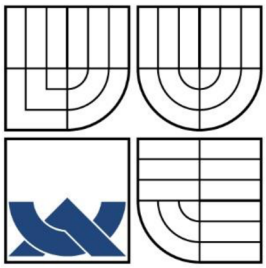
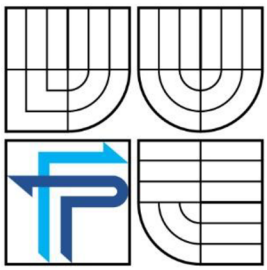


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA PODNIKATELSKÁ

ÚSTAV INFORMATIKY

FACULTY OF BUSINESS AND MANAGEMENT

DEPARTMENT OF INFORMATICS

VÝVOJ ADMINISTRAČNÍHO MODULU V PROSTŘEDÍ MS ACCESS

DEVELOPMENT OF ADMINISTRATIVE MODUL IN MS ACCESS ENVIRONMENT

BAKALÁŘSKÁ PRÁCE

BACHELOR´S THESIS

AUTOR PRÁCE — MICHAL HOLÍNEK

AUTHOR

VEDOUCÍ PRÁCE — Ing. PETR DYDOWICZ, Ph.D.

SUPERVISOR

BRNO 2014

Vysoké učení technické v Brně
Fakulta podnikatelská

Akademický rok: 2013/2014
Ústav informatiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Holínek Michal

Manažerská informatika (6209R021)

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách, Studijním a zkušebním řádem VUT v Brně a Směrnicí děkana pro realizaci bakalářských a magisterských studijních programů zadává bakalářskou práci s názvem:

Vývoj administračního modulu v prostředí MS Access

v anglickém jazyce:

Development of Administrative Modul in MS Access Enviroment

Pokyny pro vypracování:

Úvod
Vymezení problému a cíle práce
Teoretická východiska práce
Analýza problému a současné situace
Vlastní návrh řešení, přínos práce
Závěr
Seznam použité literatury

Seznam odborné literatury:

BRADEN, Melanie a Michael SCHWIMMER. Excel 2007 VBA. Velká kniha řešení. Brno: Computer Press, a.s., 2009. 685 s. ISBN 978-80-251-2698-1.
ČIHAŘ, Jiří. 1001 tipů a triků pro Microsoft Excel 2007/2010. Brno: Computer Press, a.s., 2011. 488 s. ISBN 978-80-251-2587-8.
KRÁL, Martin. Excel VBA. Výukový kurz. Brno: Computer Press, a.s., 2010. 504 s. ISBN 978-80-251-2358-4.
KRÁL, Mojmír. Excel 2010 – snadno a rychle. Praha: Grada Publishing a.s., 2010. 143 s. ISBN 80-2473-495-8.
LAURENČÍK, Marek. Programování v Excelu 2007 a 2010. Praha: Grada Publishing a.s., 2011. 192 s. ISBN 978-80-247-3448-4.

Vedoucí bakalářské práce: Ing. Petr Dydowicz, Ph.D.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2013/2014.

L.S.

doc. RNDr. Bedřich Půža, CSc. Ředitel ústavu	doc. Ing. et Ing. Stanislav Škapa, Ph.D. Děkan fakulty

V Brně, dne 02.06.2014

Abstrakt

Tato bakalářská práce se zabývá analýzou současného stavu informačního systému společnosti Allrisk, s.r.o., a návrhem nového řešení, které společnosti umožní administraci nového produktu.

Abstract

This thesis analyzes the current state of information system of Allrisk, s.r.o., and the proposal of a new solution that will allow the administration new product.

Klíčová slova

Databáze, informační systém, MS Access, VBA, PostgreSQL.

Keywords

Database, information system, MS Access, VBA, PostgreSQL.

Bibliografická citace:

HOLÍNEK, Michal. *Vývoj administračního modulu v prostředí MS Access*. Brno: Vysoké učení technické v Brně, Fakulta podnikatelská, 2013. Vedoucí bakalářské práce Ing. Petr Dydowicz, Ph.D.

Čestné prohlášení

Prohlašuji, že předložená bakalářská práce je původní a zpracoval jsem ji samostatně. Zároveň prohlašuji, že citace použitých pramenů je úplná, že jsem ve své práci neporušil autorská práva (ve smyslu zákona č. 121/200 Sb., o právu autorském a o právech souvisejících s právem autorským).

V Brně dne 29. května 2014

.....

Podpis

Poděkování

Tímto bych rád poděkoval vedoucímu mé bakalářské práce panu Ing. Petru Dydoviczi, Ph.D., za odborné rady a cenné připomínky, které jsem využil při zpracování této bakalářské práce. Dále bych chtěl poděkovat vedení společnosti Allrisk, s.r.o., za poskytnutí součinnosti a podkladových materiálů.

OBSAH

ÚVOD

V posledních letech došlo k tak velkému rozmachu informačních technologií, že bez práce na PC a internetu by společnost téměř nemohla fungovat. K tomu, abychom byli schopni ukládat a navzájem sdílet a využívat data, je potřeba centrální úložiště např. databáze. V současné době by se již nikdo nechtěl probírat stohy papíru a ručně v nich vyhledávat potřebná data.

V dnešní době vypadají datová úložiště jinak než na počátku jejich využívání. S rozvojem moderních technologií jsou požadavky na databáze mnohem větší než dříve, kdy se většina úkonů musela provádět složitě ručně a docházelo k nepřesnostem. Nejnáročnější byla část administrativní, jako jsou např. evidence změn, zaznamenávání v katalozích apod. V dnešní době se většina informací uchovává v elektronické podobě, což umožňuje nejen menší časovou náročnost, ale i jednodušší a rychlejší přístup k informacím. V současné době stačí správné jméno a heslo k požadovanému účtu a informace můžeme mít k dispozici odkudkoliv.

Vybudováním optimalizované aplikace lze z dlouhodobého hlediska ušetřit nemalé prostředky, které by jinak musely být vynaloženy na zaměstnance, kteří by tuto práci museli provádět ručně.

VYMEZENÍ PROBLÉMU A CÍLE PRÁCE

Cílem této práce je navrhnout a zrealizovat novou aplikaci v prostředí MS Access pro společnost Allrisk, s.r.o., a to na základě požadavků, které vycházejí z jejích potřeb.

V první kapitole jsou sepsány nezbytné teoretické poznatky, které byly v pozdější fázi vývoje aplikace použity.

Druhá kapitola se zaměřuje na představení společnosti. Popsána bude její organizační struktura, předmět podnikání, zmíněna její historie. Poté bude provedena analýza jejího informačního systému, který využívá pro své podnikání.

Praktická část bude věnována vlastnímu návrhu a vytvoření aplikace. Dále bude provedena cenová kalkulace. Návrh musí splňovat požadavky zadavatele. Při vytváření aplikace se vychází z provedené analýzy a z teoretických poznatků zpracovávaných v kapitole jedna a dvě.

V závěru bude zhodnoceno, zda autorem vytvořená aplikace splňuje všechny požadavky zadavatele.

1 TEORETICKÁ VÝCHODISKA

V této části jsou vyzdvižena nejdůležitější teoretická východiska. Zaměřuje se především na oblast související s návrhem a realizací databázové aplikace za použití jazyka VBA v prostředí MS Access 2003.

1.1 VBA

Programovací jazyk VBA (celý originální název Visual Basic for Application) je objektově orientovaný jazyk, který je v kancelářském balíku MS Office využívaný. Vlastní programový kód se zapisuje jako posloupnost jednotlivých příkazů. Části kódu mohou být organizovány do větších bloků např. funkcí a procedur. Napsané funkce a procedury lze dále organizovat do větších celků, jako jsou např. moduly. [1]

Jestliže je užito více příkazů na jenom řádku, použije se k oddělování dvojtečka. Pokud se příkaz nevejde na jeden řádek kódu, použije se na konci řádku dolní podtržítko, musí mu však předcházet mezera.

1.1.1 Proměnné a datové typy

Nejdříve je nutné popsat, co je *proměnná*. Jde o vyhrazený prostor v paměti, do kterého se ukládají data v programu. V jazyku VBA se proměnné mohou deklarovat dvěma způsoby: [2]

- **Explicitní deklarace** – rezervace paměti pro proměnnou se provádí za běhu programu. Zapíše se před jejím prvním použitím, většinou na začátku procedury či funkce. Při samotné deklaraci můžeme zadat i datový typ, který bude proměnná používat, to však není povinné. Pokud se datový typ nezapíše, proměnná získá automaticky datový typ *Variant*. To znamená, že proměnná může dostat jakoukoliv hodnotu libovolného datového typu. Např.: Dim Moje_jmeno as String. [2]
- **Implicitní deklarace** – Jedná se o deklaraci bez klíčového slova *Dim*. Na první pohled se může zdát lepší či rychlejší, ale má několik nevýhod či rizik. Prvním specifikem je, že proměnná má automaticky datový typ *Variant*, který zabírá větší množství paměti než jiné datové typy, což může vést ke zpomalení běhu programu. Zpomalení nemusí být v menších programech příliš znatelné, ale ve větších celcích už může být rozdíl znatelný. Hlavní nevýhodou tohoto typu deklarace je, že není

ošetřen výskyt překlepu ve jménu proměnné. Překlep je poté považován za novou proměnnou. Např.: Moje_jmeno="Michal". [2]

Typy a platnost proměnné

U proměnné je velice důležité, kdy a kde může být použita a ve které situaci si ponechá či ztratí svoji hodnotu. Z tohoto hlediska můžeme proměnné deklarovat následujícími způsoby:

Dim – proměnná je dostupná pro jednu proceduru či funkci v daném modulu. Může být použita jen v ten okamžik, kdy modul běží.

Static – jedná se o statickou proměnnou v dané proceduře či funkci.

Private – proměnná je dostupná pro všechny funkce či procedury v daném modulu.

Public – dostupnost proměnné tohoto typu je ve všech modulech, funkcích a procedurách. Zachovává si svou platnost i po skončení dané procedury či funkce. [3]

Datový typ

Zjednodušeně lze říci, že datový typ definuje a vymezuje hodnoty, kterých může proměnná nabývat. Pojmenování jednotlivých datových typů je specifické pro každý programovací jazyk.

1.1.2 Rozhodovací konstrukce

Většina programů napsaných v některém z programovacích jazyků se na první pohled může zdát jako monolitní celek, ale není tomu tak. Struktura kódu obsahuje jakýsi vlastní mechanizmus, který řídí obsažené programové instrukce. Hlavní částí tohoto mechanizmu jsou rozhodovací konstrukce, které mají za úkol dohlížet na samotný běh programu, a v některých okamžicích ho podle předem dané logiky ovlivňují. Za pomoci právě rozhodovacích konstrukcí může program reagovat na okolnosti, které se mění či vznikají, až za běhu samotného programu. [5]

Konstrukce If-Then (a její varianty)

Základní rozhodovací konstrukcí v jazyku VBA je kontrukce *If-Then*. Její deklarace se provádí pomocí příkazu *If*.

