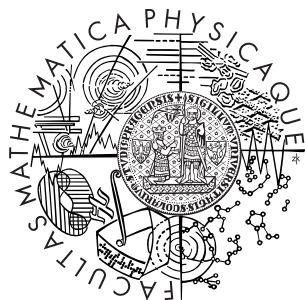


Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Jiří Kunčar

Informační systém pro jazykovou agenturu

Ústav formální a aplikované lingvistiky

Vedoucí bakalářské práce: RNDr. Miroslav Spousta

Studijní program: informatika, správa počítačových systémů

2009

Děkuji panu RNDr. Miroslavu Spoustovi za pomoc, připomínky, cenné rady a za odborné vedení bakalářské práce. Dále bych rád poděkoval firmě Primalingua s.r.o. za poskytnutí prostředků na vývoj aplikace. Speciální poděkování patří především paní majitelce Mgr. Miluši Psotové a paní RNDr. Jitce Kunčarové, která vývoj informačního systému iniciovala.

Prohlašuji, že jsem svou bakalářskou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce a jejím zveřejňováním.

V Praze dne 6.8.2009

Jiří Kunčar

Obsah

1	Existující implementace	10
2	Analýza úlohy	12
2.1	Požadavky klienta	12
2.2	Definice a upřesnění pojmů	15
3	Návrh řešení	17
3.1	Návrh databáze	17
4	Použité technologie a frameworky	22
4.1	Server	22
4.2	Klient	27
5	Programátorská dokumentace	29
5.1	Modely (<i>Models</i>)	29
5.2	Řadiče (<i>Controllers</i>)	33
5.3	Doplňky (Plugins)	36
6	Uživatelská dokumentace	37
6.1	Instalace serveru	37
6.2	Umístění aplikace	38
6.3	Inicializace databáze	39
6.4	Upřesňující informace	39
6.5	Správce (hlavní manažer)	40
6.6	Dodavatel	43

A	Ukázka XML databáze	47
A.1	DTD	47
A.2	XML	48
A.3	XPath	48
A.4	XQuery	48
B	Obsah přiloženého CD	49

Název práce: Informační systém pro jazykovou agenturu

Autor: Jiří Kunčar

Katedra (ústav): Ústav formální a aplikované lingvistiky

Vedoucí bakalářské práce: RNDr. Miroslav Spousta

e-mail vedoucího: Miroslav.Spousta@mff.cuni.cz

Abstrakt: Cílem práce je navrhnout a implementovat modulární informační systém pro firmu zabývající se výukou jazyků.

Součástí informačního systému bude zejména:

- modul pro nabídku a prodej kurzů pro veřejnost, individuální výuku, jazykovou výuku pro podniky, překlady a tlumočení
- modul produkty (výuka, překlady, tlumočení), výuka (individuální výuka, kurzy pro veřejnost, jazyková výuka pro podniky)
- modul pro News (hromadné rozesílání mailem), rozvrh výuky, třídní knihy, studijní materiály
- modul dodavatelé (smlouvy s lektory, překladateli, tlumočníky, měsíční výkazy odpracovaných hodin, měsíční přehledy nákladů na lektory, překladatele, tlumočníky)
- modul odběratelé (zápisy do kurzů pro veřejnost, objednávky, smlouvy, přílohy faktur za období od-do)
- modul pro testování znalostí (jednoduché testovací prostředí)
- modul pro správu IS (uživatelé, jejich práva, přehled změn).

Klíčová slova: informační systém, PHP, MySQL

Title: Information system of a language school

Author: Jiří Kunčar

Department: Institute of Formal and Applied Linguistics

Supervisor: RNDr. Miroslav Spousta

Supervisor's e-mail address: Miroslav.Spousta@mff.cuni.cz

Abstract: The goal of the thesis is to design and implement a modular

information system for a company involved in teaching of foreign languages.

The main parts of the information system will include, particularly:

- a module for offer and sale of courses to the public and individuals, language training for businesses, translation and interpretation
- a module of products (teaching, translation and interpretation), training (individual training courses for the public, language classes for businesses)
- a module for News (sending bulk mail), the schedule of teaching, class books, study material
- a module of suppliers (contracts with teachers, translators, interpreters, monthly class-sheets, monthly reports on the cost of staff translators and interpreters)
- a module of customers (registration in courses, orders, contracts, supplements to invoices for "from-to" periods)
- a module for testing of knowledge levels (a simple testing environment)
- a module for managing the IS (users, their rights and a summary of changes)

Keywords: information system, PHP, MySQL

Úvod

Informační systémy hrají v rychle se rozvíjejícím prostředí svou nepostradatelnou roli a Internet, který se stal dostupným pro širokou veřejnost, jim poskytl výbornou platformu umožňující další rozvoj v dříve nepříliš využívaném virtuálním prostředí. Umožňují snadné a rychlé zpracování, vyhledávání a publikování informací, které se díky němu stávají dostupné pro širší spektrum zákazníků, dodavatelů i samotných zaměstanců.

S využíváním Internetu zároveň rostou nároky na funkce, bezpečnost a dostupnost uložených informací, které jsou pro chod firem životně důležité a bývají jejich nejcenějším majetkem. Toto zvyšuje požadavky na robustnost řešení včetně nároků na nástroje použité při implementaci systému, kdy malá chyba může zapříčinit únik citlivých dat a jejich zneužití.

Před zaváděním informačního systému je nutné stanovit, čeho chce firma s jeho využitím v daném čase dosáhnout. Tomu by měla pomoci důkladná analýza firemních procesů a jejich optimalizace.

„Kdo nepozná vlastní firemní procesy, nemůže je zlepšovat.”[1]

Změna zavedených procesů bývá náročná, jak z finančního hlediska, tak z hlediska její vlastní realizace. , která se nemusí dotknout jen ...

Z původních jednoduchých aplikací na
Informačný systém, Client/Server.

Cíle práce

Cílem práce je návrh a implementace informačního systému usnadňující činnost a spolupráci více subjektů za použití vhodných opensource technologií a frameworků v použitých programovacích jazycích.

Cílem práce není vytvořit, vzhledem k náročným právním úpravám a častým změnám, komplexní účetní program, ale pouze systém umožnící vytvořit přehledy pro účetní evidenci, které mohou být zavedeny do některého již existujícího programu.

Obsah práce

- Analýza úlohy
- Požadavky
- Existující implementace - IS na míru vs. hotová řešení
- Návrh vlastní implementace
 - bezpečnost: Sanitize,
 - * SQL Injection, Cross Site Scripting
 - výkonnost (použití cache)
 - optimalizace GET/POST požadavků na stránku
- Programátorská dokumentace
- Uživatelská dokumentace

Poznámky k prepsání....

Ve druhé kapitole této práce je provedena analýza úlohy s ohledem na několik motivačních praktických příkladů. V této kapitole jsou mimo jiné stanoveny požadavky na řešení a je uveden přehled obdobných existujících implementací.

Ve třetí kapitole jsou stručně popsány technologie dále používané v této práci a je ukázáno jejich použití.

Čtvrtá kapitola popisuje návrh vlastní implementace s ohledem na požadavky stanovené v kapitole druhé. Jsou zde základní návrhová rozhodnutí a důvody pro volbu konkrétních řešení.

Pátá kapitola obsahuje programátorskou dokumentaci. Je zde popsáno technické řešení implementace a jsou nastíněny některé problémy, které bylo při implementaci nutné řešit.

Uživatelská dokumentace je obsažena v šesté kapitole. Popisuje uživatelské rozhraní vytvořených aplikací a ukazuje jejich použití na konkrétních příkladech.

V závěru jsou přehledně shrnuty výsledky, kterých bylo při vývoji systému dosaženo, a jsou naznačeny možnosti dalšího rozšíření.

