TECHNICKÁ UNIVERZITA V LIBERCI

DIPLOMOVÁ PRÁCE

Hypertextový studijní materiál
předmětů Teorie řízení

Liberec 2002 Aleš Kyral

Prohlášení

Byl jsem seznámen s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 o právu autorském, zejména §60 (školní dílo).

Beru na vědomí, že TU Liberec má právo na uzavření licenční smlouvy o užití mé DP a prohlašuji, že **s o u h l a s í m** s případným užitím mé diplomové práce (prodej, zapůjčení apod.).

Jsem si vědom toho, že užít své diplomové práce či poskytnout licenci k jejímu využití mohu jen se souhlasem TU Liberec, která má právo ode mne požadovat přiměřený příspěvek na úhradu nákladů, vynaložených univerzitou na vytvoření díla (až do jejich skutečné výše).

Diplomovou práci jsme vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím diplomové práce a konzultantem.

Datum

Podpis

PODĚKOVÁNÍ

Děkuji panu Doc. Ing. Osvaldu Modrlákovi CSc. za ochotnou pomoc a cenné rady při přípravě mnoha materiálů. Děkuji také panu Ing. Schauerovi z pedagogické fakulty za posouzení celé koncepce DP a pomoc ve výběru odborné literatury.

ANOTACE

Diplomová práce by měla vytvořit pro pedagogy a studenty TU Liberec dokumenty zahrnující podklady pro výuku a samostudium předmětů z teorie řízení. Za tímto účelem byly vytvořen nový systém, který slouží k zadávání prací a zejména k lepšímu pochopení probírané látky. Celek je svázán mezi sebou hypertextovými odkazy a je postaven na prostředcích sítě Internet.

ABSTRACT

A Diploma thesis should create documents for pedagogues and students of TU Liberec including basis for teaching and self-study subjects from Theory of control. For this purpose there should be created a new system, which serves to placing works and especially to better understanding of actualy teaching theme. All is tied up together by hypertext links and it is built on the possibilities of an Internet network.

SEZNAM ZKRATEK

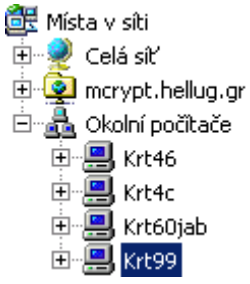
ARPANet	Advanced Research Projects Agency Network – předchůdce Internetu
ASCII	American Standard Code for Information Interchange
cache	druh vyrovnávací paměti počítače
CERN	evropská výzkumná laboratoř pro jadernou fyziku
CGI	Common Gateway Interface
CSS	Cascading Style Sheets
DNS	Domain Name Server
doména	jméno serveru
E-mail	elektronická pošta
FTP	File Transfer Protocol
FAQ	Frequently Asked Questions
Firewall	bariéra sítě starající se o znepřístupnění sítě pro cizí uživatele
GIF	Graphic Interchange Format
HTML	Hyper Text Markup Language
HTTP	Hypertext Transfer Protocol
HW	Hadrware
chat	setkání uživatelů na síti, výměna zpráv probíhá pomocí klávesnice
IP	číslo síťové karty
JPEG	Joint Photographs Expert Group
Klient	kombinace HW a SW, na kterou se dotazuje server
MWS	Matlab Web Server
PDF	Portable Document Format
PHP	Personal Home Page
POP	Point of Presence
Server	kombinace HW a SW, která dává k dispozici data
SQL	Structured Query Language
SMPT	Simple Mail Transfer Protocol
SW	Software
TCP/IP	Transmission Control Protocol / Internet Protocol
URL	Universal Ressource Locator
WWW	World Wide Web
XML	eXtensible Markup Language

Obsah

ÚVOD

Podnětem pro zadání této diplomové práce bylo zefektivnit a zkvalitnit výuku předmětů z teorie řízení. Na jejich tvorbě se začalo pracovat již minulý školní rok, což bylo i součástí mého projektu na který nyní navazuji. Pro zvýšení kvality výuky byl vytvořen kompletní program cvičení a přednášek, zautomatizovaný systém zadávání semestrálních prací a návody k laboratorním úlohám. Efektivitu zvyšujeme zavedením vizualizace do výuky, měřením úloh ze vzdáleného počítače, téměř bezpracným testováním výsledků na Matlab WEB Serveru a vytvořením autotestů (kvizů) z úzkého výřezu látky. Na tomto širokém projektu se podílí větší či menší měrou i další diplomové práce, ze kterých jsou převzaty především jejich výsledy.

K vytvoření celku jsou zapotřebí tři části: hardware, software a pedagogické zkušenosti. Body byly naplněny spuštěním nového serveru (HW), který slouží pro umístění dynamických stránek s potřebným software (PHP a MySQL) a náplní koncipovanou podle pedagogických požadavků. Tím je celek uzavřen. Hlavní pilíř tvoří studijní materiály. Jsou vytvořené ve formátu *pdf* (zvoleno zejména z důvodu daleko vyšších možností zabezpečení autorských práv) vybavených záložkami pro rychlou orientaci a pohyb v textu a aktivními odkazy, směrujícími buď na příbuzné podklady, soubory se zdrojovým programem nebo případné hotové demonstrace zprovozněné díky Matlab WEB Serveru. Vše ostatní je napsáno v jazyku HTML (někdy podporou PHP nebo MySQL podle potřeby). Jedná se o programy přednášek a cvičení, autotesty, některé návody, vstupní/výstupní stránky Matlab WEB Serveru, zadání semestrálních prací a chat.



