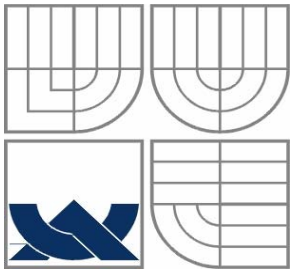


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV AUTOMATIZACE A INFORMATIKY

Faculty of mechanical engineering
Institute of Automation and Computer Science

Informační systém pro překladatelskou agenturu

INFORMATION SYSTEM FOR TRANSLATION AGENCY

DIPLOMOVÁ PRÁCE
DIPLOMA THESIS

AUTOR PRÁCE — **BC. RADEK BRODECKÝ**
AUTHOR

VEDOUCÍ PRÁCE — **ING. PETR KRČEK**
SUPERVISOR

BRNO 2008

Chtěl bych touto formou poděkovat vedoucímu své diplomové práce ing. Krčkovi za trpělivé vedení a společníkům z firmy Transcad – Ivo Leitgebovi a Davidu Varnerovi – za zajímavou a podnětnou spolupráci při vzniku a vývoji tohoto informačního systému.

Abstrakt

Diplomová práce se zabývá návrhem a implementací informačního systému pro konkrétní překladatelskou agenturu. Informační systém řeší administraci pracovníků a projektů. Klíčovou složkou systému je víceúrovňové zpracování dotazů na neznámé pojmy od externích pracovníků v rámci této překladatelské agentury a jejich zpětné odeslání pracovníkům, dále rozesílání dalších informačních dokumentů pracovníkům, plánování zdrojů a vyhodnocení práce. Navržený systém je implementován jako WWW aplikace v jazyce PHP s využitím databáze MySQL a technologie AJAX.

Abstract

Presented thesis deals with design and implementation of an information system for a local translation agency. The main purpose of the information system is to provide functions for effective user/project management. Key part of system is represented by a multi-level processing of terminology queries from external translators. Further features include distribution of documents to users, resource planning, and work evaluation. In order to provide instant 24/7 access, the system has been implemented as a WWW application using PHP script language, MySQL database, and AJAX technology.

Klíčová slova

informační systém, překladatelská agentura, zpracování dotazů, PHP, AJAX, MySQL, HTML, XML, jazyk UML, Unified Process, případ užití, diagram aktivit

Keywords

information system, translation agency, queries processing, PHP, AJAX, MySQL, HTML, XML, UML language, Unified Process, use case, activity diagram

Obsah

1 Úvod

Jako dlouholetý externí pracovník firmy Transcad jsem byl seznámen s ideou vytvoření informačního systému, který by odstraňoval některé významné nevýhody současného stavu organizace práce ve firmě. Projekt systému mi byl nabídnut i díky tomu, že jsem pro tuto firmu delší dobu jako externista pracoval a měl jsem tedy osobní zkušenost s procesy ve firmě.

Informační systém řeší několik oblastí chodu překladatelské agentury, především klíčovou a velmi specifickou oblast – zasílání dotazů od externistů, jejich zpracování a následné rozesílání zpět. Pokouší se odstranit některé největší nevýhody původního stavu, které mohou dost významně ovlivnit efektivitu práce jednotlivých externích pracovníků, například poměrně chaotické posílání dotazů prostřednictvím emailů, zasílání duplicitních dotazů, prodlevy při šíření nových pojmů k externistům nebo při aktualizacích stávajících pojmů, nutnost rozesílat často relativně rozsáhlé databáze pojmů externistům (v terminologii firmy tzv. glosáře). Díky uložení administrativních dat na jednom místě umožňuje systém zároveň i výpočet kapacit firmy, plánování a výpočet mezd za odvedenou práci. V neposlední řadě pomáhá i automatizovat některé rutinní úkoly, například rozesílání emailů s upozorněním na umístění várek na FTP, odevzdání várek, odeslání plateb na účty externistům atd. Při vytváření systému byla snaha některé jeho části parametrizovat.

Diplomová práce je obsahově rozdělena na dvě části, teoretickou a praktickou. První část pojednává o teorii informačních systémů, jejich architektuře, životním cyklu a některých typech informačních systémů. Dále obsahuje stručný popis jazyka UML a metodiky UP (Unified Process). Druhá část se zabývá samotnou realizací informačního systému. Tato část je rozdělena podle metodiky UP na dílčí etapy vývoje software – fáze zahájení, rozpracování konstrukce a zavedení, které jsou zde provedeny a konkrétně realizovány na vytvořeném informačním systému. Těžiště práce se nachází v realizaci samotného systému, proto je první teoretická část pojata spíše stručně a přehledově, také s ohledem na to, že toto téma již bylo mnohokrát zpracováno.

2 Teorie informačních systémů

2.1 Vymezení pojmů

Nejdříve je třeba provést vymezení následujících pojmů.

2.1.1 Informace

Informací se rozumí data nebo údaje, kterým člověk přisuzuje konkrétní význam a které uspokojují informační potřebu svého příjemce. I když jsou informace nehmotné, lze je zaznamenat nějakým fyzickým procesem.

2.1.2 Informační systém (IS)

Jde o soubor lidí, technických prostředků a metod, které zabezpečují získávání, přenos, zpracování, uchování a prezentaci informací pro příjemce, kterými mohou být uživatelé informačního systému či technické prostředky.

2.1.3 Informační technologie

Informační technologie jsou soubor nástrojů, metod a znalostí sloužících k činnostem, k nimž je informační systém určen (sběr, uchování, přenos informací).

2.1.4 Systémová integrace

Systémová integrace je proces zajišťující propojení (integraci) podniku a informačního systému. Dohlíží na to, aby informační systém splňoval požadavky, které se na něj kladou, aby byl budován jako celek na základě jasné a srozumitelné architektury a přesně vymezených standardů. Integrace probíhá na několika úrovních – datové, aplikační, hardwarové a technologické, metodické, na úrovni podnikových procesů a uživatelského rozhraní. Osoba či podnik, který tyto činnosti provádí, se nazývá systémový integrátor.

2.2 Komponenty informačního systému

Je důležité si uvědomit, že informační systém není tvořen jen samotným programem, ale je to soubor technických prostředků (hardware), programových prostředků (software) a organizačních prostředků (orgware), které se dále člení [citace].

Technické prostředky:

- počítače (osobní počítače, servery...)
- periferní zařízení (tiskárny, digitizéry...)
- síťové komunikační prostředky (kabeláž, routery, switche, síťové karty...)
- doplňková a podpůrná zařízení (zařízení na zálohování dat...)
- provozní materiál, dokumentace...

Programové prostředky:

- databázové systémy
- operační systémy
- standardní aplikační programy (textové editory, tabulkové procesory...)
- speciální aplikační programy, zhotovené podle individuálních požadavků

Organizační prostředky:

- Pokyny pro obsluhu a návody k obsluze.
- Provozní pokyny IS.
- Směrnice zajišťující dělbu a koordinaci prací kolem AIS.
- Směrnice stanovující zodpovědnost za správnost vkládaných dat.
- Zásady pro údržbu a inovaci AIS.
- Další směrnice a pokyny...

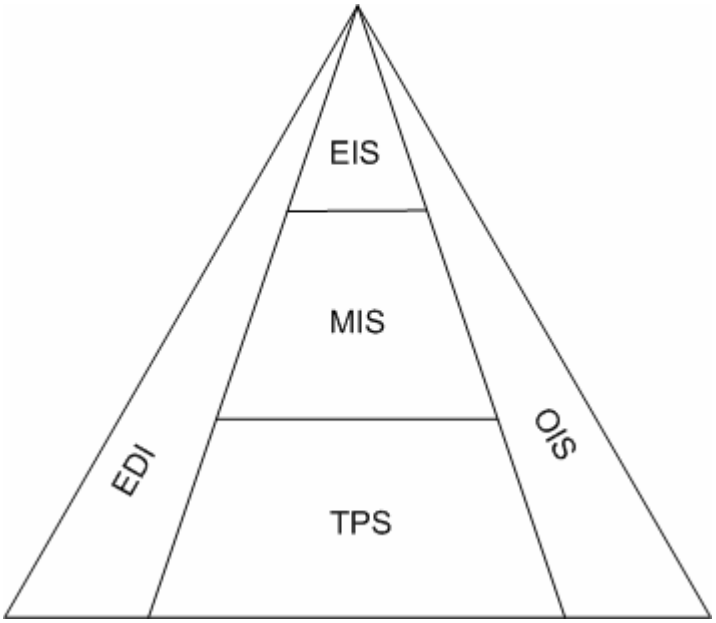
2.3 Architektura informačního systému

Architekturou rozumíme rámec pro další realizaci podle základních představ a myšlenek. Zachycuje budoucí podobu budovaného systému, jeho jednotlivé komponenty a vazby. Architektura může být otevřená, tzn. přístupná řadě změn a úprav (v rámci určitého zvoleného stylu). Tato otevřenost umožňuje přizpůsobovat měnícím se vnitřním i vnějším podmínkám.

2.3.1 Celková architektura

Celková architektura popisuje systém jako celek a zachycuje všechny podstatné dimenze informačního systému. Vychází z pochopení aktivit a cílů podniku, pro který se systém buduje. Je základní součástí pro řízení informačního systému a odvíjejí se z ní všechny další součásti IS (dílčí architektury).

V nejobecnějším případě je na celkovou architekturu pohlíženo [citace] jako na souhrn několika základních částí podle obrázku 2.1.



Obr.2.1 Celková architektura IS

2.3.2 EIS (Executive Information System)

Blok sloužící k podpoře vrcholového řízení organizace. Řeší se zde důležité úkoly, jako strategické plánování sloužící k dosažení cílů podniku, finanční plánování, vyhodnocování informací z vnitřních i vnějších procesů. Informace o různých objektech či procesech jsou

uchovávány nejenom z aktuálního okamžiku, ale i dlouho do minulosti, proto lze provádět i různé odhady na základě minulosti.

2.3.3 MIS (Management Information System)

Tento systém slouží k řízení podniku na taktické a operační úrovni, tzn. v krátkodobém či střednědobém časovém horizontu. Zabývá se takovými okruhy problémů, jako jsou ekonomika (účetnictví, majetek, finanční řízení) nebo organizační otázky (personalistika, mzdy). Tato část informačního systému je podobná i mezi organizacemi různého zaměření.

2.3.4 TPS (Transaction Processing System)

Tento blok je zaměřený na konkrétní činnost je zaměřený na konkrétní činnost organizace (na její hlavní zaměření) a na její operativní řízení. Podoba této části se liší podnik od podniku a může se skládat s dalších podpůrných komponent (CAD pro automatizovaný návrh výrobku, CAQ pro kontrolu kvality atd.).

2.3.5 OIS, EDI

Celým schématem architektury prolínají dva bloky. OIS (Office Information System) slouží pro podporu kancelářských prací, skládá se z řady aplikací, jako jsou textové editory, tabulkové procesory, programy pro elektronickou poštu, prezentaci atd. Část EDI (Electronic Data Interchange) zajišťuje komunikaci a výměnu dokumentů s okolím podniku (dodavatelé, odběratelé, banky, státní instituce atd.). V současné době se využívá především síť internet.

Konečná podoba architektury závisí na řadě faktorů. Předmět činnosti organizace ovlivňuje definici jednotlivých bloků architektury a jejich vzájemné propojení. Charakter činnosti se odráží především na úrovni TPS.

2.4 Dílčí architektury

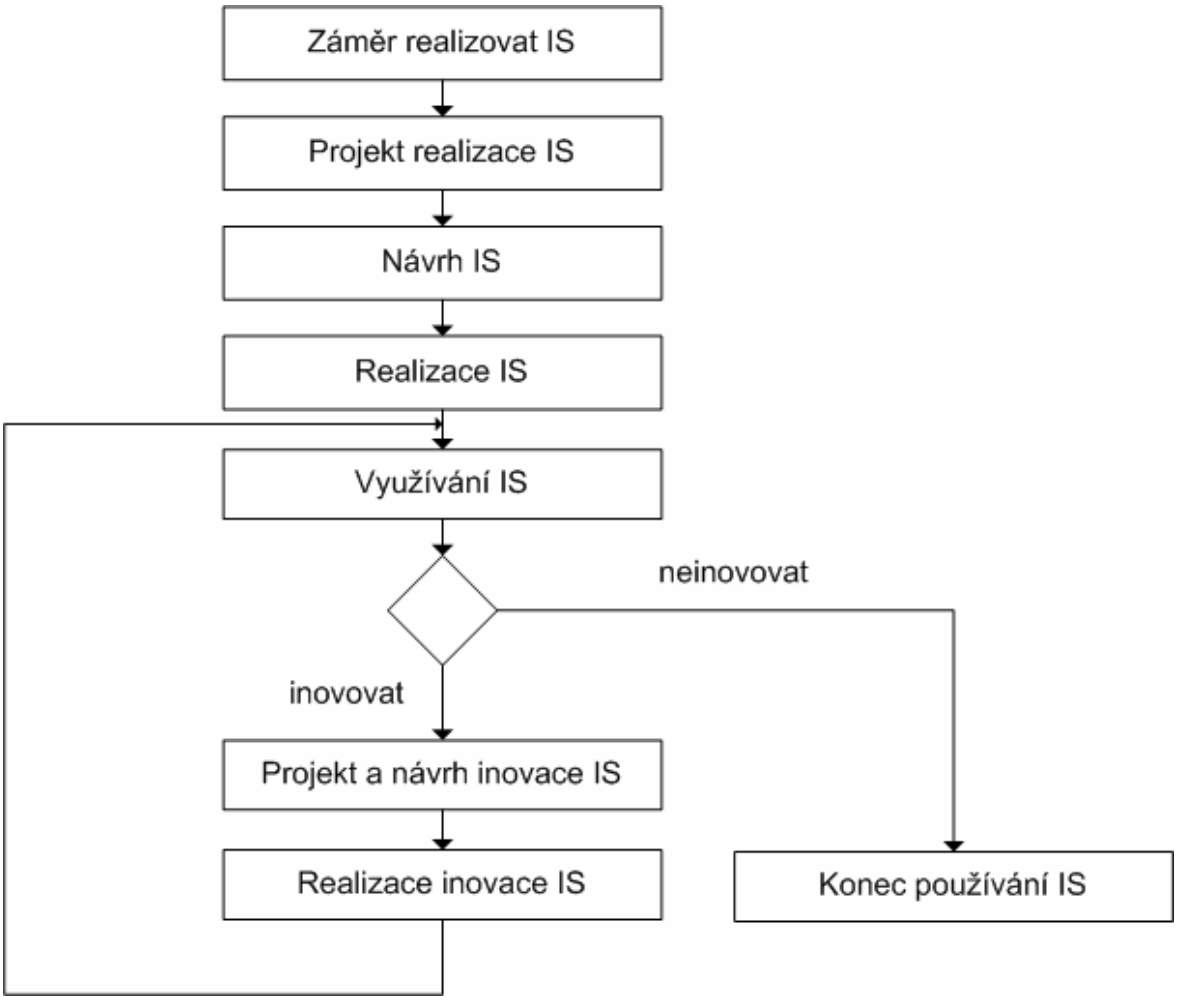
V rámci celkové architektury se rozlišuje řada dílčích, které prohlubují návrh IS.