Kapitola 1

Existující implementace

Existující aplikace lze rozdělit na několik skupin, kde ovšem žádná nepokrývá veškeré požadavky v plném rozsahu. První se více zaměřuje na správu webového obsahu. Označované anglickou zkratkou CMS¹ nebo WCMS² označující webové systémy. Tyto systémy lze rozdělit do podskupin podle způsobu, jakým prezentují uložená data.

Offline zpracování Server vygeneruje statické HTML před samotnou publikací, proto takové systémy nepotřebují, aby server aplikoval šablony na data při každém požadavku. Výhody jsou zjevné pro vytížené CMS používané převážně pro čtení dat. Naopak při častých změnách dat mohou být tyto systémy pomalé nebo neaktuální, proto je nutné si jejich nasazení řádně rozmyslet. Příkladem takového systému je například Vignette CMS³.

Online zpracování Server generuje prezentační data až na základě požadavků klienta. Pro keše ... Pokročilé systémy umožňují i úpravy základních šablon, bez zásahu do zdrojových kódů <http://tangocms.org/>
[joomla](http://joomla.org/) [drupal](http://drupal.org/) ...
<http://php.opensourcecms.com/>

Hybridní ...

¹wdw

²df

³<http://www.vignette.com/>

E-learning ... Moodle

ciele:

- * strategické (plánovanie investícií...) * taktické (vedenie, kontrola rozpočtu...) * operatívne (každodenná rutina)

Dôležité sú tiež úlohy IS:

- * manažérske (EIS - Executive IS) * taktické (DSS - Decision Support System) * vedenie (MIS - Management IS) * expertné (KWS - Knowledge Work System) * kancelárske (OIS - Office IS) * operatívne o TPS - transakčné (banky, ...) o CRM - vzťahy so zákazníkmi o RIS - rezervačné systémy o CAM - konštrukčné (CAD, ...) o GIS - geografické systémy

Kapitola 2

Analýza úlohy

Základním úkolem jazykové agentury je zajišťovat služby v oblasti jazykového vzdělání, překladů a tlumočení pro široké spektrum firemních i individuálních zákazníků.

Podniky posílají poptávky bez ohledu na nabídku konkrétních kurzů. Manažer vyřizuje zakázku elektronickou poštou, telefonicky nebo osobně. Všechny získané informace musí být schopen zaznamenat pro pozdější zajišťování vhodných dodavatelů .

Veřejnost posílá zápisové listy s ohledem na nabídku kurzů pro veřejnost. Manažer sestavuje skupiny podle zápisových listů a potvrzuje místo, termíny a cenu výuky. Současně probíhá zajišťování lektorů.

Dále jsou v kapitole popsány specifické požadavky na systém a návrhy jejich možného řešení.

2.1 Požadavky klienta

Na začátku stála žádost manažerky agentury zjednodušit proces zpracování měsíčních výkazů lektorů, překladatelů a tlumočnicků tvořících podklady pro mzdy a fakturaci služeb. Tyto měsíční výkazy neměly jednotný vzor a ani nebylo jednoduše možné, bez znalostí místních poměrů přiřadit vykázanou činnost k jednotlivým překladům, tlumočení a výuce¹.

¹Na překlady, tlumočení a výuku je jednotně nahlíženo jako na výsledek lidské práce dále označovanou jako *produkt*.

Zavedení jednotných identifikátorů

Kvůli výše uvedeným problémům se začalo s postupným zaváděním jednotných identifikátorů závazných pro všechny zúčastněné strany. Tento nepopulární krok byl ze začátku velmi těžce přijmán a trvalo několik měsíců, než se proces tvorby ustálil a začal být všemi akceptován.

- Z identifikátoru musí být zřejmé o jakou kategorii produktu se jedná, pro koho je určen a kdy byl zaveden, aby se předešlo kolizím jmen v budoucnu.
- V průběhu měsíce vkládá administrátor zadané překlady a nově zahájené kurzy.
- U produktu vloží poskytovatele, přiřadí zákazníka, studenty, lektora nebo překladatele a dále cenu za jednotku.
- U lektora či překladatele vloží jeho sazbu za jednotku.
- U kurzů vloží předpokládaný rozvrh, kde jsou uvedeny dny, čas, lektor a učebna. Tento rozvrh slouží zároveň jako pomůcka pro stanovení předběžných nákladů v budoucích měsících.

Měsíční výkazy

Po zavedení identifikátorů bylo potřeba sjednotit formuláře měsíčních výkazů a připravit dodavatele na možnost jejich elektronického vyplňování. Papírové formuláře budou do systému vkládány administrativním pracovníkem a elektronické budou potvrzeny a uzamčeny.

- Každý měsíc si administrátor může z výkazů odpracovaných jednotek lektora, překladatele a tlumočnicka zobrazit a vytisknout přehled, kde je souhrn toho, kolik jednotek opracoval v jednotlivých dnech a celkem u jednotlivých produktů.
- Systém musí zobrazit varování při překročení stanoveného počtu vykázaných odpracovaných hodin nebo přeložených normostran.

- Podle potřeby potřebuje administrátor rychle řešit suplování a změny lektorů, k tomu potřebuje aktuální stav. Příležitostně je třeba zjistit vytížení překladatelů.
- Na začátku týdne tiskne administrátor podle potřeby aktuální rozvrhy učeben a informuje zúčastněné strany o změnách. Časy individuální výuky jsou pohyblivé, lektori musí vědět, kdy je v učebnách volno.

Podklady pro fakturaci

Se vzrůstajícím objemem překladů a odučených hodin přestávalo být únosné ruční vytváření měsíčních přehledů pro zákazníky. Ti si přáli být informováni nejen o počtu odučených hodin, ale i o všech změnách zavedeném měsíčním rozvrhu a případném suplování. Zároveň musely být v systému zachyceny vazby mezi produktem, zákazníkem a vlastními účastníky tak, aby bylo možné stanovit výslednou cenu zakázky, která může být závislá nejen od počtu odučených hodin či přeložených stran, ale i od počtu účastníků.

- Před fakturací pro klienta si administrátor může sestavit seznam položek k fakturaci.
- Příloha za překlady a výuku se liší. U překladů není uvedeno jméno překladatele.

Správa lektorů a překladatelů

Práce agentury spočívá mimo jiné i ve shromažďování kontaktů na osoby zabývající se výukou, překlady a tlumočením. Musí být umožněno rychlé vyhledání a jednoduchá editace.

- V průběhu měsíce průběžně administrátor vkládá nové dodavatele tj. zájemce o spolupráci a aktualizuje změny u stávajících.
- Jedenkrát měsíčně odesílá administrátor minibuletín „News“ vybraným lektorům a překladatelům.

- „News” by měly být v systému přístupné pro lektory i překladatele, protože obsahují přílohy platné pro celý rok.

Oddělení správy dat více agentur

Vzhledem ke složitým poměrům v agentuře bylo potřeba oddělit evidenci zakázek vyřizovaných manažerem pro různé agentury a fyzické osoby, jenž s agenturami úzce spolupracují.

- Při vytváření produktu manažer zvolí poskytovatele služeb podle potřeb zákazníka.

Vícejazyková podpora

Systém je určen primárně pro jazykové agentury, kde se počítá s komunikací se zákazníky i v jiném než českém jazyce. K tomu bylo potřeba přizpůsobit systém již od počátku.

- Uživatelé by měli mít možnost si jednoduše vybrat mezi několika jazykovými variantami.

Úzká provázanost s webem

Na webových stránkách se zobrazuje aktuální rozvhy veřejných kurzů, jenž je i jinak dostupný na dveřích učeben. Ostatní rozvhy individuální výuky bude dostupný po přihlášení do systému.