umístění nového serveru v síti

Internetová adresa:	http://krt32.krt.vslib.cz/
IP:	147.230.128.46
Hardware:	Procesor: AMD Athlon, 130 544 kB RAM
Software:	Windows 2000, Apache WEB Server, PHP, MySQL Server, Matlab 5.3

základní parametry nově vzniklého serveru (HW & SW)

1. VYUŽITÍ POČÍTAČŮ PŘI VÝUCE

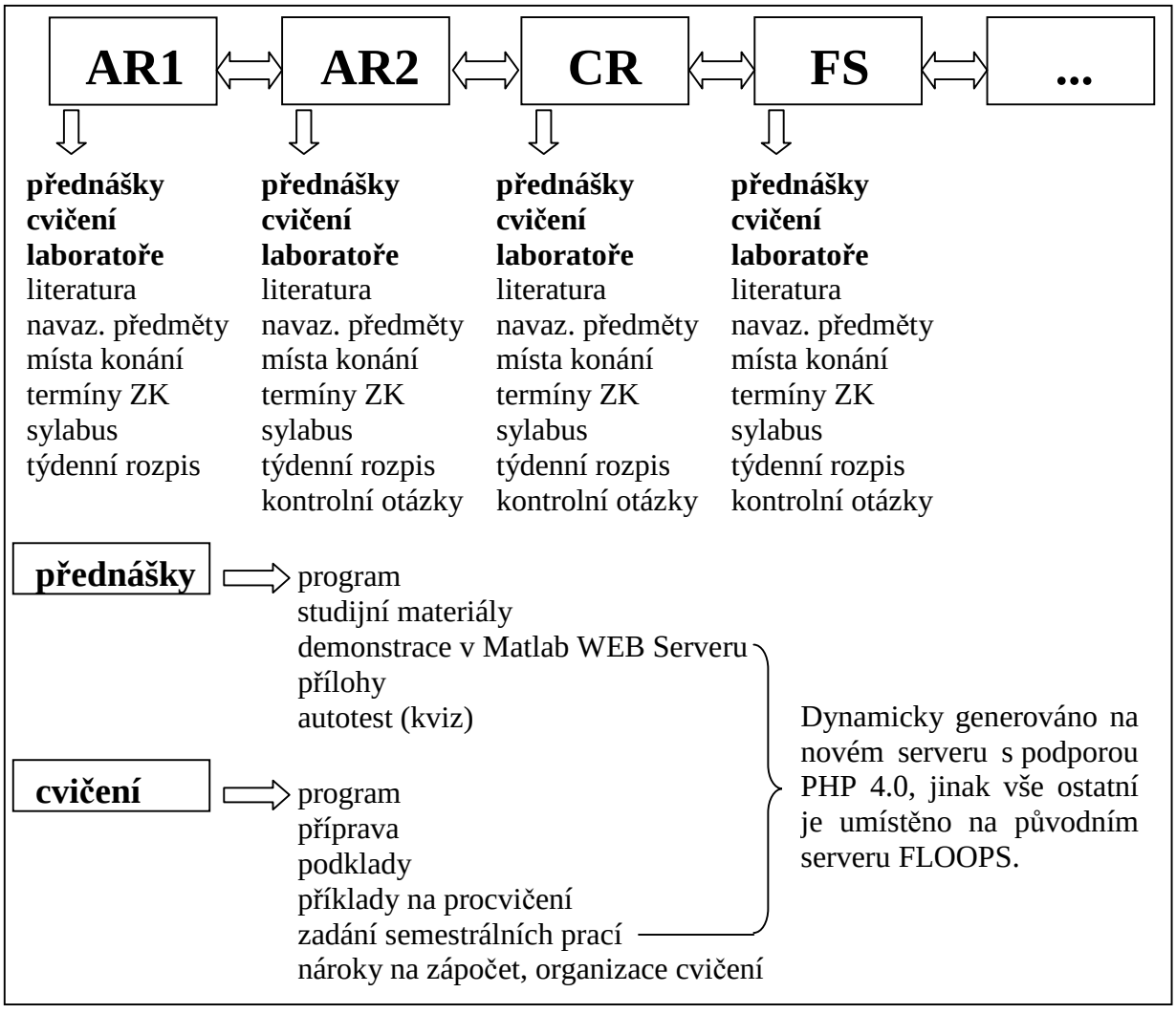
Pro úspěšné začlenění počítače do výuky je nezbytně nutné studovat, jak jej co nejúčinněji používat pro dosažení nejlepších výsledků. Mohlo by se nakonec stát, že bychom místo získání lepších výsledků obdrželi naopak horší. Tato diplomová práce je sice určena spíše pro vytvoření studijních materiálů, ale v některých částech zasahuje přímo do výuky (testování znalostí na Matlab WEB Serveru, přímá měření z WWW prohlížeče, zadávání semestrálních prací atd.) a tudíž je nutné tyto souvislosti brát v potaz.