Funkční architektura slouží k rozkladu základních bloků celkové architektury na menší části. Na nejnižší úrovni je proveden rozklad na elementární funkce (transakce). Procesní architektura popisuje činnosti, které budou v podniku probíhat. Vhodným nástrojem na modelování těchto procesů je kontextový diagram a na nižších úrovních Data Flow Diagramy (DFD), síťové grafy atd. Datová základna je navržena na základě datové architektury, kde se definují entity, jejich atributy a vztahy mezi entitami (často se vyjadřuje pomocí entitně relačního diagramu – ERD). Pomocí softwarové architektury se definuje, z jakých softwarových komponent se bude celý systém skládat. Každá komponenta má své vlastnosti (funkce, vstupní, výstupní a řídící data, algoritmy transformace dat, vývojové a provozní prostředí, které poskytuje), musí být zajištěna vzájemná spolupráce a odpovědnost těchto modulů. Hardwarová architektura určuje, jaké hardwarové prostředky budou při budování IS použity. Technologická architektura definuje způsob zpracování dat (řízené událostmi, v reálném čase, dávkové atd.), vnitřní stavba aplikací (klient/server, vrstvená architektura) a použité uživatelské rozhraní.

2.5 Životní cyklus informačního systému

Informační systém prochází podobně jako ostatní produkty určitými etapami, které se souhrnně nazývají životní cyklus. Tento cyklus začíná rozhodnutím podniku o nasazení informačního systému a končí stažením systému z používání.

Životní cyklus informačního systému lze vymezit etapami podle obrázku 2.2.



Obr. 2.2 Životní etapy IS

2.6 Vybrané typy funkcionalit současných podnikových IS

2.6.1 ERP (Enterprise Resource Planning)

Systémy ERP jsou takové systémy, které slouží k řízení podnikových dat a pomáhají plánovat celý logistický řetězec od nákupu přes sklady po výdej materiálu, řízení obchodních zakázek od přijetí až po expedici včetně plánování vlastní výroby a s tím spojené finanční a nákladové účetnictví i řízení lidských zdrojů. ERP ovlivňuje podnikové procesy, které podporuje a v mnoha případech automatizuje.

2.6.2 SCM (Supply Chain Management)

Systém pro řízení dodavatelského řetězce. Dnešní podniky se díky internetu propojují do složitějších struktur a vytvářejí vzájemně „prosíťovaná“ společenství. Jejich společným hlavním cílem je nabídnout s dostatečnou rychlostí a nízkými náklady požadovaný konkurenceschopný produkt. Řada činností začíná být outsourcována a využívají se možnosti specializovaných podniků, které disponují efektivně využitelnou technologií nebo know-how.

2.6.3 BI (Business Inteligence)

Aplikace BI představují produkty pro zlepšení kvality a výkonnosti podnikového řízení a zvýšení konkurenceschopnosti. Jsou určeny pro top a střední management a pro analytiky a plánovače – specialisty.

3 Jazyk UML a metodika UP

3.1 Jazyk UML

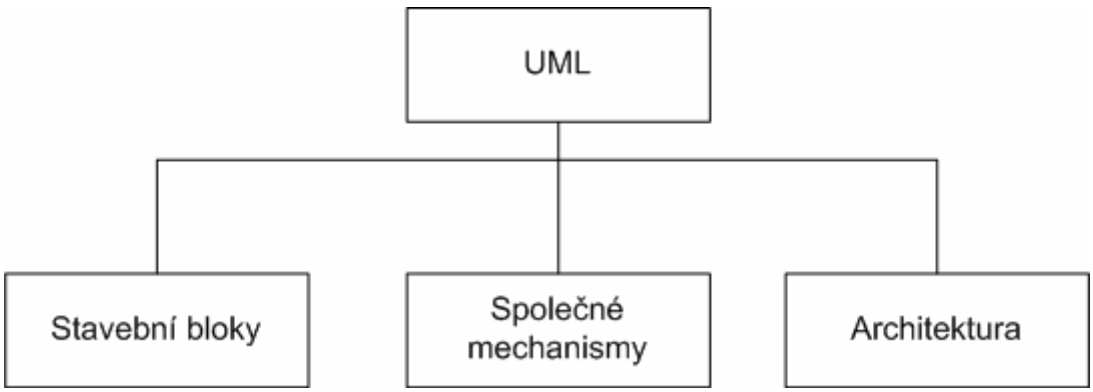
Zkratka UML znamená v českém překladu jednotný modelovací jazyk. Je to univerzální jazyk pro vizuální modelování systémů. Přestože se nejčastěji spojuje s modelování objektově orientovaných systémů, má mnohem širší využití. Jazyk UML byl navržen proto, aby spojil nejlepší existující postupy modelovacích technik a softwarového inženýrství. Proto je navržen tak, aby ho mohly implementovat všechny nástroje CASE (Computer Aided Software Engineering). Tato koncepce vychází ze skutečnosti, že rozsáhlé softwarové systémy se obvykle bez podpory nástrojů CASE neobejdou. Diagramy vytvořené v jazyku UML jsou srozumitelné pro lidi, ale navíc je mohou interpretovat i programy CASE [1].

Je důležité si uvědomit, že samotný jazyk UML nenabízí žádný druh metodiky modelování. Poskytuje pouze vizuální syntaxi, kterou můžeme využít při sestavování modelů. Také není vázán na žádnou specifickou metodiku nebo životní cyklus. Lze jej použít se všemi existujícími metodami. Metodika UP jej využívá jako vlastní syntaxi pro vizuální modelování, protože je pro tento jazyk nejlépe adaptovaná.

3.2 Struktura jazyka UML

Jak je možné vidět na obrázku 3.1, struktura jazyka UML obsahuje tyto 3 základní části:

- **Stavební bloky.** Jsou to základní prvky modelu, relace a diagramy.
- **Společné mechanismy**. Obecné způsoby, jimiž v jazyku UML dosáhnete specifických cílů.
- **Architektura**. Pohled v jazyku UML na architekturu navrhovaného systému.



Obr.3.1 Struktura jazyka UML

Stavební bloky jazyka UML jsou tři [1]:

- **Předměty (things).** Jsou to samotné prvky modelu, například třídy, rozhraní, komponenty.
- **Vztahy (relationships).** Pojítka mezi předměty. Relace určují, jak spolu dva nebo více objektů významově souvisí.
- **Diagramy (diagrams)**. Jsou to pohledy na modely UML. "Vyprávějí příběh" o systému a jsou způsobem vizualizace, co bude systém dělat (analytické diagramy) a jak to bude dělat (návrhové diagramy).

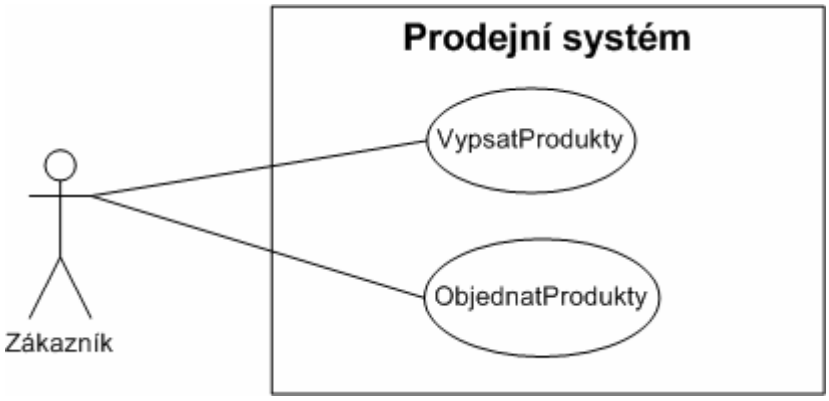
Jelikož v této práci budu používat dva druhy diagramů, a to diagramy případů užití a diagramy aktivit, nyní oba typy stručně přiblížím.

3.2.1 Diagram případu užití

Případy užití zachycují přesně funkčnost, která bude navrhovaným programem nabízena, a každý z nich popisuje jedno z použití systému. Znázorňují chování systému z pohledu uživatele. Umožňují definovat ohraničený systém a jeho vztahy s vnějším okolím. Podávají v podstatě obraz funkčnosti systému, která je vyvolávána podněty zvnějšku.

Diagram se skládá pouze z několika symbolů:

- **Aktér** – znázorněn postavičkou – může se jednat jak o osobu, ale také o organizaci, či externí systém, který nějakým způsobem spolupracuje s naším systémem.
- **Případ užití** – znázorněn elipsou – popisuje sekvenci činností, které systém může vykonat prostřednictvím interakce s vnějšími účastníky (aktéry)
- **Hranice systému** – rámeček. Jeho popiskem určujeme název sytému. Účastníky (aktéry znázorňujeme vně jeho hranic, zatímco případy užití, které charakterizují chování samotného systému, umisťujeme uvnitř rámečku.
- **Spojnice** – znázorněna plnou čarou (v UML plná čára značí přiřazení) a vyjadřující interakci mezi aktérem a případem užití nebo definující vztahy mezi samotnými případy užití (někdy se kreslí přerušovanou čarou). Spojnice mezi samotnými případy užití se nazývají relace a rozeznáváme dvoje:
  - **Relace <<uses>>** (někdy se označuje také <<include>>). Vyčlenění společného případu užití ze scénářů základních případů užití. Například při opakujících se scénářích je doporučeno tento vyjmout, než udržovat více shodných kopií. Důležité je, že základní scénář není bez rozšiřujícího kompletní.
  - **Relace <<extend>>** Rozšiřuje základní případ užití o nové doplňkové chování. Základní případ užití je však sám o sobě funkční.



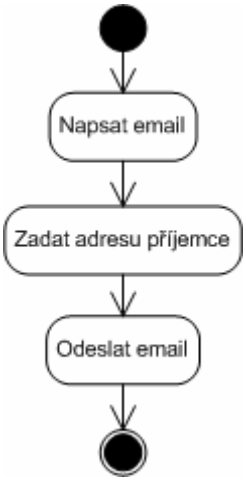
Obr 3.2 Příklad případu užití

3.2.2 Diagramy aktivit

Modelují procesy jako kolekce aktivit a přechodů mezi nimi. V diagramech se setkáme s těmito symboly:

- **Akce** – znázorněné obdélníkem se zakulacenými rohy. Popisují stav akce, nebo také aktivity, je dále nedělitelnou jednotkou diagramu. Může se jednat o krok algoritmu, nebo činnosti.
- **Dílčí aktivity** – znázorněné podobně jako akce, ale narozdíl od nich již nejsou nedělitelné a dají se rozdělit na další dílčí aktivity, nebo na dále již nerozložitelné akce.

- **Přechody** – znázorněné šipkou. Ukončení akce, či aktivity vyvolá automaticky bezprostřední přechod do dalšího stavu.
- **Hodnocení přechodu** – znázorněný kosočtvercem. K hodnocení vede vždy pouze jedna cesta, hodnocených výstupních přechodů ovšem může být mnoho, ale jen jeden může být aktivován, a to ten, který splňuje podmínku přechodu. Ty je třeba volit tak, aby se vzájemně vylučovaly a diagram aktivit zůstal deterministický.
- **Rozvětvení a spojení** – rovná vodorovná čára. Diagramy aktivit umí modelovat souběžné toky činností. Přechod lze rozvětvit do dvou (ale i více) souběžných toků a následně je zpětně synchronizovat prostřednictvím spojení.
- **Zóny** (nebo také plavební zóny z anglického originálu swimlanes) – pomocí zón můžeme specifikovat, kdo jakou aktivitu provádí. Diagram rozdělíme do vertikálních pruhů oddělených svislými čarami, kde každý pruh označuje jednotlivou třídu nebo třeba osobu a aktivity v ní jsou pak činnosti prováděné touto osobou (třídou, oddělením...).
- **Signály** – přenáší asynchronně informace mezi objekty.



Obr. 3.3 Příklad diagramu aktivit

3.3 Metodika UP

Projekt UML vznikl z potřeby nabídnout jak vizuální jazyk, tak proces tvorby softwarového vybavení. To, co známe dnes pod pojmem UML, je jazykovou částí projektu, metodika UP je část procesní. Na rozdíl od jazyka UML však není metodika UP standardem. Metodika UP je založena na metodách Ericsson (Ericsson approach, 1967), Rational (Rational Objectory Process,1996-197) a dalších zdrojích. Je to pragmatická a ověřená metoda vývoje softwaru, která zahrnuje nejlepší ověřené postupy. S metodikou UP úzce souvisí metodika RUP (Rational Unified Process), která je její komerční verzí dodávanou společností IBM. RUP obsahuje veškeré standardy a nástroje jako UP, je však navíc dodávána společně s bohatým uživatelským prostředím doplněným o úplnou dokumentaci a „rádce“ k jednotlivým nástrojům [1].

3.3.1 Iterativní a přírůstkový proces metodiky UP

Klíčovou myšlenkou metodiky UP je rozdělení softwarového projektu na řadu menších „miniprojektů“. Každý se zmiňovaných „miniprojektů“ je považován za iteraci. Důležitá je skutečnost, že každá iterace obsahuje všechny prvky normálního softwarového projektu:

- plánování,
- analýzu a návrh,

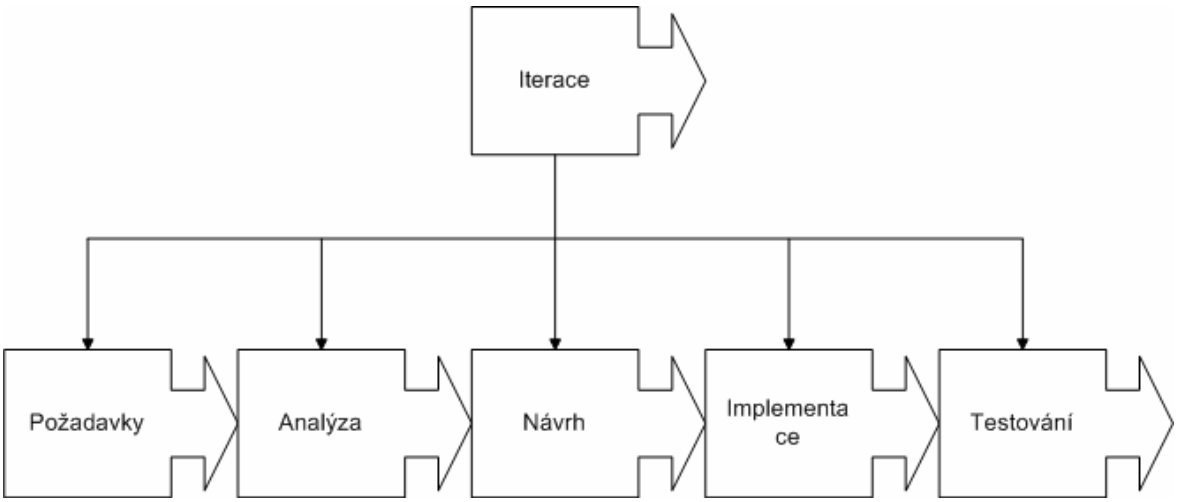
- tvorbu,
- integraci a testování,
- interní nebo externí vedení.

