VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

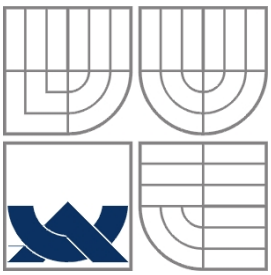
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF IMFORMATION SYSTEMS

NÁSTROJ PRO KONVERZI STATICKÝCH WEBOVÝCH STRÁNEK DO KENTICO CMS

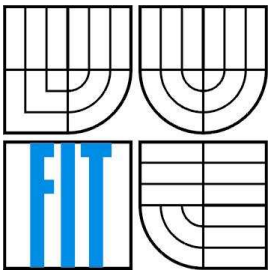
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE ROBERT STEBEL
AUTHOR

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

NÁSTROJ PRO KONVERZI STATICKÝCH WEBOVÝCH STRÁNEK DO KENTICO CMS

TOOL FOR IMPORT OF WEB PAGES INTO KENTICO CMS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

ROBERT STEBEL

VEDOUCÍ PRÁCE
SUPERVISOR

ING. LADISLAV RUTTKAY

BRNO 2010

Abstrakt

Tato práce popisuje konverzi statických webových stránek do redakčního systému Kentico CMS. Jsou zde prezentovány všechny technologie použité při řešení. Zároveň je přiblížen tento redakční systém, pro který je výsledná aplikace primárně určena. Druhá část práce se věnuje návrhu a implementaci obou částí výsledného projektu.

Abstract

This thesis describes the conversion of static Web pages into CMS Kentico CMS. There are presented all the technology used in the solution. It deals also with CMS system, on which is conversion application implemented. The second part deals with design and implementation of both parts of the final project.

Klíčová slova

HTML, konverze, Kentico, CMS, .NET, C#, XML

Keywords

HTML, conversion, Kentico, CMS, .NET, C#, XML

Citace

Robert Stebel: Nástroj pro konverzi statických webových stránek do Kentico CMS, bakalářská práce, Brno, FIT VUT v Brně, 2010

Nástroj pro konverzi statických webových stránek do Kentico CMS

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing.Ladislava Ruttkaye.
Další informace mi poskytl Ing. Martin Hejtmánek
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Robert Stebel
12.05.2010

Poděkování

Chtěl bych poděkovat Ing. Martinovi Hejmánkovi a Ing. Ladislavovi Ruttkayovi za poskytnuté rady a pomoc při psaní této práce.

Obsah

Úvod

V dávných dobách, kdy se Internet rodil do současné podoby, byla možnost vytvářet webové prezentace velice chudá. Neexistovalo moc editorů na tuto práci a samotné možnosti těchto prezentací nebyly na dnešní dobu nijak ohromující. Časem se však tyto možnosti zlepšovaly a dnes již poskytují velice pokrokové funkce. V současné době existuje již velké množství nástrojů na tvorbu webových stránek. Tyto nástroje jsou ve formě editorů nainstalovaných na počítači (CoffeeCup HTML Editor, StudioLine Web) nebo jako webové aplikace, dostupná stejně jako jiné webové stránky přes prohlížeč. Obecně jsou tyto webové aplikace známé jako redakční systémy nebo CMS (z anglického Content management system – systém pro správu obsahu).

Taktéž existuje hodně různých konverzních programů na převody různých formátů. Existuje spousta programů na převod obrázků, hudby i videa. Převádět se dá téměř cokoliv na něco jiné. Při spojení těchto dvou odvětví se dostáváme k tomuto projektu. Tentokrát se jedná o konverzi ze statických webových stránek do pokročilého redakčního systému. Tato práce se tedy zabývá převodem statických HTML stránek do jednoho z CMS systému, konkrétně do Kentico CMS.

V první kapitole uvedeme potřebné znalosti pro řešení zadaného úkolu. Probereme technologie použité při implementaci konverzní aplikace. Zejména se seznámíme se strukturou statické HTML stránky. Také si popíšeme XML, C#, .NET Framework. V krátkosti se seznámíme s programy použitými při řešení projektu.

Ve druhé kapitole se seznámíme s CMS systémy. Shrneme základní vlastnosti těchto redakčních systémů. Také si popíšeme architekturu a způsob ukládání dat CMS systému od firmy Kentico, pro který je tato aplikace primárně implementována. Popíšeme si strukturu databáze a ukážeme závazná pravidla platící pro Kentico API.

Třetí kapitola se věnuje návrhu implementace zadané úlohy. Ukážeme si konverzní model HTML stránek do Kentico CMS a návrh výsledné architektury jak klientské části, tak i části která bude integrována do CMS systému. Popíšeme si způsob prováděného testování a shrneme možná vylepšení, kterými by v budoucnu mohl tento projekt disponovat.

V závěrečné čtvrté kapitole shrneme výhody a nevýhody celého projektu.

1 Potřebné znalosti

Pro řešení zadaného úkolu je potřeba rozumět použitým technologiím – HTML, XML, .NET Framework a implementačnímu jazyku, kterým je C# . Dále si popíšeme webové služby a regulární výrazy.

1.1 Použité technologie

1.1.1 HTML

HTML je zkratka pro HyperText Markup Language, což znamená značkovací jazyk pro hypertext. Je to jednoduchý jazyk pro tvorbu statických internetových stránek, které za pomocí protokolu HTTP (HyperText Transfer Protocol – přenosový protokol hypertextu) začali tvořit Internet, jak ho známe dnes. První verze HTML jazyka vznikla kolem roku 1991 a dnes je na světě již verze s pořadovým číslem 4.01, která je z roku 1999 a používá se dodnes. Nová verze, HTML 5, je ve stádiu vývoje. Webové stránky v jazyce HTML jsou statické stránky, které se v dnešní době používají většinou v kombinaci s jinými jazyky jako například JavaScript, PHP, ASP. HTML jazyk má předepsanou strukturu každé stránky, která se pak má za pomocí prohlížeče zobrazit.

HTML dokument sestává ze tří základních celků [1]:

- Definice DTD
- Deklarace hlavičky
- Tělo, které obsahuje aktuální obsah dokumentu

Definice DTD (Document Type Definition) – je definována direktivou <?DOCTYPE
Je to první prvek celé struktury a oznamuje prohlížeči, jaká verze HTML je v dokumentu použita.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
```

Deklarace hlavičky HTML dokumentu – uvádí se párovým tagem <head> </head>
Součástí hlavičky jsou informace o aktuálním dokumentu. Především jde o nadpis dokumentu, který je povinný pro validní HTML stránku a kódování textu pro správné zobrazení speciálních znaků použité znakové sady v písemném projevu. Ukládají se zde i metadata, což jsou doplňující informace, které využívají vyhledávací roboti. Mimo jiné obsahuje také odkazy na skripty Javascriptu, odkazy na externí CSS soubory stylů stránky či jiné doplňky.

```
<head>
  <meta http-equiv="content-type" content="text/html; charset=iso-8859-2">

  <link rel="shortcut icon" href="/favicon.ico" />
  <link rel="stylesheet" type="text/css" href="default.css">

  <script type="text/javascript" src="javascript.js"></script>
  <script type="text/javascript" src="checking.js"></script>

  <title>Moje HTML stránka</title>
</head>
```

Tělo HTML dokumentu – uvádí se párovým tagem <body> </body>

Zde je definovaný samotný obsah dokumentu, který se zobrazí po otevření v prohlížeči. Obecně lze říct, že obsah může být cokoliv (obrázek, tabulka, text). Díky množství formátovacích elementů, formulářů, tabulek a odkazů vznikne samostatný webový portál sdílející informace s každým, kdo se na daný web připojí. Jako základní prvky lze považovat tagy <table>, <font>, <a>,
, . Mezi další příkazy patří také definice kaskádových stylů CSS, které dodávají stránkám potřebnou grafickou úpravu a bez něhož se v dnešní době neobejde žádný rozsáhlejší webový portál.

```
<body>
<div id="wrapper" >

      <h1> Moje HTML stránka </h1>
      <h2> příklad HTML dokumentu </h2>

      <div class="color:red">
            Robert Stebel , xstebe01@stud.fit.vutbr.cz<br>
      </div>

      <table border="2">
            <tr>
                  <td>1. buňka tabulky</td>
                  <td>2. buňka tabulky</td>
            </tr>
      </table>

      <a href="stranka.html" title="odkaz">odkaz na jinou stránku </a>
      <img src="obrazek.jpg" alt="obrazek">
</div>
</body>
```

1.1.2 XML

XML je další z řady značkovacích jazyků. Zkratka XML je z anglického Extensible Markup Language, což by se dalo přeložit jako rozšiřitelný značkovací jazyk. Byl vyvinut konsorciem W3C z předešlých poznatků značkovacích jazyků jako ideální formát pro ukládání strukturovaného a semi-strukturovaného textu. Ve skutečnosti se jedná o metajazyk, protože se používá k popisu jiných jazyků. Nemá žádný předdefinovaný seznam elementů avšak jeho syntaxe je přísnější než u HTML. XML dokument tvoří tagy (značky), elementy a entity. Elementy tvoří logickou strukturu dokumentu,

entity zase fyzickou strukturu. Velmi často se tento jazyk používá také v aplikacích pro výměnu dat mezi dvěma i více systémy díky svému konzistentnímu formátu.[2]

Přiklad jednoduchého XML dokumentu:

```
<?xml version="1.0"?>
<knihovna>
	<kniha id="1">
		<autor>King Stephen</autor>
		<nazev>Noční směna</nazev>
	</kniha>
	<kniha id="2">
		<autor>Němcová Božena</autor>
		<nazev>Babička</nazev>
	</kniha>
</knihovna>
```

1.1.3 C# (C Sharp)

Jazyk C# je vysokoúrovňový objektově orientovaný programovací jazyk, vyvinut společností Microsoft spolu s platformou .NET Framework. První verze byla vydána v roce 2002, v současné době se využívá verze 3.0, která byla vydána spolu s .NET Frameworkem 3.5 a Visual Studiem 2008. Letos vyšla nová verze 4.0, opět spolu s novou verzí .NET Frameworku a Visual Studiem. Mnoho programátorů tak začne přecházet na novější verzi. Tento jazyk je přímým následníkem jazyka C++. Je vhodný jak pro tvorbu formulářových aplikací pro Windows, tak pro webové služby a webové stránky.

Současný design C# je definován standardem ECMA a to takto [3]:

1) C# je jednoduchý, moderní, mnohoúčelový a objektově orientovaný programovací jazyk.
2) Jazyk a jeho implementace poskytuje podporu pro principy softwarového inženýrství, jako jsou: hlídání hranic polí, detekce použití neinicializovaných proměnných a automatický garbage collector. Důležité jsou také jeho vlastnosti jako: robustnost, trvanlivost a programátorská produktivita.
3) Jazyk je vhodný pro vývoj softwarových komponent distribuovaných v různých prostředích.
4) Přenositelnost zdrojového kódu je velmi důležitá, hlavně pro programátory, kteří jsou obeznámeni s C a C++.
5) Mezinárodní podpora je též velmi důležitá.
6) C# je navržen pro psaní aplikací jak pro zařízení se sofistikovanými operačními systémy, tak pro zařízení s omezenými možnostmi.
7) Přestože by programy psané v C# neměly plýtvat s přiděleným procesorovým časem a pamětí, nemohou se měřit s aplikacemi psanými v C nebo jazyce symbolických adres.

Jednoduchý přiklad aplikace v C#, která zobrazí text „Hello world“ po stisknutí tlačítka:

```
using System;
using System.Windows.Forms;
using System.Drawing;

public class HelloWorldForm : Form
{
    Label MyLabel = new Label();
    Button MyButton = new Button();

    public HelloWorldForm()
    {
        Text = "Hello world form";

        MyLabel = new Label();
        MyLabel.Text = "Hello world";
        MyLabel.Location = new Point(100, 100);
        MyLabel.Visible = false;

        MyButton = new Button();
        MyButton.Text = "Show label";
        MyButton.Location = new Point(100, 200);
        MyButton.Click += new EventHandler(MyButton_Click);


        Controls.Add(MyLabel);
        Controls.Add(MyButton);


    }

    void MyButton_Click(object sender, EventArgs e)
    {
        MyLabel.Visible = true;
    }