Podle [2] by bylo pro studenty velmi neefektivní učit se pouze s využitím počítače, ať už by se jednalo o pouhé texty nebo propracované multimediální programy. Základem vysokoškolské výuky musí být stále přímý kontakt studenta s přednášející, popř. cvičí osobou. Počítač přiměřeného výkonu můžeme o hodině naopak použít např. pro demonstraci na modelu. (Jiná situace ovšem je, jedná-li se o výuku předmětů, kde musí být počítač nevyhnutelně přítomen, jak tomu je u programovacích jazyků.) Dobrých výsledků lze dosáhnout, je-li použit jako zdroj informací pro zopakování a prohloubení problematiky. Účinnost lze zvýšit začleněním prvku hry. Vše se pak stává poutavější a student má před sebou momentálně větší motivaci učení, než by pro něj byla závěrečná zkouška. Tento fakt se nezměnil od doby Jana Ámose Komenského – „škola hrou“.

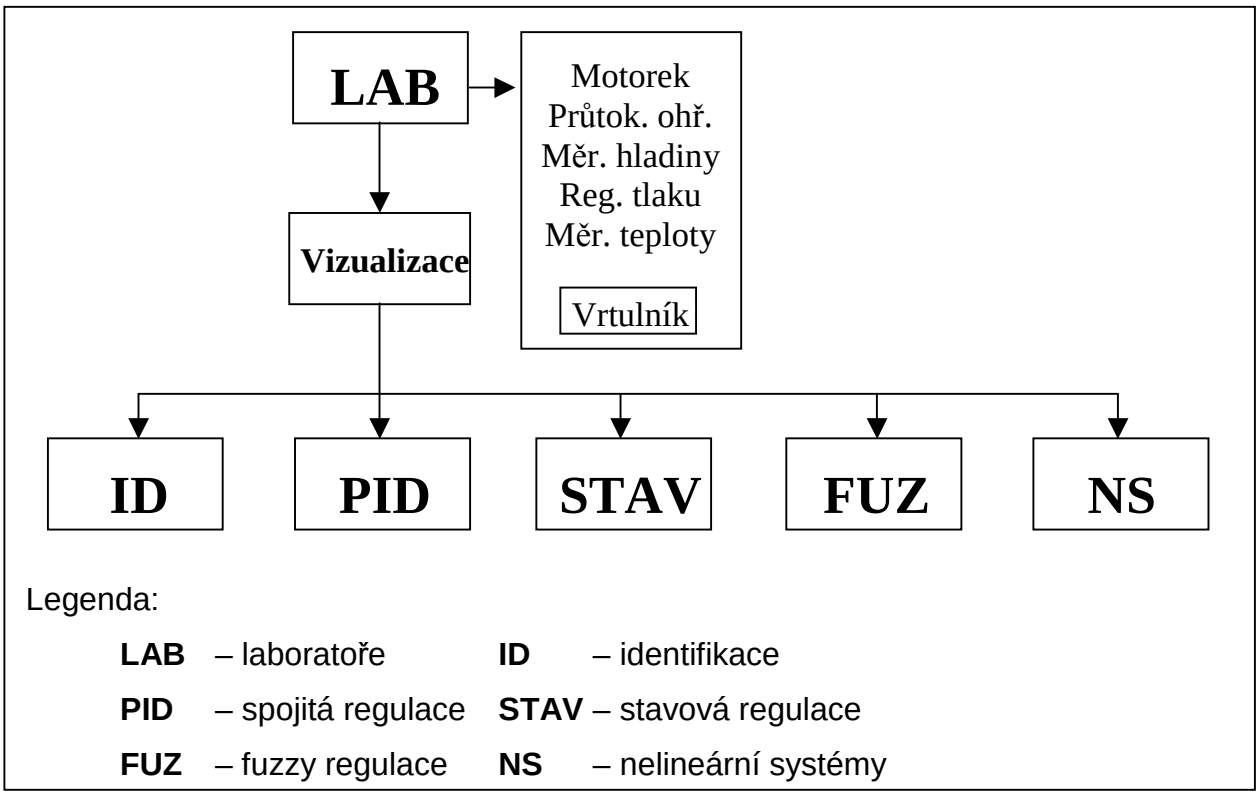
Celková koncepce mé diplomové práce je založena na hypertextových dokumentech. Jedná se o takové materiály, kde není obsah pouze jednolitý od začátku až do konce, ale můžeme se v něm kliknutím na vyznačený text či oblast dostat na odlišnou stránku, kde se uživatel doví více nebo se pouze posuneme ve stávajícím textu na jiné místo. Jeho základy položil Bush Vanenar objevením principu nelineárně strukturovaného textu a o 20 let později v roce 1965 na něj navázal Ted Nelson hypertextem [1]. Jeho princip můžeme znát z internetu, neboť je to hlavní nástroj pro přecházení na jiné stránky z vůle uživatele. V tomto případě je implicitně použito modrého podtrženého textu a po jeho přejetí kurzorem myši se šipka změní na ruku. Příkladem může být: http://www.vslib.cz/. Na tvůrci je pak vymyslet nejlepší propojení mezi dokumenty, především dle kritéria přehlednosti a jednoduchosti orientace v textu.

