ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

Katedra ekonomiky

Bakalářská práce

BUSINESS ANALÝZA A NÁVRH ŘEŠENÍ POMOCÍ UML

Jan Vašata

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Jan Vašata

Ekonomika a management

Název práce

Business analýza a návrh řešení pomocí UML

Název anglicky

Business analysis and solution design using UML

Cíle práce

Cílem bakalářské práce je analyzovat a navrhnout model informačního systému v jazyce UML. Modelovaný informační systém bude poskytovat podporu pro administraci a správu spotřebitelských úvěrů. Dílčí cíle práce jsou:
- vytvořit literární rešerši v oblasti business analýzy a návrhu informačních systémů, analýzy uživatelských požadavků a objektového modelování informačních systémů,
- vytvořit analýzu uživatelských požadavků a datový slovník,
- navrhnout objektový model, stavový model a model interakcí.

Metodika

Metodika řešené problematiky bakalářské práce je založena na studiu odborných informačních zdrojů. Získané znalosti budou využity pro návrh daného informačního systému. Vlastní řešení je realizováno použitím metod softwarového inženýrství a jazyka UML (Unified Modeling Language). Na základě rozboru teoretických poznatků a výsledků vlastního řešení budou formulovány závěry bakalářské práce.

Doporučený rozsah práce

30-40

Klíčová slova

UML, business analýza, návrh, vývoj, informační systém, případ užití

Doporučené zdroje informací

BOOCH, Grady, James RUMBAUGH a Ivar JACOBSON. The unified modeling language user guide. 2nd ed. Upper Saddle River: Addison-Wesley, 2005. ISBN 9780321267979.

BRUCKNER, Tomáš. Tvorba informačních systémů: principy, metodiky, architektury. Praha: Grada, 2012. Management v informační společnosti. ISBN isbn978-80-247-4153-6.

BUCHALCEVOVÁ, Alena a Iva STANOVSKÁ. Příklady modelů analýzy a návrhu aplikace v UML. Praha: Oeconomica, 2013. ISBN 978-80-245-1922-7.

COAD, P. Object oriented analysis. New Jersey: Prentice-Hall, 1991. ISBN 0136299814.

PAGE-JONES, Meilir. Základy objektově orientovaného návrhu v UML. Praha: Grada, 2001. Moderní programování. ISBN 80-247-0210-x.

VRANA, Ivan. Projektování informačních systémů s UML. V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta, 2008. ISBN 978-80-213-1817-5.

Předběžný termín obhajoby

2021/22 LS – PEF

Vedoucí práce

doc. Ing. Jan Tyrychtr, Ph.D.

Garantující pracoviště

Katedra informačního inženýrství

Elektronicky schváleno dne 1. 11. 2021

Ing. Martin Pelikán, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 9. 11. 2021

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 07. 03. 2022

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Business analýza a návrh řešení pomocí UML" jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu literatury na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze, dne

_____

Poděkování

Rád bych touto cestou poděkoval vedoucímu mé bakalářské práce doc. Ing. Janu Tyrychtrovi, Ph.D. za jeho odborné vedení a cenné rady, které mi při tvorbě této práce velice pomohly.

Business analýza a návrh řešení pomocí UML

Souhrn

Tato bakalářská práce se zabývá návrhem systému pro administraci a správu spotřebitelských úvěru ve finanční instituci. Tento systém lze využít k celému procesu návrhu, schvalování a vedení dokumentace při žádání o úvěr. Návrh je vytvořen pomocí modelovacího jazyka UML (Unified Modeling Language). Konkrétně je vytvořen diagram tříd, stavů a případů užití. K praktické části patří soupis požadavků na systém, vycházející s uživatelských požadavků, popis aktérů figurující v systému a datový slovník.

V teoretické části práce je uveden potřebný obecný základ grafického jazyka UML a jeho využití v businessové analýze. Je zde popsán i význam uživatelských požadavků a práce s nimi.

Klíčová slova

UML, business analýza, návrh, vývoj, informační systém, případ užití.

Business analysis and solution design using UML

Summary

This bachelor thesis deals with the design of a system for the administration and management of consumer loans in a financial institution. This system can be used for the entire process of designing, approving and maintaining documentation when applying for a loan. The design is created using the Unified Modeling Language (UML). Specifically, a diagram of classes, states, and use cases is created. The practical part includes a list of system requirements, based on user requirements, a description of the actors involved in the system and a data dictionary.

The theoretical part of the thesis presents the necessary general basis of the graphical language UML and its use in business analysis. The importance of user requirements and work with them is also described here.

Keywords

UML, business analysis, design, development, information system, use case.

Obsah

Seznam obrázků

Seznam tabulek

Seznam použitých zkratek

CASE	Computer Aided Software Engineering
EA	Enterprise Architect
IS	Informační systém
IT	Informační technologie
OMG	Object Management Group
UC	Use case
UML	Unified Modeling Language

1 Úvod

Autor ve své bakalářské práci poskytuje souhrnný pohled na business analýzu a základní principy návrhu informačních systémů pomocí zobrazení v jazyce UML (Unified Modeling Language). K tomuto tématu autora přivedla pracovní praxe v bankovním prostředí, kde měl autor příležitost podílet se na vývoji systémů potřebných ke sjednocení a zlepšení fungování administrace při zpracování bankovní dokumentace. Je možné říci, že banky obvykle disponují určitými více či méně robustními systémy na správu procesní a klientské dokumentace, nicméně tyto systémy zpravidla nejsou jednotné v rámci všech bankovních oddělení. Pro evidenci počtů smluv, stavů vyhotovení dokumentace a nestandardních řešení se v jednotlivých odděleních využívají excelovské tabulky, do kterých mohou jednotliví uživatelé zanést chyby. Z těchto individuálně řešených systémů je zpravidla problém získat relevantní data pro reporting, dochází k duplikaci a nepřehlednosti dat atd. Komplexní přístup z hlediska business analýzy poskytuje prostředek, který může (nejen) bankovním institucím poskytnout snadno přehledný rozbor dané problematiky a pomoci při návrhu optimálního řešení.

Dalším důvodem, proč si autor práce vybral téma bakalářské práce, byla možnost využití UML diagramů v praxi, a to se záměrem získat více teoretických i praktických zkušeností v rámci analýzy a návrhu systémů.

Beze sporu je možno říci, že efektivní fungování jakéhokoliv podniku nebo instituce nelze v současné době provozovat bez využití informačních technologií, především informačních systémů. Takové systémy musí být navrženy tak, aby vyhovovaly činnostem, k jakým jsou určeny. Kromě technického řešení, které spadá výhradně do kompetencí IT odborníků, je při navrhování těchto systémů nutné vzít v úvahu konkrétní agendu a pracovní postupy používané v dané instituci. Smyslem navrženého systému je vytvořit ucelenou aplikaci, která bude přehledná a předvídatelná pro všechny uživatele a umožní plnohodnotnou práci s dokumentací a následné vypovídající vyhodnocení dat. Autor má na základě své pracovní praxe v bankovním prostředí za to, že právě takový systém v prostředí českých bankovních institucí chybí nebo není komplexní.

Hlavními zdroji informací pro návrh nového systému jsou konzultace s aktéry procesů, kteří do tohoto systému nebo procesu zasahují nebo se jich týká, eventuálně analýza

současného systému nebo procesu a sestavení požadavků na nový systém nebo transformaci starého. Autor k sestavení požadavků na systém navržený v této práci využil vlastní zkušenosti a znalosti ze zmíněno prostředí.

2 Cíl bakalářské práce

Cílem bakalářské práce je analyzovat a navrhnout model informačního systému v grafickém jazyce UML. Modelovaný informační systém bude poskytovat podporu pro administraci a správu spotřebitelských úvěrů. Tento systém zefektivní a zjednoduší práci s dokumentací a ve schvalovacím procesu.

Dílčí cíle práce jsou:

- vytvořit literární rešerši v oblasti business analýzy a návrhu informačních systémů, analýzy uživatelských požadavků a objektového modelování informačních systémů;
- vytvořit analýzu uživatelských požadavků a datový slovník;
- navrhnout objektový model, stavový model a model interakcí.

3 Metodika

Metodika řešené problematiky bakalářské práce je založena na studiu odborných informačních zdrojů. Získané znalosti budou využity pro návrh daného informačního systému. Vlastní řešení je realizováno použitím metod softwarového inženýrství a jazyka UML. Na základě rozboru teoretických poznatků a výsledků vlastního řešení budou formulovány závěry bakalářské práce.

Teoretická východiska jsou v této práci rozdělena ve 4. kapitole Literární rešerše. V úvodu 4. kapitoly se autor věnuje zejména popisu business analýzy a návrhu informačního systému a vymezení základních pojmů. Popisuje analytiku jako metodu k dekompozici celku na elementární části a roli business analytika v ní. Dále je v kapitole popsán termín systém, jeho specifikace a popis modely za využití grafického jazyka UML (Unified Modeling Language) k tvorbě takových modelů. Následující kapitola je zaměřena na historii jazyka UML a analýzu uživatelských požadavků. V poslední kapitole literární rešerše budou představeny nejpoužívanější modely (diagramy), jejich tvorba, specifikace a vlastnosti.