    [STAThread]
    public static void Main()
    {
        HelloWorldForm helloform = new HelloWorldForm();
        Application.Run(helloform);
    }
}
```

V případě, že programátor již má znalost jazyka C++ není pro něj problém chápat i jazyk C#, jelikož jsou si dost podobné.

1.1.4 .NET Framework

Jak už bylo výše zmíněno, jedná se o softwarovou platformu společnosti Microsoft. Tato platforma by se z pohledu programátora dala přirovnat k systému Windows. .NET Framework je stejně jako Windows především knihovnou, poskytující přístup k velkému množství již implementovaných metod a funkcí dostupných při programování. Stejně jako Windows API lze i .NET Framework využít k ulehčení, protože spoustu věcí není třeba znova programovat. A také stejně jako je Windows prostředí pro spouštění různých aplikací, tak i .NET obsahuje operační prostředí (.NET runtime) pro

spouštění programů [19]. Ačkoliv se zdá, že tato platforma je dostupná pouze pro systém Windows není tomu tak. V současné době již existují nástroje umožňující využívat .NET Framework i na systémech Linux či Mac [4]. Základní funkcionalitu .NETu zajišťuje CLR (Common Language Runtime) na kterém jsou postaveny všechny základní knihovny a objekty [5]. Nejznámější využívané programovací jazyky jsou C# a Visual Basic.

1.1.5 Webové služby

Webové služby umožňují volání vzdálených metod přes protokol HTTP. Jsou proto široce používány jako prostředník distribuce dat mezi systémy. K výměně dat využívá protokol SOAP (Simple Object Access Protocol) [6], který je založen na standardu XML. K popisu webové služby se používá jazyk WSDL (Web Service Description Language) [7]. Webové služby dokážou operovat s mnoha datovými typy, od těch základních jako jsou `string` a `integer`, až po složitější objekty jako například `DataSet` nebo `XmlDocument`. Hlavní výhodou webových služeb je schopnost provozu na jakékoliv platformě s přístupem k protokolu HTTP.

1.1.6 Regulární výrazy

Regulární výrazy jsou mocnou pomůckou programátora. Používají se napříč celým spektrem programovacích jazyků. Je to speciální řetězec znaků, který představuje určitý vzor pro textové řetězce. Existuje více druhů regulárních jazyků – nejznámější jsou však Perl-compatible regulární výrazy a POSIX regulární výrazy [8]. Jejich síla je především ve zjednodušení kódu, i jednoduchý regulární výraz dokáže zkrátit kód o několik řádků. Prostřednictvím regulárním výrazů lze vyhledávat a nahrazovat data podle určených pravidel z textu. V tomto projektu je budeme využívat k vytahování některých dat v hlavičce HTML stránky a hledání odkazů v textových částích webové prezentace.

1.2 Použité programy

1.2.1 Microsoft Visual Studio 2008

Microsoft Visual Studio je aplikace pro vývoj pokročilých nástrojů a programů nejen v C# a platformě .NET framework. Umožňuje vývoj jak konzolových aplikací, tak i webové prostředí mu není cizí. Vývojové prostředí podporuje také vestavený FTP klient pro přístup na vzdálené servery a také možnosti připojení se na databázový server. Tento víceúčelový vývojový nástroj byl použit při většině řešení této práce.

1.2.2 SQL Server Managment Studio 2008

SQL Server Managment Studio je nástroj pro kompletní správu SQL Serveru, jehož je součástí. Tato aplikace vyniká svou přehledností. Centrem této aplikace je Object Explorer, který zpřístupňuje většinu základních operací s databázemi a jejich tabulkami. Tato aplikace byla velice nápomocná při testování ukládání dat do databáze Kentico CMS.

2 CMS Systémy

2.1 Co to je CMS Systém

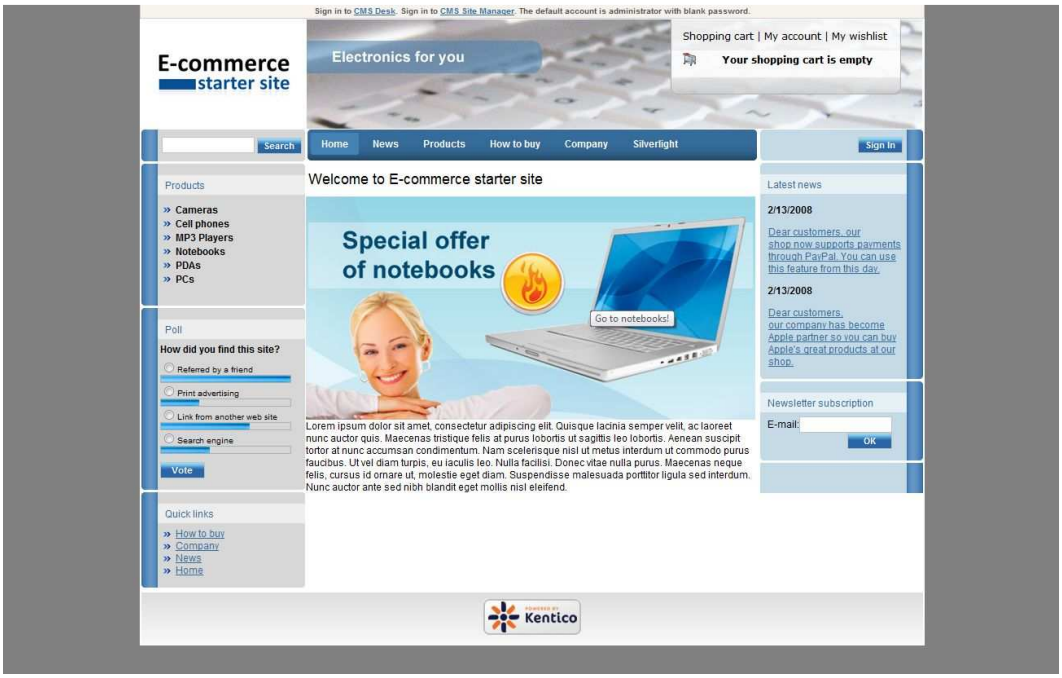
Tradiční statické webové prezentace psané pouze v HTML v dnešním moderním světě již neumí nabídnout odpovídající obsah. Nastává tedy problém, jak do webových stránek zakomponovat nějaký pokročilejší interaktivní obsah, jako třeba fórum, nebo dokonce vytvořit celý komunitní web. Tento neduh částečně odstranily nadstavby HTML ve formě PHP či ASP. Nevýhodou těchto jazyků z pohledu uživatele je nutná znalost programování v těchto jazycích. A právě toto řeší CMS systémy. Tvorba pokročilé webové aplikace je obalena do uživatelsky přívětivého prostředí a celá webová prezentace se skládá jako skládačka. Není proto vůbec problém dodatečně na stránku přidat diskuzi, obrázkovou galerii nebo dokumentu nastavit nějaké workflow či přístupové právo. CMS systém většinou nebývá nic jiného než webová aplikace, přístup k systému je zajištěn jakýmkoliv webovým prohlížečem. Jednoduchý redakční systém lze napsat i v JavaScriptu či v Javě [9]. Pokročilejší CMS systémy jsou programovány buďto v PHP nebo ASP.NET a bývají doplněny o databázový systém (MySQL, MS SQL, Oracle).

Základní funkce CMS systému [10]:

1) Kompletní správa dokumentů a operace s nimi. Uživatel může jednoduše vytvářet a modifikovat jakýkoliv dokument za pomocí WYSIWYG editoru, není tedy nutná znalost HTML
2) Řízení přístupu k dokumentům (workflow), správa uživatelů, jejich rolí a přístupových práv. Administrátor může lehce nastavit hierarchii schvalování a publikování dokumentů (např. autor – editor - korektor)
3) Správa diskuzí, blogů či jiných komunitních prvků webu
4) Správa obrázkových galerií
5) Počítání statistik přístupů na web
6) Internetový obchod
7) Správa formulářů a anket

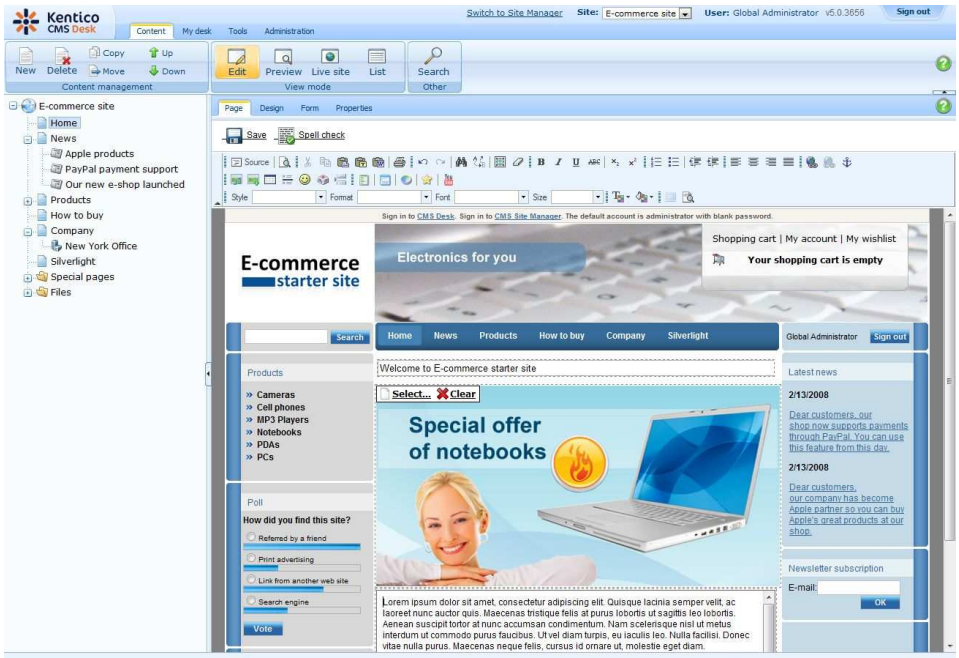
Takový CMS systém se obecně dělí na dvě části, administrátorskou a uživatelskou. Administrátorská část je nadřazena té uživatelské a umožňuje kompletní správu celého systému, od základního nastavení webových stránek až po správu uživatelů, jejich rolí a přístupových práv. Tvorba designu bývá u takových systému rovněž jednoduchá a intuitivní.

Na obrázku 1 je stránka jak ji vidí normální návštěvník. Jak je vidět, neliší se svým vzhledem od statických webů skoro vůbec.



Obr. 1: *Takto vidí stránku normální návštěvník*

V případě, že se do systému přihlásí uživatel s dostatečnými právy, nabídne se mu možnost operovat s dokumenty, měnit design stránky nebo přidávat na stránku nové funkční prvky stránky (fórum, anketu atd.) jak je vidět na obrázku 2.



Obr. 2: *Takto ji vidí administrátor*

Mezi další vlastnosti CMS systému patří také snadná modulárnost systému. Není problém vytvořit nový modul a přidat jej do stávající aplikace, čímž se může rozšířit funkčnost o nové možnosti. Mnohé CMS systémy takové moduly poskytují ke stažení přímo na svých stránkách.

Výhodou CMS systémů založených na jazyce PHP je jejich snadná dostupnost a použitelnost. Jazyk PHP je šířen pod Open Source licencí [11], což umožňuje volné použití jazyka. V kombinaci s databázovým serverem MySQL, který je dostupný také zdarma, tvoří velmi mocný nástroj k tvorbě pokročilých webových aplikací. K běhu takových CMS systémů je nutný webový server Apache. Tento server je běžně ke stažení zdarma v rámci speciálních aplikací, obsahující jak Apache server, tak i podporu PHP a rovnou i MySQL [12].

Druhá skupina, tvořená CMS systémy naprogramovaných na platformě .NET je založena spíše komerčním směrem. Typicky jsou vytvářeny v prostředí Windows, které není zadarmo. Nejrozšířenějším vývojovým prostředím je zde Visual Studio, nabízené sice zdarma v edici Express, ovšem pro pokročilejší aplikace není moc praktické. Taktéž webový hosting bývá více finančně náročný, než hosting pro PHP webové aplikace.

Mezi nejznámější CMS systémy patří [13]:

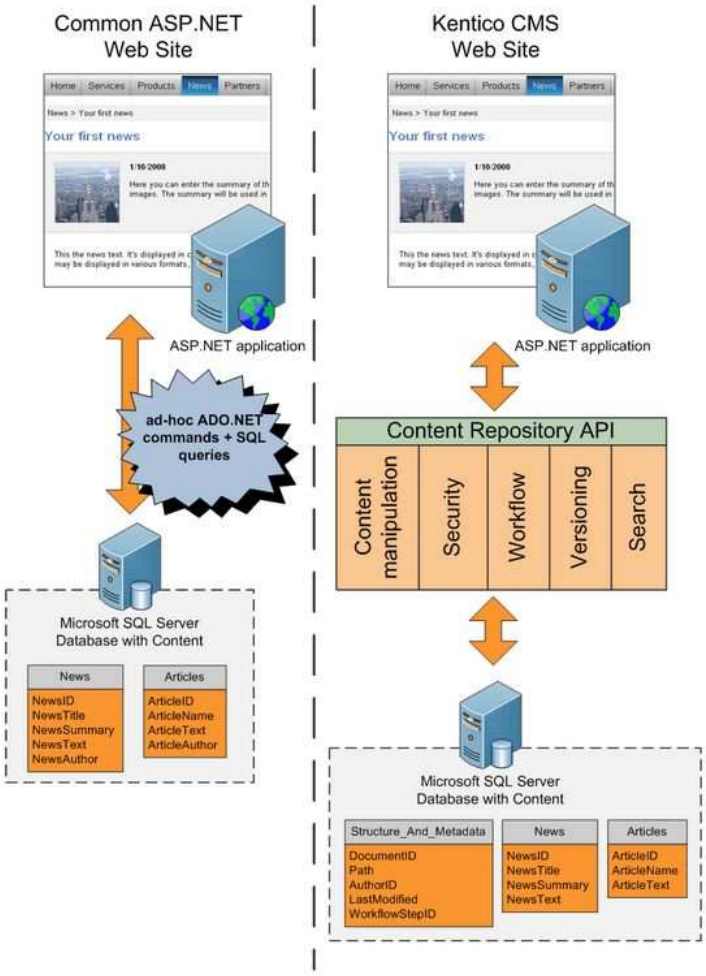
- Kentico (ASP.NET)
- Joomla (PHP)
- Drupal (PHP)
- Ektron (ASP.NET)
- Sitefinity (ASP.NET)
- DotNetNuke (ASP.NET)
- Sitecore (ASP.NET)

2.2 Kentico CMS

V této kapitole se seznámíme blíže s CMS systémem, pro který bude výsledná konverzní aplikace primárně určena.

2.2.1 Seznámení

Firma Kentico vyvíjí svůj CMS systém již šestým rokem. Za tuto dobu se tento systém stal jedním z nejvyužívanějších CMS systému pro malé a střední podniky. Za tímto úspěchem stojí určitě kvalitní práce vývojářů a pravidelné vydávání nových verzí. Letos vyšla už verze 5.5, která obsahuje mimo jiných nových funkcí i podporu nové verze .NET Frameworku 4.0. Velice dobře je tento systém hodnocen i odbornými kritiky. Tato aplikace využívá jako svůj datový sklad databázový program Microsoft SQL Server. Mezi další přednosti patří také snadná přizpůsobitelnost potřebám zákazníka.



Obr. 3: *Znázornění rozdílu mezi klasickou ASP.NET stránkou či webem a Kentico CMS [20].*

Kentico CMS disponuje celou řadou vlastností, které ve spojení s množstvím rozšiřujících modulů, tvoří pokročilý redakční systém. Ke snadné práci s textovým obsahem lze plně využít WYSIWYG editor, který je integrován do celého systému. K jednoduché tvorbě vzhledu stránek slouží v tomto systému šablony stránek, kde se jednoduše nadefinuje vzhled prostřednictvím obyčejného HTML kódu. Zároveň můžou být použity pro více stránek podobného typu. Obsah webové prezentace může být rozdělen na více druhů díky podpoře dokumentových typů. Můžete si tak vytvořit jiný dokumentový typ například pro psaní informačních článků a jiný k prezentaci nových zájmů vaší firmy. Součástí této správy obsahu je možnost definování pracovních procesů vytvářené stránky, kdy se na vydání nějakého dokumentu podílí více osob.

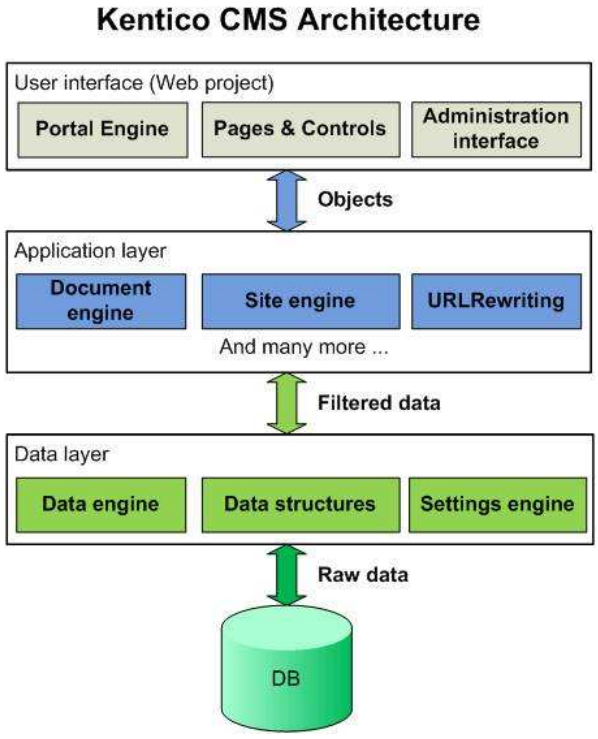
Použití tohoto systému je velice široké díky více než 20 modulům, které umožňují rychlý vývoj webových prezentací i uživatelům bez zkušeností s programováním. Každý z modulů obsahuje také množství webpart, což jsou funkční prvky, které stránce zajišťují rozšíření funkčnosti. Mezi hlavní moduly lze řadit E-commerce. Ten umožňuje kompletní správu internetového obchodu. Realizace nákupního procesu je zde otázkou dvou použitých webpart na stránkách. Další velkou

výhodou tohoto systému je možnost vytvoření komunitního portálu s podporou skupin, které mezi sebou můžou sdílet mimo jiné taky mediální galerie nebo si vyměňovat své názory a poznatky v diskuzích. Mezi další moduly patří také ankety, reporty, statistiky přístupů a mnoho dalších.

Redakční systém by měl řešit samozřejmě i otázky bezpečnosti. Tu v Kentico CMS řeší podpora jak klasické Forms autentizace (přihlašování přes webový formulář) tak i Windows autentizace (automatické přihlašování na základě účtu v Active directory systému Windows). Tyto autentizace doplňuje využití přístupových práv jednotlivých rolí a uživatelů přímo v Kentico CMS. Mezi další hlavní funkce patří také podpora SEO optimalizací. Výchozím jazykem používaným v Kentico CMS je angličtina, nicméně součástí je také mezinárodní podpora a lokalizace, což umožňuje tvorbu vícejazyčných prezentací a také zpřístupnění administračního rozhraní uživatelům bez znalosti anglického jazyka.[14]

2.2.2 Architektura

Jak již bylo dříve zmíněno Kentico CMS využívá .NET Framework a databázi Microsoft SQL Server. Celý systém je naprogramovaný v jazyce C#, což zajišťuje širokou funkcionalitu, kompatibilitu a vysoký potenciál do budoucna.



Obr. 4: *Architektura Kentico CMS [22]*

Jak je z obrázku 4 vidět, architektura je složena ze tří základních celků. Data jsou uložené v databázi a každý typ dat má svou tabulku. První vrstva nacházející se nad databází je vrstva datová (Data layer), která dotazy na databázi získává potřebná data a filtruje je. Tato vyfiltrované data se předají