2. STRUKTURA VYTVOŘENÝCH HYPERTEXTOVÝCH DOKUMENTŮ

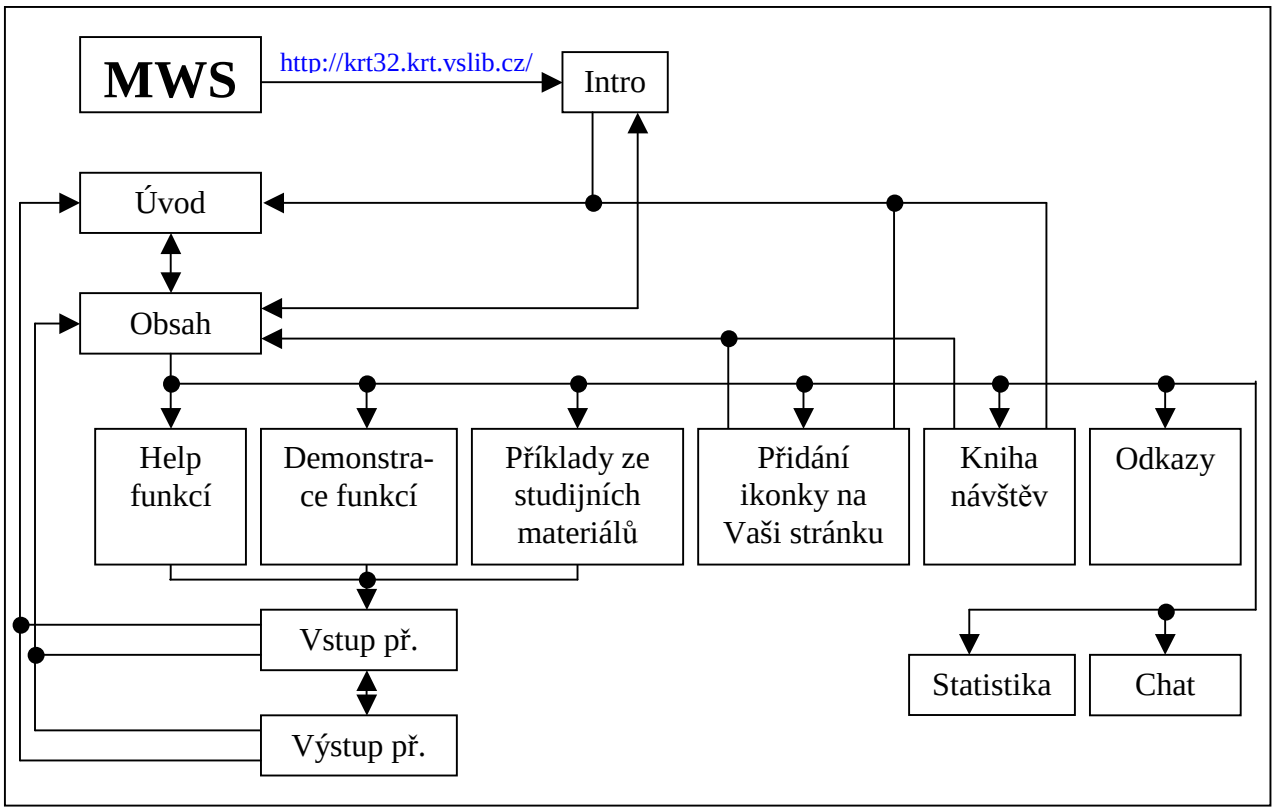
Následující tab.2-1a,b,c demonstruje vytvořenou strukturu, ke které jsme se snažili přiblížit. AR1, AR2, CR jsou zkratky vyučovaných předmětů a FS je zkratka Fakulty strojní. Tečky symbolizují možné další návaznosti.



Tab.2-1a *Struktura hypertextových dokumentů (celkové schéma)*



Tab.2-1b *Struktura hypertextových dokumentů (laboratoře)*



Tab.2-1c *Struktura hypertextových odkazů (Matlab WEB Server)*

2.1 TEORIE AUTOMATICKÉHO ŘÍZENÍ I (AR1)

Dle tab.2-1 je zřejmé, že struktura předmětu AR1 je stejná jako u ostatních s jedinou vyjímkou – chybí zde kontrolní otázky, neboť zkouška je dvousemestrová a tudíž jsou uvedeny až u AR2.

oficiální stránka předmětu:
http://www.fm.vslib.cz/~krt/krt_cz/vyuka/fm/tr1/krt_tr16.htm
studijní materiály předmětu:
http://www.fm.vslib.cz/~krtsub/fm/tr1/materialy.htm

2.2 TEORIE AUTOMATICKÉHO ŘÍZENÍ II (AR2)

oficiální stránka předmětu:
http://www.fm.vslib.cz/~krt/krt_cz/vyuka/fm/tr2/krt_tr28.htm
studijní materiály předmětu:
http://www.fm.vslib.cz/~krtsub/fm/tr2/materialy.htm

2.3 ČÍSLICOVÉ ŘÍZENÍ (CR)

Tento předmět je na rozdíl od předcházejících volitelný a prohlubuje zejména poznatky získané v AR2. Proto v něm můžeme použít i jeho studijní materiály.

oficiální stránka předmětu:
http://www.fm.vslib.cz/~krt/krt_cz/vyuka/fm/cir/krt_cir8.htm
studijní materiály předmětu:
http://www.fm.vslib.cz/~krtsub/fm/cir/materialy.htm

3. INSTALACE

Nevyhnutelnou součástí zprovoznění systému je nainstalování vlastního software. V případě zprovoznění serveru již nejde o triviální záležitost jako u jiných programů, kde celá instalace spočívá ve výběru místa na disku a pokračováním v instalaci tlačítkem `Next`. Nutnou podmínkou pro provoz Matlab WEB Serveru danou výrobcem, je použití operačního systému Windows NT nebo jako v našem případě Windows 2000.

