatel	Vybere banku, přes kterou platí, zadá potřebné informace a objednávku zaplatí.
12	Systém	Provede transakci.
13	Systém	Potvrdí transakci, uloží údaje a zašle uživateli fakturu.

SEKVENČNÍ DIAGRAM

Diagram popisující sekvenci činností při objednávce registrovaného zákazníka. Jedná se o logický koncept toho, jak mají jít jednotlivé činnosti za sebou, aby nedošlo k selhání procesu.



ZÁVĚR

V rámci předmětu Objektové modelování byly poskytnuty podmínky k seznámení se s metodami vizuálního modelování pomocí softwarových nástrojů. Tvorba UML diagramů pomocí specializovaného software je zajímavá a bylo zajímavé nahlédnout do této problematiky. Seminární práce pomohla nahlédnout na procesy, které běžně pomocí internetu vykonáváme, trochu z jiného pohledu, více si uvědomit, co se za danými procesy skrývá a dle jakých kroků se postupuje.

10.5 Modelování IS knihovny

Úvod

Práce je zaměřena na problematiku informačního systému knihovny. Jedná se o jednoduchý informační systém, který by měl nabízet základní funkce, jak pro registrované a neregistrované uživatele a administrátora.

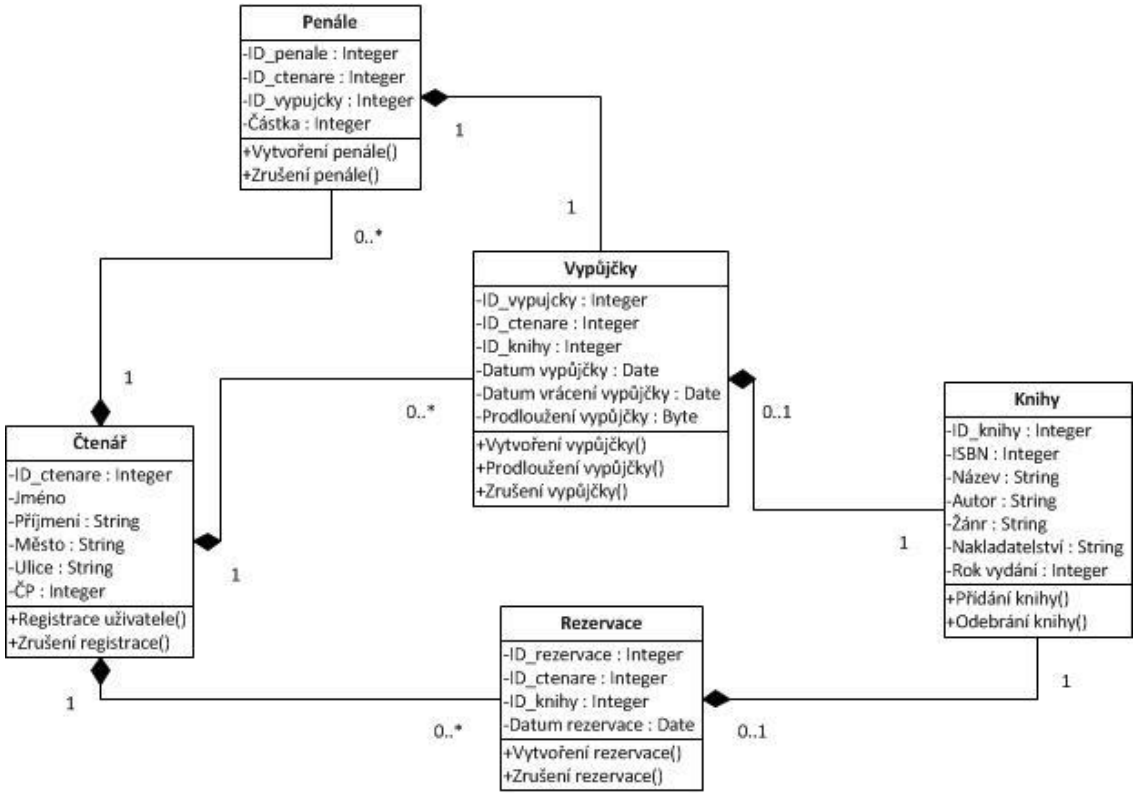
Problém nastává při půjčování knih, kdy se některé tituly nevracejí, a po delší době je složité dohledat, kdo si daný titul vypůjčil.

Funkce IS:

Prohlížení titulů, Registrace uživatele, Přihlašování uživatelů, Rezervace titulů, Správa registrovaných uživatelů, Správa titulů, Správa vypůjčených titulů

Diagram tříd

Diagram znázorňuje datový model IS jeho třídy a vztahy mezi nimi. U jednotlivých tříd jsou poté popsány jejich atributy a operace.