Po klíčovém příkazu *If* následuje podmínka, která je testována. Podmínkou může být příkaz, u kterého lze určit pravdivostní hodnotu. Podmínka je splněna, pokud má

pravdivostní hodnotu *True*. Jestli je podmínka splněna, provádějí se příkazy, které jsou zapsány mezi klíčovými slovy *If* a *Then*. V situaci, kdy podmínka splněna není, pokračuje program na první příkaz, který následuje za ukončujícím příkazem konstrukce *End If*. [5]

Konstrukce Select-Case

V tomto typu konstrukce se výraz vyhodnocuje v závislosti na jeho hodnotě. Po vyhodnocení se spustí jedna z větví kódu. Výhodou tohoto typu konstrukce je možné větvení na velké množství větví.

Konstrukce se skládá se základních klíčových slov *Select Case,* po kterých následuje porovnání hodnot, výpočet nebo jen proměnná. Konstrukce obsahuje spojení klíčových slov *Case Else*. Příkazy, které následují po těchto slovech, se provádějí v tom případě, že ani jedna z podmínek v jednotlivých větvích není splněna. Po klíčovém slovu *Case* v jedné z větví může následovat i rozsah povolených hodnot, při kterých je podmínka splněna. Např.: *Case 1 to 10*. [6]

1.1.3 Konstrukce cyklů

Cykly se při psaní programu používají v případě, kdy určitou část kódu potřebujeme provádět opakovaně za sebou. Cykly stejně jako rozhodovací konstrukce mají svou hlavičku a zakončení, mezi které se zapisují jednotlivé příkazy. V jazyce VBA se rozlišují dva typy cyklů. Záleží na tom, zda na začátku operace víme, kolikrát se cyklus má či nemá opakovat . V případě, že přesný počet není znám, porovnávají se při každé iteraci podmínky, které vyhodnotí, zda se má, nebo nemá v iteraci pokračovat. [6]

Cyklus For – Next

V případě, kdy známe přesný počet opakování, použijeme cyklus *For – Next*. Syntaxe cyklu je následující:

„For *čítač* = *začátek* To *konec* [Step *krok*] [*příkazy*] Next [*čítač*]“ [7]

Cyklus Do … While/Until

Jestliže neznáme přesný počet opakování cyklu a nelze ho určit proměnnou, použijeme cyklus typu *Do ... While/Until*. Jeho konstrukce mohou mít dvě formy podle toho, kde je zapsána vyhodnocovací podmínka.

1.1.4 Procedury ve VBA

Obecně lze říci, že procedury ve VBA jsou posloupnost příkazů, které se postupně vykonávají. V jazyku VBA se dělí procedury na dva typy. Prvním je procedura typu Subrutiny, druhým procedura typu funkce. Některé procedury očekávají vstupní parametry, kterým se říká *atributy*. Všechny příkazy, až na některé výjimky, musí být v jazyku VBA zapsány v procedurách. První výjimkou jsou proměnné deklarované na úrovni modulu. Jestliže deklarujeme uživatelské datové typy, nemusejí být zapsány v procedurách ani ty. Poslední výjimkou jsou volby s globální platností, např.*: Option Explicit, Option Base 1, Option Private.* Názvy procedur by měly vystihovat jejich účel. Nelze v nich používat klíčová slova nebo jména globálně deklarovaných proměnných či uživatelských typů.[8]

Procedury typu Sub

Někdy se označují jako klasické. Tento typ procedury nevrací kromě svého jména do volacího programu žádnou hodnotu. Používají se například pro načítání vstupních informací od uživatele, k vytištění nebo zobrazení různých informací. Mohou být také použity k manipulaci s vlastnostmi spojenými s nějakou podmínkou.[8]

Procedury typu Function

Někdy jsou také označovány jako funkční procedury. Jedná se o sadu programových příkazů, které jsou ohraničeny na začátku klíčovým slovem *Function* a na konci klíčovým slovem *End Function*. Programové příkazy obsažené ve funkci vykonávají jistou smysluplnou činnost. Každá z funkcí vrací do volací procedury určitou hodnotu. Výsledná hodnota je předávána prostřednictvím svého jména. Funkce vrací vždy právě jednu výslednou hodnotu, která je většinou předávána do jména funkce v posledním z přiřazovacích příkazů. Konstrukce procedury typu *Function* je podobná jako v předchozím příkladu, ale má drobné speciality.[8]

Volání procedur

Pro volání uložených procedur v jazyku VBA je několik způsobů. Například můžeme použít klávesu F5. Po jejím stisknutí se spustí procedura, ve které byl nastaven kurzor. Pokud však procedura očekává vstupní parametr, procedura se neprovede. Další z možností, jak spustit proceduru, je kliknutí na nějaký objekt ve formuláři. Proceduru

lze spustit i jako výskyt nějaké události. Další z možností, jak procedury spustit, je z okna *Imediate*. Stačí napsat název procedury, zapsat případné parametry a poté ji spustit pomocí klávesy Enter. Posledním a nejpoužívanějším způsobem je volání z jiné procedury. Můžeme zapsat jen název procedury, která se má zavolat. Lze také použít klíčové slovo *Call*. Příklady použití volání procedury:

- Moje_procedura(hodnota).
- Call Moje_procedura(hodnota).
- Vysledek = Moje_procedura(5).

1.2 Databáze

Už samotné slovo databáze naznačuje, že má co do činění s daty samotnými. Ukládají se do ní jednotlivá data, která jsou primárně určena k dalšímu zpracování. [9]

1.2.1 Uložení dat

Data nejsou primárně v databázích náhodně ukládána na jedné hromadě. Mají svoji přesně danou strukturu, která je velice důležitá pro pozdější práci s nimi, např. vyhledávání v datech, třídění dat apod. [9]

1.2.2 Relační databáze

V reálných databázích nejsou data uložena společně v jedné tabulce. Tento způsob by byl velice neefektivní a nepřehledný. Rozdělena jsou do logických skupin a uložena do samostatných tabulek. [9]

Aby bylo možno pracovat s tabulkami najednou, je potřeba je spojit. K tomu pomáhají tzv. relační vztahy, které mohou mít více vlastností a pravidel pro své použití. S jejich pomocí můžeme tedy vybírat data uložená v různých tabulkách, což nám pomáhá k vytváření informací z dat. [9]

1.3 MS Access

Databázový program MS Access je součástí balíku MS Office od společnosti Microsoft. Je určen uživatelům, kteří potřebují pracovat s databázemi a údaje z těchto databází vybírat či prezentovat v nějaké z forem, které produkt nabízí. [10]

1.3.1 Objekty v MS Access

Data uložená v programu MS Access mají formu tabulek, které se uloží do jediného spustitelného souboru, do něhož se také ukládají ostatní objekty v databázi. V následující části budou jednotlivé typy objektů popsány. [11]

Tabulky

Tabulky jsou objekty, ve kterých jsou uložena data. Vztahy mezi jednotlivými tabulkami jsou řešeny formou relací.

Můžeme je tvořit několika způsoby. První z možností je vytvoření tabulky pomocí *šablony*. V Accessu je několik předem připravených šablon, které lze využít. Další možností, kterou lze využít, je *vyplnění prázdné tabulky*. Pokud změníme názvy sloupců, změní se i názvy polí tabulky. Třetí z možností je úprava *návrhu tabulky*. Tento způsob dovoluje detailně definovat vlastnosti polí, definovat pole tabulky a její vlastnosti. Tabulky můžeme také importovat z jiných zdrojů, jakým je např. Excel. [10]

Pracovat s tabulkami můžeme v několika základních zobrazeních. Máme možnost zobrazit si je v tzv. *návrhovém zobrazení*, ve kterém lze měnit jejich strukturu, nastavovat vlastnosti polí nebo celé tabulky. Dalším typem je *zobrazení datového listu*, jež slouží především k prohlížení dat, která můžeme filtrovat, řadit, hledat v nich nebo doplňovat jejich souhrny. Speciální je tzv. *zobrazení kontingenční tabulky*, určené především k vizuální analýze dat. Poslední z možností je tzv. *zobrazení kontingenčního grafu*. Zjednodušeně lze říci, že v kontingenčním grafu jsou graficky zobrazena data z kontingenční tabulky. [11]

Dotazy

Dotazy lze rozdělit do dvou základních skupin. Dělí se podle dopadu na data. Základní skupiny jsou výběrové a akční.

Prvním z typů jsou dotazy *výběrové*. Vyznačují se tím, že zobrazují data z jedné či více tabulek. Pro filtrování dat lze použít různá omezovací pravidla. Můžeme např. omezit počet sloupců, které se skutečně zobrazí. MS Access nám také dovoluje zvětšit rozsah výstupu kombinací různých tabulek. Výstup je možné také rozšířit o nová pole, která jsou dána výpočtem z ostatních sloupců. Speciálním typem výstupu je situace, kdy si

připravíme křížovou tabulku, v níž analyzujeme vztah dvou nebo více polí tabulky nebo více tabulek. [11]

Akční dotazy, již podle svého názvu, provádějí s tabulkami různé akce. Prvním z typů jsou *vytvářecí* dotazy. Slouží pro vytvoření nové tabulky z obsahu jiné tabulky nebo více tabulek. Dalším typem jsou dotazy *aktualizační*. Již jejich název naznačuje, že slouží k aktualizaci hodnot v tabulce. K odstraňování vět z tabulky slouží *odstraňovací* dotazy. Opakem dotazů odstraňovacích jsou dotazy *přidávací*, které slouží k přidávání vět do tabulky. [11]

Výsledky výběrových dotazů můžeme znázorňovat jako tabulku v zobrazení datového listu formou kontingenční tabulky nebo formou kontingenčního grafu.

Formuláře

Třetím typem objektu po tabulkách a dotazech jsou v aplikaci MS Access formuláře, které především uživateli slouží k usnadnění práce. Může si přes ně zadávat, upravovat či zobrazovat data. Výhodou zobrazení dat na formuláři ve srovnání s tabulkou je to, že v něm nemusí být všechna pole a není nutné, aby byla jen v jedné řadě jako v tabulce.

Formuláře se dělí podle toho, kolik obsahují vět:

Samostatný formulář se od datového listu liší v tom, že se zobrazují data jen z jediné věty. Další druhy formulářů jsou jen kombinací mezi zobrazením datového listu nebo dotazu. Prvním z kompromisů je tzv. *rozdělený formulář*, na kterém je kromě formuláře zobrazen i datový list tabulky, který je jako podkladový. Druhým z kompromisů je *nekonečný formulář*, který je vhodný pro přidávání tlačítek a obrázků. [11]

Zobrazování formulářů je několik typů:

Prvním je *formulářové zobrazení*, při kterém není možné dělat změny, formulář lze jen jednoduše používat. Druhým je *zobrazení rozložení.* Na formuláři v tomto typu zobrazení můžeme provádět změny. Využívá se především pro nastavení velikosti polí, protože na formuláři jsou zobrazena data. Třetím je tzv. *návrhové zobrazení*. Při tomto typu zobrazení můžeme na formuláři provádět jakékoliv změny a úpravy, ale nejsou zobrazena data. Poslední tři typy zobrazení jsou *zobrazení datového listu, zobrazení kontingenční tabulky a zobrazení kontingenčního grafu.* U formuláře se téměř nevyužívají. Používají se spíše u tabulek a dotazů. [11]

Sestavy

V MS Access slouží pro prezentování dat nebo k tisku dat tiskárnou objekt se jménem *sestava*. V aplikaci můžeme tisknout i formuláře, ale sestavy jsou pro tyto účely mnohem lépe použitelné. V sestavě se tisknou vždy všechny věty, které jsou uloženy v podkladové tabulce či podkladovém dotazu. Věty můžeme omezovat pomocí různých podmínek. Typy zobrazení jsou podobné jako v případě formuláře. Rozdílem je jen zobrazení typu *náhled*. [11]

1.4 SWOT analýza

Základním úkolem SWOT analýzy je identifikace, do jaké míry jsou aktuálně používané strategie společností využívány. Jejím úkolem je dále definovat silná a slabá místa a zjistit, jak je společnost schopná vyrovnat se se změnami, které způsobuje její okolí. Rozdělení základních částí SWOT analýzy je následující: [12]

Silné stránky Výhody pro zákazníky i společnost.	**Slabé stránky** Skutečnosti, které společnost nevykonává dobře nebo konkurence v nich je lepší.
Příležitosti Skutečnosti zvyšující poptávku nebo lépe uspokojují zákazníky.	**Hrozby** Skutečnosti či trendy, které mohou snižovat poptávku nebo způsobit nespokojenost zákazníků.