aplikační vrstvě (Application layer), kde se zpracují do objektů a předají uživatelskému rozhraní (User interface). Uživatelským rozhraním se rozumí to, co je vidět v prohlížeči, ať už je to třeba výpis uživatelů nebo příspěvek v diskuzi.

Z pohledu konverzní aplikace nás budou zajímat pouze první dvě vrstvy, tedy datová a aplikační. Uživatelské rozhraní potřebovat nebudeme, protože celý konverzní proces je řízen nastavením WinForms aplikace, o které bude řeč později, a modul pro Kentico CMS žádné uživatelské rozhraní nemá.

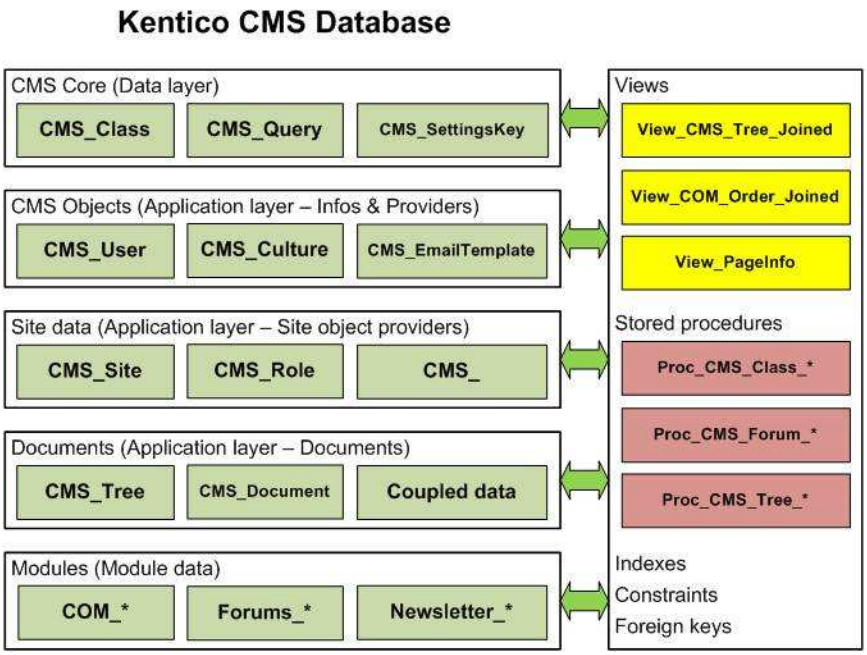
2.2.3 Databáze

Veškerá data, která tento systém uchovává, jsou uloženy v tabulkách v databázovém serveru Microsoft SQL Server. Kentico CMS 5.0, pro kterou bude zadaná úloha programována, podporuje SQL Server verze 2005 a 2008, včetně SQL Server Express Edition. V tabulce 1 je jednoduchý přehled rozdělení některých tabulek.

Příklad tabulek tvořící jádro systému	
CMS_Class	Informace o všech datových typech
CMS_Query	Seznam SQL dotazů pro všechny datové typy
CMS_SettingsKey	Seznam všech obecných nastavení
Příklad tabulek definující CMS objekty	
CMS_Role	Obsahuje záznamy o rolích uživatelů
CMS_User	Obsahuje seznam všech uživatelů
CMS_Webpart	Obsahuje seznam všech funkčních prvků stránky
Příklad tabulek obsahující data modulů	
COM_*	Tabulky modulu E-commerce
Analytics_*	Tabulky modulu Web analytics
Forum_*	Tabulky modulu Forums
Tabulky obsahující data dokumentů	
CMS_Tree	Určuje strukturu stránek webové prezentace
CMS_Document	Zde se ukládá textový obsah stránek
CONTENT_*	Tabulky speciálních dokumentových typů

Tab. 1: *Základní rozdělení tabulek v databázi Kentico CMS*

Na obrázku 5 vidíme, že součástí celé databázové struktury nejsou pouze tabulky. Jsou zde definovány také pohledy (ang. Views) a procedury. Tyto dodatečné položky umožňují snadnější a rychlejší práci se systémem.



Obr. 5: *Struktura databáze Kentico CMS [23]*

V rámci zadané úlohy, převodu statických HTML stránek do CMS systému, nás budou zajímat obzvláště dokumentové tabulky obsahující textový obsah stránek a tabulky definující vzhled stránky.

2.2.4 Kentico API

Kentico API [21] umožňuje programátorovi přístup k funkcím, které jsou v Kentico CMS zastoupeny některým z vizuálních prvků systému. Může tak programově například mazat objektů nebo měnit obsah dokumentu, aniž by musel k objektu nebo dokumentu přistupovat přes uživatelské rozhraní. Kentico API je soubor všech knihoven, díky kterým tento systém funguje a zároveň poskytuje možnost napojení jiné aplikace. Platí, že každý modul má svoji knihovnu, která zajišťuje jeho funkčnost. Toto pravidlo samozřejmě nevylučuje využití knihovny jiným modulem.

Pro lepší a rychlejší orientaci má toto API následující pravidla pro psaní samotného kódu [15] :

- *Jmenné prostory (namespace)* – kód v knihovnách je rozdělený do jmenných prostorů, všechny začínají prefixem „**CMS.**“
- *Samostatné třídy* – každá třída má samostatný soubor, pouze některé enumerace se nacházejí přímo v souborech tříd, aby se nezvyšoval zbytečně počet souborů
- *Jmenné konvence* – každá knihovna, metoda, třída, enumerace nebo proměnná má svoji vlastní jmennou konvenci
- *Konstrukce kódu* – vždy se podmínka vkládá do závorek, i když je jenom jedna, což pomáhá vyhýbat se chybám při aktualizaci kódu

- *Komentáře* – každá metoda, konstruktor nebo část kódu musí být patřičně komentována pro lehčí pochopení daného procesu
- *Regiony* – většina kódu používá regiony k zpřehlednění struktury a rychlejší navigaci k metodám
- *Upozornění* – celý kód je kompilovaný se stupněm upozornění nastaveným na 4 bez žádných hlášení
- *Přistup k datům* – přístup k datům z kódu je striktně řízen přes datovou vrstvu
- *HTML a Javascript* – HTML a Javascript jsou testované, aby splňovali specifikace a fungovali ve všech prohlížečích

Pro většinu metod platí, že nevrací standardní datový typ, ale vrací typy speciálně vytvořené pro Kentico CMS. Základní objekty tvoří třídy s koncovkou v názvu „Info“ (například CssStylesheetInfo), které obsahují definici objektu, ale nemají žádné metody. Metody, které s daným objektem pracují, jsou uloženy ve třídě s koncovým „InfoProvider“ v názvu (například CssStylesheetInfoProvider).

2.3 Ostatní CMS Systémy

Mezi nejbližší konkurenční CMS systémy patří redakční systém Sitecore, postavený na stejných základech jako systém firmy Kentico. Oba systémy tedy fungují na .NET Frameworku, implementační jazykem je C# a jako datový sklad využívají Microsoft SQL Server. Rozdíly lze však najít poměrně snadno, a to především v nabízeném portfoliu funkcí. Sitecore je déle na trhu a má proto lepší pozici na trhu. Výsledkem je to, že Sitecore má ve všem náskok. Nicméně i Kentico, jakožto o dost mladší produkt, si snaží najít své zákazníky. V počtu nabízených modulů se k Sitecore blíží velmi blízko, ztrácí pouze v některých specifických funkcích jako je podpora WebDAV nebo možnosti exportovat vytvořený web zpátky do statické HTML prezentace [13]. Daří se mu zvedat popularitu také díky zavedení takzvané Bug fixing policy – dodržení 7 denní doby na opravu chyby ve vydané verzi. Jedna z mála věcí, čím má Kentico navrch je jeho cena. Ta je v základním balíčku skoro 7x menší než u podobného řešení u Sitecore.

Kentico i Sitecore jsou produkty s uzavřeným zdrojovým kódem a tudíž silně komerční. Proto porovnáme Kentico CMS i s čistě open source variantou, redakčním systémem Joomla!. Tento redakční systém je napsán v PHP a jako databázi využívá MySQL. Výhodou takového open source systému je jednoznačně jeho cena, která je v podstatě nulová. Takovéto systémy stoupají a padají s podporou uživatelů. Díky velmi silné komunitě uživatelů-programátorů se svými funkcemi Joomla! blíží komerčním systémům. Problémem však bývá absence oficiální podpory, jakou známe například u Kentico nebo Sitecore, a proto může trvat oprava chyb delší dobu a systém tak může být zranitelný vůči hackerům.

3 Implementace

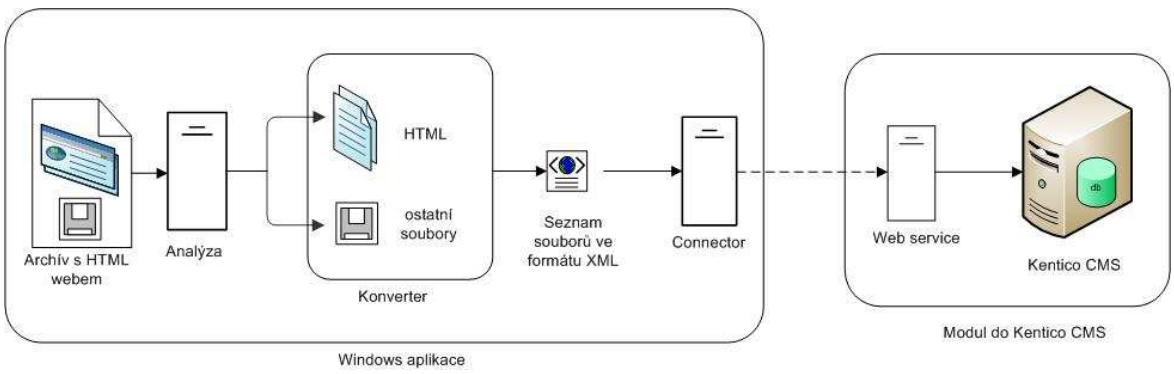
V následující kapitole popíšeme návrh modelu konverze statických HTML stránek do projektu Kentico CMS a návrh architektury převodní aplikace. Zmíníme se o programech používaných při řešení úlohy.

3.1 Konverzní model

Každý návrh konverze něčeho někam je problematický proces, nad kterým je nutné se pořádně zamyslet. Statická webová prezentace je kombinace několika různých typů souborů, které tvoří výsledný celek. Kvalitní konverze by měla být obecná a měla by bez výjimek fungovat při splnění základních podmínek vždy. Naší hlavní podmínkou je použití statického HTML webu jako vstupu do procesu konverze. Pro dosáhnutí správného výsledku musí být kód této webové prezentace validní. Jako vedlejší podmínky můžeme brát správné nastavení připojení na vzdálený server.

Navrhnutý konverzní model je rozdělen na dvě části. První část je ve formě Windows aplikace. Tato aplikace zajistí přeměnu každého souboru na speciální formát dat ve formě XML dokumentu. Tento dokument se odešle na vzdálený server s Kentico CMS, kde uložení do databáze tohoto CMS systému zajistí druhá část konverzního modelu. Tato druhá část se nachází v Kentico CMS ve formě knihovny. Přenos dat mezi těmito částmi zajišťuje služba z .NET Framework zvaná Web service.

Proces konverze je tedy následující. Jako data se vloží do aplikace archív s kompletní webovou prezentací. Soubory se roztřídí na HTML stránky a ostatní obsah, což jsou obrázky, definice stylů, definice skriptů a jiné soubory. Každá HTML stránka se rozdělí na několik částí. První část HTML dokumentu je jeho hlavička, druhá je definice vzhledu a třetí textový obsah. Z těchto částí se vytvoří objekt ve formátu XML, ve kterém budou všechny potřebné informace pro přenos. Stejný osud potká také ostatní soubory. Vznikne tak seznam XML dokumentů, které se odešlou na adresu zadaného serveru. Na serveru se provede opačný konverzní proces, tedy z XML dokumentu se vytvoří objekt vyhovující Kentico CMS.



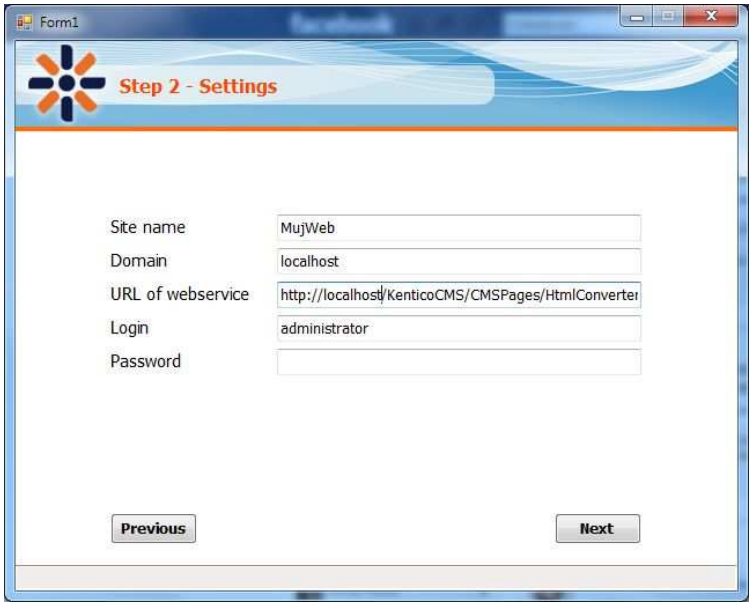
Obr. 6: *Konverzní model*

3.2 Návrh Windows aplikace

Z konverzního modelu je jasné, co všechno tato aplikace musí umět. Jedná se o WinForms aplikaci, tedy aplikaci, běžící přímo pod operačním systémem Windows. Tato aplikace musí umět přijmout archív s webovou prezentací, zpracovat jej, z obsahu souboru vytvořit XML dokument a zajistit jeho poslání na vzdálený server. Zároveň je však aplikace navrhnutá tak, aby nebyla omezena pouze na konverzi do Kentico CMS, ale je možné ji napojit i na jiné aplikace s podporou webové služby. To je zajištěno konverzí souborů webové prezentace do univerzálního formátu ve formě XML souboru, kde jsou informace o HTML souborech uloženy. Popis aplikace rozdělíme do tří částí: uživatelské rozhraní, konverze souborů a posílání na sever.

3.2.1 Uživatelské rozhraní

Uživatelské rozhraní je nedílnou součástí každé aplikace. Toto rozhraní by mělo být co nejpřehlednější a nejjednodušší, aby práci s aplikací ulehčovalo a ne ji činilo obtížnější. Konverzní aplikace je implementována formou průvodce. Každá část má svůj jednoznačný úkol a každá následující část očekává správné provedení předcházejícího kroku.



Obr. 7: *Ukázka uživatelského rozhraní konverzní aplikace*

Z obrázku 7 je jasné rozvržení všech ovládacích prvků. V horní části je grafická hlavička s číslem aktuálního kroku a jeho názvem. Mezi jednotlivými kroky se přepíná tlačítky umístěnými v dolní části. Prostřední část je tvořena obsahem jednotlivých kroků.

Tento průvodce má celkově 5 kroků:

1) Výběr archívu z disku a jeho rozbalení do dočasné složky
2) Nastavení připojení ke vzdálenému serveru a název nového webu
3) Zobrazení stromové struktury zpracovaného webu, zde končí možnost vrátit se zpět
4) Zobrazení průběhu konverze a odesílání webu
5) Ukončení aplikace

V rámci uživatelského rozhraní je třeba zmínit využití třídy `BackgroundWorker` [16]. Tato třída umožňuje spuštění nového pracovního vlákna. Tím dosáhneme fungování uživatelského rozhraní nezávisle na právě prováděné operaci. Pokud by se tato komponenta nepoužila, při časově náročných operací by mohlo dojít k dočasnému „zamrznutí“ uživatelského rozhraní. K přenosu informací mezi pracovním a zobrazovacím vlákně o právě zpracovaných datech a jejich grafické znázornění použijeme třídy `Delegate` a `Event` díky nimž lze volat metody, které jsou běžným způsobem nedostupné stejně jako v našem případě.

3.2.2 Konverze souborů a složek

Část programu zabývající se samotnou konverzí je nejdůležitější a také nejsložitější. Samotná příprava návrhu byla velice problematická. HTML má sice svá pravidla, ale mnoho programátorů je nerespektuje. Proto i v našem případě základní parsovací algoritmus zanedbává některé programátorské techniky používané pro tvorbu webové prezentace. Co přesně tato aplikace ignoruje, si popíšeme níže.

Základem konverze je rozdělení typů souborů na tři základní – složky, HTML stránky a ostatní soubory. Konverze složek je velice jednoduchý proces. Kromě názvu a umístění ve struktuře není potřeba dalších informací. Informace o složce se tedy ze souborového systému načtou do objektu `DirInfo` a za pomocí třídy `XmlProvider` se vytvoří z tohoto objektu XML dokument.

Vzor XML dokumentu pro složku webu:

