VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA PODNIKATELSKÁ
ÚSTAV INFORMATIKY (UI)

FACULTY OF BUSINESS AND MANAGEMENT
DEPARTMENT OF INFORMATICS

NÁVRH EVIDENCE DOŠLÉ POŠTY

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE MARKÉTA MÜLLEROVÁ
AUTHOR

BRNO 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA PODNIKATELSKÁ
ÚSTAV INFORMATIKY (UI)

FACULTY OF BUSINESS AND MANAGEMENT
DEPARTMENT OF INFORMATICS

NÁVRH EVIDENCE DOŠLÉ POŠTY

PROJECT OF THE REGISTRATIN INCOMING MAIL

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **MARKÉTA MÜLLEROVÁ**
AUTHOR

VEDOUCÍ PRÁCE **Ing. JIŘÍ KŘÍŽ Ph.D.**
SUPERVISOR

BRNO 2007

Vysoká škola: Vysoké učení technické v Brně | Akademický rok: 2006/2007

Fakulta: podnikatelská | Ústav: informatiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Markéta Müllerová

6209R021 - Manažerská informatika

Ředitel ústavu 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 Vám zadává bakalářskou práci s názvem:

Návrh evidence došlé pošty

Project of the registration incoming mail

Pokyny pro vypracování:

Úvod
Vymezení problému a cíle práce
Analýza problému a současné situace
Teoretická východiska práce
Vlastní návrhy řešení, přínos (efektivnost) návrhů řešení
Závěr
Seznam použité literatury
Přílohy

Rozsah grafických prací:	dle potřeby
Rozsah původní zprávy:	cca 40 stran

Seznam odborné literatury:

AVERY Janes: ASP.NET. 2003. 200 s. ISBN 80-251-0121-5
ESPOSITO Dino. ASP.NET a ADO.NET tvorba dinamických webových stránek. 2003. 352 s. ISBN 80-247-0474-9
KOCH Miloš. Datové a funkční modelování. 2004. 108 s. ISBN 80-214-2724-8
LACKO Luboslav. SQL Hotová řešení. 2003. 296 s. ISBN 80-7226-975-5
LACKO Luboslav. SQL Kapesní přehled. 2005. 96 s. ISBN 80-251-0788-4

Vedoucí bakalářské práce:	Ing. Jiří Kříž, Ph.D.
Datum zahájení bakalářské práce:	31. října 2006
Datum odevzdání bakalářské práce:	31. května 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta podnikatelská
L. S.

Ing. Jiří Kříž, Ph.D.
Ředitel ústavu

Doc. Ing. Miloš Koch, CSc.
Děkan

V Brně dne: 16. února 2007

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami:

1. Pan/paní

Jméno a příjmení: Markéta Müllerová

Bytem: Havlíčkova 37, 602 00 Brno

Narozen/a (datum a místo): 14.3.1980, Brno

(dále jen „autor“)

a

2. Vysoké učení technické v Brně

Fakulta podnikatelská

se sídlem Kolejní 2906/4, 612 00 Brno

jejímž jménem jedná na základě písemného pověření děkanem fakulty:

Ing. Jiří Kříž, Ph.D., ředitel Ústavu informatiky

(dále jen „nabyvatel“)

Čl. 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
   - □ disertační práce
   - □ diplomová práce
   - □ bakalářská práce
   - □ jiná práce, jejíž druh je specifikován jako .....

   (dále jen VŠKP nebo dílo)

Název VŠKP:	Návrh evidence došlé pošty
Vedoucí/ školitel VŠKP:	Ing. Jiří Kříž, Ph.D.
Ústav:	Informatiky
Datum obhajoby VŠKP:	červen 2007

VŠKP odevzdal autor nabyvateli v*:

- □ tištěné formě – počet exemplářů - jeden
- □ elektronické formě – počet exemplářů - jeden

* hodící se zaškrtněte

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizovaní výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti
   - □ ihned po uzavření této smlouvy
   - □ 1 rok po uzavření této smlouvy
   - □ 3 roky po uzavření této smlouvy
   - □ 5 let po uzavření této smlouvy
   - □ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: …….

…….. ……
Nabyvatel Autor

Abstrakt

Bakalářská práce se zabývá návrhem evidence doručené firemní pošty. Na základě podrobného prostudování situace ve firmě je navrženo jako nejlepší řešení zpracování aplikace pro firemní intranet. Aplikace umožňuje zaznamenávání veškeré doručené pošty, sleduje její vyřízení, umožňuje vyhledávání a tisk sestav podle různých kritérií.

Abstracts

This thesis deals with the design of the evidence of company inbox. Based on the detailed study of the situation in the company, the processing of the application for the company intranet is proposed as the best solution. This application enables recording all inbox, watching mail processing, searching, and printing of schemes according to various criteria.

Klíčová slova

Aplikace, databáze, evidence, intranet, pošta

Keywords

Application, database, evidence, intranet, mail

BIBLIOGRAFICKÁ CITACE

Bibliografická citace mé práce:

MÜLLEROVÁ, M. *Návrh evidence došlé pošty*. Brno: Vysoké učení technické v Brně, Fakulta podnikatelská, 2006. 61 s. Vedoucí bakalářské práce Ing. Jiří Kříž, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že jsem svou bakalářskou práci vypracovala samostatně a použila jsem pouze podklady uvedené v přiloženém seznamu.

V Brně dne..... .....

Markéta Müllerová

Obsah

Úvod

Každému, ať už se jedná o osobu právnickou, fyzickou nebo obyčejného občana, je čas od času doručena pomocí doručovacích společností nějaká zásilka. Pokud těchto zásilek nechodí denně mnoho, je možné je všechny v klidu přečíst a podle uvážení dále zpracovat. Jedná-li se však o zásilky doručované do větších firem, jeden člověk zpravidla nemůže zvládnout jejich čtení a řádné zpracování, nestačí mu na to ani čas ani kvalifikace. Pokud si firma nevytvoří nějaký systém, který zaručuje včasné a úplné vyřízení veškerých potřebných dokumentů, může se dostat do velmi nepříjemných situací jako např. ve firmě, kde již delší dobu pracuji, budeme jí říkat Nádobí a.s.[1].

Ve firmě je zaběhnutou praxí, že každou doručenou zásilku převezme asistentka jednatele, otevře ji, přečte si její obsah a buďto ji sama dle svých možností zpracuje, nebo předá dalšímu zaměstnanci, o kterém je přesvědčena, že je k vyřízení zásilky kompetentní. Bohužel často nastává situace, že vybraný zaměstnanec předá zásilku jinému zaměstnanci a ten ji předá dále, tak zásilka koluje po firmě, až nakonec skončí v zapomnění jako nevyřízená. Tato nevyřízená pošta však už několikrát dostala firmu do nepříjemných situací, už takhle skončily zásilky z finančních úřadů i od soudů a firma jednou přišla díky založenému dopisu také o velmi strategickou zakázku.

Jako řešení evidence doručených zásilek se pokusím navrhnout a posléze vytvořit aplikaci s pracovním názvem POSTA MM. Aplikace bude přístupná přes podnikový intranet, takže nebude třeba žádného speciálního software na jednotlivá pracoviště, která mají mít do evidence přístup, postačí běžný prohlížeč webu, který je k dispozici na každém počítači. Na straně serveru budou požadovány běžící služby internetového a databázového serveru. Tato aplikace bude ve firmě Nádobí a.s. zprovozněna a po určené době budou zhodnoceny její přínosy, popř. negativa.

Cílem mé bakalářské práce je vytvořit uvedenou aplikaci, zavést její používání ve firmě a pomocí nového systému zakládání doručených zásilek zefektivnit činnost jednotlivých

[1] Firma si nepřeje zveřejnit svůj pravý název ani identifikační údaje

zaměstnanců, zajistit kontrolu vyřizování firemní pošty, a tím také vybudovat lepší vztahy s dodavateli, odběrateli i konečnými zákazníky.

1. Analýza současného stavu firmy

1.1. O firmě

Firma Nádobí a.s. vznikla zápisem do obchodního rejstříku dne 4. 1. 1998. Při jejím založení činil základní kapitál 10.000.000,- Kč. Základní kapitál je tvořen 10 neveřejně obchodovatelnými akciemi na majitele v jmenovité hodnotě 1.000.000,- Kč za kus. Tyto akcie si stejným dílem koupili dva lidé a zároveň se postavili do vedení společnosti. Oba mají stejné pravomoci a aktivně se podílejí na chodu společnosti.

Hlavní činností firmy je nákup zboží za účelem dalšího prodeje. Firma má velmi významné postavení na trhu s prodejem domácích potřeb. Prodej realizuje dvěma způsoby: dražší a kvalitnější zboží formou přímého prodeje, zboží denní spotřeby formou velkoobchodních dodávek do hypermarketů a menších firem. Firma zboží nakupuje od mnoha významných tuzemských i zahraničních dodavatelů.

Další činností je pronájem rozsáhlého komplexu budov, skládajícího se ze sedmipodlažní administrativní budovy a několika samostatných skladovací a manipulačních ploch. K doplňkovými činnostem patří organizačně ekonomické poradenství, doprava zboží, zámečnické a elektroinstalatérské práce.

1.2. Organizační struktura firmy

Organizační struktura firmy je velmi složitá, z toho důvodu ji rozdělíme do několika částí.

1.2.1. Hlavní organizační struktura

Firma je rozdělena na 5 divizí. Některé divize se dále člení na poddivize. Činnost jednotlivých divizí i poddivizí popíši vždy u aktuální organizační struktury.

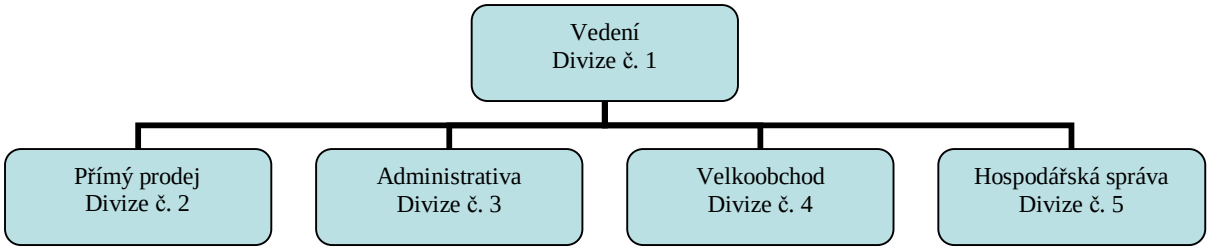


Schéma 1 – Organizační struktura firmy

1.2.2. Vedení firmy – Divize č. 1

Největší rozhodovací pravomoc jak v divize č. 1, tak i v celé firmě, mají akcionáři = majitelé firmy.

Divize č. 1 se kromě vedení, tj. majitelů firmy a sekretářky, skládá ze dvou poddivizí, právního oddělení a ekonomického oddělení. V čele obou oddělení stojí manažeři divize, kteří zodpovídají za fungování celé poddivize.

Právní oddělení (divize č. 11) má na starosti vytváření smluv, řešení právních situací včetně soudních sporů.

Ekonomické oddělení (divize č. 12) zajišťuje ekonomický chod firmy.

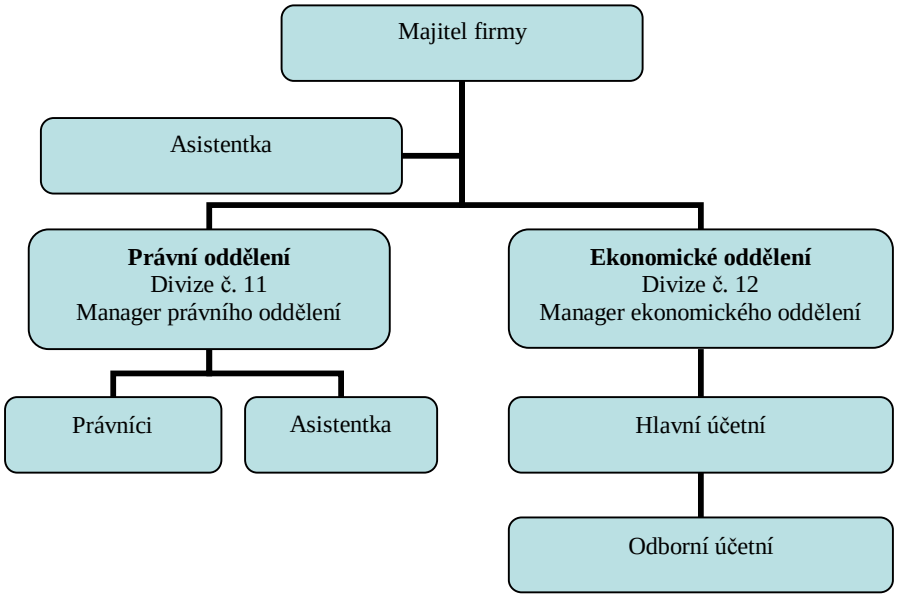


Schéma 2 - Divize č. 1 – Vedení firmy

1.2.3. Přímý prodej – Divize č. 2

Divize č. 2 se specializuje na prodej výrobků prostřednictvím přímého prodej, tj. přímého kontaktu se zákazníkem. Dlouhodobým výzkumem firma zjistila, že prodej zboží prostřednictvím obchodního zástupce, který komunikuje se zákazníkem přímo, je mnohem efektivnější, než když si zákazník vybírá zboží v regále ve velkém marketu.

Divize Přímý prodej se dělí na 3 poddivize:

Koordinace (divize č. 21) – zaměstnanci divize sestavují, zpracovávají a vyhodnocují výsledky prodeje zboží formou přímého prodeje a dbají, aby bylo na skladě vždy dostatek zboží, různými soutěžemi se snaží motivovat obchodní zástupce k lepším výkonům,
TitanPan (divize č. 22) – zabývá se prodejem kuchyňského nádobí vyrobeného z titanu, především kastrolů, hrnců a pánví,
Vakuum (divize č. 23) – zabývá se prodejem vakuového systému na skladování potravin.

Chod celé divize Přímý prodej kontroluje manažer divize. Sestavuje plán tržeb, dbá na jejich plnění a z případného nesplnění se zodpovídá majitelům firmy.

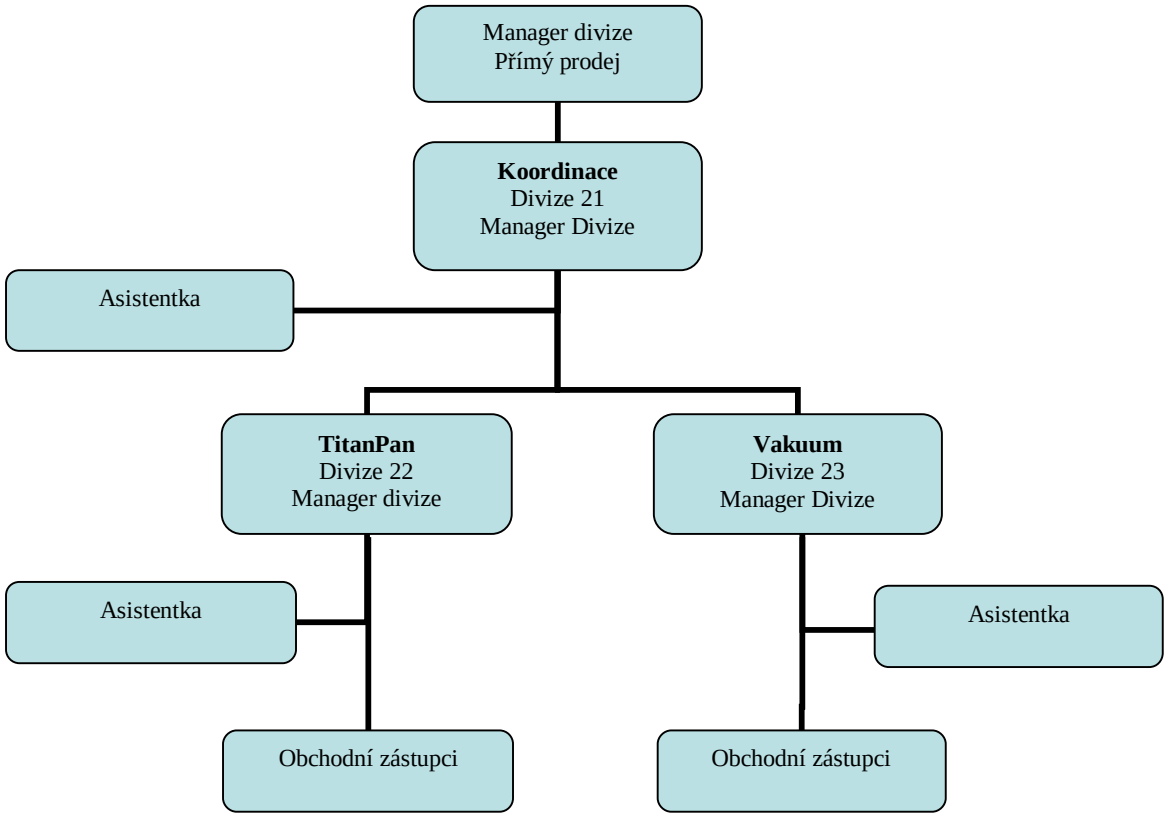


Schéma 3 - Divize č. 2 – Přímý prodej

1.2.4. Administrativa – Divize č. 3

Divize Administrativa má na starosti veškeré činnosti týkající se pohybu písemností i samotného zboží jako sledování skladových zásob, fakturování, odesílání i přímý rozvoz na místo určení.

