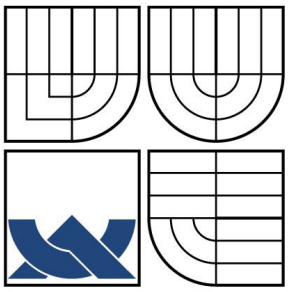
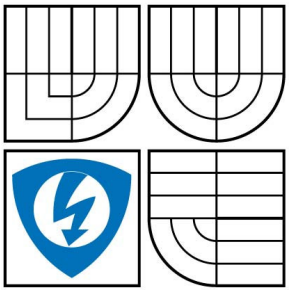


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

APLIKACE PRO SPRÁVU OBSAHU INTERNETOVÝCH STRÁNEK

APPLICATION FOR ADMINISTRATION OF CONTENT INTERNET PAGES.

DIPLOMOVÁ PRÁCE
MASTER´S THESIS

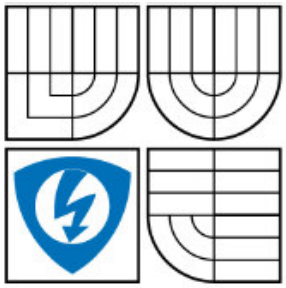
AUTOR PRÁCE — Bc. Jiří KAPLAN

AUTHOR

VEDOUCÍ PRÁCE — Ing. Jan KACÁLEK

SUPERVISOR

BRNO 2008



VYSOKE UCENI
TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Kaplan Jiří Bc.	***ID:***	54062
Ročník:	2	***Akademický rok:***	2007/2008

NÁZEV TÉMATU:

Aplikace pro správu obsahu internetových stránek

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s jazykem C# a prostředím .NET. Diskutujte možnosti využití .NET prostředí pro vytvoření aplikace pro spravu internetových stránek. Podle zíkaných informací navhněte strukturu databáze, které by tuto správu umožňovaly a tuto aplikaci realizujte.

DOPORUČENÁ LITERATURA:

[1] C# 2005 velká kniha řešení, Computer Press 2007, ISBN 978-80-251-1620-3
[2] Troelsen Andrew, C# a .NET 2.0 profesionálně, ZONER Press 2006, ISBN 80-86815-42-0

Termín zadání:	11.2.2008	***Termín odevzdání:***	28.5.2008

Vedoucí práce: Ing. Jan Kacálek

prof. Ing. Kamil Vrba, CSc.
předseda oborové rady

LICENČNÍ SMLOUVA

POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami:

1. Pan/paní

Jméno a příjmení: Jiří Kaplan

Bytem: Luční 2600/68, 61600, Brno - Žabovresky

Narozen/a (datum a místo): 28.6.1984 v Brně

(dále jen „autor“)

a

2. Vysoké učení technické v Brně

Fakulta elektrotechniky a komunikačních technologií

se sídlem Údolní 244/53, 602 00, Brno

jejímž jménem jedná na základě písemného pověření děkanem fakulty:

prof. Ing. Kamil Vrba, CSc.

(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:	Aplikace pro správu internetových stránek
Vedoucí/ školitel VŠKP:	Ing. Jan Kacálek
Ústav:	Telekomunikací
Datum obhajoby VŠKP:	

VŠKP odevzdal autor nabyvateli v*:

□ tištěné formě – počet exemplářů 2

□ elektronické formě – počet exemplářů 2

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

* hodící se zaškrtněte

(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: 28.5.2008

…….. ……

Nabyvatel Autor

Anotace:

Tato diplomová práce je zaměřena na popis prostředí .NET Framework, jeho verzí, architektury, assembly a jejich formátu. Popisuje programovací jazyk C#, jeho podporované programovací postupy a atributy. Obsahuje popis ASP.NET, jeho charakteristiku, používané adresáře, ovládací prvky a návrhový vzor Model-View-Controller. Dále popisuje vhodnost .NET prostředí pro tvorbu aplikací a porovnání s PHP. Krátce seznamuje s ADO.NET, vázáním dat, databázemi, jejich typy a jazykem SQL a redakčním systémem. Na konci je popsána struktura databáze navrhnutého redakčního systému, administrace uživatelů a jeho části a funkce.

Klíčová slova:

.NET Framework, ASP.NET, C#, databáze, SQL, CMS

Abstract:

This master's thesis is focused on description of .NET Framework, versions, architecture, assembly and data formats. Describes programming language C#, his supporting programming processes and architecture. Contains description of ASP.NET, characteristics, used directories, control items and design model Model-View-Controller. Next part describes usability .NET background for creation of aplication and comparsion with PHP. Shortly introduces with ADO.NET, databinding, databases, his types and SQL language and content management system. At the end is desribed database structure of drafted content management system, user administration and his parts and functions.

Keywords:

.NET Framework, ASP.NET, C#, database, SQL, CMS

PROHLÁŠENÍ:

Prohlašuji, že svou diplomovou práci na téma Aplikace pro správu obsahu internetových stránek jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce. Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.“

V Brně dne 28.5.2008 ……..

(podpis autora)

PODĚKOVÁNÍ:

Děkuji vedoucímu diplomové práce Ing. Janu Kacálkovi, za velmi užitečnou metodickou pomoc a cenné rady při zpracování diplomové práce.

V Brně dne 28.5.2008 ……..

(podpis autora)

Obsah

Seznam obrázků

1. Úvod

Tato diplomová práce čtenáře seznamuje s prostředím .NET Frameworku, jeho verzemi, architekturou a jazykem C#. Jsou v ní popsány výhody a nevýhody .NET a porovnání s konkurentem PHP, dále obsahuje popis databází a jazyka SQL.V závěru je popsán hlavní cíl této práce a to návrh struktury databáze redakčního systému a jeho realizace spolu s popisem jeho funkcí a možností. Redakční systém neboli CMS (Content Management Systém) je výkonný nástroj, který velmi zjednodušuje administraci webových stránek. Uživatel nemusí mít žádné znalosti HTML, respektive nemusí vůbec nic vědět o programování webu. To otevírá dveře k tvorbě webových stránek tisícům uživatelů. Důkazem toho jsou například blogy, kterých je jen na českém internetu bez počet.

2. .NET Framework

.NET Framework je moderní platforma, která zjednodušuje vývoj aplikací v prostředí internetu. Platforma může být do operačního systému Microsoft Windows přidána, nebo v případě novějšího operačního systému je v něm již obsažena. Následuje výčet hlavních vlastností poskytovaných platformou .NET Framework.

Úplná interoperabilita s existujícím kódem – Existující binární jednotky COM se mohou kombinovat s jednotkami .NET a obráceně. Dále je možno volat z kódu .NET knihovny založené na C. To je tzv. Pinvoke.
Kompletní integrace jazyků – Podpora dědění, zpracování výjimek a ladění přes jazyky.
Společný engine pro všechny .NET jazyky – Každý jazyk rozumí definované sadě typů.
Knihovna základních tříd – Poskytuje jednotný objektový model, který používají všechny .NET jazyky a ochraňuje před voláním API v surovém stavu.
Bezpečnost - .NET povoluje běh kódu s různým stupněm důvěry bez použití sandbox.

2.1 Verze .NET Framework

.NET Framework byl vydáván v následujících verzích.

Verze 1.0
První verze .NET Framework. Uvolněna 13. 2. 2002, dostupná pro Windows 98, NT 4.0, 2000 a XP.

Verze 1.1
První významný upgrade. Publikován 3. 4. 2003. Distribuován jako samostatný balík nebo jako část MS Visual Studio .NET 2003. Toto je první verze obsažená přímo v operačním systému a to MS Windows server 2003. Změny obsažené v této verzi jsou:

- Vestavěná podpora pro mobilní ASP.NET. -dříve pouze jako doplněk.
- Změny v bezpečnosti.
- Vestavěná podpora pro ODBC a Oracle databáze.

- .NET Compact Framework. Verze .NET Frameworku pro malé zařízení.
- IPv6
- Změny v API

Verze 2.0

Uvolněno s Visual Studio .NET 2005, MS SQL Server 2005 a BizTalk 2006. Jako samostatný balíček publikováno 22. 1. 2006. Poslední verze s podporou pro MS Windows 2000. Změny oproti verzi 1.1:

- Změny v API
- Podpora 64 bitové platformy
- Nově hostující API pro původní aplikace přející si hostovat instanci .NET runtimu?
- Nová kontrola dat s deklarovaným vázáním dat
- Nový vzhled pro ASP.NET podporující skiny či themes
- .NET Micro Framework – verze .NET Framework pro domácí elektroniku
- Vylepšení ovládacích prvků ASP.NET

Verze 3.0

Tato verze je integrovaná ve Windows Vista a Windows Server 2008. Může být přidána i do Windows XP a Windows Server 2003. Obsahuje čtyři nové významné komponenty.

Windows Presentation Foundation

WPF – grafický subsystém přímo příbuzný s XAML. Zprostředkovává konzistentní programovací model pro tvorbu aplikací a čistě odděluje uživatelské prostředí a algoritmy obsluhující výměnu informací mezi databází a uživatelským rozhraním. Může být umístěn na desktopu nebo hostován webovým prohlížečem.

Windows Communication Foundation

WCF – komunikační subsystém k podpoře spojování aplikací na jednom nebo několika sítí spojenými počítači.