USE CASE DIAGRAM

Diagram případu užití zobrazuje procesy spojené se správou knih, rezervací, výpůjček a čtenářů. V diagramu jsou zobrazeny tři základní role host, čtenář a administrátor.

Host

Jedná se o běžného uživatele, který není přihlášen. Tudíž má omezené možnosti, co se týče využívání systému. Jedinými funkcemi, které jsou mu k dispozici je vyhledávání knih v databázi a možnost se zaregistrovat do systému.

Čtenář

Tato role náleží již přihlášenému uživateli. Tento již může, kromě vyhledávání knih, také provádět rezervace titulů, rušit rezervace, spravovat své výpůjčky (vrácení knih, placení za upomínky) a spravovat informace na svém čtenářském profilu.

Administrátor

Administrativní pracovník, může v systému využívat služeb přidávat a vyřazovat tituly či čtenáře, potvrzovat rezervace, výpůjčky a registrace uživatelů.

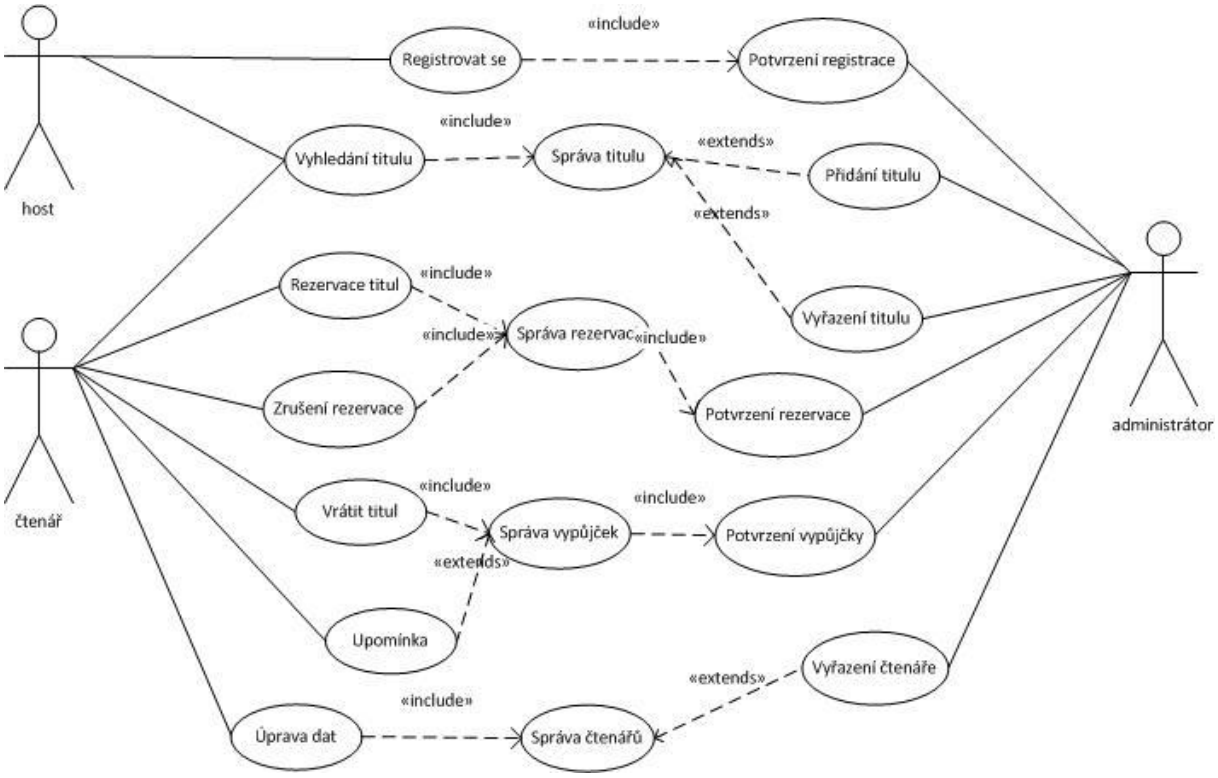
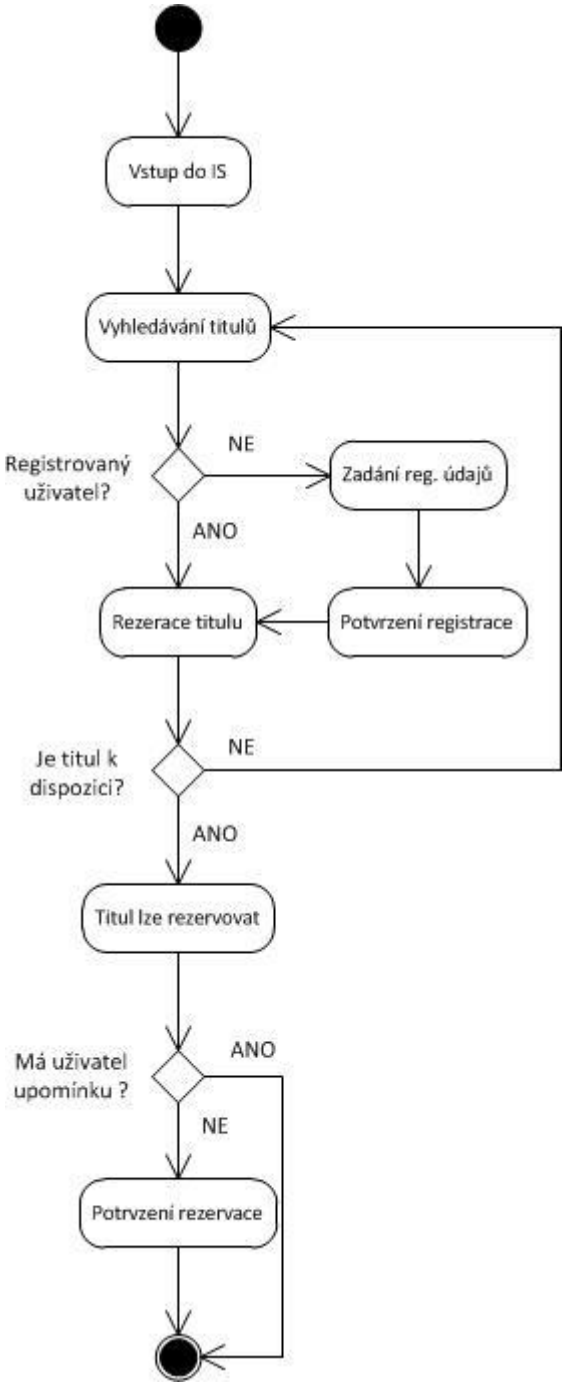