V této divizi jsou také zaměstnáni správci počítačové sítě a technici, kteří mají na starosti chod veškerého počítačového vybavení firmy, a také poštovní oddělení, které zabezpečuje evidenci doručených i odeslaných zásilek a veškerou komunikaci mezi firmou a doručovacími společnostmi.

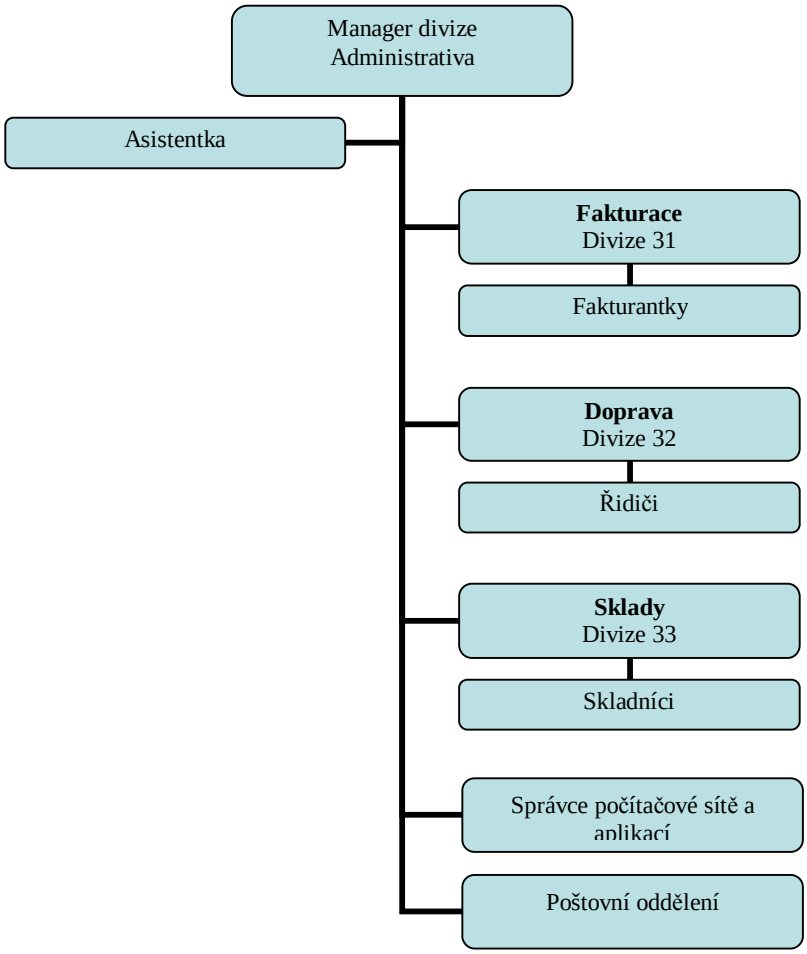


Schéma 4 - Divize č. 3 – Administrativa

1.2.5. Velkoobchod – Divize č. 4

Divize Velkoobchod zprostředkovává prodej zboží zákazníkovi pomocí dodávek do velkých hypermarketů, supermarketů i malých soukromých obchůdků. V jejím sortimentu nalezneme především zboží denní potřeby od kuchyňských výrobků přes zboží sloužící jako doplňky do koupelen po užitečné a praktické výrobky pro kutily.

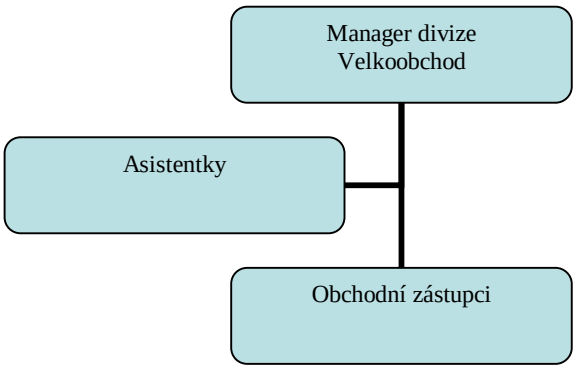


Schéma 5 - Divize č. 4 – Velkoobchod

1.2.6. Hospodářská správa – Divize č. 5

Hlavní činností divize Hospodářská správa je údržba a modernizace celého areálu firmy. Neboť komplex budov a ostatních ploch je velmi rozsáhlý, velkou část těch prostor firma pronajímá, což přináší každoročně nemalé finanční prostředky. Tyto prostředky jsou používány především na správu areálu. Z tohoto důvodu firmy zaměstnává poměrně velkou základnu obslužného personálu.

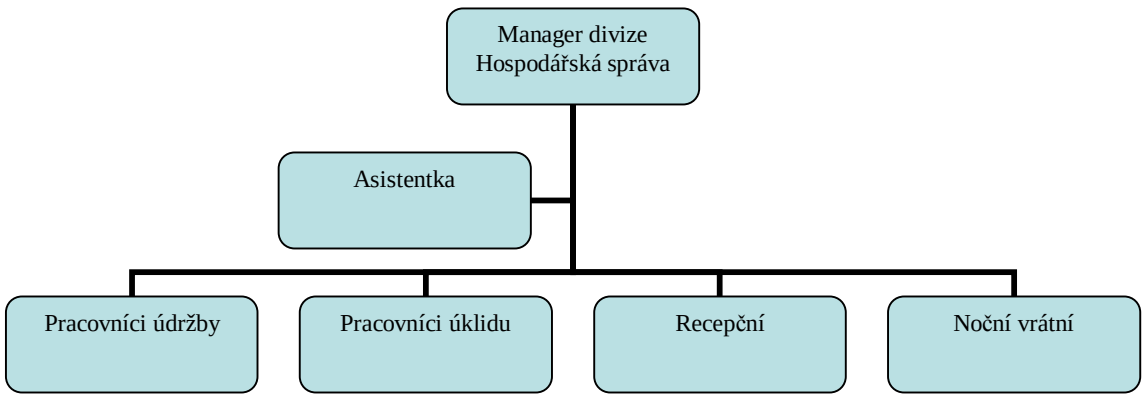


Schéma 6 - Divize č. 5 - Hospodářská správa

1.3. Hardwarové a softwarové vybavení společnosti

V dnešním technicky vyspělém světě si snad nikdo nedokáže představit vykonávat jakoukoliv kancelářskou práci bez výpočetní techniky. Firma Nádobí a.s. vynaložila

v roce 2006 nemalé finanční prostředky na modernizaci svého počítačového vybavení. Majitelé firmy si dobře uvědomují, že pokud zaměstnancům nabídnou kvalitní technologie, jejich práce se zefektivní, a tím firmě vrátí vložené finanční prostředky.

1.3.1. Hardwarové vybavení

1 ks server DELL 840

40 ks pracovních stanic DELL Optiplex GX270

20 ks tiskáren Kyocera Ecosys 1020D

5 ks stolních scannerů HP ScanJet 3800

1.3.2. Softwarové vybavení

Server

Serverový operační systém Microsoft Server 2000 a MS-SQL Server 2000

Osobní počítače

Každý osobní počítač používá jako operační systém Windows XP Proffesional a dále má instalován balík kancelářských programů Microsoft Office Proffesional 2003 (vše na základě výhodného programu pro menší firmy - Open Licence firmy Microsoft). Všechny počítače jsou vybaveny také antivirovým software Symantec Antivirus. Naprostá většina uživatelů má na své pracovní stanici oprávnění User, tzn. omezené možnosti poškodit počítač neodbornými zásahy.

Účetní a ekonomický program

Již řadu let využívá firma pro zpracování účetnictví a ekonomických výstupů modulární řešení programu firmy ABRA software a.s..

Ke své činnosti využívá následující moduly z řady ABRA G3:

- Jádro systému
- Prodej
- Nákup
- Skladové hospodářství
- Business Intelligence
- Mzdy a personalistika
- Banka

- Pokladna
- Majetek
- Účetnictví a výkazy
- Reporty, sestavy a formuláře

Ostatní programy

Na několika počítačích je instalován TotalCommander (pohodlná správa souborů a adresářů) a na jednom počítači také FinePrint pdfFactory (export dokumentů do PDF).

1.4. Hardwarové a softwarové potřeby společnosti

1.4.1. Hardwarové potřeby

Jak bylo již v paragrafu 1.3 zmíněno, firma v roce 2006 vynaložila nemalé finanční prostředky na zmodernizování počítačového vybavení společnosti. Klíčový pracovníci nyní pracují na velmi moderních zařízeních, což šetří jejich čas a finance firmy. V následujících letech má společnost v plánu další investice do hardwaru, neboť technologie jdou stále kupředu a to, co je dnes považováno za velmi výkonné stroje, bude za pět let nedostačující.

1.4.2. Softwarové potřeby

Jako aktuální softwarovou potřebu a zároveň požadavek majitele firmy shledávám v aplikaci pro evidenci došlé pošty. Tato pošta je nyní zapisována na poštovním oddělením do knihy došlé pošty, kde jsou uvedeny údaje o datu doručení, jménu odesílatele a jménu zaměstnance, kterému byla pošta předána, ale tento systém řeší potřeby firmy nedostatečně.

1.5. Softwarové nedostatky společnosti

Základním problémem je nesystematičnost evidence, rozdělování a hlavně nedostatečné sledování vyřízení doručené pošty. Z tohoto důvodu se také firma před časem zajímala o rozšiřující modul programu ABRA G3 Evidence pošty. Bohužel však nebyla splněna očekávání majitelů ani managerů firmy.

Jako řešení tohoto problému se pokusím navrhnout a posléze vytvořit aplikaci pro evidenci doručené pošty, která by splňovala požadavky vedení firmy. Systém bude

postaven jako aplikace pro podnikový intranet, takže nebude třeba žádného speciálního SW na jednotlivá pracoviště, která mají mít do evidence přístup.

2. Teoretické poznatky nezbytné pro návrh aplikace

2.1. Historie

2.1.1. Databázové systémy

„Počátek databází a databázového zpracování dat můžeme nalézt v 60. letech, kdy firma IBM vytvořila první databázový systém založený na hierarchickém modelu. Tento model předpokládal hierarchické uspořádání dat, podobné jako organizační struktura organizace. Datová základna byla tvořena stromy, které měly mezi nadřízeným a podřízeným uzlem vztah 1:n (jeden nadřízený uzel má jednoho nebo více následníků).

V 70. letech se začaly stromy v hierarchickém modelu propojovat a bylo možné popisovat všechna data a vazby mezi nimi, jak se vyskytují v realitě. V polovině 70. let se proto začaly objevovat zcela nové databázové systémy založené na relacích a relační algebře. 80. léta pak patří již zcela relačním databázím a jazyku SQL, který se postupně stal jediným prostředkem pro práci s tímto typem databázových systémů.“ (10), str. 28)

2.1.2. Databázový jazyk SQL

„V letech 1974 až 75 probíhal ve firmě IBM výzkum týkající se možnosti využití relačních databází. Pro tento projekt bylo nutné vytvořit sadu příkazů, kterými by se relační databáze ovládala. Vznikl tak jazyk SEQUEL (Structured English Query Language). Jak vypovídá jeho název, bylo cílem tvůrců vytvořit jazyk, ve kterém by se příkazy tvořily syntakticky co nejblíže běžnému jazyku (angličtině). Tento jazyk byl v upravené formě SEQUEL/2 použit v databázovém systému SYSTÉM R v roce 1977.

Výhody relačního přístupu si uvědomily i další firmy. Tak v roce 1979 uvedla na trh firma Relational Software, Inc. (nyní Oracle Corporation) svůj relační databázový systém s názvem Oracle. Firma IBM nadále vylepšovala svůj systém a v roce 1981 uvedla nový systém SQL/DS a později, v roce 1983, DB2. Vznikaly však i databázové systémy dalších firem – např. Progress, Informix a SyBase. V těchto systémech se používaly různé verze jazyku SEQUEL, který se přejmenoval na SQL (Structured Query Language).

Vzrůstající význam relačních databází si vyžádal nutnost standardizace jazyka pro jejich používání. Americký standardizační institut (ANSI) původně zamýšlel vytvořit standard na základě zcela nového jazyka RDL. V roce 1984 se však ukázalo, že jazyk se SQL stává standardem de facto. Proto se nový standard založil na tomto jazyku a bývá označován jako SQL86 podle roku 1986, ve kterém byl přijat.

V průběhu následujících let se ukázalo, že původní standard obsahuje některé nedostatky a naopak v něm nejsou obsaženy některé důležité prvky týkající se zejména integrity dat. Proto byl v roce 1992 přijat nový standard označovaný jako SQL-92 nebo pouze SQL2. Za základní rozšíření je považováno přidání primárních klíčů a definice integritních omezení dat v tabulkách. V současné době je stále ještě v návrhu verze standardu SQL3, která by měla zejména reflektovat objektový přístup a objektové databáze obecně.“ (10), str. 28-29)

2.1.3. ASP.NET

ASP.NET je 4 roky stará technologie pro vytváření webových aplikací. Podobně jako u Active Server Pages (ASP) jde o programový kód, který se provádí na straně serveru. ASP.NET nemá se starým Active Server Pages téměř nic společného. Zachovává si několik jeho výhod (práce s ActiveX prvky) a přidává mnoho nových možností (práce s plnohodnotnými jazyky a .NET Frameworkem). (7)

2.2. Teoretické poznatky pro návrh databáze

2.2.1. Databázové systémy

Databázový systém získáme, jestliže sloučíme data, jejich vazby a nástroje, pomocí kterých tato data vytváříme, aktualizujeme, vyhledáváme a rušíme.

Data jsou zprávy, které člověk zachytí a porozumí jim. Uživatel si je může uložit pro pozdější zpracování, transformovat je do jiné podoby, např. zaznamenat na papír nebo do počítače. Data mají vypovídací schopnost.

