VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

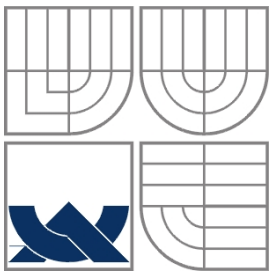
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

PUBLIKAČNÍ SYSTÉMY STRUKTUROVANÝCH DOKUMENTŮ

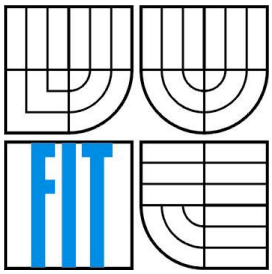
BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE Lukáš Máčel
AUTHOR

BRNO 2007



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

PUBLIKAČNÍ SYSTÉMY STRUKTUROVANÝCH DOKUMENTŮ

THE PUPLISHING SYSTEMS OF STRUCTURED DOCUMENTS

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR

Lukáš Máčel

VEDOUCÍ PRÁCE
SUPERVISOR

prof. Ing. Tomáš Hruška, CSc.

BRNO 2007

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2006/2007

Zadání bakalářské práce

Řešitel: **Máčel Lukáš**
Obor: Informační technologie
Téma: **Publikační systémy strukturovaných dokumentů**
Kategorie: Databáze

Pokyny:

1. Seznamte se s programovacím jazykem Visual Basic a jeho použitím pro implementaci serverové části webové aplikace, prostudujte DOM model webové stránky a práci s ním prostřednictvím JavaScriptu a technologie AJAX.
2. Navrhněte knihovnu pro webové uživatelské rozhraní s využitím technologie AJAX a systém správy strukturovaných dokumentů zahrnující generování jejich obsahu včetně vzájemných vztahů pro prezentaci na webu.
3. Realizujte prototyp publikačního systému pro správu dokumentů systému řízení kvality ISO 9000.
4. Zhodnoťte přínos navržené architektury publikačního systému ve srovnání se standardními CMS systémy.

Literatura:

- Document Object Model (DOM): http://www.w3.org/DOM/
- Visual Basic Reference: http://msdn.microsoft.com/vbasic/reference/
- Mozilla Developer Center - Javascript documentation: http://developer.mozilla.org/en/docs/JavaScript

Při obhajobě semestrální části projektu je požadováno:

- Body 1. a 2.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním paměťovém médiu (disketa, CD-ROM), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Hruška Tomáš, prof. Ing., CSc.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2006
Datum odevzdání: 15. května 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

doc. Ing. Jaroslav Zendulka, CSc.
vedoucí ústavu

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami

1. Pan

Jméno a příjmení: **Lukáš Máčel**
Id studenta: 84233
Bytem: Chudobova 2493/59, 615 00 Brno
Narozen: 25. 07. 1984, Brno
(dále jen "autor")

a

2. Vysoké učení technické v Brně

Fakulta informačních technologií
se sídlem Božetěchova 2/1, 612 66 Brno, IČO 00216305
jejímž jménem jedná na základě písemného pověření děkanem fakulty:

.....
(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
bakalářská práce

Název VŠKP: Publikační systémy strukturovaných dokumentů
Vedoucí/školitel VŠKP: Hruška Tomáš, prof. Ing., CSc.
Ústav: Ústav informačních systémů
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:
tištěné formě počet exemplářů: 1
elektronické formě počet exemplářů: 2 (1 ve skladu dokumentů, 1 na CD)

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 pořizování výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti:
   - ☒ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

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

Článek 3
Závěrečná ustanovení

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

V Brně dne: .....

.....	.....
Nabyvatel	Autor

Abstrakt

Tato bakalářská práce se zabývá problematikou pořizování a zveřejňování strukturovaných dokumentů v elektronické podobě. Publikační systém je realizován jako webový server. Dokumenty jsou pořizovány externím textovým procesorem MS Word. Pro specifikaci struktury a grafického vzhledu je využita šablona dokumentu. Publikační systém nabízí webové rozhraní pro aktualizace publikovaných dokumentů založené na přístupu AJAX. Prototyp publikačního systému slouží pro vedení firemní dokumentace v oblasti managementu jakosti a splňuje požadavky dané standardem ISO 9000.

Klíčová slova

AJAX, ISO 9000, publikační systém, strukturovaný dokument

Abstract

The publishing of electronic structured documents is the focus of this bachelor's thesis. The publishing system is implemented as a web server. Documents are written by the external MS Word text processor. The document layout and structure are specified by a document template. The publishing system has a web interface based on AJAX to update published documents. The publishing system's prototype is designed to manage company documentation in the field of quality management and fulfils ISO 9000 standards.

Keywords

AJAX, ISO 9000, publishing system, structured document

Citace

Lukáš Máčel: Publikační systémy strukturovaných dokumentů, bakalářská práce, Brno, FIT VUT v Brně, 2007

Publikační systémy strukturovaných dokumentů

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením
prof. Ing. Tomáše Hrušky, CSc.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Lukáš Máčel
9.5.2007

Poděkování

Na tomto místě bych chtěl poděkovat vedoucímu mojí bakalářské práce
prof. Ing. Tomáši Hruškovi, CSc. za zájem, připomínky a čas, který věnoval mé práci.

Obsah

Úvod

S publikačními systémy se setkáváme velmi často. Jako příklad uvedu jednoduché diskusní fórum, prostřednictvím kterého můžeme prezentovat svoje názory a myšlenky ostatním. Pomocí internetového prohlížeče se připojíme k serveru, který nám zobrazí jednotlivé příspěvky uživatelů. K dispozici je také rozhraní pro přidání vlastního příspěvku, který je ve formě krátké textové zprávy odeslán na server. Ten zprávu přijme, opatří ji časovým razítkem a zobrazí ostatním čtenářům. Publikací tedy rozumíme proces uveřejnění příspěvku. Aplikace běžící na serveru, která odpovídá na dotazy připojených uživatelů – klientů, představuje jednoduchý publikační systém.

Fóra umožňují publikovat pouze krátké reakce na diskutovanou problematiku. Ve své práci se budu zabývat publikačními systémy, které zveřejňují rozsáhlejší dokumenty v elektronické podobě. Na rozdíl od příspěvku do fóra má publikovaný dokument složitější strukturu. Obsah je členěn pomocí odstavců a jednotlivých úrovní nadpisů na menší jednotky. Mohou se zde vyskytovat tabulky nebo grafy. Vytvořit takový dokument vyžaduje použití sofistikovanějších nástrojů. Při návrhu publikačního systému se proto zaměřím nejen na to, jak realizovat samotné zveřejnění dokumentu, ale také na pořizování dokumentů samotných, aby proces tvorby plynule navazoval na publikaci.

Cílem mojí práce je tedy vytvořit publikační systém, který bude pracovat se strukturovanými dokumenty pořízenými elektronickou cestou. Pojmem strukturovaný dokument ovšem nemyslím pouze jeho grafickou reprezentaci, ale také to, že v dokumentu je možné rozlišit části a těm přidělit jistou syntax a sémantiku. Dokument může například obsahovat hlavičku, kde je uvedeno jméno autora, datum pořízení, typ dokumentu, atd. Hlavička představuje strukturovanou část dokumentu, kterou musí publikační systém analyzovat a získat z ní atributy dokumentu důležité pro další zpracování.

Ve své práci jsem se rozhodl vytvořit prototyp publikačního systému, který bude určen pro vedení firemní dokumentace v oblasti řízení jakosti. Jasné požadavky na strukturu dokumentu jsou dány standardem ISO 9000. Hlavním úkolem publikačního systému bude umožnit managementu vytvářet jednotlivé směrnice, metodiku a pravidla a publikovat je zaměstnancům firmy.

Jako důležitý cíl si kladu umožnit uživateli pořizovat dokumenty v textovém procesoru, na který je zvyklý. Publikační systémy často nabízí vlastní nástroje pro tvorbu dokumentů. Ty jsou však pro vytváření složitých strukturovaných dokumentů nevhodné. Uživatel navíc s těmito nástroji neumí pracovat.

V kapitole 1 se budu blíže zabývat publikačními systémy. Zaměřím se na jejich architekturu a typické služby. Porovnám také současné publikační systémy podle formy publikace a na základě tohoto srovnání vyberu způsob pořizování dokumentu, který je pro uživatele nejpřirozenější.

Kapitolu 2 věnuji normě ISO 9000. Shrnu požadavky, které klade na strukturu publikovaných dokumentů. Uvedu také služby, které musí publikační systém nabízet, aby byly splněny požadavky normy na řízení dokumentace.

Kapitola 3 obsahuje návrh publikačního systému. Na začátku popíšu základní koncepční rozhodnutí – volbu technologie, textového procesoru a způsob realizace databáze. Potom uvedu služby, které bude publikační systém nabízet. Vytvořím také konceptuální model publikačního systému určeného pro nasazení ve firmě. V závěru kapitoly se zaměřím na syntaktický popis strukturované části dokumentu a využití šablon.

V kapitole 4 popíšu implementaci serverové části publikačního systému. Nejdříve představím koncept univerzální hodnoty. Dále se budu zabývat manipulací se vstupními dokumenty, jejich kontrolou a převodem na internetové stránky. Uvedu, jakým způsobem bude realizována databáze dokumentů a také, jak se dynamicky vytvoří jednotlivé stránky publikačního systému.

Kapitola 5 obrátí pozornost na klientskou část systému. Zaměřím se na tvorbu uživatelsky příjemného rozhraní s využitím AJAXu. Popíši také knihovnu pro uživatelské rozhraní, kterou jsem pro tyto potřeby vytvořil.

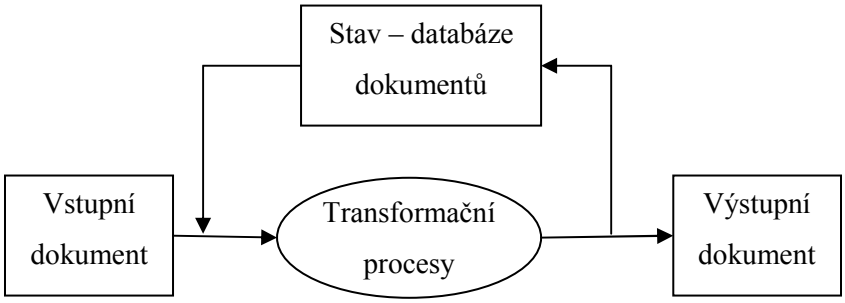
Kapitola 6 shrnuje přínos této práce. Uvedu také další možnosti rozšíření publikačního systému.