Windows Workflow Foundation

WF – Technologie pro definování, spouštění a řízení organizace práce.

Windows CardSpace
Softwarová komponenta, která bezpečně ukládá digitální identitu a zprostředkovává jednotné rozhraní pro výběr identity pro jednotlivé operace jako třeba přihlašování na webové stránce.

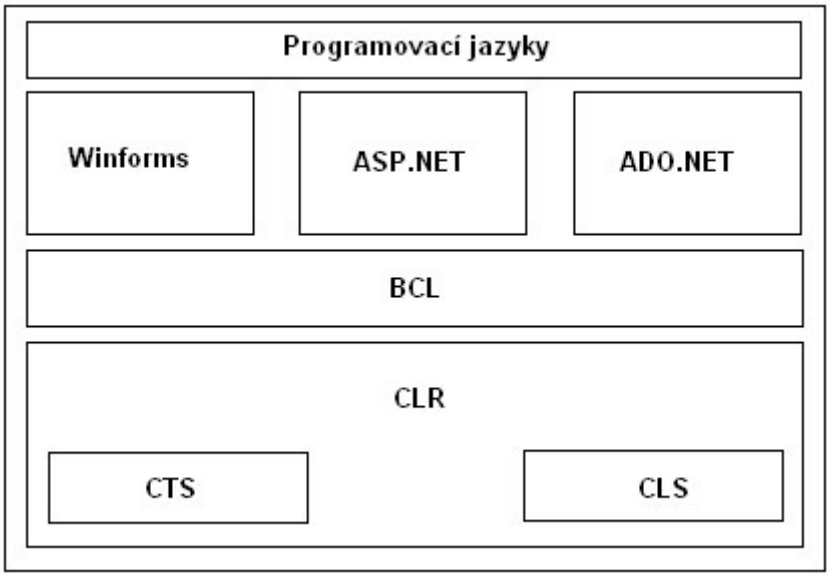
Verze 3.5
Oficiálně uvolněna 19. listopadu 2007. Používá CLR verze 2.0 SP1, které přidává několik vlastností a metod pro podporu prvků jako např. LINQ. Tyto změny nemají vliv na aplikace napsané pro verzi 2.0. Plná podpora pro .NET Framework 3.5 je obsažena ve Visual Studiu 2008. Zdrojový kód BCL je uvolněn v licenci Microsoftu. S touto verzí je uvolněn i nový .NET Compact Framework zprostředkovávající LINQ, http kompresi a zlepšuje podporu audia pro mobilní zařízení. Následují některé další změny:

- Nové vlastnosti jazyka v C# 3.0 a VB.NET 9.0 kompileru
- Anonymní typy se statickým rozhraním
- LINQ (Language Integrated Query - komponenta přidávající schopnost nativně se dotazovat na data do .NET jazyků, užívající syntaxi připomínající SQL)
- Podpora stránkování pro ADO.NET
- V ADO.NET synchronizační API pro synchronizaci lokální cache a paměti serveru

2.2 Architektura .NET

Základem platformy .NET Framework je společný jazykový modul (Common Language Runtime – CLR) . Důvodem jeho vytvoření bylo zjednodušení vývoje aplikací, vytvoření bezpečného prostředí a podpora více programovacích jazyků. Slouží jako výkonové jádro pro řízené aplikace. Spravuje spouštění kódu a poskytuje základní služby, stará se o nízkoúrovňové operace jako např. správa paměti. Nad tímto modulem stojí knihovna základních tříd (Basic Class Library – BCL), která obsahuje řadu funkcí pro práci se soubory, sítí, bezpečností, službami aj. Následující vrstvu tvoří Windows Forms, ASP.NET a ADO.NET. Windows Forms zprostředkovává přístup k přirozeným Windows rozhraním obalením existujícího Windows API do kódu, který se spouští pod správou CLR. Pro tvorbu dynamických webových aplikací, webových stránek a XML webových služeb je ASP.NET. ADO.NET slouží pro přístup k datům a datovým službám. Na úplném vrcholu struktury se

nachází jazyky. (Obr. 1.) Na každou úlohu si pak v .NET Framework můžeme vybrat libovolný podporovaný jazyk, který nám nejvíce vyhovuje. Na výběr máme např. Visual Bacis, C++, C#, Python, Pearl atd.



Obr. 1: Architektura .NET Framework 2.0

CLI

Common Language Infrastructure je otevřená specifikace vyvinutá Microsoftem, která popisuje spustitelný kód a běhové prostředí jádra .NET Framework. Definuje prostředí, které dovoluje rozmanité vysokoúrovňové jazyky používané na různých platformách bez předchozího přepsání pro danou platformu. Implementace CLI Microsoftem se jmenuje CLR.
CLR se skládá ze společného systému typů (CTS – Common Type System) a společné specifikace jazyků (CLS – Common Language Specification). Společný systém typů popisuje datové typy a programovací konstrukce, které CLR podporuje. Specifikuje komunikaci entit a podrobnosti o její reprezentaci ve formátu metadat . NET.

CLS

Definuje podmnožinu typů a programových konstrukcí, na kterých se mohou dohodnout všechny programovací jazyky .NET

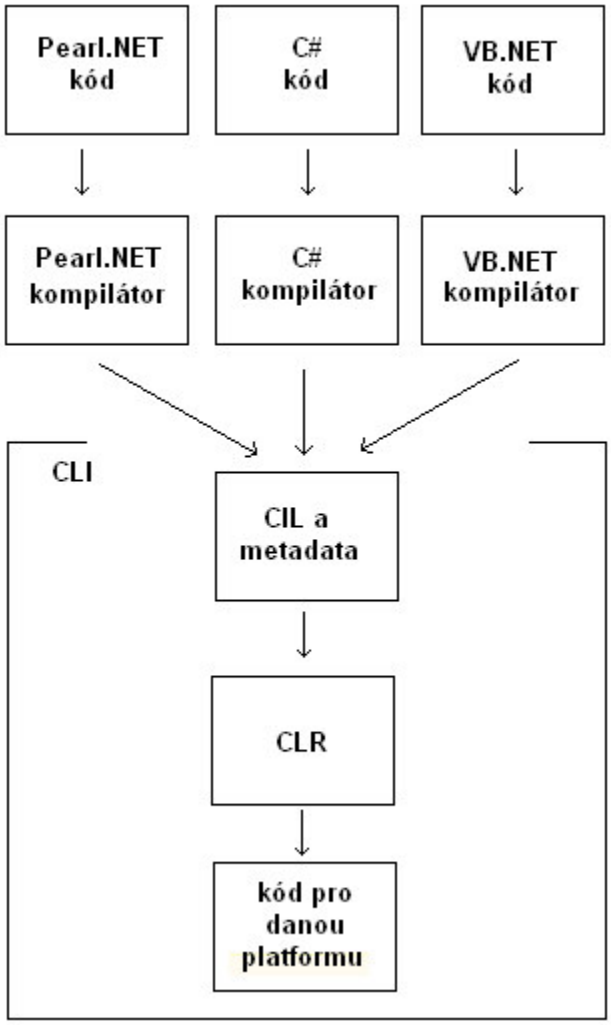
BCL

Knihovna základních tříd je dostupná pro všechny programovací jazyky .NET. Zapouzdřuje všemožné základní věci. Těmi jsou např. vlákna, grafická realizace, práce se soubory, komunikace s hardwarem, interakci s databázemi, manipulaci s XML dokumenty

2.3. Assembly .NET

Kompilátory platformy .NET Framework produkují pouze řízené moduly, které požadují CLR ke svému běhu. CLR ovšem nepracuje přímo s řízenými moduly, ale pracuje s assembly. Assembly je logicky uspořádaná skupina řízených modulů, nebo resource modulů. Dále je Assembly nejmenší jednotkou, která se používá, chrání a čísluje verzemi. (Fajfr, 2006).

I přes to, že binární jednotky .NET mají stejnou příponu jako COM a binární jednotky Win32 (.dll nebo .exe), nejsou si obsahem podobné. Binární jednotky .dll .NET neexportují metody pro usnadnění komunikace s runtimem COM. Nepopisují se pomocí typových knihoven COM a neregistrují se v systémovém registru. Nejdůležitější je však to, že binární jednotky .NET obsahují intermediální jazyk CIL a metadata typů a neobsahují instrukce specifické pro platformy. Z toho vyplývá, že se musí CIL následně zkompilovat ještě pro danou platformu (Obr. 2.). To se provádí tzv. kompilátorem JIT (Just In Time – za pochodu). Každé CPU je vybavené kompilátorem JIT a je pro něj optimalizováno. Např. na kapesních PC je kompilátor vybaven pro běh v prostředí s malou pamětí. Dále je zkompilovaný strojový kód uchován v paměti vhodným způsobem pro danou platformu pro další použití. Při dalším volání se tedy již nemusí kompilovat.



Obr. 2: Assembly

2.3.1. Formát assembly .NET

Assembly .NET se skládá z následujících prvků.

Záhlaví Win32

Toto záhlaví ustanovuje systémům Windows možnost načítat a manipulovat s assembly. Data záhlaví identifikují druh aplikace (konzolová, s GUI, knihovna), jejímž hostitelem mají být Windows.

Záhlaví CLR

Záhlaví CLR je blok dat, jenž musejí podporovat všechny soubory .NET, aby mohly být hostovány CLR. Záhlaví definuje přepínače umožňující runtimu pochopit rozvržení řízeného souboru.