2.2.2. Funkční modelování

„Zabývá se zkoumáním a algoritmizací činností, procesů, které v informačním systému probíhají. Při popisu činností v IS můžeme provádět hierarchický rozklad funkcí od

nejobecnějšího až do elementárních funkcí, které mají uživatelé k dispozici – například od modulu účetnictví obecně až k výpočtu DPH na faktuře. Funkce vyšší úrovně vznikají pojmenováním určité skupiny nižších funkcí bezprostředně podřízené úrovně. Počet úrovní dekompozice modelu není omezen a liší se podle konkrétního řešeného případu.“ (5), str. 65)

2.2.2.1.Procesní diagram

„Při tvorbě procesního diagramu kreslíme na levou stranu stránky události, které proces ovlivňují, na pravé straně zachycujeme jednotlivé automatizované i neautomatizované činnosti. Diagram můžeme kreslit na různých stupních podrobnosti podle potřeby. Zpravidla začínáme na nejvyšším stupni zobecnění a podle potřeby diagram převádíme na dílčí řešení jednotlivých částí.“ (5), str. 73)

2.2.2.2.Stavový diagram

„V této metodě zkoumáme možné stavy objektů, které mohou nastat a snažíme se popsat, co tyto změny přechodu mezi stavy vyvolá. U tohoto diagramu si musíme uvědomit, že jej vždy kreslíme pro určitou entitu (objekt) a zkoumáme možné přechody jeho stavů. V elipsách jsou zakresleny možné stavy. Šipky určují, za jaké podmínky přechází systém z jednoho stavu do druhého, přičemž u šipek rozepisujeme příslušné podmínky.“ (5), str. 73)

2.2.2.3.Diagram toku dat

„Diagram toku dat (Data Flow Diagram) je jedna z nejpoužívanějších metod funkčního modelování. Můžeme z něj vyčíst návaznost jednotlivých činností v rámci úlohy, jaké datové vstupy a výstupy se v úloze objevují (tedy s jakými soubory a doklady se pracuje) a kdo jednotlivé činnosti provádí.

DFD diagramy můžeme kreslit na různé rozlišovací úrovni. Obvykle začínáme zachycením systému jako celku a postupně rozpracováváme jednotlivé funkce až na úroveň jedné úlohy.“ (5), str. 74)

2.2.2.4.Vývojový diagram

„Vývojový diagram patří společně s DFD diagramem k nejpoužívanějším. Jeho hlavní výhodou je možnost zachytit velmi dobře větvení zpracování podle splnění či nesplnění požadovaných podmínek.

Při kreslení vývojového diagramu dodržujeme přirození směr „shora dolů“ a „zleva doprava“, ve kterých nemusíme malovat na spojovacích větvích šipky. V případě, že bychom museli větve křížit, použijeme spojky a tomuto problému se tak vyhneme.“ (5), str. 80)

2.2.3. Relační datové modely

Relační datové modely nám umožňují zachytit v modelu data o zkoumaných objektech, ale jejich vzájemné vztahy, což nám umožňuje přiblížit se více reálnému světu. (5), str. 24)

Entita – je objekt reálného světa, který je schopný nezávislé existence a je jednoznačně odlišný od ostatních objektů. Příkladem entity může být například občan:
Josef Novák, narozen 22.2.1960 v Příbrami, bytem v Olomouci, rodné číslo 600222/1234
(6), str. 27)

Vztah – je vazba mezi dvěma nebo více entitami

Atribut – je funkce, která přiřazuje jednotlivým entitám nebo vztahům hodnotu. Tato hodnota určuje některou podstatnou vlastnost entity nebo vztahu. (6), str. 27)

Proces návrh systému spočívá jednak v identifikaci typů entit jako množin objektů stejného typu, v identifikaci typů vztahů, do kterých budou entity vstupovat a v přiřazení atributů, které blíže popisují vlastnosti jednotlivých entit a vztahů. (6), str. 27)

2.2.4. Integrita relačního modelu

2.2.4.1.Integritní omezení pro entity

Doménová integrita – každá hodnota každého z atributů relace (položky věty) musí být z množiny hodnot (domény) pro daný atribut přípustných:

1. definice domény jako množiny hodnot (může být využito více atributů)
2. specifikace povolených hodnot pro daný atribut (položku věty):
   - typ pole (datový typ),
   - povinné zadání položky, neprázdná hodnota,
   - jedinečnost hodnot v rámci sloupce,
   - rozsah hodnot – minimální, maximální hodnota,
   - implicitní (standardní hodnota),
   - maska pro vkládání,
   - seznam přípustných hodnot (číselník).

Entitní integrita – každá relace musí mít určen tzv. primární klíč – jeden nebo více atributů, jejichž hodnoty jednoznačně identifikují každý z řádků relace:

1. je jednoznačná, tzv. v relaci neexistuje druhá n-tice (věta tabulky), která by pro tuto množinu atributů měla stejné hodnoty,
2. je minimální, tzv. žádný atribut není možné vypustit, aniž by se porušilo pravidlo 1 (žádná z jejích podmnožin nemá tuto vlastnost).

Primární klíč je základním prostředkem adresace n-tice relace a platí zde tato pravidla:

- u každého atributu primárního klíče nesmí chybět hodnota (doména),
- každá n-tice relace musí být v každém okamžiku identifikovatelná hodnotou primárního klíče.

Kandidátní klíč je totéž jako primární klíč, se stejnými vlastnostmi, ale není vybrán jako primární klíč. V relaci může být více kandidátních klíčů, jeden z nich je vybrán jako primární klíč, ostatní kandidátní klíče se nazývají alternativní.

Referenční integrita - definuje vztah dvou tabulek, a to pomocí cizích klíčů.

Cizí klíč je atribut, který splňuje tyto nezávislé vlastnosti:

1. každá hodnota je buď plně zadaná nebo plně nezadaná,
2. existuje jiná relace s takovým primárním klíčem, že každá zadaná hodnota cizího klíče je identická s hodnotou primárního klíče nějaké n-tice této jiné relace.

Cizí klíč nám tedy společně s primárním klíčem jiné tabulky umožňuje vytvářet spojení, což je hlavní účel relačního datového modelu. Platí zde pravidla referenční integrity:

- cizí a odpovídající primární klíč musí být definovány na stejné doméně (soulad hodnot),
- databáze nesmí obsahovat žádnou nesouhlasnou hodnotu cizího klíče (5), str. 28-30).

2.2.4.2.Integritní omezení pro vztahy entit

Vztah 1:1 – každá jedna n-tice relace odpovídá jedné (nebo žádné) n-tici jiné relace.
Vztah 1:N – každá jedna n-tice relace odpovídá jedné nebo více n-ticím jiné relace.
Vztah N:M – několik n-tic relace odpovídá jedné nebo více m-ticím jiné relace. V praxi se tento vztah aplikuje pomocí spojovací tabulky, tzn. vztah N:M se rozloží na vztahy N:1 a 1:M.

2.2.5. Normalizace

Normalizace je činnost, při které upravujeme návrhy datových struktur tak, aby splňovaly zvolené normalizační formy – úrovně. Tyto normalizační formy či pravidla vycházejí z požadavku na efektivní ukládání dat a minimalizují redundance při zachování integrity a konsistence dat. Datový model, který porušuje některou z normalizačních forem není navržený optimálně. Při normalizaci databáze na vyšší normalizační úrovni musí být normalizována na všech předcházejících.

Normalizace je postupná dekompozice relací (tabulek) do vhodnějšího tvaru, tak aby:

- byla zachována bezeztrátovost při zpětném spojení
- byly zachovány závislosti
- bylo odstraněno opakovaní informací (redundance) (5), str. 55-56)

2.2.5.1.První normální forma – multizávislost

Tabulka splňuje podmínku první normální formy, jestliže všechny atributy (sloupce) jsou atomické, tzn. dále nedělitelné. Jeden sloupec nesmí obsahovat více druhů údajů nebo, řečeno matematickou terminologií, – musí obsahovat skalární hodnotu. Hodnotou sloupce také nesmějí být relace. Když databázová tabulka tyto podmínky nesplňuje, je v takzvané nulté normální formě, a je třeba ji rozložit.

2.2.5.2.Druhá normální forma – funkční závislost

Tabulka splňuje podmínku druhé normální formy, jestliže splňuje podmínku první normální formy a každý atribut kromě primárního klíče musí být úplně závislý na celém primárním klíči. Druhá normální forma se proto týká jen tabulek, které mají více primárních klíčů. Když má tabulka jen jeden primární klíč, podmínka pro druhou normální formu je splněna automaticky.

2.2.5.3.Třetí normální forma – tranzitivní závislost

Tabulka je v třetí normální formě jestliže je v druhé normální formě a zároveň neexistují závislosti neklíčových sloupců tabulky.

2.2.5.4.Čtvrtá normální forma

Tabulka je ve čtvrté normální formě, jestliže je v třetí normální formě a popisuje jen jeden fakt anebo souvislost.

2.2.5.5.Pátá normální forma

Tabulka je v páté normální formě, jestliže je ve čtvrté normální formě a není možné do ní přidat nový sloupec, popřípadě skupinu sloupců bez toho, aby se rozpadla na několik dílčích tabulek. (6), str. 44-48)

2.2.6. Použité příkazy SQL

Příkazy SQL se dělí do několika skupin, hlavní jsou:

- Příkazy pro manipulaci s daty (zkráceně označované jako **DML**) jsou příkazy pro získání dat z databáze a pro jejich úpravy.
- Příkazy pro definici dat (zkráceně **DDL**) jsou příkazy, kterými se vytvářejí struktury databáze – tabulky, indexy, pohledy a další objekty.

Při vytváření aplikace byly použity DDL příkazy:

- CREATE TABLE – slouží k vytvoření databázové tabulky,
- DROP TABLE – slouží k odstranění exitující tabulky včetně všech jejich řádků.

Ve vlastní aplikaci potom DML příkazy:

- SELECT – slouží k výběru dat podle různých kritérií,
- INSERT – pomocí tohoto příkazu můžeme do tabulky přidat nové řádky,
- UPDATE – slouží ke změně hodnot v jednom nebo více sloupcích u jednoho nebo více řádků tabulky,
- DELETE – slouží k vymazání jednoho nebo více řádků tabulky.

Dále byly použity vestavěné funkce a řídící struktury např.:

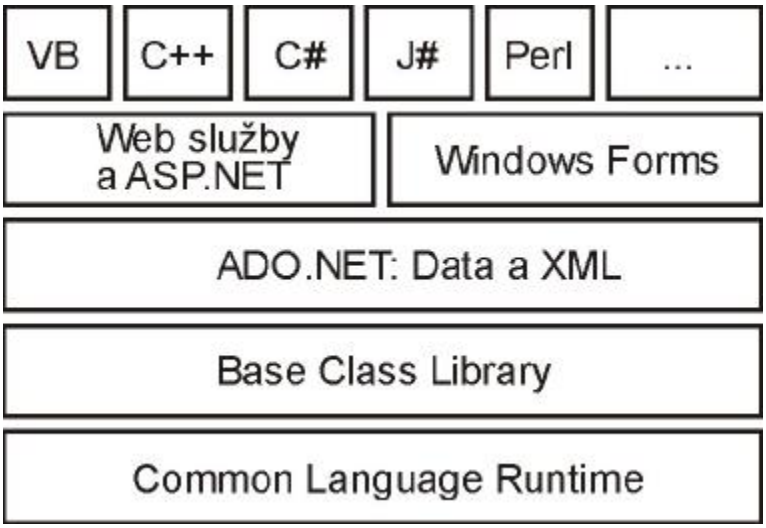
- ISNULL – nahradí případnou null hodnotu za hodnotu v argumentu příkazu,
- IF / ELSE – řídí provádění na základě podmínky,
- EXISTS (SELECT …) – ověřuje existenci datového výběru.

2.3. Teoretické poznatky pro návrh ASP

2.3.1. Technologie ASP.NET

V rámci ASP.NET můžete standardně pracovat s programovacími jazyky, jaký je třeba Visual Basic .NET nebo C#.NET. Pozoruhodnou vlastností této technologie je, že VB.NET v ASP.NET se od VB.NET, ve kterém programujete ve Visual Studiu, neliší. Pracujete tak se stejným jazykem a se společným běhovým prostředím, kterým je .NET Framework. Jde o takzvaný řízený kód (MSIL, Microsoft Intermediate Language), což není ani strojový jazyk ani zdrojový kód. Je to tzv. mezijazyk, který se vytvoří při kompilaci a při spuštění je zpracováván prostředím CLR (Common Language Runtime). (3)

2.3.1.1.Architektura .NET frameworku



Obrázek 1 - Architektura .NET frameworku (13)

Na nejnižší úrovni se nachází *CLR- Common Language Runtime* realizující základní infrastrukturu, nad kterou je framework vybudován. Nad CLR se nachází několik hierarchicky umístěných knihoven. Ty jsou rozděleny do jmenných prostorů. Základem je knihovna nazvaná *Base Class Libary*. Nad ní je knihovna pro přístup k datům a práci s XML soubory a nad ní sada knihoven usnadňující práci s uživatelským rozhraním. Je rozdělena do dvou skupin: *pro usnadnění vytváření webových aplikací a pro vytváření „klasických" aplikací.* Poslední vrstvu tvoří nelimitovaná množina programovacích jazyků. Jejich základní vlastnosti definuje *CLS – Common Language Specification*. V současné době jsou firmou Microsoft podporovány čtyři jazyky Visual Basic, C++, C# a JScript. Tato množina ale není uzavřena a jakýkoliv výrobce ji může rozšířit. Celá tato architektura je podpořena novou verzí vývojového nástroje Visual Studio .NET/Visual Studio 2003. (13)

2.3.2. Výhody ASP.NET oproti ASP

- Díky kompilovanému kódu běží aplikace rychleji a více chyb je zachyceno už při vývoji.
- Uživatelsky definované ovládací prvky lze použít jako šablony, čímž se významně redukuje duplicitní kód.

- Podobný přístup jako k aplikacím pro Windows zjednodušuje přechod od jednoho prostředí k druhému.
- Bohatý výběr ovládacích prvků a knihoven tříd velmi zrychluje vývoj aplikací.
- Programátoři mají na výběr velké množství programovacích jazyků.
- Schopnost cachovat celou stránku nebo pouze její části podstatně zvyšuje výkon serveru.
- Lze jej provozovat na různých operačních systémech i webových serverech, např. IIS (Windows), Apache (Windows, Linux s open source implementací .NETu Monem). (1)

2.3.3. Konfigurace ASP.NET

ASP.NET a .NET framework nabízí velice výkonný a flexibilní model založený na konfiguračních souborech ve formátu XML. Tyto soubory obsahují velké množství různých konfiguračních nastavení, která lze definovat pomocí správně formátovaného XML. Pokud je v některém z těchto konfiguračních souborů provedena změna, nová nastavení jsou znovu zkompilována a při dalším požadavku na aplikaci nebo soubor uložen do cache. (2), str. 3)

Hlavní dva soubory se nazývají machine.config a web.config.

„Nastavení v souboru machina.config se vztahují na celý server, zatímco web.config pouze na aplikaci, které náleží. Pokud je v nastavení v aplikačním souboru web.config v rozporu s nastavením v serverovém souboru machine.config, použijí se nastavení ze souboru web.config, který má vyšší prioritu. Konfigurační soubory jsou bezpečné, takže k nim nelze přistupovat pomocí požadavků http. Pokud dojde k pokusu o stažení tohoto souboru, je uživateli vrácena chyba http 403.“ (2)1), str. 12)

2.3.4. Použité ovládací prvky

2.3.4.1.Menu

Menu je ovládací prvek zajišťující veškerou funkčnost pro vytvoření kompletní a inteligentní navigace. Položky lze vytvořit při návrhu aplikace, nebo mohou být

vytvářeny až za běhu, případně i přímo vázány na zdroj dat. Vzhled lze zvolit z předpřipravených šablon.

2.3.4.2.TextBox

TextBox je ovládací prvek pro zadávání řetězce znaků. Je obdobou HTML tagu <input type="text"… , ale s možností vázání ke zdroji data a zobrazit jeho obsah.

2.3.4.3.ListBox

ListBox je ovládací prvek pro výběr položky ze seznamu. Je obdobou HTML tagu select, ale s možností vázání ke zdroji data a zobrazit jeho obsah.

2.3.4.4.DataGrid a GridView

„Ovládací prvek DataGrid vykresluje vícesloupcovou mřížku, plně přizpůsobitelnou pomocí šablon. Přestože má tento ovládací prvek rozsáhlé programovací rozhranní a velké množství atributů, generuje tabulku HTML s roztroušenými odkazy, které zajišťují vysokou míru interaktivity, například příkazy pro řazení a stránkování.