DIAGRAM AKTIVITY

Diagram zobrazuje užití informačního systému, kdy do něj vstoupí nepřihlášený uživatel a chce si vypůjčit knihu. Systém ho následně vyzve k přihlášení, příp. registraci, zjistí, zda je titul k dispozici a zda již čtenář nemá nějakou upomínku.



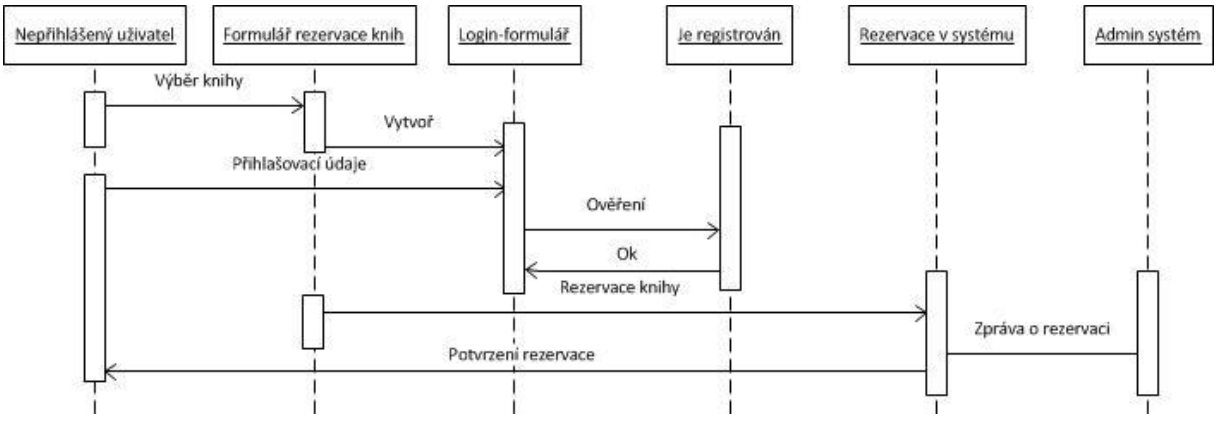
SCÉNÁŘ PŘÍPADU UŽITÍ

Tato část již slovně popisuje události, při nichž dochází od vstupu nepřihlášeného uživatele po uložení veškerých dat do databáze systému.