Praktická část práce je zahrnuta v 6. kapitole. V jejím úvodu je rozepsána analýza požadavků na software, určující budoucí funkce a fungování systému. Poté je z těchto požadavků navržen funkční model informačního systému, umožňující efektivní správu těchto úvěrů a zjednodušující práci uživatelům. Model bude obsahovat role, tedy jeho aktéry, jednotlivé objekty a jejich vlastnosti (atributy). Tento popis systému bude rozšířen o model stavů, událostí a změn objektů v čase. V závěru bude vytvořen a popsán model případů užití, obsahující souhrn činností vykonávaných při zpracování a správě úvěrových produktů.

V praktické části bude navržen systém za použití modelovacího jazyka UML nejnovější notace verze 2.5.1. Diagramy budou použity dle specifikace jazyka v dokumentu OMG Unified Modeling Language (OMG UML) Cook a kol. (2015).

Autor bude ve své práci používat nástroj pro komplexní modelování Enterprise Architect (EA) od společnosti Sparx Systems. Tento CASE (Computer Aided Software Engineering) nástroj je vhodný pro systémovou analýzu a návrh modelů s využitím diagramů UML. Existují i další nástroje na modelování procesů a systémů jako například Microsoft Visio, nicméně ten podle autora neumožňuje tak komplexní práci s modely a propojení všech elementů, které umožňuje zmíněné EA.

4 Literární rešerše

4.1 Business analýza a návrh IS

Cílem této kapitoly je především vymezení jednotlivých pojmů a přiblížení charakteristik analýzy prostředí firmy a následného modelování informačních systémů. Z pohledu modelování nových procesů a IS (Informačních systémů) autor pokládá znalost a pochopení prostředí firmy za klíčové. K pochopení anatomie firmy je nutná analýza jejího prostředí, aktérů a jejich požadavků. Základní pojem „analýza“ značí při modelování IS postup nebo kroky, jejichž výstupem je logický model vytvářeného systému (Potančok, Pour, Chramostová, 2020, s. 191).

Systém je kognitivní model dané části reálného světa, vhodně rozložen do kolekce subsystémů nebo sady prvků organizovaných k dosažení konkrétního účelu a popsaných sadou modelů z různých hledisek. Zmíněný subsystém je určité seskupení prvků, z nichž některé tvoří specifikaci a popis chování nabízeného ostatními obsaženými prvky. Subsystém je jednoduše součástí systému a používá se k rozložení složitého systému na téměř nezávislé části. Systém na jedné úrovni abstrakce může být subsystémem systému na vyšší úrovni abstrakce (Booch, Jacobson, Rumbaugh, 1998, s. 421).

Pojmem model rozumíme zjednodušení reality, vytvořený za účelem lepšího pochopení vytvářeného systému, jinými slovy je to významově uzavřená abstrakce systému. Subsystém představuje rozdělení prvků většího systému na nezávislé části, model je rozdělení abstrakcí, které tento systém vizualizují, specifikují, konstruují a dokumentují (Booch, Jacobson, Rumbaugh, 1998, s. 422).

Business analýza představuje osvědčené způsoby jednání realizace business (nikoliv „jen“ IT – Informační technologie) změn v organizaci na základě definovaných potřeb a doporučených řešení, která dodají hodnotu zainteresovaným stranám. Tuto činnost vykonávají ve firmách business analytici, o kterých pojednává další kapitola práce. Výsledná řešení zahrnují vývoj nových nebo vylepšených informačních systémů, nicméně rozsah řešení bývá zpravidla širší a zahrnuje také změny v oblastech organizačních struktur, firemních procesů, redefinici rolí, definice business pravidel, žádoucí změny v dovednostech lidí apod.

Témata spojené s business analytikou jsou velmi široká, vzhledem k tomu se autor rozhodl zaměřit zejména na návrh řešení a specifikace systémů a procesů pomocí grafického jazyka UML, který bude blíže popsán v jedné z dalších kapitol.

Autor by rád zdůraznil důležitost vymezení prostředí firmy, kde se business analýza odehrává. K tomu je zásadní znalost pochopení anatomie firmy v důrazu na komplexnost, vidění firmy jako celek ve všech souvislostech. Dále pochopení podstaty a význam všech komponent řízení, včetně dat, datových zdrojů, IT (Informační technologie) a jejich vzájemný vztah (Potančok, Pour, Chramostová, 2020, s. 18).

Anatomie firmy je tedy základ analytického pohledu na firemní obsah. Můžeme ho vymezit jako souhrn všech komponent řízení a jejich charakteristik, specifikace obsahu komponent a určení vazeb mezi nimi (Potančok, Pour, Chramostová, 2020, s. 20).

Základními kameny anatomie firmy jsou právě komponenty řízení a jejich vzájemné vazby. Do nich můžeme zahrnout popis klíčových aktivit a detailů procesů, nazývajících se souhrnně úlohy. Dále scénáře, kterými definujeme situace a doporučená řešení těchto situací nebo vzniklých problémů. V anatomii firmy jsou také zásadní role jednotlivých účastníků. Rolemi určujeme typy pracovních pozic a jejich funkční náplň. Tuto náplň měříme pomocí metrik, tedy měřitelným ukazatelem, zvyšujícím objektivitu a přesnost při hodnocení řízení nebo procesu. Měření lze provádět pomocí kvalitních dat a dokumentace. Tato data představují vstup nebo výstup jednotlivých procesů neboli úloh.

Metrika znamená sledovaná a měřená hodnota určitého ukazatele. Účelem metriky je zpřesnění, objektivizace a konkretizace určitého důležitého jevu pro řízení firmy. U jednotlivých procesů a úloh je důležité určení, co se bude měřit, kde tyto data získávat a jak naměřené hodnoty využívat. Nejdůležitějšími charakteristikami metrik je jejich obsah a účel, zdroje, měrné jednotky a logické sdružování a odvozování metrik do větších a souvisejících celků (Potančok, Pour, Chramostová, 2020, s. 32).

Strukturu úloh, podle publikace (Potančok, Pour, Chramostová, 2020, s. 22-23), můžeme rozdělit do třech úrovní podle granularity procesů, tedy od nejpodrobnější až po zcela rámcové. První úrovní je doména, tedy vlastní kategorie z pohledu anatomie firmy. Například zemědělská firma, úřad veřejné správy nebo IT vývojová firma. Druhou úrovní určujeme jednotlivé oblasti řízení ve firmě. Zde by byly příkladem oddělení bezpečnosti, controllingu nebo informační systémy veřejné správy. Třetí úrovní jsou dílčí úlohy a

procesy, pro oblast bezpečnosti je to úloha zabezpečení emailové schránky, pro IS veřejné správy je to úloha vytvoření zadávacího listu na vývoj určité aplikace nebo funkce.

Aplikační software je program určený koncovému uživateli informačního systému jako nejvyšší úroveň softwarových prostředků. Celkový koncept informačního systému, jinými slovy vizi vyvíjeného IS, nazýváme architekturou informačního systému. Zachycuje jednotlivé komponenty a jejich vazby (Potančok, Pour, Chramostová, 2020, s. 191).

Analýza v objektově orientovaném programování je vlastně zkoumání procesu a tvorba objektově orientovaného modelu, který zachycuje požadované chování programového díla. Záměrem analýzy z pohledu objektově orientovaného analytika je tvorba analytického modelu. Tento model znázorňuje, co má systém dělat, nicméně se nezabývá detaily, jak by toto mělo být provedeno. Jakým způsobem se má systém zachovat, se zabývá aktivita následující po analýze, kterou je návrh (Arlow, Neustadt, 2008, s. 138).

Hlavními přínosy objektově orientovaného přístupu je snížení složitosti při tvorbě moderního softwaru. Snížení složitosti se docílí použitím vyšší abstrakce při návrhu takového systému a důraz na modularitu, znuvupoužitelnost a standardizaci (Merunka, Pergl, Pícka, 2005, s. 18).

4.2 Business analytik

Business analytici pomáhají v rámci jednotlivých organizací porozumět potřebám změny a jejím důvodům, navrhnout možná řešení a doporučit optimální řešení, které organizaci přinese největší benefit. Business analytik v širším slova smyslu řeší obsahovou a logickou stránku jednotlivých dílčích úloh v rámci projektů a zajišťuje především spolupráci na business strategii a business plánu. Konzultuje problematiku s uživateli, formuluje prostřednictvím analytické dokumentace uživatelské požadavky, vytváří procesní modely, analyzuje návrhy a optimalizuje firemní procesy. Vytváří objektové a datové modely a definuje IT služby a podporující business procesy. Business analytik také definuje parametry pro fungování softwaru, řídí nasazení nových aplikací dle zadání. Definuje funkcionality a vytváří zadání pro řešení specializovaných aplikací a v neposlední řadě zpracovává projektové a uživatelské dokumentace (Potančok, Pour, Chramostová, 2020, s. 30). Uživatelská dokumentace obsahuje manuály pro koncové uživatele, aplikační podporu nebo systémové administrátory. Projektová dokumentace obsahuje hlavní cíle, definuje

rozsah projektu a problémy na které je potřeba se zaměřit. Dotyčný business analytik k tomuto zkoumání a analýzám, kromě vlastního zkoušení a testování, využívá nejčastěji diskuzi s uživateli nebo metodiky.

Je vidět, že náplň práce business analytika může být velmi široká, nicméně přesná pracovní náplň se může v každé organizaci lišit. V některých organizacích může být tato role vnímána i jako analytik trhu a obchodních příležitostí nebo spíše čistě datový a procesní analytik. Autor v práci pod pojmem „Business analytik“ bude používat roli (osobu), která spojuje svět IT a uživatelů. Převádí uživatelské požadavky do řeči programátorů, pomocí graficky znázorněných diagramů, některým CASE nástrojů.