```
<root>
	<type>dir</type>
	<relativepath>\images</relativepath>
	<name>images</name>
</root>
```

Ostatními soubory jsou myšleny všechny soubory, které nejsou HTML stránky. Jedná se teda například o obrázky, CSS styly v souboru, textové soubory nebo JavaScriptové skripty. Základ je stejný jako při konverzi složek. Do XML souboru se přidají ještě informace o příponě souboru a v závislosti na příponě souboru také data v textovém nebo binárním formátu. Textové data jsou

použity pro soubory s CSS styly. Objekt `FileInfo` pro převod na XML dokument zajistí třída `FileInfoProvider`.

Vzor XML dokumentu pro soubory webu:

```
<root>
       <type>file</type>
       <relativepath>\images\header.png</relativepath>
       <name>header</name>
       <extension>png</extension>
       <binarydata> binární data v Base64 </binarydata>
</root>
```

Zdaleka nejtěžší je konverze samotných HTML stránek. Jak bylo již výše zmíněno, skládá se ze tří částí a v každé části může být téměř jakýkoliv obsah odpovídající danému kontextu. K řešení této konverze byly použity regulární výrazy, knihovna HtmlAgilityPack [17] a rekurze. HtmlAgilityPack je HTML Parser, který zjednodušuje operace s HTML kódem a ulehčuje navigaci k jednotlivým HTML tagům. Ukážeme si tedy na jednoduchém příkladu, jak to bude fungovat. Máme obyčejnou HTML stránku, která ovšem v originálním zdrojovém kódu není kompatibilní s Kentico CMS. Vychází to především ze způsobu uložení dat v tomto systému.

Příklad jednoduché HTML stránky, která nám pomůže pochopit způsob konverze:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="cs">
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta http-equiv="Content-Language" content="cs" />
    <meta name="language" content="cs" />
    <meta name="keywords" content="konverze, test, html, Kentico" />
    <meta name="description" content="Testovaci stranka pro HTML konverzi" />
    <link rel="stylesheet" type="text/css" media="print" href="style.css" />
    <title>Test konverze</title>
    <script type="text/javascript" src="more.js"></script>
</head>
<body>
  <div id="wrapper">
    <div id="header">
      <a href="http://www.kentico.com/" title="kentico.com">
         <img src="images/header.jpg" alt="header" title="header" />
      </a>
    </div>
    <div id="navigation">
      <div id="navigation-global">
         <ul>
               <li><a class="active" href="index.html" title="stranka 0">Hlavní
               stránka</a></li>
               <li><a href="stranka1.html" title="stranka 1">Stránka 1</a></li>
         </ul>
      </div>
    </div>
    <div id="content">
      <span id="breadcrumbs"><a href="index.html">Hlavní stránka</a></span>
      <h1>Testovací stránka pro konverzi do Kentico CMS</h1>
        <p> Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean eleifend augue /p>
      <a href="index.html">odkaz</a>
    </div>
  </div>
    <div id="footer"> <a href="mailto:xstebe01@stud.fit.vutbr.cz">Robert Stebel</a> </div>
</body>
</html>
```