Krok	Role	Scénář
1	Uživatel	Vstoupí do IS
2	Systém	Zobrazí možnost vyhledávání, přihlášení, nebo registrace
3	Uživatel	Zvolí registraci
4	Systém	Zobrazí formulář pro registraci
5	Uživatel	Zadá své údaje a provede registraci
6	Systém	Potvrdí registrací a zobrazí potvrzení
7	Systém	Zobrazí formulář pro přihlášení
8	Uživatel	Zadá své údaje a přihlásí se
9	Systém	Zobrazí uživateli možnost vyhledávání
10	Uživatel	Vybere titul k rezervaci
11	Systém	Provede kontrolu, zda je titul k dispozici a zda nemá uživatel upomínku
12	Systém	Nahlásí uživateli, že titul je možné si rezervovat
13	Uživatel	Potvrdí rezervaci
14	Systém	Uloží údaje o rezervaci do databáze a pošle zprávu administrátorovi

SEKVENČNÍ DIAGRAM

Na následujícím diagramu je znázorněno, jak bude probíhat komunikace mezi aktérem a komponenty systému v čase.



Závěr

Práce byla zaměřena na problematiku modelování informačního systému. Pro vypracování byl zvolen zjednodušený informační model knihovny, který řeší funkce a vztahy mezi registrovanými uživateli, knihami a jejich rezervacemi a výpůjčkami. Důležitou roli při návrhu informačních systému je přijat na veškeré komponenty a části, které budou následně uživatelé a administrátoři využívat a přijít na způsob, které z těchto složek a jakým způsobem provázat, aby spolu bezchybně komunikovaly a také nezatěžovaly zbytečně systém.

Při pracování seminární práce bylo využito produkt Microsoft Visio 2010. Produkt jsem stáhl prostřednictvím služby MSDN.AA.

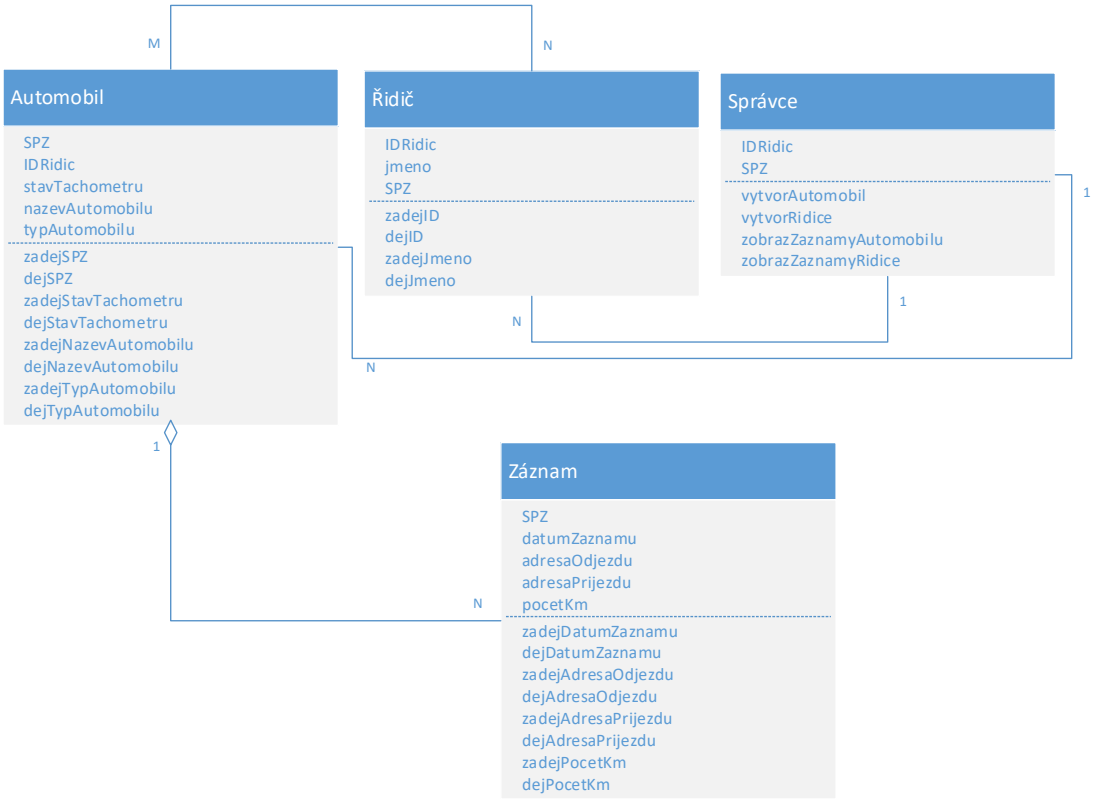
10.6 Kniha jízd

Úvod

Tato aplikace má za úkol evidovat seznam vozidel a informace o ujetých kilometrech, lokalit odjezdu a příjezdu vozidel a časové údaje o jízdě. Umožňuje zadávat a prohlížet tyto údaje a zobrazovat patřičné statistiky vozidel a vytvářet tiskové sestavy.