Prostřednictvím ovládacího prvku DataGrid lze vytvářet jednoduché, datově vázané sloupce, které zobrazují údaje načtené ze zdroje dat, šablonové sloupce pro vlastní rozvržení obsahu buněk, a v neposlední řadě sloupce založené na příkazech, které umožňují do mřížky přidat specifickou funkčnost." (3), str. 44)

GridView je obdobou původního prvku DataGrid pro verzi ASP.NET 2.0

2.3.4.5.DetailsView

Mnohé jeho rysy jsou shodné s prvkem GridView, od něhož se však liší v jedné důležité vlastnosti – Zatímco GridView zobrazuje více datových řádků najednou, DetailsView zobrazuje právě jeden řádek. Velmi často se proto používá k editaci dat (k čemuž lze ale použít i GridView) anebo ke vkládání dat. (14)

2.3.4.6.SqlDataSource

SqlDataSource zprostředkovává přístup k datům. Navzdory poněkud zavádějícímu názvu, lze využít nejenom pro přístup k datům SQL serveru, ale i k dalším relačním datovým zdrojům přístupným přes OLEDB/ODBC rozhraní.

2.3.4.7.Data-binding

Společnou vlastností uvedených ovládacích prvků je tzv. data-binding.

„Data-binding je princip, který umožňuje spojovat poskytovatele dat (typicky databázi) s konzumenty dat (typicky ovládacím prvky umístěnými na formuláři). Důležité je, že při data-bindingu není nutné psát žádný kód. Data-binding je záležitost nastavení datového zdroje a konzumenta dat a typicky se provádí prostřednictvím pohodlného vizuálního rozhraní (odborným názvem – deklarativně). (14)

2.3.5. Použité příkazy ASP.NET

Přestože je velká většina programového kódu generována přímo Visual Studiem, nelze se psaní zdrojového kódu vyhnout úplně. „Klasické programování“ bylo použito např. na stránce NovaPosta.aspx, aby mohla existovat univerzální stránka jak pro zadávání nové pošty, tak pro úpravu již existujícího záznamu. V obsluze události Page_Init se rozlišuje podle parametru z URL, zda se má zakládat nový záznam, a také se rozhoduje o zakázání editace některých polí v případě dodatečné úpravy položky (dodatečně nelze měnit např. datum doručení).

„Dim - deklarace proměnné

Proměnné se deklarují dvěma způsoby. V metodách se deklarují příkazem Dim nebo Static, za kterým následuje název proměnné, typ a případně i inicializační hodnota. V případě hodnotových typů deklarace vypadá takto:

```
Dim int As Integer
Dim lng As Long = 10
Dim str As String = „Ahoj“
```

První řádek deklaruje proměnnou int typu integer, která bude mít implicitní hodnotu 0. Druhá proměnná lng je inicializovaná na hodnotu 10. Třetí proměnná str je typu string a má implicitní hodnotu „Ahoj“.“ (4) ,str. 16)

Př. z aplikace:

```
Dim APrijato As Date
Dim AVyridit As Date
Dim AId As Integer
```

DateAdd - Vrací datum zvětšené o určený časový interval

Funkci lze použít k přidání nebo odečtení určeného časového intervalu k datu. Například ji lze použít k výpočtu datumu za 30 dní ode dneška nebo čas 45 minut od aktuálního času. Pro připočítávání k datumu lze použít den roku ("y"), den ("d") nebo den v týdnu ("w"). Funkce nevrací chybné datum.

Př. z aplikace:

```
AVyridit = DateAdd("d", 14, APrijato)
```

Sub…End Sub – deklarace procedury

Deklaruje jméno, argumenty a script tvořící tělo Sub procedury. Typické použití je pro části kódu, které je potřeba vykonávat na více různých místech programu. Zápis jejich zdrojového kódu se provede pouze jednou a odkazem na jejich jméno se provádí jejich volání. Při volání procedur lze navíc použít parametry, kterými se proceduře předají vstupní hodnoty. Předávání argumentů je možné hodnotou a také odkazem. Předání odkazem znamená, že procedura může sama parametry modifikovat, čímž se parametry stávají vstupně/výstupními.

Procedura může pracovat s proměnnými hlavního programu, ale může také obsahovat vlastní deklarace platné pouze uvnitř těla procedury.

V objektově orientovaném programování jsou procedury použity zejména jako obsluha událostí.

Př. z aplikace – obsluha po stisku tlačítka:

```
Protected Sub Button1_Click(ByVal sender As Object, ByVal e As
System.EventArgs)
      DS_Zam.Insert()
End Sub
```

If ... Then ... ElseIf … Else … End If – podmíněné provádění

Pomocí příkazu If lze zajistit provádění části programu pouze při splnění definované podmínky. Je-li podmínka hodnoty True, provedou se příkazy následující za Then. Je-li podmínka False, pak jsou vyhodnoceny bloky ElseIf (jsou-li nějaké). Je-li nalezena podmínka True, jsou provedeny příkazy následující příslušné Then. Není-li žádná podmínka bloku ElseIf vyhodnocena jako True (nebo není-li žádný blok ElseIf), provedou se příkazy následující Else. Po provedení příkazu následujících Then nebo

Else, pokračuje provádění příkazem následujícím za End If. Bloky Else a ElseIf jsou volitelné. Lze mít libovolné množství bloků ElseIf, ale žádný se nesmí objevit po bloku Else. Příkazy If mohou být vnořené, tzn. obsažené v jiných příkazech If.

Př. z aplikace:

```
If AId > 0 Then
      DetailsView1.UpdateItem(False)
Else
      DetailsView1.InsertItem(False)
End If
```

Select Case … Case … Case Else … End Select – podmíněné provádění

Podobně jako předchozí konstrukce jde o řízení provádění programu v závislosti na podmínce. Konstrukce Select je vhodná při větším množství testovaných podmínek, kde by se muselo použít mnohonásobné ElseIf, ale na rozdíl od If nelze použít podmínky složité, testuje se pouze hodnota jediného výrazu.

Př. z aplikace:

```
Select Case AId
Case 0
      DS_Posta.SelectCommand = DS_Posta.SelectCommand.ToString & "
      WHERE p.Vyriz_ID=1"

Case Is < 0
      Label1.Visible = False
      DropDownList1.Visible = False

Case Else
      DS_Posta.SelectCommand = DS_Posta.SelectCommand.ToString & "
      WHERE p.Vyriz_ID=1 AND Zam_ID=" & AId
      DropDownList1.SelectedValue = AId
      Session("FavouriteID") = AId

End Select
```

CType – přetypování

Příkaz CType provádí explicitní přetypování proměnné. S jeho pomocí lze změnit typ proměnné např. z integer na string.

V objektově orientovaném programování se používá ale zejména v případech, kdy je potřeba upřesnit obecně deklarovaný parametr. Např. když se obsluze nějaké události předává parametr s obecným typem Object. Abychom mohli přistupovat k vlastnostem a metodám konkrétního objektového typu, musíme provést jeho přetypování. Typ proměnné tím tedy vlastně neměníme, ale pouze upřesňujeme, abych mohli s odkazovaným objektem pracovat.

Př. z aplikace:

```
Protected Sub DropDownList1_SelectedIndexChanged(ByVal sender As
Object, ByVal e As System.EventArgs)
      Response.Redirect("~/PrehledPosty.aspx?id=" & CType(sender,
      DropDownList).SelectedValue)
End Sub
```

With … End With – zjednodušení zápisu

Pomocí bloku mezi klíčovými slovy With a End With lze přistupovat k vlastnostem nebo metodám objektového typu bez opakovaného uvádění kompletního zdlouhavého zápisu, který je definován pro celý blok pouze jednou.

Př. z aplikace:

```
With DetailsView1.Fields do
      Item(8).Visible = False
      Item(9).Visible = False
      Item(10).Visible = False
End With
```

Rem – poznámka

Klíčovým slovem Rem, případně znakem ‘ (apostrof) se uvozuje poznámka. Vše až do konce řádku je při provádění programu ignorováno. Psaní poznámek je důležité zejména u složitějších konstrukcí, a to např. pro jejich případné pozdější úpravy. Pomocí poznámek lze také snadno zakázat provádění částí programu např. při ladění aplikace.

Př. z aplikace:

```
REM sestaveni podminky WHERE
```

2.3.6. Jazyk HTML

Jazyk HTML je značkovací jazyk, určený pro webové dokumenty. Ve své podstatě je to směs textů k zobrazení a značek, tzv. tagů, které určují, jak se má obsažený text zobrazovat. I když moderní editory vytvářejí HTML kód prakticky automatizovaně, nelze se bez jeho znalosti obejít, často je třeba do výsledného kódu provádět konečné úpravy. S tvorbou statických HTML stránek už jsem měla nějaké předchozí zkušenosti.

2.3.7. Kaskádové styly - CSS

Stejně jako u předchozího se jedná o prostředek, který definuje výslednou podobu stránky. Použití technologie css poskytuje více formátovacích možností a také lepší

přehlednost výsledného kódu, protože popis vzhledu může být zcela oddělen od vlastního obsahu dokumentu.

2.3.8. Skriptovací jazyk Visual Basic - VB

Jazyk VB je v mé aplikaci použit pro zápis vlastního zdrojového kódu aplikace. Tento kód je prováděn na straně serveru a teprve jeho výsledek je posílán do prohlížeče klienta. Zjednodušeně řečeno – HTML kód není načítán ze souboru jako u statické stránky, ale je výsledkem vykonaného skriptu. Přestože většina kódu je generována přímo prostředím Visual Studia v době návrhu aplikace, nelze se bez znalosti skriptovacího jazyka obejít. Technologie ASP.NET umožňuje pořizovat zdrojové kódy prakticky v libovolném jazyce, pro který je k dispozici překladač. Z vestavěných variant (Visual Basic Skript, C#, J#) jsem zvolila prvně jmenovaný, jehož zápis se pro mě jeví nejsnáze srozumitelný.

2.3.9. Objektově orientované programování - OOP

Starší verze Visual Basic se s novou verzí, pokud jde o práci s objekty, vůbec nemohou rovnat. Visual Basic .NET podporuje dědičnost (pouze jednoduchou) u objektů, tvorbu rozhraní, možnost implementace jednoho nebo více rozhraní, přetěžování metod a vlastností, vytváření delegátů, překrývání, konstruktory a destruktory, sdílené členy prostě vše, co by měl moderní objektový jazyk podporovat.

Ve starších verzích Visual Basic šlo samozřejmě také vytvářet i používat objekty, ovšem Visual Basic .NET pracuje jen s objekty. Proto je vlastně vše, co děláte ve Visual Basic .NET, objektové programování. (4), str. 33)

2.3.10. Příkazy Transact SQL - T-SQL

Podobně jako u HTML kódu, máme při programování ve Visual Studiu k dispozici vygenerovaný kód T-SQL příkazů. Často je však potřeba do příkazů zasáhnout a základní znalost jazyka SQL je nutností. V této oblasti jsem si vystačila se znalostmi z předchozího studia (příkazy SELECT, INSERT, UPDATE, JOIN, DELETE apod.)

2.4. Software

2.4.1. Microsoft Visual Studio Express

Visual Studio Express je soubor produktů pro rychlý vývoj aplikací, který představila firma Microsoft v roce 2004. Tato produktová řada je určena zejména pro začátečníky a studenty a lze ji stáhnout a používat zdarma.

„Produkty z této skupiny jsou uzpůsobeny vždy pro jeden programovací jazyk platformy .NET, tzn. Visual Basic, C++, C# nebo J#. Pátým produktem je Visual Web Developer určený k tvorbě webových aplikací. V tomto případě jsou pro vývoj k dispozici jazyky Visual Basic, C# a J#. Všechny tyto produkty jsou šířeny odděleně a i v případě instalace na jednom počítači je nelze nijak integrovat. Obsahují pouze základní prostředí Visual Studia pro design aplikací a psaní kódu, nedodávají se k nim žádné rozšiřující nástroje a ani není možné je propojit s externími aplikacemi pro reporting nebo správu verzí zdrojového kódu. (12)

2.4.2. SQL Server 2005 Express

SQL Express je nástupcem MSDE, tedy volně distribuovatelným databázovým strojem na bázi SQL Serveru. Oproti MSDE má menší množství omezení, hlavně je zrušeno omezení na nejvýše pět současně běžících dotazů, což znamená, že Express lze úspěšně použít i pro webové aplikace. Novinkou je také dostupnost jednoduchého nástroje pro správu, Management Studio Express. (11)

Na rozdíl od ostatních verzí SQL Serveru, není toto řešení vhodné pro rozsáhlejší projekty, mimo jiné kvůli technickým omezením:

- Maximální velikost datového souboru databáze je 4GB.
- Je podporován pouze jeden procesor (SQL Server Express lze nainstalovat na víceprocesorový systém, ale bude využívat pouze jeden).
- Podporováno je maximálně 1 GB RAM (Pokud má systém více paměti, tato nebude využita).
- Není přítomna služba SQL Server Agent. (8)

3. Návrh aplikace

3.1. Volba technologií

Prvním krokem zpracování aplikace bylo vybrat nejlepší způsob řešení s ohledem na základní požadavky, které bych shrnula následovně:

- přehledné výstupy,
- jednoduché zadávání,
- minimální náklady na vývoj i provoz aplikace.

Ve snaze vyhovět uvedeným bodům jsem jako možné řešení navrhla zpracovat celý systém jako aplikaci pro podnikový intranet. Toto řešení má především následující výhody:

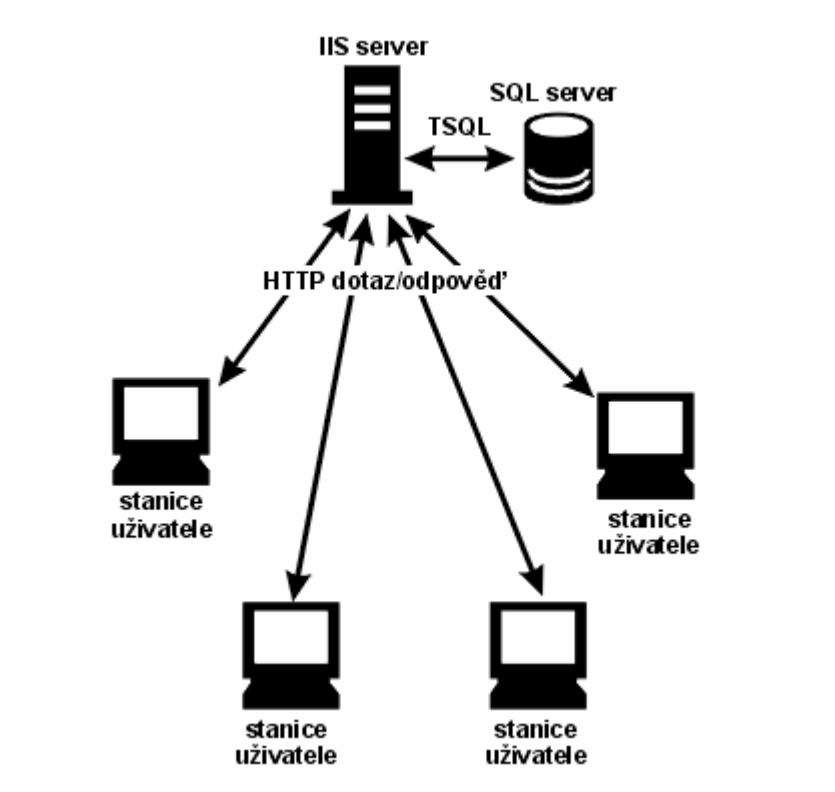
- nulové náklady na koncového uživatele (všichni již mají své pracovní stanice vybaveny připojením k podnikové síti a prohlížečem internetu),
- nulové náklady na intranetový server (ve firmě je již v provozu server se službou webového serveru),
- nulové náklady na datový sklad (lze použít databázi v bezplatné verzi),
- jednoduchá obsluha aplikace (uživatelé se nebudou muset učit obsluhovat nový software, aplikace je přístupná jako běžné webové stránky).

Vlastní technologie pro aplikaci je de facto předurčen stávajícím programovým vybavením serveru. Zbývá se tedy rozhodnout mezi technologiemi ASP a ASP.NET, které jsou obě na webovém serveru Microsoft k dispozici. Rozhodování bylo po prostudování informací na několika www stránkách zabývajících se programováním webových aplikací jednozančné - zvítězil novější ASP.NET. Na rozdíl od starší technologie ASP je pro ASP.NET k dispozici kvalitní vývojový nástroj, který je navíc ve verzi Express zdarma.