Každá iterace generuje vlastní základní linii (baseline), která se skládá z částečně kompletní verze finálního systému a z přidružené projektové dokumentace. Zmiňované základní linie jsou vrstveny tak dlouho, dokud není dosažena konečná podoba vytvářeného systému. Rozdíl mezi dvěma základními liniemi je označován za přírůstek. Právě proto je životní cyklus projektů podle metodiky UP označován jako interaktivní a přírůstkový.

3.3.2 Pracovní postupy iterace

V každé iteraci existuje pět následujících základních pracovních postupů (workflows), které určují, co je potřeba provést, a způsob, jak daného cíle dosáhnout (viz obrázek 3.4):

- **Požadavky.** Zachycují, co by měl systém dělat.
- **Analýza.** Vybroušení požadavků a jejich strukturování.
- **Návrh.** Realizace požadavků v architektuře systému.
- **Implementace**. Tvorba softwaru.
- **Testování**. Ověření, zda implementace funguje tak, jak se očekává.



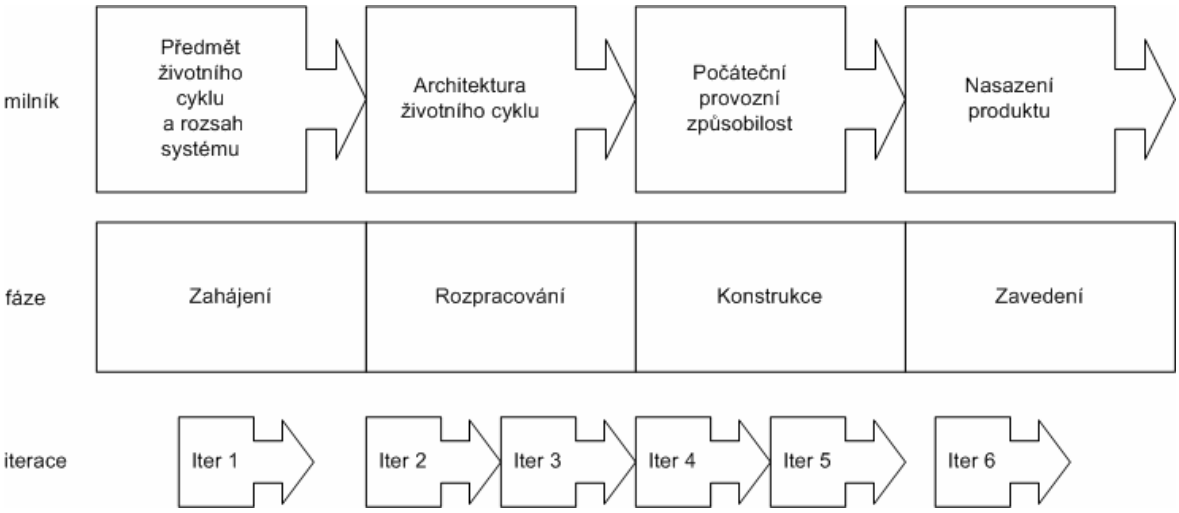
Obr. 3.4 Pracovní postupy iterace.

3.3.3 Struktura metodiky UP

Na obrázku 3.5 je znázorněna struktura metodiky UP. Životní cyklus projektu je rozdělen na 4 fáze:

- **Zahájení** – období plánování.
- **Rozpracování** – období architektury.
- **Konstrukce** – počátky provozuschopnosti.
- **Zavedení** – nasazení produktu do uživatelského prostředí.

Všechny fáze končí definovanými hlavními milníky, které jsou indikátory pokroku projektu. Každá fáze může obsahovat jednu nebo více iterací. V každé iteraci lze realizovat pět základních pracovních postupů a libovolný počet dodatečných pracovních postupů.



Obr. 3.5 Fáze životního cyklu projektu

Jak projekt přechází z jedné fáze do druhé, mění se i objem práce vykonávané v každém z pěti základních pracovních postupů. Například ve fázi zahájení je objem testování nulový, protože není co testovat. Podobně ve fázi zavedení již nedochází ke sběru požadavků a jejich analýze atd.

4 Fáze Zahájení

Ve fázi zahájení byla provedena analýza firmy, proveden výběr technologií, stanoven rozsah systému, definovány hlavní požadavky a vytvořeno základní schéma systému.

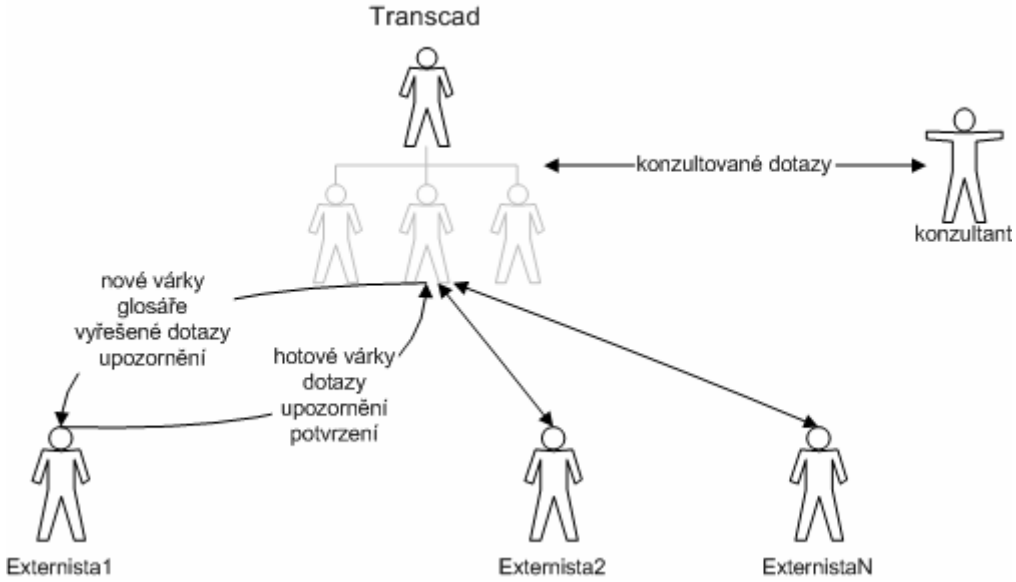
4.1 Analýza firmy Transcad

4.1.1 Profil firmy

Transcad je firma podnikající v oblasti překladatelství. Zájmovým jazykem byla zvolena angličtina, hlavním oborem pak technické překlady z oblasti výpočetní techniky, elektrotechniky, systémů GIS a strojírenství. V současnosti se zabývá lokalizacemi a překlady sofistikovaných systémů a aplikací z oborů CAD, CAE, CAM, ERP, FM a CIM, dále technickou literaturu (publikace zaměřené na nové technologie – MS Windows, aplikace MS Office, hardware PC, programovací a „internetové“ jazyky atd.). Nejnovějším polem působnosti firmy jsou také lokalizace webových stránek a tzv. dokumentaristika neboli tvorba dokumentace k dodanému softwaru.

4.1.2 Struktura firmy

Samotnou firmu tvoří dva společníci, kteří dále zaměstnávají proměnlivý počet externistů, často i z řad studentů. Při občasných konzultacích jsou kontaktováni externí konzultanti.



Obr. 4.1 Organizační schéma firmy Transcad

Jak je patrné z organizačního schématu firmy, směrem od firmy Transcad proudí k externistům texty (tzv. *várky*) určené k přeložení, většinou jako datové soubory ve formátu některého z používaných překladových programů (například Idiom Worldserver, SDL Trados atd.). Dále směřují k externistům databáze pojmů neboli *glosáře*, vyřešené dotazy a upozornění na změnu terminologie a různé události, například na novou databázi pojmů, novou várku atd.

Externisté odesílají firmě Transcad hotové várky a dotazy, dále podobně jako Transcad potvrzují a oznamují odeslání a přijetí várky, stažení databáze pojmů atd.

4.2 Slovník projektu zavedení informačního systému

Pro projekt zavedení informačního systému firmy Transcad (dále jen IS Transcad) byl vytvořen následující slovník:

Tab. 4.1 Slovník projektu informačního systému

Pojem	Definice
dotaz	Datová struktura dle tabulky 4.2. vyměňovaná mezi firmou Transcad, externisty a konzultanty. Synonyma: žádná
externista	Pracovník pracující pro firmu Transcad. Synonyma: externí pracovník
glosář	Databáze pojmů, obvykle z předchozí verze překládaného projektu. Obsahuje desítky až stovky odborných pojmů v angličtině týkajících se daného projektu, jejich český ekvivalent a komentář k pojmu. Externistům je většinou k dispozici ve formě tabulky aplikace MS Excel. Synonyma: žádná
konzultant	Osoba, se kterou firma Transcad konzultuje pojmy nebo dotazy. Synonyma: žádná
mzdový list	Datová struktura obsahující informace o jedné nebo více várkách provedených za určitou dobu, například za měsíc, externistovi, který práci provedl, hrubé a čisté mzdě a datumu odeslání peněz. Synonyma: ML
pohled	Webové rozhraní umožňující přístup do informačního systému. Synonyma: žádná
Rádce	Funkce IS pomáhající externistům s vyhledáváním neznámých pojmů, realizovaná jako interaktivní HTML stránka s technologií AJAX. Synonyma: žádná
Transcad	Vlastní překladatelská agentura tvořená 2 společníky. Synonyma: žádná
upozornění	Email s upozorněním na důležitou událost, například odeslání várky, odeslání platby na účet apod. Synonyma: žádná
várka	Objem práce rozesílaný externím pracovníkům. Obvykle je tvořena větším počtem denních jednotek. Součástí várky jsou soubory určené ke zpracování za daný časový úsek a informace o várce, například počátečním termínu, koncovém termínu atd. Také datová struktura dle tabulky 3.3. Synonyma: žádná

Tab. 4.2 Datová struktura dotazu

Atribut	Datový typ
id_dotazu	int unsigned
datum	timestamp
glosar	tinyint
eng_pojem	text
cze_pojem	text
kontext	text
komentar	text
odkaz	varchar(128)
stav	enum
id_projektu	int
id_osoby	int

Tab. 4.3 Datová struktura várky

Atribut	Datový typ
id_varky	int
cislo_varky	int
id_osoby	int
id_projektu	int
pocet_jednotek	float
počet_souboru	smallint
zacatek	date
konec	date
pocet_dni	smallint
chyby	varchar
srazka	float
cena	float
stav	enum
id_ml	int

4.3 Představa o funkcích nového informačního systému

Při schůzkách se společníky z firmy Transcad byly zaznamenány největší nedostatky současného stavu organizace ve firmě a navrženy funkce nového informačního systému, které mají tyto nedostatky řešit. Výsledky jsou shrnuty v tabulce 3.4.

Tabulka 4.4 Představa o funkcích nového IS

Současný stav	Nevýhoda	Zlepšení pomocí nového IS
Posílání dotazů pomocí emailů	Zmatky v systému značení emailů, nejednotnost struktury dotazů.	Pokládání dotazů přes jednotné WWW rozhraní. IS sám řeší systém značení dotazů.
Posílání dotazů pomocí emailů	Pokládání duplicitních dotazů různými nebo i stejnými externisty	IS nabízí externistům řešení dotazu s výskytem hledaného pojmu z centrální databáze, a to okamžitě po vložení pojmu do databáze.
Nutnost rozesílat rozsáhlé glosáře.	Velký objem posílaných dat.	Data uložena na serveru v databázi, ke které přistupuje externista pomocí WWW rozhraní.
Externisté zapomínají stáhnout nové verze glosářů.	Externisté nepracují s aktuálními pojmy.	Externista má vždy aktuální pojmy díky uložení dat v ústřední databázi.
Externisté zapomínají potvrzovat důležité události (přijetí várky, odeslání hotové várky).	Vedení firmy nemá přehled o aktuálním stavu firmy.	Systém odesílá upozornění sám po provedení daného úkonu.
Administrace externistů a projektů prováděna v samostatných aplikacích (Microsoft Excel).	Nutnost používat při sledování firmy více aplikací.	Všechny prostředky správy a plánování v jednom IS.
Výpočet kapacit firmy pomocí samostatných aplikací.	Nutnost používat při sledování firmy více aplikací.	Všechny prostředky správy a plánování v jednom IS.

4.4 Výběr technologií

Díky tomu, že firma má k dispozici webový server Apache s podporou PHP a databází MySQL, bylo rozhodnuto použít pro systém tyto technologie. Jde o velmi známou, oblíbenou a často používanou kombinaci, která se někdy označuje zkratkou LAMP (Linux, Apache, MySQL, PHP) nebo PHP triad. Jelikož jde o dlouho používané a prověřené produkty, jsou všechny poměrně spolehlivé a nabízí širokou paletu funkcí a možností, ať už se jedná o nastavení serveru Apache nebo nepřeberné množství funkcí a rozšíření jazyka PHP. Další nezanedbatelnou výhodou je bezplatná dostupnost všech těchto produktů. Protože se jedná o známé a dlouhodobě používané technologie, bude zde uvedena jen stručná charakteristika s přehledem použitých verzí. Podrobněji je zmíněna jen technologie AJAX v kapitole 6.

4.4.1 Apache HTTP Server

Apache HTTP Server je webový server s otevřeným kódem pro různé platformy, například Linux, Microsoft Windows nebo BSD. Instalace serveru je ke stažení zdarma na stránkách projektu na adrese *http://httpd.apache.org/*. Na serveru s informačním systémem běží verze Apache 2.0.54.

4.4.2 MySQL

Databázový systém vytvořený švédskou firmou MySQL AB. Lze jej instalovat na servery s různými operačními systémy, například Linux nebo Windows. Na serveru s informačním

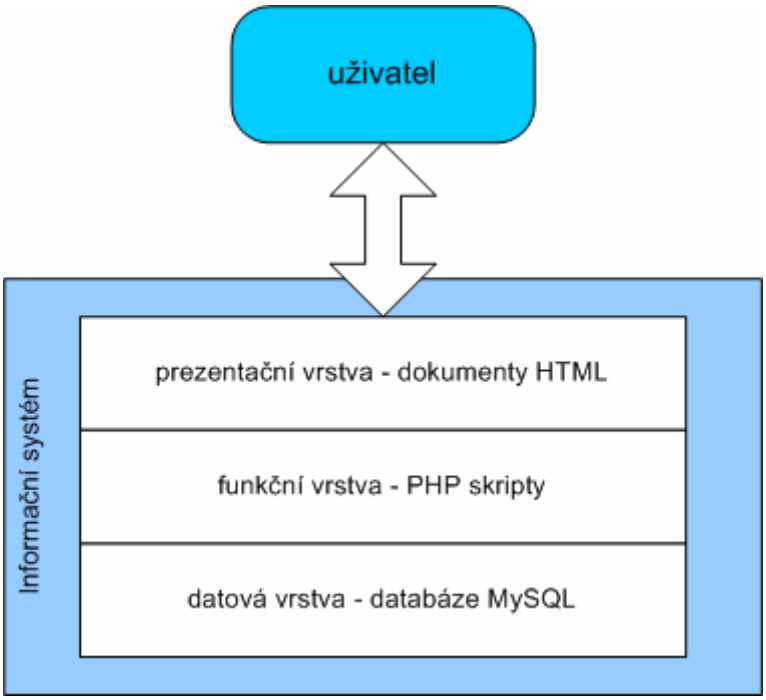
systémem je momentálně instalována verze MySQL 5.0.51. Databáze MySQL je dostupná ke stažení na adrese http://www.mysql.com.