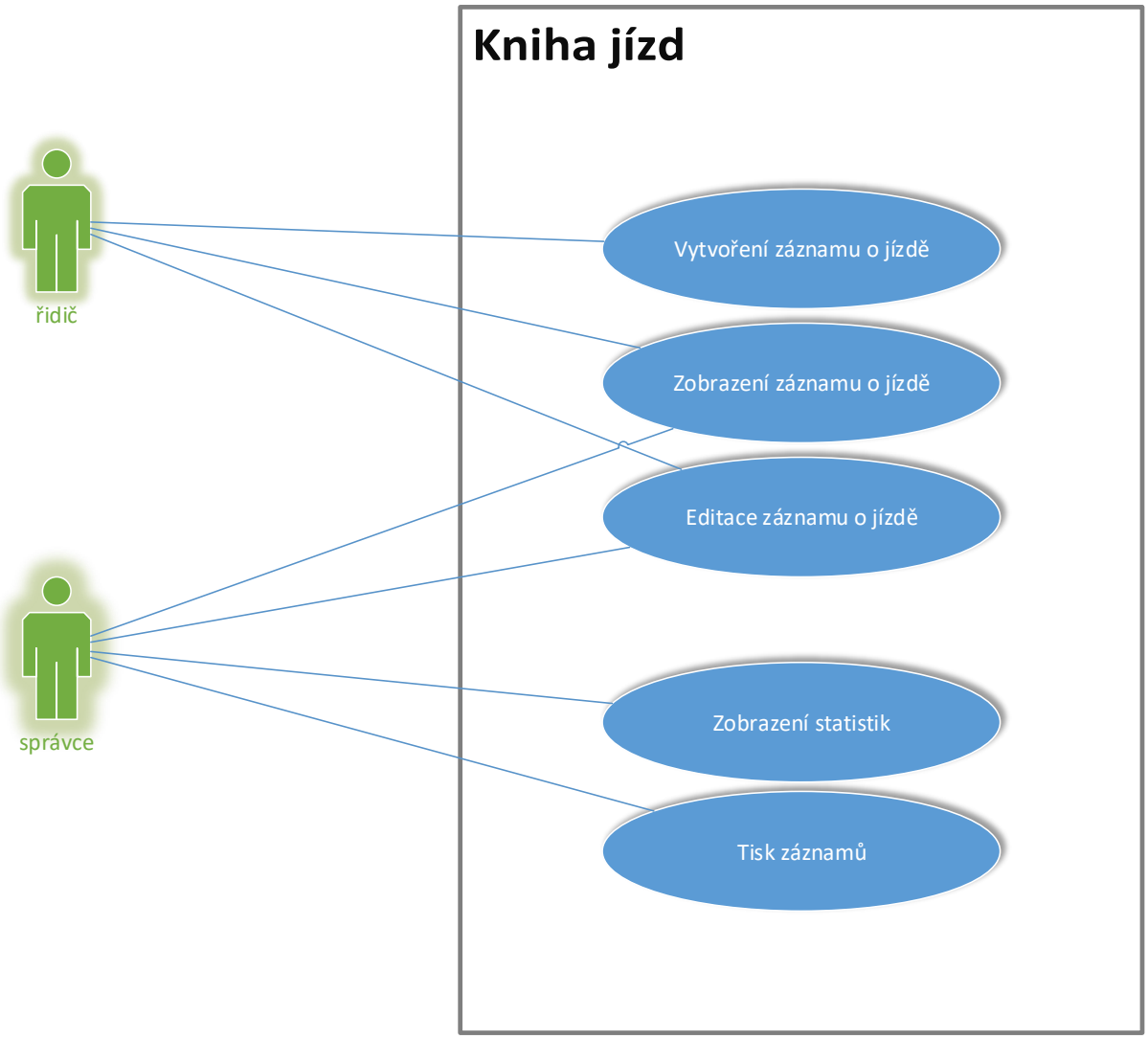
Diagram tříd

V diagramu jsou znázorněny třídy, které jsem vytvořil na základě interakce uživatelů s aplikací. Třídy mají mezi s sebou vazby v podobě agregace a asociace.