Obrázek 1: SWOT analýza

Zdroj: [12]

SWOT analýza se skládá ze dvou základních analýz – SW a OT. Doporučený postup je začínat nejprve druhou jmenovanou, což je analýza příležitostí a hrozeb. V ní se zkoumají nejen příležitosti a hrozby z vnějšího prostředí, nazývaného též *makroprostředí*. Jedná se např. o technologické, politické, právní, ekonomické či sociálně-kulturní prostředí. [12]

Dalším prostředím zkoumaným v této části je tzv. *mikroprostředí*. Sem spadají např. zákazníci, dodavatelé, odběratelé a konkurence. Po podrobné OT analýze následuje analýza SW, která se zabývá vnitřním prostředím společnosti. Patří sem mezilidské vztahy, kultura, organizační struktura a management. [12]

2 ANALÝZA SOUČASNÉHO STAVU

Nyní bude popsána druhá významná část – analýza současného stavu, jež se bude zabývat základními informacemi o společnosti. Zaměřím se na její organizační strukturu. Jedna sekce se zaměří na portfolio produktů společnosti. V neposlední řadě budou popsány jednotlivé části informačního systému společnosti a provedena potřebná analýza ke zjištění potřeb společnosti v oblasti vývoje nových částí informačního systému. Poslední část této sekce bude věnována zhodnocení provedených analýz.

2.1 Základní informace

Obchodní firma:	Allrisk, s.r.o.
Sídlo:	Brno-Komárov, Komárovská 263/20, PSČ 617 00.
Identifikační číslo:	49610929.
Právní forma:	Společnost s ručením omezeným.
Předmět podnikání:	Činnosti zástupců pojišťovny a makléřů. Zprostředkování velkoobchodu a velkoobchod v zastoupení. Maloobchod v nespecializovaných prodejnách. Programování. Činnosti v oblasti nemovitostí na základě smlouvy nebo dohody. Zprostředkovatelské činnosti realitních agentur. Poradenství v oblasti řízení. Ostatní profesní, vědecké a technické činnosti. Ostatní vzdělávání j. n.
Statutární orgán (jednatelé):	Ing. Ondřej Polák, Jiří Toman.
Datum zápisu:	20. 7. 2003 [13]

2.2 Profil společnosti

Společnost Allrisk, s.r.o., je jednou z největších makléřských pojišťovacích formací působících na území České republiky. Založena byla v roce 2003. Krátce po svém vzniku rychle zaujala na českém trhu významné postavení. Poslední čtyři roky sídlí

v nové centrále v Brně, v němž zřídila ještě několik poboček. Regionální ředitelství je v Praze a síť poboček je rozseta po mnoha dalších městech České republiky, např. v Ostravě, Hodoníně, Blansku, Olomouci, Břeclavi, Jihlavě atd. [14]

Společnost má platnou smlouvu na krytí rizik souvisejících s profesí odpovědnosti ve výši 1 700 000 EUR. Po rozhodnutí České národní banky společnost splnila všechny nároky a tím se stala registrovaným pojišťovacím zprostředkovatelem. [14]

2.3 Organizační struktura

Ve vedení společností stojí dva jednatelé. V současnosti má přibližně 50 zaměstnanců pracujících na hlavní pracovní poměr. Příjmy pro ni tvoří i síť obchodníků, kteří jejím jménem podle svého zařazení do divize uzavírají pojistné smlouvy, úvěry, hypotéky apod. Aktuálně je jich téměř 800. Zařazeni jsou do týmů spadajících pod vedoucí jednotlivých kanceláří či jiných obchodních skupin.

Zaměstnanci, kteří nepatří do řad obchodníků, jsou rozdělení do několika oddělení a sídlí v centrále společnosti. Každé má svého ředitele a vedoucí pracovníky. Prioritu má „Obchodní“ oddělení, které má na starosti správu pojistných smluv. Pracuje v něm osm zaměstnanců. Významné je oddělení „Marketingu“, které se zabývá, jak již názvu vyplývá, marketingem a PR. O účetnictví se stará „Ekonomické“ oddělení s osmi členy. Předposledním je oddělení „Likvidace“, které se stará o likvidaci a řešení pojistných událostí klientů. Posledním, ale zdaleka ne významem, je oddělení „IT“, které se stará o chod vlastního informačního systému společnosti a vším s tím spojeným. V současné době v něm pracuje dvanáct zaměstnanců.

IT oddělení je rozděleno do několika vývojových týmů. První je skupina pracovníků, kteří mají na starosti vývoj webových aplikací, druhá se stará o starší informační systém společnosti postavený na základech MS Access, další částí je vedoucí analytický tým. Poslední je tým asistentek, které mají na starosti odesílání dat do pojišťoven apod.

2.4 Portfolio produktů

Společnost se zabývá poskytováním velkého množství produktů od různých pojišťoven a partnerů. Jednotlivé produkty jsou zařazovány do skupin spadajících do následujících tzv. divizí:

Divize pojištění – s tímto typem produktů společnost začínala. Jde především o pojištění automobilů, a to jak povinného ručení, tak havarijního pojištění. Jedná se však dále o penzijní připojišťování nebo pojištění majetku. Společnost se však zaměřuje i na životní a úrazové pojištění nebo pojištění průmyslu a podnikatelů. V poslední době se více rozvíjí speciální pojištění na cesty a rybářské pojištění. [15]

Divize pojištění právní ochrany – tato divize se zaměřuje na pojišťování právní ochrany motorových vozidel, domů, bytů a domácností. Poskytuje i telefonickou službu s právními informacemi. Nejnovějším produktem v tomto portfoliu je produkt s názvem „Rodinný právník“. [15]

Divize finančních služeb – jak už je z názvu zřejmé, zaměřuje se především na hypoteční úvěry, leasing, podnikatelské či spotřebitelské úvěry. Nejde však jen o úvěry, ale také o stavební spoření, účetní služby nebo investice. [15]

Divize realitních služeb – jedná se především o prodeje či pronájmy rodinných domů, bytů a rekreačních objektů. Nabízejí se však i pozemky, případně pronájmy komerčních a jiných prostor. [15]

Divize klientského servisu – nabízí podpůrné produkty pro ostatní divize většinou pro divizi pojištění, především pojištění automobilů. Tato divize se zabývá likvidacemi pojistných událostí a asistencemi při dopravní nehodě. Společnost poskytuje i zapůjčení volných automobilů, které jsou poskytnuty klientům po nehodě jako náhradní vozidlo. [15]

2.5 Současný stav IS společnosti

V následující sekci budou jednotlivé části popsány z hlediska jejich technologie a z hlediska jejich využívání pro chod společnosti.

2.5.1 IS v MS Access

První verze IS na platformě MS Access byla vytvořena již na konci roku 2003. Systém, se postupem času vyvíjel až do současné podoby, kdy se jedná o několik modulů, které se specializují na určité úkony či na samostatné produkty. Vývoj systému a příslušných modulů stále pokračuje. V následující části budou jednotlivé moduly stručně popsány.

Modul „Auto-půjčovna“ – je to jeden z novějších modulů a byl vytvořen pro zaměstnance z divize klientského servisu. Především se v něm řeší zadávání a editace výpůjček jednotlivých aut. Modul však poskytuje i informace o právě probíhajících likvidacích pojistných událostí apod.

Modul „Import“ – ani tu se nejedná o jeden z původních modulů. Vyvinut byl později pro potřeby asistentek v centrále společnosti. Dodnes se využívá jako nástroj pro importování různých dat přicházejících z pojišťoven apod. S jeho pomocí se stahují i bankovní platby.

Modul „Pokladna“ – je využívaný pro potřeby asistentek pracujících na přepážkách a pro zaměstnance, kteří osobně přijímají platby od klientů. Zabudovány jsou v něm funkce na různé úpravy dokladů a možnost jejich editace apod.

Modul „Pošta“ – nejvíce je využíván na obchodním oddělení. Vytvořen byl pro potřeby správy dokumentů, které se ve společnosti pohybují, aby bylo možno dohledat, kde se který dokument nachází. Modul rovněž eviduje dokumenty, které přicházejí od pojišťoven, a také ty, které společnost posílá svým klientům a obchodním partnerům.

Modul „Statistiky“ – tzv. pracovní stůl jednotlivých obchodníků. Je to jeden z nejdůležitějších modulů společnosti z hlediska získávání informaci z dat. V něm obchodník vidí své sjednané smlouvy a body, které za ně získal. Dále v něm zjišťuje resty u smluv, které musí vyřešit. Využíván je též pro upomínání neplatících klientů.

Modul „SQL“ – vznikl jako druhý v pořadí. Je to jeden ze dvou modulů, ve kterém probíhá samotné sjednávání smluv. Vyvinut byl pro sjednávání retailových smluv. Probíhají v něm i všechny potřebné operace jako stornování smluv, editace údajů o klientech nebo filtrování jednotlivých smluv. V tomto modulu se vytvářejí např. životní pojištění, různé typy úvěrů a hypoték.

Modul „Flotila“ – jedná se o první aplikaci, která vůbec byla pro společnost vyvinuta hned po jejím založení, když začínala s pojišťováním automobilů. V poněkud pozměněné formě slouží tomuto účelu doposud. Jeho název byl odvozen ze způsobu pojišťování vozidel tzv. flotilovou metodou. Specializovaný je jen pro sjednávání pojištění, které je spojeno s automobily.

2.5.2 Webová část

Postupem času se společnost snaží svůj informační systém transformovat do webového prostředí. I webové aplikace jsou rozděleny do několika samostatných prostředí a různých technologií.

„Portál“ – vůbec první webová aplikace vytvořená pro společnost. Postaven je na technologii .NET. Vývoj této aplikace probíhá již několik let. Před tím, než se do něho zapojili přímo zaměstnanci společnosti Allrisk, podílely se na něm i externí firmy. Nejdůležitějším úkolem portálu je vytváření nabídek pojištění a vytváření nových smluv. Další funkcí, která není nikde jinde v systému možná, je sjednání (a storno) cestovního pojištění.

„Sjednavač Effective“ – webová aplikace, která byla vytvořena před dvěma lety pro sjednávání nového produktu s názvem Effective. Celá aplikace je vystavěná na frameworku Django, který je napsán v jazyce Python. Slouží pro sjednávání produktu, k potvrzování podpisu smlouvy a ke generování potřebných dokumentů.

2.6 SWOT analýza společnosti

Analýza je zaměřena na společnost jako celek.