CASE, znamená počítačem podporované softwarové inženýrství nebo vývoj software s využitím počítačové podpory, jedná se o objektově orientované nástroje. Jde o použití podpůrného softwaru při vývoji nebo údržbě počítačových programů a vychází právě z modelovacího jazyka UML. Tato podpora se může využít v různých stádiích životního cyklu programu, tedy již při sběru požadavků, při tvorbě analýz, samotném návrhu řešení i při programování. Nástroje, které se pro tuto podporu a automatizaci vývojového procesu používají, se nazývají právě CASE nástroje (Kanisová. Müller, 2007, s. 12).

4.3 Modelování pomocí jazyka UML

UML je grafický jazyk pro vizualizaci, specifikaci, konstrukci a dokumentaci v objektově orientovaném přístupu. UML nám poskytuje standardizovaný způsob jak popsat funkcionality systému, procesy a také konkrétní věci, jako jsou třídy napsané v konkrétním programovacím jazyce, databázová schémata a opakovaně použitelné softwarové komponenty (Booch, Jacobson, Rumbaugh, 1998, s. XV). Jazyk UML podrobně nevyjasňuje jak postupovat při analýze nebo návrhu řešení, ale udává rámec a poskytuje nástroje pro takové činnosti. To znamená, že samotný jazyk udává pouze vizuální syntaxi, která může být využita při tvorbě modelů.

Jazyk UML se skládá z mnoha grafických prvků, které lze vzájemně kombinovat do podoby diagramů. Tyto kombinace mají pevně daná pravidla v jednotlivých typech diagramů (Schmuller, 2001, s. 16).

K pochopení struktury UML můžeme jazyk rozdělit do třech základních stavebních bloků (Arlow, Neustadt, 2008, s. 35):

1. Předměty (things) – samotné prvky modelu.
2. Vztahy (relationships) – ty spojují jednotlivé předměty. Vztahem rozumíme relaci a ta určuje, jak spolu dva nebo více předmětů významově souvisí.
3. Diagramy (diagrams) – to jsou náhledy (pohledy) na jednotlivé modely UML. Zobrazují sbírky předmětů a vysvětlují pomocí vizualizace, co tyto předměty v softwarovém systému dělají a jak to budou dělat.

UML je univerzální jazyk pro vizuální modelování systémů a byl navržen pro spojení nejlepších existujících postupů modelovacích technik a softwarového inženýrství. Je navržen takovým způsobem, aby ho mohli interpretovat již zmíněné CASE nástroje. Diagramy vytvořené v jazyku UML jsou srozumitelné pro široké spektrum lidí, ale navíc je lze snadno interpretovat v CASE nástroji (Arlow, Neustadt, 2008, s. 28).

UML je jazyk, který umožňuje modelovat jednoduché i složité aplikace pomocí stejné formální syntaxe. Tak umožňuje sdílet výsledky práce s ostatními návrháři. Je to výborný způsob ke specifikaci, stavbu a dokumentaci softwarových systémů (Kanisová. Müller, 2007, s. 11-12).

K UML se účastníci analýz dostávají nejčastěji prostřednictvím diagramů. Diagram je vyobrazení nějaké skutečnosti obsahující sadu prvků, které mohou být spojeny určitými vztahy dávajícími smysl v této realitě. Účel těchto diagramů je představit si pohled na systém v různé úrovni abstrakce. Model systému v UML si tak můžeme představit jako zmenšený model auta nebo budovy. Tento model popisuje, co má systém dělat, neříká však, jak to má udělat nebo jak se má systém implementovat.

Pro znázornění vize a představy je tato metoda klíčová, před vynálezem UML byl vývoj systémů založen spíše na metodě „pokus-omyl“. Analytik nemusel vždy správně pochopit problematiku a mohl tudíž vytvořit návrh řešení, který byl pro klienta nesrozumitelný. Výsledky analýzy nemusely být jasné a srozumitelné ani programátorům a mohl tak být vytvořen program, který se těžko používal nebo neřešil klientův problém (Schmuller, 2001, s. 14).

UML uspokojuje důležitou potřebu při vývoji softwaru a systémů, a to modelování způsobem, který je snadno srozumitelný a umožňuje vývojáři soustředit se na celkový obraz. UML pomáhá vidět a řešit ty nejdůležitější aktuální problémy, protože brání tendenci nechat se rozptylovat velkým počtem detailů, které je lépe potlačit na pozdější modelování. Při modelování a konstrukci se abstrahuje existující systém reálného světa, což umožňuje klást otázky díky modelu a získávat tak správné odpovědi (Chronoles, Schardt, 2003, s. 10).

4.3.1 Historie jazyka UML

Před rokem 1994 neexistovala žádná obecně přijatá metodika pro vizuální modelování softwarových systémů. A právě potřeba kvalitních a srozumitelných návrhů sebou přinesla i požadavky na standardy notace návrhu, kterým by analytici, klienti a vývojáři rozuměli.

První verze jazyka UML byla vytvořena v roce 1995 společně Grady Boochem, Jamesem Rumbaughem a Ivarem Jacobsonem (Arlow, Neustadt, 2008, s. 29). Před tímto datem se každý z nich snažil vytvořit vlastní metodologii, nicméně zlom nastal až ve chvíli, kdy začali společně pracovat ve společnosti Rational Software. Pozitivní výsledky prvních návrhů jazyka UML a hlad na trhu zapříčily vzniku konsorcia UML. Jeho členy byly společnosti jako Hewlett-Packard, Microsoft, Oracle, Intellicorp, DEC, Rational a další. Toto konsorcium vytvořilo v roce 1997 verzi 1.0 jazyka UML a poskytlo licence skupině OMG (Object Management Group), (Schmuller, 2001, s. 16). OMG převzala podporu a dále vyvíjí další verze. UML se od té doby stalo jakýmsi obecně přijímaným standardem v softwarovém vývoji. Skupina OMG také vytvořila komplexní standardy a požadavky na certifikace, udělované po zvládnutí relativně rozsáhlých testů v různých úrovních znalostí UML notací. Díky společnosti OMG se UML stalo ISO (International Organization for Standardization) standardem všeobecně platným na trhu (Chronoles, 2018, 28).

V průběhu let mezinárodní společnost OMG s volným členstvím, složená například z vládních institucí, akademických institucí, konečných uživatelů a obchodníků, vytvářela a vydávala nové a vylepšené verze UML standardů. Tyto nové verze byly doplněny o další možnosti notací, nové způsoby znázornění a tím se rozšiřovala uplatnitelnost jazyka v praxi. Pro příklad můžeme uvést UML 1.1 (1997), 1.3 (1999), 1.4 (2001), 1.5 (2003) a s větším rozšířením verze 2.0 (2005), (Chronoles, 2018, 28-29). V současnosti je používána aktuální verze 2.5, kterou popíše jedna z dalších kapitol této práce, přičemž je vyvíjena již verze 2.6.

4.3.2 Uživatelské požadavky

Pojem požadavek chápeme jako popis nebo specifikaci jisté funkce nebo i vlastnosti, která by měla být implementována ve vyvíjeném systému. Požadavek je tedy přání uživatele (Kanisová. Müller, 2007, s. 16). Požadavky lze rozdělit na dva základní typy, funkční a nefunkční.

Funkční požadavek může být například zpracování určitého dokumentu nebo zanesení informace do systému a další práce s ní. Další práce s touto informací by byl další funkční požadavek. Nefunkční požadavek může být například rychlost odezvy systému nebo zabezpečení.

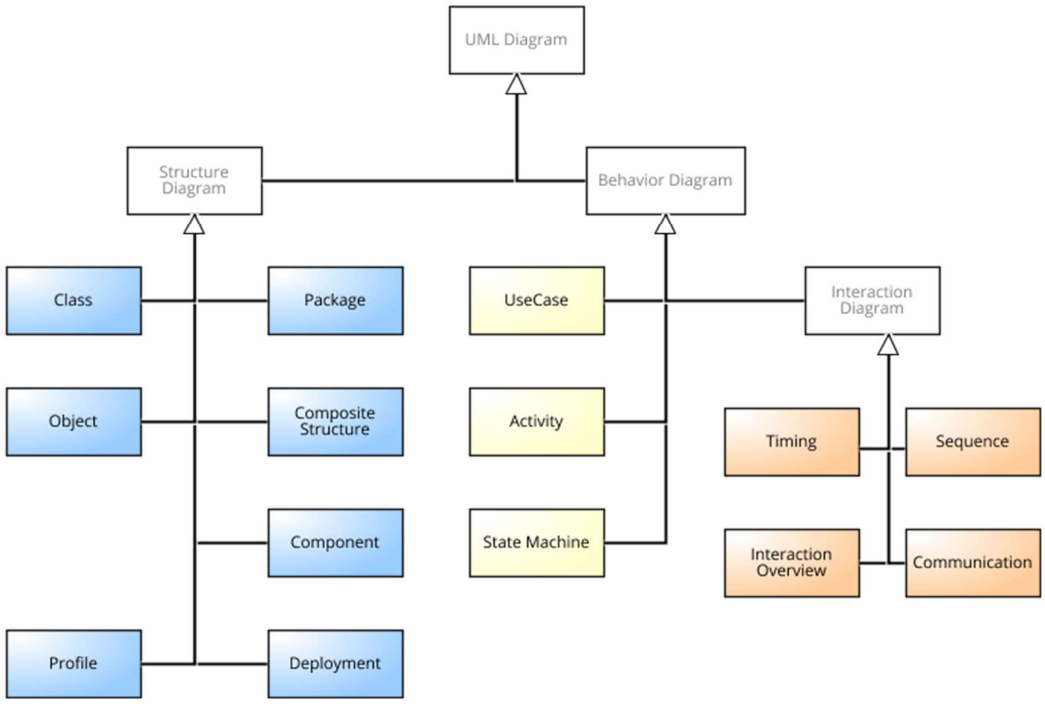
Požadavky na systém nejsou součástí jazyka UML, nicméně jsou s ním spjaté a nástrojem, který se používá pro zachycení potřeby je soubor případů užití (Use Case). Tento soubor představuje veškeré funkce, které má zamýšlený systém nabídnout uživateli. Případy užití nicméně umí zachytit pouze funkční požadavky uživatelů. Nefunkční požadavky musí být promítnuty do technické architektury systému.