4.4.3 PHP

PHP je skriptovací programovací jazyk používaný především pro programování dynamických internetových stránek. Skripty jazyka PHP běží na straně serveru. I když je založen na jazyku C, kombinuje v sobě vlastnosti více jazyků a má poměrně volnou syntaxi, což ho společně s dalšími vlastnostmi, zejména bezplatnou dostupností, činí značně oblíbeným a používaným. Od verze 5 byla výrazně zlepšena podpora objektů. Jazyk PHP má však i některé významné nevýhody, například není zaručena zpětná kompatibilita mezi verzemi, fakt, že programátor nemá zaručené cílové prostředí nebo že mnoho věcí přímo ovlivňujících chování PHP lze změnit pouze mimo soubory projektu (např. v konfiguračním souboru php.ini). Server s informačním systémem podporuje PHP verze 5.2.0. PHP je možné stáhnout ze stránky http://www.php.net.

4.5 Horizontální architektura systému

Informační systém používá klasickou třívrstvou architekturu. Prezentační vrstvu systému tvoří dokumenty HTML, ke kterým přistupují uživatelé pomocí webových prohlížečů. Tyto dokumenty jsou generovány PHP skripty. Data pro zpracování získává systém z databáze MySQL. Vzhledem k funkčním požadavkům informačního systému a potřebě dynamicky měnit obsah některých stránek, zejména u funkce Rádce, ale i v jiných částech systému, například u tvorby nových várek, bylo dále rozhodnuto o použití technologie AJAX (asynchronní javascript).



Obr 4.2 Horizontální architektura systému

4.6 Rozdělení systému na moduly

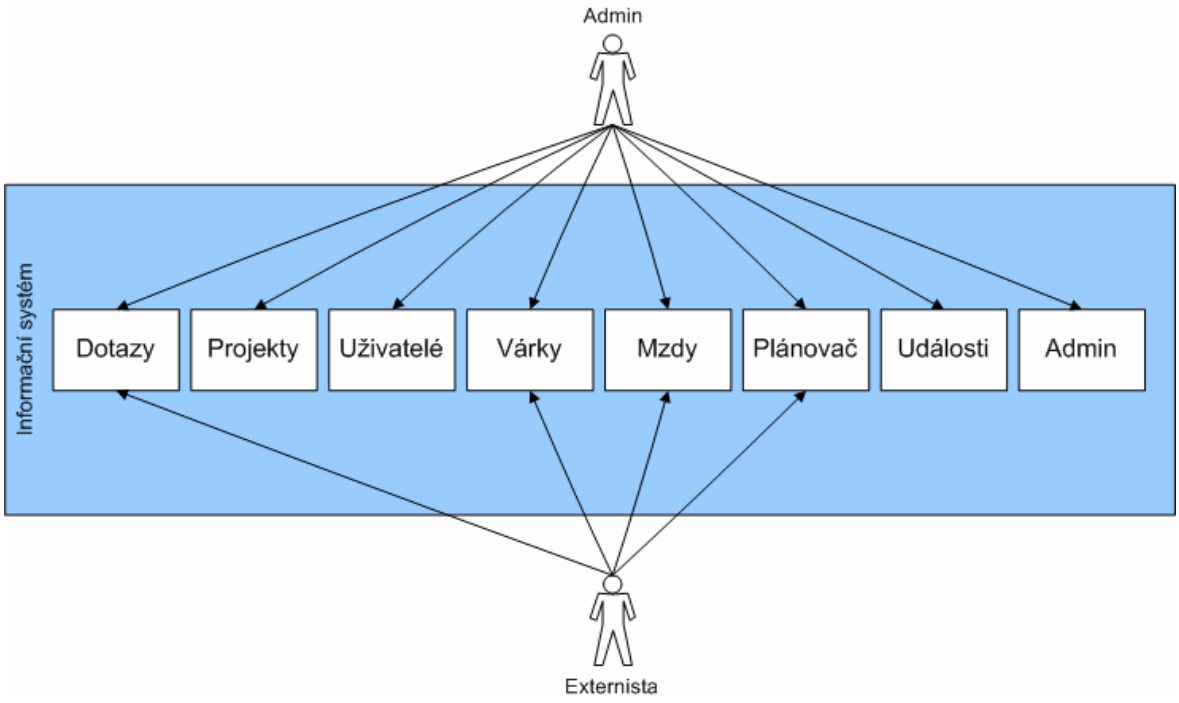
Systém se člení do několika částí nazývaných moduly. Jednotlivé moduly tvoří zároveň záložky v kartovém rozhraní systému v hlavním pohledu. Přehled modulů a jejich hlavních funkcí je v následující tabulce.

Tab. 4.5 Přehled modulů informačního systému

Název modulu	Popis modulu
Dotazy	• Realizuje vícekrokové zpracování dotazů od externistů. • Umožňuje uživatelům přístup k bázím pojmů uložených v tabulkách vyřešených dotazů nebo importovaných z externích databází (glosáře). • Součástí modulu je funkce *Rádce*, která pomáhá externistům hledat zadaný pojem a zobrazuje výsledky hledání.
Projekty	• Slouží pro administraci projektů. • Umožňuje: o vytvoření a odstranění projektu, o zadání informací o projektu, o import a export glosářů, o přiřazení uživatelů k projektům, o nastavení dalších atributů projektu (barva, soubor informací atd.).
Uživatelé	• Slouží pro administraci uživatelů systému. • Umožňuje: o vytvářet a odstraňovat uživatele, o zadávat informace o uživatelích, o nastavovat stavy a úrovně uživatele.
Várky	• Slouží k administraci várek. • Umožňuje vytvořit, editovat a odstranit várku. • Slouží pro sledování toku práce ve firmě.
Mzdy	• Realizuje výpočet odměn pro externisty • Slouží pro vytváření a export mzdových listů.
Plánovač	• Slouží pro sledování volných pracovních kapacit ve firmě a plánování zdrojů.
Události	• Slouží pro sledování vybraných událostí v systému a hromadné rozesílání zpráv externistům.
Admin	• Slouží pro správu systému, umožňuje měnit některá nastavení systému.

4.7 Rozdělení přístupových práv, pohledů a přístupu k modulům

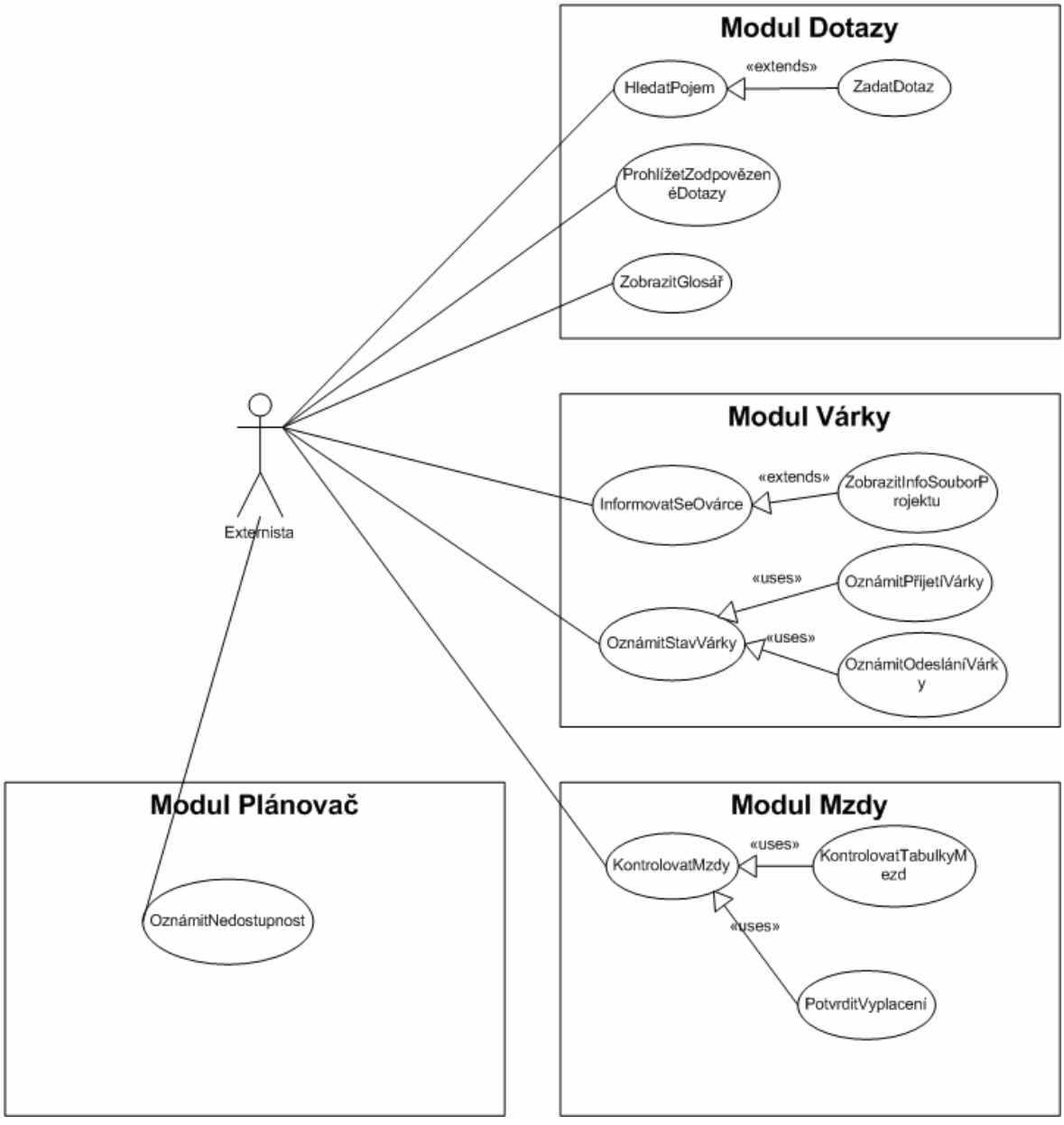
S ohledem na funkční požadavky systému byly definovány 2 kategorie přístupových práv, a to kategorie **Admin** a **Externista**. Jak názvy napovídají, do kategorie Externista spadají všichni externisté firmy, zatímco kategorii Admin budou používat společníci z firmy Transcad. Dostupnost modulů pro tyto kategorie je znázorněna na obrázku 3.5. I když mají tyto kategorie přístup ke stejným modulům, pro obě kategorie se WWW rozhraní těchto modulů liší.



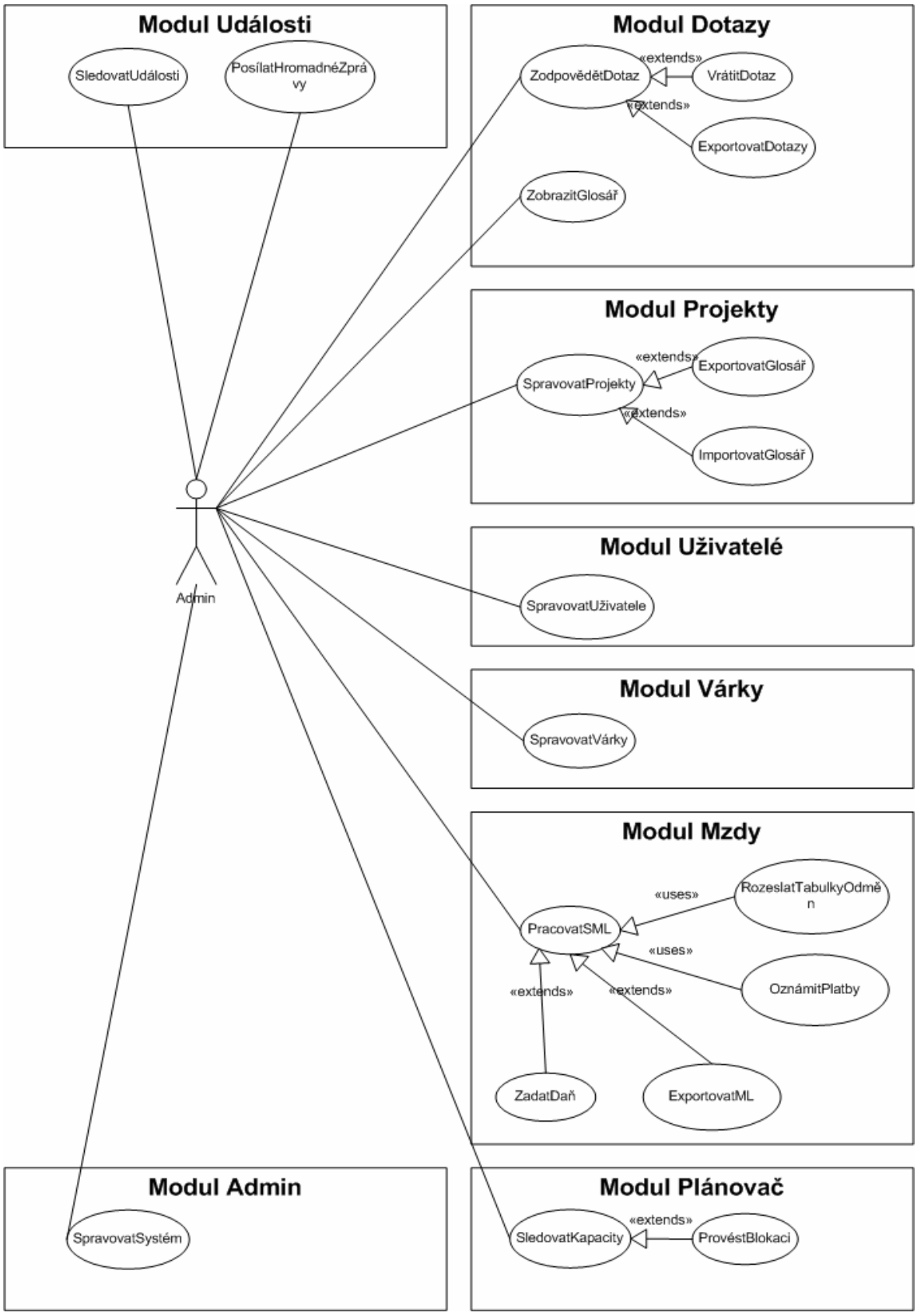
Obr. 4.3 Rozdělení přístupu k modulům podle přístupových práv