Kód CIL

Kód CIL obsažený v jádru se kompiluje za pochodu do instrukcí, které jsou specifické pro platformu a CPU.

Metadata

Popisují všechny typy (třídy, struktury atp.) a také všechny členy typů (metody, vlastnosti atp.). Obsahuje i externí typy, na něž odkazuje. Toto je klíčový prvek platformy .NET. Pomocí metadat runtime umisťuje typy uvnitř jednotky a rozvrhuje je v paměti

Manifest

Popisuje samotnou assembly. Dokumentuje všechny moduly, všechny externí assembly, na které se aktuálně odkazuje a zřizuje verzi assembly.

2.4. Reflexe

Jedním z důsledků použití mechanismu metadat pro všechny typy v prostředí .NET frameworku je možnost tyto typy v našem programu prozkoumávat a tato věc je nazývána reflexe. Jmenný prostor, který obsahuje nemalý počet tříd, jež námi mohou být použity pro manipulaci s danými elementy konkrétní aplikace, nese název System.Reflection.(Puš, 2005) Je možno například získat seznam všech typů, metod, vlastností, událostí a členských proměnných uvnitř dané assembly. Dá se zjistit i sada podporovaných rozhraní, parametry metody a další detaily. Následuje několik klíčových tříd jmenného prostoru System.Reflection.

- Assembly – třída obsahující metody umožňující načíst assembly, prozkoumávat a manipulovat s ní
- AssemblyName – odhaluje podrobnosti ukryté za identitou assembly
- EventInfo – obsahuje informace o dané události
- FieldInfo – obsahuje informace o dané datové členské proměnné
- PropertyInfo – obsahuje informace o dané vlastnosti
- MethodInfo – obsahuje informace o dané metodě
- MemberInfo – definuje chování typů EventInfo, FieldInfo, MethodInfo a PropertyInfo
- Module – umožňuje přístup k modulu uvnitř vícesouborové assembly
- ParameterInfo – obsahuje informace o parametru

3. C#

C# je programovací jazyk, který byl uveden společně s Visual Studiem .NET na začátku roku 2002. Je to vlastně zjednodušená, syntakticky čistá, vylepšená a striktně objektově orientovaná verze C++. Pro programování v C# je bezpodmínečně nutné mít nainstalovaný .NET Framework. To samé platí pro spouštění programů v C# vytvořených. C# umí produkovat pouze kód vykonavatelný uvnitř runtime .NET. Pro popis takového kódu se používá termín řízený kód (managed code). Binární jednotka obsahující tento řízený kód se jmenuje assembly. Jazyk nabízí následující schopnosti:

- Nepožadují se žádné ukazatele. Není třeba s nimi přímo manipulovat.
- Automatická správa paměti pomocí garbage collection.
- Formální syntaktické konstrukce pro výčty, struktury a vlastnosti tříd.
- Možnost jednoduše přetěžovat operátory.
- Podpora programovacích technik založených na rozhraních.
- Podpora aspektově orientovaného programování.

3.1 Programovací postupy

Programovací jazyk C# podporuje jak postupy objektově orientované programování, tak generického programování. Objektově orientovaným programováním rozumíme programovací vzor, který pro vytváření aplikací užívá objekty. Při programování se využívá vlastností těchto objektů. Tyto vlastnosti jsou např. vnitřní paměť, metoda objektu, schopnost přijmotu a zpracovat zprávu z venku a obsahovat jiné objekty.

Dalším programovacím postupem je generické programování. Při generickém programování je kód psán tak, že je oddělen algoritmus od datových typů. Jsou odděleny způsobem, že kód algoritmu je možno brát jako obecný, bez ohledu na použité datové typy. Teprve dosazením datových typů lze kód algoritmu chápat jako konkrétní.

3.2. Atributy

V programovacím jazyce se u jednotlivých částí kódu deklaruje řada informací. Například u třídy můžeme uvést jaké rozhraní má implementována, typ návratové hodnoty aj. Atributy jsou tedy konstrukce doplňující elementy kódu. (assembly, typy, moduly, členy, atd.) o určité informace. Atributy jsou stejně jako vše ostatní objekt. Každý z nich je instancí třídy Systém.Attribute. V C# jsou specifikovány napsáním atributu do hranatých závorek. Podle konvence by název představující třídu atributu končil slovem Attribute. Atributů může být i více a oddělují se čárkami. Atributy mohou přebírat parametry rozšiřující množství doplňujících informací a můžeme tvořit i svoje vlastní atributy. Musíme proto vytvořit třídu dědící přímo nebo ne ze třídy Systém.Attribute.

4. ASP.NET

ASP.NET je systém na tvorbu webových stránek, webových aplikací a XML webových služeb. Je to nástupce Microsoft technologie ASP. ASP.NET je postaven na CLR, dovolující psát kód v jakémkoliv .NET jazyce.

4.1 Charakteristika

V klasickém ASP je logika vložena do HTML. To je vhodné, pokud je kód krátký. Pracuje se s ním pohodlněji, jelikož je naráz vidět kód i HTML značky. Mezi výhody tohoto řešení patří snadnější odesílání jinému vývojáři, snadnější přejmenování z důvodu neexistujících závislostí mezi soubory a správa souborů v systému pro řízení zdrojového kódu.

V opačném případě jsou vytvořeny dva soubory. .aspx a .aspx.cs nebo .aspx.vb. Tomu se říká kód v pozadí - code-behind. Tato technika oddělí programovací kód od prezentační logiky HTML. Je vhodné jej použít tehdy, máme-li dost kódu a na jeho vývoji se podílí více lidí. Výhody takového řešení jsou: úplná separace HTML, na kterém pracují designéři a kódu, který zpracovávají programátoři. Kód se dá využít ve více souborech a nemají k němu přístup další nezainteresované osoby.

4.2 Adresáře

Při tvorbě stránek pomocí ASP.NET si můžeme vytvořit libovolnou adresářovou strukturu, nicméně existuje určitá skupina názvů, které mají pro APS.NET speciální význam.

App_Browsers: Složka pro definiční soubory prohlížeče identifikující prohlížeče a jejich schopnosti

App_Code: Složka, kam se ukládá zdrojový kód, který se kompiluje jako součást aplikace. Kód se kompiluje při požadavku na stránku a je automaticky přístupný.
App_Data: Složka pro ukládání databází
App_GlobalResources: Obsahuje soubory .resx přístupné každé stránce.
App_LocalResources: Obsahuje soubory .resx, které jsou určeny pro konkrétní stránku.
App_Themes: Zde se ukládají soubory určující vzhled stránky a ovládacích prvků ASP.NET.
App_WebReferences: Obsahuje schémata a jiné soubory asociované nějakou službou v aplikaci.

4.3. Ovládací prvky

Důležitou částí ASP.NET jsou serverové ovládací prvky (server controls). Jsou to třídy .NET Frameworku reprezentující vizuální prvky na webovém formuláři. Některé jsou jednoduché, mapované na konkrétní značku HTML. Jiné realizují komplexní zobrazení, složené z více HTML prvků. Serverové ovládací prvky se dělí do několika kategorií:

Webové ovládací prvky: Třídy duplikující základní značky HTML. Mají však smysluplnější a konzistentnější vlastnosti a metody. Tím pádem se snadněji deklarují a lépe se k nim přistupuje.

Bohatě vybavené ovládací prvky: Vyspělé ovládací prvky realizující velké množství HTML značek i JavaScriptu u klienta.

Validační ovládací prvky: Prvky umožňující snadné ověření, zda s nimi asociované vstupní ovládací prvky splňují standardní a definovaná pravidla.

Existují i další, specializovanější skupiny ovládacích prvků:

Datové ovládací prvky: Seznamy a mřížky, jejichž účelem je zobrazovat rozsáhlé objemy dat. Nebo ovládací prvky pro zdroje dat umožňující vázat se na různé zdroje dat bez nutnosti vytvářet další kód.

Navigační ovládací prvky: Určeny ke snadnému zobrazování mapy webu a dalších navigačních prvků.

Přihlašovací ovládací prvky: Prvky podporující autentizaci pomocí formuláře. Toto je model ASP.NET pro autentizaci uživatelů založený na databázi a sledování statusu uživatelů.

Ovládací prvky pro části webu: Ovládací prvky podporující WebParts. Pomocí tohoto modelu se budují konfigurovatelné webové portály založené na komponentách.

Ovládací prvky pro mobilní zařízení: Podobné webovým ovládacím prvkům, ale přizpůsobeno pro podporu mobilních klientů.

I přes množství těchto prvků obsažených v ASP.NET je někdy potřeba nějaký upravit, či vytvořit zcela nový prvek, vyhovující přesně naší představě. Mohou se tvořit dva druhy ovládacích prvku. Uživatelské ovládací prvky (user controls) a vlastní serverové ovládací prvky (custom controls).

Uživatelské ovládací prvky (user controls) jsou vlastní ovládací prvky tvořené kódem HTML a serverovým skriptem. Jednou vytvořený uživatelský prvek můžeme znovu využívat na dalších stránkách stejné webové aplikace. Tyto prvky se používají v případě jednodušších ovládacích prvku.