3.1 INSTALACE APACHE HTTPD SERVERU

Server Apache je volně šiřitelný a používá se zhruba v 60% případech (viz. [5]). Instalační program si můžeme stáhnout z domovské stránky producenta na adrese http://www.apache.org/ v sekci download. (Dáme pozor, abychom omylem nevybrali verzi pro UNIX). Instalací nás provede sám program. Poté musíme provést několik nutných úprav v konfiguračním souboru `httpd.conf`, který by se měl nacházet v adresáři `C:\Program Files\Apache Group\Apache\`. Případné další doladění již záleží na správci serveru. Soubor tedy otevřeme a

- za parametr `ServerAdmin` napíšeme emailovou adresu správce serveru
- za parametr `ServerName` napíšeme jméno shodné s názvem počítače
- na disku vytvoříme adresář, ve kterém budou umístěny soubory určené pro WEBové stránky, např. `C:\MWS`
- za parametr `DocumentRoot` vepíšeme do uvozovek cestu k adresáři vytvořeném v předchozím kroku, o několik řádků níže učiníme to samé u parametru `<Directory "C:/MWS">`
- za parametru `DirectoryIndex` můžeme napsat soubory oddělené mezerou, které se v napsaném pořadí budou automaticky spouštět, bude-li uvedena pouze necelá část internetové adresy (např. uvedeme zde soubory *index.htm index.html*, potom po zadání: http://krt32.krt.vslib.cz/fce/ se bude snažit server otevřít nejprve http://krt32.krt.vslib.cz/fce/index.htm nebo v případě jeho marného hledání

http://krt32.krt.vslib.cz/fce/index.html; neuspěje-li žádný z nich, objeví se hlášení o špatně zadané adrese

- v adresáři `C:\MWS` vytvoříme další podadresář s názvem `cgi-bin`
- za parametr `ScriptAlias /cgi-bin/` napíšeme do uvozovek cestu k němu (`"C:/MWS/cgi-bin/"`)

Soubor `httpd.conf` poté uložíme. Klikneme na **Start -> Nastavení -> Ovládací panely -> Nástroje pro správu -> Služby**. Pravým tlačítkem na myši poklepáme na `Apache` a restartujeme jej. Proběhlo-li všechno v pořádku, potom po zadání http://127.0.0.1/ ve WEBovém prohlížeči se zobrazí obsah adresáře `C:\MWS`. Tato adresa je pouze lokální (nazývá se také localhost) a je rezervována pro provoz bez připojení k síti Internet. V případě připojení k síti, mají ostatních počítače umožněn stejný přístup přes Vaši IP adresu (např. http://147.230.128.46/). Lze ji zjistit třeba tak, že zvolíme **Start -> Spustit** a napíšeme `ping jméno_PC`. (Ve Win2000 nepracuje program winipcfg!) Můžete si samozřejmě zaregistrovat vlastní doménu.

3.2 INSTALACE MATLAB WEB SERVERU (MWS)

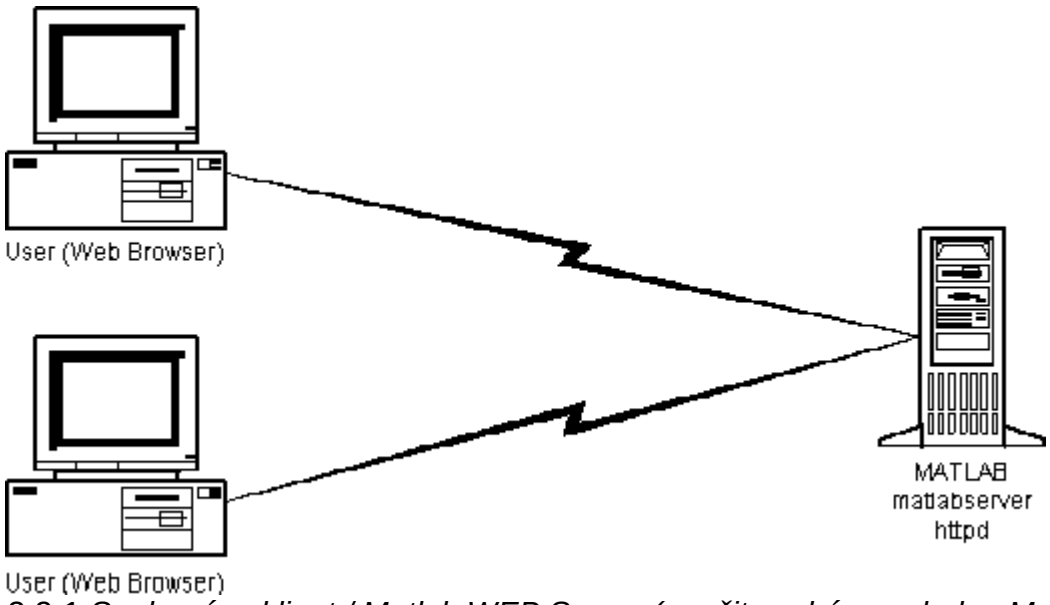
Matlab WEB Server je jedna z volitelných součástí Matlabu. Stačí si jej při instalaci v úvodním výběru zaškrtnout a program provede vše za Vás. Musíme ovšem mít již zprovozněný server, jinak nám tato volba nebude umožněna. Všimněme si, že ve službách přibulde položka `MATLAB Server`. Poté si vytvoříme složku `c:\MWS\fce`. Bude sloužit čistě konkrétním aplikacím MWS. Otevřeme si adresář `C:\MATLAB11\webserver\` a v něm soubor `matlabserver.conf`. Za jedinou direktivou `-m` je číslo, které nastavuje maximální počet povolených spuštění MATLABu, tedy kolik klientů může server najednou obsloužit. Prozatím můžeme nechat nastavení na `1`, později podle okolností lze číslo zvýšit. Otevřeme následující adresář `bin`. Odsud nakopírujeme soubor `matweb.exe` do složky `cgi-bin`. Tam vytvoříme i nový soubor s názvem `matweb.conf`. Budeme jej upravovat vždy, když uděláme nový vstupní formulář volající MWS. Jeho obsah musí mít strukturu z tab.3.2-1 (uvádím zde pouze 3 nutné parametry; ostatní slouží spíše pro ladění a jsou popsané v dokumentaci):