Chyba ve zpracování požadavků nebo opomenutí nějakého požadavku vede většinou k neúspěchu celého projektu. Je tedy nutná neustálá kontrola procesu specifikace, přezkoumání a schvalování požadavků se zákazníkem.

4.4 Diagramy jazyka UML

Diagramy jazyka UML můžeme rozdělit, jak je vidět na Obrázku 1, na dvě hlavní skupiny, tedy na strukturální diagramy (Structure Diagram), diagramy chování (Behavior Diagram) a podskupinu diagramy interakcí (Interaction Diagram).

Obrázek 1.: Struktura diagramů podle UML 2.5



Zdroj: UML 2.5 Intro [online]. Dostupné z: https://training-course-material.com/training/UML_2.5_Intro

V dalších kapitolách autor popisuje tři základní modely analýzy a jejich aspekty, podle skript profesora Vrany z katedry informačního inženýrství České zemědělské univerzity.

4.4.1 Diagram tříd

Diagramy tříd (class diagrams) jsou nejběžnější diagramy používané v objektově orientovaných systémech. Zobrazují skupiny tříd, objekty v systému, jejich spolupráci a jejich vztahy. Primárně se tyto náhledy využívají k podpoře požadavků na systém nebo služby, které by měly poskytovat uživatelům (Booch, Jacobson, Rumbaugh, 1998, s. 107).

Třída je kategorie nebo skupina věcí, které mají podobné vlastnosti a stejné nebo podobné chování (Schmuller, 2001, s. 17). Třída má v sobě 3 základní typy popisů. Je to její název, informace o atributech a informace o operacích, které třída dělá. Lze použít i další dodatečné popisy, například omezení třídy. Diagram třídy se využívá k modelování statického náhledu na systém, patří tedy mezi strukturní diagramy. U tříd je důležité určení vztahů mezi nimi. Navzájem mohou být spojené vztahem asociace, agregace, kompozice a generalizace (specializace), (Kanisová. Müller, 2007, s. 55).

Nejdůležitějším popisem třídy je její název. V případě, že již máme název třídy určený, můžeme přistoupit k určení atributů této třídy. Atribut je základní nositel informací o objektu a je definován jménem, formátem a může mít i upřesněnou viditelnost. Název atributu pojmenovává danou vlastnost objektu.

Nedílné součásti popisu objektu při jeho modelování jsou operace. Mohou to být například aktualizace, vykonání operace s daty nebo poskytování rozhraní jiným objektům. Operace musí mít svůj popis stejně jako atributy. Mohou se popsat svým názvem, seznamem parametrů a návratovou hodnotou. Tento popis operace by měl být v modelu unikátní a přesně vystihnout, co daná operace dělá. Například *Zobraz informace* (Kanisová. Müller, 2007, s. 56).

Jako příklad si uvedeme základní atributy pro třídu auto. Třída auto může mít atributy jako značka, model a sériové číslo a vlastní operace jako nastartovat, sednout do auta nebo přidat palivo. Formát atributů by v tomto případě byl pro značku string (řetězec), tedy řada znaků a mezer. Formát atributu model by byl také string a atribut sériové číslo by byl integer, tedy číselná řada omezená celými čísly. Můžeme používat i další typy atributů, jako boolean, který určuje logickou pravdu jako true nebo false. Jiné hodnoty v něm být nemohou. Dále se využívá typ date. Takový atribut musí být ve formátu datum (Chronoles, Schardt, 2003, s. 45). Existují i další speciální datové typy, nicméně ty nebudeme využívat.

Třídy a objekty jsou základními kameny všech objektově orientovaných systémů. Každý objekt je instancí určité třídy, která definuje společnou množinu rysů (atributů a operací či metod), které jsou vlastní všem instancím dané třídy (Arlow, Neustadt, 2008, s. 144).

Příkladem může být opět Škoda 120. Tento typ vymezuje vlastnosti všech specifických instancí této třídy, ale Škoda 120 sériové číslo 25130079 je již specifickou instancí třídy a nazývá se objektem.

4.4.2 Diagram stavů

Stavový diagram (state machine) je specializací diagramů chování systému (behavior diagram). Je to způsob zápisu vývoje chování a změny stavů v tomto vývoji. Chování určitého objektu se modeluje pomocí tzv. uzlů, tedy stavů, které jsou propojeny nasměrovanými přechody. Tyto přechody mezi jednotlivými stavy jsou vyvolány určitou událostí.

V knize *The Unified Modeling Language User Guide (Uživatelská příručka)* je použití stavových diagramů popsáno jako „modelování dynamických aspektů systému, které z větší části zahrnuje specifikaci životního cyklu instancí třídy, případů užití nebo celého systému. Tyto instance mohou reagovat na události, jako jsou signály, operace nebo plynutí jen času. Když dojde k události, dojde k nějaké aktivitě v závislosti na aktuálním stavu objektu“ (Booch, Jacobson, Rumbaugh, 1998, s. 287).

Stavový diagram tedy modeluje dynamické chování objektu v čase a upřesňuje jeho chování. Základními prvky jsou stavy, události a přechody (Booch, Jacobson, Rumbaugh, 1998, s. 292):

1. Stav zvažuje podmínku a situaci, která může nastat. Tedy když objekt odpovídá určité podmínce, vykonává aktivitu, nebo čeká na určitou událost.
2. Událost je určitá aktivita, specifikovaná v čase.
3. Přechod je vztah mezi dvěma stavy objektu a posun od jednoho ke druhému.

Modelováním diagramů stavů popisujeme reagovaní objektů na vnější události, které vznikají mimo tento objekt. A právě tento životní cyklus je modelován jako řada stavů, přechodů a událostí a současně chování objektu v důsledku předchozího impulzu (Arlow, Neustadt, 2008, s. 428).

V našem jednoduchém příkladu s automobilem Škoda by se stavový diagram dal vyjádřit stavy vypnuto a zapnuto. Přechodovou událostí by bylo otočení klíčku v zapalování, tedy událost vypnout a zapnout. Těmito událostmi by se přecházelo ze stavů vypnuto a zapnuto.

4.4.3 Diagram interakcí

Diagram interakcí je součástí diagramů chování a zobrazuje určitou posloupnost aktivit. Každá činnost je v něm zobrazena jako pole aktivit a může obsahovat další vnořené diagramy aktivit.

Podle publikace (Arlow, Neustadt, 2008, s. 252-417) rozlišujeme 4 typy diagramů interakce a každý z nich zobrazuje odlišné aspekty interakce:

1. Sekvenční diagramy (sequence diagrams) vyjadřují časovou posloupnost zpráv a impulzů, předávaných mezi objekty. Znázorňují interakci mezi čárami života (lifeline) jako časově uspořádanou posloupnost akcí.
2. Komunikační diagramy (communication diagrams) zdůrazňují strukturální vztahy mezi objekty, tedy způsob spojení životních čar. Na rozdíl od sekvenčních diagramů se v nich čáry života zobrazují jako komunikační kanály pro předávání zpráv.
3. Diagramy zjednodušené interakce (interaction overview diagrams) ukazují složité chování pomocí množiny jednodušších interakcí. Jsou to stručnější pohledy na aktivity, odkazují na jiné interakce a jejich výskyty.
4. Diagramy časování (timing diagrams) zdůrazňují aspekty související s načasováním interakcí. Znázorňují význam toku času a správné načasování událostí.

Podle skript profesora Vrany z katedry informačního inženýrství České zemědělské univerzity (Vrana, 2014), lze modelování interakcí rozlišit, podle různé úrovně abstrakce. Tedy na vrcholové úrovni jsou to případy užití (Use Case), které popisují interakce systému s okolím. O úroveň níž, tedy podrobnější pohled, jsou sekvenční diagramy. Které popisují výměnu zpráv v čase v rámci množiny objektů. Sekvenční diagramy jsou vhodné pro popis posloupnosti chování z pohledu uživatele. A poslední detailní pohled se zobrazí v diagramu aktivit. Ten ukazuje tok řízení mezi jednotlivými výpočetními kroky (operacemi), uvedenými v sekvenčním diagramu. Jsou v něm tedy datové i řídící toky.

Autor v dalších kapitolách vytvoří a popíše model stavů a případů užití zvoleného systému na správu úvěrů.

4.4.3.1 Diagram případů užití

Základní úkol diagramu případů užití (use case) je sběr požadavků na systém, které má tento systém umět vykonávat. Tyto procesy nebo funkce jsou vykonávány uživatelem systému.