1 Publikační systémy

V současné době přibývá stále více dokumentů v elektronické podobě. Dochází tak přirozeně i ke změně způsobu, jakým jsou dokumenty zveřejňovány - publikovány. Roli redaktora, který přijímá články od různých autorů a sestavuje z nich výsledný obsah novin, může vykonávat i program běžící na počítači.

1.1 Definice

Aplikace, která přijímá elektronické dokumenty, kontroluje jejich obsah a nějakou formou je prezentuje skupině uživatelů (čtenářů), představuje publikační systém. Následující obrázek zobrazuje základní prvky publikačního systému a jejich vztahy.



Obr. 1-1 Schéma publikačního systému

Vstupem systému je elektronický dokument, který je transformován na výstupní dokument v požadovaném formátu. V systému je obsažena **zpětná vazba**. Výstupy transformačních procesů systému nejsou závislé pouze na vstupu, ale také na jeho aktuálním stavu, který je charakterizován dříve publikovanými dokumenty. Ty jsou uchovávány prostřednictvím databáze.

1.2 Architektura

Publikační systém je otevřeným systémem. Jeho úkolem je obsloužit větší skupinu uživatelů a umožnit zveřejňovat dokumenty i uživatelům, kteří pracují na vzdáleném počítači. Z tohoto důvodu je nejvhodnější architekturou publikačního systému **architektura klient-server**.

Klientská část systému běží na počítači uživatele. Zajišťuje připojení k serveru a buď nabízí přímo rozhraní pro tvorbu publikovaného dokumentu, nebo již vytvořený dokument odešle na server.

Serverová část systému zpracovává dotazy klientů. Existují dvě možnosti. Klient posílá nový dokument, který má být publikován, nebo žádá o zobrazení některého z publikovaných dokumentů. Na serveru se také nachází databáze, kde jsou uloženy všechny publikované dokumenty.

1.3 Služby

Publikační systémy nabízejí mimo zveřejnění dokumentu také další služby. Nejčastěji se můžeme setkat s následujícími službami:

- řízení přístupu pro publikaci a čtení dokumentů,
- třídění dokumentů do kategorií podle jejich obsahu,
- fulltextové vyhledávání.

1.4 Klasifikace podle formy publikace

U strukturovaného dokumentu můžeme rozlišit:

1. strukturovanou část,
2. vlastní text dokumentu.

Strukturovaná část obsahuje formálně definované atributy dokumentu. Příkladem může být hlavička dokumentu, kde jsou uvedeny údaje o autorovi, datu pořízení, platnosti dokumentu, aj. Za strukturovanou informací potom následuje vlastní text dokumentu.

Pro obě části je vhodný jiný způsob pořizování obsahu. Publikační systém musí analyzovat strukturovanou informaci dokumentu a provést kontroly, jestli jednotlivé atributy dokumentu splňují formální definici. Proto je výhodné, aby strukturovaná část dokumentu byla pořizovaná prostřednictvím **aplikačního formuláře**, který uživatele intuitivně vede k uvedení správných hodnot atributů. Jednotlivé atributy mohou být navíc okamžitě kontrolovány a v případě chyby je uživatel „přinucen“ hodnotu opravit.

Pořizování textové části naopak vyžaduje použití **textového procesoru**, který nabízí prostředky pro kvalitní tvorbu dokumentů.

1.4.1 Způsoby pořizování

Zaměřím se nyní podrobněji na přístupy, které volí současné publikační systémy pro pořizování zmíněných částí strukturovaného dokumentu.

Velmi často je volena strategie **pořizování dokumentu výhradně pomocí rozhraní**, které nabízí publikační systém. Tento přístup má výhodu v tom, že veškerá práce uživatele na dokumentu může být kontrolována a je možné reagovat na chybný vstup.

Pro pořizování strukturované části dokumentu je tato strategie vhodná. Rozhraní obvykle obsahuje formulář s příslušnými kontrolami polí. Problém nastává při pořizování samotného textu

dokumentu. Publikační systém musí nabídnout kvalitní textový procesor. Praxe ukazuje, že výsledkem takové snahy jsou spíše jednoduché textové editory. Slovo „editory“ uvádím záměrně, protože kvalita pořizování textu zde není na požadované úrovni. Jako důležitý cíl jsem si stanovil nenutit uživatele používat nástroje, se kterými neumí pracovat. Tuto strategii proto při návrhu publikačního systému nevyužiji.

Řešení uvedeného problému je **oddělit pořizování obsahu strukturované části dokumentu od psaní samotného textu.** Uživatel tedy napíše nestrukturovanou část dokumentu v textovém procesoru, na který je zvyklý. Potom specifikuje atributy dokumentu pomocí formuláře nabízeného publikačním systémem a pořízený dokument pošle na server jako přílohu.

Přístup splňuje požadavky na pořizování obou částí dokumentu. Jako nevýhodu však vidím porušení integrity procesu vytváření dokumentu. Pro uživatele je přirozenější a také příjemnější, když bude moci v textovém procesoru pořizovat celý dokument, včetně jeho strukturované části.

Poslední strategie tedy spočívá v **pořizování obou částí dokumentu v externím textovém procesoru**. Jedná se o přístup atypický, který u současných publikačních systémů není příliš využíván. Podle mého názoru ale nejlépe vyhovuje potřebám uživatele. Vytváření dokumentu probíhá pouze prostřednictvím textového procesoru. Proces pořizování dokumentu není rozdělen.

V úvodu kapitoly jsem uvedl, že pro kontrolu strukturované části je vhodný formulář, protože je možné ihned odhalit chybný vstup. V této variantě ovšem kontroly průběžně provádět nelze, protože pořizování probíhá mimo publikační systém. Jsou tedy kladeny vyšší nároky na validaci strukturované části na serveru.

Formulář také poskytoval názorný popis struktury. Publikační systém musí najít podobný prostředek, jak uživateli sdělit požadovanou strukturu. Cílem je uživatele přirozeně vést k správnému pořízení atributů dokumentu. Jinak hrozí nebezpečí, že strukturovaná část dokumentu bude obsahovat příliš mnoho chyb a proces pořizování bude potom zdlouhavý.

I přes zmíněná úskalí považuji poslední strategii za nejlepší. Pro publikaci nemusí mít uživatel žádné další znalosti. Postačuje schopnost pracovat s textovým procesorem. Využiji proto tento přístup při návrhu prototypu publikačního systému.

2 ISO 9000

ISO - International Organization for Standardization byla založena v roce 1947. Jedná se o mezinárodní organizaci zabývající se vývojem a sjednocováním standardů. Organizaci tvoří síť národních institutů s centrem v Ženevě. V České republice je zástupcem Český normalizační institut.

Hlavní činností organizace je vývoj technických norem. Mezi nejznámější a mezinárodně uznávané patří soustava norem ISO 9000. Hlavním cílem těchto norem je poskytnout návod na vybudování **systému řízení kvality**. ISO 9000 není určeno pro specifický druh produktu, lze ho uplatnit ve všech oblastech výroby a služeb.

2.1 Systém managementu jakosti

Systém řízení kvality je postaven na přesné identifikaci procesů probíhajících ve firmě a zjištění jejich vzájemných vazeb. Jednotlivé procesy je třeba monitorovat, provádět jejich měření a konat patřičné kroky pro jejich zlepšení. Centrem pozornosti systému je zákazník, jehož potřeby a požadavky musí být v organizaci vnímány a plněny.

Kromě řízení samotných procesů vyžaduje ISO 9000 také vedení patřičné dokumentace a její řízení. Právě pro tuto potřebu bude sloužit prototyp publikačního systému, který chci vytvořit. Opustím nyní obecné principy managementu kvality a zaměřím se na konkrétní požadavky normy týkající se struktury dokumentů, které budou prostřednictvím publikačního systému zveřejňovány zaměstnancům firmy.

2.2 Vedení a řízení dokumentace

2.2.1 Struktura dokumentu

Dokumentace je tvořena soustavou **řízených** dokumentů, které popisují všechny základní činnosti firmy. Obsah řízeného dokumentu je závazný pro vymezenou skupinu zaměstnanců. Každý řízený dokument má definovanou strukturu a formální úpravu.

Řízený dokument je opatřen hlavičkou, kde jsou uvedeny základní vlastnosti dokumentu. Povinně se zde musí objevit:

- číslo a jméno dokumentu,
- autor dokumentu a datum vytvoření,
- schvalovatel a datum, kdy dokument vstupuje v účinnost,
- seznam pracovních funkcí, pro které je dokument určen,
- typ dokumentu (směrnice, pravidlo, metodika),

- stručný popis obsahu dokumentu,
- seznam klíčových slov.

2.2.2 Životní cyklus dokumentu

Směrnice, pravidla a metodika jsou dokumenty řízené. Existují přesně definované aktivity, které lze s dokumenty provádět. Každý řízený dokument pochází životním cyklem.

První životní fází je **vytvoření** dokumentu. Potom je možné dokument **připomínkovat**. V platnost vstupuje až po jeho **schválení**. Dokument je následně **distribuován** mezi zaměstnance a dochází k procesu **seznamování**. Jednou z možných akcí je **modifikace** dokumentu. Po skončení platnosti dokumentu dochází k jeho **archivaci.**

Z životního cyklu dokumentu vyplývá, že publikační systém musí zajistit seznámení zaměstnanců s nově zveřejněným dokumentem. Dále je nutné, aby bylo možné modifikovat již publikované dokumenty. Tyto požadavky promítnu do návrhu publikačního systému.