2.2 Definice a upřesnění pojmů

Agentura

Informační systém

Informační systém (IS) je systém pro sběr, udržování, zpracování a poskytování informací a dat[2].

Produkt

Zakázka

CSS

Kapitola 3

Návrh řešení

3.1 Návrh databáze

Pro ukládání dat bude realizováno v relační databázi. Ta poskytuje funkce pro vytvoření, čtení, změnu a smazání uložených informací¹ a v tomto případě se jeví jako nejlepší ze způsobů ukládání dat.

V rámci testování byl rovněž vytvořen návrh zjednodušené verze XML² databáze [@todo priloha] uchovávané v souboru a zpracovávanho pomocí jazyka XPath, XQuery či XSLT [@todo popis]. Výsledky byly dobré při zpracování omezeného množství dat a jednodušivatel'ském přístupu k datům. Se vzrůstajícím objemem uložených dat začala projevovat prostorová náročnost použitého značkovacího jazyka a bylo nutné použít kompresi a rozdělení dat do více souborů. Druhou možnou variantou bylo použití některé specializované XML databáze nebo relační databáze s podporou XML. Toto řešení, ale nakonec nebylo implementováno ani v rámci testování. Protože ačkoliv jsou XML databáze v komerčních produktech na velmi dobré úrovni, jejich implementace ve volně dostupných aplikacích mírně zaostává.³

Návrh dobře strukturované relační databáze vyžaduje především pochopení vztahů a procesů fungování firmy, která ji bude využívat. Pro-

¹Někdy je používání zkratka *CRUD* z anglických slov *create*, *read*, *update*, *delete*.

²eXtensible Markup Language @todo

³@todo <http://www-01.ibm.com/software/data/db2/express/>

blémem s optimalizací navrženého modelu pomáhají řešit algoritmy logického návrhu relační databáze⁴, které se snaží zajistit, aby navržená databáze byla prostorově efektivní, eliminovala vznik anomálií a zároveň umožňovala rychlé zpracování uživatelských požadavků. Proces návrhu můžeme rozdělit na čtyři základní fáze[3]: analýza požadavků a specifikace, konceptuální návrh, logický návrh a fyzický návrh.

Analýza požadavků

Analýza požadavků vychází z informací popsanych na straně 12 a snaží se je co nepřesněji interpretovat pro použití v dalších fázích návrhu.

Konceptuální návrh

Produkty

Ne všechny specifické informace o daných produktech je potřeba udržovat jako speciální atributy. Zvítězila proto varianta jedné tabulky s minimem položek pro produkt. V budoucí rozšiřování by mělo být realizováno pomocí tabulek se specifickými informacemi.

- *Identifikátor* obsahuje informace o jazycích, typu produktu a stručný jednoznačný popis.
- *Kategorie* určuje zařazení produktu do skupin. Některé skupiny produktů mohou být veřejnosti online nabízeny na webových stránkách.
- *Poskytovatel* produktu je subjekt, který se stará o jeho realizaci. Hledá dodavatele i odběratele a jsou na něj vydávány faktury.
- *Časové informace* jsou používány dle potřeb pro určení doby konání kurzů, tlumočení nebo upřesnění požadavků na vyhotovení překladu.

⁴@todo vyjmenovat a nejake strucne popsat

- *Popis* je veřejná informace k produktu pro všechny zúčastněné strany.
- *Nastavení způsobu účtování* pro správné stanovení ceny produktu (započítat množství, započítat počet účastníků na přihlášce).
- *Jednotka a jednotková cena*

Smlouvy

V návrhu se počítá s kontrolou maximálního počtu odpracovaných jednotek. Některé možnosti nastavení jsou pevně zakomponované a půjde je změnit pouze zásahem do zdrojových kódů nebo databáze.

- *Dodavatel* je smluvním partnerem jazykové agentury - *poskytovatele* jazykových služeb
- *Maximální počet jednotek*, které může na smlouvu vykonat.
- *Sazba, jednotka, způsob platby a typ smlouvy* umožňují generování účetních přehledů.

Výkazy

Tabulka výkazů se nedrží papírové podoby, která je pro ukládání v databázi nevhodná z důvodu malé hustoty záznamů (obvykle 5-10 záznamů za měsíc na jeden produkt). Jednotlivé záznamy budou ukládány v oddělené tabulce s dalšími upřesňujícími informacemi.

Z požadavků rovněž vyplívá potřeba uzamykání výkazu k danému datu. Jelikož se jedná o velmi důležitá data, měla by jejich kontrola prováděna i na databázové úrovni pomocí triggeru, pokud to bude daná databáze bude podporovat.

- *Produkt*, ke kterému se výkaz vztahuje.
- *Sazba za jednotku a jednotka*, jenž náleží dodavateli, jako odměna.
- *Smlouva* na níž je prováděna daná činnost.

- *Datum* před kterým již není možné přidávat, editovat nebo mazat odpracované jednotky.

Události

Seznam událostí je přiřazen k výkazu a každá událost je doplněna o upřesňující informace.

- *Výkaz*, ke kterému se událost vztahuje.
- *Počet jednotek* typu uvedeného v přiřazeném výkazu.
- *Datum* a *čas*, kdy byla práce vykonávána. Slouží rovněž jako rozvrh hodin.
- *Místo* vybrané z číselníku.

Příhlášky (objednávky)

Tabulka slouží k evidenci zákazníků a produktů, jenž si objednali.

- *Produkt*, ke kterému se přihláška vztahuje.
- *Zákazník* vybraný z tabulky uživatelů.

Účastníci

Účastníci jsou přiřazeni k danému produktu, přes přihlášku, kde je definován plátce.

- *Přihláška*
- *Účastník* vybraný z tabulky uživatelů.

Uživatelé

O všech uživatelích systému je potřeba vést obdobné informace. Proto jsou uloženi v jedné tabulce a o jejich specifikaci je rozhodnuto až přiřazením do skupiny. Každý uživatel může být ve více skupinách.

- *Osobní údaje* - jméno, příjmení, titul a zobrazované jméno
- *Přihlašovací údaje* - přihlašovací jméno, heslo a stav účtu
- *Doručovací a fakturační adresa*
- Ostatní nečleněné informace

Číselníky

Byly navrženy zjednodušení práce s vyplňováním formulářů a odstranění duplicit v tabulkách.

- *Místa*
- *Kategorie*
- *Jednotky*

Uvedené informace jsou zpracovány do zjednodušeného ER diagramu v příloze [@todo zjednodušený ER]

Logický návrh

@todo

Fyzický návrh

@todo

@notes V uvedeném návrhu je potřeba zajistit integritu dat a optimalizace pro rychlejší přístup k datům. K tomu nám

Oracle XML DB <http://www.oracle.com/technology/tech/xml/xmldb/index.html>

Evidence produktů

Jedna z hlavních výhod XML, která ovšem není v tomto případě využívána, je snadná výměna dat mezi aplikacemi.

Návrh dobře strukturované relační databáze je nejnáročnějším prvkem celé práce. Vyžaduje jak

Podklady pro fakturaci dodavatelům a mzdy

Finanční vyrovnání subjektů v IS

Kapitola 4

Použité technologie a frameworky

Použité technologie byly částečně determinovány požadavy klienta^{2.1}. Tím, že se mělo jednat o systém využívající výhradně open-souce technologie tak, aby nebyly zvyšovány náklady na nákup licencí.

4.1 Server

Tento webový informační systém využívá technologii PHP (jazyk, interpret a knihovny), která vychází se skriptovacího víceúčelového jazyka, jenž byl původně vyvinut pro tvorbu dynamických webových stránek. Z tohoto využití vznikla i zkratka z anglických slov *Personal Home Page*, které byly nahrazeny slovy *PHP: Hypertext Preprocessor* dající vznik rekurzivní zkratce¹.