Syntaxe:	**Příklad:**

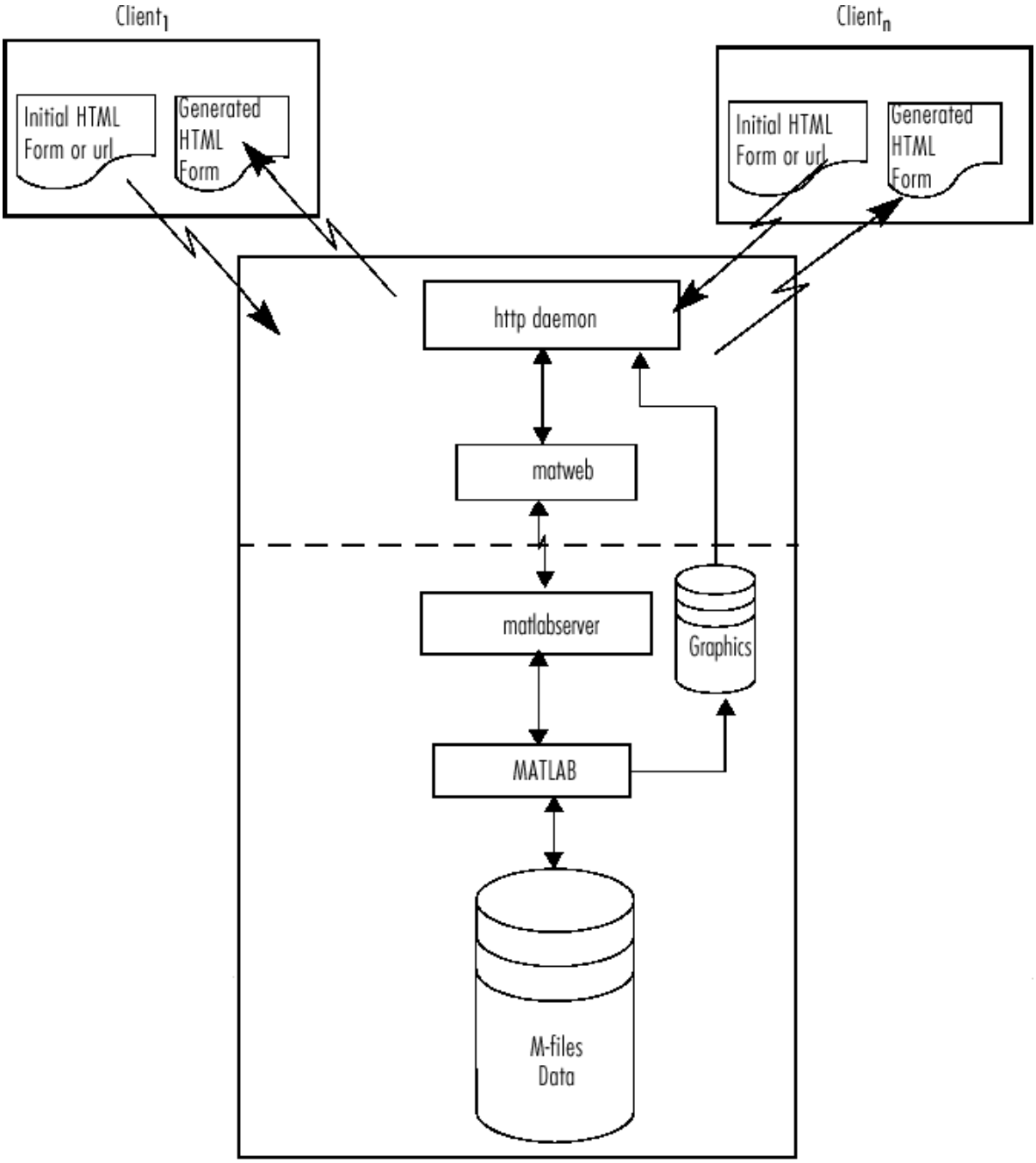
`[název m-souboru1]`
`mlserver=název_počítače`
`mldir=umístění_m-souboru1`
 	
`[název m-souboru2]`
`mlserver=název_počítače`
`mldir=umístění_m-souboru2`
 	
atd.
	`[BodeDg]`
	`mlserver=krt99`
	`mldir=C:\MWS\fce`

Tab.3.2-1 *Struktura souboru matweb.conf*

Jak pracuje celý systém demonstrují následující dva obrázky převzaté z dokumentace obr.3.2-1 a obr.3.2-2.