Vlastní serverové ovládací prvky (custom controls). Toto jsou prvky, které se jeví jako serverové ovládací prvky, ale zapouzdřují funkce, které serverový prvek neobsahuje. Používá se v případě potřeby složitějších prvků.

4.4. Model-View-Controller

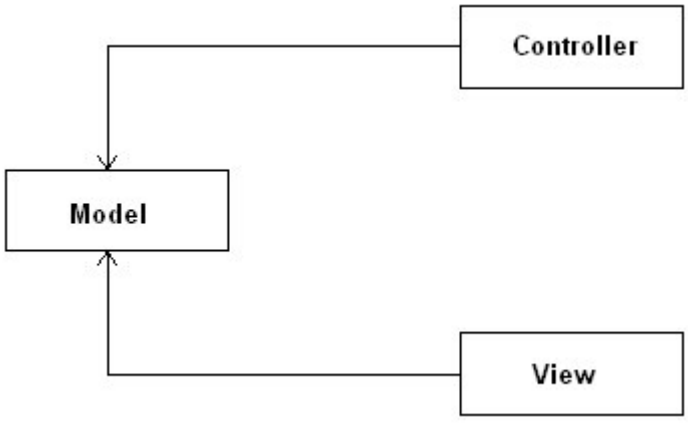
MVC je architektonický vzor používaný při vývoji software (Obr. 3.) . Vývojáři podle něj rozdělují aplikaci na tři nezávislé části: datový model aplikace (model), uživatelské rozhraní (pohled) a řídící logiku (řadič). Při modifikaci uživatelského rozhraní není ovlivněno zpracování dat a naopak při reorganizaci dat, není ovlivněno uživatelské rozhraní. Toho je

docíleno oddělením logiky a přístupu k datům od prezentace dat vložením prostředníka - řadiče. Důležité je, že pohled a řadič závisí na modelu. Nicméně, model nezávisí na pohledu ani na řadiči. Toto je klíčová výhoda separace. Jednotlivé komponenty mají následující funkce:

Model (model): Reprezentace informací, se kterou pracuje aplikace. MVC nespecifikuje vrstvu přístupu k datům, protože je vyrozuměna spodní vrstvou nebo zapouzdřena modelem.
View (pohled): Převádí data reprezentovaná modelem do podoby pro prezentaci uživateli.
Controller (řadič): Zpracovává a odpovídá na události, typicky uživatelské akce a může vyvolat změny v modelu nebo pohledu.

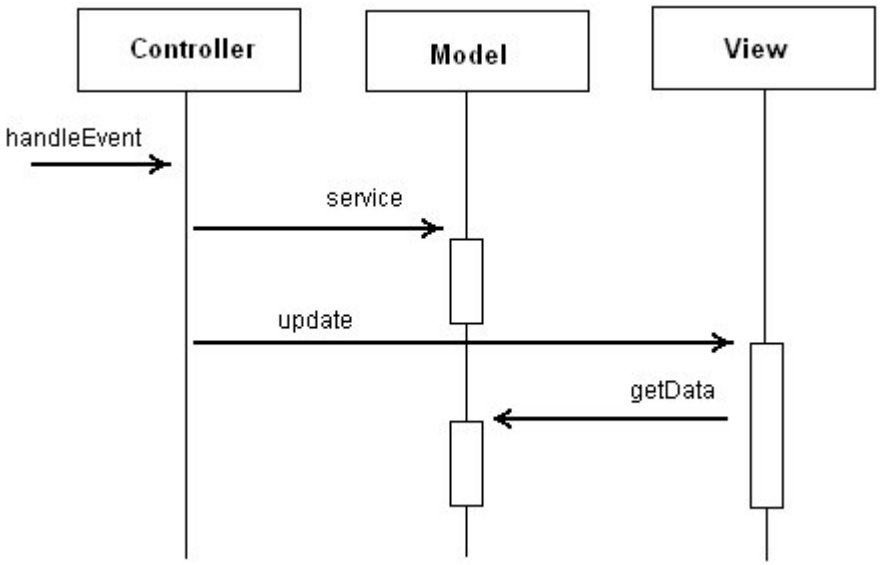
Obecný princip MVC

- Akce v uživatelském rozhraní (např. stisk tlačítka)
- Oznámení z uživatelského rozhraní řadiči
- Přístup řadiče k modelu a případná aktualizace na základě provedené akce uživatelem (např. aktualizace nákupního košíku)
- Zpracování dat doménovou logikou (např. přepočítání celkovou cenu položek v ceníku)
- Aktualizovaný model je použit pohledem pro zobrazení zaktualizovaných dat uživateli (např. vypsání obsahu košíku) Model není závislý na pohledu - nepotřebuje z něj žádné informace, ale pohled získává data přímo z modelu.



Obr. 3. Struktura tříd MVC

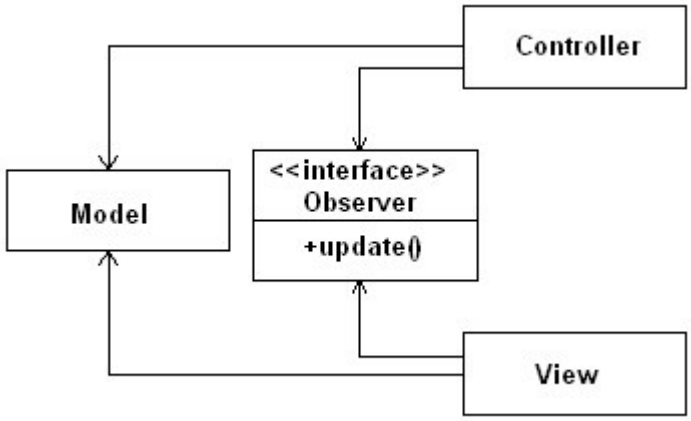
Existují dvě varianty MCV. Těmi jsou pasivní a aktivní model. Pasivní model (Obr. 4.) je použit, když řadič manipuluje výhradně s modelem. Řadič modifikuje model a pak informuje pohled, že model byl změněn a měl by být obnoven. V tomto scénáři je model nezávislý na pohledu a řadiči, což znamená, že zde nejsou prostředky pro model, aby oznámil změny v jeho stavu. Příkladem tohoto je model HTTP protokolu. Není jednodušší cesta pro prohlížeč dostat asynchronní updaty ze serveru. Prohlížeč zobrazí pohled a odpovídá na uživatelský vstup, ale nedetekuje změny v datech na serveru. Pouze v případě, že uživatel výslovně požaduje obnovení, je server dotázán na změny.



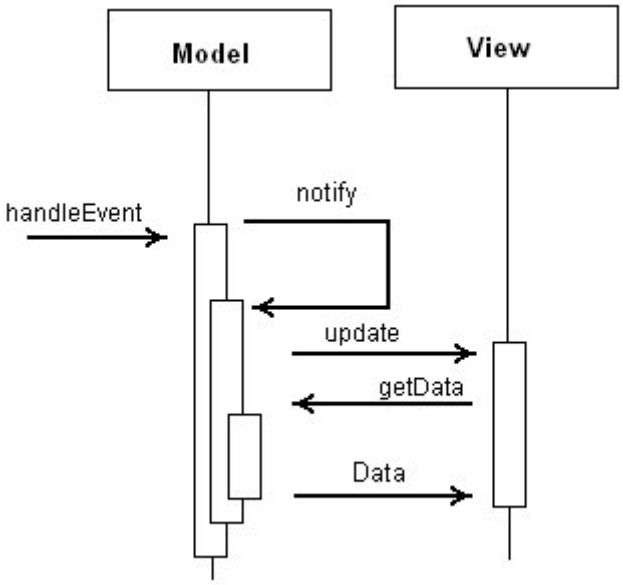
Obr. 4: Chování pasivního modelu

Aktivní model je použit, když model změní stav bez zapojení řadiče. To se může stát, když ostatní zdroje mění data a změny musí být reflektovány v pohledu. Například display burzovního telegrafu. Obdržíte burzovní data z externího zdroje a chcete obnovit pohled, když burza data změní. Protože jen model dokáže detekovat nastalé změny, musí je notifikovat pohledu, aby obnovil display. Nicméně jednou z motivací pro použití MVC vzoru je udělat model nezávislý na pohledu. Když model musí hlásit změny pohledu, využije se vzor

pozorovatel (observer). Zprostředkovává mechanismus na upozornění jiným objektům o změně stavu bez závislosti na něm (Obr. 5.). Individuální pohled implementuje rozhraní pozorovatele a registruje ho s modelem. Model sleduje seznam všech pozorovatelů odebírající změny. Když se model změní, opakuje se to skrz všechny registrované pozorovatele a upozorňuje se na změnu. Tento přístup je často nazýván jako veřejný odběr. Model nikdy nepotřebuje specifické informace o některém pohledu. Ve scénáři, kde řadič musí být informován o změně modelu, vše co řadič musí udělat je, že implementuje rozhraní pozorovatele a odebere změny modelu. (Obr. 6.)