Výhodou použití PHP je existence interpretu pro různé operační systémy a podobnost jeho syntaxe s C, Javou.

Nevýhodou, která brzdí dalšímu rozvoji a rozšíření, je absence normy (k datu vydání BP). Jazyk je tak de facto standardizovaný interpretem a množstvím lidí², kteří jej využívají. I když existují mnohé polemiky a

¹http://cs.wikipedia.org/wiki/Rekurzivní_zkratka

²dle statistiky na <http://www.php.net/usage.php>

Tabulka 4.1: Srovnání vybraných PHP frameworků

živé diskuze mezi jeho zastánci a odpůrci o jeho výkonnosti, bezpečnosti a vhodnosti pro velké projekty, existují výjimky³, které tyto názory vyvrací a zároveň se podílejí na vývoji, a tak se snaží přispět k jeho větší výkonnosti a bezpečnosti.

Vývoj jazyka sebou nese i stinné stránky. Po létech vývoje dochází k úpravám API⁴ některých vestavěných funkcí a změna syntaxe. To může zapříčinit, že po aktualizaci interpretu jazyka, přestanou fungovat některé části nebo celá aplikace. Řešením ovšem není zůstat na několik let staré verzi, ve které mohly být objeveny chyby.

Nespornou výhodou používání frameworků spočívá jednodušším vývoji aplikací a minimalizací rizika chyb v jinak ručně psaném jádru aplikace. Toto je zajištěno pouze pokud má kvalitní a úplnou dokumentaci a je zastřešen silnou komunitou nebo společností zajišťující jeho vývoj. Některé frameworky rovněž dokáží do jisté míry zakrýt rozdíly mezi verzích. Děje se tak, za cenu zpomalení některých částí aplikace, díky kontrole verze překladače a vykonáním alternativního kódu.

Při výběru frameworku byl kladen důraz hlavně na kvalitní dokumentaci, rozšiřitelnost a možnost práce s různými relačními databázemi. Frameworky, které splňují většinu požadavků jsou Zend⁵, Symfony a CakePHP.

- Symfony - používá yml na konfigurování

CakePHP

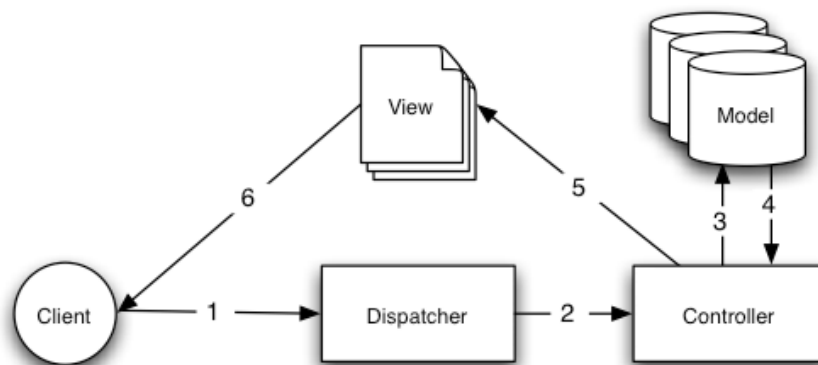
CakePHP je open source[[@todo](#)], rapid development[[@todo](#)] framework[4], který dává příležitost věnovat se návrhu schématu a logiky aplikace.

MVC *Model-View-Controller* (Model-Pohled-Řadič) - softwarová architektura oddělující data, uživatelské rozhraní a logiku aplikace[6].

³Facebook, YouTube, Wikipedia (MediaWiki) a další

⁴anglická zkratka *Application Programming Interface*, označuje sadu funkcí, procedur či tříd programu či knihovny, jenž mohou být využívány programátorem

⁵<http://framework.zend.com/>



Obrázek 4.1: Diagram MVC dotazu[6]

Výsledný kód aplikace se stává přehlednějším a umožňuje změnu libovolné komponenty s minimálními dopady na ostatní části aplikace (např. změna databáze, výstup v XML či jiném formátu, ovládání přes webové rozhraní nebo konzoli).

Obrázek 4.1 zobrazuje základní schéma procesu zpracování požadavku klienta na URL⁶. Klient pošle (1) požadavek na URL, které je plánovačem (*Dispatcher*) zkontrolováno a předáno (2) správnému přepínači (*Controller*). Ten se na základě parametrů, které obdrží od plánovače, rozhodne o spuštění správné akce a předání parametrů. V plánovači je obsažena vlastní aplikační logika (včetně např. kontroly přihlášení uživatele), která může využívat modely pro přístup k uloženým datům a jejich modifikaci (3 a 4). Až plánovač získá všechna potřebná data předá (5) je zbývajících vrstev - pohledu (*View*). V té jsou data zformátována do požadované podoby a poslány (6) klientovi.

ORM *Object-relational mapping* (objektově relační mapování) - programátorská technika určená pro konverzi dat mezi nekompatibilními systémy určenými pro ukládání dat a objektově orientovanými

⁶ *Uniform Resource Locator*[7] (jednotný lokátor zdrojů) popisuje sémantiku a syntaxi řetězce znaků sloužící k přesné specifikaci zdroje informací v prostředí Internetu.

jazyky[8]. Největší výhoda tedy spočívá v odstínění způsobu práce s odlišnými zdroji dat. Ať už se jde o různé relační databáze (MySQL, PostgreSQL, Oracle, MSSQL) nebo dokonce vlastní definované zdroje dat. Může se jednat formátované soubory např. CSV[@todo], XML nebo za pomoci API ke službám jako LDAP, Twitter, IMAP a další. V CakePHP verze 1.2 jsou data z tabulky mapovány na dvojrozměrné pole.

Mezi další výhody, které CakePHP nabízí, patří SEO optimalizace stránek, podpora i18n⁷ i l10n⁸ pro tvorbu vícejazyčných aplikací. Kód je již zapracován do jádra frameworku a otestován komunitou, díky které není potřeba psát již jednou napsané části, ale možné šetřit lidské zdroje na vlastní vývoj aplikační logiky.

CakePHP zastává programátorskou filosofii *DRY* - Don't Repeat Yourself. Ta zdůrazňuje, že jednotlivé části kódu by se v programu neměly opakovat, jelikož se snižuje srozumitelnost kódu a při modifikaci se musí upravovat stejný kód na více místech, což často vede k nesnadno odhalitelným chybám programu. K tomuto účelu se používají „Pomocníci“ (*Helpers*), kteří daný kód generují. Při potřebě změny je lze snadno upravit, bez potřeby zásahu do míst odkud jsou voláni.

Za největší přínos samotného frameworku v počátku návrhu aplikace považují „pekárnu“ kódu (*baker*) a podporu „lešení“ (*scaffolding*), které umožňuje pracovat s automaticky generovaným rozhraním podporujícím CRUD. Pekárna umí vygenerovat modely i s validačními kritérii, prezentační vrstvu a řadiče. Navíc sama najde relace mezi tabulkami a přidá tuto informaci do odpovídajícího modelu.

Hledání relací probíhá na základě názvů tabulek a jejich atributů. Proto je doporučeno při práci s CakePHP dodržovat tyto konvence, které nejsou nijak závazné a jdou změnit ve specifikaci každého modelu.

- názvy tabulek anglicky v množném čísle

⁷„Příprava aplikace na podporu různých kulturních zvyklostí.”[9]

⁸Jedná se o „doplnění aplikace o data specifická pro konkrétní národní/regionální prostředí”[10]

- cizí klíče stejně jako tabulka v jednotném čísle a končí na „_id”
- primární klíč má název „id”

Díky těmto úsporám je možné se zaměřit na ergonomii dané aplikace a její možnou optimalizaci, která je ovšem limitována výkonností použitého skriptovacího jazyka. Tato omezení lze, minimalizovat kešováním⁹ částí zpracovaného zdrojového kódu v paměti nebo jeho předkompilací [APC, Xcache, File].