Obr.3.2-1 *Spolupráce klient / Matlab WEB Server (použito schéma z helpu Matlabu)*



Obr.3.2-2 *Princip práce Matlab WEB Serveru (použito schéma z helpu Matlabu)*

3.3 INSTALACE PHP

PHP (původním významem Personal Home Page) je skriptovací jazyk na straně serveru. Je opět volně šiřitelný a pro svou snadnost a široké možnosti i velmi oblíbený. Bez něj nebo obdobného jazyka stejného zaměření by server například neuměl neomezeně pracovat se soubory. Instalační program si můžeme stáhnout z http://www.php.net/ v sekci download. (Dáváme pozor, abychom vybrali verzi pro Windows.) Na disku `C` vytvoříme pro program nový adresář `php` a do něj instalaci rozbalíme.

Pro jeho provoz potřebujeme dále přidat **Službu IIS (Internet Information Services)** z instalačního CD Windows 2000, která není implicitně nainstalována. Zvolíme tedy **Start -> Nastavení -> Ovládací panely -> Přidat nebo ubrat programy -> Přidat nebo ubrat součásti systému Windows** a zaškrtneme Službu IIS. Instalaci spustíme stiskem tlačítka **Další**.

Z adresáře `C:\php\` zkopírujeme soubory `php.ini` a `php4ts.dll` do `C:\WINNT\`. Ve zkopírovaném `php.ini` upravíme některé parametry:

- u `extension_dir = c:/php/extensions`
- dále seznam `extension` odpoznámkujeme smazáním znaku `#` tak, aby tomu odpovídal adresář `c:/php/extensions`
- budeme-li odesílat emaily, nastavíme parametr `SMTP` na adresu SMTP serveru, např: `SMTP = tyto.vslib.cz` a `sendmail_from` na existující email, ze kterého jako by pošta bude pocházet (pozor na zneužití!) `sendmail_from = MWS@centrum.cz`

Stiskneme **Start -> Nastavení -> Ovládací panely -> Nástroje pro zprávu -> Správce služeb sítě Internet**. Otevřeme název počítače, pravým tlačítkem myši klikneme na Výchozí server WWW a zvolíme **Vlastnosti**. V záložce `Filtry ISAPI` stiskneme tlačítko **Přidat…**. Do názvu filtru napíšeme `PHP` a do Programu `C:\PHP\SAPI\php4isapi.dll`. Potvrdíme stiskem **OK**. Otevřeme záložku `Domovský adresář` a stiskneme **Konfigurace…**. Přidáme novou aplikaci. Program opět

nastavíme na `C:\PHP\SAPI\php4isapi.dll` a příponu na `.php`. Všechny okna zavřeme stiskem **OK**.

Do konfiguračního souboru `httpd.conf` serveru Apache přidáme následující řádky a poté server dříve popsaným způsobem zrestartujeme.

```
ScriptAlias /php/ "c:/php/"
AddType application/x-httpd-php .php .html. .htm
Action application/x-httpd-php "/php/php.exe"
LoadModule php4_module c:/php/sapi/php4apache.dll
```

V adresáři `c:\MWS\` můžeme vytvořit testovací soubor *phpinfo.php*. Jeho obsah budou tvořit následující řádky:

```
<?php
      phpinfo();
?>
```

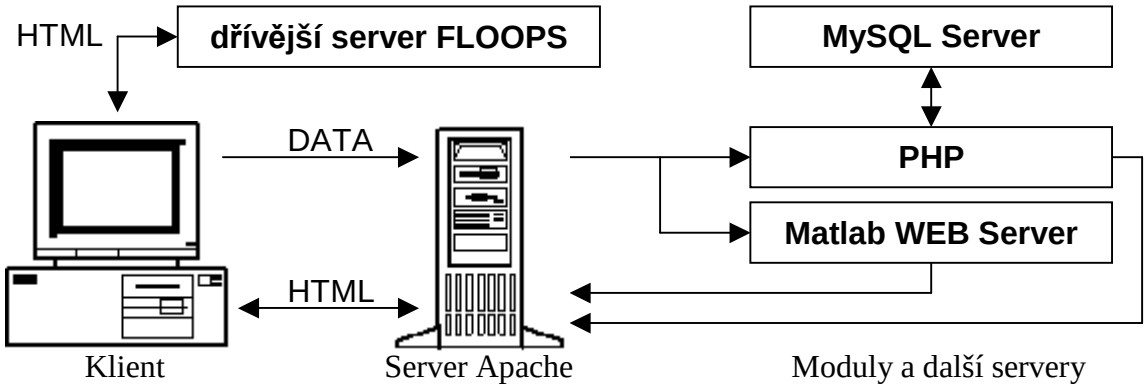
Pracuje-li vše jak má, tak po napsaní adresy http://127.0.0.1/phpinfo.php do WEBového prohlížeče, by se měly zobrazit parametry nastavení PHP.