Silné stránky:

- Dobrá likvidita společnosti.
- Konkurenční výhody.
- Exkluzivita u některých produktů.
- Dobré postavení na trhu.
- Individuální přístup ke klientům.

Slabé stánky:

- Vysoká mzdová náročnost některých oddělení.
- Koncentrované řízení společnosti.

Příležitosti:

- Nárůst propojištěnosti stávajících klientů.
- Využití problémů konkurence (přechod z otevřených flotil).
- Získávání klíčových zaměstnanců konkurence.

Hrozby:

- Ztráta obchodních partnerů.
- Odchod klíčových zákazníků ke konkurenci.
- Odchod klíčových zaměstnanců.

2.7 SWOT analýza IS společnosti

Další z analýz je již více zaměřena na obor, kterým se autor ve společnosti zabývá. Vybrána byla nejnovější část informačního systému společnosti a prověřena SWOT analýzou.

Silné stránky:

- Jednoduchost zadávání nových smluv.
- Použití na více platformách.
- Použití odkudkoliv.

Slabé stánky:

- Omezená funkcionalita.
- Nemožnost dostatečné editace údajů.
- Nezachytávání změn po sjednání.

Příležitosti:

- Získávání nových zákazníků.
- Příjmy pro společnost z jiného produktu.
- Získání zkušeností v novém odvětví.

Hrozby:

- Odliv zákazníků kvůli nemožnosti upravovat údaje.
- Nemožnost zjistit cizí zásah do dat.

2.8 Analýza Python vs. MS Access

V následující části jsou sepsány jednotlivé argumenty pro a proti u dvou technologií, mezi kterými se rozhodovalo, která z nich bude vybrána pro další vývoj.

2.8.1 Analýza vývoje v Pythonu

Jedná se o objektově orientovaný jazyk používaný pro webové aplikace. Ve společnosti se používá Framework Django.

Výhody – webová aplikace je použitelná odkudkoliv, kde je připojení k internetu. Použít se může na více platformách, jako jsou mobilní zařízení apod. Jedná se o opensource technologii, z čehož vyplývá, že se nemusejí platit žádné licenční poplatky.

Nevýhody – vývoj aplikace pomocí technologie je časově náročnější. Programátorů ovládajících tuto technologii není v současnosti mnoho.

2.8.2 Analýza vývoje v MS Access

Jedná se část balíku MS Office od společnosti Microsoft.

Výhody – vývoj v prostředí MS Access je rychlý, snadný a programátorsky přívětivý. Většina úkonů se vytváří přes návrhové zobrazení pomocí ovládacích prvků vývojové aplikace.

Nevýhody – nutnost placení licenčních poplatků, nezbytnost připojení na terminály, přes které se připojuje k samotné aplikaci.

2.9 Hodnocení SWOT analýz

V následující části budou postupně zhodnoceny provedené jednotlivé SWOT analýzy a z nich vyvozeny důsledky pro zpracování této bakalářské práce.

2.9.1 Zhodnocení SWOT analýzy společnosti

Jak je dobře viditelné z první analýzy, má společnost ve svých silných stránkách velké množství klíčových vlastností. Z analýzy vyplynulo, že se zejména zaměřuje na stávající klienty. Snaží se jim nabídnout co největší možné portfolio produktů. Nesnaží se bojovat cenou, ale ve srovnání s konkurencí přichází s něčím novým, exkluzivním.

Mezi slabší stránky společnosti patří to, že nemá vybudovanou větší sít řídících pracovníků. Většinu důležitých a klíčových rozhodnutí stále provádějí jednatelé. Z toho vyplývá, že by měli méně pracovat pro svoji společnost, ale o to více pracovat na své společnosti, aby byla ve výsledku více samostatná a fungovat i bez jejich přímého dohledu.

2.9.2 Zhodnocení SWOT analýzy IS společnosti

Tato analýza se zabývala nejnovější částí informačního systému zaměřenou na sjednávání nového produktu. Je zřejmé, že systém byl vytvořen ve své první fázi jen pro sjednávání a označování podepsaných smluv. Z analýzy vyplynulo, že tuto část současný systém bezpochyby splňuje.

V další části bylo zjištěno, že potřebné úpravy ve smlouvách, údajích o klientech a jiných potřebných změnách, budou muset být v nejbližší době doprogramovány.

2.10 Finální rozhodnutí

Pomocí SWOT analýzy a IS společnosti spolu s autorovou analýzou výhod a nevýhod jednotlivých možností technologií bylo rozhodnuto, že pro vývoj administračního rozhraní pro různé editace bude použita technologie založená na aplikaci MS Access. Rozhodujícími argumenty byla rychlost a cena vývoje. Licenční poplatky se musí stále platit kvůli ostatním modulům.

3 NÁVRH ŘEŠENÍ

V následující kapitole budou popsána jednotlivá praktická řešení, která budou autorem tvořena na základě provedených analýz. Záměrem je vytvořit co nejvíce použitelnou aplikaci podle specifikací zadavatele. První část se zaměří na datovou strukturu, ve které budou uložena všechna potřebná data. Po vzniku samotné databáze bude možno pustit se do vlastního tvoření uživatelské aplikace.

3.1 Návrh databáze

Jelikož se jedná o programování modulu pro nový produkt, filozofií společnosti je, že všechny nové produkty se již nebudou ukládat do staré struktury, která je ve společnosti používaná od počátku. Slouží zatím pro ukládání dat ke starším produktům a postupně se bude přesouvat do nové datové struktury.

3.1.1 Použitý databázový systém

Společnost se již dříve rozhodla, že jako databázový systém bude používat produkt *PostgreSQL*. Jedná se o objektově-relační databázový systém.

Jeho velkou výhodou je, že se vydává pod licencí typu MIT. Z toho vyplývá, že se jedná o open source a free software. Znamená to, že produkt není vlastněný žádnou firmou. Jeho vývoj probíhá za pomoci komunity a vývojářů společností, které ho používají.

ODBC – pro přístup k tabulkám uloženým na databázovém serveru se využívá technologie ODBC. Pro správné používání musí být na terminálu nastaven ODBC ovladač. Vlastní připojování k tabulkám je řešeno funkcionalitou vytvořenou pomocí jazyka VBA přímo v aplikaci. Jedná se především o nastavení tzv. *Connection stringu* a spuštění procedury pro každou tabulku.

3.1.2 Hrubá struktura (použitá schémata)

Dříve byly všechny tabulky vytvořeny v jednom schématu. To však bylo dost nepřehledné. Nová struktura se proto chce těchto problémů vyvarovat. Vytvoření schémat nejenže napomůže lepší orientaci, ale správa oprávnění k jednotlivým schématům a tabulkám bude mnohem přehlednější a efektivnější.

V následující části budou popsána jednotlivá schémata, která byla pro správné fungování budoucí aplikace potřebná.

Schéma Insurace – jedná se o jedno z nejdůležitějších schémat, ve kterém jsou ukládány informace pro evidenci pojištění klientů.

Schéma Partner – jedno z klíčových schémat. Ukládají se v něm informace o jednotlivých obchodních partnerech společnosti, o jejích produktech a sazbách.

Schéma Personal – schéma je primárně určeno pro evidenci informací o klientech. Ukládají se v něm však i informace o adresách a podobných doplňkových informacích.

Schéma Public – nejstarší schéma v celé databázové struktuře, které bylo dříve určeno pro ukládání všech informací. U autorovy aplikace se toto schéma použije jen v minimálním množství.

Schéma Register – jedná se o tzv. číselníkové schéma. V něm jsou uloženy obecné číselníky používané napříč celou databázovou strukturou. Konkrétnější číselníky jsou uloženy v jednotlivých vlastních schématech, ke kterým náleží.

Schéma Sync – jak již bylo několikrát uvedeno, v současné době se ve společnosti používají dvě databázové struktury. V některých případech není jiná možnost, jak provést mezi strukturami synchronizaci, než vytvořit tabulky, které napomohou ukládání potřebných informací z databáze a jejich vybírání z ní.

Schéma Sys – toto schéma je námi uměle vytvořené pro tzv. systémové tabulky. Jedná se o tabulky, které jsou přístupné napříč systémem. Využívají se v něm na více místech.

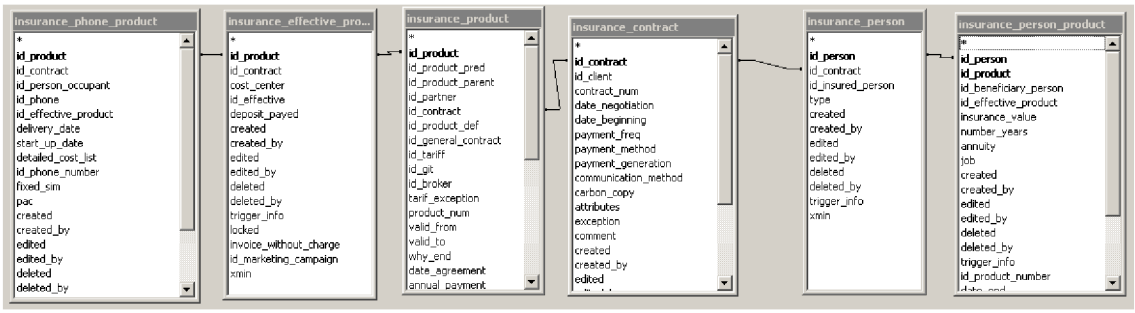
Schéma Tariff – obsahuje informace o jednotlivých sazebnících u produktů. Schéma je speciální tím, že obsahuje i větší množství vlastních číselníků, které nejsou uloženy ve schématu *Register*, ale právě ve schématu *Tariff*.

Schéma Doc – název byl odvozen z účelu, pro který bylo vytvořeno. Jedná se o schéma, které eviduje informace o dokumentech v systému proudících. V našem případě se bude jednat hlavně o smlouvy a o rozúčtování za poskytované služby.

Tabulky mimo schéma – jedná se o speciální typy tabulek, které nejsou uloženy na databázovém stroji. Jelikož bude aplikace vytvořena v prostředí MS Access, jedná se o lokálně uložené tabulky vytvořené přímo ve vývojovém prostředí. Tabulky jsou používány pro pomocné výpočty apod.

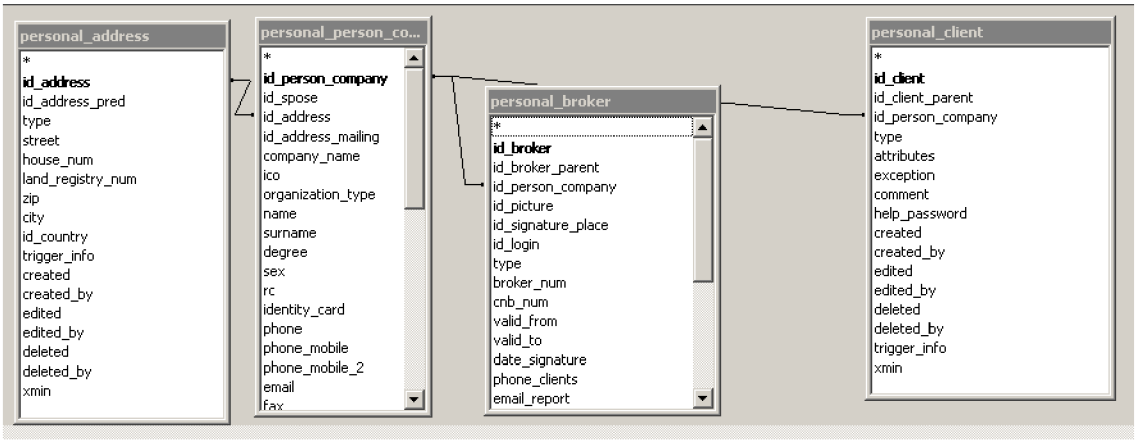
3.1.3 Ukázky ERD diagramů (vybraná schémata)

Následující ERD diagram patří schématu instance.