3 Návrh publikačního systému

3.1 Základní koncepční rozhodnutí

3.1.1 Použitá technologie

Prvním důležitým krokem při návrhu je výběr technologie, kterou použiji při realizaci publikačního systému. Nabízí se dvě možnosti. Buď vytvořit desktopovou aplikaci, anebo implementovat publikační systém jako webovou aplikaci.

Desktopové aplikace v době psaní této práce stále převažují nad webovými. Důvodem je především to, že prostředky pro tvorbu webových aplikací byly donedávna na nízké úrovni a neumožňovaly vytvářet interaktivní rozhraní, bez kterého se u netriviální aplikace neobejdeme. Situace se významně změnila s prosazením nového přístupu označovaného jako AJAX. Díky němu lze posílat na server asynchronní žádost a rozhraní tak může rychle reagovat na akci uživatele podobně, jako je tomu u desktopové aplikace. Jaké jsou tedy výhody a nevýhody každé varianty?

Desktopová aplikace může obsahovat řadu pokročilých komponent jako je stromové menu, datová mřížka nebo komplexní formulář. Všechny tyto prvky jsou schopny reagovat na podněty uživatele, měnit svůj stav a kooperovat s dalšími komponentami. Vývoj takového rozhraní je výrazně rychlejší než u webové aplikace. Pomocí automatických návrhářů, které nabízí současná integrovaná prostředí, lze velmi jednoduše vytvořit výsledné rozhraní.

Nevýhodou desktopové aplikace je naopak nutnost instalovat na počítač speciální software. Problém spočívá také v aktualizacích aplikace, která klade nároky jak na provozovatele, tak i na samotného uživatele.

U webové aplikace zmíněný problém s instalací a údržbou nenastává, protože pro provoz postačuje, aby měl uživatel nainstalován internetový prohlížeč. Práce s aplikací formou internetových stránek je navíc uživatelsky příjemná.

Nepříjemností je naopak nekompatibilita prohlížečů, která výrazně zpomaluje vývoj aplikace. Rovněž tvorba rozhraní je náročnější. Dialogové prvky, jež se dají vytvořit pomocí jazyka HTML, nedostačují a je nutné vybudovat vlastní knihovnu pro vytváření uživatelského rozhraní.

Podle mého názoru se v budoucnosti uplatní více webové aplikace. Současný trend ukazuje, že náskok desktopových aplikací v tvorbě kvalitního rozhraní se rychle zmenšuje. Rozšíření internetových prohlížečů společně s oblibou internetu mluví jasně pro webové aplikace. V úvodu jsem si kladl požadavek, aby uživatel nemusel měnit svoje návyky a mohl používat nástroje, které má k dispozici. Rozhodl jsem proto realizovat publikační systém jako webovou aplikaci.

3.1.2 Volba externího textového procesoru

V kapitole 1.4.1 jsem na základě analýzy způsobu pořizování dokumentů stanovil jasný požadavek. Tvorba dokumentu včetně strukturované informace se odehrává v externím textovém nástroji, na který je uživatel zvyklý. Druhým významným bodem návrhu je proto výběr textového procesoru.

Následující seznam uvádí důležitá kritéria, která musí externí textový procesor splňovat:

- tvorba dokumentů se složitou strukturou (tabulky, grafy, odkazy),
- podpora šablon (schémat) dokumentů,
- jednoduché, intuitivní rozhraní,
- **dostupnost,**
- jednoduchá manipulace s dokumentem ze strany publikačního systému.

Vyznačil jsem především dostupnost nástroje, protože i velmi kvalitní textový procesor nesplní dobře svoji úlohu, pokud s ním uživatel neumí pracovat.

Po prozkoumání dostupných nástrojů pro pokročilou tvorbu textů jsem vybral **MS Word**. Jedná se o nejrozšířenější textový procesor. Práce s MS Word je podle mého názoru pohodlná a umožňuje vytvořit i dokumenty s komplikovanou strukturou. Nechybí podpora šablon, kde je možné definovat strukturu a vzhled dokumentu. S dokumenty vytvořenými v MS Word lze také dobře manipulovat programově. Blíže se této problematice věnuji v kapitole 4.2.

Existují také další kvalitní nástroje na vytváření textů. Jako příklad uvedu systém LATEX, ve kterém není problém precizně vysázet dokument se složitou strukturou. Jeho nevýhodou je ale menší rozšíření než je u MS Word a tím i větší pravděpodobnost, že uživatel se bude muset práci se systémem učit.

3.1.3 Databáze

Lze očekávat, že jednotlivá pravidla, směrnice, ale i samotná struktura organizace se bude v čase měnit, a proto bude potřeba jednotlivé dokumenty aktualizovat a publikovat jejich novou verzi. Z tohoto předpokladu vyplývá nutnost uchovávat dokumenty v databázi, aby se podle nových požadavků daly přepracovat. Nabízí se možnost využít služeb některého standardního databázového stroje (mySQL, Oracle, Gupta, …) nebo ukládat dokumenty přímo do souborového systému serveru.

První možnost poskytuje efektivnější vyhledávání a organizaci dat, ale opět přináší problém zmiňovaný v 3.1.1. Pokud uživatel nemá na svém počítači potřebný databázový stroj, musí ho instalovat. Z tohoto důvodu jsem volil **uchování dokumentů v souborovém systému**. Výhoda spočívá také v tom, že dokumenty je možné rychle a především kvalitně vytisknout. Server pošle uživateli dokument ve formátu MS Word a tisk je potom možné provést přímo pomocí textového procesoru.

3.2 Služby publikačního systému

3.2.1 Řízení přístupu

Mezi základní služby publikačního systému bude patřit řízení přístupu k jednotlivým dokumentům. Aby uživatel (zaměstnanec) mohl číst a publikovat dokumenty, musí se nejdříve do systému přihlásit. Vzhledem k tomu, že publikační systém je určený pro interní potřeby firmy, lze zaměstnance identifikovat podle jednoznačného jména jeho firemního počítače. Publikační systém proto nemusí obsahovat dialog pro zadání přihlašovacího jména a hesla a zaměstnanec je ušetřen zbytečné práce s opakovaným přihlášením do publikačního systému. Autentizace je ponechána na samotném serveru, který si ověří potřebné informace v operačním systému. Seznam zaměstnanců bude uložen v systémovém souboru uživatelů.

Zaměstnanec zastává ve firmě jistou funkci, ze které vyplývá, jaké dokumenty se ho týkají. Publikační systém bude umožňovat definovat pravidla přístupu. Každé pravidlo se skládá z množiny funkcí, pro které je určeno, a z testovací podmínky. Ta může blíže specifikovat aplikaci pravidla. Na základě pravidla systém vyhodnotí, jestli uživatel má právo dokument číst nebo dostane-li přístup pro publikaci dokumentu.

Publikační systém promítne omezení přístupu pro daného zaměstnance přímo do navigace systému. Zobrazí pouze položky menu, které odkazují na povolené dokumenty. Tento přístup je podle mého názoru lepší, než nabídnout uživateli kompletní navigaci a potom ho „zasypat“ spoustou hlášení, že vybraný dokument nemá právo číst. Uživatel pak zbytečně pátrá po důvodu, proč dokument nesmí číst, a to nemusí být žádoucí.

3.2.2 Navigace a vyhledávání

Dokumenty musí být přehledně tříděny do kategorií podle jejich obsahu. Pro jednoduchou navigaci využiji **víceúrovňové menu** s odkazy na příslušné dokumenty.

Pro snazší orientaci bude součástí stránek také **navigační cesta** mapující pozici prohlíženého dokumentu v hierarchii stránek. Tento navigační prvek bývá označován jako *breadcrumb menu*. Pokud například uživatel prohlíží dokument v oblasti *Sekce1* s názvem *Doc1*, publikační systém vypíše následující cestu:

Hlavní stránka > sekce1 > doc1

Hlavní stránka představuje odkaz na výchozí internetovou stránku, která se uživateli zobrazí po připojení na server. *Sekce1* je potom odkazem na dokument vyšší úrovně.

Součástí publikačního systému bude **fulltextové vyhledání**. Uživatel zadá hledané slovo nebo slovní spojení, systém prohledá všechny publikované dokumenty a vypíše odkazy na ty, ve kterých se

hledaný řetězec vyskytuje. U každého odkazu bude uveden text z blízkého okolí nalezeného slova. Uživatel si tak může ověřit, jestli je slovo nebo slovní spojení v požadovaném kontextu.

Pro snazší hledání dokumentů bude k dispozici **rejstřík** používaných pojmů a klíčových slov a také **katalog** globálních pojmů vyskytujících se v těchto dokumentech.

3.2.3 Odkazy

Správné používání odkazů je pro text velmi důležité. Uživatel se během čtení může okamžitě podívat na odkazovaný dokument nebo definici pojmu a neztrácí čas hledáním odkazovaného zdroje. Praxe na internetu bohužel ukazuje, že tento přístup řada autorů elektronických dokumentů nedodržuje. V textu se sice vyskytují slovní spojení jako „viz. článek XY" nebo „podívejte se do sekce Z", ale bez použití hypertextového odkazu. Uživatel je „slepými" odkazy rozptylován a dochází tak ke zhoršení čitelnosti textu.

Důležitou službou publikačního systému je proto správné sestavení odkazů při převodu publikovaného dokumentu do formátu HTML. Uživatel při pořizování dokumentu nezná adresu, na které se nachází odkazovaný zdroj. Uvede pouze jeho jednoznačný název a publikační systém musí všechny takovéto odkazy substituovat za příslušnou URL adresu popisující umístění zdroje.

3.2.4 Aktualizace dokumentu

Publikace dokumentu není pouze jednorázová, kdy se dokument vystaví a zpřístupní oprávněným čtenářům. Firemní dokumentace se v čase vyvíjí a jednotlivé dokumenty je potřeba aktualizovat. Publikační systém musí nabídnout rozhraní, prostřednictvím něhož bude možné upravit potřebný dokument.