Další možnou optimalizací, která sice přímo nesouvisí s CakePHP, ale je v něm snadno implementovatelná, je snížení počtu požadavků na stránku. S používáním javascriptových frameworků a knihoven se snadno může stát, že počet vkládaných odkazů na skripty a kaskádové styly (CSS) se vyšplhá až k desítkám a začne se neúměrně prodlužovat doba potřebná na stažení všech potřebných částí. Zvyšuje se tak počet požadavků na server a díky režii protokolu HTTP¹⁰ je ve výsledku stažen větší objem dat. Tento problém úspěšně řeší projekty *jsMin*¹¹ a *CSSTidy*¹².

MySQL

MySQL je multiplatformní systém pro řízení báze dat (DBMS, zkráceně databázový systém), který poskytuje všechny potřebné vlastnosti nutné pro běh tohoto informačního systému:

- cizí klíče (foreign-keys)
- poddotazy (subselect)
- pohledy (views)
- transakce
- trigger

⁹z anglického slova *cache*, označuje vyrovnávací paměť

¹⁰@todo HTTP Transfer Protokol - textově orientovaný

¹¹<http://code.google.com/p/jsmin-php/>

¹²<http://csstidy.sourceforge.net/>

- fultextové vyhledávání

Systém byl původně vyvinut Michaellem Wideniusem a Davidem Axmarkem v roce 1994, kteří se podíleli na založení firmy MySQL AB. Ta byla až do roku 2008, kdy byla provedena akvizice s firmou Sun Microsystems, jednou z největších open source společností na světě. MySQL je nabízeno jak pod bezplatnou licenci, tak pod komerční licenci s plnou technickou podporou[15]. Těší velké oblibě hlavně mezi vývojáři webových aplikací, kde je součástí platformy označované jako *LAMP*, která obsahuje *Linux*, *Apache*, *MySQL* a *PHP*. Obecně jednotlivé části mohou být zaměněny (např. PHP za Perl nebo Python, MySQL za Postgres) při zachování zkratky, která je zaužívaným označením pro operační systém, webový server, databázový systém a skriptovací jazyk sloužící k obsluze a generování webových stránek.

4.2 Klient

Pro vlastní běh aplikační logiky je možné se spolehnout, že serverová část aplikace bude zpracována jednou verzí PHP interpretu v uzavřeném a otestovaném prostředí. Naproti tomu klientská část bude prezentována na odlišných operačních systémech v mnoha prohlížečích nejrůznějších verzích. Základními požadavky kladené na prohlížeč jsou:

- XHTML 1.0¹³
- CSS 2.1¹⁴
- JavaScript¹⁵

S XHTML a CSS prohlížeče problémy nemívají. Horší je to, ale s implementacemi Javascriptu. Toto bylo vyřešeno díky provázanosti CakePHP

¹³<http://www.w3.org/TR/xhtml1/>

¹⁴<http://www.w3.org/TR/CSS2/>

¹⁵JavaScript je dialektem ECMAScriptu. Minimálním implementovaným standardem by měl být ECMA-262, revize 3:<http://www.ecma-international.org/publications/standards/Ecma-262.htm>.

a javascriptového frameworku Prototype[[@todo odkaz](#)] a jeho rozšíření Script.aculo.us[[@todo odkaz](#)], který se snaží zakrýt rozdíly mezi prohlížeči.

Prototype není jediným či nejlepším frameworkem. Existují i jiné, které mají rozsáhlejší schopnosti v oblasti používání dotazovacího jazyka XPath nebo tvorby GUI¹⁶.

Prototype a jeho rozšíření

Prototype umožňuje

- JS - Prototype, Script.aculo.us, Livepipe

¹⁶z anglických slov *Graphical User Interface*, uživatelské prostředí, jenž umožňuje uživateli ovládat aplikaci pomocí grafických ovládacích prvků

Kapitola 5

Programátorská dokumentace

V následující kapitole je popsáno a vysvětleno propojení databázového schématu s třídami, základní adresářová struktura projektu a metody použitých tříd.

5.1 Modely (*Models*)

Všechny modely v aplikaci jsou odvozeny od třídy *Object*, která zakryta třídou *Overloadable*. Ta navržena tak, aby zakryla rozdíly mezi PHP verze 4 a 5.

Modely jsou determinovány návrhem databáze. Relace mezi jednotlivými modely odpovídají relacím v databázi a je možné je upravovat ve vlastnostech dané třídy. Existují čtyři druhy relací, jenž jdou v CakePHP modelovat, a mají své vlastní nastavení. Relace 1:1 (*hasOne*), 1:N (*hasMany*) a N:1 (*belongsToMany*) mají většinu nastavení společnou:

className jméno třídy asociovaného modelu

foreignKey jméno cizího klíče v asociovaném modelu

fields seznam polí, jenž mají být načteny při použití dané asociace

order část SQL dotazu sloužící k definici pořadí výsledků

conditions část SQL dotazu sloužící k filtraci nalezených výsledků

Při modelování relace M:N (*hasAndBelongsToMany* - *HABTM*) je potřeba nadefinovat tabulku (*joinTable*), přes kterou bude prováděno spojení nalezených dat, a cizí klíč do asociované tabulky (*associationForeignKey*).

Pro získání dat se používá metoda *find(\$typ, \$parametry)* na příslušném modelu. První parametr určuje typ dotazu, který může být jedním z následujících:

all najde všechny data splňující podmínky definované ve druhém parametru

first najde první záznam vyhovující podmínkám (liší se od předcházejícího ve struktuře vráceného pole)

count vrátí počet vyhovujících záznamů

list vygeneruje pole s hodnotami ze sloupce tabulky definované v proměnné *\$displayField* daného modelu indexované standardně primárním klíčem

threaded vygeneruje vnořené pole podle zadaného intervalu (*lft* - *rght*)¹

neighbors vrátí položku před (*prev*) a za (*next*) položkou, která by byla nalezena pomocí *find('first', \$podminky)*

Pole s vyhledávacími podmínkami může vypadat jako v následujícím příkladu, kde jsou uvedeny všechny základní možnosti. Další parametry mohou být přidávány pro doplňkové rozšíření chování jednotlivých modelů.

```
array(  
    'conditions' => array('Model.field' => $thisValue),  
    //vygeneruje klauzuli WHERE Model.field = $thisValue  
    'recursive' => 1, //int  
    'fields' => array('Model.field1'),  
    //sloupečky v dotazu SELECT
```

¹Detailní popis příkladu, jak uchovávat stromové struktury v relační můžete nalézt na <http://dev.mysql.com/tech-resources/articles/hierarchical-data.html>.

```

'order' => array('Model.created', 'Model.field3 DESC'),
//definice klauzule ORDER BY
'group' => array('Model.field'),
//sloupečky použité v klauzuli GROUP BY
'limit' => n,
//počet záznamů na stránku
'page' => i, //najdi i-tou stranu
'callbacks' => true
//možné hodnoty jsou: false, 'before', 'after'
)

```

Zajímavým parametrem je *recursive*, který určuje do jaké hloubky mají být načítána data. Pokud data z asociovaných modelů nejsou potřeba, je vhodné jej nastavit na *-1*.

V systému, kde je potřeba vyhledávat sdružovat data podle modelů, jenž jsou v řetězci asociací dále, bylo výchozí chování parametru nedostatečné. Naštěstí existují doplňky, které takové činnosti umožňují.