Datový sklad jsem navrhla pro MS SQL server, který je opět ve verzi Express dostupný zdarma. Omezení verze Express 2.4.2 nijak nelimitují použití v mé aplikaci.

3.2. Schéma komunikace

Uživatel ze své stanice přes webový prohlížeč zobrazuje formuláře pro zadávání nebo výstup dat. Na základě požadavků, které odejdou z prohlížeče k webovému serveru jsou prováděny T-SQL příkazy nad databází.



Obrázek 2 - Schéma komunikace

3.3. Konfigurace

3.3.1. Konfigurace serveru IIS

K ladění aplikace postačuje vestavěný virtuální server, ale pro testování a zavedení do ostrého provozu je potřeba nakonfigurovat službu IIS na Windows serveru.

3.3.1.1.Uživatelské účty na IIS

Při konfiguraci přístupových oprávnění ke službě IIS lze volit mezi anonymním přístupem a přístupem na základě ověření, které se dále dělí na další podle způsobu, jakým má ověření proběhnout. Pro naše účely postačilo nastavení anonymního přístupu,

protože všichni zaměstnanci, kteří mají doménový účet (mohou se přihlásit k podnikové síti) jsou zároveň oprávněni pracovat s aplikací Pošta MM.

3.3.2. Konfigurace MS-SQL serveru

Podobně je třeba konfigurovat a v našem případě i nainstalovat MS SQL Server. Využila jsem zkušeností s předchozího studia, kdy jsem instalaci databázového serveru prováděla na svém domácím počítači.

3.3.2.1.Uživatelské účty na SQL serveru

Na SQL Serveru 2005 je možné používat uživatelské účty domény Windows a také účty založené přímo na SQL serveru.

3.3.2.2.Účet pro běh aplikace

Aby mohla má aplikace přistupovat k datům na SQL serveru, bylo nutné vytvořit pro ni účet. Princip spočívá v tom, že k SQL serveru přistupuje aplikace prostřednictvím tohoto účtu, nikoliv uživatelé jednotlivě. Parametry tohoto přihlášení jsou zabudovány do aplikace pomocí ConnectionStringu v konfiguračním souboru web.config.

Aplikační účet dostal přiděleny následující role:

„db_datareader

Role je určena uživatelům, kteří potřebují prohlížet data v databázi. Členové této role mohou vybírat všechna data z libovolné uživatelské tabulky v databázi.

„db_datawriter

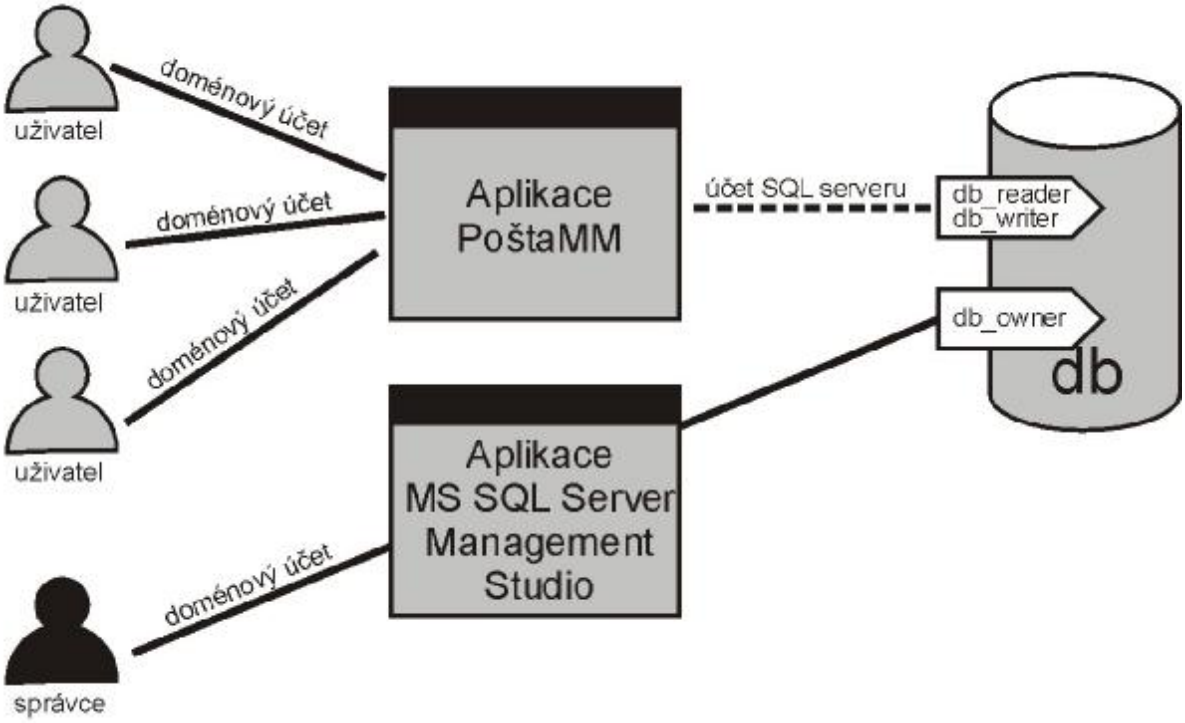
Role je určená uživatelům, kteří potřebují přidávat nebo modifikovat libovolná data v jakékoliv uživatelské tabulce v databázi. Členové této role smí provádět následující příkazy na libovolném objektu databáze: DELETE, INSERT a UPDATE.“ (7), str. 244)

3.3.2.3.Účet pro administraci

Pro správu databáze byla dvěma uživatelským účtům domény Windows přidělena role:

„db_owner

Role je určena uživatelům, kteří potřebují kompletní kontrolu nad všemi aspekty databáze. Členové této role mohou přidávat oprávnění, měnit nastavení databáze, provádět údržbu databáze a jakékoli další administrační úkoly v databázi včetně odstranění databáze.“ (7), str. 244)



Obrázek 3 - Schéma oprávnění při přístupu k SQL serveru

3.4. Popis procesu

a) Sekretářka zadá do informačního systému firmy veškerou doručenou poštu se všemi požadovanými náležitostmi.
b) Sekretářka poštu rozdělí dle divizí, kterých se pošta týká, a předá ji jednotlivým asistentkám divize.
c) Asistentka divize si poštu přečte a rozhodne, zda je schopna či není schopna poštu vyřídit.
d) V případě, že asistentka divize není schopna poštu vyřídit, předá ji zpět sekretářce a proces se vrátí do bodu b).

e) V případě, že asistentka divize je schopna poštu vyřídit, poštu vyřídí a zaznamená jakým způsobem byla pošta vyřízena.
f) Vyřízenou poštu předá asistentka divize zpět sekretářce a ta zadá do informačního systému firmy místo, kde se bude doručená pošta spolu s popisem vyřízení archivovat.
g) Sekretářka uskuteční vlastní archivaci založením pošty včetně způsobu vyřízení na příslušné místo do odpovídajícího šanonu.

3.5. Postup při zpracování aplikace

3.5.1. Návrh struktury databáze a vytvoření tabulek

3.5.1.1.Funkční modelování

Nejprve si celý proces namodelujeme. Využijeme k tomu procesní, stavový a vývojový diagram.

Procesní diagram

Tato metoda zkoumá, jak na události, které proces ovlivňují, reagují jednotlivé automatizované i neautomatizované činnosti.

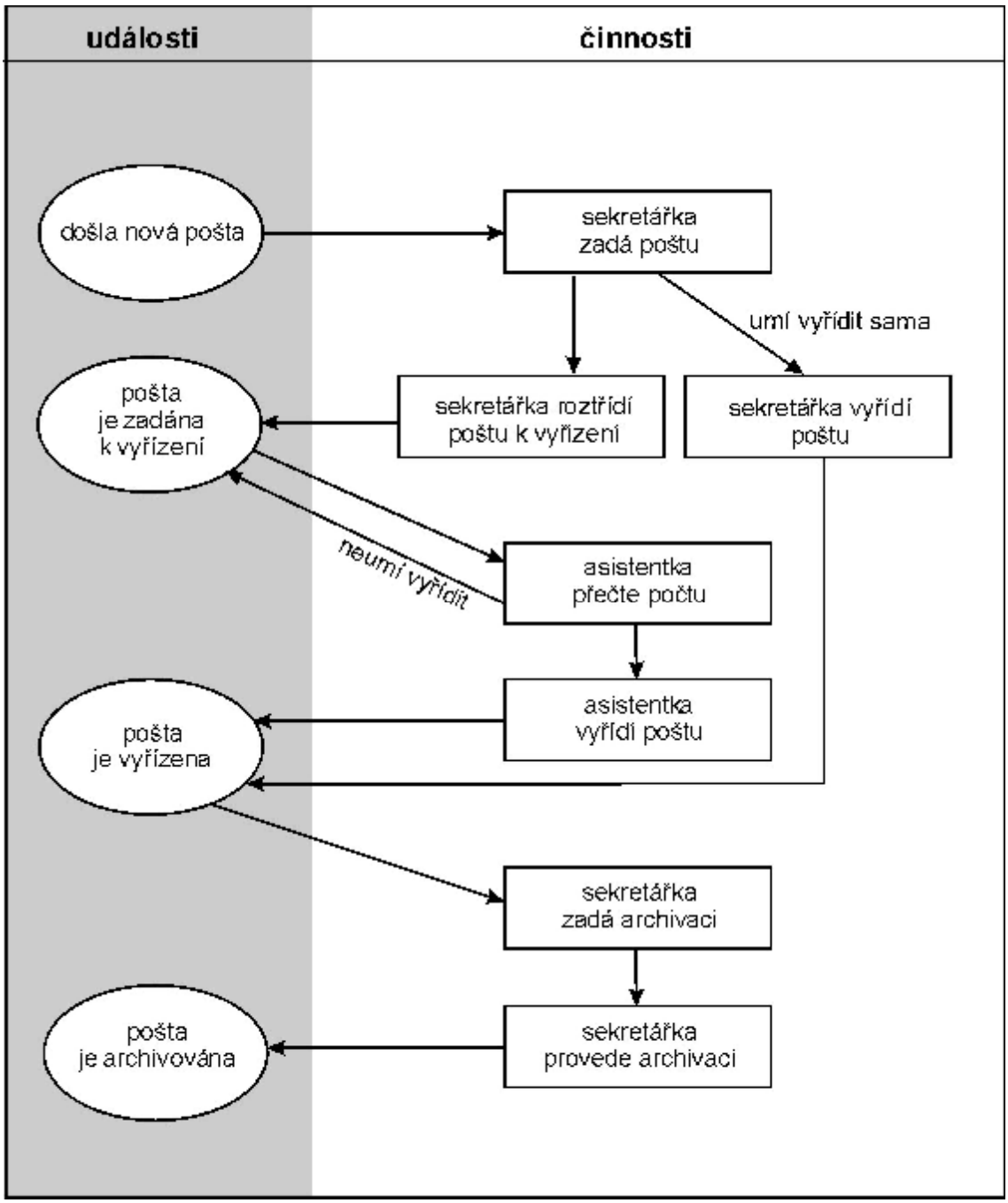


Diagram 1 - Procesní diagram

Stavový diagram

V této metodě zkoumáme možné stavy objektů, které mohou nastat a snažíme se popsat, co tyto změny přechodu mezi stavy vyvolají.

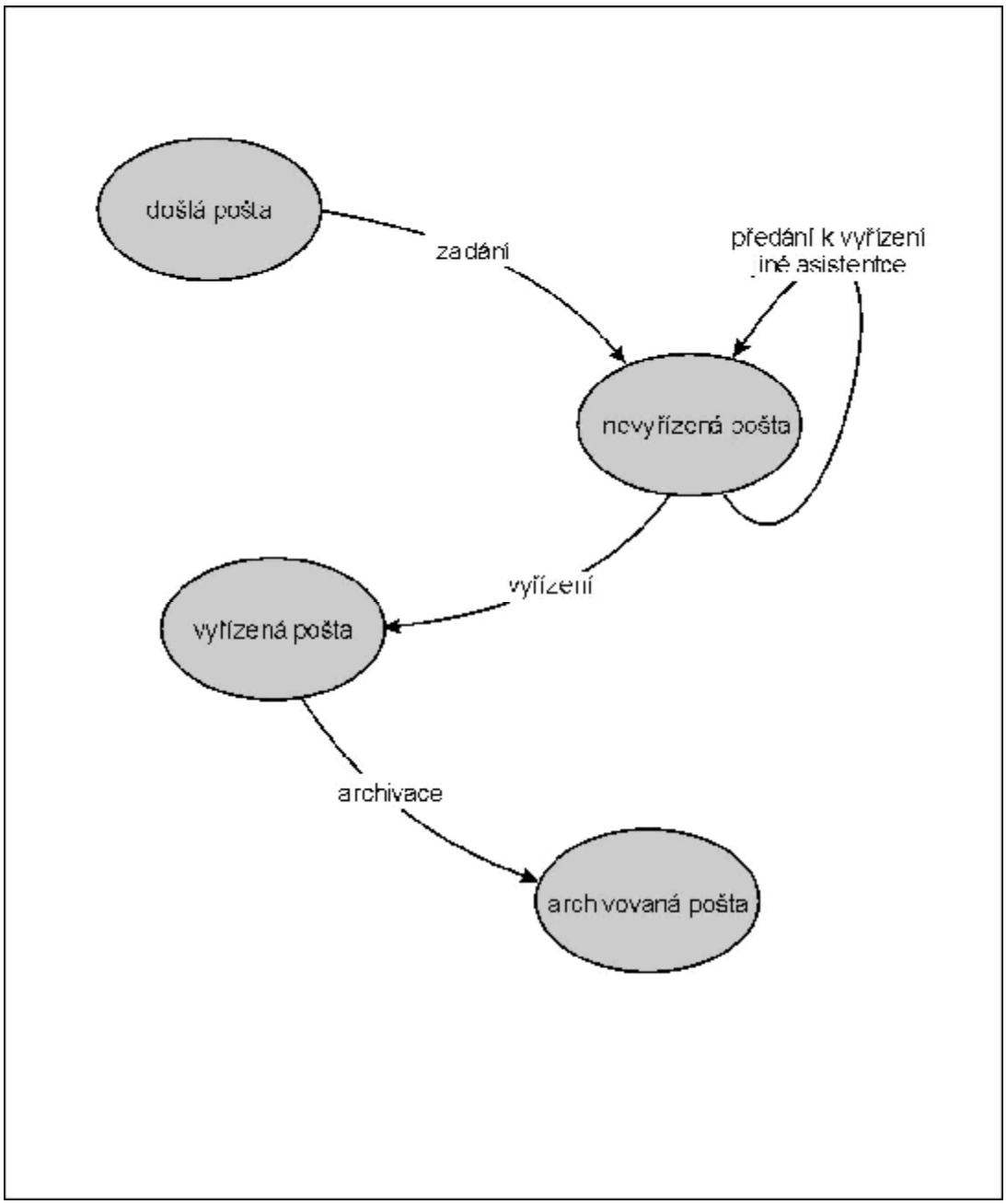


Diagram 2 – Stavový diagram

Vývojový diagram

Vývojový diagram zachycuje velmi dobře větvení zpracování podle splnění či nesplnění požadovaných podmínek. V uvedeném příkladu jde o postup při zadávání.