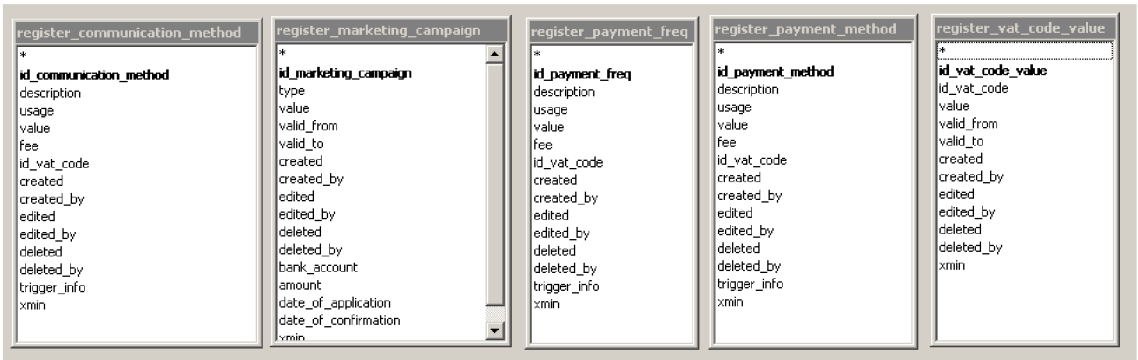
Obrázek 2: ERD diagram schématu Insurance

Obrázek, který následuje, je ERD diagram, který náleží schématu Personal.



Obrázek 3: ERD diagram schématu Personal

Pro ukázku bylo vybráno i schéma Register, ve kterém jsou viditelné jednotlivé obecné číselníky. Rozdíl je v tom, že mezi tabulkami ve schématu nejsou žádné relační vztahy.



Obrázek 4: ERD diagram schématu Register

3.2 Vytvoření uživatelského rozhraní

Když už je vytvořena datová struktura, je dalším krokem samotné vytvoření aplikace. Společnost již má několik administračních modulů pro jiná využití, a proto vzhled aplikace, kterou vytváří autor, se bude ostatním podobat. Jeho vizí je, že vzhled aplikace včetně všech ovládacích prvků by měl být co nejjednodušší a co nejpřehlednější.

3.2.1 Úvodní okno

Úvodní okno je důležité v tom, že se jako první spouští hned po otevření aplikace. Jeho úkolem je vytvářet rozcestník. Na následujícím obrázku je vytvořené úvodní okno celé aplikace.

Obrázek 5: Úvodní okno aplikace

Zdroj: (Vlastní)

Název aplikace – v horní části okna je umístěn název celého modulu. Jako název bylo vybráno pojmenování *„TEA“*. Jedná se o spojení slov telekomunikační administrátor. Je to i pojmenování pozice zaměstnance, který bude aplikaci používat.

Logo – na všech modulech používaných ve společnosti se používá logo Allrisk, které je zaregistrované a chráněné.

Tlačítko *„Přidat smlouvu ARF“* – tlačítko poslouží pro otevření vlastního formuláře, na kterém bude možno sjednat novou smlouvu produktu, pro který je aplikace vytvořena.

Tlačítko „Nastavení“ – po kliknutí na toto tlačítko se dostane uživatel do sekce, ve které bude moci provádět operace, které přímo nesouvisejí se sjednáváním či filtrováním informací. Bude se jednat např. o funkce na odesílání sms nebo e-mailů klientům, zobrazování marketingových akcí apod.

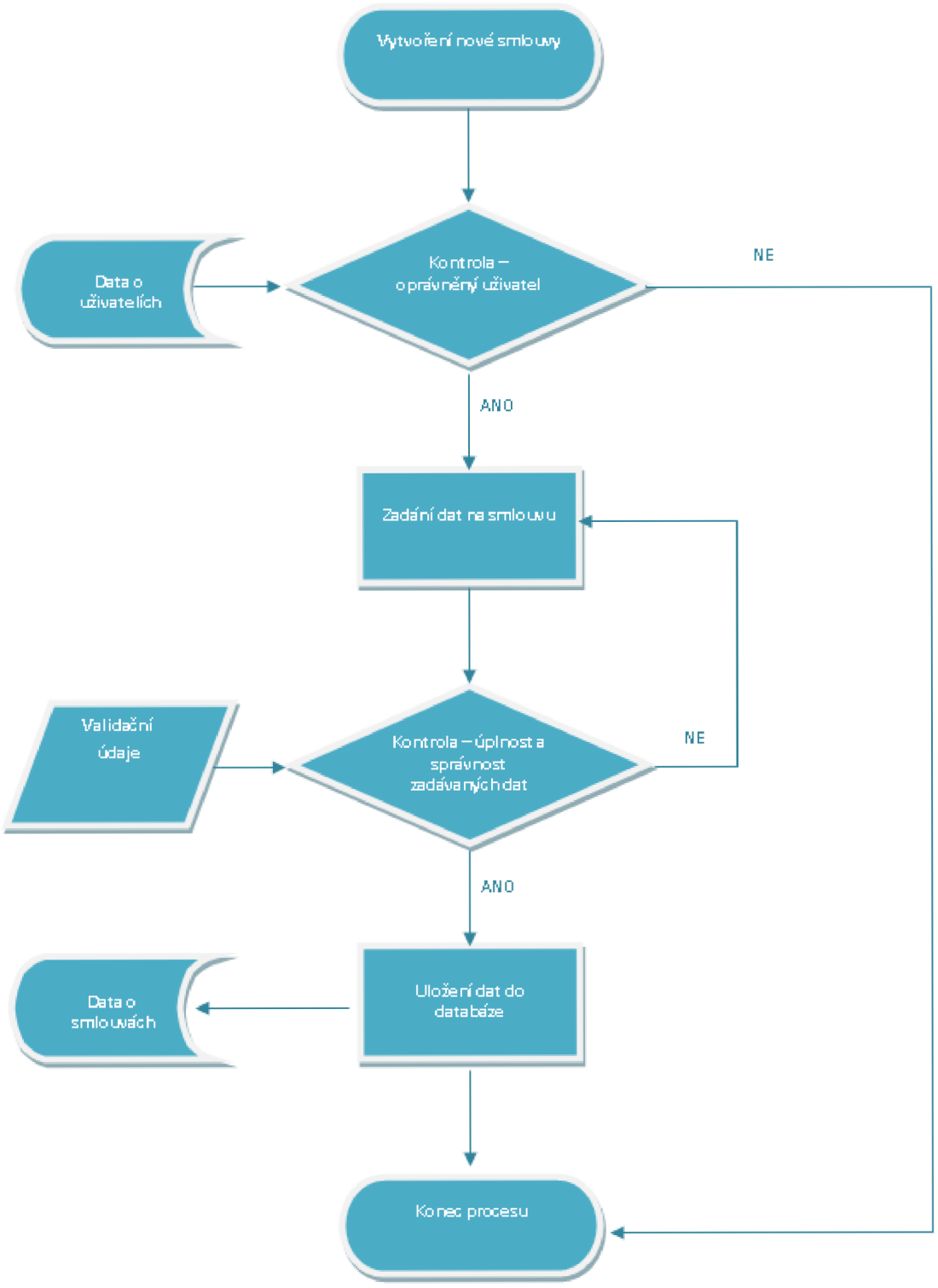
Tlačítko „Filtr smluv“ – tlačítko, které bude nejvíce používané. Po kliknutí na ně se zobrazí formulář, na kterém budou jednotlivé ovládací prvky, pomocí nichž se budou moci sjednané smlouvy filtrovat podle předem stanovených parametrů.

Tlačítko „I“ – po kliknutí se zobrazí jednoduchý formulář s nejdůležitějšími informacemi o technické podpoře a kontaktu na helpdesk.

Tlačítko „Konec“ – po stisku tohoto tlačítka se celá aplikace zavře.

„Červené číslo“ – jedná se o číselnou hodnotu, která vypovídá o vývojové verzi administračního modulu.

3.2.2 Vývojový diagram – nová ARF smlouva

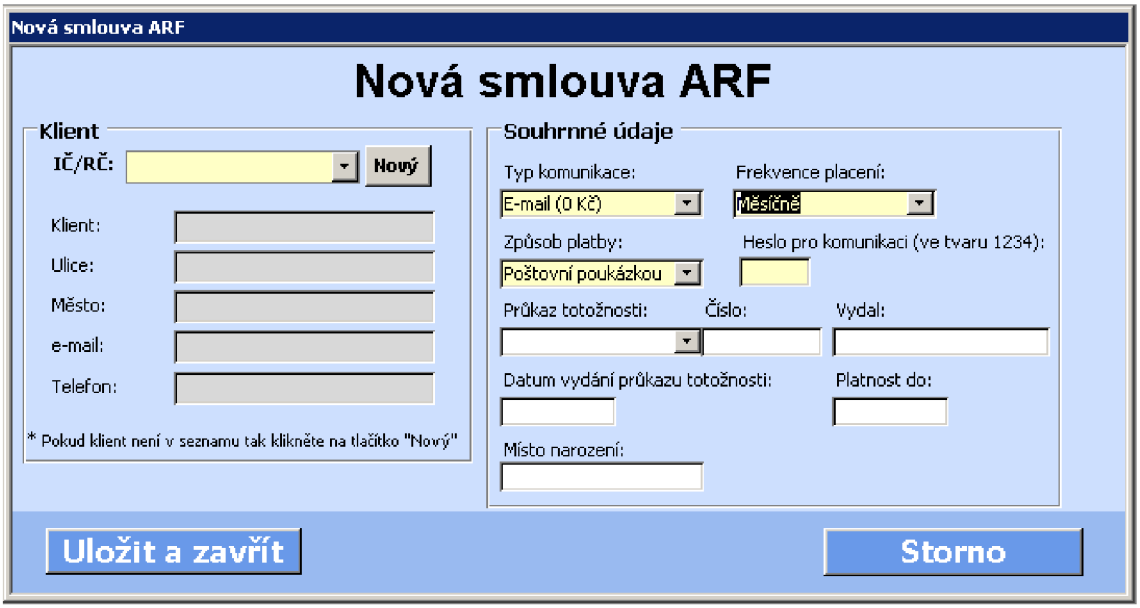


Obrázek 6: Diagram – zadání nové ARF smlouvy

Zdroj: (Vlastní)

3.2.3 Formulář pro vytvoření ARF smlouvy

Formulář, který nemá za účel filtrovat či editovat již zadaná data. Jeho účelem je zadávání nových smluv do systému.