Obr. 5: Použití pozorovatele k oddělení modelu od pohledu v aktivním modelu



Obr. 6: Chování aktivního modelu

4.5. Vhodnost .NET prostředí pro tvorbu aplikací

ASP.NET jako technologie pro vývoj webových aplikací má konkurenta v PHP. Obě aplikace jsou rozdílné, mají své výhody a nevýhody a je tedy na vývojáři, pro kterou se rozhodne. Zatímco ASP.NET je součástí mohutného .NET Frameworku, musí se řídit jeho pravidly a působí složitě, jednoduší PHP je samostatná menší technologie s účelovými nástroji. Mezi syntaxí PHP a ASP.NET je také velký rozdíl. Zatímco PHP skriptování využívá přímočaré funkce a tudíž je jednoduší, ASP.NET je o poznání náročnější objektově orientované programování. V .NET Frameworku je na výběr více programovacích jazyků, které lze použít, PHP je zároveň platforma i jediný skriptovací jazyk. U ASP.NET můžeme kód rozdělit do dvou soborů a oddělit tak programovací kód od prezentační logiky, u PHP se vše píše do jednoho souboru - tzv. spaghetti code. Velká výhoda PHP je v platformní nezávislosti a v tom, že je vyvíjeno jako open source, tudíž je zdarma.

Ze své povahy je tedy PHP vhodné pro jednoduší projekty, kde je brán zřetel na cenu, jednoduchost programování a je požadována multiplatformnost. Po ASP.NET naopak sáhne vývojář velkého projektu dbající na komplexnost celého projektu, robustnosti řešení a podpory.

5. ADO.NET

ADO.NET je sada softwarových komponent, které nejčastěji programátoři používají pro přístup a modifikaci dat uložených v relačních databázích. Mohou být využity i pro přístup k nerelačním zdrojům. ADO.NET je někdy považováno za evoluci ADO (ActiveX Data Object), ale bylo změněno natolik, že ho lze považovat za zcela nový produkt. ADO.NET používá vícevrstvou architekturu a skládá se ze dvou hlavních částí. Těmi jsou „Data provider“ a „DataSets“ .NET Framework je vybaven čtyřmi poskytovateli a jsou to: SQL Server, OLE DB, Oracle, ODBC.

Data provider

Tyto třídy zprostředkovávají přístup ke zdroji dat, jako je Microsoft SQL Server, nebo Oracle databáze. Každý zdroj dat má svou vlastní sadu dodavatelů objektů, ale všichni mají společnou sadu pomocných tříd.

Connection: Zprostředkovává propojení použité pro komunikaci se zdroji dat.

Command: Používá se k provedení akcí na zdroji dat jako je čtení, updatování nebo smazání.

DataAdapter: Post používaný pro přenos dat mezi zdrojem dat a objektem DataSet

DataReader: Třída používaná k efektivnímu zpracování dlouhého seznamu výsledků najednou.

DataSets

DataSets je skupina tříd reprezentující v paměti data, které zprostředkovává relační model bez ohledu na zdroj dat, který obsahuje. Reprezentuje kompletní obsah tabulek, stejně jako propojení mezi tabulkami.

Objekt DataTable reprezentuje samostatnou tabulku v databázi, která má jméno, řádky a sloupce.

DataView řadí data a filtruje záznamy.

DataColumn reprezentuje sloupec tabulky. Obsahuje jeho jméno a typ.

DataRow reprezentuje řádek v tabulce a povoluje čtení a updatování obsažených informací.

DataRowView reprezentuje jeden řádek z rozdílu DataView. Rozdíl mezi DataRow a DataRowVier je důležitý při opakování výsledků.

DataRelation popisuje propojení mezi tabulkami.

Constraint popisuje vynucené vlastnosti databáze jako unikátnost hodnot v primárním klíčovém sloupci.

5.1. Vázání dat

O vázání dat se mluví v případě, kdy je svázán zdroj dat s ovládacím prvkem, který automaticky zobrazuje data. Toto vázání není programátorské, ale deklarativní. Definuje se ve stránce .aspx s ovládacími prvky. Většina prvků podporuje jednoduché vázání dat. Některé podporují i složité vázání dat a vznikají tak různé seznamy či mřížky. Při jednoduchém vázání je prvek vázán jen k jedné hodnotě, přičemž nemusí být vázáno něco, co je vidět přímo na stránce. Při složitém vázání se nenaplňuje prvek přímo daty, ale volá se metoda DataBind(), která ze zdroje data extrahuje a zobrazí.

Tabulka dat neboli relace je uspořádaná do sloupců a řádků. Sloupce značí atributy a řádky jsou hodnoty atributu. Primární klíč je jednoznačnou identifikací řádku tabulky. Cizí klíč omezuje spojení jednoho, nebo více sloupců tabulky do jiné tabulky. Pokud jsou hodnoty shodné, řádek tabulky je rozšířen řádkem cizí tabulky.

6. Databáze

Dnes je velmi mnoho odvětví postaveno na databázích. Eviduje se v nich v podstatě vše, počínaje knihami v knihovně, přes seznam klientů bank až po evidenci občanů na úřadech. Databáze se dá zjednodušeně popsat jako úložiště dat. K jejich používání existuje několik důvodů. Databáze poskytuje rychlejší přístup k datům, může k nim přistupovat paralelně, má zabudován systém uživatelských práv a pomocí dotazů můžeme vybírat jen požadované množiny dat. Databází je mnoho typů, níže jsou uvedeny jsou některé z nich.

Relační databáze

Základem databáze je tabulka (relace). Tabulky jsou mezi sebou propojeny a tento soubor tabulek pak tvoří databázi. Tabulka je určena primárním klíčem a propojení tabulek se nazývá funkční závislost. Tabulka se závislostí na jinou má jak primární klíč, tak cizí klíč. Relační databáze jsou jednoduché a díky tomu hojně rozšířené.

Objektově orientované databáze

Databáze funguje na principech objektového programování. Základem databáze není tabulka, ale objekt. Každý takový objekt má vlastnosti a metody, které manipulují s hodnotami vlastností. Jednotlivé záznamy jsou instance objektu s určitou hodnotou. Díky možnosti využití dědičnosti, zapouzdření a polymorfismu jsou rozšířeny možnosti tvorby aplikací.

Deduktivní databáze

Deduktivní databáze funguje na principu logického programování. Použití těchto databází je úplně jiné a využívají se např. pro vědecké účely (umělá inteligence). V těchto databázích se používají relace a deduktivní pravidla.

Temporální databáze

Využívá se tam, kde je potřeba zpracovávat nová a zároveň stará data. Jedná se o rozšíření předchozích typů databází. Oproti nim nejsou časově závislé. Je zaveden systém časových razítek a může se tedy dotazovat na minulost, přítomnost a budoucnost.

6.1 SQL

V této práci je využíváno relační databáze a jazyka SQL, což je strukturovaný dotazovací jazyk pro práci s daty v relační databázi. Jeho předchůdce SEQUEL se začal vyvíjet už v 70. letech minulého století. SQL bylo nakonec standardizováno jako jazyk relačních databází. Každá relační databáze tento standard podporuje, ale nejsou implementovány vždy všechny normy. V tomto projektu bude dále používán jako zdroj dat databáze MS SQL Server 2005. SQL příkazy jsou rozděleny do čtyř skupin. Jsou to:

- Příkazy pro manipulaci s daty - příkazy získají data z databáze a mohou je upravovat.
- Příkazy pro definici dat – příkazy na tvorbu struktury databáze.
- Příkazy pro řízení dat – řízení transakcí a nastavování přístupových práv.
- Ostatní příkazy – pro přidávání uživatelů a nastavování systémových parametrů.

7. Redakční systém

Redakční systém je aplikace běžící na serveru, která umožňuje správu obsahu webových stránek. Správu s pomocí redakčního systému může provádět i laický uživatel neznalý HTML a programování webových stránek obecně. Takovýto systém nevyžaduje instalaci žádných drahých aplikací na klientské PC a není tudíž omezen na stroj, na který je nainstalován. Uživatel může stránku administrovat odkudkoliv a stačí mu k tomu pouze internetový prohlížeč. Pro malou firmu a jednotlivce je to nástroj pro snadné a efektivní aktualizování, pro velké firmy prakticky nezbytnost. Hlavním cílem této práce je návrh struktury databáze redakčního systému a jeho realizace. Systém se skládá ze dvou částí. První část je administrační, ta dovoluje vkládat data, mazat a editovat. Druhou je prezentační, tj. ta, kterou vidí běžný návštěvník.