Parsování takové stránky probíhá postupně od začátku dokumentu. Nejdřív se tedy získá verze použitého HTML dokumentu (DOCTYPE). K jeho zjištění se využije jednoduchý regulární výraz. Další potřebná data jsou uloženy v hlavičce HTML stránky. V hlavičce může být mnoho různých typů informací, jako jsou metadata nebo odkazy na externí soubory Javascriptu nebo kaskádových stylů. A protože konverze by měla být obecná, každý tento typ je do XML dokumentu vložen zvlášť. Jako poslední je potřeba projít samotné tělo dokumentu. Jak již víme z kapitoly o Kentico CMS vzhled a obsah stránky se ukládá každé zvlášť, do jiných tabulek v databázi. Proto tento proces musí získat nejen textový obsah stránky ale také zjistit její vzhled. V rámci řešení této úlohy je zanedbána možnost definice vzhledu pomocí tabulek, která se dříve s velkou oblibou používala. Aby tedy tento proces fungoval správně, musí být definován vzhled stránky párovým tagem DIV.

Jako základní vstup parsovacího algoritmu je celý obsah tagu body. Tento algoritmus je zajištěn hlavní metodou `string ParseHTML(string input,bool first)`. Tato metoda má dva parametry. Jedním je HTML kód a druhým je indikátor jestli se jedná o první zavolání nebo volání v rámci rekurze. Výstupem této metody je upravená část zdrojového kódu stránky, která bude postupně tvořit definici vzhledu stránky. Pomocná metoda `string AddContentToArray(string text)` ukládá získaný textový obsah do globální struktury spolu s pořadovým číslem a vrací zástupný text `##CONTENT##==pořadové_číslo==##` použitý v definici vzhledu stránky. Zástupný text využijeme k pozdějšímu zjišťování, kam tento textový obsah patří při opětovném sestavování stránky. V případě, že se metoda `ParseHTML` zavolá poprvé a tedy jako vstup je celý obsah, kontroluje se na začátku a konci stránky existence nějakého textu, který je mimo tag DIV. Poté se vstupní kód načte do objektu definovaného knihovnou HtmlAgilityPack. Následně se vyhledá tag div. Pokud je nalezen, načte se jeho obsah opět do objektu z HtmlAgilityPack. V případě, že jsou zde přítomny vnořené tagy, zavolá se opět metoda `ParseHTML` avšak s rozdílem jeho vstupu. Ten je v tomto případě pouze HTML kód právě zpracovávaného divu. Touto rekurzí se docílí projití celé struktury HTML stránky i pro hodně vnořených DIV tagů. Když se algoritmus zanoří tak hluboko, že právě procházený DIV již nemá žádné jiné vnořené DIV tagy, zavolá se metoda `AddContentToArray` a uloží se obsah tohoto DIVu do seznamu textových obsahů. Volající metodě se vrací upravená část HTML kódu se zástupným textem místo původního obsahu. Ten se přidá do definice vzhledu stránky. Po projití celé stránky máme textový obsah uložený ve struktuře spolu s pořadovým číslem a máme zároveň i definici vzhledu stránky tvořenou DIV tagy a zástupným textem se stejným pořadovým číslem jako má odpovídající text v seznamu. Grafické znázornění celého parsovacího algoritmu je v příloze 2.

Vzor XML dokumentu pro HTML stránky webu:

```
<root>
	<type>html</type>
	<name>test</name>
	<relativepath>\test.html</relativepath>
	<headcontent>
		<doctype> definice verze HTML </doctype>
		<head>
			<metatags> meta tagy </metatags>
			<linktags> link tagy </linktags>
			<scripttags> skript tagy </scripttags>
			<title> Titulek stránky </title>
	</head>
	</headcontent>
	<layout> definice vzhledu složená z DIV tagů a zástupných textů </layout>
	<bodycontent>
		<content number="0"> obsah části 0 </content>
		<content number="1"> obsah části 1 </content>
		<content number="2"> obsah části 2 </content>
	</bodycontent>
</root>
```

3.2.3 Posílání na vzdálený server

Když máme celou webovou prezentaci uloženou v seznamu XML dokumentů, musíme ji dopravit na vzdálený server s Kentico CMS. K tomu využijeme webovou službu. Třída `CMSConnector` za pomocí webové služby, umístěné na vzdáleném serveru zprostředkuje spojení mezi Windows aplikací a naším modulem v Kentico CMS. Komunikace probíhá ve třech krocích. První krok komunikace probíhá po vyplnění přihlašovacích údajů a URL adresy na vzdálenou webovou službu. V tomto kroce se ověří dostupnost této služby na zadané adrese a v případě, že se zde nachází, také opravnění daného uživatele k vytvoření nové webové prezentace. Druhý krok komunikace spočívá v odeslání pokynu k vytvoření prázdné webové prezentace z předdefinované šablony, ke které se zpracované stránky budou přidávat. Tato šablona má vhodně přednastaveny některé vlastnosti tak, aby práce s novým webem byla co nejjednodušší.

3.2.4 Výsledná architektura aplikace

Výsledná architektura aplikace plně vyhovuje třívrstvému návrhu. Skládá se tedy z prezentační, aplikační a datové vrstvy. Prezentační vrstvu zde reprezentuje uživatelské rozhraní. Tedy Windows okno, které dává uživateli možnost ovlivnit nastavení konverze. Uživatelské rozhraní zde tvoří pouze

jedna třída `Form1`. Z této třídy se přímo volají pouze metody z `CMSConnector` a `ParseEngine` o kterých se zmíníme později.

Aplikační vrstva je tvořena většinou přítomných tříd, díky náročnosti některých operací a také zachování přehlednosti a oddělení logických celků do vlastních tříd. Třída `CMSConnector` zajišťuje komunikaci se vzdáleným serverem. Třída `ParseEngine` je vstupní třídou ke konverzní logice. Obsahuje cyklus, kde vstupem je seznam souborů uložených na disku a výstupem převedený seznam souborů v XML formátu. Seznam souborů se získává přes datovou vrstvu. Každý z typů souborů má svůj objekt a třídu, která tento objekt vytváří. Jména těchto tříd jsou odvozeny z pravidel platící pro Kentico API. Máme tak objekty `DirInfo`, `FileInfo` a `HtmlInfo`. Třídy vytvářející tyto objekty jsou `DirInfoProvider`, `FileInfoProvider` a `HtmlInfoProvider`. Vytvořené objekty se převádí do XML metodami z třídy `XmlProvider`, kde pro každý typ objektu je zvlášť metoda.

Datová vrstva je mezičlánek mezi datovým skladem, v našem případě souborový systém na disku, a aplikační vrstvou, která s těmito soubory pracuje. Tato část obsahuje dvě třídy – `ArchiveProvider` a `FileSystemProvider`. První třída je použita k rozbalení archívu s webovou prezentací do dočasné složky, ze které pak třída `FileSystemProvider` bude získávat informace. `FileSystemProvider` tak poskytuje aplikační vrstvě přístup k souborům na disku. Ten je potřeba při vytváření stromové struktury prezentace ve třetím kroku průvodce a pak také hlavně při čtení jednotlivých souborů. Na konci konverze také maže dočasnou složku se soubory, které již nejsou potřeba. Grafické znázornění celé architektury aplikace je přiloženo v příloze 3.

3.3 Návrh modulu v Kentico CMS

Tento modul bude za pomocí webové služby přijímat XML dokumenty s daty webové prezentace a ukládat je do databáze Kentico CMS. Na rozdíl od konverzní aplikace nemá žádné uživatelské rozhraní. K ukládání využijeme již existující metody z Kentico API.

Jak už bylo zmíněno u Windows aplikace, než se do modulu začnou posílat soubory, je zapotřebí ověřit autorizaci daného uživatele a vytvořit ze šablony prázdný web. Při autorizaci je potřeba rozlišovat dvě možnosti uložení hesla – čistý text nebo SHA1 [18]. Vytvoření prázdného webu je řešeno pomocí třídy `CMSImportExport` z Kentico API. V rámci Kentico CMS se jedná o import prázdné webové prezentace.

Struktura tohoto modulu je téměř totožná s konverzní částí Windows aplikace, pouze směr konverze je opačný. Nyní budeme převádět informace z XML dokumentu do databáze Kentico CMS. Stejně jako ve Windows aplikaci jsme rozdělily jednotlivé položky na složky, HTML soubory a ostatní soubory. Pro zjednodušení práce s daty se tyto data z XML dokumentu načtou do objektu odpovídajícího typu souboru. Toto načítání do objektů řeší třída `XmlProvider`. Následně se

objekty předají jejich odpovídající třídě a metodě `AddToSite`, která ukládání do Kentico CMS zajistí.

3.3.1 Ukládání složek

Stejně jako byla jednoduchá konverze složek do XML dokumentu, je i jednoduché uložení tohoto objektu do databáze. V Kentico CMS se realizuje složka pomocí dokumentového typu CMS.Folder. Celá webová prezentace je uložena v databázi a proto se nejedná o složku v pravém slova smyslu, je to pouze její zastoupení pro lepší organizaci struktury prezentace. Data složky se tedy z XML dokumentu načtou do objektu `DirInfo` a předají metodě `AddToSite` ze třídy `DirInfoProvider`. Ta využije třídy obsažené v `CMS.TreeEngine` z Kentico API, pro práci se stromovou strukturou webové prezentace.

3.3.2 Ukládání souborů

Soubory jiné než HTML stránky se ukládají, opět trochu složitěji než složky. Výjimku tvoří pouze soubory šablon CSS stylů, které sice používají stejný XML dokument jako ostatní soubory, ale ukládají se na jiné místo v systému. CSS styly se ukládají do zvláštní tabulky `CMS_CSSStylesheet` a také uživatelské rozhraní v systému je pro ně odlišné. Tyto šablony se budou ukládat do databáze s prefixem názvu nové webové prezentace. Tímto odstraníme možné chyby v případech, kdy by více webů používalo stejný název pro tento soubor se styly. Dále je nutné každý takový soubor projít a upravit adresy na lokální soubory prezentace. Bez této úpravy by se některé obrázky nemusely na webu zobrazit.

Pro odkazování na soubory využívá Kentico CMS tzv. Alias path. Tato Alias path může být odlišná od názvu dokumentu. V případě, že by tato Alias path zůstala stejná, hrozí v systému kolize stejných URL adres, protože dokumenty se zde neukládají s příponami. Pokud se pokusíme uložit stejný název více než jednou, systém tuto duplicitu opraví přidáním pořadového čísla k Alias path. Tato úprava je však pro nás nežádoucí, protože bychom se nemohly na tento upravený dokument odkazovat. Proto tuto Alias path pro soubory při ukládání budeme měnit. Nový formát bude `/<název_souboru>-<přípona_souboru>`, například pro soubor „image.jpg“ bude výsledná Alias path tedy `/image-jpg`. URL na tento soubor se tvoří z Alias path a přípony „.aspx“, bude tedy `/image-jpg.aspx`. K uložení těchto ostatních souborů použijeme dokumentový typ CMS.File, který je bude reprezentovat ve stromové struktuře. Tento dokumentový typ je speciálně upraven pro uchovávání souborů. Data uloženého souboru se ale ukládají ve formě binárních dat v tabulce `CMS_Attachment`. Data načtená do objektu `FileInfo` opět předáme metodě `AddToSite`, tentokrát třídě `FileInfoProvider`. Tato metoda bude velmi podobná té

z `DirInfoProvider` avšak využívá kromě již zmíněných tříd z `CMS.TreeEngine` také další z `CMS.FileManager` pro uložení binárních dat souboru do zvláštní tabulky a vytvoření zpětné reference k CMS.File dokumentu.