Aktualizaci dokumentu jsem navrhl následujícím způsobem. Seznam publikovaných dokumentů bude reprezentován tabulkou obsahující jejich názvy. Po výběru konkrétního řádku se uživateli zobrazí dialog dokumentu. Zde budou zobrazeny bližší informace o vybraném dokumentu a také nabídka na stažení aktuální verze dokumentu a uložení nové verze. Uživatel si uloží soubor dokumentu na svůj klientský počítač, provede úpravy a aktualizovaný dokument pošle zpět na server. Publikační systém musí provést standardní proces jako při první publikaci a vytvořit HTML reprezentaci dokumentu.

Norma ISO 9000 požaduje, aby byli zaměstnanci informováni o všech změnách, které se jich týkají. K tomuto účelu využiji výchozí stránku publikačního systému. Zobrazím zde seznam odkazů aktualizovaných dokumentů pro daného uživatele. U každého dokumentu bude odkaz pro odsouhlasení změn.

3.3 Klasifikace dokumentů

Publikační systém bude pracovat s dokumenty, které se dají klasifikovat do několika kategorií podle následujících kritérií.

První možné rozdělení je podle toho, jestli se u dokumentu očekává, že bude v čase upravován. Rozlišuji dvě kategorie:

1. **Řízené** dokumenty budou umožňovat vytváření nových verzí. Dokumenty se dají dále rozdělit na:
    a. **SPM dokumenty (s**měrnice, **p**ravidla a dokumenty popisující **m**etodiku),
    b. **standardní** dokumenty – ostatní dokumenty, u nichž se očekává aktualizace.
2. **Neřízené** dokumenty se v čase měnit nebudou. Jedná se o obecné informace o systému nebo dokumenty, které se zobrazí při chybovém stavu nebo neúspěšném vyhledávání.

Dokumenty lze dále klasifikovat podle úlohy v publikačním systému na **systémové** a **uživatelské.** Uživatelské dokumenty se týkají zaměstnanců**.** Systémové dokumenty jsou nezbytné pro samotný publikační systém. Jejich přehled společně s významem je uveden v tab. 3-1.

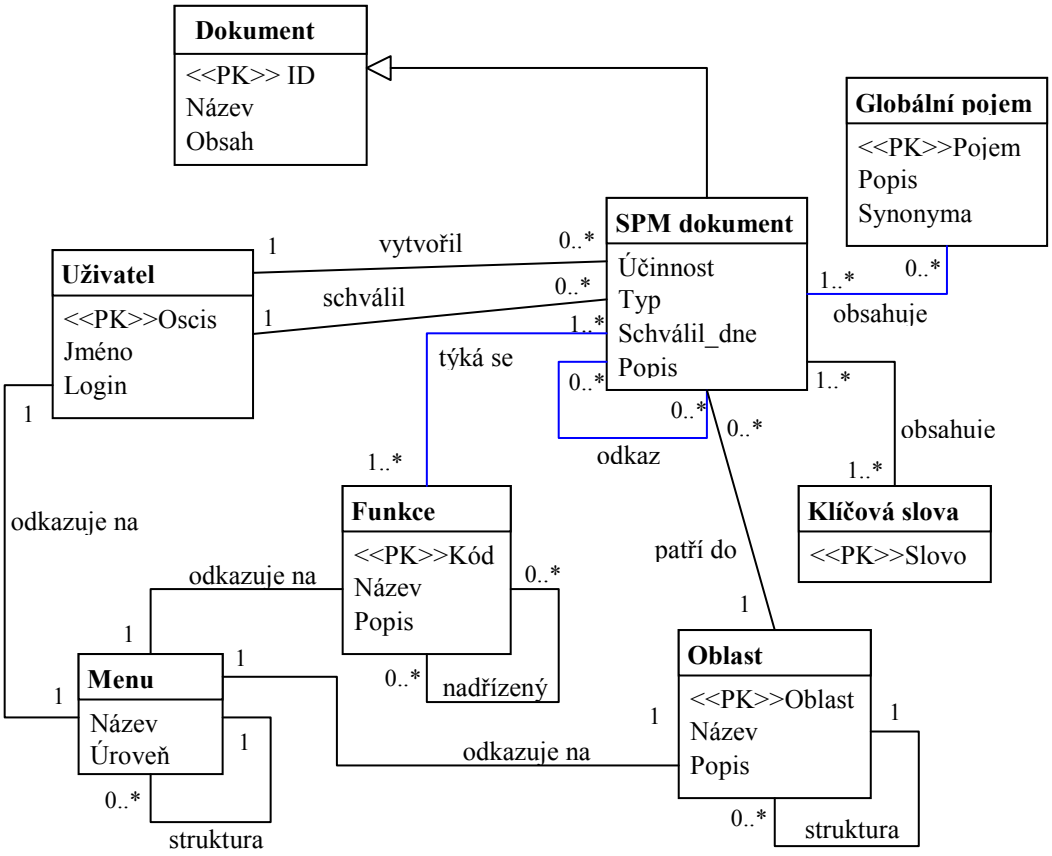
Typ	**Význam**
Menu	Popisuje jednotlivé položky strukturovaného menu, jejich pozici a úroveň.
Uživatelé	Seznam uživatelů s údaji pro přihlášení a odkazem na funkce, které zastávají.
Funkce	Popisuje hierarchické členění firmy, jednotlivé pracovní skupiny a funkce.
Oblast	Oblasti a podoblasti činnosti.
Globální pojmy	Seznam všech pojmů, na které je možné se v dokumentech odkazovat.

Tab. 3-1 Systémové dokumenty

Důležité je, že se všemi systémovými dokumenty se pracuje stejným způsobem jako s uživatelskými. Jejich pořizování a změny se provádí výhradně pomocí MS Word.

3.4 Konceptuální model

Stav publikačního systému je popsán množinou dokumentů. Jejich důležité atributy a vzájemné vztahy zachycuje UML diagram na obr. 3-1. Pro lepší přehlednost jsem vypustil vztah specializace entit *Uživatel, Menu, Funkce, Oblast* (systémové dokumenty) s entitou *Dokument.* Vztahy označené modrou barvou jsou obsaženy v obsahu dokumentu v podobě strukturované informace nebo odkazu.



Obr. 3-1 UML diagram

3.5 Vyjádření strukturované informace

Prvním krokem pro vyjádření strukturované informace je zvolit její syntaktický popis. Rozhodl jsem využít jednoduchou **tabulku**, která dovoluje přehledně uvést všechny důležité atributy dokumentu, které má uživatel při pořizování uvést. Tabulka bude nejčastěji figurovat v podobě hlavičky dokumentu, kde je soustředěna strukturovaná informace. Tabulku lze využít také pro dokumenty, které popisují například slovník pojmů. V takovém případě bude obsahem celého dokumentu jedna velká tabulka, jejíž řádky budou obsahovat informace o jednotlivých pojmech. Z předešlého vyplývá, že význam řádků a jednotlivých buněk tabulky je rozdílný. Jakým způsobem tedy sdělit uživateli i publikačnímu systému jejich sémantiku?

Pro uživatele můžeme každou buňku, resp. každý sloupec opatřit názvem vystihujícím její význam. Vycházím z předpokladu, že uživatel jednotlivým názvům polí či sloupců rozumí a bude je správně interpretovat.

Publikační systém musí mít k dispozici popis tabulky. Ten zahrnuje souřadnice buňky, její název a také přípustnou hodnotu. Na základě těchto informací může tabulku analyzovat a extrahovat strukturovanou hodnotu, kterou tabulka popisuje.

V nestrukturované části dokumentu budou také potřeba označit různé typy odkazů (na dokumenty, globální pojmy,…). Pro tuto potřebu využiji **pojmenovaného stylu formátování**. Uživateli nabídnu styl, který má použít pro daný typ odkazu. Opět význam stylu musí uživatel pochopit z jeho názvu. Publikační systém potom musí nahradit fragmenty textu označené tímto stylem za příslušné hypertextové odkazy.

3.6 Šablony

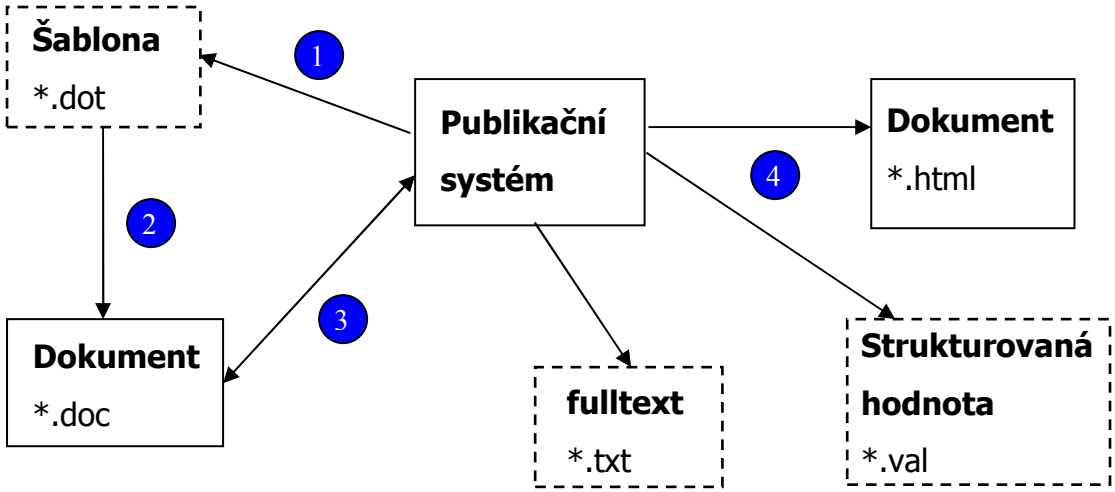
MS Word nabízí možnost založit nový dokument na šabloně. V ní je možné definovat potřebné styly pro formátování textu, tabulky, obrázky a další grafické prvky, které bude mít uživatel k dispozici pokaždé, když vytvoří nový dokument na dané šabloně. Výchozí šablonou pro nový dokument je standardně šablona *normal.dot*. Význam šablon dokumentů z hlediska publikačního systému rozdělím do dvou oblastí.