8. Návrh struktury databáze.

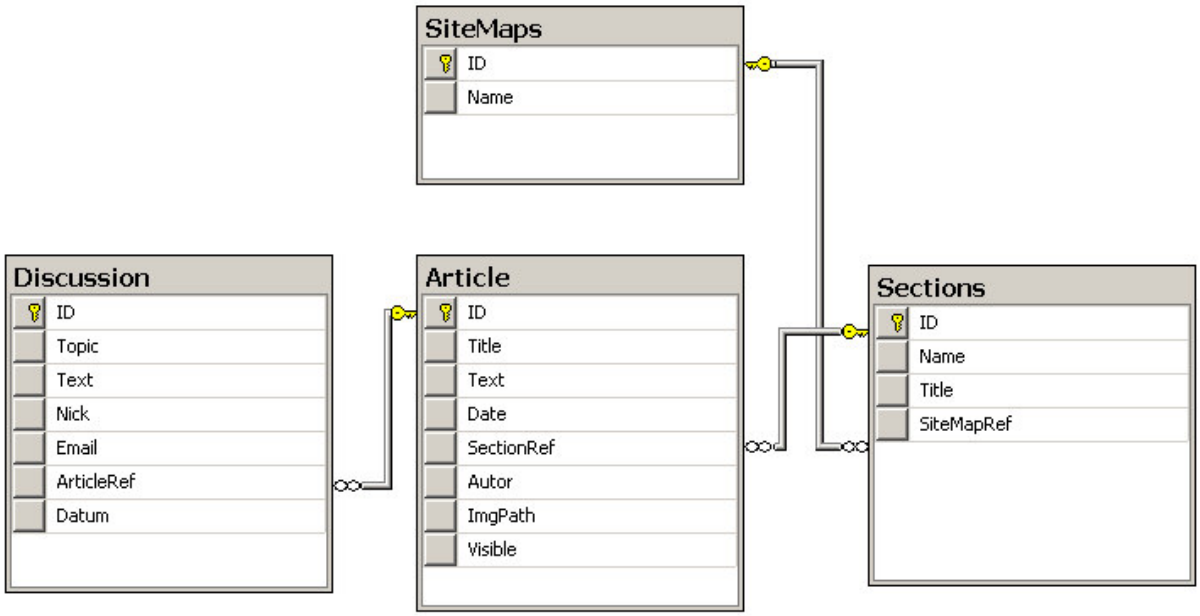
Byla navržena základní struktura (Obr. 7.) databáze redakčního systému. Jedná se o relativně jednoduchou databázi.Obsahuje čtyři tabulky popsané níže.

Tabulka **SiteMaps**
ID – primární klíč.
Name – jméno sitemapy.

Tabulka **Sections**
ID – primární klíč.
Name – popis sekce v SiteMaps.
Title – název sekce.
Cizí klíč SiteMapRef – ukazující do řádku SiteMaps.

Tabulka **Article**
ID – primární klíč.
Title – název článku.
Text – vlastní text článku.
Date – datum vložení článku.
SectionRef – cizí klíč, ukazatel do tabulky Sections. Udává příslušnost k sekci.
Autor - Jméno autora.
ImgPath – cesta k obrázku připojeného k článku.
Visible – určuje, zda článek bude viditelný, či ne.

Tabulka **Discussion**
ID – primární klíč
Topic – nadpis příspěvku.
Text – text příspěvku.
Nick – jméno autora.
Email – email autora.
ArticleRef – cizí klíč ukazuje, ke kterému článku příspěvek patří.
Datum – datum vložení příspěvku.



Obr. 7. Schéma struktury databáze.

9. Tvorba systému

Redakční systém byl naprogramován v jazyce C# ve Visual Studiu 2005. Součástí instalace Visual Studia 2005 je i databázový server Microsoft SQL Server 2005, na kterém byla umístěna používaná databáze, vytvořená podle návrhu v této práci. Databáze, stejně jako celý program byly používány lokálně, nebyly tedy umístěny na serveru v internetu. Práce na síti je emulována komponentou Visual Studia ASP.NET Development Server.

9.1 Databáze

Ovládání Visual Studia je velmi intuitivní a rychlé. Obsahuje komponentu Server Explorer, která umožňuje spravovat databázi. Nebylo tedy třeba využívat externího programu jako např. SQL Server Management Studio. Dovoluje tvorbu, editaci, mazaní a plnění databázových tabulek, databázového diagramu, testování dotazů na databázi atp. Pomocí této komponenty byly vytvořeny tabulky podle návrhu, nastaveny parametry buněk a vytvořeny vzájemné relace mezi tabulkami. U relací bylo nastaveno kaskádovité mazání dat. Při odstranění dat v nadřazené tabulce se automaticky odstraní i data nich závislá. Při odstranění sekce se tak smažou i články patřící do této sekce a příspěvky k článkům. Dále zde bylo nastaveno automatické vyplňování primárního klíče. Číslo ID se automaticky inkrementuje a nemůže tak dojít k zadání jednoho ID dvakrát.

9.2 Struktura

Pro tvorbu vlastního systému byl založen projekt podle šablony „ASP.NET Web site“, který po zvolení programovacího jazyka a umístění projektu předpřipraví adresářovou strukturu. Ta byla doplněna o adresáře „Autor“ a „Pictures“ pro ukládání stránek dostupných pouze pro autora resp. ukládání obrázků k jednotlivým článkům. Další důležité prvky mimo vlastních souborů s kódem jsou:

MasterPage. - tento soubor obsahuje jakousi šablonu celého projektu. Vloží se zde tabulka, která nadefinuje celkový vzhled. Bylo zvoleno rozložení se záhlavím a jednou stranou. Tyto prvky se při prohlížení stránek nemění a umisťují se zde prvky, které by měly být neustále viditelné, jako např. mapa stránek v tomto případě zastoupená prvkem menu, prvek pro zobrazení zda je čtenář/autor přihlášen atp. Mění se jen obsah v okně prvku ContentPlaceHolder, ve kterém jsou zobrazovány vlastní stránky.

Web.config - tento soubor obsahuje konfigurační informace. Jsou zde uloženy informace o typu autentizace, použitá témata nebo řetězec pro připojení k databázi.

Web.sitemap -soubor definující zdroj pro prvky jako menu, zprostředkovávající mapu stránek.

App_Data - adresář obsahující vytvořenou databázi používanou programem pro ukládání dat a databázi vytvořenou administračním nástrojem.

App_Themes - v tomto adresáři jsou uloženy grafická témata využívané jednotlivými stránkami. Můžeme zde např. globálně nastavit velikost písma pro určité prvky, barvu atp.

9.3 Administrace uživatelů

Administrace uživatelů je vytvořena pomocí nástroje Web Site Administration Tool obsaženém ve Visual Studiu. Bezpečnostní politika je zajištěna pomocí uživatelů a rolí. V roli se nastavuje pro který adresář platí a zda jde o pravidlo povolení nebo zakázání. Poté uživatelům přiřadíme role. Byly vytvořeny dvě role a to sice autoři a registrovaní čtenáři. Uživatelé v roli autora mají pak povolen vstup na stránky umožňující editaci, generovaní sitemapy, mazání článků a sekcí. Registrovaní čtenáři mohou vkládat příspěvky ke článkům. Ostatní – nepřihlášení mohou jen číst.

10. Části redakčního systému

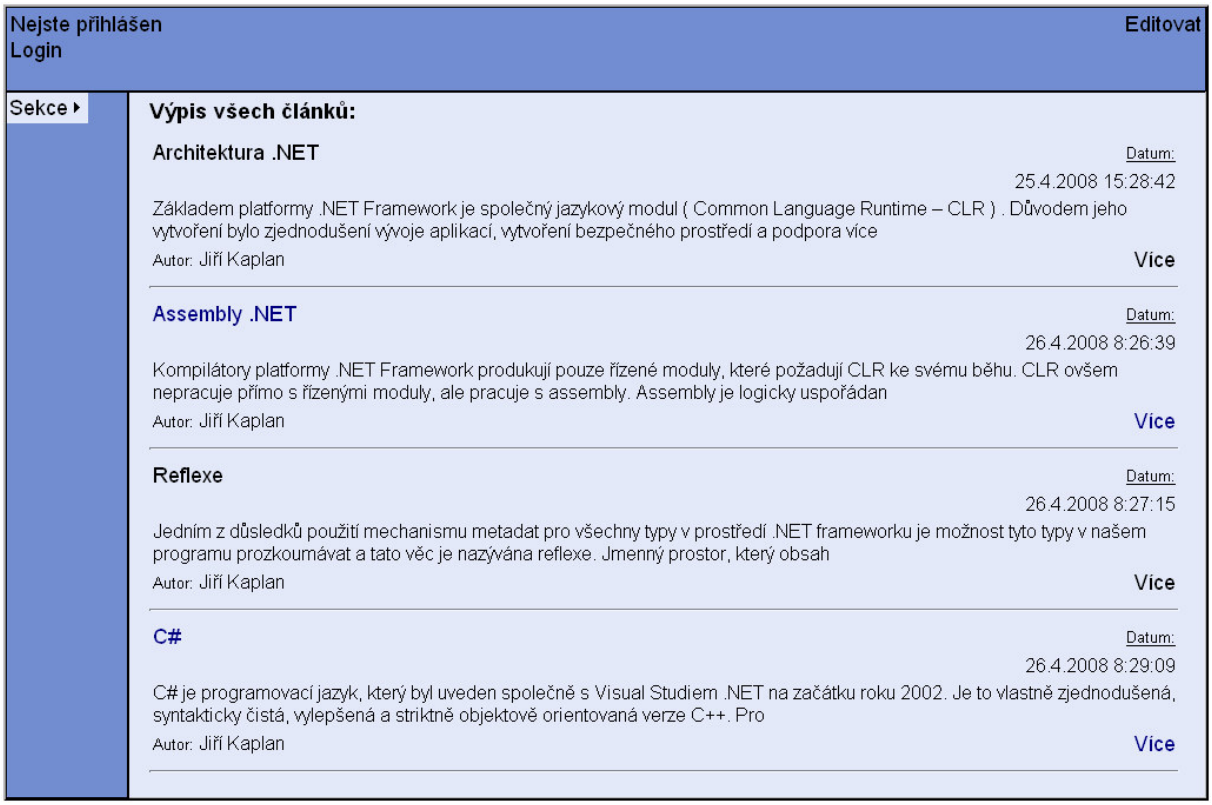
Systém je rozdělen na dvě části. Jedna část je prezentační, druhá administrační. V prezentační části jsou všem zobrazovány veškeré sekce, články a diskuse ke článkům. Ke druhé, administrační části, má přístup pouze vlastník webových stránek, který disponuje uživatelským jménem a heslem pro přístupu k této části.