Rozšíření chování (*Behaviors*)

V základní výbavě CakePHP je obsaženo rozšíření *Containable*, které umožňuje dynamicky přidávat asociované modely bez omezení hloubky rekurze². Je to ale pouze syntaktický cukr k metodám *bindModel()* a *unbindModel()*, které umožňují definovat pouze asociace příslušící danému modelu. *Containable* přidává metodu *contain()*, jenž je volána i pokud je nalezen index „contain” v poli s vyhledávacími parametry metody *find()*. Ta se postará o dynamickou asociaci potřebných modelů.

Také výše zmíněné rozšíření má určité nedostatky a to hlavně pokud je potřeba filtrovat data podle modelů asociovaných ve větší hloubce. Proto je použité rozšíření „*Linkable*”[13], které má obdobnou syntaxi nastavení, ale vytváří jeden SQL dotaz za pomoci klauzulí JOIN.

Pro usnadnění práce s relacemi typu M:N přes tabulku s klíči je použito rozšíření „*Extend Associations*”[14]. To umožňuje jednoduché při-

²Je ovšem potřeba vzít na vědomí, že CakePHP tak vygeneruje značné množství dotazů, které mohou mít vliv na výkon dané aplikace.

dání nebo smazání asociací, bez ovlivnění ostatních již existujících. K tomu slouží nově definovaná sada metod:

habtmAdd(\$model, \$assoc, \$id, \$assoc_ids) přidá asociace k danému záznamu v modelu

habtmDelete(\$model, \$assoc, \$id, \$assoc_ids) smaže asociace s danou kombinací klíčů

habtmDeleteAll(\$model, \$assoc, \$id) smaže všechny asociace k danému záznamu modelu

A pro úplnost jsou zde uvedeny zbývající rozšíření.

Slugable Vytváří texty použitelné v URL z definovaných sloupců tabulky.

Versionable Ukládá aktuální verzi záznamu před změnou či smazáním.

Logable Zaznamenává akce uživatelů.

Translate @todo

Úpravy základních metod *AppModel*

Jelikož je kontrola konzistence dat po přidání, smazání a editaci záznamu ponechána na databázi, bylo potřeba zajistit zpracování chybových hlášení z databáze. To se děje v metodě *AppModel::onError()*, která je volána pokaždé, když metoda *DboSource::execute()* vrátí chybu. V našem případě je chyba zpracována a chybová hláška je uložena pro pozdější prezentaci uživateli.

Pro plnou lokalizaci aplikace bylo potřeba rozšířit metodu *AppModel::invalidate(\$záznam, \$chybovéHlášení)* o překlad chybového hlášení, pokud daný záznam neprošel validací. Lokalizaci lze rovněž provádět při generování pohledu, ale to se z hlediska údržby kódu nejeví jako nejvhodnější řešení.

5.2 Řadiče (*Controllers*)

Řadiče jsou používány ke správě aplikační logiky. Pro možnost používat stejný kus kódu v různých řadičích je vhodné používat komponenty. Tyto komponenty usnadňují autentifikaci, autorizaci uživatelů, vyhledávání v databázi, posílání emailů nebo validaci uživatelských formulářů.

Komponenty

Auth/Authmd5 Standardní knihovna přidává k heslu řetězec definovaný v `app/config/core.php` v interní proměnné `Security.salt`. Tato vlastnost byla v systému na obtíž, hlavně kvůli přenositelnosti již zadaných hesel v předchozí aplikaci. Proto byla vytvořena komponenta `Authmd5`, která nepřidává zabezpečovací řetězec k heslu. Další drobnou změnou oproti původní komponentě je automatické hashování hesla z formuláře i když není zadáno přihlašovací jméno.

P28n Při načtení komponenty se zkontroluje zda si již uživatel někdy vybral jazyk, který byl uložen do cookie pomocí metody `change($lang)`. Pokud se jazyk v cookie nenachází zvolí se výchozí jazyk aplikace podle proměnné `$_SERVER['HTTP_ACCEPT_LANGUAGE']`, pokud je prohlížečem nastavena. V opačném případě je použit jazyk definovaný v konstantě `DEFAULT_LANGUAGE` v souboru `app/config/core.php`.^[11]

Password Helper Vygeneruje „hezké“ heslo zadané délky bez opakujících se znaků. Jednotlivé znaky jsou vybírány z řetězce `PasswordHelper::possible`.^[12]

Autocomplete Přidává metodu `autocomplete()`, která zpracuje získaná data z formuláře (`data[Model][položka]`), vyhledá vyhovující záznamy a vygeneruje seznam s výsledky. Přidání doplňkových podmínek pro vyhledávání je možné v metodě `beforeFilter()` daného řadiče nebo jeho předka. Děje se tak nastavením vlastnosti `AutocompleteComponent::handles`, kde index v zadaném poli určuje položky tabulky, kterých se mají podmínky týkat. Následující příklad

je vybrán z řadiče produktů, kde má být zákazníkům umožněno vyhledávat pouze v produktech, které se jich přímo týkají.

```
function beforeFilter() {
  parent::beforeFilter();
  $this->Autocomplete->handles = false;
  // standardně je doporučeno vypnutí všech políček
  if($this->Auth->user()) {
    if (isset($this->params['prefix'])) {
      if ($this->params['prefix'] === 'admin') {
        // podle prefixu můžeme omezit práva
        $this->Autocomplete->handles = array('Product.*');
      }
      if ($this->params['prefix'] === 'customers') {
        $this->Product->Behaviors->attach('Linkable');
        // dynamické načtení chování
        $this->Autocomplete->handles = array(
          'Product.name' => array(
            'link' => array('Application'),
            'conditions' => array(
              'Application.user_id' => User::get('id'))
          );
        // zákazníci mohou vyhledávat pouze v produktech,
        // kterých se účastní
      );
    }
  }
}
```

Ajaxupdate Slouží pro usnadnění editace záznamů přes AJAX za podpory javascriptivé knihovny *control.js*, která je součástí balíku *scriptaculous.js*.

Email Slouží k odesílání emailů. Lze používat šablony, html zprávy a přidávat přílohy.

Filter Komponenta pro validaci vyhledávacích formulářů a vygenerování podmínek pro vyhledávání.

Popis jednotlivých řadičů

V této části budou popsány nejdůležitější části aplikační logiky. Při vytváření základní kostry byla použita „pekárna“ kódu, jejíž vygenerovaný kód byl posléze upraven.

Uživatelé

Nejdůležitějšími akcemi jsou bezpochyby přihlášení a odhlášení uživatele. Ty jsou zajitěny komponentou *Auth*, která je nastavena v řadiči aplikace (*AppController*) v metodě *beforeFilter()*.

login() Po přihlášení je aktualizována položka „poslední přihlášení“ v databázi, nastavena zpráva uživateli a uživatel je přesměrován na požadovanou nebo výchozí stránku.

logout() Smaže cookie, nastaví zprávu o úspěšném odhlášení a přesměruje na výchozí stránku.

account() Slouží ke změně hesla uživatele. Pokud jsou poslány data a hesla projdou validací, je uživateli nastaveno nové heslo.

recover() Uživatel si může nechat zaslat nové heslo po zadání emailu. Je vytvořen token s platností 1 den. V budoucnu by bylo dobré rozšířit možnost obnovy hesla o kontrolní otázku.

verify(\$token) Zkontroluje zda se zadaný token nachází v databázi a pokud ano, zašle uživateli nově vygenerované heslo.

register() Jednoduchá registrace uživatele po jejímž dokončení je uživateli zaslán aktivační email.

Produkty

admin_export() Vygeneruje CSV soubor s nalezenými produkty.

admin_shedule()

5.3 Doplnky (Pluginy)

Vyhledávací modul

Nejdůležitějším požadavkem na tento modul je rychlost. Jelikož byla vybrán databázový systém MySQL, který ve verzi 5.2 umožňuje fulltextové vyhledávání pouze nad tabulkami typu MyISAM, bylo potřeba zajistit, že půjde rychle vyhledávat data obsažená tabulkách typu InnoDB.