První se týká grafického vzhledu dokumentu. Se systémem bude pracovat více uživatelů a hrozí nebezpečí, že někteří využijí jiné styly, a bude tak porušena jednotnost dokumentu. Šablona nabízí řešení tohoto problému. Pomocí ní je uživateli hned na začátku pořizování jasně sděleno, jaké styly může používat. Programově lze také kontrolovat, jestli uživatel založil dokument na požadované šabloně.

Druhou významnou úlohu hraje šablona při pořizování strukturované informace. V kapitole 3.5 jsem pro syntaktický popis strukturované informace zvolil tabulku a pojmenovaný styl formátování. Díky šabloně lze tabulku s popisem polí nebo sloupců vložit do každého nového dokumentu založeného na této šabloně a uživatel jednoduše vyplní jednotlivá pole. Pojmenovaný styl formátování je prvkem množiny povolených stylů a jeho užití vyplývá z jeho názvu.

Použití šablon usnadňuje vytváření dokumentu. Intuitivně vede uživatele k pořizování strukturované informace a navíc mu ušetří mnoho práce, protože má k dispozici všechny potřebné styly a může se tedy soustředit na samotný obsah dokumentu. Díky šablonám lze jednoduše získat strukturovanou hodnotu dokumentu a také kontrolovat její správnost.

3.7 Schéma práce se systémem



Obr. 3-2 Schéma práce se systémem

Navrhl jsem základní rysy publikačního systému. Nyní popíši, jak bude probíhat práce se systémem. Obr. 3-2 ukazuje jednotlivé fáze pořizování a publikace dokumentu.

V prvním kroku si uživatel vybere typ dokumentu, který chce publikovat, a stáhne si ze serveru prázdný dokument založený na příslušné šabloně.

Druhý krok reprezentuje samotnou tvorbu dokumentu. Uživatel pomocí MS Word vytvoří obsah nového dokumentu. Používá přitom styly, které definuje šablona, a strukturovanou část vytvoří vyplněním příslušných buněk tabulky.

V dalším kroku uživatel pošle dokument na server s požadavkem na publikaci. Server analyzuje strukturovanou část dokumentu a pokud nalezne chybu, označí nevyhovující pole tabulky nebo špatné odkazy pomocí komentářů. Okomentovaný dokument je potom vrácen uživateli na opravení. Oboustranná šipka ve schématu naznačuje, že tato fáze může probíhat několikrát, dokud uživatel nedodá dokument s validní strukturovanou informací.

Náplní čtvrtého kroku jsou akce, které provede publikační systém po úspěšném přijmutí publikovaného dokumentu. Kromě transformace dokumentu do formátu HTML vygeneruje publikační systém dva pomocné soubory. První obsahuje extrahovanou strukturovanou informaci, aby publikační systém nemusel opětovně analyzovat celý dokument. Druhý reprezentuje obsah dokumentu jako prostý text pro fulltextové vyhledávání.

4 Serverová část aplikace

4.1 Volba programovacího jazyka

Pro implementaci serverové části aplikace jsem vybral jazyk **Visual Basic**. Na následujících řádcích se pokusím objasnit, co mě vedlo k této volbě.

Serverová část publikačního systému bude velmi často potřebovat interakci s aplikací MS Word. Bude třeba analyzovat obsah dokumentů určených pro publikaci a také převádět obsah dokumentů do formátu HTML pro zobrazení ve formě internetových stránek. Hledal jsem proto jazyk, který by umožňoval jednoduše a efektivně pracovat s vybraným textovým procesorem. Další podmínkou výběru byla nutnost implementovat webovou aplikaci a v poslední řadě také moje zkušenosti s daným jazykem.

MS Word podporuje tvorbu maker, pomocí nichž lze přistupovat k objektovému modelu aplikace a manipulovat s obsahem dokumentů. Pro jejich vytváření slouží jazyk VBA (Visual Basic for Application). Jedná se o objektově orientovaný jazyk, jehož průnik s jazykem Visual Basic je významný. Ve Visual Basic lze pracovat s aplikací MS Word stejným způsobem. Tento přístup navíc řeší problém spojený s používáním maker. Jejich přítomnost v dokumentu totiž představuje bezpečností riziko, a uživatel je tak nucen odsouhlasit varovný dialog před tím, než může dokument použít.

Visual Basic je vhodný i pro tvorbu webových aplikací. Integrované prostředí dodávané s jazykem nabízí možnost vytvořit aplikaci, která je přímo spustitelná na IIS (Internal Information Server). Práce s prostředím je příjemná, umožňuje ladění programu přímo na serveru.

Z těchto důvodů jsem pro práci na serverové části publikačního systému vybral právě jazyk Visual Basic.

4.2 Objektový model MS Word

Veškerá komunikace s aplikací MS Word je realizována prostřednictvím objektového modelu aplikace. Model je hierarchicky uspořádaný a na jeho vrcholu se nachází objekt *Application*. Pod ním nalezneme velké množství dceřiných objektů, pomocí nichž lze přistupovat k otevřeným dokumentům, manipulovat s jejich obsahem, realizovat vyhledávání, atd. Cílem této kapitoly není popsat celý model, ale uvést stěžejní objekty a jejich důležité metody a vlastnosti, které jsem využil při tvorbě publikačního systému.

4.2.1 Získání objektu aplikace

Pro programové používání MS Word je nejdříve třeba získat objekt aplikace. Visual Basic umožňuje přidat komponentu MS Word, která zpřístupní objekt aplikace již v době kompilace. Tento přístup má ovšem významné omezení. V případě, že v systému bude nainstalována jiná verze MS Word, než je verze komponenty používané programem, může dojít k nepříjemným problémům plynoucím z nekompatibility verzí.

Rozhodl jsem proto vytvářet objekt aplikace **dynamicky** až za běhu aplikace. Využívám metody *CreateObject("Word.Application")*, která vrátí instanci aplikace v aktuální verzi, jež je k dispozici v systému.

4.2.2 Použité objekty

Tab. 4-1 přehledně popisuje objekty, které jsem využil při implementaci publikačního systému.

Název objektu	**Popis**
Application	Objekt aplikace. Otevřené dokumenty jsou přístupné pomocí kolekce *Documents*. Pomocí metod *open()*, *save()* a *saveAs()* kolekce lze otevírat/ukládat dokumenty.
Document	Model dokumentu.
Table	Reprezentuje tabulku. Vlastnost *Rows* obsahuje kolekci řádků. Každý řádek *Row* má potom vlastnost *Cells* poskytující přístup k jednotlivým buňkám řádku *Cell*.
Range	Představuje „obsah“ objektu. Je přístupný přes vlastnost *Range* příslušného objektu (například tabulky, dokumentu, atd.).
Find	Slouží pro vyhledávání v oblasti (objekt *Range*). Nabízí řadu vlastností korespondujících s poli formuláře pro vyhledávání. Vyhledávání se spouští metodou *Execute()*. Objekt *Range* je potom modifikován podle výsledku hledání.

Tab. 4-1 Použité objekty modelu aplikace MS Word

4.3 Webová aplikace ve VB

Pro vývoj webových aplikací umožňuje Visual Basic založit projekt typu **IIS Application**. Základem každého takového projektu je speciální třída *Webclass*, kde je implementováno, jakým způsobem bude server odpovídat na dotazy klienta. Jeho součástí jsou obvykle šablony internetových stránek ve formátu HTML, do nichž je dynamicky doplněn obsah podle konkrétního požadavku. Jednotlivé části *Webclass* se označují jako tzv. *Webitems* a vytváří se pomocí návrháře (designer). Celá třída je umístěná v modulu návrháře (*.Dsr).

Pro každý *Webclass* je automaticky vygenerován ASP(Active Server Pages) skript. Když klient posílá žádost na server, spustí se nejdříve tento skript, který vyvolá událost *Start()* třídy *Webclass*.

Zároveň skript zpřístupní **serverové objekty**, prostřednictvím kterých aplikace získá podrobné informace o žádosti klienta a může realizovat výstup. Serverové objekty shrnuje tab. 4-2.

Název objektu	**Popis**
ASPError	Informace o chybách při vykonávání ASP skriptu.
Request	Obsahuje informace o HTTP požadavku (odeslané formuláře, serverové proměnné a cookies).
Response	Reprezentuje odpověď serveru (zobrazovaný obsah, hlavičky).
Server	Zpřístupňuje serverové metody a vlastnosti.
Session	Umožňuje uchovávat informace během sezení.

Tab. 4-2 Serverové objekty

4.4 Struktura projektu

Řešení publikačního systému jsem rozložil do více modulů. Tab. 4-3 popisuje jejich význam. Moduly s příponou *.cls* představují třídy, pomocné procedury jsou uloženy v modulech s koncovkou *.bas*. Stěžejním třídám projektu se budu věnovat v následujících kapitolách.

Název modulu	**Popis**
Doc.cls	Model dokumentu.
DocBase.cls	Databáze dokumentů.
DocBaseGlobal.bas	Globální procedury.
Expression.cls	Kompilátor a interpret výrazů.
Rules.cls	Správa bezpečnostních pravidel.
Tree.cls	Operace nad acyklickým grafem.
UploadedFile.cls	Objekt natahovaného souboru.
Values.cls	Model univerzální hodnoty.
webDocBase.Dsr	Hlavní webová aplikace.
WebDownload.cls	Zapouzdření HTTP protokolu pro download souborů.
WebUpload.cls	Zapouzdření HTTP protokolu pro upload souborů.
Word.bas	Operace nad aplikací MS Word.

Tab. 4-3 Popis modulů projektu

4.5 Koncept univerzální hodnoty

Koncept univerzální hodnoty (dále UV – universal value) realizuje třída ***Values.cls***. V publikační systému budu neustále dynamicky pracovat se strukturovanou hodnotou. Využiji ji pro reprezentaci

strukturované části dokumentu, ale i pro popsání obsahu jednotlivých internetových stránek. Jakýkoli dotaz nad databází dokumentů vrátí UV. Potřebuji tedy pružnou strukturu, jejíž vlastnosti budou pokaždé jiné.