3.3.3 Ukládání HTML stránek

Jako poslední popíšeme ukládání samotných HTML stránek. I zde se data z XML dokumentu převedou na objekt, v tomto případě `HtmlInfo` a ten se následně předá metodě `AddToSite` ze třídy `HtmlInfoProvider`. V Kentico CMS se zobrazitelná stránka skládá ze dvou částí, které jsou uloženy v systému zvlášť – obsahu a vzhledu. Tyto části máme v objektu `HtmlInfo` a musíme je patřičně upravit, aby tyto data v Kentico CMS tvořily stejnou stránku, jako byla ta původní v prostém statickém HTML.

Vzhled stránky je v tomto systému definován šablonou stránky. Tato šablona je tvořena HTML kódem stejným jako ve statických prezentacích doplněný o speciální webpart zóny, do kterých se vkládají webparty – funkční bloky, které rozšiřují funkcionalitu webové prezentace. Informace o tom, kde tyto webpart zóny patří, máme uloženy ve formě zástupných textů, které jsme získaly v konverzní aplikaci. Zde tyto zástupné texty vyměníme za definici webpart zóny. Každá zóna bude obsahovat jednu editační webpartu. Navrhnutý modul umí rozpoznat dva typy editační webparty – text a obrázek. Zde se však ukládá pouze nastavení těchto webpart. Do této šablony se také budou ukládat informace, které jsme získaly z hlavičky HTML souboru – tedy meta, script a link tagy. Pro každou HTML stránku se tak vytvoří šablona, která bude dokumentu definovat vzhled.

Při ukládání obsahu do systému Kentico CMS je potřeba vzít na vědomí změnu URL adres, protože původní odkazy by nám tady nefungovaly. Proto je potřeba tyto adresy upravit na nový formát. K tomu využijeme regulárních výrazů. Každý prvek ze seznamu textu prohledáme na výskyt odkazů a ty přepíšeme. Na HTML stránce rozlišujeme tři typy odkazů – na HTML dokumenty, CSS styly a na ostatní obrázky. Z odkazů na jiné HTML stránky se odstraní stará přípona a pomocí `TreeUtils.GetSafeUrlPath` metody z `CMS.TreeEngine` se získá nová adresa a připojí se k ní nová koncovka „.aspx“. Odkazy na externí CSS styly, které jsme v našem systému uložily do tabulky `CMS_CSSStylesheet`, nahradíme `~/CMSPages/GetCSS.aspx?stylesheetname=<nový_název_CSS_stylu>`. Tímto dosáhneme textového zobrazení stylů a jejich správnou aplikaci na webovou stránku. Odkazy na ostatní soubory, jak už bylo zmíněno výše, převedeme na formát `/<název_souboru>-<přípona>.aspx`. Takto upravené texty se ukládají pomocí rozhraní třídy `EditableItems` k nově vytvořenému dokumentu. Navržený algoritmus umí poznat dva typy textové informace. V případě, že se za pomocí regulárního výrazu rozpozná zpracovávaný zdrojový kód pouze jako tag obrázku <img> s adresou na obrázek, který může být obalen odkazovacím párovým tagem <a>, použije se editační webparta pro obrázek. V opačném případě se na stránku přidá webparta pro editaci obecného textu. Po upravení a následném uložení všech stránek

a jejich částí do Kentico CMS, je tato webová prezentace připravena plnit své funkce z prostředí redakčního systému.

3.3.4 Výsledná architektura modulu

Architektura tohoto modulu do Kentico CMS je velmi podobná Windows aplikaci. Modul nemá žádné uživatelské rozhraní a tedy ani prezentační vrstvu. Jak už bylo dříve zmíněno, obsahuje třídu webové službu `HtmlConverterService`, která přijímá požadavky z Windows aplikace a předává je třídě `HtmlConverterProvider` ke zpracování. Tato třída tedy obsahuje metody na zjištění oprávnění uživatele na vytvoření nové webové prezentace, vytvoření základní kostry webu a přijmutí jednotlivých souborů.

Soubory se z XML dokumentu načtou do odpovídajících objektů ve třídě `XmlProvider`. Znovu tak získáme objekty `DirInfo`, `FileInfo` a `HtmlInfo`, které dědí základní vlastnosti z třídy `ItemInfo`. Každý z objektů se v odpovídajících třídách `DirInfoProvider`, `FileInfoProvider` a `HtmlInfoProvider` metodou `AddToSite` uloží do datového skladu. K napojení na datový sklad Kentico CMS je plně využito Kentico API. Součásti modulu je pomocná třída `ConverterHelper`, která obsahuje pomocné funkce k logování chyb v prostředí tohoto redakčního systému a získání cesty k rodičovskému dokumentu pod který se nový dokument bude vkládat. Grafické znázornění celé architektury implementovaného modulu je přiloženo v příloze 4. Zdrojový kód webové služby je v příloze 5.

3.4 Testování

Testování probíhalo na třech webových prezentacích různého zaměření a také vzhledu. Každá z prezentací byla od jiného autora, kvůli odlišným programovacím návykům tvorby webu. To pomohlo k lepšímu otestování. Právě testování různých druhů vzhledu vedlo v tomto projektu v co možná nejlepší odladění aplikace. Testování probíhalo postupně při implementaci a podařilo se tak velkou část problémů odladit bez velkých potíží a přepisování kódu. Správné fungování celého projektu bylo testováno pomocí ladícího módu Microsoft Visual Studia. Ke kontrole správného ukládání dat v databázi Kentico CMS byla využita aplikace SQL Server Managment Studio. Vizuální stránka webové prezentace se lehce kontrolovala přes jakýkoliv webový prohlížeč s otevřeným redakčním systémem Kentico CMS. Tímto testováním byly odhaleny nejen chyby v návrhu tohoto projektu ale i některé nedostatky Kentico CMS. Protože se jedná o webový redakční systém, který jako každý jiný webový systém využívá kaskádové styly, může na některých místech dojít k nepříjemné deformaci uživatelského rozhraní z důvodu kombinace více definic kaskádových stylů.

3.5 Možná vylepšení

Zde shrneme další možný vývoj projektu, směrem k lepší funkcionalitě. Tyto nápady vyvstaly při řešení celého projektu.

3.5.1 Rozšíření typů vstupu

Konverzní aplikace, tak jak byla navrhnuta, počítá pouze se vstupem ve formě archívu s webovou prezentací. Do budoucna by bylo možné rozšířit možnosti vstupu například o výběr samotné složky na disku nebo možnost FTP přístupu k uložené prezentaci. Další z možných vstupů zahrnuje vložení přímé WWW adresy k hlavní stránce prezentace. Tato možnost však s sebou nese další problémy k řešení. Ten hlavní je zajištění průchodu přes všechny soubory, i když na začátku nevíme jejich přesný počet či umístění.

3.5.2 Detekce hlavičky a patičky prezentace

Problém statických webů je především v duplicitě zdrojového kódu. Hlavička a patička, nebo jakákoliv jiná část webu, která se na všech stránkách opakuje, musí být přítomna v každém souboru. Výhodou stránek vytvářených v pokročilejších technologiích, například v PHP, je možnost vytvořit zvlášť stránky reprezentující hlavičku a patičku stránky a ty pak do každé stránky vkládat. Výhodou je méně duplicitního kódu a v případě změny se tato změna děje pouze na jednom místě. To samé nabízí i Kentico CMS. Nabízejí se dvě varianty jak toho docílit v budoucnu i u tohoto projektu. Tou jednodušší možností by bylo manuální označení v každé stránce. Uživatel by na každé stránce označil co je hlavička a patička webové stránky. Avšak tím by se trochu ztratilo z automatické konverze. Tou složitější možností a zachování plně automatické konverze, by mohlo být porovnávání obsahu všech stránek a vygenerování kódu, který by se opakoval. Ten by se vložil na hlavní stránku, v Kentico CMS označovaném jako „root“. Zásadní otázkou se jeví stupeň přesnosti u takto duplicitních zdrojových kódů, kdy v případě i malé odchylky může konverze skončit neúspěchem.

3.5.3 Lepší rozpoznávací schopnost formátování

Jak už bylo zmíněno, tato konverzní aplikace rozpoznává vzhled webové stránky na základě formátovacích tagů DIV. Nicméně najde se ještě v dnešní době hodně statických prezentací, které místo DIVů používají formátování pomocí tabulek. Rozšířením stávajícího algoritmu o rozpoznávání i těchto tabulkových vzhledů by se docílilo větší univerzálnosti. V souvislosti s touto možností by však bylo potřeba vyřešit rozpoznání, zda se skutečně jedná o vzhled definovaný tabulkami nebo jestli se jedná obyčejnou tabulku.

Další úskalí takových statických prezentací bývá jejich validita. Tento projekt počítá s validním kódem a v případě nedodržení nemůže zaručit správnou funkčnost. Proto by rozpoznávací algoritmus mohl v budoucnu být „chytřejší“ a umět některé nevalidní kódy sám opravit.

4 Závěr

Tento projekt byl víceméně experimentálního zaměření zadaný firmou Kentico. Zabýval se možným rozšířením o import dnes už téměř nepoužívaných statických webových prezentací. Hlavní otázkou zde bylo, jestli je tato konverze vůbec možná, a pokud ano, do jaké míry. Při řešení tohoto úkolu bylo potřeba řešit množství překážek a do jisté míry účelně zanedbávat některé části. Tyto části nepřímo souvisely s konverzí a jejím řešením by se tento projekt prohloubil k řešením přesahující rámec této práce. Avšak aby se na ně nezapomnělo, byly shrnuty v části popisující možná rozšíření. Dosažené výsledky ukázali, že tato konverze je možná, v případě že zdrojová prezentace splňuje požadavky dané navrhnutou architekturou. Další vývoj aplikace bude potřeba směřovat cestou větší citlivosti v otázce rozpoznávání HTML kódu. To by vedlo k větší univerzálnosti a širšímu uplatnění v praxi.

Výhodou této konverze je fakt, že převedené statické stránky se na první pohled tváří jako by se nic nezměnilo. Když se ale uživatel přihlásí do uživatelského rozhraní Kentico CMS, najednou má daleko více možností než měl předtím. Texty na svých stránkách může editovat za pomocí WYSIWYG editoru a přímo vidí, jak bude text vypadat. Rozšíření stránek o diskuzi nebo interaktivní obrázkovou galerii zvládne během chvilky. V rámci správy obsahu je tedy práce s převedenou webovou prezentací o mnoho přívětivější.

Samozřejmě má tento projekt i řadu nevýhod. Hlavní nevýhodou ve stávajícím řešení, pokud neuvažujeme již vzpomínané možnosti vylepšení, je správa vzhledu webové prezentace. Tím, že aplikace neumí rozeznat opakující se hlavní části, je editace vzhledu stejně zdlouhavá, jako bývala. Další nevýhoda se blíží k otázce, zda tato aplikace bude pro potenciálního uživatele přínosná. Pokud má někdo v dnešní době statickou prezentaci, svědčí to o určité zastaralosti a při přechodu na redakční systém si nechá udělat webovou prezentaci úplně novou.

Díky zajištěné univerzálnosti konverzní aplikace nemusí být Kentico CMS jediným redakčním systémem na který je tento projekt aplikovatelný. Může se tak jednat o základ konverzních nástrojů pro webové aplikace, protože podobný nástroj v současné době nenabízí žádný ze známých redakčních systémů.

Přínosem této práce je rozšíření redakčního systému Kentico CMS o možnost importu statických webových stránek. Dále jsme se seznámili s tímto redakčním systémem na programové úrovni a naučili se využívat jeho API pro další rozvoj tohoto systému.

5 Literatura

[1] W3C Recommendation [online]. 24.12.1999 [cit. 2010-05-04]. HTML 4.01 Specification. Dostupné z WWW: <http://www.w3.org/TR/html401/>.

[2] BRADLEY, Neil. XML kompletní průvodce. [s.l.] : [s.n.], 1998. Co je XML?, s. 18-20. ISBN 80-7169-949-7.

[3] ECMA : C# Language Specification. [s.l.] : [s.n.], 2006. 531 s. Dostupné z WWW: <http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-334.pdf>

[4] Mono [online]. [cit. 2010-05-04]. URL: < http://www.mono-project.com/Main_Page>.