Kapitola 6

Uživatelská dokumentace

Tato uživatelská dokumentace si klade za cíl stručně přiblížit čtenáři výhody informačního systému a jeho základními způsoby ovládání. V jednotlivých částech je vysvětleno, jak má správce postupovat od úvodní instalace, přes inicializaci databáze, uložení informací o uživateli, přidání produktů až po tisk účetních podkladů.

Dodavatelům a zaměstnancům je názorně předvedeno, jak správně a včas vyplnit měsíční výkazy a jak vést evidenci docházky účastníků kurzů.

Zákazníkům je vysvětleno, jak jednoduše zjistit, kolik z objednaných služeb již bylo zaplacen a zkontrolovat průběh aktuálních kurzů či počet přeložených stran překladu.

Studenti si mohou zkontrolovat svojí docházku a zobrazit rozvrh hodin na další týdny.

6.1 Instalace serveru

Pro běh serverové aplikace je nutné mít sprovozněný program, který umožňuje zpracování zdrojových kódů a prezentaci výstupu protokolem HTTP popřípadě HTTPS. Nejznámějším volně dostupným programem je Apache [<http://httpd.apache.org/>], který umožňuje pomocí modulů [<http://httpd.apache.org/modules/>] přidat podporu pro jazyk PHP

nutný k běhu IS.

Stažení a instalace Apache

Pokud používáte některou z moderních linuxových distribucí, zkuste nejdříve projít repozitáře [@todo vysvětlit], zda se zde nenachází již hotový balík upravený pro snadnější instalaci a konfiguraci.

Debian a jemu podobní (Ubuntu, Kubuntu, ...):

```
apt-get install apache2
```

Gentoo

```
emerge apache2
```

Pokud jste požadovaný balík nenašli či používáte jiný operační systém, můžete ze stránek projektu¹ vybrat odkaz vedoucí na požadovaný balík zdrojových kódů nebo předkompilovanou aplikaci pro Váš operační systém.

Konfigurace Apache

Pro správnou funkci aplikace je potřeba doinstalovat, popřípadě pouze povolit následující moduly: *mod_php5*, *mod_rewrite* a *mod_ssl*. Pro správnou funkci zabezpečeného připojení [@todo definovat zabezpečené připojení] je nutné vygenerovat certifikáty a upravit konfiguraci stránek.

Stažení a instalace MySQL serveru

uživatelské jméno: isadb

heslo: isapass

6.2 Umístění aplikace

Pokud máme server správně nakonfigurovaný, zkopírujeme složku s aplikací do adresáře určeného konfigurací Apache (obvykle */var/www* či *C:\\Program Files\\apache2\\www* - budeme označovat *ROOT*). Pro větší

¹<http://httpd.apache.org/download.cgi>

bezpečnost je doporučeno změnit „webroot” v konfiguraci apache na složku *ROOT/app/webroot*. Dále je potřeba nastavit přihlašovací údaje k databázi podle 6.1 předchozí části v souboru *ROOT/app/config/database.php*.

6.3 Inicializace databáze

V souboru *ROOT/app/config/sql/isa_init.sql* se nachází MySQL 5.0+ kompatibilní skript, který vytvoří tabulky a naplní je daty nutnými k prvnímu přihlášení administrátora.

6.4 Upřesňující informace

Pro další čtení manuálu je potřeba upřesnit několik důležitých pojmů, které se budou dále vyskytovat.

Systémová skupina je nutná pro správné fungování IS.

Tato sekce je rozdělena podle rolí definovaných v IS.

Správce: pověřený uživatel s plnými právy ke všem modulům systému.

Editor: osoba s omezenými právy k editaci vybraných modulů.




Dodavatelé: zaměstnanec, brigádník či jiný subjekt vykonávající zadanou práci.

Odběratelé: zákazníci, kteří si objednali libovolný produkt.

Poskytovatelé: subjekty, které jsou vedeny „pod jednou střechou” a sdílí část informací.

Účastníci: studenti jednotlivých kurzu.

Výše popsané role mohou být změněny či zakázány administrátorem systému. Čtenáři je doporučeno číst pouze části, jenž se ho týkají.

Poznámka: Text u odkazu na zobrazení, editaci nebo smazání záznamu mohou být nahrazeny po řadě obrázky: ,  a .

6.5 Správce (hlavní manažer)

Má standardně nastavena veškerá přístupová práva ke všem modulům systému.

Správa IS

Tento modul obsahuje moduly pro nastavení jednotlivých částí systému, správu uživatelů, skupin, práv a ostatních číselníků (daně, místnosti, kategorie produktů).

Uživatelé (Users)

Modul UŽIVATELÉ nabízí veškeré nastavení potřebné pro definování možností při užívání systému osobami majícími vztah k IS. Dále jsou zde uchovány veškeré osobní informace důvěrného charakteru.

Filtr seznam uživatelů [/admin/Users/index]: V horní části se nachází odkazy na akce související s uživateli. Filtry pro práci se seznamem uživatelů jsou umístěny nad hlavní tabulkou a v hlavičce tabulky s funkcí „našeptávače” a jejich zm

Přidání uživatele [/admin/Users/add]: Odkaz na formulář je hlavním menu a v částech, které na uživateli závisí. Jedinou povinnou položkou je *zobrazované jméno*, která slouží jako popis ve všech výběrových filtrech. Ostatní položky můžete vyplnit až při jejich potřebě ve výpisu. Volba *Aktivní* slouží k aktivaci uživatelského účtu. Uživatel je schopen se přihlásit pouze pokud je jeho účet aktivní.

Editace uživatele: Uživatele vyhledejte přes filtr seznamů uživatelů nebo přes formulář rychlého hledání. V prním případě klikněte na Vyberte odpovídající skupinu k editaci a pro uložení změn použijte tlačítko *Uložit*. V případě chybového hlášení zkontrolujte všechny skupiny údajů.

Zobrazení uživatele: @todo

Nastavení či změna hesla: V přehledu uživatele klikněte na odkaz Změnit heslo a vyplňte nové heslo do obou kolonek. Správce může editovat hesla všem uživatelům a měl by je o této změně informovat zabezpečeným kanálem[@todo definovat zabezpeceny kanal], aby se předešlo zneužití jejich účtu.

Skupiny a oprávnění

V tomto modulu můžete přidávat a mazat uživatelské skupiny a oprávnění. Pro zachování správného fungování systému nemažte tyto systémové skupiny: customers, employees, providers, students a admin.

Při implicitní nastavení má každá ze systémových skupin práva definované a přiřazené oprávnění k prefixovaným akcím jednotlivých modulů. Tyto oprávnění jsou definovány tímto způsobem: **:customers_**, **:employees_**, **:providers_**, **:students_**, **:admin_** a říkají, že daná skupina může v libovolném modulu spouštět akce začínající jejich jménem. V příkladu byl použit expanzní znak „*”, který je možné používat k nahrazení libovoného počtu libovolných znaků, a „:”, jenž určuje hranici mezi modulem a akcí (např. oprávnění *Users:*_view* umožní zobrazit všechny prefixované akce *view* v modulu UŽIVATELÉ).

Každá skupina může mít přiřazeno více definovaných oprávnění a je na správci, jak s nimi bude zacházet.

Číselníky

V této části se seznámíme se všemi číselníky, jejich funkcemi v systému a možnostmi editace. Všechny dále uvedené moduly obsahují automaticky generovaný číselný identifikátor (dále jen *id*), který slouží pro interní potřeby systému a není možné ho změnit. Dále je potřeba upozornit na fakt, že vytvořené položky v číselníku, které již byly v systému přiřazeny nějakým záznamům, nelze vymazat. Vymazání je umožněno až když je číselník u daných záznamů změněn.