5 Fáze Rozpracování

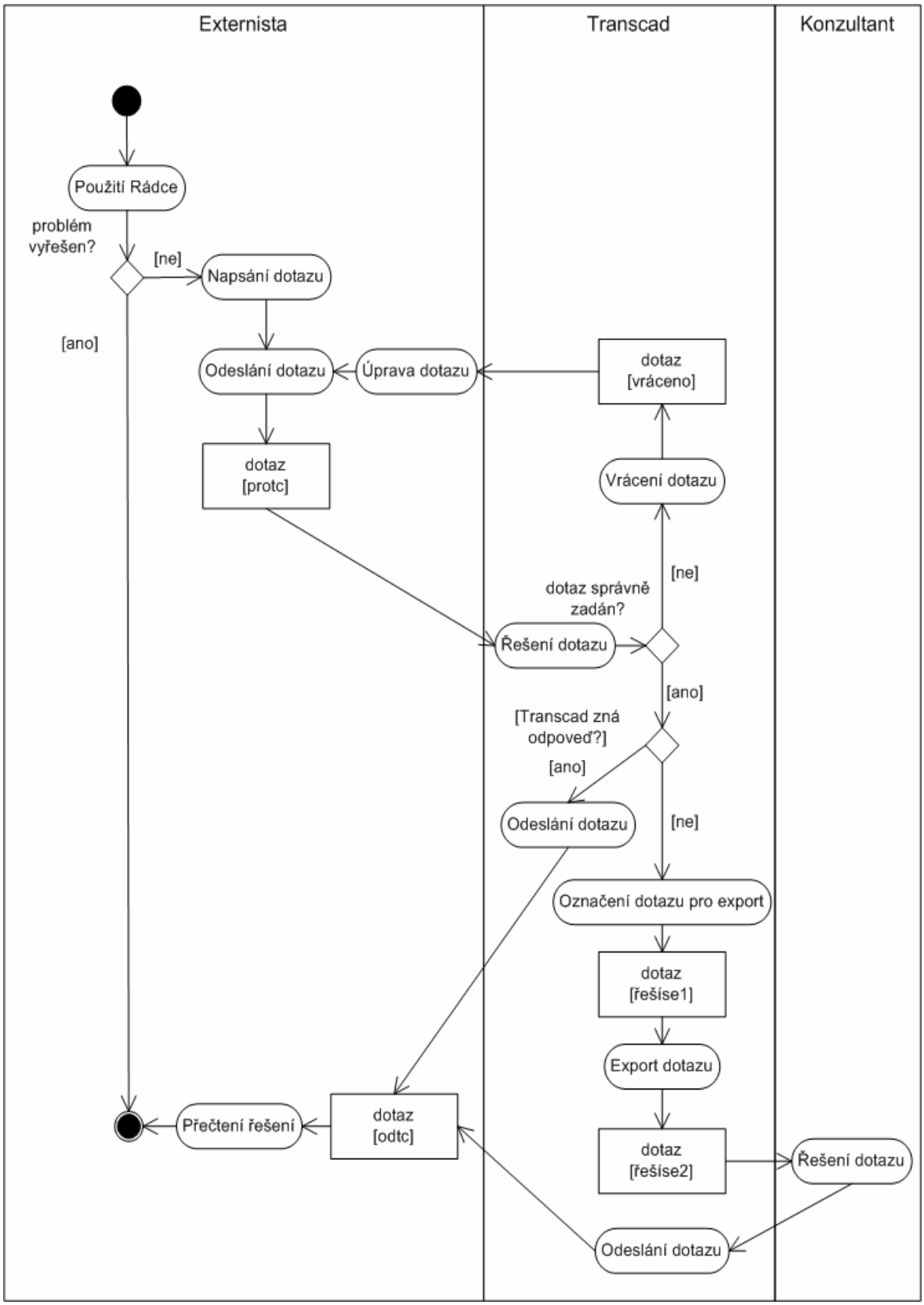
Ve fázi Rozpracování byl vytvořen model případů užití, nalezeni aktéři a vytvořeny diagramy vybraných případů užití, scénáře případů užití a jejich diagramy aktivit.



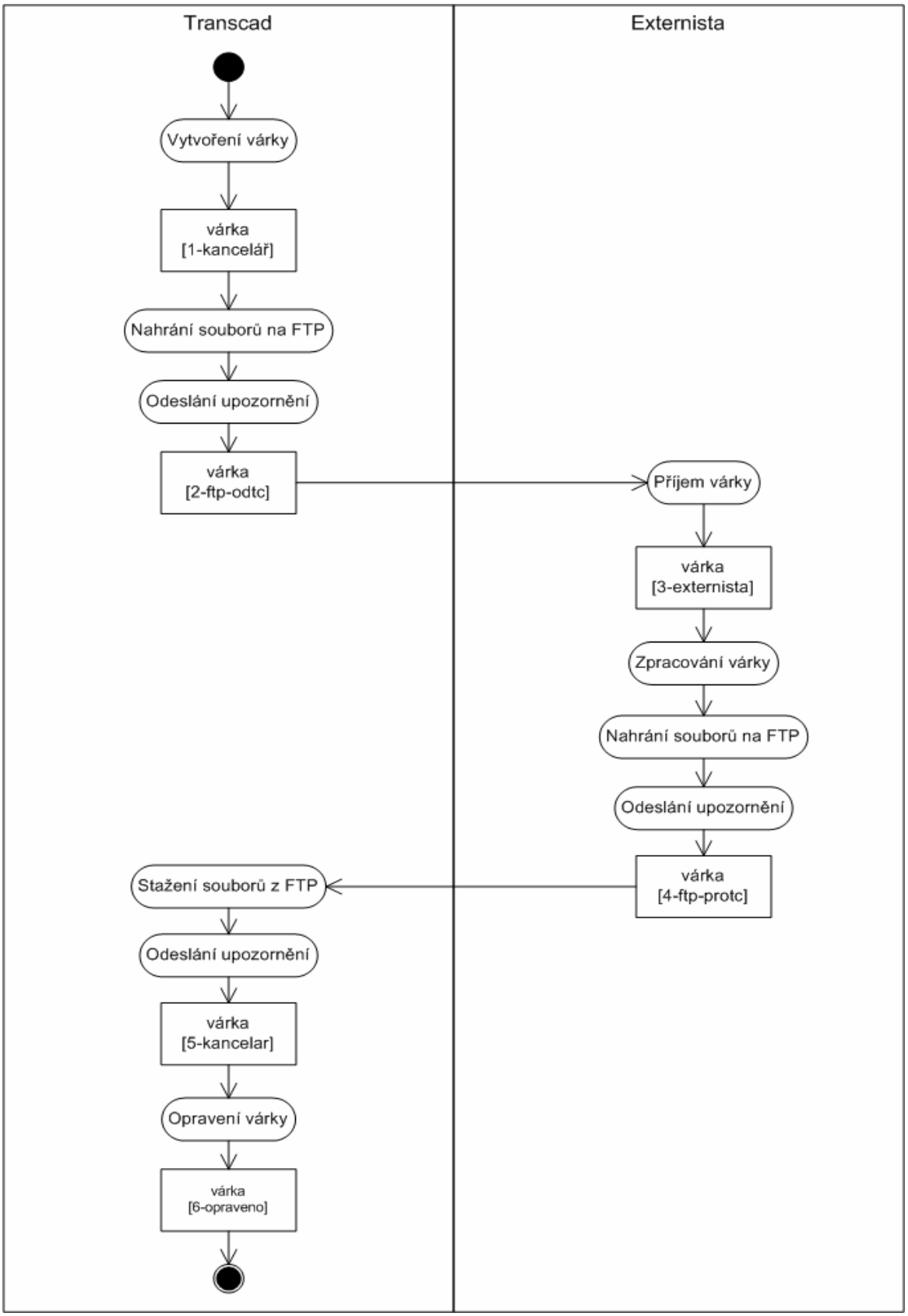
Obr. 5.1 Diagram případu užití Práce se systémem – externista



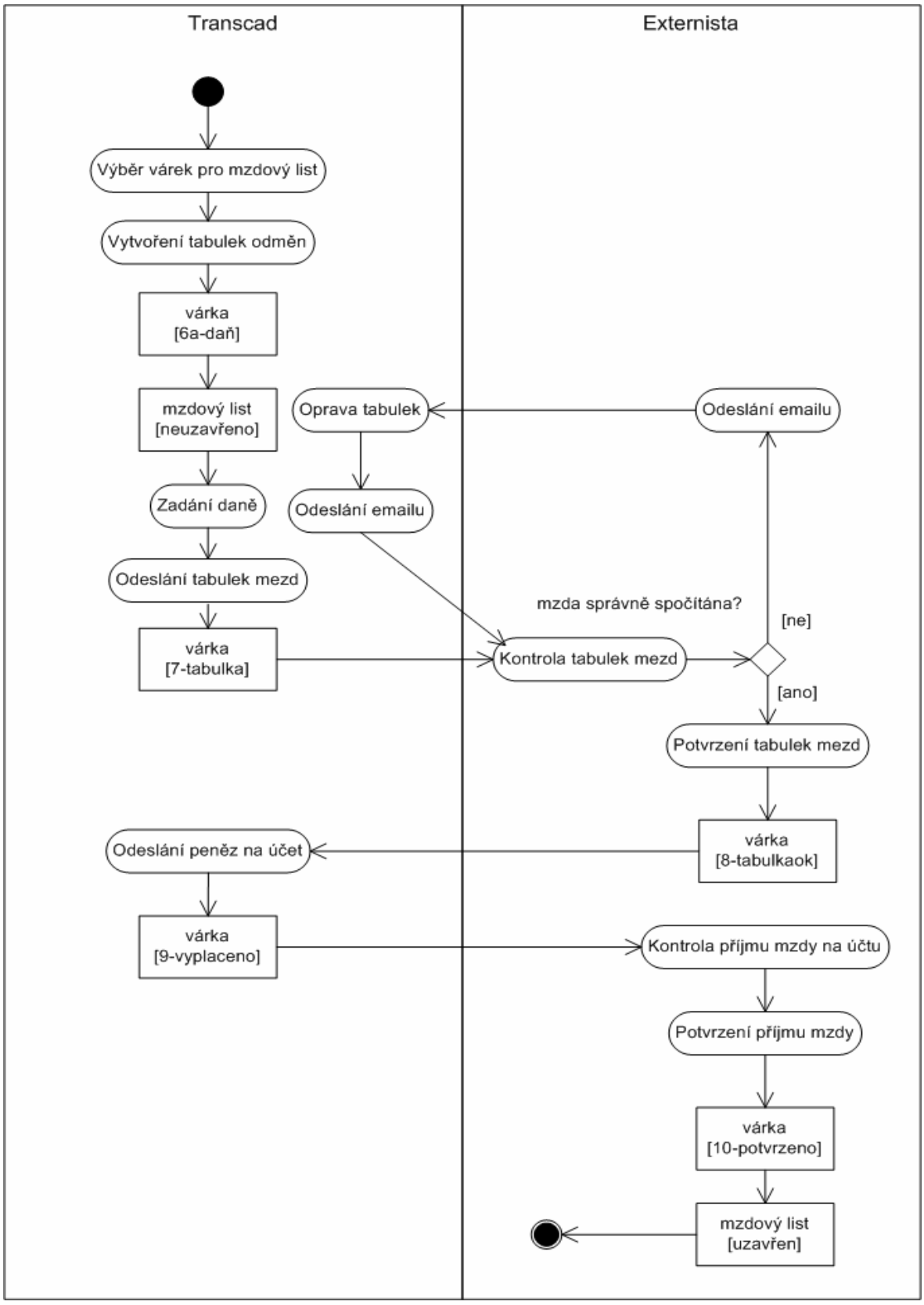
Obr 5.2 Diagram případu užití Práce se systémem – admin



Obr 5.3 Diagram aktivit Řešení dotazů



Obr. 5.4 Diagram aktivit Práce s várkami



Obr. 5.5 Diagram aktivit Správa mzdových listů

6 Fáze Konstrukce

Ve fázi Konstrukce bylo navrženo relační schéma databáze a vytvořeny jednotlivé moduly informačního systému.

6.1 Návrh relačního schématu databáze

Jelikož návrh tohoto informačního systému je z velké části úloha databázového charakteru, bylo správné navržení tabulek a relací mezi nimi klíčovou částí celého projektu. Proto bylo třeba správně provést dekompozici problému, stanovit datové typy pro jednotlivé sloupce tabulek, nalézt vhodné relace a stanovit jejich kardinalitu.

Správně navržená databáze by měla splňovat podmínky některé normální formy. Obecně platí, že čím je tabulka ve vyšší normální formě, tím kvalitněji je tabulka navržena. Stručný přehled normálních forem databází je v tabulce 6.1.

Tab. 6.1 Přehled normálních forem (NF)

Označení normální formy	Definice
0NF	Tabulka je v nulté normální formě právě tehdy, existuje-li alespoň jedno pole, které obsahuje více než jednu hodnotu.
1NF	Tabulka je v první normální formě, jestliže lze do každého pole dosadit pouze jednoduchý datový typ (jsou dále nedělitelné)
2NF	Tabulka je ve druhé normální formě, jestliže je v první a navíc platí, že existuje klíč a všechna neklíčová pole jsou funkcí celého klíče (a tedy ne jen jeho částí)
3NF	Tabulka je ve třetí normální formě, jestliže každý neklíčový atribut není transitivně závislý na žádném klíči schématu neboli je-li ne druhé normální formě a zároveň neexistuje jediná závislost neklíčových sloupců tabulky
BCNF (Boyce-Coddova normální forma)	Tabulka je v Boyce-Coddově normální formě, jestliže pro každou netriviální závislost X->Y platí, že X obsahuje klíč schématu R
4NF	Tabulka je ve čtvrté normální formě, je-li ve třetí a popisuje pouze příčinnou souvislost (jeden fakt)
5NF	Tabulka je v páté normální formě, pokud je ve čtvrté a není možné do ní přidat nový sloupec (skupinu sloupců) tak, aby se vlivem skrytých závislostí rozpadla na několik dílčích tabulek

Výsledkem dekompozice problému je relační schéma databáze, které se nachází v přílohách v části A. Schéma používá notaci Crow foot.

V relačním schématu databáze je možné pozorovat, že tabulka tb_ml, ve které jsou uložena data mzdových listů pracovníků, má nižší normální formu, než by mohla mít, neboť pole příjmení je závislé na neklíčovém poli id_osoby. Při důkladné dekompozici problému by zde sloupec příjmení vůbec nebyl a pole by se načítalo z tabulky tb_uzivatele. V tomto případě bylo však zvoleno toto řešení, a to z toho důvodu, že po odchodu externisty z firmy dojde i k jeho smazání z databáze, avšak mzdové listy pracovníků je třeba uchovávat pro další použití i po jejich odchodu.

Pro vytvoření tabulek a jejich správu byl s výhodou použit phpMyAdmin. phpMyAdmin je nástroj pro správu databází MySQL prostřednictvím webového rozhraní, napsaný v jazyce PHP. Je volně dostupný na adrese *http://www.phpmyadmin.net/*.

V následující části uvádím stručný seznam všech tabulek s jejich atributy. Názvy většiny sloupců jsou natolik popisné, že význam sloupců budu uvádět pouze u těch, u kterých není účel na první pohled zřejmý.

Tabulka tb_uzivatele

Tabulka obsahuje osobní data uživatelů, jejich úroveň pro přístup do systému, heslo a další údaje potřebné pro chod systému.