10.1 Prezentační část

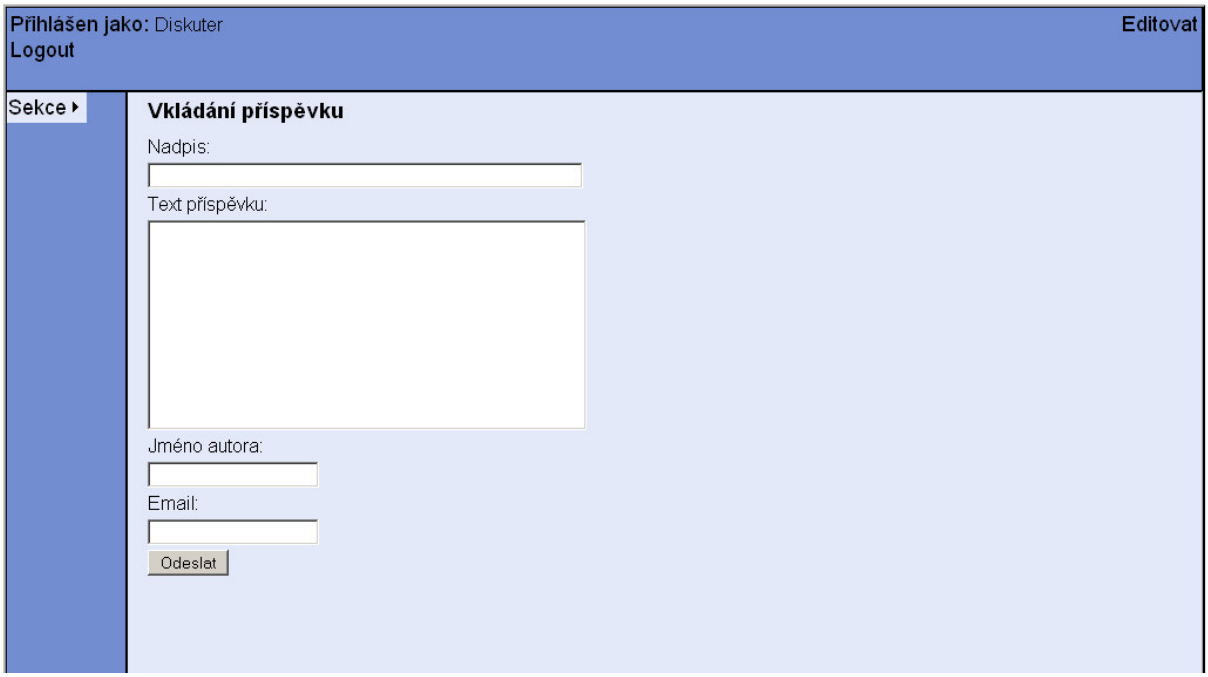
Tato část je ihned viditelná po vstupu na stránky (Obr.8). Je to klasická webová prezentace, kde pro přístup není vyžadováno heslo ani jiná autorizace. Návštěvník může procházet sekce, číst články a zobrazovat diskuse ke článkům. V záhlaví okna je neustále vypisováno, v jaké části se návštěvník nachází. Články zobrazené v seznamech se zobrazují zkráceně. Teprve po kliknutí na název článku, nebo odkaz „Více“ se článek zobrazí celý i s možnosti přidání příspěvku nebo prohlédnutí diskuse. Návštěvník ovšem nemůže do diskusí přispívat. Pro přispívání musí mít založen účet, ke kterému je přiřazena role registrovaného čtenáře. Účet může založit vlastník prezentace přes Web Site Administration Tool .

10.1.1 Přispívání do diskuse

Pokud má čtenář založen svůj účet a chce psát příspěvek (Obr. 9), přihlásí se tlačítkem „Login“ v levém horním rohu. Poté je mu umožněno vkládat pod články příspěvky. Na stejném místě se může poté odhlásit.



Obr. 8: Prezentace.



Obr. 9: Psaní příspěvků.

10.2 Administrační část

Tato část je skryta pod tlačítkem „Editovat“ v pravém horním rohu. Pro vstup do ní musí být uživatel – administrátor stránek přihlášen ke svému uživatelskému účtu, jenž mu umožňuje do administrační části vstupovat. Zde má na výběr z několika možností.

10.2.1 Přidání sekce

Tímto odkazem se administrátor dostane na stránku přidávání sekce. Zde se zadává název nové sekce a její popisek, zobrazovaný při najetí kurzorem na její název v menu. Název i popisek je omezen na 50 znaků. Zaškrtávací pole slouží k určení, zda má být sekce viditelná. Pokud ještě nemá být zveřejněn a bude se dále editovat, pole zůstane nezatržené.Tlačítkem „Vlož“ se sekce uloží do databáze (Obr.10). Po vložení sekce je nutné vygenerovat znovu sitemapu, aby nová sekce byla zobrazena v menu. Bez vygenerování se v menu zobrazují jen sekce vložené před posledním vygenerováním.

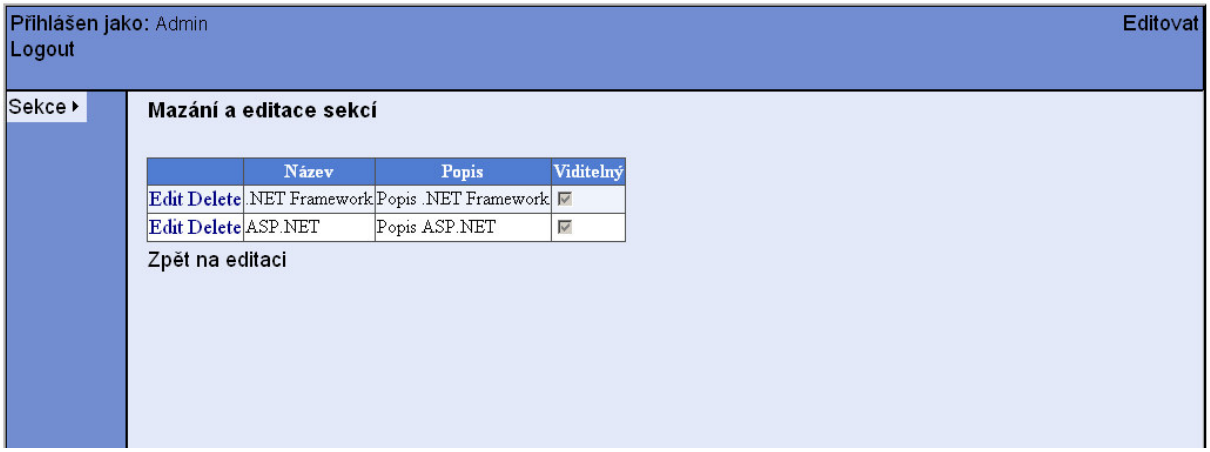
10.2.2 Generování sekce

Do části generování sitemapy se vstupuje odkazem „Vygenerovat sitemapu“ v sekci editace. Zde se nachází jen jedno tlačítko, jehož stisknutím se provede vygenerování.

Obr. 10: Přidání sekce.

10.2.3 Mazání a editace sekce

V této části může administrátor smazat nebo editovat název či popisek existující sekce a to, zda sekce bude vidět. Existující sekce jsou vypsány v tabulce a operace se provádí tlačítky „Edit“ a „Delete“ (Obr.11). V případě, že je sekce smazána, smažou se automaticky i veškeré články v ní obsažené a všechny příspěvky k těmto článkům. Po každém smazání či editaci musí být opět vygenerována sitemapa.