Obrázek 7: Formulář pro vytvoření ARF smlouvy

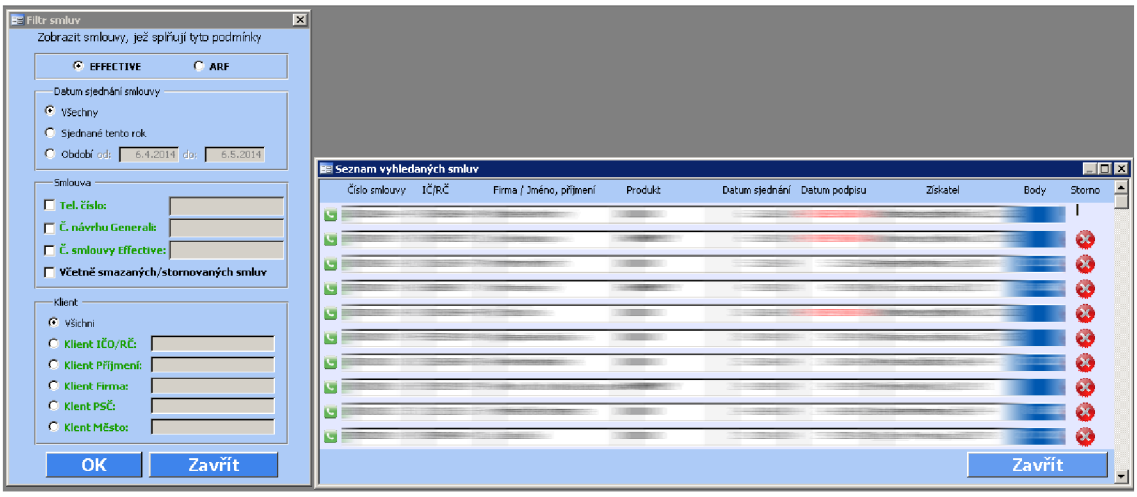
Zdroj: (Vlastní)

Formulář je logicky rozdělen na dvě části. V první části, která je na levé straně formuláře, jsou informace o klientovi. Pokud klient není v databázi, může se založit nový záznam po kliknutí na tlačítko „Nový“, které je vedle políčka IČ/RČ. Na pravé straně formuláře jsou ostatní informace, které jsou potřebné pro vytvoření nové smlouvy.

Důležitou informací je, že všechna pole, která jsou použita v systémech společnosti a mají žlutou barvu, jsou povinná a musí být vyplněna.

3.2.4 Formuláře k filtrování smluv

K tomu, aby bylo možno pracovat s jednotlivými smlouvami a datovými informacemi, je potřeba nástroj pro jejich filtrování. Proto byl vytvořen formulář pro zadávání jednotlivých omezujících parametrů. Na něj navazuje další, na kterém se zobrazují vyfiltrované smlouvy. Oba tyto formuláře jsou zobrazeny na následujícím obrázku.



Obrázek 8: Formuláře k filtrování smluv

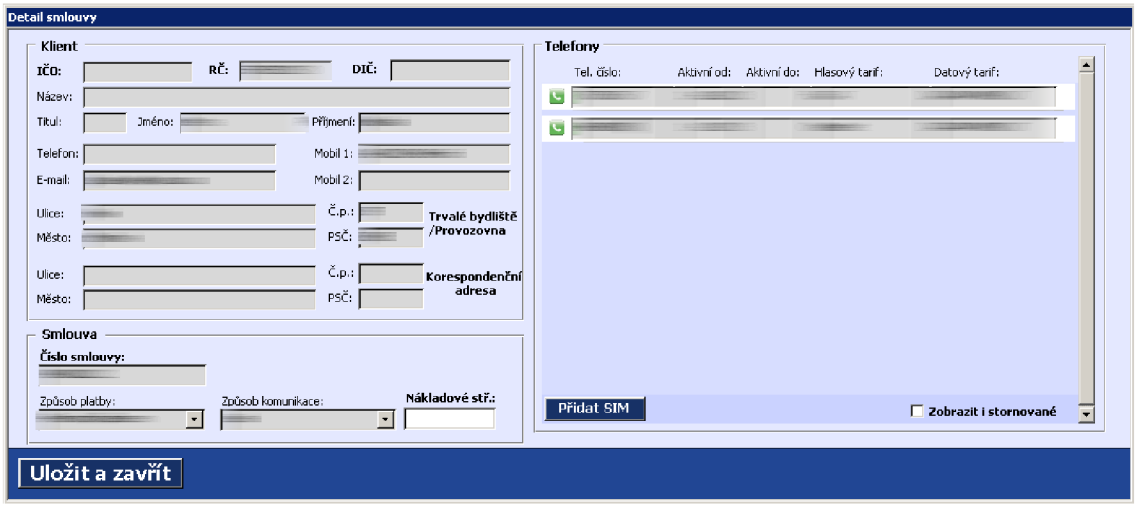
Zdroj: (Vlastní)

3.2.5 Formuláře pro zobrazování detailů

Detail smlouvy – jde o formuláře, ve kterých se již zobrazují podrobné informace k jednotlivým smlouvám. Jedná se především o formuláře pro detail smluv, ve kterých jsou jednotlivé informace uspořádány do logických skupin.

Pokud se jedná o smlouvu sjednanou na webu, je formulář rozdělen do čtyř částí. Obsahuje informace o klientovi, o samotné smlouvě, o telefonních číslech a poslední část je zaměřena na smlouvu k životnímu pojištění.

U smlouvy, která je vytvořená pomocí této aplikace, tak chybí část o životním pojištění. Na detail smlouvy se uživatel dostane pomocí zeleného tlačítka, které je u každého záznamu na vyfiltrovaných smlouvách. Detail smlouvy sjednané v aplikaci vypadá následovně.



Obrázek 9: Detail smlouvy

Zdroj: (Vlastní)

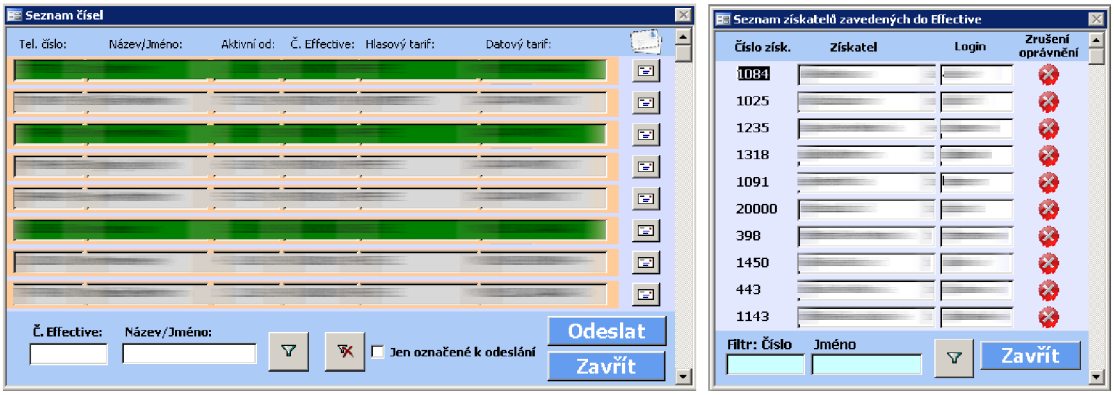
Detail telefonu – druhou skupinou informací, které je třeba zobrazovat, jsou informace k telefonním číslům. Na obrázku je vidět, že v horní části jsou umístěny informace o tom, kdy byl produkt přidán apod. Ve druhé části jsou informace o hlasových a datových tarifech ke konkrétnímu číslu. Na tomto formuláři je přidána i funkcionalita na změnu hlasového, popřípadě datového tarifu. Nejdůležitější funkcionalitou, kterou lze na tomto formuláři spustit, je stornování telefonního čísla.

Obrázek 10: Detail telefonu

Zdroj: (Vlastní)

3.2.6 Formuláře pro obsluhu

Telekomunikační administrátor bude muset při své práci v některých případech kontaktovat klienta. Pro tento případ byl vytvořen formulář, ve kterém jsou zapsána všechna telefonní čísla přidaná ke smlouvě. Administrátor si může označit, na která čísla se má SMS odeslat. Pro komfortnější práci byly naprogramovány i filtry, s jejichž pomocí je možné vyhledávat konkrétní čísla. Stejně tak funguje i odesílání e-mailů.



Obrázek 11: Formulář pro odesílání SMS a e-mailů a seznam uživatelů

Zdroj: (Vlastní)

Druhý formulář, vybraný na ukázku, zobrazuje seznam všech uživatelů, kteří mají nastavena oprávnění k tomu, aby mohli produkt sjednávat. Administrátor i v tomto případě může jednotlivé obchodníky filtrovat. Pokud již obchodník nemá mít oprávnění pro sjednávání produktu, může mu je administrátor odebrat.

3.3 Funkcionalita v aplikaci

V této části se bakalářská práce podrobněji zaměřila na jednotlivé funkcionality, které má telekomunikační administrátor pro svoji práci k dispozici. Funkce jsou umístěny na dvou místech. Prvním typem jsou funkce a procedury vytvořené a uložené přímo v aplikaci. Dalším místem, kde jsou funkce uloženy, je databáze. Při operacích se smlouvami a telefonními čísly se používají uložené procedury vytvořené v prostředí databázového systému PostgreSQL.

3.3.1 Operace se smlouvou

Pokud se jedná o smlouvy, dají se v systému se smlouvou provádět tři základní operace:

Přidání nové ARF smlouvy – jedna z mála větších operací, která ještě není řešena formou uložené databázové procedury. Přidání smlouvy je zjednodušeně řečeno posloupnost několika SQL dotazů, které se postupně provádějí. Ve vlastní funkci je několik ověření vstupních údajů a ověření toho, zda dotazy proběhly v pořádku.

Smazání smlouvy – jedná se o ukončení smlouvy ve chvíli, kdy ještě nezačala platit, nebo nezačaly platit žádné produkty navázané na tuto smlouvu. Znamená to, že nebyl ještě potvrzen podpis na smlouvě a nebyly vygenerovány žádné povinné dokumenty. V systému je nastaveno, že tento úkon se může provádět maximálně dva týdny ode dne sjednání smlouvy. V praxi je tato operace prováděna pomocí databázové procedury, která má tři vstupní parametry – identifikátor smlouvy, identifikátor uživatele, který smazání provádí, a důvod smazání smlouvy.

Ukončení (storno) smlouvy – jedná se o situaci, kdy již není možno smlouvu jednoduše smazat. Jde o případ, kdy už je potvrzen podpis na smlouvě. V zásadě jde o dva případy, kdy je potřeba smlouvu stornovat různými způsoby:

- **Storno k počátku** – tato situace nastane, když je podpis již potvrzen, ale produkty ještě nezačaly platit. V této situaci lze stornovat smlouvy a její produkty k počátku.
- **Storno k datu** – jedná se o situaci, kdy se klient rozhodne ukončit smlouvu a některý z produktů na smlouvě již nějakou dobu platí a všechny produkty se stornují k jednomu datu.

Stornování se provádí uloženými databázovými procedurami, které jsou, co se týče vstupních parametrů, podobné těm na smazání. Přibývá zde však další parametr – datum storna.

Změna tarifu na smlouvě – jedná se o nejnáročnější a nejdůležitější funkcionalitu. Klient o ni může požádat a telekomunikační administrátor tento úkon může provést. Změna se provádí na detailu smlouvy a všechny potřebné vstupní parametry zadává administrátor pomocí připraveného formuláře. Po potvrzení vstupních parametrů se z aplikace pomocí jazyka VBA zavolá databázová procedura, která potřebnou změnu zařídí.