Sloupec	Typ	Vlastnost	Popis
id_osoby	int unsigned	auto increment	primární klíč
jmeno	varchar(32)	not null	
prijmeni	varchar(32)	not null	
kod_osoby	varchar(8)	not null	firemní kód osoby
adresa_trvaleho_bydliste_ulice	varchar(32)		
adresa_trvaleho_bydliste_mesto	varchar(32)		
adresa_trvaleho_bydliste_psc	varchar(8)		
adresa_postovni_ulice	varchar(32)		
adresa_postovni_mesto	varchar(32)		
adresa_postovni_psc	varchar(32)		
rodinny_stav	enum		
studium_skola	varchar(32)		
studium_obor	varchar(32)		
rodne_cislo	varchar(12)		
cislo_uctu	varchar(32)		
email	varchar(32)		
icq_cislo	tinyint		
telefon_mobil	varchar(32)		
telefon_2	varchar(32)		
poznamky	text		
datum_zapisu	date		datum přijetí ve firmě
ftp_pristup_heslo	varchar(250)		heslo pro přístup do systému
uroven	enum		úroveň práv
stav	enum		
cislo_posledni_varky	smallint	not null	pro AJAX návrh
id_posledni_projekt	int_unsigned		

Tabulka tb_projekty

Obsahuje data týkající se projektů.

Sloupec	Typ	Vlastnost	Popis
id_projektu	int unsigned	auto increment	primární klíč
nazev_projektu	varchar(64)	not null	
info_html_soubor	varchar(128)		odkaz na soubor s informacemi o projektu
stav			
id_barvy			vazba na tb_barvy

Tabulka tb_varky
Obsahuje data týkající se várek.

Sloupec	Typ	Vlastnost	Popis
id_varky	int unsigned	auto increment	primární klíč
cislo_varky	int		
id_osoby	int unsigned	not null	vazba na tb_uzivatele
id_projektu	int unsigned	not_null	vazba na tb_projekty
komentar	text		
pocet_jednotek	float		
pocet_souboru	smallint		
zacatek	date		
konec	date		
pocet_dni	smallint		
chyby	longtext		
komentar	text		
srazka	float		
puvodni_cena	float		
cena	float		
stav	enum		
id_ml	int unsigned		vazba na tb_ml

Tabulka tb_udalosti
Tabulka slouží pro zaznamenávání sledovaných systémových událostí.

Sloupec	Typ	Vlastnost	Popis
id_udalosti	int unsigned	auto increment	primární klíč
typ	enum		
id_osoby	int unsigned		vazba na tb_uzivatele
datum	date		
text	varchar(250)		

Tabulka tb_rel_prj_uziv
Relační tabulka pro vytvoření vazby typu M:N mezi projekty a uživateli.

Sloupec	Typ	Vlastnost	Popis
id_vazby	int unsigned	auto increment	primární klíč
id_projektu	int unsigned		vazba na tb_projekty
id_osoby	int unsigned		vazba na tb_uzivatele

Tabulka tb_dotazy
Tabulka s daty dotazů.

Sloupec	Typ	Vlastnost	Popis
id_dotazu	int unsigned	auto increment	primární klíč
datum	timestamp	current timestamp	
glosar	tinyint		určuje, zda se dotaz exportuje při exportu glosáře, hodnota 0 nebo 1
eng_pojem	text		hledaný pojem
cze_pojem	text		český návrh nebo řešení dotazu
kontext	text		obvykle věta obsahující pojem
komentar	text		komentář k dotazu
odkaz	varchar(128)		odkaz na soubor s dotazem
stav	enum		stav dotazu
id_projektu	int unsigned		vazba na tb_projekty
id_osoby	int unsigned		vazba na tb_uzivatele

Tabulka tb_barvy
Tabulka barev přiřazených projektům pro podbarvení várek v modulu Várky.

Sloupec	Typ	Vlastnost	Popis
id_barvy	int unsigned	auto increment	primární klíč
kod_barvy	varchar(7)		hexadecimální kód
nazev_barvy	varchar(16)		uživatelem zadaný vlastní název barvy

Tabulka tb_planovac
Obsahuje datumy blokovaných nebo rezervovaných dní jednotlivých externistů pro modul Plánovač.

Sloupec	Typ	Vlastnost	Popis
id_zaznamu	int unsigned	auto increment	primární klíč
id_osoby	int unsigned		vazba na tb_uzivatele
datum	date		rezervovaný nebo blokovaný den
typ	enum		typ – blokace nebo rezervace

Tabulka tb_admin
Obsahuje některá nastavení systému.

Sloupec	Typ	Vlastnost	Popis
id_zaznamu	int unsigned	auto increment	primární klíč
puvodni_cena	float	not null	cena za jednotku práce (bez srážky po opravě) zadaná při vytvoření várky.
obnoveni	int	not null	obnovení glosáře v milisekundách
zobrazit_první_stav	tinyint	not null	nastavení pohledu externisty

Tabulka tb_ml
Tabulka obsahující data mzdových listů.

Sloupec	Typ	Vlastnost	Popis
id_ml	int unsigned	auto increment	primární klíč
kod_ml	int		číslo mzdového listu
id_osoby	int unsigned		vazba na tb_uzivatele
prijmeni	varchar(32)		příjmení externisty
dan	float		zadaná daň
datum_odeslani			datum odeslání peněz na účet
uzavreno	tinyint		hodnota 0 nebo 1
celkem_hrubeho	float		hrubá mzda
celkem_cisteho	float		čistá mzda

6.2 Modul Dotazy

Při vytvoření pohledu a modulu Dotazy bylo zejména klíčové vytvoření funkce Rádce, která je realizována pomocí technologie AJAX.

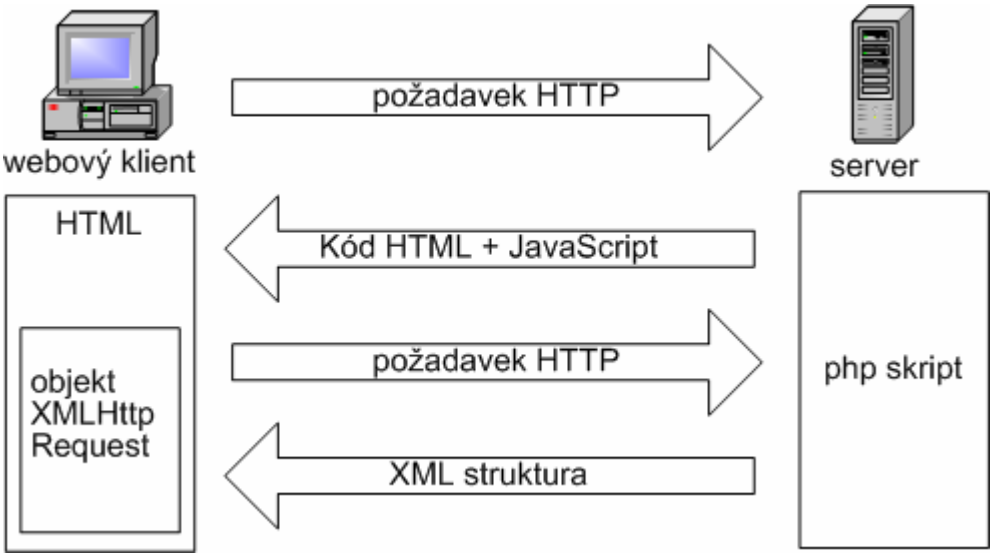
6.2.1 Použití technologie AJAX při tvorbě funkce Rádce

Zkratka AJAX znamená Asynchronous JavaScript and XML. AJAX není sám o sobě implementací technologie či softwarovým produktem. Je to pojem označující současné použití následujících technologií:

- Document Object Model (DOM),
- XMLHttpRequest,
- HTML, CSS,
- JavaScript

Znakem technologie AJAX je zvýšená interaktivita webových stránek, dosažená díky výměně malých objemů dat "na pozadí". Základním stavebním kamenem je objekt XMLHttpRequest, který umožňuje asynchronní volání serveru.V klasickém webovém modelu každá změna stavu na klientovi vyžaduje obnovení celého uživatelského rozhraní. Vše probíhá v pevně dané posloupnosti kroků. Nejdříve je vygenerována žádost o změnu stavu, pak dochází k odeslání požadavku na server, k vyřízení požadavku a vše končí zasláním kompletního uživatelského rozhraní s daty. Jednotlivé kroky jsou vzájemně synchronizovány. Naopak AJAX díky objektu XMLHttpRequest může vyvolat libovolný počet nezávislých požadavků, jejichž

výsledky mohou ovlivnit pouze patřičné části uživatelského rozhraní, bez nutnosti jeho celkového znovunačítání. Schéma použití technologie AJAX je na následujícím obrázku.



Obr.6.1 Použití technologie AJAX

Technologii AJAX využívají například i známé portály Google nebo Yahoo. Právě zmiňovaná schopnost měnit dynamicky obsah části stránky vedla při vytváření tohoto informačního systému k nápadu vytvořit jakýsi "online slovník", který by pracoval nad určenou databází a vyhledával v ní zadaný pojem. Tento slovník byl pak označen pracovním názvem Rádce a je k dispozici v modulu Dotazy v pohledu externisty.

6.2.2 Možné nevýhody použití technologie AJAX

Použití této technologie ve funkci Rádce s sebou bohužel přináší i několik nevýhod.

6.2.2.1 Závislost na JavaScriptu a modelu DOM

Ajax je zcela závislý na JavaScriptu a modelu DOM (Document Object Model). Z tohoto důvodu je nutné mít v prohlížeči zapnutou podporu JavaScriptu, kterou však někteří uživatelé zapínají neradi. Tento problém však nejde nijak obejít, a proto je nutné mít při používání systému zapnutou podporu JavaScriptu.

Dalším problém představuje různá implementace modelu DOM v různých prohlížečích nebo různých verzích stejného prohlížeče. Tento problém je nutné ošetřit více způsoby vytvoření objektu XMLHttpRequest pro jednotlivé prohlížeče, což je demonstrováno na následující části kódu ze souboru *script.js*.

```
function createXmlHttpRequestObject()
{
  // reference na objekt XMLHttpRequest
  var xmlHttp;
  // vytvoření objektu XMLHttpRequest pro Internet Explorer
  if(window.ActiveXObject)
  {
    try
```

```
    {
      xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
    }
    catch (e)
    {
      xmlHttp = false;
    }
  }
  // vytvoření objektu XMLHttpRequest pro Mozilla Firefox, Operu
etc.
  else
  {
    try
    {
      xmlHttp = new XMLHttpRequest();
    }
    catch (e)
    {
      xmlHttp = false;
    }
  }
  // vrácení vytvořeného objektu nebo chybová hláška
  if (!xmlHttp)
    alert("Chyba vytvoreni objektu XMLHttpRequest.");
  else
    return xmlHttp;
}
```

Úplný výčet metod objektu XMLHttpRequest je v tabulce v příloze.

6.2.2.2 Doba odezvy

Při použití technologie AJAX je třeba brát v úvahu dobu odezvy, tedy interval mezi požadavkem a odezvou serveru. Proto bylo třeba funkci Rádce testovat a ověřovat, zda nevznikají velké prodlevy, které by mohly uživatele mátnout. Testování proběhlo na zkušebních bází pojmů získaných z předchozích projektů o velikosti několika set pojmů a doba odezvy byla shledána uspokojivou.

Dalším problém může představovat rendrování stránky, která se při změně obsahu na malý okamžik mění. Právě tato nutnost překreslování spolu se změnami malých částí obrazovky činí dobu odezvy více nápadnou. I v tomto případě se systém choval poměrně uspokojivě. Upozornění na dobu odezvy bude začleněno do dokumentace systému pro budoucí uživatele, aby nebyli tito zmatení.

6.2.3 Jazyk XML a jeho implementace ve funkci Rádce

XML (eXtensible Markup Language, česky rozšiřitelný značkovací jazyk) je obecný značkovací jazyk, který byl vyvinut a standardizován konsorciem W3C. Umožňuje snadné vytváření konkrétních značkovacích jazyků pro různé účely a široké spektrum různých typů dat.

Jazyk je určen především pro výměnu dat mezi aplikacemi a pro publikování dokumentů. Umožňuje popsat strukturu dokumentu z hlediska věcného obsahu jednotlivých částí, nezabývá se sám o sobě vzhledem dokumentu nebo jeho částí. Prezentace dokumentu (vzhled) se potom definuje připojeným stylem. Další možností je pomocí různých stylů provést transformaci do jiného typu dokumentu, nebo do jiné struktury XML.

Jazyk XML má v současné době široké spektrum použití, používají ho například čtečky RSS, protokol pro IM (Instant Messaging) Jabber nebo souborový formát Office Open XML.

6.2.3.1 Syntaxe jazyka XML

Narozdíl např. od jazyka HTML je efektivita XML je silně závislá na struktuře, obsahu a integritě. Aby byl dokument považován za správně strukturovaný, musí mít nejméně následující vlastnosti

- Právě jeden kořenový (root) element.
- Neprázdné elementy musí být ohraničeny startovací a ukončovací značkou. Prázdné elementy mohou být označeny tagem „prázdný element“.
- Všechny hodnoty atributů musí být uzavřeny v uvozovkách – jednoduchých (') nebo dvojitých ("), ale jednoduchá uvozovka musí být uzavřena jednoduchou a dvojitá dvojitou. Opačný pár uvozovek může být použit uvnitř hodnot.
- Elementy mohou být vnořeny, ale nemohou se překrývat; to znamená, že každý (ne kořenový) element musí být celý obsažen v jiném elementu.
- Jména elementů v XML rozlišují malá a velká písmena.