USE CASE DIAGRAM

Zobrazuje role, které se při používání aplikace vyskytují (řidič, správce) a jejich možnosti interakce s tímto programem.



SCÉNÁŘE PŘÍPADU UŽITÍ

Ukazují příklad možného využití aplikace v roli správce, který chce zobrazit statistiky o jízdách vozidel a řidiče, který vkládá nový záznam o jízdě.

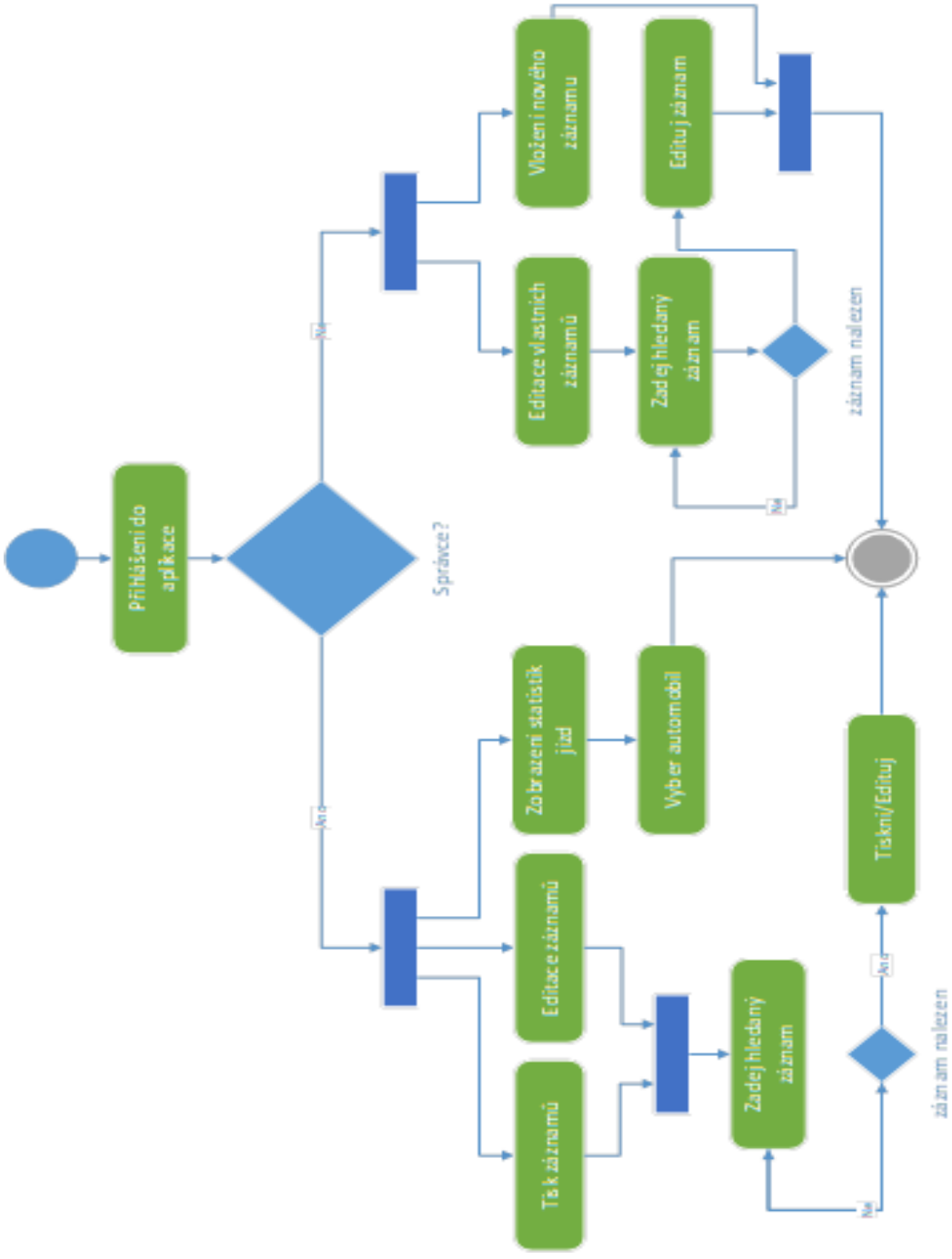
Přihlášení správce

Krok	Role	Akce
1	Uživatel	Otevře aplikaci
2	Uživatel	Přihlásí se do systému jako správce
3	Systém	Ověří zadané údaje a autorizuje přihlášení
4	Uživatel	Z nabídky zvolí sledování statistik jízd
5	Systém	Zobrazí menu statistik
6	Uživatel	Zadá údaje pro zobrazení požadovaných dat
7	Systém	Na základě zadaných údajů zobrazí příslušná data
8	Uživatel	Prohlédne si zobrazené statistiky
9	Uživatel	Odhlásí se a uzavře aplikaci