Modelování případů užití je způsob zachycení uživatelských požadavků. Díky tomuto modelu dostaneme povědomí o základních hranicích systému, určíme aktéry a dále specifikujeme funkce a užití systému, určíme i alternativní scénáře (Arlow, Neustadt, 2008, s. 91).

Je to množina akcí, které jsou vykonávané systémem a které vedou k určitému výsledku. Tento výsledek je důležitý právě pro daného účastníka případu užití (actor). Účastníka lze definovat jako roli mimo systém, ve kterém se dané akce vykonávají (Chronoles, 2018, s. 76). Jinými slovy je případ užití popis chování systému a zároveň způsob nashromáždění údajů o požadavcích na systém z pohledu uživatele nebo také aktéra (Schmuller, 2001, s. 18).

Aktéra můžeme definovat jako určitou roli, ve které vystupuje uživatel v rámci komunikace se systémem. Aktéři spouští případy užití. Jeden aktér může v systému spouštět více případů užití, stejně tak jeden případ užití může být prováděn více aktéry (Kanisová. Müller, 2007, s. 37-38).

V návaznosti na minulý příklad s automobilem Škoda můžeme případ rozšířit pomocí případu užití a prostřednictvím definice účastníka. Účastníkem by v tomto případě byl uživatel auta a jeho případem užití by byla jízda autem. Tento uživatel zahajuje případ užití. Uživatelem může být osoba i jiný systém.

5 Praktická část

5.1 Návrh systému pro administraci a správu spotřebitelských úvěrů

Aplikace Loan admin je určena pro poskytování a správu spotřebitelských úvěrů v bankovním prostředí. Její účel je tedy především k u ulehčení a automatizaci kroků při správě úvěrových smluv a přidružených procesů. Aplikace musí umět ukládat a pracovat s daty z úvěrových smluv a umožnit další práci s nimi. Pro účely této práce bude autor pracovat s premisou, že aplikace Loan bude určena pro interní uživatele pracující v oddělení Credit Maintenance finanční instituce. Tito uživatelé vykonávají činnosti spojené se správou smluvní dokumentace, schvalováním úvěrových smluv, změn parametrů úvěrů a zpracování požadavků klientů. Uživatele můžeme rozdělit na 3 skupiny, které mají v procesech každý odlišnou funkci. Jejich popisem se zabývá kapitola 6.3.1 Popis aktérů. Do aktérů jsou zahrnuti i klienti a administrátoři systému.

Hlavními přínosy systému Loan admin je zpracování informací a ustálení procesu schvalování úvěrů a předcházení chybám. Systém umožňuje zautomatizování procesů opakujících se činností, ukládání a zpětnou kontrolu těchto činností. Zároveň je tím zvýšena efektivita a umožněno měření efektivity podle počtů klíčových objektů systému. Tím dochází k výraznému posunu během vyhledávání relevantních informací a tyto informace jsou klíčové k samotnému nastavení úvěru jako produktu finanční instituce. Systém je klíčový v rámci uchovávání kvalitních dat, proto by měl mít možnost nastavit validace na vkládání jednotlivých údajů. Mohou se dále definovat další pravidla stanovující, které položky a atributy jsou nezbytné a jaký mají mít formát. Výsledkem je snížení chybovosti, které je dalším přínosem automatizovaného systému.

5.1.1 Požadavky na systém

Hlavním požadavkem na navrhovaný informační systém je zefektivnění činnosti oddělení Credit Maintenance, zamezení chyb v procesu schvalování úvěrů a bezpečná práce s osobními údaji klientů. Funkce systému vycházejí z níže popsaných požadavků.

Aplikace bude vyvinuta na platformě .NET a bude založena na třívrstvé architektuře. Její architektura tedy bude rozdělena na prezentační vrstvu, aplikační vrstvu a datovou vrstvu.

Prezentační vrstva je viditelná pro uživatele, zajišťuje vstup požadavků a prezentaci výsledků. Aplikační vrstva zajišťujíce výpočty a operace prováděné mezi vstupně-výstupními požadavky a daty. Datová vrstva zajišťuje práci s daty, tedy systém řízení dat a základní datově-funkční operace zajišťující ukládání, výběr, předzpracování, integritu a audit dat.

Funkční požadavky

- Systém umožní zaměstnancům zpracovávat žádosti o úvěr;
- Systém při zpracování žádosti o úvěr musí umožnit hodnocení úvěruschopnosti žadatelů;
- Systém bude generovat smluvní dokumentaci dle zadaných dat.

Nefunkční požadavky

- Systém bude rozlišovat 4 typy uživatelů: Pracovník telefonní linky/bankéř, Schvalovatel, Pracovník back-office, Systémový administrátor;
- Systém bude využívat zabezpečení při přenosu dat a ochranu osobních údajů klientů;
- Systém bude přístupný pomocí internetu.

5.1.2 Popis aktérů

Pracovník telefonní linky/bankéř

Pracovník telefonní linky a bankéř jsou interními pracovníky finanční instituce, kteří zadávají do aplikace požadavky klientů. Tyto požadavky dále zpracovávají další pracovníci, tedy schvalovatelé a back-office.

Schvalovatel

Schvalovatel je interní pracovník finanční instituce, který přezkoumává vstupní data klientů a hodnotí schválení úvěru klientovi. Jeho rozhodnutí je udělit nebo neudělit úvěr klientovi, případně vrátit k opravě žádost na základě prověření informací.

Pracovník back-office

Pracovník back-office je interní pracovník finanční instituce, který zpracovává požadavky klientů, zadané pracovníky telefonní linky nebo bankéři. Generuje v aplikaci úvěrovou dokumentaci a kontroluje vstupní data.

Klient

Klient je zákazník finanční instituce, který má zájem o spotřebitelský úvěr. Klient vstupuje do databáze aplikace v moment uložení žádosti o úvěr.

Systémový administrátor

Systémový administrátor spravuje aplikaci, databázi a servery. Nastavuje také uživatelská práva.

5.1.3 Datový slovník

Datový slovník obsahuje obecný popis obsahu dat v databázi. Definuje pravidla formátování, integrity dat a omezení.

Tabulka 1.: Datový slovník

Entita	Jméno atributu	Typ	Klíč	Komentář
Klient	rodnéČíslo	Int	A	Rodné číslo
	čísloOp	Int	N	Číslo občanského průkazu
	jméno	String	N	Jméno klienta
	příjmení	String	N	Příjmení klienta
	adresa	String	N	Město, ulice a číslo popisné a PSČ klienta
	email	String	N	E-mailová adresa klienta
Schvalovatel	idZaměstnance	Int	A	Číslo zaměstnance
	jméno	String	N	Jméno zaměstnance
	oprávnění	String	N	Oprávnění k úkonům v aplikaci
Pracovník telefonní linky/bankéř	idZaměstnance	Int	A	Číslo zaměstnance
	jméno	String	N	Jméno zaměstnance
	oprávnění	String	N	Oprávnění k úkonům v aplikaci
Pracovník back-office	idZaměstnance	Int	A	Číslo zaměstnance
	jméno	String	N	Jméno zaměstnance

	oprávnění	String	N	Oprávnění k úkonům v aplikaci
Systémový administrátor	idZaměstnance jméno oprávnění	Int String String	A N N	Číslo zaměstnance Jméno zaměstnance Oprávnění k úkonům v aplikaci
Žádost o úvěr	čísloŽádosti stavŽádosti datumPodpisuŽádosti	Int String Datum	A N N	Číslo žádosti Stav žádosti Datum podpisu žádosti
Návrh smlouvy	čísloNávrhu stavNávrhu datumOdeslániNávrhu datumSchváleníNávrhu čísloŽádosti	Int String Date Date Int	A N N N N	Číslo návrhu Stav návrhu Datum odeslání návrhu Datum schválení návrhu Číslo žádosti
Smlouva	čísloSmlouvy stavSmlouvy datumPodpisu platnostOd platnostDo čísloNávrhu	Int String Date Date Date Int	A N N N N N	Číslo smlouvy Stav smlouvy Datum podpisu Platnost smlouvy od data Platnost smlouvy do data Číslo návrhu
Úvěr	čísloSmlouvy úrokováMíra celkováVýšeÚvěru početSplátek výšeSplátky platnostOd platnostDo	Int Int Int Int Int Date Date	A N N N N N N	Číslo smlouvy Úroková míra Celková výše úvěru Počet splátek Výše splátky Platnost smlouvy od data Platnost smlouvy do data
Scoring	rodnéČíslo čísloOp jméno příjmení příjmy	Int Int String String Int	A N N N N	Rodné číslo Číslo občanského průkazu Jméno klienta Příjmení klienta Příjmy klienta

	výdaje	Int	N	Výdaje klienta
	zaměstnání	String	N	Zaměstnání klienta (HPP/DPČ/DPP/IČO)
	dobaZaměstnání	Int	N	Počet měsíců v současném zaměstnání
	zaměstnáníNeurčitáDoba	Boolean	N	Zaměstnání na dobu neurčitou nebo určitou
	věk	Int	N	Věk žadatele
	vzdělání	String	N	Vzdělání žadatele
	výšeDalšíchÚvěrů	Int	N	Hodnota celkových úvěrů klienta
	splátkyDalšíchÚvěrů	Int	N	Hodnota celkových měsíčních splátek klienta
	score	Int	N	Výsledné vyhodnocení schválení úvěru
Prescoring	rodnéČíslo	Int	A	Rodné číslo
	čísloOp	Int	N	Číslo občanského průkazu
	jméno	String	N	Jméno klienta
	příjmení	String	N	Příjmení klienta
	příjmy	Int	N	Příjmy klienta
	výdaje	Int	N	Výdaje klienta
	záznamVRegistrech	Boolean	N	Ohodnocení informací databáze neplatičů
	záznamExekuce	Boolean	N	Ohodnocení informací databáze exekucí
Kalkulace	rodnéČíslo	Int	A	Rodné číslo
	jméno	String	N	Jméno klienta
	příjmení	String	N	Příjmení klienta
	příjmy	Int	N	Příjmy klienta
	výdaje	Int	N	Výdaje klienta
	výšeDalšíchÚvěrů	Int	N	Hodnota celkových úvěrů klienta

	splátkyDalšíchÚvěrů	Int	N	Hodnota celkových měsíčních splátek klienta
	úrokováMíra	Int	N	Úroková míra
	celkováVýšeÚvěru	Int	N	Celková výše úvěru
	početSplátek	Int	N	Počet splátek
	výšeSplátky	Int	N	Výše splátky

5.1.4 Funkční modely informačního systému

S použitím jazyka UML zde bude definován informační pro správu a zpracování úvěrových produktů ve finanční instituci.