[5] Interval.cz [online]. 08. 03. 2002 [cit. 2010-05-04]. Architektura .NET frameworku. Dostupné z WWW: <http://interval.cz/clanky/architektura-net-frameworku/>.

[6] Msdn [online]. 2003 [cit. 2010-05-04]. Understanding SOAP. Dostupné z WWW: <http://msdn.microsoft.com/en-us/library/ms995800.aspx>.

[7] Msdn [online]. 2003 [cit. 2010-05-04]. Understanding WSDL. Dostupné z WWW: <http://msdn.microsoft.com/en-us/library/ms996486.aspx>.

[8] Regulární výrazy [online]. [cit. 2010-05-04]. Úvod. Dostupné z WWW: <http://www.regularnivyrazy.info/>.

[9] TiddlyWiki [online]. c2009 [cit. 2010-05-04]. Dostupné z WWW: <http://www.tiddlywiki.com/>.

[10] Wikipedia [online]. 2010 [cit. 2010-05-04]. Systém pro správu obsahu. URL: <http://cs.wikipedia.org/wiki/Syst%C3%A9m_pro_spr%C3%A1vu_obsahu>.

[11] PHP [online]. [cit. 2010-05-04]. PHP Licensing. Dostupné z WWW: <http://www.php.net/license/>.

[12] Apache friends [online]. [cit. 2010-05-04]. XAMPP. Dostupné z WWW: <http://www.apachefriends.org/en/xampp.html>.

[13] CMS Matrix [online]. [cit. 2010-05-04]. CMS Matrix. Dostupné z WWW: <http://cmsmatrix.org/matrix/cms-matrix/>.

[14] Kentico CMS [online]. [cit. 2010-05-04]. Complete ASP.NET CMS for Your Web Site. Dostupné z WWW: <http://www.kentico.com/free-cms-asp-net.aspx>.

[15] Kentico CMS Devnet [online]. [cit. 2010-05-04]. Quality of Kentico CMS source code. URL: <http://devnet.kentico.com/Blogs/Martin-Hejtmanek/October-2008/Quality-of-Kentico-CMS-source-code.aspx>.

[16] Msdn [online]. 2008 [cit. 2010-05-04]. BackgroundWorker Class. URL: <http://msdn.microsoft.com/en-us/library/system.componentmodel.backgroundworker.aspx>.

[17] CodePlex [online]. 2009 [cit. 2010-05-04]. Html Agility Pack . Dostupné z WWW: <htmlagilitypack.codeplex.com/>.

[18] SECURE HASH STANDARD. [s.l.] : [s.n.], 17.4.1995. 21 s. Dostupné z WWW: <http://www.digistamp.com/reference/fip180-1.pdf>.

[19] ROBINSON, Simon, et al. C# programujeme profesionálně. [s.l.] : [s.n.], 2003. Co je to vlastně .NET?, s. 2-3. ISBN 80-251-0085-5.

[20] Kentico CMS 5.0 Developer's Guide [online]. 2010 [cit. 2010-05-10]. Where is the content stored?. Dostupné z WWW: <http://devnet.kentico.com/docs/devguide/index.html?where_is_the_content_stored.htm>.

[21] Kentico Devnet [online]. 2010 [cit. 2010-05-10]. Kentico API. Dostupné z WWW: <http://devnet.kentico.com/downloads/kenticocms_api.zip>.

[22] HEJTMÁNEK, Martin. Kentico CMS Architecture. [s.l.] : [s.n.], 17.6.2008. Architecture - overview, s. 1.

[23] HEJTMÁNEK, Martin. Kentico CMS Architecture. [s.l.] : [s.n.], 17.6.2008. Architecture - Database, s. 1.

6 Seznam příloh

Příloha 1. Manuál
Příloha 2. Návod na připojení zdrojového kódu modulu k projektu Kentico CMS
Příloha 3. Grafické znázornění algoritmu, který parsuje HTML stránku a získává obsah a vzhled
Příloha 4. Grafické znázornění architektury Windows aplikace
Příloha 5. Grafické znázornění architektury modulu v Kentico CMS
Příloha 6. Zdrojový kód webové služby
Příloha 7. CD

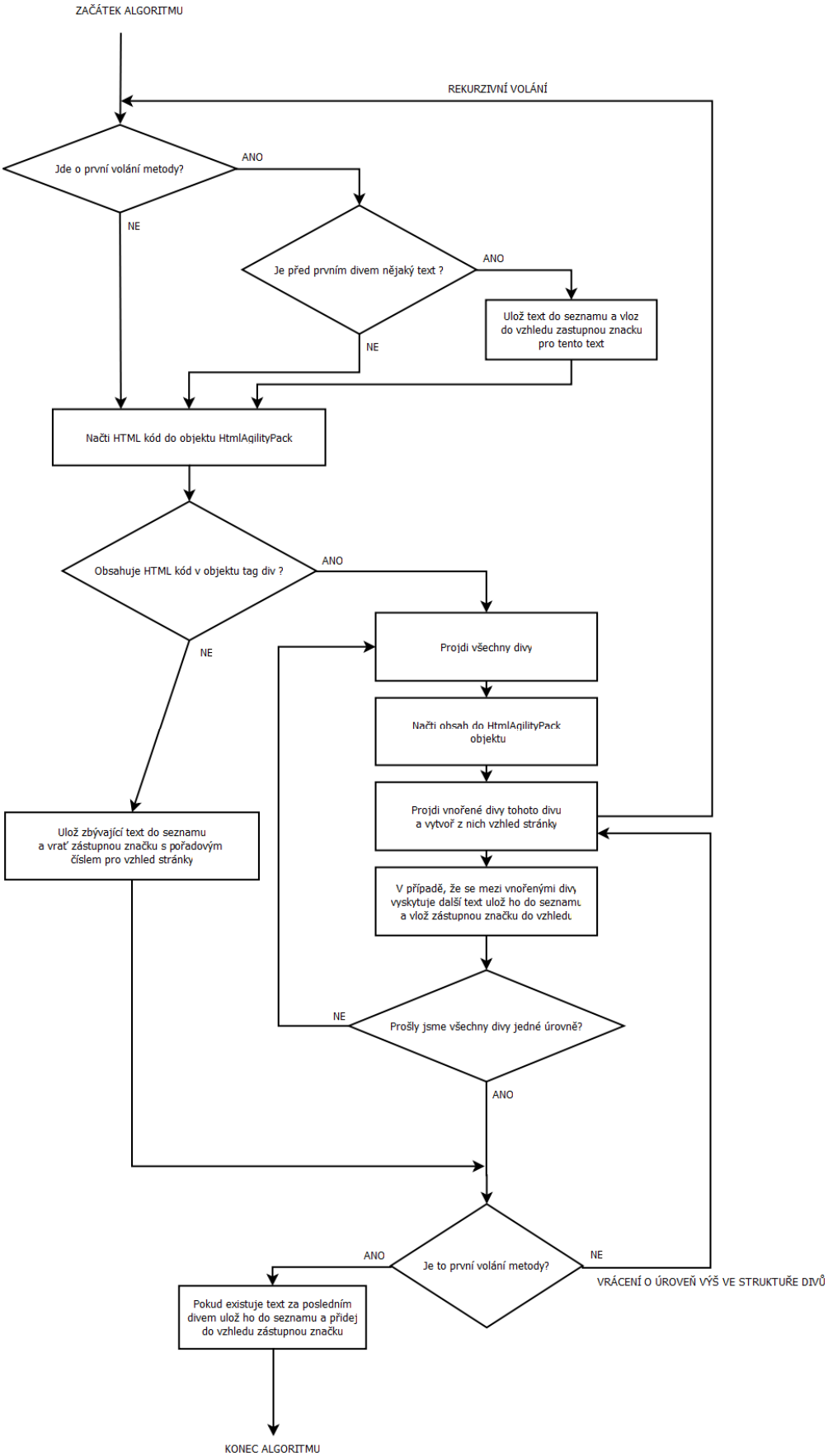
6.1 Příloha 1

1. Stáhněte si program Kentico CMS 5.0 nebo vyšší z adresy http://www.kentico.com/download.aspx . Na přiloženém CD je verze 5.0.
2. Ověřte, zda máte nainstalován všechen potřebný software. Seznam potřebného software je umístěn zde http://www.kentico.com/download/system-requirements.aspx
3. Nainstalujte Kentico CMS. Pokyny k instalaci jsou uvedeny v instalačním manuálu (http://devnet.kentico.com/docs/devguide/index.html?installation_overview.htm)
4. Není potřeba instalovat žádný z ukázkových webů.
5. Data ze složky „KenticoCMS“ umístěné na CD zkopírujte do nainstalovaného projektu (např. C:\inetpub\wwwroot\KenticoCMS)
6. Složku „Aplikace“ z CD zkopírujte na disk a spusťte „HTMLConverter.exe“ z této složky
7. Projděte průvodce. Jako vstupní ukázkový archív můžete použít prezentace.zip umístěný ve složce „Testovaci_weby“ na CD. Přihlašovací jméno je „administrator“ bez hesla.
8. Po projití všech kroků průvodce ve Windows aplikaci je nově vytvořená webová prezentace připravena k použití

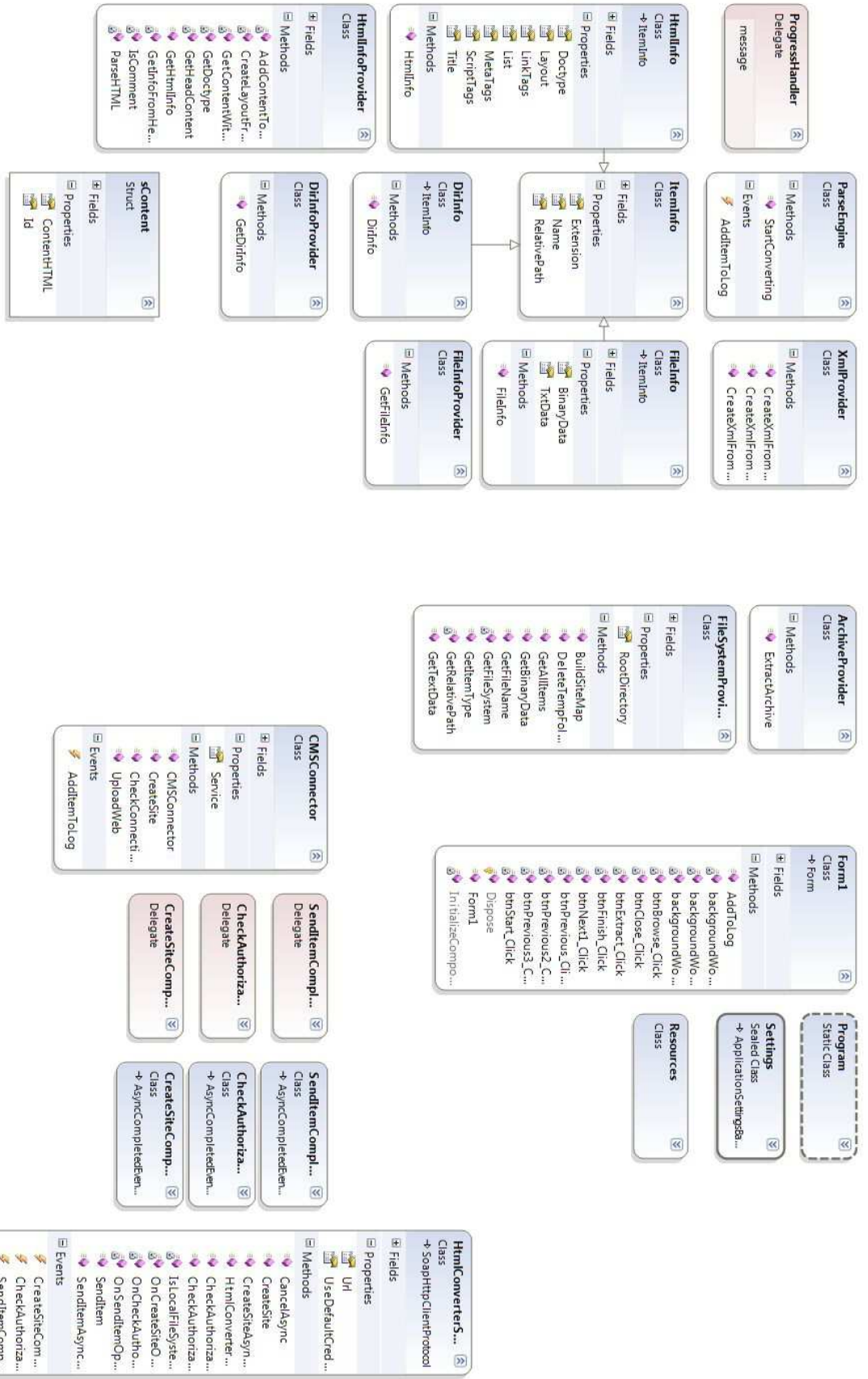
6.2 Příloha 2

1. Nainstalujte program Kentico CMS podle návodu z přílohy 1
2. Zkopírujte složku HTMLConverter do projektu Kentico CMS, tato složka je umístěna na CD - \source\Modul\
3. Otevřete nainstalovaný projekt v programu Microsoft Visual Studio 2008 (~/WebProject.sln).
4. Zvolte možnost přidat k projektu existující projekt a přidejte HTMLConverter/HtmlConverter.csproj
5. Nahraďte všechny reference na knihovny začínající prefixem CMS v adresáři References projektu HTMLConverter za aktuální, které jsou umístěny v adresáři Bin:
   "CMS.TreeEngine.dll"
   "CMS.DataEngine.dll"
   "CMS.EventLog.dll"
   "CMS.FileManager.dll"
   "CMS.FormEngine.dll"
   "CMS.GlobalHelper.dll"
   "CMS.IDataConnectionLibrary.dll"
   "CMS.ImportExport.dll"
   "CMS.PortalEngine.dll"
   "CMS.SettingsProvider.dll"
   "CMS.SiteProvider.dll"
   "CMS.Staging.dll"
6. V projektu Kentico CMS přidejte referenci na projekt HTMLConverter
7. Zkopírujte složky App_Code, App_Data a CMSPages z CD (/KenticoCMS) do složky s Kentico CMS
8. Zkompilujte celý solution