Jednoduchá hodnota je v UV tvořena dvojicí: (jméno, hodnota). Hodnota může být i složená. V takovémto případě kromě jména a hodnoty (vlastní hodnotu mít nemusí) obsahuje seznam dceřiných hodnot, které mohou být opět složené. Strukturovaná hodnota tak vytváří n-ární strom. Uložené hodnoty mohou být typu řetězec, číslo, boolean (true, false) nebo se může jednat o množinu řetězců (seznam hodnot).

Pro uložení UV jsem chtěl původně využít XML. Tento formát byl však náročný na zpracování a práce s UV byla pomalá. Sáhl sem proto k rychlejší a také přehlednější textové reprezentaci. Pro oddělení jednotlivých úrovní (pater stromu) používám jednu mezeru. Př. 4-1 ukazuje reprezentaci konkrétní UV.

Př. 4-1 Ukázka UV

```
Data
  Type = "form"
  LabelSep = ":"
  Check = "ucinnost>=schvalil"
  Loc = 1
  Item
   docID
    Type = "text"
```

Za povšimnutí stojí vlastnost *Check*, jejíž hodnotu tvoří výraz. Vytvořil jsem jednoduchý jazyk umožňující manipulaci s UV. Blíže se mu budu věnovat v další kapitole.

4.5.1 Jazyk pro manipulaci s univerzální hodnotou (UVL)

Při práci s UV se ukázalo jako velmi užitečné mít k dispozici jednoduchý jazyk, který by umožňoval vytvářet výrazy. Implementoval jsem podporu standardního výrazu, jak je známa z jiných jazyků.

S numerickým typem lze provádět základní aritmetické operace (sčítání, odčítání, násobení, dělení). V publikačním systému se velmi často pracuje s řetězci. Jazyk proto umožňuje jejich zřetězení a nabízí prostředky pro získání podřetězce. Pro vytváření šablon generovaných stránek, které jsou popsány v kapitole 4.8.2, bylo důležité zařadit podporu podmíněného větvení. Lze tedy používat logické výrazy. Velmi často je také potřeba procházet seznam dceřiných hodnot, a proto jazyk podporuje jednoduchý cyklus. Zařadil jsem také množinové operace. Důvodem bylo především vytváření bezpečnostních pravidel. Jednoduše je potom možné testovat, zda-li funkce přihlášeného uživatele je prvkem množiny povolených funkcí.

4.6 Dokument

Nejdůležitější třídou projektu je třída reprezentující model dokumentu, která popisuje významné vlastnosti a operace nad publikovaným dokumentem. Reprezentuje ji třídní modul ***Doc.cls***.

4.6.1 Extrakce strukturované části

Základem práce s dokumentem je získání jeho strukturované informace. Cílem je vyjádřit strukturovanou hodnotu dokumentu pomocí UV. Extrakce strukturované části je důležitá nejen pro samotnou publikaci, ale i pro operaci kontroly strukturované informace dokumentu.

V kapitole 3.5 návrhu jsem zvolil pro syntaktické vyjádření strukturované informace tabulku, která je obsažena v šabloně dokumentu. Publikační systém má pro každou šablonu uložena metadata, kde je popsána vazba buněk tabulky na jednotlivé vlastnosti struktury. Metadata jsou rovněž vyjádřena pomocí UV. Př. 4-2 ukazuje popis strukturované hodnoty systémového dokumentu uživatelů. Tučně jsou vyznačeny důležité vlastnosti struktury. Vzhledem k rozsahu metadat jsem uvedl pouze jejich část.

Př. 4-2 Metadata uživatelů

```
Uziv
 Name = "Seznam uživatelů"
 Data
  Type = "tab"
  Loc = 1
  start = 2
  row
   Type = "row"
   oscis
    Type = "num"
    Loc = 1
   jmeno
    Type = "text"
    Loc = 2
   funkce
    Type = "set"
    Loc = 3
    Sep = ","
    Range = "Funkce.kod"
```

Popis strukturované informace obsahuje složená hodnota *Data*. Nejdříve jsou uvedeny informace o způsobu uložení strukturované části dokumentu. Vlastnost *Type* udává, že půjde o tabulku. Jedná se o první tabulku dokumentu (*Loc*) a analýza začne druhým řádkem (*start*). V prvním jsou názvy sloupců, aby uživatel chápal význam buněk tabulky.

Analýza dále pokračuje po řádcích (*row*). Důležitá je identifikace jednotlivých buněk v rámci řádku. Pozice buňky je dána opět vlastností (*Loc*). U každé je navíc uveden typ hodnoty, která se má

v buňce vyskytovat. Během analýzy je tak možné přímo kontrolovat, jestli uživatel strukturovanou informaci pořídil správně.

Výsledkem zpracování dokumentu je vytvoření UV. Její textovou reprezentaci uvádím v př. 4-3.

Př. 4-3 Strukturovaná hodnota vyjádřená UV

```
Uziv
 Data
  row
   oscis = 1
   jmeno = "Máčel Lukáš"
   funkce = {"RSPO","RVYR","FPRP"}
  row
   oscis = 2
   jmeno = "Dvořáková Marie"
   funkce = {"ROBP","FMPP"}
```

Díky UVL je možné používat také výrazy. Není problém vyjádřit i složitější kontrolu vstupu. Ukázkou je použití vlastnosti *Range* u buňky reprezentující funkce uživatele. Všechny prvky množiny (*Type = "set"*) musí být existující funkce. Seznam funkcí je v systémovém souboru *Funkce.values*, který má již publikační systém převedený do UV a není tedy problém zjistit, jestli se zde hodnota funkce vyskytuje.

4.6.2 Analýza textu dokumentu

Kromě strukturované informace je třeba analyzovat také zbývající text dokumentu. Mohou se zde vyskytovat odkazy na jiné dokumenty, funkce nebo globální pojmy. Odkaz je identifikován na základě jednoznačně pojmenovaného stylu formátování. Pro každý typ odkazu se provede vyhledání nad celým textem. Pro tyto potřeby využívám **objekt *Find*** aplikace MS Word.

Po nalezení odkazu je třeba provést dvě operace. Na místo stylu je vložen funkční odkaz, který se při transformaci do HTML formátu nahradí hypertextovým odkazem. Pochopitelně se ověřuje, jestli je odkaz v pořádku. Druhou akcí je aktualizace systémového dokumentu globálních pojmů (*DocPojmy.values*), aby se nové pojmy objevily v rejstříku pojmů.

Při nahrazování odkazů na dokumenty může dojít k situaci, že právě publikovaný dokument odkazuje na jiný, který bude zveřejněn teprve po něm. V takovéto případě nejde o chybu. Doplnění a kontrola cyklických odkazů se provádí později.

4.6.3 Transformace do HTML formátu

Důležitou operací publikace dokumentu je převod dokumentu ve formátu Word na HTML formát. Na první pohled lze složitou transformaci jednoduše vyřešit použitím metody ***SaveAs()*** s parametrem *HTML*, kterou nabízí objekt aplikace MS Word. Dojde k automatickému převodu komplikované

struktury dokumentu na HTML stránku. Všechny styly jsou vyjádřeny pomocí kaskádových stylů. Vyřeší se také transformace tabulek. Vygenerovaná stránka obsahuje všechny obrázky a použité odkazy.

Převod ovšem není úplně bezproblémový. Je nutné provést ještě drobné korekce. Všechny kaskádové styly ve stránce extrahuji do samostatného souboru **.css*. U několika je nutné pozměnit pravidla stylu. Dále je třeba opravit textové a obrázkové odkazy. MS Word nevygeneruje správnou URL adresu odkazu.

4.6.4 Kontrola a publikace

Dostávám se k dvěma stěžejním operacím s dokumentem. První je kontrola strukturované hodnoty a druhou samotná publikace.

Kontrola spočívá v získání strukturované části dokumentu. Podle metadat se pro každé pole ověří, zda hodnota má správný typ, a provedou si případně další kontroly, které jsou součástí popisu strukturované hodnoty. Dále se zjišťuje platnost jednotlivých odkazů v textu na funkce a globální pojmy. Odkazy na dokumenty se v této fázi neověřují, protože ještě odkazovaný dokument nemusí existovat.

Publikace dokumentu nejdříve znovu zkontroluje strukturovanou část dokumentu a potom přistoupí k samotné transformaci vstupního dokumentu. Jsou generovány a uloženy tyto soubory:

- strukturovaná hodnota jako UV (**.values*),
- reprezentace v HTML (**.html*),
- kaskádové styly (**.css*),
- textová verze (**.txt*).

Textová verze dokumentu slouží pro fulltextové vyhledávání. Vytvoří se odstraněním všech tagů a přebytečných bílých znaků z HTML reprezentace dokumentu. Operace vyhledání potom prochází všechny textové reprezentace publikovaných dokumentů a snaží se nalézt požadovaný řetězec, který zadal uživatel prostřednictvím rozhraní pro vyhledávání.

Všechny publikované SPM dokumenty jsou uloženy v registru řízených dokumentů (*SPMDoc.values*). Při zveřejnění nového dokumentu musí dojít k aktualizaci registru. Rovněž se provedou příslušné změny v rejstříku globálních pojmů a klíčových slov (*DocPojmy.values*).

4.7 Databáze dokumentů

Stav publikačního systému představuje databáze zveřejněných dokumentů. Základní služby databáze jsou realizovány v třídě ***DocBase.cls***.

Databázový model tvoří množina hierarchicky pojmenovaných dokumentů. U každého dokumentu je možné si vybrat reprezentaci, s kterou chceme pracovat. Dokument je k dispozici minimálně ve formě UV (**.values*). Může mít také další formy *(*.html, *.txt, *.css, *.doc*).

Databáze uchovává strukturované hodnoty všech publikovaných dokumentů. Můžeme ji tedy vnímat jako jedinou UV. Důležitou službou databáze je vyhledávání, pomocí kterého jsou realizovány všechny kontroly strukturovaných částí dokumentů. V př. 4-2 bylo například potřeba ověřit hodnotu uvedenou v poli funkce. Pro tento úkol stačí požádat databázi o UV dokumentu funkcí a vyhledat postupně uvedené funkce.

4.7.1 Podpora zabezpečení

V návrhu jsem uvedl, že publikační systém musí řídit přístup k publikovaným dokumentům podle toho, jaký uživatel se systémem pracuje. Důležitou službou databáze dokumentů je tedy přihlášení uživatele.