Přihlášení řidiče

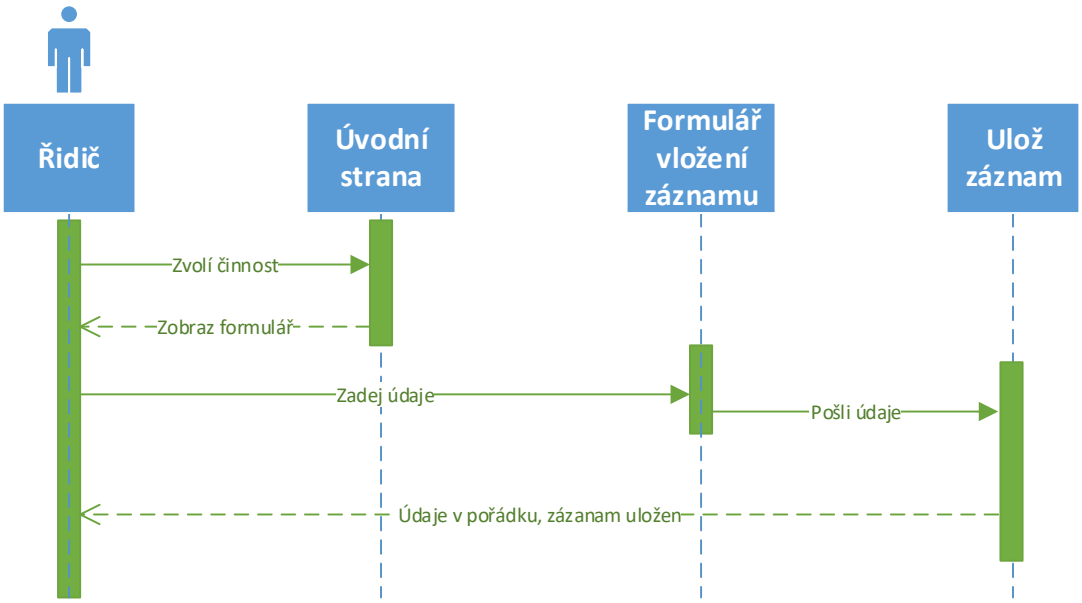
Krok	Role	Akce
1	Uživatel	Otevře aplikaci
2	Uživatel	Přihlásí se do systému jako řidič
3	Systém	Ověří zadané údaje a autorizuje přihlášení
4	Uživatel	Z nabídky zvolí vložení nového záznamu
5	Systém	Zobrazí formulář pro zadání záznamu
6	Uživatel	Zadá údaje požadované systémem
7	Systém	Uloží údaje do databáze
8	Uživatel	Odhlásí se a uzavře aplikaci

DIAGRAM AKTIVIT



SEKVENČNÍ DIAGRAM

Zobrazuje časové posloupnosti aktivity uložení záznamu o jízdě, kterou provádí uživatel a systém na jeho kroky reaguje.



ZÁVĚR

V této práci je pomocí UML diagramů zachycena struktura aplikace „Kniha jízd“. Je v ní popsáno několik možných scénářů, které charakterizují tento systém. Všechny diagramy jsou zpracovány pomocí programu Visio.

SHRNUTÍ STUDIJNÍ OPORY

Studijní opora „Objektové metody modelování v příkladech“ je určena studentům studijního programu manažerská informatika na Obchodně podnikatelské fakultě v Karviné, Slezské univerzity v Opavě.

Vznikla na základě zkušeností autora s vedením seminářů a přednášek v předmětu Objektové metody modelování.

Studijní opora ve spojení s předchozí studijní oporou „Objektové metody modelování s využitím jazyka UML“ poskytuje ucelený náhled do problematiky analytického procesu návrhu informačního systému, tvorby software nebo dokumentace software. Postihuje fázi návrh systému, analýzy systému předcházející zahájení programátorských prací.

Velký důraz je kladen při výkladu látky na příklady z praxe, které provázejí celý text a je jim věnována samostatná závěrečná kapitola.

Problematika objektových metod modelování, návrh informačních systémů a tvorba software je velmi rozsáhlá. Pokrývá celý vývojový proces od návrhu informačního systému, přes jeho programování až po jeho nasazení, údržbu a inovování. Tato studijní opora je věnována zejména části analýzy, která předchází programování.