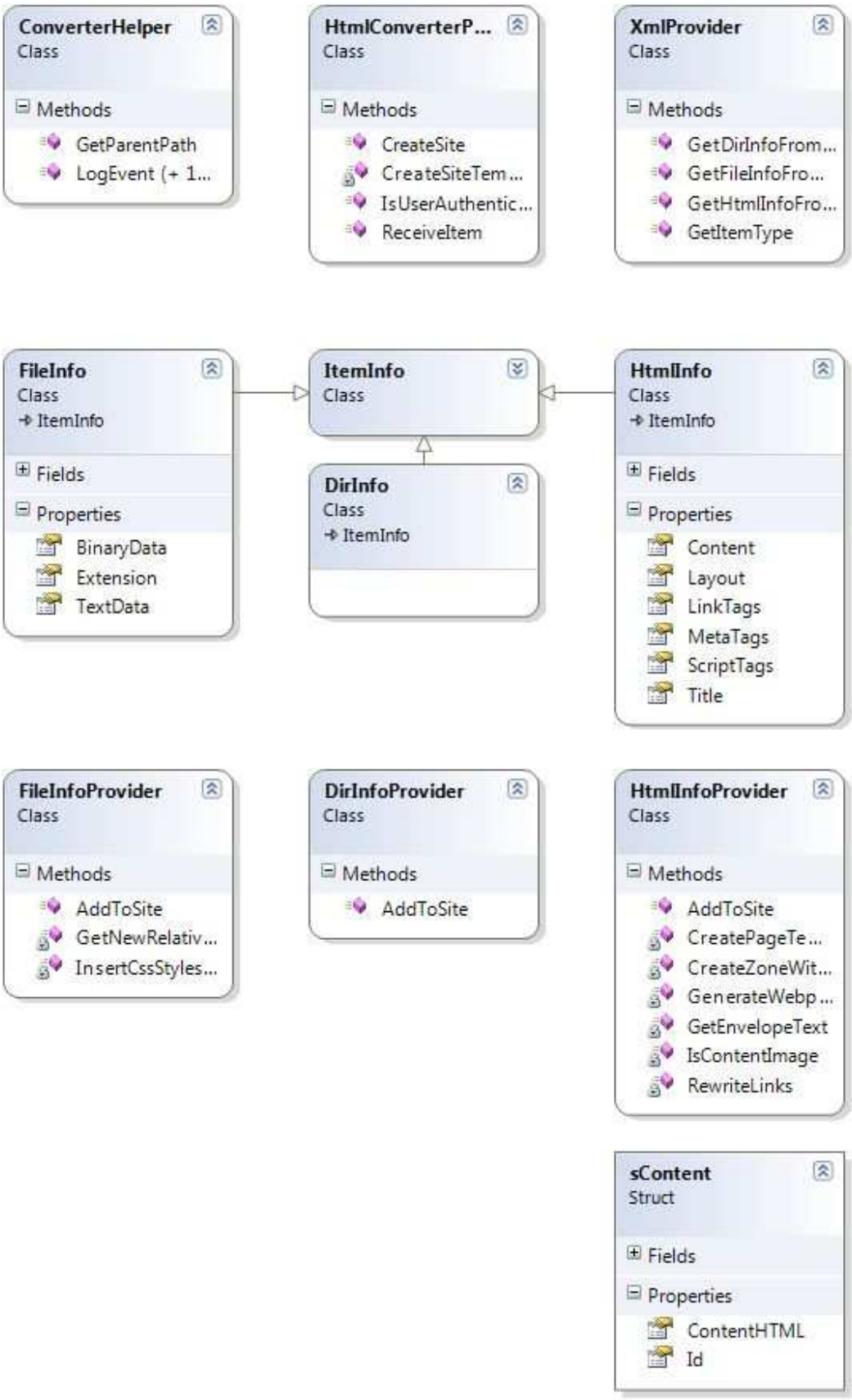
6.3 Příloha 3


ZAČÁTEK ALGORITMU
REKURZIVNÍ VOLÁNÍ
Jde o první volání metody?
ANO
NE
Je před prvním divem nějaký text ?
ANO
Ulož text do seznamu a vloz do vzhledu zastupnou znacku pro tento text
NE
Načti HTML kód do objektu HtmlAgilityPack
Obsahuje HTML kód v objektu tag div ?
ANO
NE
Projdi všechny divy
Načti obsah do HtmlAgilityPack objektu
Projdi vnořené divy tohoto divu a vytvoř z nich vzhled stránky
V případě, že se mezi vnořenými divy vyskytuje další text ulož ho do seznamu a vlož zástupnou značku do vzhledu
Prošly jsme všechny divy jedné úrovně?
NE
ANO
Ulož zbývající text do seznamu a vrať zástupnou značku s pořadovým číslem pro vzhled stránky
Je to první volání metody?
ANO
NE
VRÁCENÍ O ÚROVEŇ VÝŠ VE STRUKTUŘE DIVŮ
Pokud existuje text za posledním divem ulož ho do seznamu a přidej do vzhledu zástupnou značku
KONEC ALGORITMU

6.4 Příloha 4


ProgressHandler
Delegate
message
HtmlInfo
Class
→ ItemInfo
Fields
Properties
Doctype
Layout
LinkTags
List
MetaTags
ScriptTags
Title
Methods
HtmlInfo
HtmlInfoProvider
Class
Fields
Methods
AddContentTo...
CreateLayoutFr...
GetContentWit...
GetDoctype
GetHeadContent
GetHtmlInfo
GetInfoFromHe...
IsComment
ParseHTML
ParseEngine
Class
Methods
StartConverting
Events
AddItemToLog
ItemInfo
Class
Fields
Properties
Extension
Name
RelativePath
DirInfo
Class
→ ItemInfo
Methods
DirInfo
DirInfoProvider
Class
Methods
GetDirInfo
sContent
Struct
Fields
Properties
ContentHTML
Id
XmlProvider
Class
Methods
CreateXmlFrom...
CreateXmlFrom...
CreateXmlFrom...
FileInfo
Class
→ ItemInfo
Fields
Properties
BinaryData
TxtData
Methods
FileInfo
FileInfoProvider
Class
Methods
GetFileInfo
ArchiveProvider
Class
Methods
ExtractArchive
FileSystemProvi...
Class
Fields
Properties
RootDirectory
Methods
BuildSiteMap
DeleteTempFol...
GetAllItems
GetBinaryData
GetFileName
GetFileSystem
GetItemType
GetRelativePath
GetTextData
Form1
Class
→ Form
Fields
Methods
AddToLog
backgroundWo...
backgroundWo...
backgroundWo...
btnBrowse_Click
btnClose_Click
btnExtract_Click
btnFinish_Click
btnNext1_Click
btnPrevious_Cli...
btnPrevious2_C...
btnPrevious3_C...
btnStart_Click
Dispose
Form1
InitializeCompo...
Program
Static Class
Settings
Sealed Class
→ ApplicationSettingsBa...
Resources
Class
CMSConnector
Class
Fields
Properties
Service
Methods
CMSConnector
CreateSite
CheckConnecti...
UploadWeb
Events
AddItemToLog
SendItemCompl...
Delegate
SendItemCompl...
Class
→ AsyncCompletedEven...
CheckAuthoriza...
Delegate
CheckAuthoriza...
Class
→ AsyncCompletedEven...
CreateSiteComp...
Delegate
CreateSiteComp...
Class
→ AsyncCompletedEven...
HtmlConverterS...
Class
→ SoapHttpClientProtocol
Fields
Properties
Url
UseDefaultCred...
Methods
CancelAsync
CreateSite
CreateSiteAsyn...
HtmlConverter...
CheckAuthoriza...
CheckAuthoriza...
IsLocalFileSyste...
OnCreateSiteO...
OnCheckAutho...
OnSendItemOp...
SendItem
SendItemAsync...
Events
CreateSiteCom...
CheckAuthoriza...
SendItemComp...

6.5 **Příloha 5**


ConverterHelper
Class
Methods
GetParentPath
LogEvent (+ 1...
HtmlConverterP...
Class
Methods
CreateSite
CreateSiteTem...
IsUserAuthentic...
ReceiveItem
XmlProvider
Class
Methods
GetDirInfoFrom...
GetFileInfoFro...
GetHtmlInfoFro...
GetItemType
FileInfo
Class
ItemInfo
Fields
Properties
BinaryData
Extension
TextData
ItemInfo
Class
DirInfo
Class
ItemInfo
HtmlInfo
Class
ItemInfo
Fields
Properties
Content
Layout
LinkTags
MetaTags
ScriptTags
Title
FileInfoProvider
Class
Methods
AddToSite
GetNewRelativ...
InsertCssStyles...
DirInfoProvider
Class
Methods
AddToSite
HtmlInfoProvider
Class
Methods
AddToSite
CreatePageTe...
CreateZoneWit...
GenerateWebp...
GetEnvelopeText
IsContentImage
RewriteLinks
sContent
Struct
Fields
Properties
ContentHTML
Id

6.6 Příloha 6

```
using System.Web.Services;
using System.Xml;
using HtmlConverter;

/// <summary>
/// Summary description for WebConverterService
/// </summary>
[WebService(Namespace = "http://localhost/webservice1/")]
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
public class HtmlConverterService : System.Web.Services.WebService {

    public HtmlConverterService()
    {
    }

    /// <summary>
    /// zjisti zda je uzivatel opravnen k vytvoreni noveho webu
    /// </summary>
    /// <param name="login">prihlasovaci jmeno</param>
    /// <param name="password">prihlasovaci heslo</param>
    /// <returns>true pokud je opravnen, false pokud neni</returns>
    [WebMethod]
    public bool CheckAuthorization(string login,string password)
    {

        HtmlConverterProvider webc = new HtmlConverterProvider();
        return webc.IsUserAuthenticated(login, password);

    }

    /// <summary>
    /// vytvori v Kentico CMS prazdnou kostru webove prezentace
    /// </summary>
    /// <param name="sitename">nazev noveho webu</param>
    /// <param name="domain">domena noveho webu</param>
    /// <param name="login">prihlasovaci jmeno uzivatele</param>
    /// <returns>true pokud je webova kostra vytvorena v poradku, false
       pokud neni</returns>
    [WebMethod]
    public bool CreateSite(string sitename,string domain, string login)
    {

        HtmlConverterProvider webc = new HtmlConverterProvider();
        return webc.CreateSite(sitename, domain,login);

    }

    /// <summary>
    /// prijme soubor a posle ho na zpracovani do konverzniho modulu
    /// </summary>
    /// <param name="doc">XmlDocument se souborem</param>
    /// <param name="sitename">nazev webu</param>
    /// <param name="login">prihlasovaci jmeno uzivatele</param>
    /// <returns>true pokud se ulozil v poradku, false pokud nastala
       chyba</returns>
    [WebMethod]
    public bool SendItem(XmlDocument doc,string sitename,string login)
    {

        HtmlConverterProvider webc = new HtmlConverterProvider();
        return webc.ReceiveItem(doc,sitename,login);

    }
}
```