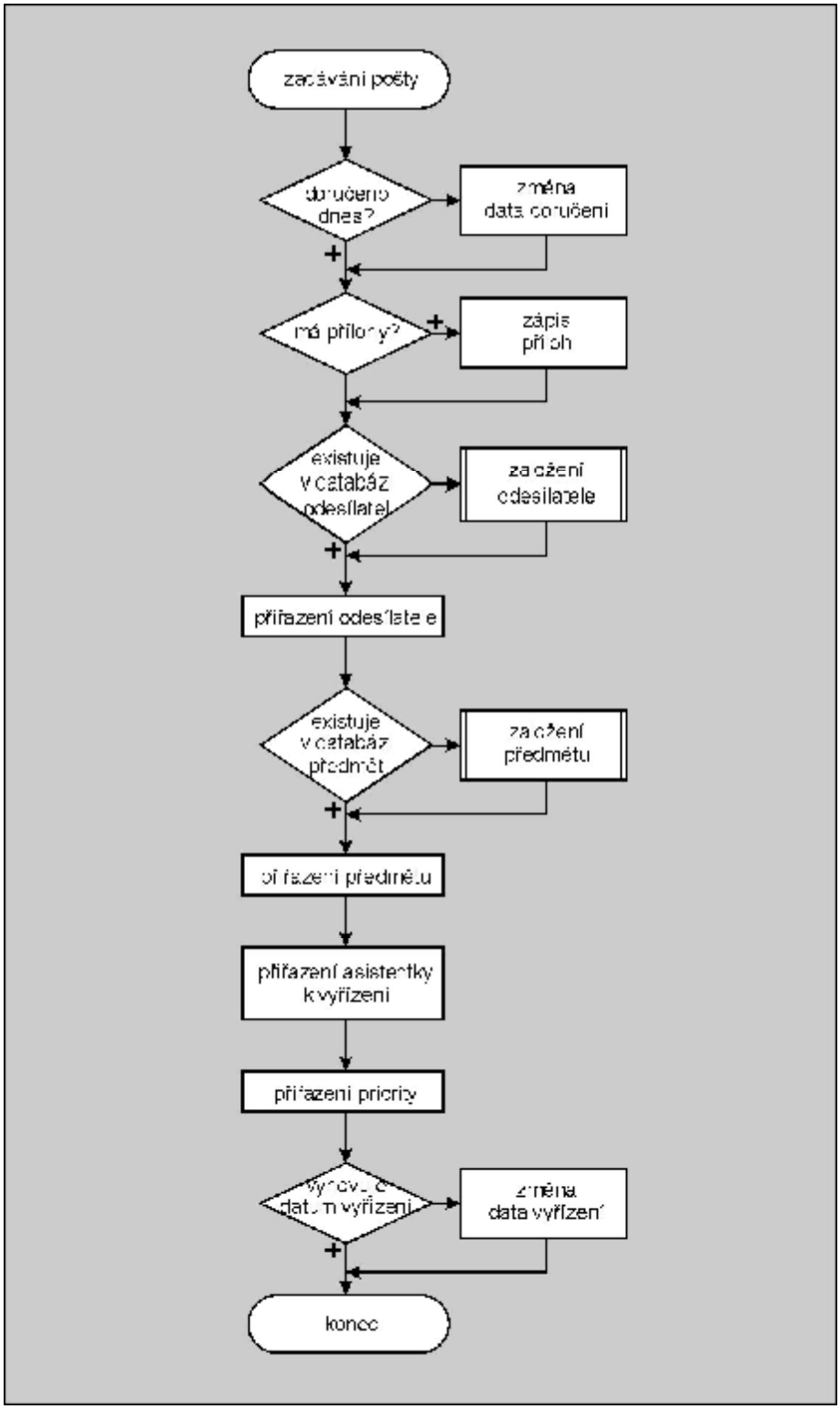
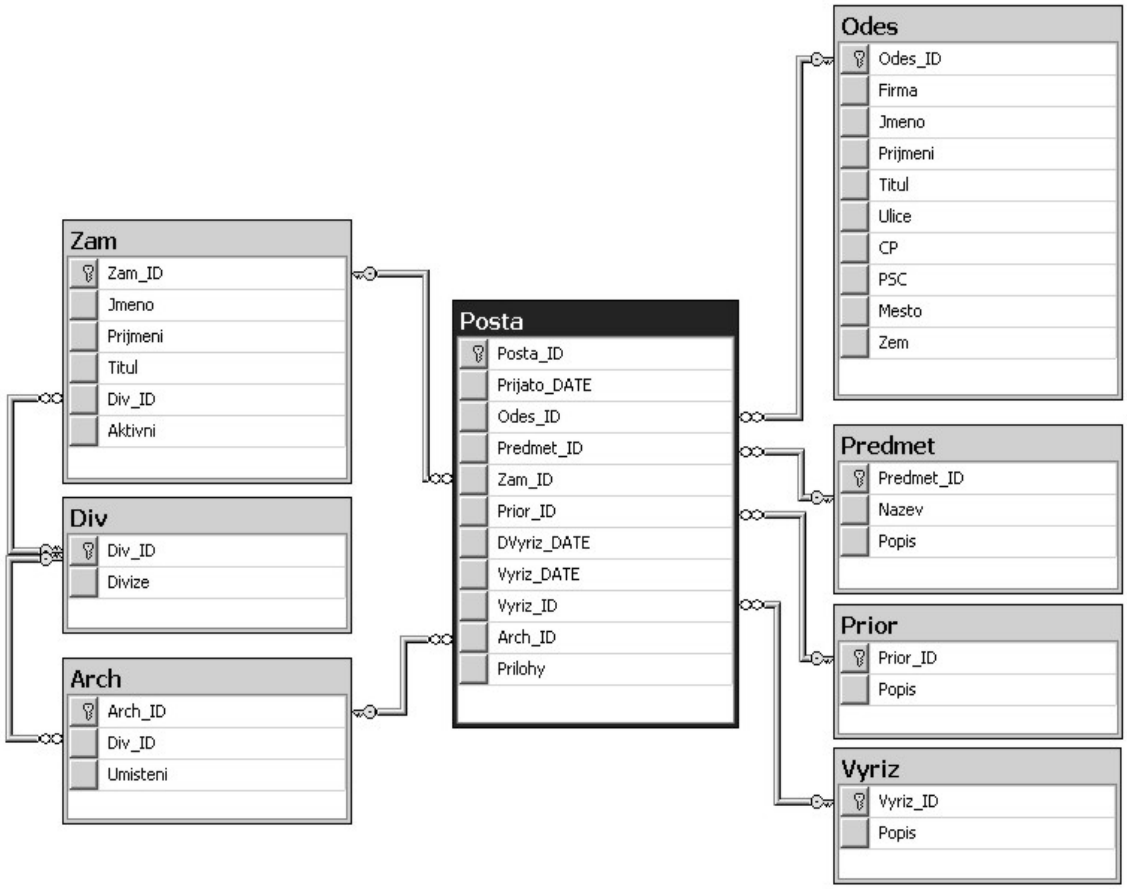


Diagram 3 - Vývojový diagram

3.5.1.2.Návrh struktury databáze

Celá databáze je navržena tak, aby vyhověla podmínkám integrity relačního modelu 2.2.4 a všem normálním formám 2.2.5.



Obrázek 4 - Návrh struktury databáze

Na obrázku číslo 4 je přehled všech navržených tabulek. U každé tabulky je vyznačen primární a cizí klíč. Integritní omezení pro vztahy mezi jednotlivými tabulka je 1:N.

Základní tabulka - **Posta - tabulka došlé pošty**

Posta_ID - vlastní klíč záznamu

Prijato_DATE - datum přijetí

Odes_ID - ► cizí klíč v tabulce odesílatelů

Předmět_ID - ► cizí klíč v tabulce předmětů (různá korespondence se často váže k již existujícímu předmětu)

Zam_ID	-	► cizí klíč v tabulce zaměstnanců (kdo má poštu vyřídit, údaj může být při předání na jiného pracovníka změněn)
Prior_ID	-	► cizí klíč v tabulce priority (jaká je priorita vyřízení)
DVyriz_DATE	-	datum doporučeného vyřízení (aplikace předvyplní lhůtu 14 dnů, lze upravit při zadávání)
Vyriz_DATE	-	datum skutečného vyřízení (doplněno v akci vyřízení)
Vyriz_ID	-	► cizí klíč v tabulce způsobů vyřízení (jak bylo vyřízení provedeno)
Arch_ID	-	► cizí klíč v tabulce archivů (kam byla fyzicky zásilka uložena)
Prilohy	-	popis příloh (volný text pro popis příloh)

Ostatní tabulky

Zam	–	zaměstnanci
Div	–	divize
Arch	–	archivační místa
Odes	–	odesílatelé
Predmet	–	předměty došlé pošty
Prior	–	číselník priorit
Vyriz	–	číselník způsobů vyřízení

3.5.1.3.Manuální naplnění tabulek zkušebními daty

Prvotní naplnění daty proběhlo „ručně“, tj. přímou editací tabulek v prostředí SQL Management Studia.

3.5.2. Návrh aplikace

3.5.2.1.Vytvoření master-page stránky

Master-Page je vlastně šablona, která umožňuje v prostředí ASP.NET definovat jednotný vzhled pro vlastní stránky webové aplikace. Může zahrnovat např. i aktivní ovládací prvky, které se potom na každé stránce odvolávající se na příslušnou master-page opakují. V mém případě jsem do Master-Page vložila menu aplikace.

3.5.2.2.Menu aplikace a úvodní stránka

Pro menu aplikace jsem použila stromovou strukturu s jedním z vestavěných designů.

Obrázek 5 – Úvodní stránka

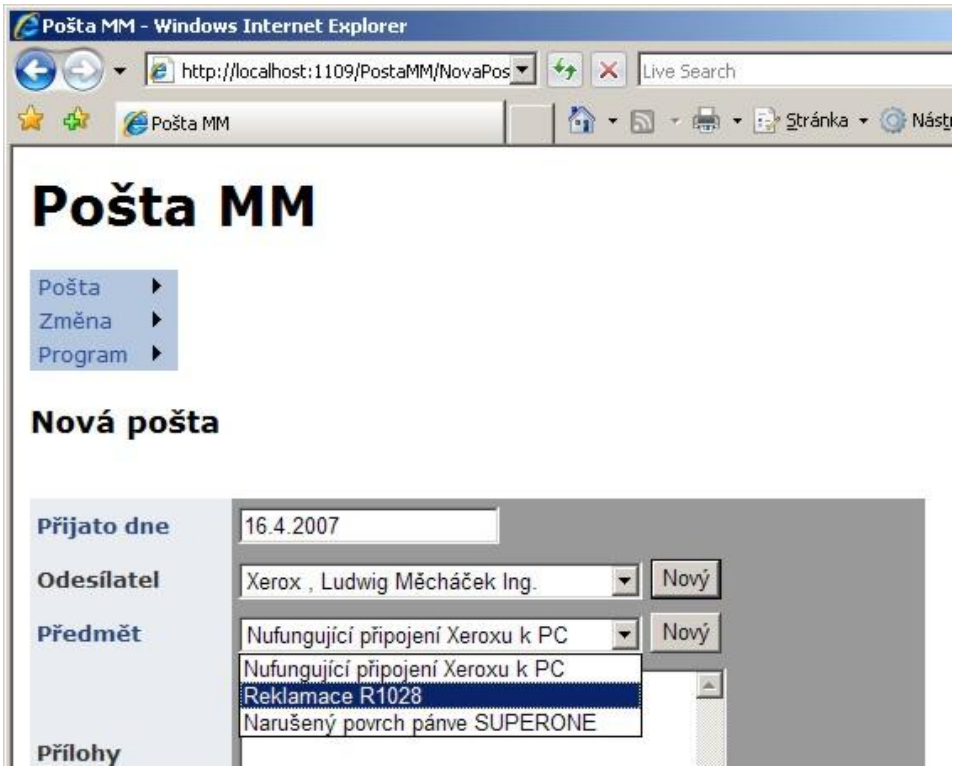
Menu má následující strukturu:

- Pošta
  - Nová Pošta
  - Přehled pošty - k vyřízení
  - Přehled pošty – úplný
- Změna
  - Zaměstnanci
  - Divize
- Program
  - Ukončit
  - O programu

3.5.2.3.Stránka pro zadávání nové pošty

Zdrojový kód (viz příloha 1) je společný pro stránky Zadávání nové pošty i pro Úpravu záznamu. Podle způsobu volání jsou potom některá pole skrytá nebo nemají povolenou editaci.

Výsledný design je pak spojením Master-Page a stránky Nová Pošta.



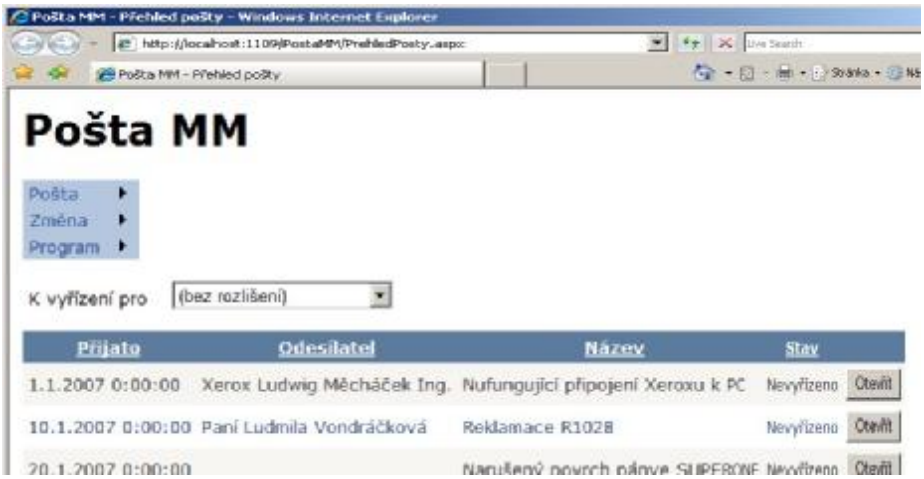
Obrázek 6 - Stránka Nová Pošta

Pro pohodlnější zadávání jsou výběry s číselníků řešeny jako rozbalovací seznamy s možností doplnit položku v případě, že se v seznamu nenachází. Datum přijetí je předvyplněno aktuálním datem a datum doporučeného vyřízení aktuálním datem + 14 dnů.

3.5.2.4.Stránka s přehledem pošty

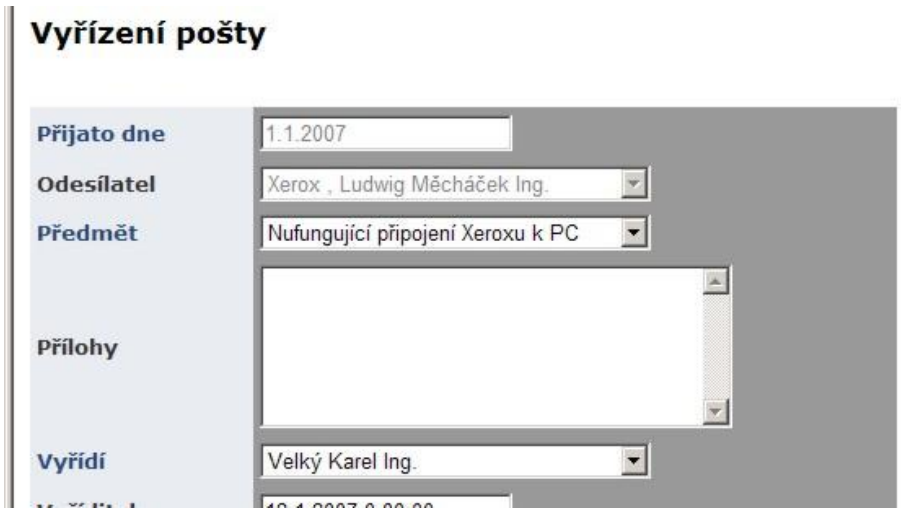
Přehled pošty lze zobrazit úplný nebo jen poštu k vyřízení. Oba seznamy lze třídit podle libovolného kriteria. V případě, že seznam přesáhne 20 položek, je automaticky

rozdělen na jednotlivé stránky. Nad oběma seznamy lze nastavit filtr, který propustí pouze položky určené k vyřízení pro zadaného zaměstnance.



Obrázek 7 - Přehled pošty

Každou položku je možné dále otevřít v detailním zobrazení tlačítkem Otevřít. Toto zobrazení zároveň slouží k vyřízení pošty, příp. k předání k vyřízení jinému zaměstnanci.



Obrázek 8 - Vyřízená pošta

Některé položky jsou na rozdíl od formuláře pro vkládání nové pošty již nepřístupné pro editaci (Přijato dne, Odesílatel), ale např. přiřazení k předmětu je možné změnit, to vyplynulo z provozní potřeby při testování aplikace.