5.1.4.1 Diagram tříd aplikace Loan admin

Statická struktura navrhovaného systému je zobrazena pomocí UML diagramu tříd. Vztahy mezi třídami a jejich atributy poskytují další informace o struktuře systému. K popisu jednotlivých tříd informačního systému je nejdříve definováno chování systému, vycházející ze specifikace případů užití.

Kalkulace, prescoring a scoring jsou definovány jako samostatné objekty z důvodu obsahu důležitých atributů v procesu schválení úvěru. S těmito objekty pracují aktéři systému a vkládají do nich data klientů.

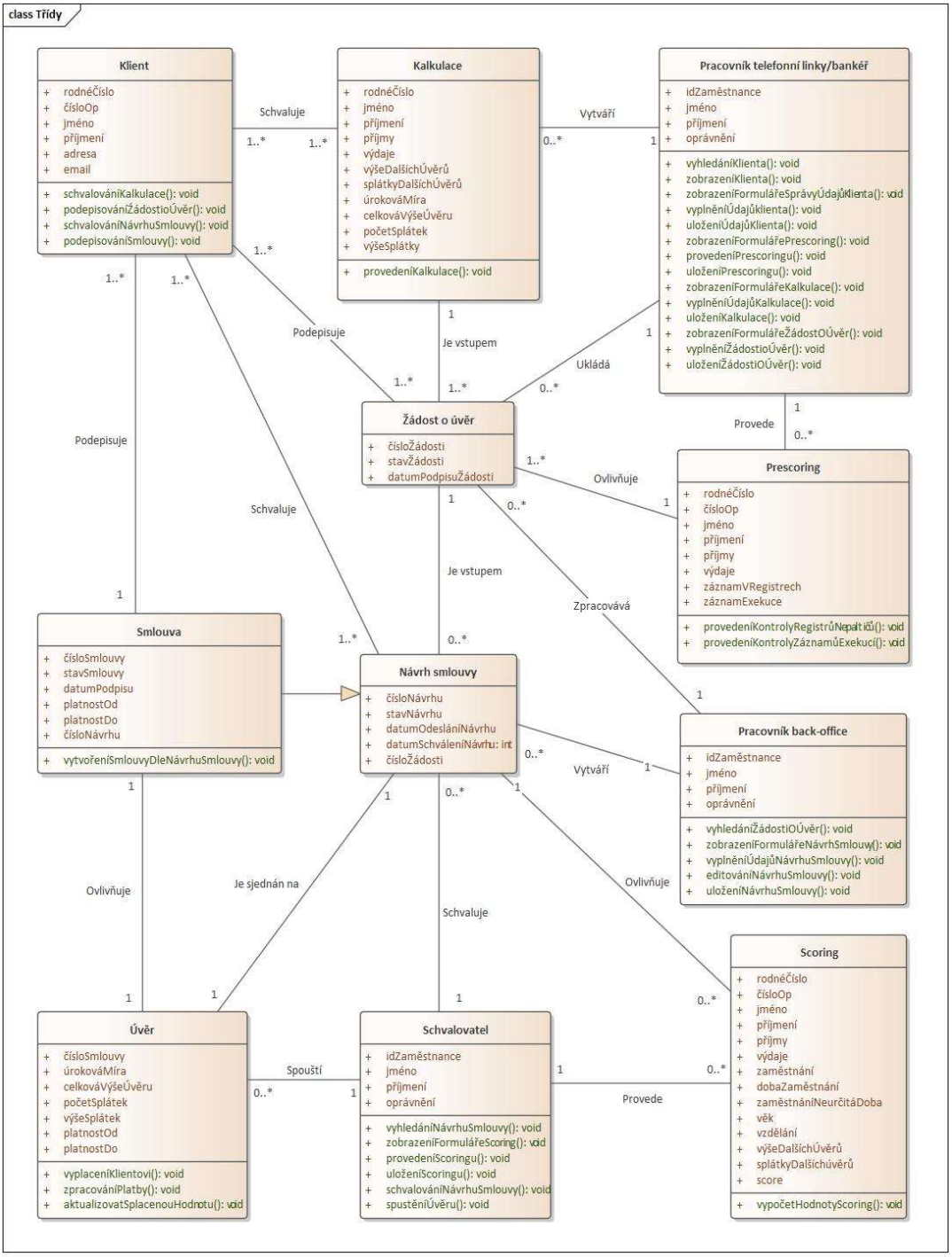
Dokumentace k úvěru je rozdělena do třech samostatných objektů, které mají v procesu rozdílné úlohy, a pracují s nimi různí aktéři. Prvotně pracovník finanční instituce získá potřebná data od klienta a udělá prescoring a kalkulaci úvěru. V případě kladného výsledku prescoringu a schválení kalkulace klientem, tento pracovník vytvoří žádost o úvěr. Při přijetí žádosti o úvěr pracovník back-office vypracuje návrh smlouvy a další aktér systému, tedy schvalovatel zkontroluje všechny údaje v návrhu smlouvy a provede scoring. Scoring je finální prověření finančních možností klienta s množstvím aspektů, které mají vliv na schválení úvěru. Po provedení scoringu schvalovatel udělí číselný výsledek a schválí nebo neschválí úvěr klientovi.

V případě schválení je klient informován o výsledku procesu, pracovník back-office na základě dat v systému vytvoří úvěrovou smlouvu a odešle ji vybranému bankéři, aby zajistil podpis klienta.

V případě neschválení je klient informován o výsledku procesu a je mu vysvětlen důvod rozhodnutí.

Informování klienta o výsledku, respektive podpis smlouvy, probíhá dle volby klienta, tedy buď telefonicky z call-centra, nebo prostřednictvím bankéře. E-mailovou notifikaci klient obdrží v obou případech.

Obrázek 2.: Diagram tříd



5.1.4.2 Stavové diagramy aplikace Loan admin

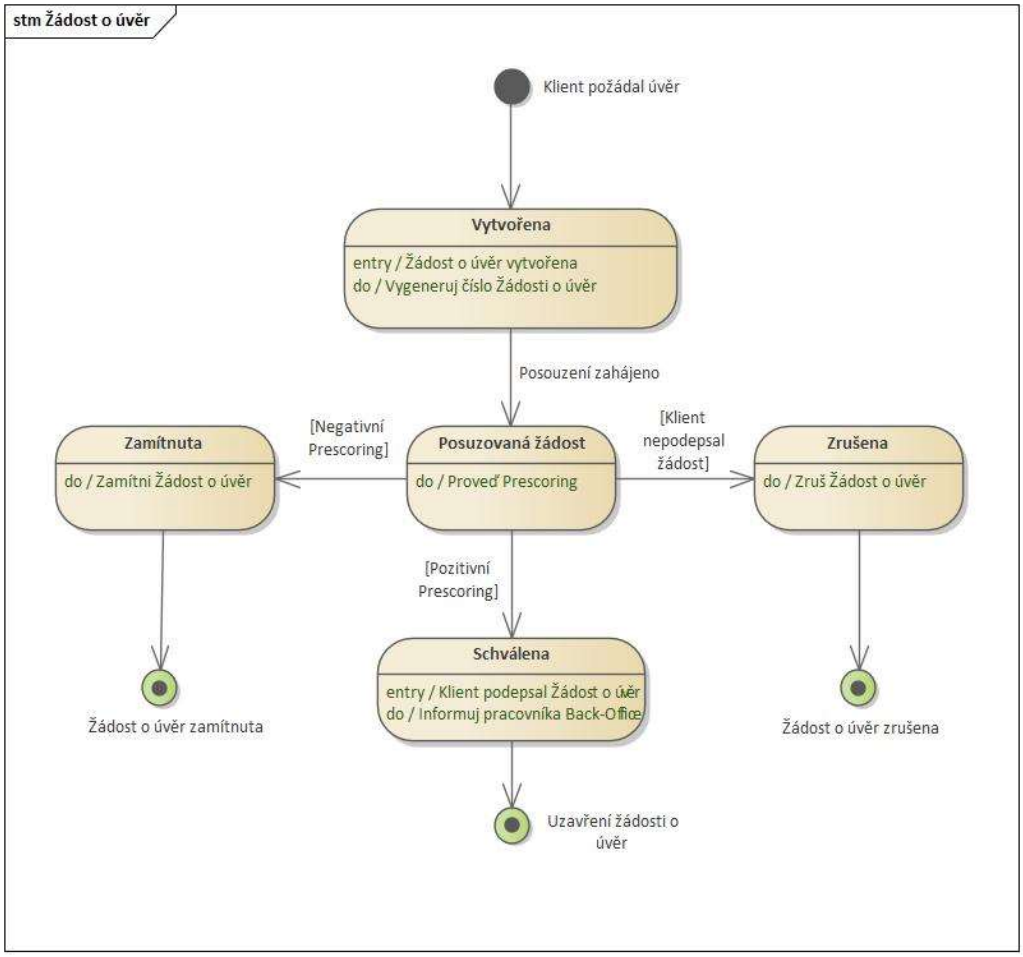
Chování objektů, tedy jednotlivé stavy a přechody mezi nimi, je zobrazeno pomocí stavových diagramů. Tyto diagramy obsahují počáteční a koncový stav.