3.3.2 Operace s telefonním číslem

I s telefonními čísly je potřeba provádět operace. Z toho důvodu byla v jazyce VBA vybudována funkcionalita pro následující operace:

Storno telefonního čísla – tato funkcionalita je potřebná v situaci, kdy se za telefonní číslo platí. Může se použít tehdy, kdy si klient přeje stornovat jen číslo, ale samotnou smlouvu chce ponechat platnou. Vstupními parametry databázové procedury jsou identifikátor telefonního čísla, datum, ke kterému se má číslo stornovat, důvod storna a login uživatele provádějícího ukončení.

Přidání nového čísla – klient může mít na smlouvě podle tarifu různý počet nevázaných čísel. Systém hlídá, aby se nestalo, že by na smlouvě byl větší počet čísel, než povoluje definice tarifu smlouvy. Samotné přidání čísla provede uložená databázová procedura. Na formuláři pro přidání nového čísla je výběr ze tří možností přidělení čísla. Administrátor může zadat telefonní číslo, které vybere ze seznamu vygenerovaných čísel od poskytovatele. Druhou možností je nechat si přidělit číslo automaticky a funkcionalita napsána přímo v prostředí aplikace klientovi přidělí číslo automaticky. Poslední možností je ruční vyplnění čísla. Jedná se o situaci, kdy klient převádí číslo od jiného operátora.

Převedení čísla na jinou smlouvu – jedná se o situaci, kdy je potřeba z nějakého důvodu existující číslo převést z jedné aktivní smlouvy na druhou. Nesmí dojít k tomu, že by číslo bylo nějakou dobu neaktivní. Funkce provede převod tak, že na původní smlouvě se telefonní číslo ukončí k požadovanému datu a na nové smlouvě se založí nové potřebné záznamy. Na smlouvě vždy musí zůstat jedno vázané číslo.

Změna vázaného čísla za nevázané – tato operace se provádí kombinací dvou výše popsaných funkcionalit. Nejdříve se číslo k požadovanému datu ukončí a znovu se založí na stejné smlouvě, ale s jiným parametrem vázanosti. Nemůže však dojít k situaci, kdy by na smlouvě nebylo žádné vázané číslo. Pro tento převod jsou vyvinuty funkcionality, které mají dvě možnosti použití. Buď se stornuje původní vázané číslo a nahradí zcela novým, nebo se vázaným číslem stane jiné nevázané číslo na smlouvě již platné.

Změna tarifů na čísle – každé telefonní číslo má definovanou kombinaci hlasového a datového tarifu. Kombinace jsou předem definovány a hlídány již na úrovni databáze.

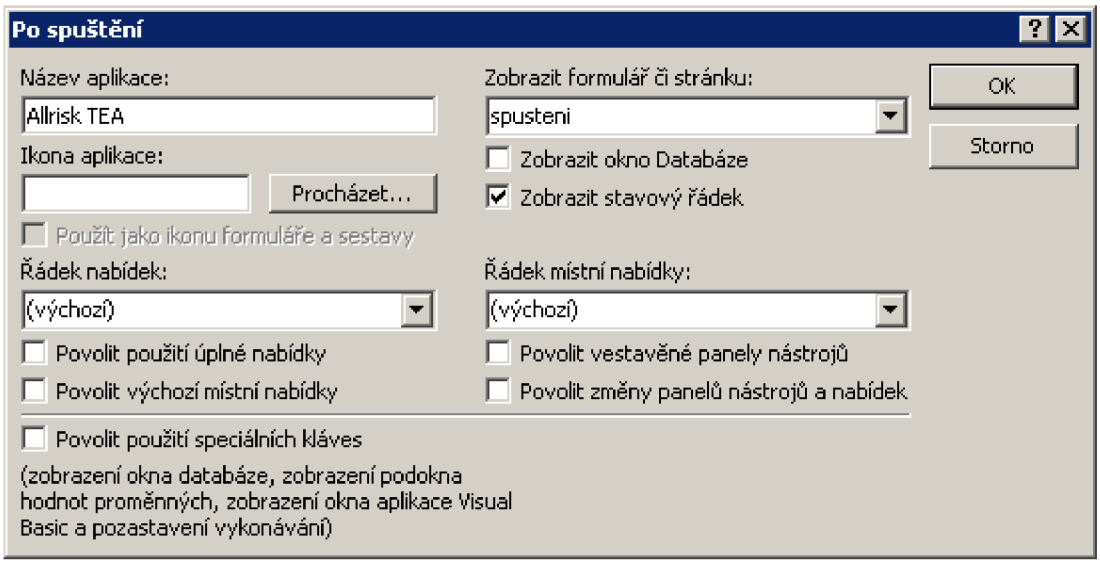
Při změně tarifu se tato kontrola provádí znovu. Technicky se změna opět realizuje databázovou procedurou.

Všechny předchozí operace jsou povoleny jen u čísel, která jsou k datu změny aktivní a platná. Hlídání tohoto stavu je uskutečňováno na úrovni jazyka VBA a tato situace je kontrolována před každou výše popsanou operací.

3.4 Implementace aplikace

Všechny aplikace běžící v prostředí MS Access jsou ve společnosti spouštěny z tzv. terminálů. Jedná se o virtuální pracovní stanice, ke kterým se uživatelé připojují přes vzdálenou plochu. Jde o jeden z nejdůležitějších zabezpečovacích prvků.

Vytvoření verze – aplikace, která je vytvořená v prostředí MS Access, nabízí možnost automatického vytvoření uzamčené verze. V ní jsou omezeny různé ovládací prvky. Uživatel může využívat jen ovládací prvky, které vývojář povolil, nebo byly v programu nově vytvořeny. Původní soubor má koncovku *.mdb a zamčení aplikace způsobí vytvoření nové spustitelné verze, která má novou koncovku *.mde. Při vytváření zamčené aplikace je možné nastavit několik vstupních parametrů, které jsou viditelné na dalším obrázku.



Obrázek 12: Nastavení oprávnění po spuštění

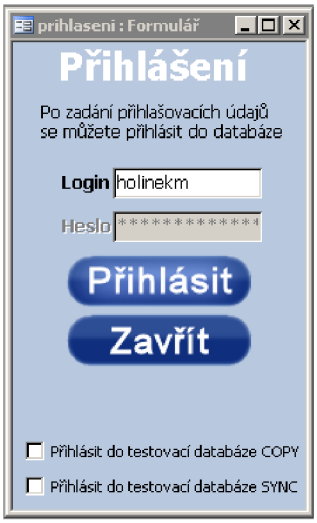
Zdroj: (Vlastní)

Uživateli se povolí jen zobrazení stavového řádku, které způsobí, že uvidí informace pouze při provádění některých definovaných úkonů.

Vydání nové verze – vývoj aplikace stále pokračuje, neboť je v některých případech potřebné vydat uživateli novou verzi programu. Vydání probíhá automaticky. Uživatel ho pozná po spuštění aplikace podle změny čísla aplikace. Vydávání verze je prováděno v několika krocích. Prvním je nakopírování uzamčené verze na místo k tomu určené. Druhým je spuštění skriptu, které zařídí vlastní rozkopírování předem definovaným uživatelům, kteří mají mít ke své práci aplikaci k dispozici.

3.4.1 Přihlášení do aplikace

Aplikace je povolena jen několika vybraným zaměstnancům. Z tohoto důvodu bylo potřeba vytvořit zabezpečení uživatelským jménem, které uživatel spolu s heslem zapíše do přihlašovacího okna, které vypadá takto:



Obrázek 13: Přihlašovací okno

Zdroj: (Vlastní)

Prvním krokem je, že se ověří uživatelské jméno a heslo a uživatelovo oprávnění spustit aplikaci. Druhým krokem je, že se odmapují připojené tabulky. V posledním kroku se všechny potřebné tabulky namapují zpět. Děje se tak z důvodu bezpečnosti, protože seznam a forma tabulek se mohou změnit. Pokud všechny předem popsané operace proběhnou v pořádku, aplikace se spustí a je připravena k používání.

3.5 Ekonomické zhodnocení

V následující části budou vyčísleny jednotlivé náklady, které vznikly při tvorbě aplikace. Záměrně nebyly započítány licenční poplatky, které se musí platit za ostatní

moduly používané ve společnosti, ale jen náklady, které souvisely s vývojem samotné aplikace.

3.5.1 Náklady na vývoj a provoz

V následující tabulce jsou uvedeny jednotlivé části vývoje aplikace spolu s jejich časovou náročností.

Druh práce	Časová náročnost v týdnech
Analýza problémů	1
Grafický návrh	2
Naprogramování	4
Testování	1
Implementace	1
Celkem	**9**

Tabulka 1: Náklady na vývoj

Zdroj: (Vlastní)

Časovou náročnost projektu bylo nutné převést na relevantní měnu. Pracovní hodina byla proto ohodnocena na 130 Kč. Cena vývoje aplikace tak činila 46 800 Kč.

Náklady na provoz – o provoz aplikace se bude příležitostně starat některý z programátorů. Jedná se přibližně o jednu hodinu týdně na potřebné úpravy.

S aplikací bude pracovat primárně jeden uživatel, který se v případě nepřítomnosti nahradí jiným. Hodinová taxa tohoto zaměstnance byla vypočítána na 90 Kč/hodinu.

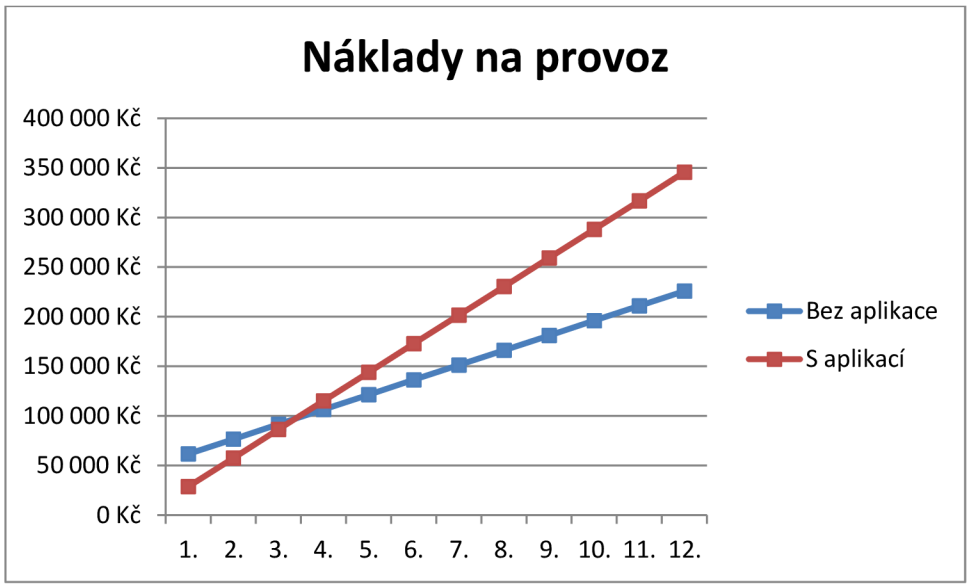
Z těchto hodnot vyplývá, že náklady uživatele na provoz aplikace budou měsíčně činit 14 400 Kč a na úpravy programátorem 520 Kč. V součtu se jedná o 14 920 Kč.