3.5.2.5. *Výstupy*

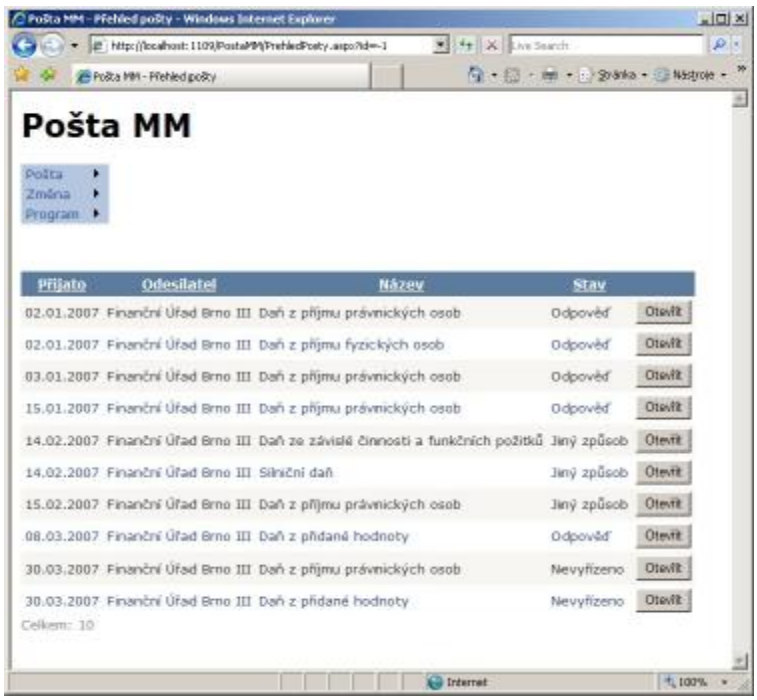
Aplikace umožňuje použití filtrů a také tiskové výstupy.

Druhy filtrů:

- podle odesílatele,
- podle data doručení,
- podle předmětu,
- podle způsobu vyřízení,
- podle zaměstnance, který má poštu vyřídit.

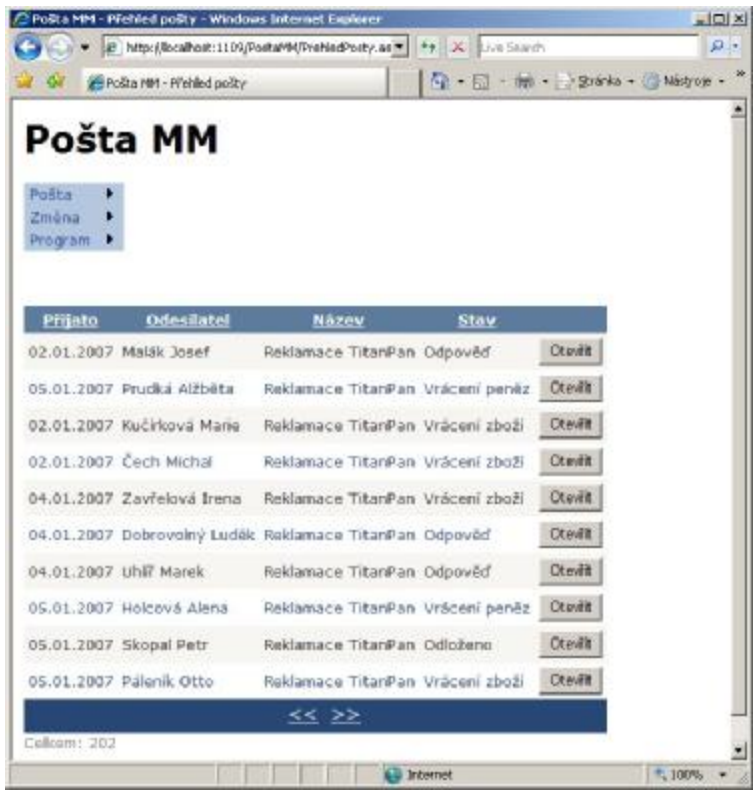
Jednotlivé filtry lze vzájemně kombinovat. Veškeré výstupy na obrazovku lze také tisknout.

Příklady jednotlivých filtrů:



Obrázek 9 - Filtr podle odesílatele

Obrázek 10 - Filtr podle data doručení



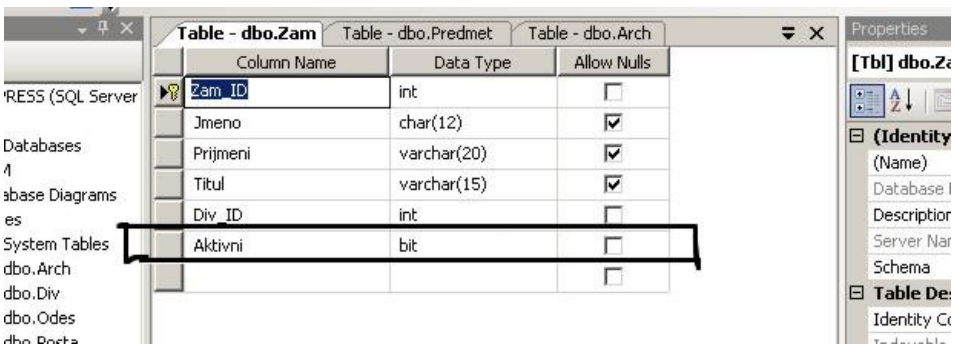
Obrázek 11 - Filtr podle předmětu

3.5.3. Úpravy po zkušebním provozu aplikace

3.5.3.1.Systém oprávnění

Pro aplikaci byl původně navržen systém několika druhů oprávnění, které měly řešit právo pořizovat nové záznamy v došlé poště, označovat poštu za vyřízenou atd. Z provozních důvodů nakonec vyplynulo, že všichni zaměstnanci, kteří mají s došlou poštou nakládat, musí mít povoleno provádět veškeré činnosti. Např. právo k vyřízení musela dostat i sekretářka, která původně poštu jen třídila, protože i ona může v některých specifických případech poštu sama vyřídit. Podobně jako oprávnění předávat poštu k vyřízení, které měli mít pouze vedoucí pracovníci, ale docházelo pouze ke zbytečnému zdržení, bez jiného efektu.

Výsledný stav je takový, že rozlišuje uživatele pouze na ty, kteří mají právo přístupu do systému jako takového a na ty, kteří přístup nemají. Jediným omezením je zavedení atributu Aktivni do tabulky zaměstnanců, který povoluje přiřadit zaměstnanci poštu k vyřízení. Rozlišují se tak zaměstnanci, kteří nemají právo z nějakého důvodu vůbec nebo přechodně vyřizovat poštu.



Obrázek 12 - Systémová oprávnění

3.5.3.2.Ošetření chyb

Zlepšení ošetření případů, kdy by byla narušena integrita databáze, např. když má být odstraněn zaměstnanec, který má přiřazenou poštu k vyřízení. Tyto případy sice řeší relace, ale ošetření bylo zavedeno i do aplikační úrovně.

4. Provoz aplikace

4.1. Aplikace v praxi

Aplikace Pošta MM byla spuštěna do testovacího provozu 20.12.2006. Zaměstnanci byli seznámeni s veškerými funkcemi aplikace a zároveň byli proškoleni tak, aby od 2.1.2007 mohlo začít testování celého systému ve skutečných provozních podmínkách.

Během testovacího provozu byly, jak už bylo výše zmíněno v bodě 3.5.3, doladěny některé funkce a odstraněny drobné nedostatky. Nyní je aplikace plně využívána.

4.2. Archivace

Jeden z požadavků efektivního vyřizování doručené pošty byl vytvořit zcela nový systém archivace již vyřízené pošty. Tento požadavek vznikl z důvodu potřeb návratu ke starší poště a také kvůli přehlednosti historie dlouhodobých korespondencí, např. s finančními úřady, s jednotlivými zákazníky uplatňujícími reklamace apod.

Návrh

1. Veškerá vyřízená pošta bude archivována u sekretářky majitele firmy,
2. jednotlivé dopisy se budou zakládat spolu s odpovědí, popř. jiným způsobem vyřízení, podle identifikačního čísla, které mu bude přiřazeno v aplikaci Pošta MM, do šanonů,
3. jednotlivé šanony budou mít své identifikační číslo.

Tento návrh byl přijat a byla vytvořena další opatření.

Opatření

1. Sekretářka vždy jednou za týden zkontroluje, zda je veškerá vyřízená korespondence odevzdána k archivaci, v opačném případě zajistí odevzdání této korespondence,
2. je zakázáno již archivovanou poštu vytahovat ze šanonů,
3. v případě, že nějaký zaměstnanec přeci jen doručený dopis potřebuje, sekretářka mu vytvoří kopii, originál musí vždy zůstat v šanonu.

5. Závěr

Rekapitulace cílů práce:

1. vytvoření aplikaci pro evidenci doručené pošty,
2. zavedení této aplikace ve firmě,
3. zavedení nového systému zakládání doručených zásilek ,
4. kontrola vyřizování firemní pošty,
5. zefektivnění činnost jednotlivých zaměstnanců,
6. vybudovaní lepších vztahů z dodavateli, odběrateli i konečnými zákazníky.

Body 1 – 4 byly úspěšně splněny. Splnění bodů 5 – 6 se projeví až postupem času, ale již teď víme, že aplikace usnadnila zaměstnancům práci a za 1. čtvrtletí 2007 se nestalo, že by doručená pošta skončila jako nevyřízená.

Věřím, že projekt byl úspěšný a bude dlouho sloužit ke spokojenosti majitele, zaměstnanců i zákazníků.

Literatura

1) *ASP.NET* [on-line]. Poslední úprava 2007-05-02. [citace 2007-05-03]. URL: <http://cs.wikipedia.org/wiki/ASP.NET>
2) AVERY, Janes: *ASP.NET*. 1.vyd. 2003. 200 s. ISBN 80-251-0121-5
3) ESPOSITO, Dino. *ASP.NET a ADO.NET tvorba dynamických webových stránek*. 1. vyd. 2003. 352 s. ISBN 80-247-0474-9
4) GÜRTLER, Martin, KOCICH Pavel. *Visual Basic .NET – Hotová řešení*. 1.vyd. 2005. 312 s. ISBN 80-251-0367-6
5) KOCH, Miloš. *Datové a funkční modelování*. 1.vyd. 2004. 108 s. ISBN 80-214-2724-8
6) LACKO, Luboslav. *SQL Hotová řešení*. 1.vyd. 2003. 296 s. ISBN 80-7226-975-5
7) LÁNSKÝ, Lukáš. *Úvod do ASP.NET* [on-line]. [citace 2007-05-01]. URL: <http://www.pcsvet.cz/art/article.php?id=3632>
8) *Microsoft SQL Server Express I. - základní popis* [on-line]. Poslední úprava 2007-04-28. [citace 2007-05-03]. URL: <http://www.aspdotnet.cz/ASPdotNETcz/tabid/36/articleType/ArticleView/articleId/22/Microsoft-SQL-Server-Express-I--zkladn-popis.aspx>
9) STANEK, William R. *Microsoft SQL Server 2005, Kapesní rádce administrátora*. 1.vyd. 2007. 544 s. ISBN 80-251-1211-X
10) ŠIMŮNEK, Milan. *SQL Kompletní kapesní průvodce*. 1.vyd. 2006. 248 s. ISBN 80-7169-692-7
11) VALÁŠEK, Michal A. *Express vývojové nástroje - .NET 2.0 zdarma* [on-line]. Poslední úprava 2000-2007. [citace 2007-05-01]. URL: <http://www.aspnet.cz/Articles/60-express-vyvojove-nastroje-net-2-0-zdarma.aspx>
12) VŠE Praha, Katedra informačních technologií. *Microsoft Visual Studio 2005* [on-line]. [citace 2007-05-03]. URL: <http://nb.vse.cz/~zelenyj/it380/eseje/xrihj07/vs2005.htm>

13) *Začínáme s ASP.NET 2.0 – 1. díl: úvod, instalace* [on-line]. Poslední úprava 2005-09-06. [citace 2007-05-01]. URL: <http://www.zive.cz/h/Programovani/AR.asp?ARI=125787>

14) *Začínáme s ASP.NET 2.0: 6. díl: práce s daty v ASP.NET I.* [on-line]. Poslední úprava 2005-11-01. [citace 2007-05-01]. URL: <*http*://www.zive.cz/h/Programovani/AR.asp?ARI=126341&CAI=2037>