V návrhu systému jsou vytvořeny stavy pro dokumentaci procesu schvalování úvěru. Jedná se tedy o žádost o úvěr, návrh smlouvy a smlouvu.

5.1.4.2.1 Stavy žádosti o úvěr

Jednotlivé stavy žádosti na úvěr jsou navrženy dle požadavků na systém a obsahují všechny potřebné informace stavů objektu a přechodů mezi nimi.

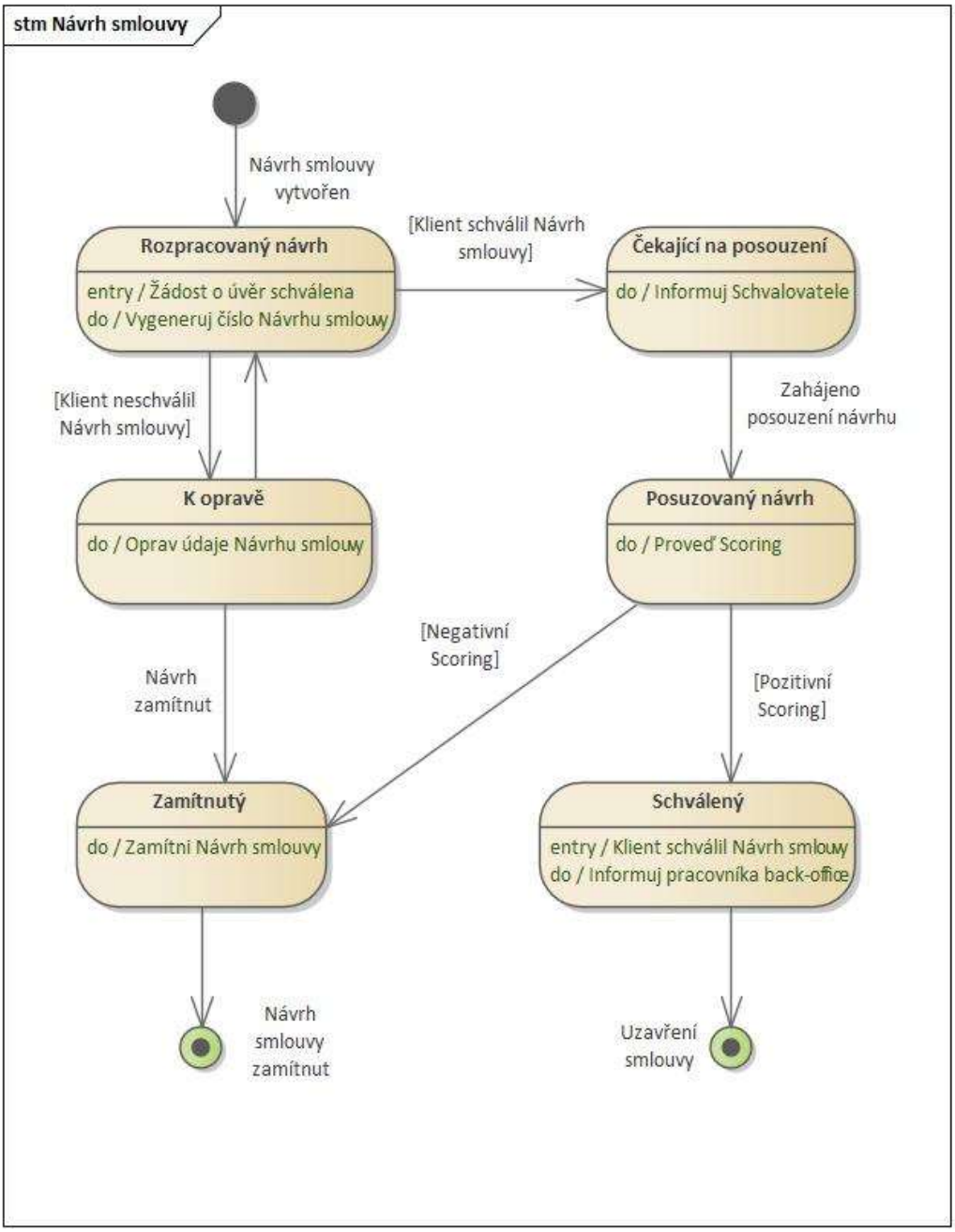
Obrázek 3.: Stavový diagram – Žádost o úvěr



5.1.4.2.2 Stavy návrhu smlouvy

Rozdílné stavy objektu žádost o úvěr jsou zde nutné kvůli kontrole a scoringu prováděného schvalovatelem. Je zde také možnost opravy návrhu v případě nesouhlasu klienta s návrhem.

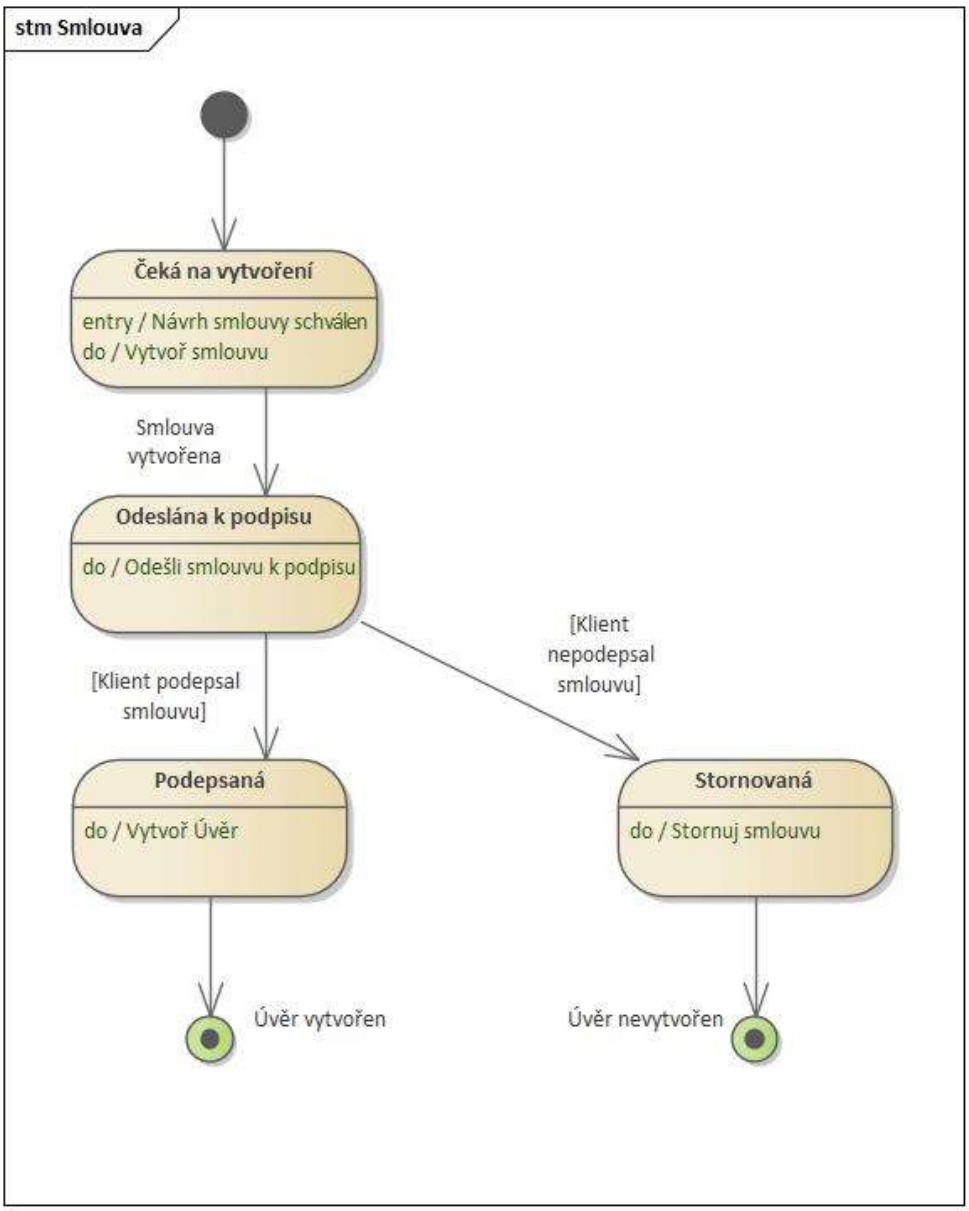
Obrázek 4.: Stavový diagram – Návrh smlouvy



5.1.4.2.3 Stavy smlouvy

V smlouvě je také možná oprava údajů, stejně jako v návrhu smlouvy. Je nutná možnost opakované kontroly a opravy údajů v dokumentech v procesu schvalování úvěru. Musí zde být i možnost stornování celého procesu z důvodu odmítnutí nabídky klientem nebo jako zásah schvalovatele v případě potřeby zastavení procesu. Stav smlouvy *Podepsaná* značí konečný stav, po kterém se vytvoří úvěr dle nastavených a schválených parametrů.