Popsané poznatky a přístupy při návrhu informačních systémů jsou nezbytným vědomostním vybavení studenta studijního programu „Manažerská informatika“ Slezské univerzity v Opavě, Obchodně podnikatelské fakultě v Karviné.

LITERATURA

[1] [Arl2008] ARLOW, J. NEUSTADT, I. UML 2 a unifikovaný proces vývoje aplikací: Objektově orientovaná analýza a návrh prakticky. Přel. Bogdan Kiszka. Dotisk 1.vydání, Brno: Computer Press, 2008. 567 s. ISBN 978-80-251- 1503-9.

[2] [Ald2005] ALDORF, F. Metodika RUP - diplomová práce, 2005, VŠE Praha

[3] [Boo1998] BOOCH, G. JACOBSON, I. RUMBAUGH, J. The Unified Modeling Language User Guide. 1.vyd. Addison Wesley, 1998. ISBN 0- 201-57168-4.

[4] [Buch2007] BUCHALCEVOVÁ, A. PAVLÍČKOVÁ, J. PAVLÍČEK, L. Základy softwarového inženýrství - materiály ke cvičení. 1.vyd. Praha: Vysoká škola ekonomická, 2007. 222 s. ISBN 987-80-245-1270-9.

[5] [Fow2003] FOWLER, M. UML Distilled: A Brief Guide to the Standard Object Modeling Language. 3. edition. Addison Wesley, 2003. 208 s. ISBN 0- 321-19368-7.

[6] [Kan2004] KANISOVÁ, H. Miller, M. UML srozumitelně, 1.vyd, Brno: Computer Press, 2004. 158 s. ISBN 80-251-0231-9.

[7] [Pen2003] PENDER, Tom. UML Bible. 1.vyd. Indianapolis: Wiley Publishing, Inc., 2003. ISBN 0-7645-2604-9.

[8] [Rum2004] RUMBAUGH, J. JACOBSON, I. BOOCH, G. The Unified Modeling Language Reference Manual. 2.vyd. Addison-Wesley, 2004. ISBN 032124562853

[9] [Schm2001] SCHMULLER, J. Myslíme v jazyku UML : knihovna programátora. přel. Jiří Hynek. 1.vyd. Praha: Grada, 2001. 360 s. ISBN 80-247-0029-8.

[10] [Str2007] STŘÍŽOVÁ, V. HORNÝ, S. SVATÁ, V., VÁCLAVÍKOVÁ, M. Systémové pojetí (hospodářské) organizace. 1.vyd. Praha: Oeconomica, 2007. 239 s. ISBN 978-80-245-1265-5.

[11] [Lef2010] LEFFINGWELL, D. *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise***.** Addison-Wesley Professional, 2010, 560 s., ISBN 0-321-63584-1

[12] [Fow2009] FOWLER, M**.** Destilované UML**.** Grada Publishing a.s. 2009. 176 s. ISBN 978-80-247-2062-3

[13] [Fra2014] FRANĚK Z. Objektové metody modelování, učební text, SU OPF 2014, 115 s., elektronická verze „online“, ISBN 978-80-7510-081-8

Internetové zdroje:

[1] http://mpavus.wz.cz

[2] http://uml.czweb.org/index.html

[3] http://www.rational.com

[4] http://www.ibm.com

[5] http://www.sparx.com

[6] http://ecom.ef.jcu.cz/web2/download/podklady/zakladni-pojmy-ea.pdf

[7] http://dspace.upce.cz/bitstream/handle/10195/64679/E1.pdf?sequence=1&isAllowed=y

[8] https://www.root.cz/clanky/nastroje-pro-tvorbu-uml-diagramu/

PŘÍLOHA Č. 1: SEZNAM OBRÁZKŮ

PŘÍLOHA Č. 2: SEZNAM TABULEK

PŘEHLED DOSTUPNÝCH IKON


Čas potřebný ke studiu


Cíle kapitoly


Klíčová slova


Nezapomeňte na odpočinek


Průvodce studiem


Průvodce textem


Rychlý náhled


Shrnutí


Tutoriály

Definice


K zapamatování


Případová studie


Řešená úloha

Věta


Kontrolní otázka


Korespondenční úkol


Odpovědi

Otázky


Samostatný úkol


Další zdroje


Pro zájemce


Úkol k zamyšlení

Název: Objektové metody modelování v příkladech

Autor: **RNDr. Zdeněk Franěk, Ph.D.**

Vydavatel: Slezská univerzita v Opavě

Obchodně podnikatelská fakulta v Karviné

Určeno: studentům SU OPF Karviná

Počet stran: 130