Login uživatele je získán od IIS serveru. Databáze prochází UV dokumentu uživatelů (*Uziv.values*) a snaží se nalézt uživatele s příslušným loginem. Pokud neuspěje, uživateli je přístup odepřen. V opačném případě je získána identita uživatele v systému.

Pro uživatele jsou určeny pouze některé dokumenty. Publikační systém proto obsahuje bezpečností pravidla, pomocí kterých se uživateli zobrazí jenom to, co má vidět. Bezpečnostní pravidla jsou definována ve speciálním souboru *Restrict.values*, který spravuje administrátor. Neexistuje pro něj dokument ve formátu Word. Pravidlo týkající se uživatele je vybráno na základě jeho funkcí uvedených v dokumentu uživatelů.

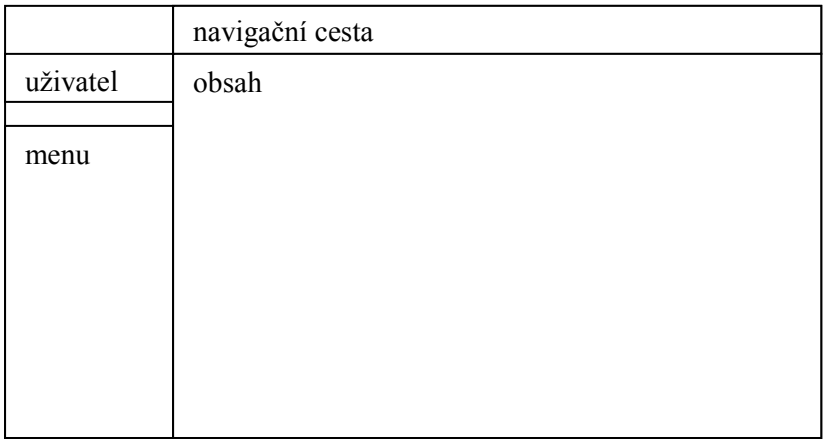
Pravidlo umožňuje definovat podmínku nad požadovanou URL stránky a zamezit tak jejímu zobrazení. Důležitá je také možnost filtrovat samotná data. V UV hodnotě se potom objeví pouze povolené hodnoty. Automaticky se tak pracuje s obsahem přípustným pro daného uživatele.

4.8 Generování stránek publikačního systému

Veškerý kód spojený s generování stránek publikačního systému je umístěn v návrháři webové třídy ***webDocBase.Dsr***.

4.8.1 Schéma stránek

Publikační systém je tvořen soustavou vzájemně provázaných internetových stránek. Kostru tvoří hlavní stránka (master page), která definuje pozici navigačních prvků a prostor pro zobrazení samotného obsahu. Následující obrázek ukazuje schéma stránky, která se zobrazí uživateli publikačního systému.



Obr. 4-1 Schéma stránek

Oblast *uživatel* zobrazuje jméno přihlášeného uživatele. Po levé straně je umístěno *menu* pro navigaci po stránkách. V prostoru označeném jako *obsah* se objeví stránka dokumentu, kterou chce uživatel prohlížet. Nad dokumentem se nachází *navigační cesta* mapující pozici prohlíženého dokumentu v hierarchii stránek.

4.8.2 Šablony stránek

Podobně jako dokumenty vytvářené uživatelem mají svoji šablonu i generované stránky publikačního systému. Tím je docíleno oddělení obsahu stránky od jeho vzhledu. Šablona vystupuje v databázi jako standardní dokument, s kterým se pracuje ve formátu HTML.

Pro vložení obsahu do šablony se používá šablonový jazyk založený na UVL. Pomocí znaku „%“ se definuje oblast, ve které je možné jazyk používat. Zbytek šablony tvoří HTML tagy.

Procedura generující stránku načte dokument šablony a vytvoří požadovaný obsah reprezentovaný UV. Potom dojde k interpretaci šablony proti tomuto obsahu. Výsledkem je čistý HTML kód s popisem stránky, která je zobrazena uživateli.

Použití šablony ukazuje př. 4-5. Úkolem je vygenerovat přehled publikovaných dokumentů. Nejdříve se pomocí databáze sestaví UV reprezentující obsah generované stránky. Pro každý dokument budu uchovávat informaci o ID, názvu, datu jeho pořízení a také URL popisující cestu k dokumentu. Výsledná UV má tuto strukturu:

Př. 4-4 Obsah stránky vyjádřený pomocí UV

```
data
 radky
  row
   datum = "25.4.2007"
   kod = "D0001"
   link = "docbase.asp?ID=D0001"
   nazev = "jmeno dokumentu"
   popis = "popis"
```

Interpretace šablony uvedené v př. 4-5 probíhá následujícím způsobem. Příkaz *%(radky%* představuje cyklus přes všechny dceřiné hodnoty vlastnosti *radky*. V každém průchodu cyklu se tedy vygeneruje část šablony uzavřená mezi *%(radky%* a *%)* a do výsledné tabulky se pro každý dokument přidá samostatný řádek. Na příslušná místa je dosazena hodnota požadované vlastnosti pomocí příkazu *=jmeno_vlastnosti*. V buňkách řádku se potom objeví vybrané informace o dokumentu.

Př. 4-5 Šablona generované stránky

```
<h1>Přehled dokumentů</h1>
   <table class="clean">
      <tr>
         <th class="col1">ID</th>
         <th class="col2">Název</th>
         <th class="col4">Publikováno</th>
      </tr>
      %(radky%
      <tr>
         <td class="col1">
           <a href="%=link%" title="%=popis%">%=kod%</a></td>
         <td class="col2">
           <a href="%=link%" title="%=popis%">%=nazev%</a></td>
         <td class="col4">%=datum% </td>
      </tr>
      %)%
   </table>
```

5 Implementace klientské části

Uživatel komunikuje se serverem prostřednictvím internetového prohlížeče. Pomocí protokolu HTTP posílá požadavky na zobrazení jednotlivých dokumentů. Server žádosti zpracuje a zašle stránku ve formátu HTML, která se objeví uživateli v okně prohlížeče.

Pro prohlížení zveřejněných dokumentů model žádost-odpověď naprosto dostačuje a na straně klienta proto není potřeba provádět žádné akce. Problém nastává v okamžiku, kdy je potřeba vytvořit rozhraní pro publikaci dokumentů nových a umožnit také aktualizaci dříve zveřejněných dokumentů.

Uživatel pracuje s rozhraním a očekává okamžitou reakci na akce, které provádí. Tento základní požadavek ovšem splněný není, protože zpracování žádosti trvá určitou dobu, po kterou nelze s rozhraním pracovat. Stránka reaguje na podněty uživatele pomalu a rozhraní se nedá rozumně používat. Podstata problému spočívá v samotném protokolu HTTP, který je primárně navržen pro prohlížení statického obsahu a ne pro potřeby interaktivní webové aplikace.

V kapitole 3.1.1 jsem uvedl, že v současné době existuje způsob, jak nastíněný problém vyřešit. Jedná se nový přístup označovaný jako AJAX. Předtím, než se budu věnovat implementaci klientské části, proto uvedu strategii, kterou AJAX využívá pro překonání problémů spojených s modelem žádost-odpověď.

5.1 AJAX

Zkratka AJAX je z anglického **Asynchronous JavaScript And XML**. Tento přístup využívá následující technologie:

- základy internetové prezentace - XHTML a CSS,
- XML pro výměnu dat,
- DOM model internetových stránek pro jejich dynamickou změnu,
- XMLHTTPRequest pro asynchronní komunikaci se serverem,
- JavaScript pro propojení předešlých částí.

5.1.1 Problémy modelu žádost-odpověď

Standardní komunikace klient-server přináší dva základní problémy, které brání vytvářet interaktivní aplikaci.

Prvním je objem dat posílaných na klienta. Server pokaždé odešle úplně novou stránku. Aktualizace rozhraní přitom vyžaduje pouze malé změny. Opakované generování stejného obsahu významně zpomaluje celou aplikaci.

Druhý problémem je synchronní způsob komunikace. Klient čeká, až server odpoví. Během zpracování dotazu není možné provádět jakékoliv akce. Práce s rozhraním tak postrádá potřebnou plynulost, která je pro uživatele důležitá.

5.1.2 Řešení komunikace klient-server

Pro redukci posílaných dat ze serveru nabízí AJAX dvě strategie:

1. provádění aktualizace přímo na klientu pomocí DOM modelu stránky,
2. vybudování datového mostu na server pro získání nových dat.

Změny rozhraní v důsledku akce uživatele nemusí nutně vyžadovat nová data. Někdy je potřeba pouze skrýt nebo naopak zobrazit některé prvky rozhraní. DOM model umožňuje přistupovat k obsahu stránky a změnit ji podle aktuální potřeby. Na server není třeba posílat žádný dotaz a odezva rozhraní je velmi rychlá.

Pokud se změna rozhraní neobejde bez kontaktování serveru, vstupuje do hry objekt XMLHTTPRequest. Pomocí něho je možné vybudovat datový most, po kterém server pošle pouze potřebná data. Ty jsou potom prostřednictvím DOMu použity pro modifikaci rozhraní. Nejčastěji se data posílají ve formě XML.

Pro odstínění modelu žádost-odpověď zbývá vyřešit synchronizaci klienta na odezvu serveru. Objekt XMLHTTPRequest nabízí možnost poslat dotaz asynchronně. Uživatel tedy může využívat rozhraní během doby, kdy se na serveru sestavuje odpověď. Jakmile data dorazí, dojde k jejich zpracování a stránka je aktualizována. Asynchronní komunikace představuje klíčovou strategii, která otevírá cestu k interaktivní webové aplikaci. Podle ní je pojmenován i celý přístup.

5.1.3 Document object model (DOM)

Při interpretaci zdrojového kódu stránky klientským prohlížečem vzniká také objektový model dokumentu. Pomocí JavaScriptu, který může být součástí stránky, je možné s modelem pracovat a měnit tak vzhled a obsah dokumentu.