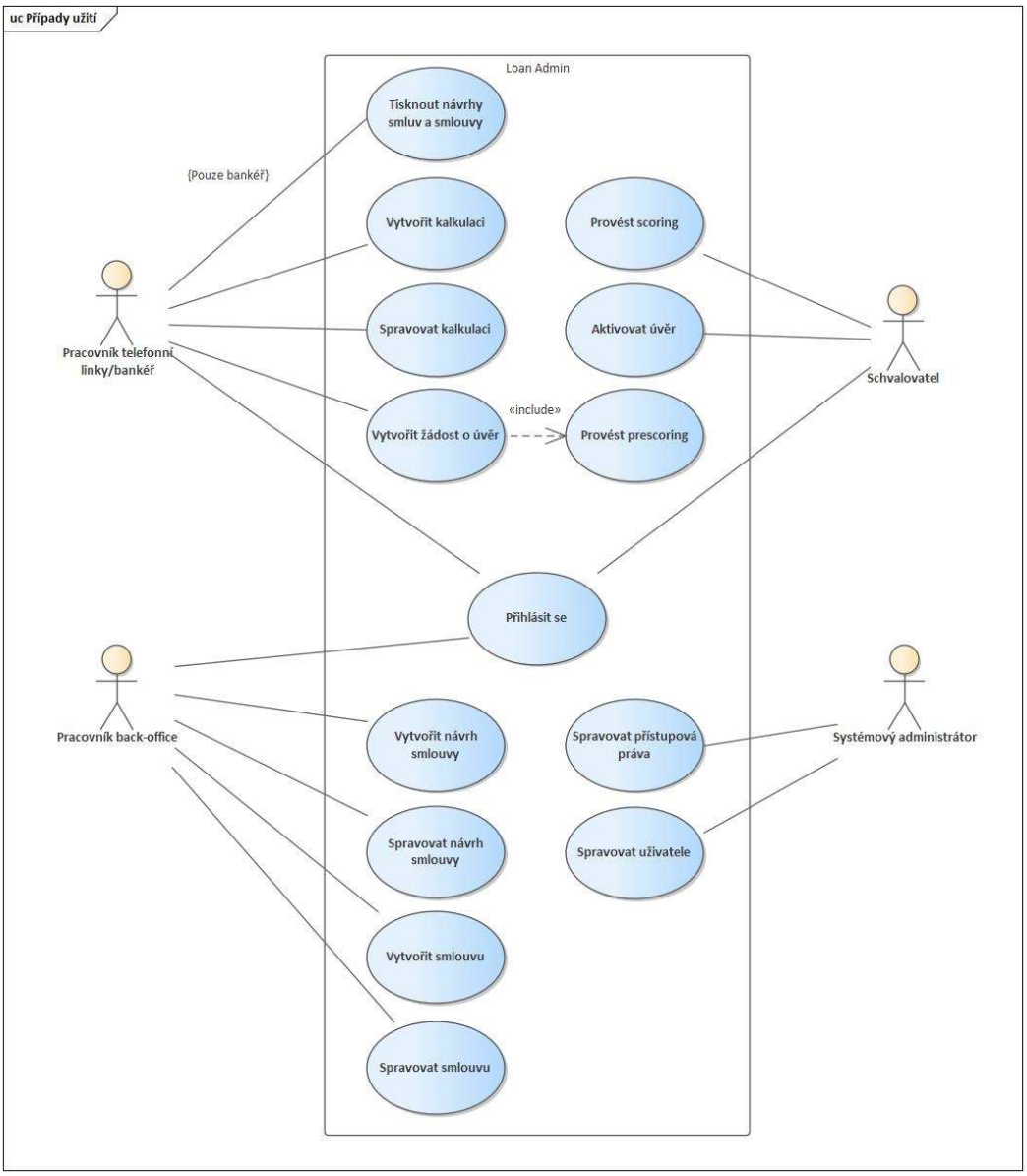
Obrázek 5.: Stavový diagram – Smlouva



5.1.4.3 Diagram případů užití aplikace Loan admin

Diagram případů užití se využívá k zobrazení chování systému z pohledu uživatele a zachycuje, kteří aktéři se systémem pracují a jaké aktivity v rámci systému uskutečňují. Jednotlivé případy užití jsou klíčové v identifikaci klíčových aktérů systému a jejich činností. Tento diagram vychází z funkčních požadavků na systém. Tyto požadavky určují, jaké aktivity a chování má systém nabízet a zpracovávat. Nefunkční požadavky naproti tomu popisují vlastnosti a omezení systému, proto se v diagramu nezobrazují.

Obrázek 6.: Diagram případu užití



6 Závěr

Hlavním cílem této práce bylo provést analýzu a navrhnout model informačního systému určeného pro finanční instituce, a to za účelem poskytování, administrace a zpracování spotřebitelských úvěrů. Informační systém je navrhován tak, aby svým aktivním uživatelům zjednodušil a zefektivnil práci. To mimo jiné zahrnuje navržení procesů tak, aby byla zajištěna vysoká efektivita, srozumitelnost a zároveň aby docházelo k prevenci chyb. K navržení tohoto systému byly použity diagramy v grafickém jazyce UML v nejnovější verzi 2.5.1. Konkrétně byl vytvořen pohled na statickou strukturu systému. Ten je znázorněn diagramem tříd, který zobrazuje vztahy mezi objekty vykonávajícími chování tohoto systému. Dále byl vytvořen stavový model a model interakcí, který byl vyjádřen diagramem případů užití. Součástí návrhu je i analýza uživatelských požadavků a datový slovník.

Dílčími cíli této bakalářské práce bylo vytvoření literární rešerše v oblasti business analýzy a návrhu informačních systémů a dále pak vysvětlení obecných pojmů a přístupů v této problematice. Dalšími dílčími cíli je popis objektového modelování informačních sytému za pomocí UML a snaha vysvětlit obecné charakteristiky UML. Práce se dále zabývala popisem aspektů uživatelských požadavků a jejich dělením.

Při tvorbě modelů autor zvolil CASE nástroj Enterprice Architect 15.1, který umožňuje tvorbu všech diagramů UML. Také umožňuje využití syntaxi nejnovější verze jazyka UML 2.5.

Na závěr je vhodné zmínit, že diagramy použité v této práci se do své finální podoby dotvářely postupně, a to v průběhu celé specifikace systému a požadavků na něj. Tato skutečnost je běžnou praxí, k úpravám požadavků a jednotlivých parametrů ze strany zadavatele zpravidla dochází v průběhu celého procesu tvorby informačního systému.

Dle autora jsou hlavním přínosem práce prezentované diagramy UML, které zobrazují konkrétní systém a které byly vytvořeny na základě analýzy uživatelských požadavků a procesů probíhajících v průběhu zpracování dokumentace ke spotřebitelským úvěrům. Při tvorbě diagramů autor vycházel ze svých pracovních zkušeností a nastudovaných teoretických znalostí.

7 Seznam použité literatury

1. ARLOW, Jim a Ila NEUSTADT. UML 2 a unifikovaný proces vývoje aplikací: Objektově orientovaná analýza a návrh prakticky. Brno: Computer Press, 2008. ISBN 978-80-251-1503-9.
2. BOOCH, Grady, Ivar JACOBSON a James RUMBAUGH. *The unified modeling language user guide*. Addison Wesley, 1998. ISBN 0-201-57168-4.
3. CHONOLES, Michael Jesse. *OCUP 2 Certification Guide: Preparing for the OMG Certified UML 2.5*. United States: Elsevier Books, 2018. ISBN 978-0-12-809-640-6.
4. CHONOLES, Michael Jesse a James A. SCHARDT. *UML 2 For Dummies*. New Jersey: John Wiley, 2003. ISBN 978-0-7645-2614-5.
5. COOK, Steve, Conrad BOCK, Pete RIVETT, Tom RUTT, Ed SEIDEWITZ, Bran SELIC a Doug TOLBERT. *OMG Unified Modeling Language TM (OMG UML)*. Version 2.5.1. Object Management Group (OMG), 2015. Dostupné z www: https://www.omg.org/spec/UML/2.5/PDF.
6. KANISOVÁ, Hana a Miroslav MÜLLER. *UML srozumitelně*. 2. vydání. Brno: Computer Press, 2007. ISBN 80-251-1083-4.
7. MERUNKA, Vojtěch, Robert PERGL a Marek PÍCKA. *Objektově orientovaný přístup v projektování informačních systémů: skriptum.* Praha: ČZU, 2005. ISBN 80-213-1352-8.
8. POTANČOK, Martin, Jan POUR a Veronika CHRAMOSTOVÁ. *Business analytika v praxi*. Praha: Vysoká škola ekonomická, Nakladatelství Oeconomica, 2020. ISBN 978-80-245-2382-8.
9. SCHMULLER, Joseph. *Myslíme v jazyku UML: knihovna programátora*. Praha: Grada publishing, 2001. ISBN 80-247-0029-8.
10. VRANA, Ivan. *Skripta katedry Informačního inženýrství České zemědělské univerzity: Projektování informačních systémů s UML*. Praha: ČZU, 2014. ISBN 978-80-213-1817-5.