Kategorie obsahuje název, zkratku, jednoduchý slovní popis a zaškrťovací políčko určující, zda se produkty v této kategorii považují za veřejné a má se zobrazovat jejich rozvňh na webových stránkách. Účel číselníku spočívá v rozčlenění množství produktů do skupin, podle kterých lze vytvářet tiskové sestavy (přílohy faktur, rozvrhy, měsíční přehledy nákladů, ...).

Místa obsahují název a adresu. Slouží k odkazu na místo v událostech.

Štítky obsahují název a odkaz na nadřazený štítek, pro možnost tvorby hierarchické struktury článků.

Produkty

Modul PRODUKTY je závislý na správně rozdělených uživatelič do systémových skupin. Rovněž je doporučeno předvyplnit číselník kategorií produktů.

Vytvoření nového produktu (jednoduchá verze)

Vyberte kategorii a poskytovatele. Dále vyplňte název produktu a informaci o datu zahájení a předpokládaném či žádaném datu ukončení prací. Dále si řekněme, jak se vypočítá výsledná cena (bez DPH) pro zákazníka. Od toho se bude odvíjet další vyplňování formuláře.

Stanovte si, jak budete daný produkt nabízet a od čeho se odvíjí náklady. V případě, že se náklady odvíjí od počtu participujících osob zašrtněte možnost *Počítat účastníky*. Pokud nevíte dobředu kolik hodin bude odučeno či kolik normostran bude účtováno a náklady na ně nejsou fixní, zaškrtněte možnost *Počítat množství*. Celková cena bude spočítána takto:

- jednotková cena je vynásobena počtem účastníků pokud byla odpovídající volba zaškrtnuta

- mezisoučet je vynásoben součtem odpracovaných jednotek v událostech ve výkazech přiřazených v produktu pokud byla volba *počítat množství* zaškrtnuta
- @todo možná konverze

Ted' už zbývá doplnit zbývajících povinné položky a to jednotku a jednotkovou cenu.

Výkazy

Vytvoření a přiřazení výkazu k produktu

Vytvoření a přiřazení výkazu k produktu provádějte pouze pokud je produkt, ke kterému chcete přidat výkaz, již vytvořen.

V menu klikněte na položku *Přidat výkaz* a vyplňte následný přehledný formulář. U dodavatele se rozlišuje se i typ smlouvy uvedený v závorce. Sazba a jednotka jsou rovněž povinné položky.

Jinou možností, jak přiřadit nový výkaz, je přes modul PRODUKTY (nebo přes „rychlé hledání“), kde lze snadno vyhledat daný produkt. Zobraze si detaily nalezeného produktu a ve skupině *Přiřazené výkazy* klikněte na odkaz *Přidat výkaz*.

Přidání události k výkazu

Tipy na urychlení práce

Zvláštní formulář na přidání produktu spolu se zákazníkem i dodavatelem umožní rychlejší zadávání většího množství nových produktů.

6.6 Dodavatel

Dodavatel má přístup, ke všem produktům, kde měl přiřazen alespoň jeden výkaz.

zadávání výkazu,
evidence docházky do kurzů

Závěr

Zavedení informačního systému je běh na dlouhou trať ... příprava dodavatelů i odběratelů na změny ve způsobu vykazování práce ...

Nedostatky Pro lepší ochranu uložených dat by se měl zavést ACL - Access Control List. V současné verzi systému je díky

Literatura

- [1] <http://www.informacny-system.sk>
- [2] Wikipedia: *Informační systém*,
http://cs.wikipedia.org/wiki/Informační_systém
- [3] Dana Soukupová: *Algoritmy logického návrhu relační databáze*, 2004
- [4] Cake Software Foundation: *The Cookbook*
<http://book.cakephp.org/>
- [5] Cake Software Foundation: *Porozumění Model-Pohled-Controller*
<http://book.cakephp.org/cz/view/10/Understanding-Model-View-Controller>
- [6] Wikipedia: *Model-view-controller*
<http://en.wikipedia.org/wiki/Model-view-controller>
- [7] Berners-Lee, Masinter & McCahill: Uniform Resource Locators (URL), *RFC1738*, Prosinec 1994
<http://www.ietf.org/rfc/rfc1738.txt>
- [8] Wikipedia: *Object-relational mapping*
<http://en.wikipedia.org/wiki/O-RM>
- [9] *I18n - Internacionalizace*
http://kore.fi.muni.cz:5080/wiki/index.php/I18n_-_Internacionalizace

- [10] *PV168/Lokalizace a internacionalizace*
http://kore.fi.muni.cz:5080/wiki/index.php/PV168/Lokalizace_a_internationalizace
- [11] Jason Chow: *P28n, the top to bottom persistent internationalization tutorial*
<http://bakery.cakephp.org/articles/view/p28n-the-top-to-bottom-persistent-internationalization-tutorial>
- [12] *Random password generator component for CakePHP*, Červen 2008
<http://www.solitechgmbh.com/2008/06/11/random-password-generator-component-for-cakephp/>
- [13] Rafael Bandeira: *Linkable Behavior. Taking it easy in your DB*, Listopad 2008
<http://blog.rafaelbandeira3.com/2008/11/16/linkable-behavior-taking-it-easy-in-your-db/>
- [14] Brandon Parise: *HABTM Add & Delete Behavior*, Květen 2007
<http://bakery.cakephp.org/articles/view/add-delete-habtm-behavior>
- [15] <http://www.mysql.com/>

sadsa
 dasd

Dodatek A

Ukázka XML databáze

A.1 DTD

```
<!-- Evidence kurzu -->
<!ELEMENT courses (course+,user+)>
  <!ELEMENT course (description,teachers,participants,
    events)>
    <!ATTLIST course id ID #REQUIRED>
  <!ELEMENT teachers (teacher)*>
    <!ELEMENT teacher (price)>
      <!ATTLIST teacher
        id ID #REQUIRED
        uid IDREF #REQUIRED
        master (0|1) "0"
      >
    <!ELEMENT participants (participant)*>
      <!ELEMENT participant (price)>
        <!ATTLIST participant
          id ID #REQUIRED
          uid IDREF #REQUIRED
        >
      <!ELEMENT price (#PCDATA)>
        <!ATTLIST price currency (CZK|EUR|USD) #IMPLIED >
    <!ELEMENT events (event)*>
      <!ATTLIST events
        from CDATA #REQUIRED
```

```

        to CDATA #REQUIRED
    >
<!--ELEMENT event (content,attendance*)>
<!--ATTLIST event
    teacher_id IDREF #REQUIRED
    date CDATA #REQUIRED
    time_from CDATA #REQUIRED
    time_to CDATA #IMPLIED
>
<!--ELEMENT content (#PCDATA|lesson)*>
<!--ELEMENT lesson (#PCDATA|chapter)*>
    <!--ELEMENT chapter (#PCDATA)>
<!--ELEMENT attendance EMPTY>
<!--ATTLIST attendance
    participator_id IDREF #REQUIRED
>
<!--ELEMENT description (#PCDATA)>
<!--ELEMENT user (name, (phone)*, (email)?)>
<!--ATTLIST user id ID #REQUIRED>
<!--ELEMENT name (lastname,firstname)>
<!--ELEMENT firstname (#PCDATA)>
<!--ELEMENT lastname (#PCDATA)>
<!--ELEMENT phone (#PCDATA)>
    <!--ATTLIST phone type (wired | mobile) "mobile">
<!--ELEMENT email (#PCDATA)>
<!-- Definice entity pro zjednodušení práce -->
<!--ENTITY at "@">
<!--ENTITY pcz "+420">

```

A.2 XML

A.3 XPath

A.4 XQuery

Dodatek B

Obsah přiloženého CD

Bakalářká práce tisknutelný PDF soubor s textem této práce.