Seznam diagramů

Seznam obrázků

Seznam schémat

Seznam příloh

```
<%@ Page Language="VB" MasterPageFile="~/MasterPage.master" Title="Pošta MM" %>
<%@ MasterType VirtualPath="~/MasterPage.master" %>

<script runat="server">

    Dim APrijato As Date
    Dim AVyridit As Date
    Dim AId As Integer
    REM Dim ACalPrijato As Calendar
    REM AId = -1
    Dim Nova As Boolean


    Protected Sub Page_Init(ByVal sender As Object, ByVal e As System.EventArgs)

        AId = Request.QueryString("id")

        If AId > 0 Then
            lblTitle.Text = "Vyřízení pošty"
            REM DetailsView1.DefaultMode = DetailsViewMode.Edit
            DetailsView1.ChangeMode(DetailsViewMode.Edit)
            REM AId = 15
            DS_Posta.SelectCommand = "SELECT * FROM [Posta] WHERE Posta_ID=" & AId.ToString
            REM DS_Posta.Select(DataSourceSelectArguments.Empty)
            REM DetailsView1.DataBind()

            Nova = False
        Else
            lblTitle.Text = "Nová pošta"
            REM DetailsView1.DefaultMode = DetailsViewMode.Insert
            REM DetailsView1.ChangeMode(DetailsViewMode.Insert)
            Nova = True
            DetailsView1.Fields.Item(8).Visible = False
            DetailsView1.Fields.Item(9).Visible = False
            DetailsView1.Fields.Item(10).Visible = False
```

```
    End If

    REM New ACalPrijato = TemplateControl.FindControl("CalendarPrijato")


    APrijato = FormatDateTime(Date.Now, 1)
    '... mozna jinak pri editaci existujiciho
    REM CalendarPrijato.SelectedDate = APrijato
    RefreshDVyriz()

End Sub

Protected Sub RefreshDVyriz()
    AVyridit = DateAdd("d", 14, APrijato)
End Sub


Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
    REM DetailsView1.InsertItem(False)
    REM DetailsView1.EmptyDataTemplate
End Sub

Protected Sub CalendarPrijato_SelectionChanged(ByVal sender As Object, ByVal e As System.EventArgs)

    REM txtPrijato.Text = CalendarPrijato.SelectedDate
    REM CalendarPrijato.Visible = False
    REM RefreshDVyriz()
End Sub

Protected Sub btnPrijato_Click(ByVal sender As Object, ByVal e As System.EventArgs)

    REM CType(DetailsView1.TemplateControl.FindControl("CalendarPrijato"), Calendar).Visible = True

    With CType(DetailsView1.TemplateControl.FindControl("CalendarPrijato"), Calendar)
```

```
        .Visible = Not .Visible
    End With

End Sub

Protected Sub txtPrijato_Load(ByVal sender As Object, ByVal e As System.EventArgs)
    CType(sender, TextBox).Text = APrijato
End Sub


Protected Sub CalendarPrijato_Load(ByVal sender As Object, ByVal e As System.EventArgs)
    CType(sender, Calendar).SelectedDate = APrijato
End Sub

Protected Sub Button3_Click(ByVal sender As Object, ByVal e As System.EventArgs)
    DS_Posta.Insert()
End Sub

Protected Sub DS_Posta_Inserting(ByVal sender As Object, ByVal e As System.Web.UI.WebControls.SqlDataSourceCommandEventArgs)
    REM e.Command.Parameters("@Odes_ID").Value = CType(DetailsView1.FooterRow.FindControl("Odes_ID"), DropDownList).SelectedValue
End Sub

Protected Sub DropDownList4_Load(ByVal sender As Object, ByVal e As System.EventArgs)
    REM CType(sender, DropDownList).Visible = Not Nova
End Sub

Protected Sub TextBox2_Init(ByVal sender As Object, ByVal e As System.EventArgs)
    CType(sender, TextBox).Text = FormatDateTime(DateAdd(DateInterval.Day, 14, Now.Date), 2)
End Sub

Protected Sub TextBox1_Init(ByVal sender As Object, ByVal e As System.EventArgs)
    CType(sender, TextBox).Text = FormatDateTime(Now.Date.ToString, 2)
End Sub

Protected Sub Button3_Click1(ByVal sender As Object, ByVal e As System.EventArgs)
```

```
        If AId > 0 Then
            DetailsView1.UpdateItem(False)
        Else
            DetailsView1.InsertItem(False)
        End If
    End Sub

    Protected Sub TextBox1_DataBinding(ByVal sender As Object, ByVal e As System.EventArgs)
        CType(sender, TextBox).Text = FormatDateTime(CType(sender, TextBox).Text, 2)
    End Sub

    Protected Sub DetailsView1_ItemUpdated(ByVal sender As Object, ByVal e As System.Web.UI.WebControls.DetailsViewUpdatedEventArgs)
        Response.Redirect("PrehledPosty.aspx?id=0")
    End Sub

    Protected Sub btnNovyPredmet_Click(ByVal sender As Object, ByVal e As System.EventArgs)
        Response.Redirect("NovyPredmet.aspx")
    End Sub

    Protected Sub Button1_Click(ByVal sender As Object, ByVal e As System.EventArgs)
        Response.Redirect("NovyOdesilatel.aspx")
    End Sub

    Protected Sub DS_Posta_Updating(ByVal sender As Object, ByVal e As System.Web.UI.WebControls.SqlDataSourceCommandEventArgs)
        REM If e.Command.Parameters(6).Value = 0 Then e.Command.Parameters(6).Value = System.DBNull.Value
    End Sub

</script>

<asp:Content ID="Content1" ContentPlaceHolderID="ContentPlaceHolder1" Runat="Server">
    <h3>
        <asp:Label ID="lblTitle" runat="server" Text="Nová pošta"></asp:Label> </h3>
    <br />
    <asp:DetailsView ID="DetailsView1" runat="server" AutoGenerateRows="False" CellPadding="4"
        DataKeyNames="Posta_ID" DataSourceID="DS_Posta" ForeColor="#333333" GridLines="None"
```

```
Height="50px" Width="532px" DefaultMode="Insert" OnItemUpdated="DetailsView1_ItemUpdated">
<FooterStyle BackColor="#5D7B9D" Font-Bold="True" ForeColor="White" />
<CommandRowStyle BackColor="#E2DED6" Font-Bold="True" />
<EditRowStyle BackColor="#999999" />
<RowStyle BackColor="#F7F6F3" ForeColor="#333333" />
<PagerStyle BackColor="#284775" ForeColor="White" HorizontalAlign="Center" />
<Fields>
    <asp:BoundField DataField="Posta_ID" HeaderText="Posta_ID" InsertVisible="False"
        ReadOnly="True" SortExpression="Posta_ID" Visible="False" />
    <asp:TemplateField HeaderText="Přijato dne" SortExpression="Prijato_DATE">
        <EditItemTemplate>
            <asp:TextBox ID="TextBox1" runat="server" Enabled="False" OnDataBinding="TextBox1_DataBinding"
                OnInit="TextBox1_Init" ReadOnly="True" Text='<%# Bind("Prijato_DATE") %>'></asp:TextBox> 
        </EditItemTemplate>
        <InsertItemTemplate>
            <asp:TextBox ID="TextBox1" runat="server" Text='<%# Bind("Prijato_DATE") %>' OnInit="TextBox1_Init"></asp:TextBox>
        </InsertItemTemplate>
        <ItemTemplate>
            <asp:Label ID="Label7" runat="server" Text='<%# Bind("Prijato_DATE") %>'></asp:Label>
        </ItemTemplate>
    </asp:TemplateField>
    <asp:TemplateField HeaderText="Odes&#237;latel" SortExpression="Odes_ID">
        <EditItemTemplate><asp:DropDownList ID="DropDownList1" runat="server" DataSourceID="DS_Odes" DataTextField="Odesilatel"
                DataValueField="Odes_ID" Width="240px" SelectedValue='<%# Bind("Odes_ID") %>' Enabled="False" EnableTheming="True">
        </asp:DropDownList>
        </EditItemTemplate>
        <InsertItemTemplate>
            <asp:DropDownList ID="DropDownList1" runat="server" DataSourceID="DS_Odes" DataTextField="Odesilatel"
                DataValueField="Odes_ID" Width="240px" SelectedValue='<%# Bind("Odes_ID") %>'>
            </asp:DropDownList>
            <asp:Button ID="Button1" runat="server" Text="Nový" OnClick="Button1_Click" />
        </InsertItemTemplate>
        <ItemTemplate>
            <asp:Label ID="Label2" runat="server" Text='<%# Bind("Odes_ID") %>'></asp:Label>
        </ItemTemplate>
```

```
</asp:TemplateField>

<asp:TemplateField HeaderText="Předmět" SortExpression="Predmet_ID">
    <EditItemTemplate>
        <asp:DropDownList ID="DropDownList3" runat="server" DataSourceID="DS_predmet" DataTextField="Nazev"
            DataValueField="Predmet_ID" Width="240px" SelectedValue='<%# Bind("Predmet_ID") %>'>
        </asp:DropDownList>
    </EditItemTemplate>
    <InsertItemTemplate>
        <asp:DropDownList ID="DropDownList3" runat="server" DataSourceID="DS_predmet" DataTextField="Nazev"
            DataValueField="Predmet_ID" Width="240px" SelectedValue='<%# Bind("Predmet_ID") %>'>
        </asp:DropDownList>
        <asp:Button ID="btnNovyPredmet" runat="server" Text="Nový" OnClick="btnNovyPredmet_Click" />
    </InsertItemTemplate>
    <ItemTemplate>
        <asp:Label ID="Label3" runat="server" Text='<%# Bind("Predmet_ID") %>'></asp:Label>
    </ItemTemplate>
</asp:TemplateField>
<asp:TemplateField HeaderText="Př&#237;lohy" SortExpression="Prilohy">
    <EditItemTemplate>
        <asp:TextBox ID="TextBox3" runat="server" Height="94px" ReadOnly="True" Text='<%# Bind("Prilohy") %>'
            TextMode="MultiLine" Width="284px"></asp:TextBox>
    </EditItemTemplate>
    <InsertItemTemplate>
        <asp:TextBox ID="TextBox3" runat="server" Height="94px" Text='<%# Bind("Prilohy") %>'
            TextMode="MultiLine" Width="284px"></asp:TextBox>
    </InsertItemTemplate>
</asp:TemplateField>
<asp:TemplateField HeaderText="Vyř&#237;d&#237;" SortExpression="Zam_ID">
    <EditItemTemplate>
        <asp:DropDownList ID="DropDownList6" runat="server" DataSourceID="DS_Zam" DataTextField="Column1"
            DataValueField="Zam_ID" Width="240px" SelectedValue='<%# Bind("Zam_ID") %>'>
        </asp:DropDownList>
    </EditItemTemplate>
    <InsertItemTemplate>
```

```
        <asp:DropDownList ID="DropDownList6" runat="server" DataSourceID="DS_Zam" DataTextField="Column1"
            DataValueField="Zam_ID" Width="240px" SelectedValue='<%# Bind("Zam_ID") %>'>
        </asp:DropDownList>
    </InsertItemTemplate>
    <ItemTemplate>
        <asp:Label ID="Label4" runat="server" Text='<%# Bind("Zam_ID") %>'></asp:Label>
    </ItemTemplate>
</asp:TemplateField>
<asp:TemplateField HeaderText="Vyř&#237;dit do" SortExpression="DVyriz_DATE">
    <EditItemTemplate>
        <asp:TextBox ID="TextBox7" runat="server" Text='<%# Bind("DVyriz_DATE") %>'></asp:TextBox>
    </EditItemTemplate>
    <InsertItemTemplate>
        <asp:TextBox ID="TextBox2" runat="server" OnInit="TextBox2_Init" Text='<%# Bind("DVyriz_DATE") %>'></asp:TextBox>
    </InsertItemTemplate>
    <ItemTemplate>
        <asp:Label ID="Label8" runat="server" Text='<%# Bind("DVyriz_DATE") %>'></asp:Label>
    </ItemTemplate>
</asp:TemplateField>
<asp:TemplateField HeaderText="Priorita" SortExpression="Prior_ID">
    <EditItemTemplate><asp:DropDownList ID="DropDownList2" runat="server" DataSourceID="DS_Prior" DataTextField="Popis"
            DataValueField="Prior_ID" Width="240px" SelectedValue='<%# Bind("Prior_ID") %>'>
    </asp:DropDownList>
    </EditItemTemplate>
    <InsertItemTemplate>
        <asp:DropDownList ID="DropDownList2" runat="server" DataSourceID="DS_Prior" DataTextField="Popis"
            DataValueField="Prior_ID" Width="240px" SelectedValue='<%# Bind("Prior_ID") %>'>
        </asp:DropDownList>
    </InsertItemTemplate>
    <ItemTemplate>
        <asp:Label ID="Label1" runat="server" Text='<%# Bind("Prior_ID") %>'></asp:Label>
    </ItemTemplate>
</asp:TemplateField>
<asp:BoundField DataField="Vyriz_DATE" HeaderText="Vyř&#237;zeno dne" SortExpression="Vyriz_DATE" />
<asp:TemplateField HeaderText="Stav" SortExpression="Vyriz_ID">
```

```
                <EditItemTemplate>
                    <asp:DropDownList ID="DropDownList5" runat="server" DataSourceID="DS_Vyriz" DataTextField="Popis"
                        DataValueField="Vyriz_ID" Width="240px" SelectedValue='<%# Bind("Vyriz_ID") %>'>
                    </asp:DropDownList>
                </EditItemTemplate>
                <InsertItemTemplate>
                    <asp:DropDownList ID="DropDownList5" runat="server" DataSourceID="DS_Vyriz" DataTextField="Popis"
                        DataValueField="Vyriz_ID" Width="240px" SelectedValue='<%# Bind("Vyriz_ID") %>'>
                    </asp:DropDownList>
                </InsertItemTemplate>
                <ItemTemplate>
                    <asp:Label ID="Label5" runat="server" Text='<%# Bind("Vyriz_ID") %>'></asp:Label>
                </ItemTemplate>
            </asp:TemplateField>
            <asp:TemplateField HeaderText="Archivov&#225;no" SortExpression="Arch_ID">
                <EditItemTemplate>
                    <asp:DropDownList ID="DropDownList44" runat="server" DataSourceID="DS_Arch"
                        DataTextField="Column1"    DataValueField="Arch_ID"    Width="240px"    SelectedValue='<%#    Bind("Arch_ID")    %>'
OnLoad="DropDownList4_Load">
                    </asp:DropDownList>
                </EditItemTemplate>
                <InsertItemTemplate>
                    <asp:DropDownList ID="DropDownList4" runat="server" DataSourceID="DS_Arch"
                        DataTextField="Column1"    DataValueField="Arch_ID"    Width="240px"    SelectedValue='<%#    Bind("Arch_ID")    %>'
OnLoad="DropDownList4_Load">
                    </asp:DropDownList>
                </InsertItemTemplate>
                <ItemTemplate>
                    <asp:Label ID="Label6" runat="server" Text='<%# Bind("Arch_ID") %>'></asp:Label>
                </ItemTemplate>
            </asp:TemplateField>
        </Fields>
        <FieldHeaderStyle BackColor="#E9ECF1" Font-Bold="True" />
        <HeaderStyle BackColor="#5D7B9D" Font-Bold="True" ForeColor="White" />
        <AlternatingRowStyle BackColor="White" ForeColor="#284775" />
```

```
        <FooterTemplate>
            <asp:Button ID="Button3" runat="server" OnClick="Button3_Click1" Text="Zapsat" />
        </FooterTemplate>
    </asp:DetailsView>
    <br />
    <asp:SqlDataSource ID="DS_Posta" runat="server" ConnectionString="<%$ ConnectionStrings:PostaMMConnectionString %>"
        DeleteCommand="DELETE FROM [Posta] WHERE [Posta_ID] = @Posta_ID" InsertCommand="INSERT INTO [Posta] ([Prijato_DATE], [Odes_ID], [Predmet_ID], [Zam_ID], [Prior_ID], [DVyriz_DATE], [Vyriz_ID]) VALUES (@Prijato_DATE, @Odes_ID, @Predmet_ID, @Zam_ID, @Prior_ID, @DVyriz_DATE,1)"
        SelectCommand="SELECT * FROM [Posta]" UpdateCommand="UPDATE [Posta] SET [Predmet_ID] = @Predmet_ID, [Zam_ID] = @Zam_ID, [Prior_ID] = @Prior_ID, [DVyriz_DATE] = @DVyriz_DATE, [Vyriz_DATE] = @Vyriz_DATE, [Vyriz_ID] = @Vyriz_ID, [Arch_ID] = @Arch_ID WHERE [Posta_ID] = @Posta_ID" OnInserting="DS_Posta_Inserting" OnUpdating="DS_Posta_Updating">
        <DeleteParameters>
            <asp:Parameter Name="Posta_ID" Type="Int32" />
        </DeleteParameters>
        <UpdateParameters>
            <asp:Parameter Name="Predmet_ID" Type="Int32" />
            <asp:Parameter Name="Zam_ID" Type="Int32" />
            <asp:Parameter Name="Prior_ID" Type="Int32" />
            <asp:Parameter Name="DVyriz_DATE" Type="DateTime" />
            <asp:Parameter Name="Vyriz_DATE" Type="DateTime" />
            <asp:Parameter Name="Vyriz_ID" Type="Int32" />
            <asp:Parameter Name="Arch_ID" Type="Int32" />
            <asp:Parameter Name="Posta_ID" Type="Int32" />
        </UpdateParameters>
        <InsertParameters>
            <asp:Parameter Name="Prijato_DATE" Type="DateTime" />
            <asp:Parameter Name="Odes_ID" Type="Int32" />
            <asp:Parameter Name="Predmet_ID" Type="Int32" />
            <asp:Parameter Name="Zam_ID" Type="Int32" />
            <asp:Parameter Name="Prior_ID" Type="Int32" />
            <asp:Parameter Name="DVyriz_DATE" Type="DateTime" />
        </InsertParameters>
    </asp:SqlDataSource>
    <asp:SqlDataSource ID="DS_Prior" runat="server" ConnectionString="<%$ ConnectionStrings:PostaMMConnectionString %>"
```

```
        SelectCommand="SELECT * FROM [Prior]"></asp:SqlDataSource>
    <asp:SqlDataSource ID="DS_Odes" runat="server" ConnectionString="<%$ ConnectionStrings:PostaMMConnectionString %>"
        SelectCommand="SELECT Odes_ID, Firma+', '+Prijmeni+' '+Jmeno+' '+Titul AS Odesilatel FROM Odes"></asp:SqlDataSource>
    <asp:SqlDataSource ID="DS_predmet" runat="server" ConnectionString="<%$ ConnectionStrings:PostaMMConnectionString %>"
        SelectCommand="SELECT [Predmet_ID], [Nazev] FROM [Predmet]"></asp:SqlDataSource>
    <asp:SqlDataSource ID="DS_Arch" runat="server" ConnectionString="<%$ ConnectionStrings:PostaMMConnectionString %>"
        SelectCommand="SELECT a.Arch_ID, d.Divize+' / '+a.Umisteni FROM Arch a&#13;&#10;LEFT OUTER JOIN Div d ON a.Div_ID=d.Div_ID">
    </asp:SqlDataSource>
    <asp:SqlDataSource ID="DS_Zam" runat="server" ConnectionString="<%$ ConnectionStrings:PostaMMConnectionString %>"
        SelectCommand="SELECT Zam_ID, Prijmeni+' '+Jmeno+' '+Titul FROM Zam"></asp:SqlDataSource>
    <asp:SqlDataSource ID="DS_Vyriz" runat="server" ConnectionString="<%$ ConnectionStrings:PostaMMConnectionString %>"
        SelectCommand="SELECT [Vyriz_ID], [Popis] FROM [Vyriz]"></asp:SqlDataSource>
    <br />
</asp:Content>
```