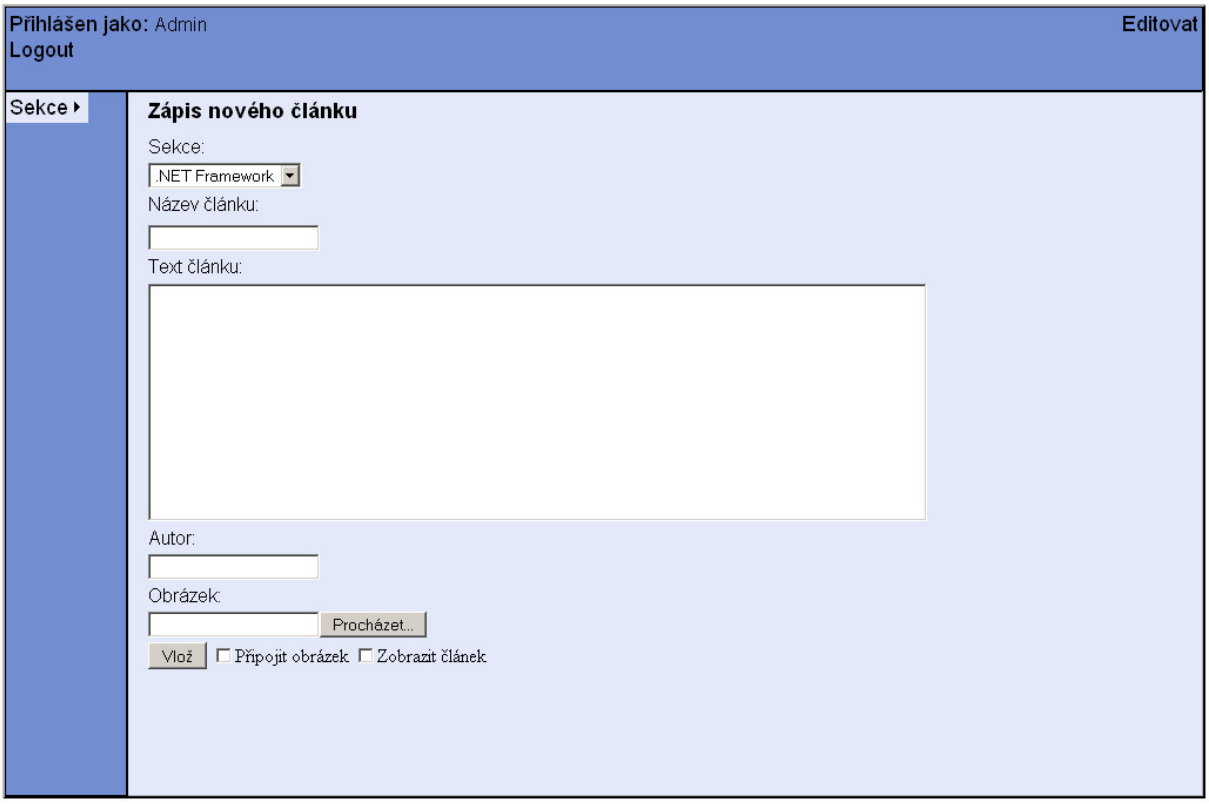
Obr. 11: Mazání a editace sekcí.

10.2.4 Přidání článku

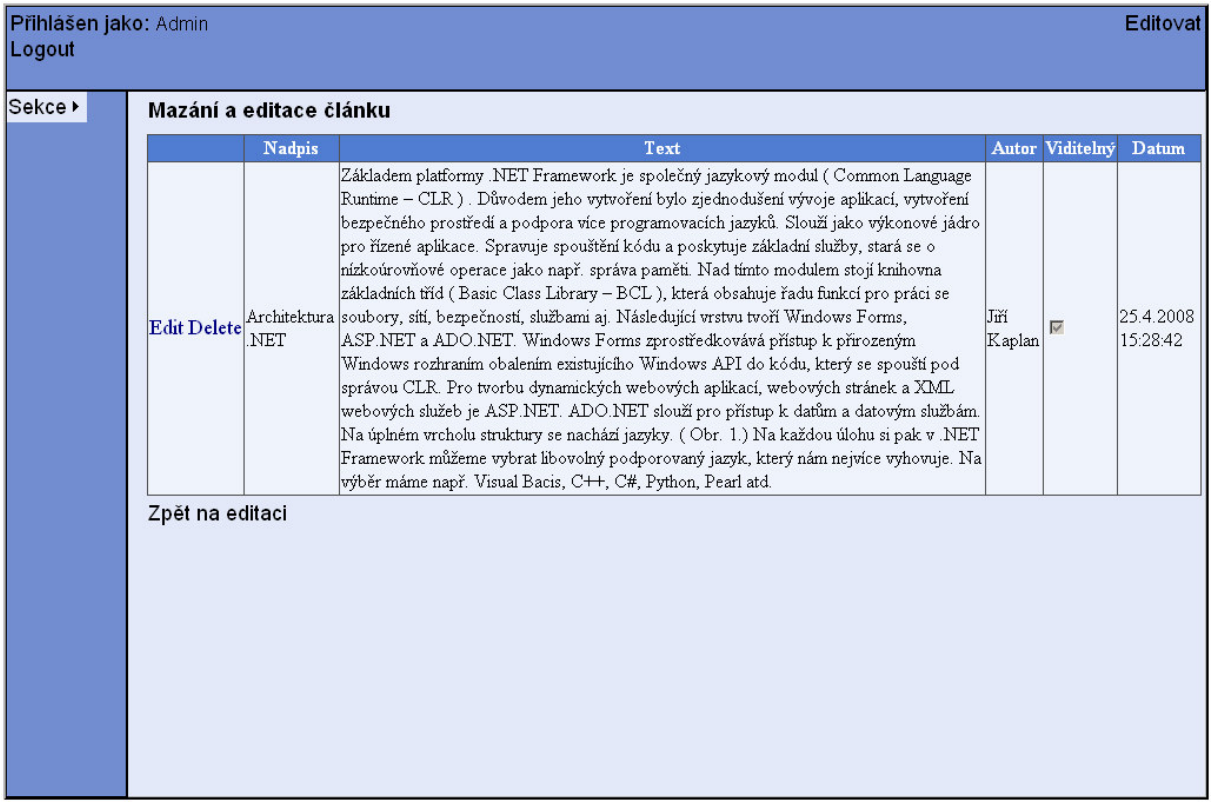
Při vytváření článku (Obr. 12) musí být pomocí rolovacího menu vybrána sekce, do které nový článek bude patřit. V tomto menu se zobrazují i sekce přidané před aktualizací sitemapy. Do pole název článku je zadán nadpis a do následujícího pole je zapsán vlastní text článku. Text je omezen na 2000 znaků a nadpis na 50 znaků. Do dalšího pole se zadá jméno autora, omezeno je na 30 znaků. Pokud má být k textu přiložen obrázek, načte se pomocí následujícího tlačítka „Procházet“ a zaškrtnutím pole „Připojit obrázek“ se potvrdí záměr obrázek vložit. Další zaškrtávací pole slouží k určení, zda má být článek zobrazen čtenářům. Pokud se jedná o koncept, co ještě nemá být zveřejněn a bude se dále editovat, pole zůstane nezatržené. Poté se tlačítkem „Vlož“ článek uloží do databáze a obrázek se ze svého původního umístění nahraje do složky s redakčním systémem.

10.2.5 Mazání a editace článku

Část podobná s mazáním a editací sekce. Zde je možno článek buď pozměnit, nebo úplně smazat pomocí tlačítek „Edit“ a „Delete“ (Obr. 13). Při editaci lze změnit jeho název, vlastní text, autor, zda bude viditelný a datum.



Obr. 12: Přidání nového článku.



Obr. 13: Mazání a editace článku.

9. Závěr

V této semestrální práci je popsáno prostředí .NET Framework, jeho verze a architektura. Byly prozkoumány možnosti .NET pro správu webového obsahu a porovnáno s PHP. Jsou popsány základy ASP.NET a ADO.NET. Dále je v práci obsaženo stručné seznámení s programovacím jazykem C#. V práci je uveden take popis některých druhů databází a stručný popis jazyka SQL. Pratickou částí této práce je pak návrh struktury databáze a následná tvorba redakčního systému na ní založeného. Redakční systém je nástroj vhodný k tvorbě webových stránek i pro uživatele, jenž nemají znalosti HTTP či programování. Jeho struktura je navržena pro tvorbu jednoduchých webových stránek obsahující články dělené do sekcí. Ke každému článku mohou být přidány diskusní příspěvky. Články a sekce může vkládat pouze autor stránek. Může také určit, zda budou viditelné a ke každému článku může vložit obrázek. Tím pádem bude moci editovat text, který zůstane uložený na serveru a k jeho editaci se může vrátit, aniž by byl mezitím přístupný veřejnosti. Odpadá tak nutnost ukládání rozdělané práce mimo webové stránky. Diskusní příspěvky mohou vkládat pouze registrovaní uživatelé, ověření autorem, čímž se zabrání vkládání spamu do diskuse od neregistrovaných uživatelů.

10. Literatura

[1] FAJFR. *Architektura Microsoft .NET Framework – 1. díl* [online]. 2006 [cit. 2007-12-12]. Dostupný z WWW: <http://programujte.com/index.php?akce=clanek&cl=2006042604-architektura-microsoft-net-framework-%96-1-dil>.

[2] PUŠ, Petr. *Poznáváme C# a Microsoft.NET 36. díl – úvod do reflexe* [online]. 2005 [cit. 2007-12-15]. Dostupný z WWW: <http://www.zive.cz/default.aspx?section=21&server=1&article=126122>.

[3] BOREK, Bernard. *Využití objektových technologií při vývoji webových aplikací:*. [s.l.], 2004. 55 s. Bakalářská práce.

[4] Troelsen Andrew, C# a .NET 2.0 profesionálně, ZONER Press 2006, ISBN 80-86815-42-0

[5] Jason Price, C# programování databází, Grada 2005, ISBN 80-247-0982-1

[6] Jeff Prosise, Microsoft .NET Webové aplikace v :NET Framework, C# a ASP.NET, Computer Press 2002, ISBN 80-7226-879-1

[7] MacDonald Matthew, Szpuszta Mario, ASP.NET 2.0 a C# tvorba dynamických stránek profesionálně, ZONER Press 2006, ISBN 80-86815-38-2

[8] *Model-View-Controller* [online]. 2007 [cit. 2007-12-18]. Dostupný z WWW: <http://msdn2.microsoft.com/en-us/library/ms978748.aspx>.

[9] SKŘIVAN, Jaromír. Databáze nejsou jen MySQL. *Interval.cz* [online]. 2002 [cit. 2008-04-20]. Dostupný z WWW: <http://interval.cz/clanky/databaze-nejsou-jen-mysql/>.