3.5.2 Náklady na provoz bez aplikace

Pokud by nebyla vytvořena aplikace, museli by se o agendu spojenou s řízením všech operací, jež probíhají s novým produktem, starat dva zaměstnanci. Z toho vyplývá, že by se náklady na operace prováděné s tímto produktem vyšplhaly na 28 800 Kč.

3.5.3 Vyhodnocení variant

Z následujícího grafu jsou zřejmé předchozí náklady pro jednotlivé varianty. Hodnoty na ose x jsou po sobě jdoucí měsíce.



Obrázek 14: Náklady za jednotlivé varianty

Zdroj: (Vlastní)

Z grafu je patrné, že varianta vytvoření aplikace, je jednoznačně pro společnost přínosnější a z dlouhodobého hlediska i méně nákladná. Již mezi třetím a čtvrtým měsícem od nasazení aplikace budou náklady nižší než u varianty, u které by aplikace nevznikala.

3.6 Přínosy práce

V této části jsou v bodech shrnuty jednotlivé přínosy, které autorova práce společnosti přinesla. Některé body byly již výše podrobněji popsány.

Hlavní přínosy

- Výrazné snížení nákladů na administrování produktu.
- Snížení mzdových nákladů (nemusí se přijímat nový zaměstnanec).
- Nižší náklady povedou ke zvýšení zisku.

Vedlejší přínosy

- Evidování smluv.
- Evidování změn na smlouvách.
- Propojení s dosavadním systémem.
- Evidování dokumentace.
- Úspora papírové evidence.
- Efektivní využití lidské práce.

Jak je z jednotlivých bodů zřejmé, vedla autorova řešení k jednoznačným finančním úsporám a také k efektivnějšímu evidování jednotlivých operací, prováděných v životním cyklu produktu.

Velký přínos je v tom, že systém bude sám evidovat jednotlivé změny prováděné na smlouvách. Nebude tedy docházet kvůli odlišným informacím k rozporům, jako tomu bylo u tištěných dokumentů.

ZÁVĚR

V první části bakalářské práce byla sepsána všechna teoretická východiska, potřebná v dalších fázích pro další vypracovávání stanovených úkolů. Autor se především zaměřil na popsání logiky využití jazyka VBA v reálné praxi.

Jakmile byla získána ucelená představa o všech teoretických východiscích potřebných pro začátek vlastní realizace, bylo třeba orientovat se na vlastní informační systém společnosti. Nejprve bylo nutné vyhledat o ní všechny potřebné informace: Čím se zabývá, které produkty a v jaké kvalitě poskytuje. Důležitá byla její organizační struktura, neboť bylo nutné mít představu, s kým se bude muset o jednotlivých částech projektu jednat, kdo je bude hodnotit a schvalovat. Poté byly provedeny všechny potřebné analýzy informačního systému společnosti. Ze zjištění vyplynulo, že je potřebné vytvořit aplikaci pro telekomunikačního administrátora. Pak byly provedeny analýzy možných technologií pro vývoj. Z nich vyšla nejlépe technologie s použitím MS Access a jeho jazyka VBA.

Teprve po zpracování potřebných analýz bylo možné pustit se do vlastní realizace aplikace. V první fázi bylo vytvořeno grafické rozhraní aplikace. Její finální vzhled vznikl po konzultacích s vedoucími pracovníky. Po zhotovení grafiky bylo nezbytné vytvořit potřebnou funkcionalitu pro budoucí práci v aplikaci. Nejdříve byla připravena funkcionalita na obsluhu jednotlivých formulářů a ostatních ovládacích prvků aplikace. Když byla funkční, bylo nutné pro práci s daty uloženými v databázi vytvořit databázové procedury.

Poslední fází této bakalářské práce bylo ekonomické zhodnocení celého projektu. Z provedených výpočtů je zcela zřejmé, že rozhodnutí vytvořit aplikace bylo správné a vede k úspoře nákladů.

Všechny úkoly, které byly vytyčeny na začátku práce, byly splněny. Aplikace se v současné době používá a je v ostrém provozu. Její vývoj však neskončil a v další fázi bude pokračovat.

SEZNAM POUŽITÝCH ZDROJŮ

[1] ANDRÁSSY, Roland. *Visual Basic for Applications: V prostředí MS Office* [online]. Strojnícka fakulta TU Košice. © 2003 [cit. 19.01.2014]. Dostupné z: http://www.sjf.tuke.sk/transferinovacii/pages/archiv/ transfer/6 - 2003/pdf/148 - 150.pdf

[2] CADFORUM.CZ. *Používání proměnných a operátorů* [online]. ©2012 [cit. 19.01.2014]. Dostupné z: http://www.cadforum.cz/cadforum/Vyvojove - prostredky - AutoCADu/Pasmo/Casti/VisualBasic/VBAZakladyProg/355vbazpmodulyaprocedury.htm#VBA_sub

[3] LASÁK, Pavel. *Proměnné ve VBA* [online]. ©2006-2014 [cit. 19.01.2013]. Dostupné z: http://office.lasakovi.com/excel/vba-teorie-zaklady/promene-excel-vba/

[4] HANÁK, Ján. *Visual Basic .NET 2003 - začínáme programovat*. Praha: Grada Publishing, a.s., 2004. 180 s. ISBN 80-247-0864-7.

[5] ČERNÝ, Jaroslav. *Excel 2000-2007: záznam, úprava a programování maker*. Praha: Grada Publishing, a.s., 2008. 183 s. ISBN 978-80-247-2305-1.

[6] LASÁK, Pavel. *For Next - Cykly - Excel VBA* [online]. ©2006-2014 [cit. 01.20.2013]. Dostupné z: http://office.lasakovi.com/excel/vba-teorie-zaklady/promene-excel-vba/

[7] SOBEK, Milan. *OpenOffice.org: tipy a triky pro záznam a úpravu maker*. Praha: Grada Publishing, a.s., 2006. 152 s. ISBN 80-247-1374-8.

[8] CADFORUM.CZ. *Standardní moduly a procedury ve Visual Basicu pro Aplikace* [online]. ©2012 [cit. 21.01.2014]. Dostupné z: http://www.cadforum.cz/cadforum/Vyvojove-prostredky-AutoCADu/Pasmo/Casti/VisualBasic/VBAZakladyProg/355vbazpmodulyaprocedury.htm#VBA_sub

[9] PÍSEK, Slavoj. *Access 2010: podrobný průvodce*. Praha: Grada Publishing, a.s., 2011. 160 s. ISBN 80-247-7333-3.

[10] PÍSEK, Slavoj. *Access 2003: snadno a rychle*. Praha: Grada Publishing, a.s., 2004. 124 s. ISBN 80-247-0148-0.

[11] KUBÁLEK, TOMÁŠ a MARKÉTA KUBÁLKOVÁ. *Databázový systém Microsoft Access 2007*. Praha: Oeconomica, 2009. 236 s. ISBN 978-80-245-1518-2.

[12] JAKUBÍKOVÁ, Dagmar. *Strategický marketing -Strategie a trendy.* Praha: Grada Publishing, a.s., 2008. 269 s. ISBN 80-247-2690-4.

[13] Administrativní registr ekonomických subjektů. MINISTERSTVO FINANCÍ ČESKÉ REPUBLIKY. *Administrativní registr ekonomických subjektů - Ministerstvo financí České republiky* [online]. ©2013 [cit. 19.03.2014]. Dostupné z: http://wwwinfo.mfcr.cz/cgi-bin/ares/darv_res.cgi?ico=49610929&jazyk=cz&xml=1

[14] ALLRISK.CZ. *Profil společnosti* [online]. ©2014 [cit. 19.03.2014]. Dostupné z: http://www.allrisk.cz/profil-spolecnosti.html

[15] ALLRISK.CZ. *Allrisk* [online]. ©2014 [cit. 19.03.2014]. Dostupné z: http://www.allrisk.cz/

SEZNAM OBRÁZKŮ, TABULEK, GRAGŮ

SEZNAM PŘÍLOH

Příloha č. 1: Databázová procedura na ukončení čísla

```
CREATE OR REPLACE FUNCTION insurance.effective_end_phone_product(idproduct
integer, broker character varying, reason integer, end_date date DEFAULT
NULL::date)
  RETURNS character varying AS
$BODY$

DECLARE
	result		varchar(255) := '';
	cnt_product	integer;
	validTo			date;
	validFrom	date;
	idphonenumber		integer;

BEGIN
	IF end_date is null THEN
		end_date := now();
	END IF;

	/* Kontrola, zda existuje produkt */
	select		count(1),		date(p.valid_from),		date(p.valid_to),
pp.id_phone_number
		into cnt_product, validFrom, validTo, idphonenumber
	from insurance.product p, insurance.phone_product pp
	where p.id_product = idproduct
		and p.id_product = pp.id_product
		and p.deleted is null
		group by p.valid_from, p.valid_to, pp.id_phone_number;

	IF cnt_product <> 1 or cnt_product is null THEN
		result := 'Telefonni produkt neexistuje nebo je smazany';
		return result;
	END IF;

	/* Kontrola, zda lze k danemu datu zmenit produkt (tarif, ...)
vzhledem k vytvorenym fakturam.
	   Pokud byla v pozdejsim datu vytvorena faktura s timto produktem,
nelze jiz zmenit.
	   Kontroluji pomoci db procedury. Ta vraci prazdny retezec, pokud lze
zmenit. Pokud ne, vraci vypis.
	*/
	select doc.test_product_invoice(idproduct, end_date)
	  into result;

	IF result <> '' THEN
	  return result;
	END IF;

	IF validFrom > end_date AND validFrom is not null THEN
		result := 'Datum pocatku produktu je vetsi nez zadane datum
ukonceni.
		return result;
	END IF;

	IF validTo > end_date OR validTo is null THEN
```

```
            update insurance.product set
                  valid_to = end_date,
                  why_end = reason,
                  id_contract_ev_end = 1,
                  edited_by = broker,
                  edited = now()
            where id_product = idproduct;

            update partner.phone_number set
                  valid = 0,
                  edited_by = broker,
                  edited = now()
            where id_phone_number = idphonenumber;

      END IF;

return result;

END;

$BODY$
  LANGUAGE plpgsql VOLATILE
  COST 100;
ALTER   FUNCTION   insurance.effective_end_phone_product(integer,   character
varying, integer, date)
  OWNER TO postgres;
```

Příloha č. 2: Funkce na připojení tabulek do aplikace

```
Public Function Tabulky(DBname, PuvName As String) As Integer
On Error GoTo Err_Tabulky

Dim cnn As String
Dim tfd As TableDef
Dim dbs As Variant
Dim pokus As Integer

Set dbs = CurrentDb
cnn = "ODBC;DATABASE=" & JmenoDatabaze & ";UID=" & AktLogin & ";PWD=" &
AktPSW & ";DSN=" & DSN & ""
       Set tfd = dbs.CreateTableDef(DBname)
       tfd.Connect = cnn
       tfd.SourceTableName = PuvName
       dbs.TableDefs.Append tfd
       Tabulky = 0             'funkce provedena spravne

Exit_Tabulky:
    Exit Function
Err_Tabulky:
    Tabulky = -1            'vraceni chybove hodnoty funkce
    pokus = err.number
    MsgBox err.description & " (kód 106)"
    GoTo Exit_Tabulky
End Function
```