DOM model je tvořen množinou objektů reprezentujících jednotlivé elementy dokumentu. Tyto objekty vytvářejí n-ární strom dokumentu, jehož vrcholem je objekt reprezentující samotný dokument. Jednotlivé uzly stromu jsou provázány ukazateli, prostřednictvím kterých lze stromem procházet. DOM model umožňuje vytvářet dynamicky nové uzly a libovolný uzel zrušit nebo jej přemístit na jiné místo ve stromě.

5.1.4 Událostní model

Pokud má rozhraní reagovat na podnět uživatele, musí umět rozpoznat, že na něm uživatel provedl nějakou akci. V modelu dokumentu jsou pro tuto potřebu implementovány tzv. **události**. Každá uživatelská akce (posunutí myši, kliknutí na některý element dokumentu, rozbalení rolovacího menu),

ale i natažení internetové stránky, vytvoří událost. Událost je reprezentována objektem, který nese důležité informace o jejím vzniku. Nalezneme zde typ události, její cílový element a další vlastnosti jako poloha kurzoru myši, kombinaci kláves při události vyvolané z klávesnice aj.

Jakým způsobem ale vzniklou událost odchytit a zpracovat? DOM model umožňuje na libovolný uzel stromu dokumentu připojit tzv. **posluchač událostí** (Event Listener). Prostřednictvím něho se určí, na jaký typ události se bude reagovat a jak bude tato událost zpracována. Pokud dojde na elementu k události správného typu, posluchač automaticky zavolá příslušný kód pro obsluhu. Element může mít více posluchačů. DOM model nabízí i prostředek pro jejich zrušení.

5.2 Knihovna pro webové rozhraní

Rozhraní publikačního systému pro publikování a aktualizaci dokumentů vyžaduje používání složitějších dialogových prvků. Bude potřeba vytvořit komplexní formulář, jehož pole budou vzájemně provázaná a také tabulku pro zobrazení publikovaných dokumentů, která musí nabídnout segmentaci obsahu.

Jazyk HTML nabízí pouze základní dialogové prvky. Jejich chování sice lze ovládat pomocí událostí DOMu, ale budování rozhraní je značně komplikované a stojí mnoho úsilí. Každý prvek podporuje jiné události. Jeho hodnota se nastavuje jiným způsobem. Chybí zde jednotný přístup pro manipulaci s dialogy. Celou situaci ještě zhoršuje odlišné chování prvků v různých internetových prohlížečích.

Rozhodl jsem se proto vytvořit knihovnu pro webové aplikace, která by usnadnila budování rozhraní. Snažím se zapouzdřit všechny základní dialogové prvky tak, aby nabízely **jednotné rozhraní**, dalo se jednoduše definovat jejich chování a vzájemné vazby.

Snažil jsem se využít všechny výhody, které nabízí přístup AJAXu. Knihovna podporuje asynchronní komunikaci se serverem. Kromě toho jsem přidal možnost vytvářet složitější ovládací prvky jako komplexní formulář nebo segmentovací tabulka.

5.2.1 Komplexní formulář

Komplexní formulář je reprezentován třídou **domForm**. Cílem abstrakce je nabídnout jednotné rozhraní pro práci s jeho poli. Vytvořil jsem třídu **formElement**, která představuje abstraktní pole formuláře.

Nastavení a získání hodnoty pole je realizováno pomocí metod *setValue()* a *getValue()*. Manipulaci s polem formuláře je možné zamezit pomocí metody *disable()*. Z důležitých vlastností uvedu popisek pole *label*, jehož získání je v klasickém HTML formuláři komplikované.

Ze třídy formElement potom dědí další typy dialogů. Je jich celkem šest (tlačítko, textový vstup, rozbalovací nabídka, přepínač a zatrhávací pole). Každý implementuje předepsané metody,

zapouzdřuje rozdíly v manipulaci s prvkem a podporuje základní události. V HTML existuje mnohem více typů, ale funkčně vždy spadají do jedné z uvedených kategorií.

Třída domForm se vytváří na základě standardního formuláře, jehož element je předán konstruktoru třídy. Při prvním natažení se analyzuje obsah formuláře a vytvoří se kolekce objektů zabalujících původní pole formuláře. Díky tomuto přístupu se potom s formulářem jednoduše pracuje a je možné vytvářet i komplexní formuláře.

5.2.2 Segmentovací tabulka

Součástí rozhraní pro aktualizaci dokumentů bude tabulka obsahující seznam všech zveřejněných dokumentů. Seznam může být rozsáhlý, a proto tabulka musí podporovat segmentaci obsahu. Součástí knihovny je dialogový prvek, který tuto funkci podporuje.

Obr. 5-1 Segmentovací tabulka

Segmentovací tabulka je reprezentována třídou **domTable**. Zobrazuje pouze definovaný počet řádků. Při pravém okraji tabulky je umístěn posuvník, pomocí kterého je možné procházet celý obsah tabulky. Řádky jsou průběžně aktualizovány prostřednictvím AJAXu. Pohyb posuvníku vyvolá událost, na základě které je vyslán asynchronní požadavek na server pro zaslání nového segmentu řádků.

6 Závěr

Navrhl a vytvořil jsem publikační systém, který umožňuje zveřejňovat strukturované dokumenty v elektronické podobě.

Publikační systém realizuji jako webový server. Uživatelé se k tomuto serveru připojí a pomocí internetového prohlížeče mohou zobrazit jednotlivé publikované dokumenty v podobě internetových stránek. Je možné definovat bezpečnostní pravidla a pomocí nich řídit přístup k jednotlivým dokumentům. Všechny publikované dokumenty jsou uloženy v souborovém systému serveru.

Pro pořizování dokumentů jsem zvolil nejrozšířenější textový procesor MS Word. Uživatel si stáhne dokument založený na šabloně, která definuje grafický vzhled dokumentu a popisuje také jeho strukturovanou část. Ta je syntakticky vyjádřena pomocí jednoduché tabulky.

Po vytvoření je dokument odeslán na server, který zkontroluje strukturovanou část dokumentu. Pokud dokument obsahuje chybné hodnoty atributů, dojde k jeho vrácení uživateli. Dokument obsahuje chyby vyznačené pomocí komentářů. V opačném případě se provede transformace a výsledný dokument je zveřejněn oprávněným uživatelům.

Publikační systém nabízí přehlednou navigaci. Dokumenty jsou roztříděny do kategorií podle obsahu. Pro snadné vyhledávání lze použít fulltext nebo rejstřík globálních pojmů a klíčových slov. K dispozici je webové rozhraní pro modifikaci dříve publikovaných dokumentů.

Implementovaný publikační systém slouží pro vedení firemní dokumentace v oblasti managementu jakosti. Struktura dokumentů je navržena v souladu s normou ISO 9000. Publikační systém rovněž splňuje požadavek na seznamování zaměstnanců s novými verzemi dokumentů.

6.1 Přínos práce

Za největší přínos práce považuji možnost pořizovat strukturovaný dokument pomocí externího textového procesoru. Uživatel pracuje s nástroji, na které je zvyklý. Vytváření dokumentu je pro něj pohodlné. Strukturovaná informace vyjádřená tabulkou plní dobře funkci běžného formuláře. Díky připraveným šablonám má uživatel k dispozici všechny formátovací styly a může se plně soustředit na samotný obsah dokumentu. Pokud umí pracovat s MS Word, pak zároveň dokáže i sám publikovat. Mým hlavním cílem bylo vyjít vstříc uživateli a využít maximálně jeho znalosti. To se podle mého názoru podařilo.

ISO 9000 představuje mezinárodně uznávaný standard. V současné době řada firem zavádí systém řízení kvality. Prototyp publikačního systému, který jsem vytvořil, může významně pomoci při jeho budování. Nabízí kvalitní a rychlý způsob, jak řídit firemní dokumentaci.

Přínos vidím také ve vytvoření znovupoužitelné knihovny pro webové uživatelské rozhraní. Knihovna nabízí vyšší úroveň abstrakce při vývoji rozhraní a srovnává rozdíly mezi různými internetovými prohlížeči. Vytvořil jsem také podporu pro asynchronní komunikaci pomocí AJAXu.

6.2 Další vývoj projektu

Při testování systému na síti jsem se setkal s problémy spojenými s oprávnění přístupu k aplikaci MS Word. Vzhledem k tomu, že celý publikační systém s textovým procesorem neustále spolupracuje, je správné nastavení práv naprosto nezbytné. Situaci bych v budoucnu řešil instalačním programem, který zajistí správná nastavení.

V 2.2.2 uvádím, že před schválením nové či modifikované verze řízeného dokumentu probíhá proces připomínkování. Publikační systém zatím nenabízí žádnou podporu pro dodatečné úpravy před samotným schválením. Jako rozšíření je tedy možné zavést vývojovou verzi dokumentu, která se nezobrazuje uživatelům a slouží pro autora pracujícího na nové verzi dokumentu.

Literatura

[1] Garrett, J. J.: *Ajax: A New Approach to Web Applications* [online]. February 18, 2005. URL: http://adaptivepath.com/publications/essays/archives/000385.php

[2] Hégaret, P., aj.: *Document Object Model (DOM)* [online]. January 19, 2005. URL: http://www.w3.org/DOM/

[3] Microsoft Corporation: *Visual Basic Reference* [online]. URL: http://msdn.microsoft.com/vbasic/reference/

[4] Mozilla Developer Center: *Javascript documentation* [online]. URL: http://developer.mozilla.org/en/docs/JavaScript

[5] Roman, S.: *Programujeme makra ve Wordu*. 1. vydání. Computer Press, Praha, 2000. 365 s. ISBN 80-7226-273-4

[6] Sysel, J.: *Koncepce managementu kvality* [online]. URL: http://www.cestovni-ruch.cz/hotelieri/iso9000d.php

Seznam příloh

Příloha 1. Obsah přiloženého CD

Součástí práce je CD, které obsahuje následující soubory:

- text bakalářské práce ve formátu PDF,
- text bakalářské práce ve formátu DOC,
- programovou dokumentaci,
- zdrojové soubory programu,
- odkaz na demonstrační stránky publikačního systému.