6.2.3.2 Návrh XML struktury používané ve funkci Rádce

S ohledem na pravidla syntaxe jazyka XML uvedená v oddílu 6.2.3.1 byla navržena XML struktura, kterou odesílá skript radce.php po přijetí požadavku HTTP od objektu XMLHttpRequest a která obsahuje data nalezená při prohledávání báze pojmů. Příklad struktury je uveden v následujícím kódu.

```
<?xml version="1.0" encoding="windows-1250" standalone="yes"?>
<response>
      <odpovedi>
            <odpoved>
                  <original>Code Coloring Data</original>
                  <preklad>Data barevného zvýraznění kódu</preklad>
                  <kontext>není</kontext>
                  <komentar>není</komentar>
            </odpoved>
            <odpoved>
                  <original>Code Hints</original>
                  <preklad>Tipy při psaní kódu</preklad>
                  <kontext>není</kontext>
                  <komentar>není</komentar>
            </odpoved>
      </odpovedi>
</response>
```

Uvedenou XML strukturu pak zpracovává následující část kódu jazyka JavaScript:

```
// získání XML obdržené ze serveru
 xmlResponse = xmlHttp.responseXML;

//zachyceni chyb u IE a Opery
 if (!xmlResponse || !xmlResponse.documentElement)
 throw("Neplatna struktura XML:\n"+xmlHttp.Responsetext);
//zachyceni chyb u Firefoxu
```

```
 var rootNodeName=xmlResponse.documentElement.nodeName;
 if (rootNodeName=="parsererror")
 throw("Neplatna struktura XML:\n"+xmlHttp.Responsetext);
// získání elementu dokumentu (kořenový element) XML struktury
 xmlRoot = xmlResponse.documentElement;
 //získání odpovědí original,preklad,kontext,komentar
 originalArray=xmlRoot.getElementsByTagName("original");
 prekladArray=xmlRoot.getElementsByTagName("preklad");
 kontextArray=xmlRoot.getElementsByTagName("kontext");
 komentarArray=xmlRoot.getElementsByTagName("komentar");
```

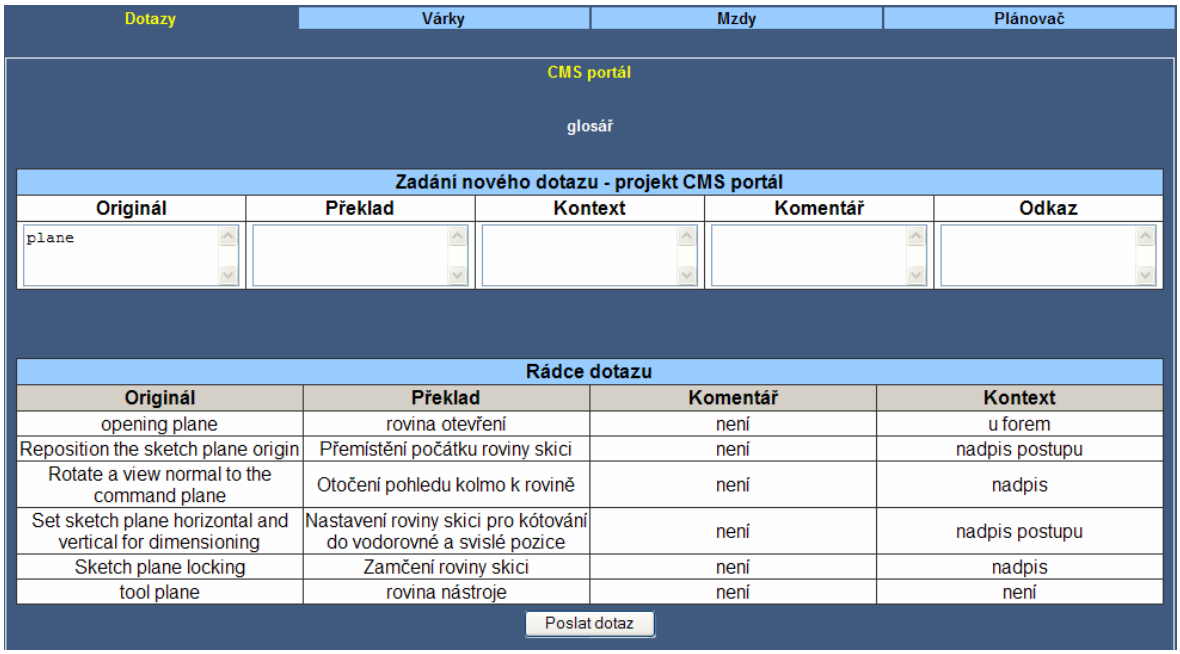
Jak je z kódu patrné, i tentokrát se mezi sebou jednotlivé prohlížeče liší. Zatímco Internet Explorer a Opera používají stejný mechanismus zachycení neplatné struktury, u prohlížeče Mozilla Firefox bylo nutné přikročit k jinému řešení zachycení chyby. Tím je získaná struktura XML načtena do polí, která jsou následně dále zpracovávána, tedy jsou k nim přidány potřebné HTML tagy a jsou společně vykreslena ve webovém prohlížeči jako tabulka.

6.2.4 Pohledy modulu Dotazy

Modul Dotazy nabízí pohledy pro externistu i admina. Nejdříve stručně popíši pohled externisty..

6.2.4.1 Modul Dotazy – pohled externisty

V pohledu externisty je k dispozici lišta, ve která jsou zobrazeny názvy projektů, na které je externista přiřazen. Klepnutím na záložky může externista přepínat mezi projekty. V drtivé většině případů však bude v liště jen jeden projekt. Dále je zde k dispozici odkaz Glosář. Klepnutím na tento odkaz zobrazí externista v novém okně úplný glosář daného projektu. Glosář se automaticky znovu načítá podle hodnoty zadané v modulu Admin.



Obr. 6.2Modul Dotazy – pohled externisty

Dále má externista k dispozici 5 textových polí, Orignál, Překlad, Kontext, Komentář a Odkaz. Zadáním hledaného pojmu do pole Originál může hledat externista pojem v databázi. V tabulce Rádce dotazu se pak zobrazují nalezené výsledky. Pokud není pojem vyřešen, externista zadá další textová pole. Do pole Překlad zadá svůj návrh řešení, do pole Kontext celou větu, ve které se pojem nachází, do pole Komentář případný komentář a do pole Odkaz http adresu souboru, na kterém pracuje, nebo jeho název. V případě, že zadá adresu http, funguje text jako aktivní odkaz. Klepnutím na tlačítko Poslat dotaz se dotaz odešle k dalšímu zpracování Transcadem . Vyřešené dotazy externisty se zobrazí v tabulce Dotazy za posledních 5 dní a samozřejmě se přidají do databáze, takže se při dalším hledání nabídnou všem externistům, a do glosáře. V případě, že byly externistovi vráceny nějaké dotazy, zobrazí se tabulka Vrácené dotazy, kde může externista klepnutím na odkaz Zpracovat přenést dotaz do pracovních textových polí, upravit a znovu odeslat.

6.2.4.2 Modul Dotazy – pohled admina

V pohledu admina jsou také k dispozici záložky s kartami pro přepínání mezi projekty. Pod nimi se nachází odkaz Nové dotazy, který zobrazí tuto stránku, Řešené dotazy, který zobrazí pohled pro export dotazů konzultantům, a Glosář pro zobrazení glosáře.V pohledu glosáře může admin dotazy upravovat a mazat.

Hlavní součástí okna je tabulka pro hromadné řešení dotazů. V tabulce jsou stejná pole jako u externisty, tedy Originál, Překlad, Kontext, Komentář a Odkaz (odkaz tentokrát jako funkční odkaz, aby mohl admin po klepnutí hledat text přímo v daném souboru). Dále pak zatržítko, zda se má dotaz zařadit do glosáře, aby byl k dispozici po exportu glosáře pro další použití. Poslední položkou u každého dotazu je rozbalovací seznam Stav, kde může admin vybrat stav dotazu. Implicitní hodnota je odtc, tedy dotaz se vrátí jako vyřešený externistovi. Dále je možné dotaz přepnout do stavu řeší se1 pro zpracování konzultantům, vrátit externistovi nebo smazat. Klepnutím na tlačítko Další stránka se dotazy na této stránce zpracují a přejde se na dalších 30 dotazů.

V pohledu Řešené dotazy lze dotazy označené zatržítkem Zpracovat buď exportovat pro zaslání konzultantům (a přepnout tak do stavu řeší se), nebo klepnutí na tlačítko Vyřešit řešené odeslat po úpravě zpět externistům.

6.3 Modul Projekty

Modul projekty nabízí pouze pohled admina. Nechybí zde lišta pro navigaci mezi projekty, kde lze navíc zobrazit přehled všech projektů, kde lze projekty mazat, a vytvořit nový projekt. Je zde možné nahrát soubor s informacemi pro externisty. Tento soubor je umístěn na internetu a obsahuje pokyny k překladu pro externisty platné pro tento projekt. Odkaz na tento soubor se vloží k informacím o várkách daného projektu. Dále se zde nachází velmi důležitá funkce Export glosáře, která exportuje glosář do textového souboru s tabulátorem jako oddělovačem. Tento soubor lze pak načíst pro další zpracování v aplikaci Microsoft Excel. Funkce Import glosáře pak načte textový soubor ve stejném formátu (s tabulátorem jako oddělovačem) do databáze do glosáře. Maximální velikost souboru je 10 MB.

Pomocí dalších ovládacích prvků v tomto okně lze měnit stav projektu z aktivního na pasivní, kdy není projekt vidět v navigační liště, měnit barvu projektu a přiřazovat externisty k projektu.

6.4 Modul Uživatelé

Pohled modulu Uživatelé je dostupný pouze pro admina. Slouží podobně jako pohled projekty k administraci uživatelů. Lze zde přidávat, odstraňovat a upravovat uživatele. Uživatel ve stavu Neaktivní se zobrazí pouze v přehledu všech uživatelů. Přístupové heslo zde lze zadat, ale zadané heslo se nezobrazí, neboť je nelze zpětně dešifrovat z uloženého haše.

6.5 Modul Várky

Pohledy modulu Várky jsou dostupné pro externistu i admina. Externista může v pohledu Várky získávat informace o svých várkách a pomocí tlačítek generovaných podle stavu může oznamovat stažení várky z FTP a nahrání várky na FTP. Tato dynamicky generovaná tlačítka se zobrazí, pokud se várka nachází ve stavu 2 nebo 3. Tlačítko přepne stav várky a zároveň odešle Transcadu email s upozorněním. Světle zeleně jsou podbarveny várky, u kterých leží aktuální datum v rozsahu začátek várky – konec várky, tedy várky, na kterých by měl externista pracovat.

Pohled Várky pro admina umožňuje kompletní správu várek. V horní části pohledu se nachází filtr a odkaz na pohled pro vytvoření várky. Pod rozbalovacím seznamem měnícím stav várky se u některých stavů nachází dynamická tlačítka umožňující práci s várkami, která slouží pro intuitivnější ovládání várek. Pomocí těchto tlačítek lze oznámit odeslání várky na FTP, přijetí várky a opravení várky. Klepnutím na kód externisty, číslo várky a název projektu lze upravit data související s těmito položkami. Ve stavu 6-opraveno se aktivuje odkaz ve sloupci chyby, kde lze zadat srážku a chyby.

6.6 Modul Mzdy

Modul Mzdy také nabízí pohledy pro externistu i admina. V pohledu externisty je možné zjistit informace o proplacení opravených várek, potvrdit tabulky mezd, existují-li nějaké, a nakonec potvrdit přijetí mzdy. V pohledu admina nabízí tento pohled další tři karty:

- **Várky k proplacení**. Na této kartě může admin vybrat opravené várky pro vytvoření mzdového listu. Pro výběr jsou dostupné várky se stavem 6-opraveno. Lze vybrat zároveň várky od více uživatelů, nebo přidat várky k existujícím mzdovým listům
- **Tabulky odměn**. Na této kartě může admin zadat daň a odeslat vytvořenou mzdovou tabulku externistovi a po jejich pak oznámit vyplacení
- **Přehled vyplacených várek**. Na této kartě se nachází přehled mzdových listů a mzdové listy se zde dají exportovat.

6.7 Modul Plánovač

Modul Plánovač je dostupný v pohledu admina i externisty. V pohledu externisty se zobrazí kalendář tohoto a 3 dalších měsíců a externista zde může pomocí zatržítek označit dny, kdy bude nedostupný. V pohledu admina je možné prohlédnout si rezervované dny u všech externistů a spočítat, kdy bude splněn určitý počet pracovních jednotek. Na kartách jednotlivých externistů je dále možné provést jejich blokaci – označit dny, kdy se s externistou počítá na nějaký projekt. Blokované dny nemůže externista rezervovat a mění výpočet pracovních jednotek.

6.8 Modul Události

V modulu události může admin sledovat vybrané události v systému, zejména události iniciované ze strany externistů. Dále je zde možné hromadné odesílání emailů externistům.

6.9 Modul Admin

Modul Admin má záložky karet pojmenované podle modulů. Na jednotlivých kartách se nachází některé volby nastavení. Lze zde například přidat novou barvu projektu, nastavit obnovování glosáře atd. Také se zde nachází nastavení samotného účtu admina. Na email účtu admina chodí emaily s upozorněním na události v systému.

7 Fáze Zavedení

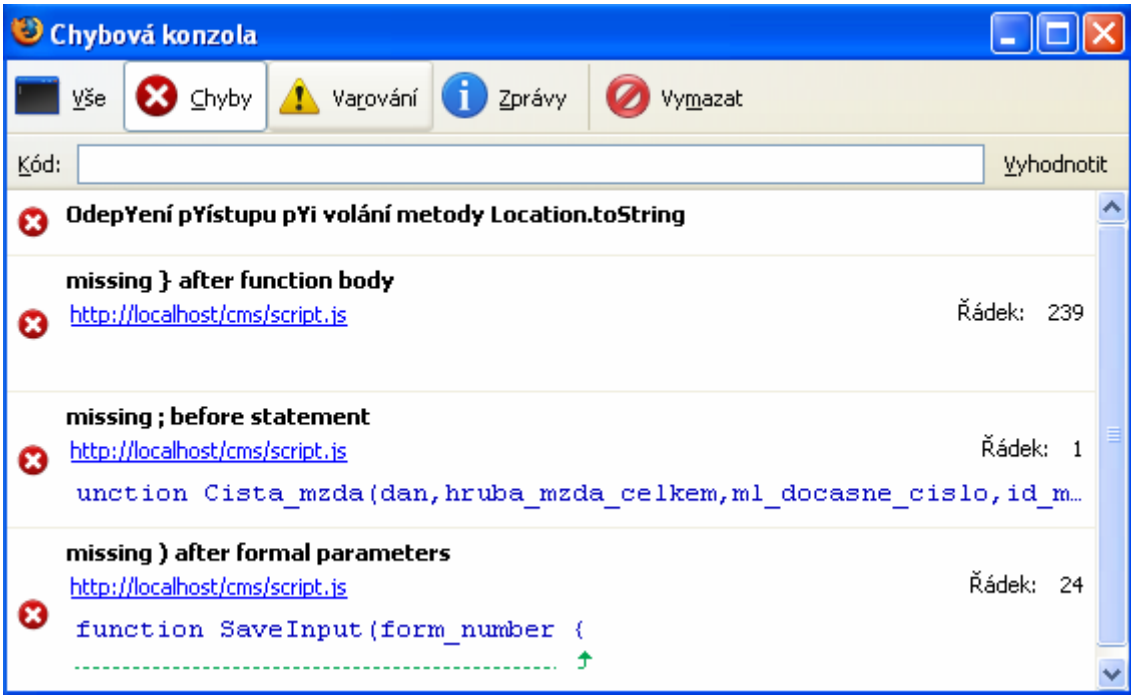
7.1 Ladění

7.1.1 Ladění skriptů PHP

Jednou z největších nevýhod jazyka PHP je, že ve standardní distribuci chybí ladící (debugovací) nástroj. Jelikož mám standardní distribuci, při ladění nezbývalo než podle čísel řádků v chybových hlášeních hledat v aplikaci PSPad daný řádek a podle chybového hlášení se snažit chybu odladit. Nicméně i pro středně zkušeného PHP programátora je tato metoda dostatečně efektivní a rychlou.

7.1.2 Ladění skriptů AJAX

Jak jsem se zmínil v části 6.2.2, technologie AJAX je závislá na implementaci modelu DOM daného prohlížeče a stejně se i liší chování prohlížečů v případě chyby v souboru javascriptu. Zatímco Internet Explorer 8 zobrazí chybovou ikonu ve stavovém řádku a varovný dialog s číslem řádku, na kterém došlo k chybě , prohlížeče Opera 9.2 a Mozilla Firefox chybu neoznámí, oba však nabízejí velmi podobné chybové konzoly, kde lze chyby v javascriptu sledovat a nalézt. Osobně jsem pro ladění skriptů AJAX používal konzolu prohlížeče Firefox, která mi přišla o něco více uživatelsky příjemná.



Obr. 7.1 Chybová konzola prohlížeče Mozilla Firefox

7.2 Zabezpečení

7.2.1 Možná bezpečnostní rizika

Mezi nejběžnější bezpečnostní rizika PHP aplikací srovnatelných s tímto systémem patří zejména:

- Používání globálních proměnných ve vlastních PHP aplikacích

- Nedostatky v zabezpečení serverů, na kterých jsou aplikace umístěny a provozovány.
- Nulová nebo téměř žádná ochrana proti podvržení falešného SQL dotazu (SQL injection)
- Nízké zabezpečení citlivých údajů, například hesel.
- Umožnění formátování vkládaného textu pomocí značek HTML.

7.2.2 Aplikované prvky zabezpečení

V informačním systému jsou aplikovány následující prvky zabezpečení:

- nepoužívají se globální proměnné (direktiva REGISTER GLOBALS v souboru PHP.INI se nastaví na hodnotu OFF).
- informační systém je umístěn na zabezpečeném serveru
- po jistém čase nečinnosti dojde vždy k automatickému odhlášení přihlášeného uživatele (ukončení session).
- ochrana proti podvržení falešného SQL dotazu (parametry předávané pomocí proměnných, zejména metodou GET, jsou zpracovány pomocí PHP funkce mysql_escape_string.
- hesla jsou ukládána do databáze jako otisk (hash) s využitím kryptografické technologie MD5 (dochází k jednosměrnému šifrování řetězců hesel, klient posílá na server pouze otisk hesla , který se porovnává s otiskem, uloženým na serveru. Z otisku se nedá algoritmem dopočítat zasílaný řetězec).

7.3 Další budoucnost systému

Při tvorbě tohoto systému jsem neměl ještě dostatečné zkušeností jak s tvorbou rozsáhlých webových projektů pomocí jazyka HTML a PHP, tak celkově s tvorbou systémů. V několika následujících bodech se pokusím navrhnout některé možné koncepce zlepšení funkčnosti systému.

7.3.1 Použití objektového přístupu programování

Objektové programování má obecně známé výhody, tedy dědičnost, zapouzdření a polymorfismus. V PHP 5 je již vylepšena podpora objektů, i když samozřejmě nedosahuje úrovně například jazyka C++. U objektového programování je důležitá analýza a navržení samotných objektů, jejich vlastní implementace již představuje relativně jednodušší část projektu. Původně jsem se u tohoto systému rozhodl pro klasický procedurální přístup z důvodů jeho malého rozsahu, postupem času však stále zřetelněji vyvstávala potřeba objektového přístupu. Ten by měl i tu výhodu, že vytvořené třídy by bylo po malých úpravách (nebo s využitím dědičnosti) možné opakovaně používat i v dalších projektech. Rozhodně představuje přechod k objektovému programování jeden z výrazných a nutných směrů vývoje systému.

7.3.2 Použití modulu mod_rewrite serveru Apache

Modul mod_rewrite nabízí stroj pro přepisování adres URL za běhu. Stroj využívá parser založený na regulárních výrazech. S touto technologií jsem se seznámil při práci ve firmě zabývající se profesionální tvorbou webů a hostingem. Bohužel to bylo až ve fázi projektu, kdy byla celková změna koncepce nemožná z časových důvodů. V budoucnosti je však použití této technologie v systému rozhodně potřebné a přínosné.

7.3.3 Zvýšení zabezpečení

Při zabezpečení systému byly použity jen některé základní koncepce. Při dalším vývoji systému je nezbytně nutné věnovat zvýšenou pozornost tomuto aspektu, ať už se jedná o lepší zabezpečení javascriptů nebo o zabezpečení session.

7.3.4 Větší kontrola vstupů a větší testování chování systému

Systém byl testován ve stavu provozu za očekávaných podmínek lidmi, kteří jej sami navrhovali. Předpokládalo se, že jeho instalaci bude provádět odborník. Není tedy dostatečně ošetřeno chování například za situace, že nejsou kompletně zadány projekty, uživatelé atd. Dále nedošlo k testování systému v ostrém provozu. Je tedy téměř jisté, že systém ještě obsahuje chyby, které je třeba odladit.

7.3.5 Vypracování uživatelské dokumentace

Dalším nutným krokem je vypracování dokumentace systému pro jeho uživatele.

7.4 Instalace systému

Funkčnost systému je garantována pro verze PHP, Apache a MySQL uvedené v kapitole 4. Všechny produkty jsou dostupné ke stažení na uvedených adresách. Při jejich instalaci je potřeba postupovat podle pokynů uvedených na stránkách produktů. Další možností je stáhnout kompletní samoinstalační distribuce, například PHP Triad, WampServer nebo EasyPHP, které obsahují všechny tři produkty. Nevýhodou distribucí je obtížnost (u některých distribucí nemožnost) aktualizace jednotlivých částí.

Pro zprovoznění samotného IS je potřeba zkopírovat soubory aplikace php do kořenového adresáře serveru. Pak je třeba v souboru /connect/connect.php editovat údaje pro připojení k databázi, tedy přihlašovací jméno a heslo a jméno databáze a vložit vlastní jméno a heslo pro přihlášení k databázi MySQL. Důležité je nastavit v databázi i souboru connect.php stejné kódování pro zobrazení českých znaků, například cp1250_czech_cs, které je momentálně v souboru connect.php použito. Následně se ve stejném adresáři zavolá skript database_create.php, který vytvoří v databázi tabulky. Nebo je možné databázi importovat pomocí textového souboru database.sql, který se také nachází v adresáři connect.

V adresáři /glosar se nachází soubor glosar.txt. Tento soubor je ukázkou skutečného glosáře pro import.

8 Závěr

Cílem této diplomové práce bylo vytvořit pro překladatelskou agenturu specializovaný informační systém zefektivňující některé procesy ve firmě. Podařilo se vytvořit podle dodaných požadavků funkční systém, a to pomocí zdarma dostupných technologií. Již v současnosti však vyvstaly další nezbytné kroky vývoje systému, které jsem uvedl v kapitole 7. V době dokončení této práce byl systém testován pouze se společníky z firmy Transcad a nebyl dosud nasazen v reálném provozu. Nelze tedy zatím zkoumat skutečnou míru zvýšení efektivity práce a přínos tohoto systému pro firmu. Nicméně se tvorbě systému plně postupovalo podle dodaných požadavků. Instalace systému je popsána v předchozí části.

V celé této práci nikde neuvádím žádný časový plán ani kontrolu jeho dodržování. Je to proto, že původní časový plán se z různých důvodů zcela rozpadl. Mezi hlavní důvody patří nárůst zkušeností a postupné objevovaní nových technologií, například technologie AJAX, a s nimi souvisejících skokových přechodů a celkových změn koncepce. Dalším důvodem byla i neujasněnost přesných požadavků a jejich přidávání nebo úpravy v průběhu projektu. Nakonec byl projekt pojat spíše jako průzkum možných technologií a různých koncepcí systému, test jejich aplikace v praxi a vytvoření funkčního základu pro další vývoj. Při realizaci zakázky na komerční systém by bylo nutné věnovat této oblasti zvýšenou pozornost a pravděpodobně hlouběji prostudovat metodiku řízení projektů nebo angažovat odborníka na tuto oblast.

Při tvorbě systému byla použita metodika Unified Process a jazyk UML. Oba tyto nástroje se ukázaly jako velmi silné, a to přesto, že byla využita pouze malá část jejich možností. I v tomto malém projektu by bylo vhodné použít například diagramy aktivit daleko masivněji. Diagramy případů užití a diagramy aktivit se ukázaly být vynikajícím prostředkem pro jednoduché, efektivní a přehledné zobrazování funkcí systému. Stejně tak metodika UP poskytuje dobré vodítko pro efektivní vytváření systémů. I když má tento systém zdánlivě malý rozsah a oba nástroje jsou určeny spíše pro tvorbu rozsáhlých a objektově orientovaných systémů, jejich použití bylo i v tomto případě nesporně velkým přínosem.

Literatura

[1] ARLOW, Jim.; NEUSTADT, Ila.; *UML2 a unifikovaný proces vývoje aplikací*. Brno : Computer Press, 2007. ISBN 978-80-251-1503-9.

[2] NARAMORE, E.; GERNER, J.; LE SCOUARNEC, Y.; STOLZ, J;. GLASS, M.K. *PHP5, MySQL, Apache : Vytváříme webové aplikace*.1.vydání Brno : Computer Press, 2006. ISBN 80-251-1073-7.

[3] DARIE, C.; BRINZAREA, B.; CHERECHES-TOSA, F.; BUCICA, M; *AJAX a PHP : Tvoříme interaktivní webové aplikace*.1.vydání Brno : Zoner Press, 2006. ISBN 80-86815-47-1.

[4] KOSEK, Jiří. *PHP : Tvorba interaktivních internetových aplikací*.1.vydání Praha : Grada Publishing, 1998. 492 s. ISBN 80-7169-373-1.

[5] BASL, Josef; BLAŽÍČEK, Roman. *Podnikové informační systémy : Podnik v informační společnosti*. 2.vydání Praha : Grada Publishing, 2008. ISBN 978-80-247-2279-5.

[6] KOSEK, Jiří; *HTML : Tvorba dokonalých webových stránek*.1.vydání Praha : Grada Publishing, 1998. ISBN 80-7169-608-0.

[7] CASTRO, Elisabeth; *HTML, XHTML a CSS : Názorný průvodce tvorbou www stránek*.1.vydání Brno : Computer Press, 2007. ISBN 978-80-251-1531-2.

[8] FLANAGAN, David; *JavaScript : kapesní příručka*. Helion S.A, 2004. ISBN 83-7361-466-4.

[9] ŠIMŮNEK, Milan; *SQL : kompletní kapesní průvodce*. 1. vydání Praha : Grada Publishing, 1999. 248 s. ISBN 80-7169-692-7.

[10] HEROUT, Pavel; *Učebnice jazyka C*. 4.vydání České Budějovice : Kopp, 2005. ISBN 80-7232-20-6.

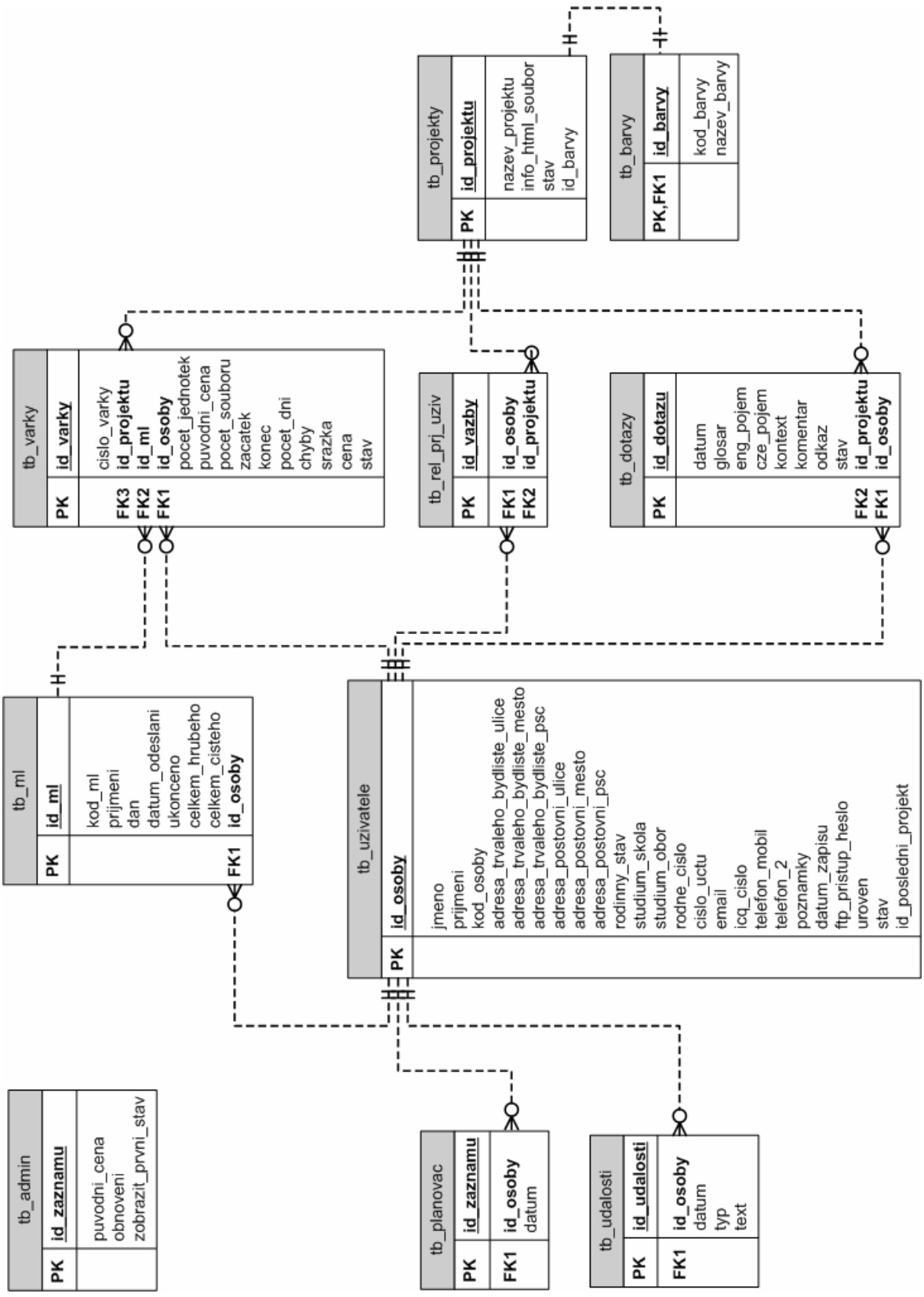
[11] PHP: Hypertext Preprocessor. Dokumentace k PHP [online].
Dostupné z: <http://www.php.net/>

[12] MySQL :: MySQL 5.0 Reference Manual. Dokumentace k MySQL [online].
Dostupné z: <http://dev.mysql.com/doc/refman/5.0/en/>

[13] World Wide Web Consortium. Stránky konsorcia World Wide Web [online].
Dostupné z: <http://www.w3.org/>

[14] Wikipedia. Online encyklopedie [online].[cit. 15.5.2008].
Dostupné z: <www.wikipedia.org/>.

A Relační schéma databáze



B Metody a vlastnosti objektu XMLHttpRequest

Metoda/vlastnost	Popis
abort()	Ukončí aktuální požadavek.
getAllResponseHeaders()	Vrátí hlavičky odpovědí jako řetězec.
getResponseHeader("headerLabel")	Vrátí jednu hlavičku odpovědi jako řetězec.
open("method", "URL"[.aszncFlag[,"username"[,password]]])	Inicializuje parametry požadavku.
send(content)	Provede požadavek http.
setRequestHeader("label","value")	Nastaví dvojici label/value hlavičky požadavku.
onreadystatechange	Používá se pro funkci zpětného volání, která ovládá změny požadavku.
readystate	Vrátí stav požadavku: 0=neinicializovaný 1=zavádí se 2=je zaveden 3=přechodný 4=kompletní
responseText	Vrátí odpověď serveru jako řetězec.
responseXML	Vrátí odpověď serveru jako
Status	Vrátí stavový kód požadavku.
statusText	Vátí zprávu stavu požadavku.