3.4 INSTALACE MySQL SERVERU

Znova se jedná o volně šiřitelný produkt švédské společnosti TcX, u kterého nemusíme platit licenci, pokud není využíván k ziskové činnosti. MySQL je systém správy relačních databází podnikové úrovně spouštěný ve více vláknech [14]. Instalační soubory si můžeme stáhnout z adresy http://www.mysql.com/ a jak už je zvykem, jsou k dispozici dvě hlavní verze pro Windows a Unix. Vlastní instalace spočívá pouze spuštěním souboru `setup.exe`. Tím jsou veškeré instalace na počítači (serveru) ukončeny. Systém MySQL oživíme spuštěním souboru `mysqld.exe` v adresáři `C:/mysql/bin`. V této složce se můžeme dále podívat i na program `winmysqladmin.exe`, který nám umožňuje rychle se zorientovat o aktuálních stavech databází a vlastně celého systému (při prvním spuštění musíme zadat jméno a heslo hlavního (rootového) správce). K vlastnímu ovládání a programování lze použít konzolové okno zadáním `C:\mysql\bin\mysql –p` do příkazového řádku DOSu. V našem případě přistupujeme k databázi přes funkce

PHP. K ovládání používáme dotazovací jazyk SQL (Structured Query Language). Systém plně odpovídá standardu ANSI SQL92 (viz. dále).

Následující schéma na obr.3.4-1 demonstruje vzájemnou spolupráci všech součástí (dřívějších i nových). Zpracovávání se dělí na dvě větve podle konkrétního požadavku klienta.



Obr.3.4-1 Schéma spolupráce jednotlivých součástí systému

Stručně bych uvedl hlavní funkce jednotlivých částí:

- *klient* uživatel sedící u vzdáleného počítače,
- *server* počítač obsluhující klienty (jsou různé typy, např.: WEBový, MAILový, FTP, LDAP atd.), musí být samozřejmě stálé zapnutý,
- *HTML* výsledný formát předávaný klientovi,
- *JavaScript* skriptovací jazyk na straně klienta,
- *PHP* skriptovací jazyk na straně serveru,
- *Server MySQL* server zajišťující práci s databázemi,
- *Matlab WEB Server* server sloužící ke komunikaci s MATLABem a
- *Server Apache* WEBový server

Dále je na úvodní stránce vytvořena úvodní animace (ve světě je pro tyto účely zažitý název „intro“), mající za účel upoutat návštěvníka stránek. Jak je obecně zvykem, je pro tyto účely použita technologie Flash, která je v současné době jedna z nejnovějších na WEBu.

4. NÁVODY

Pro další provoz stránek, jejich obnovu a případné opravy je nutné vytvořit dokumentaci, která popisuje jejich funkci a dává novému administrátorovi střídající stávajícího dobrý odrazový můstek do začátku. Tomu budou také sloužit následující kapitoly.

4.1 WEBOVÉ STRÁNKY OBECNĚ

Jednoduše řečeno, pod WEBovými stránkami si můžeme představit to, co vidíme na Internetu. Nejpoužívanější formát pro jejich tvorbu je HTML (nebo zkráceně HTM, v počátku používán některými fyziky k propojení vědeckých dokumentů do skupiny serverů CERN). Kromě nich může prohlížeč zobrazovat také soubory *txt*, *pdf*, *doc*, *xls* a další. Nejsou ale již tak efektivní pro toto použití, neboť byli vyvinuty pro jiné účely. Zaměříme se tedy na HTML.

HTML je zkratka anglického výrazu Hyper Text Markup Language. Můžeme jej přeložit jako: hypertextový značkový jazyk. Dobře vystihuje jeho podstatu. Struktura by se dala ve stručnosti popsat jako obyčejný text formátovaný specielními, většinou párovými značkami. Nezáleží u nich na tom, jsou-li napsány velkými či malými písmeny. Zdrojový kód můžeme psát v obyčejném textovém editoru, ale pro urychlení práce je vhodné použít nějaké specializované vývojové prostředí. Z českých produktů bych vybral: **Golden HTML Editor** (je k dispozici na adrese: http://www.oknet.cz/lide/pavelp/ghe/) a ze zahraničních doporučuji **HomeSite** (http://www.macromedia.com/software/homesite/). Na tomto místě bych se chtěl zmínit o existenci tzv. kaskádních stylů (soubory mají příponu *css*). Jedná se o soubory, ve kterých můžeme dát značkám HTML různé vlastnosti. To samozřejmě můžeme i v samotném zdrojovém kódu, ale pokud vše máme popsané ve zvláštním souboru, lze jej používat v libovolném počtu stránek a tím jednoduše dosáhnout celkově jednotného vzhledu. V tomto směru šel ještě dále jazyk XML, kde je definován nejen vzhled, ale i obsah. Ke zdrojovému kódu stránky se dostaneme pomocí menu. V Exploreru zvolíme: **Zobrazit** -> **Zdrojový kód** nebo také kliknutím pravým tlačítkem myši na stránku a volbou: **Zobrazit zdrojový kód**.

4.1.1 Nejpoužívanější HTML značky na stránkách studijních materiálů

Obecná struktura HTML dokumentu má tvar:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
```

sem patří: název stránky, scripty, styly, definování znakové sady, apod.

```
</head>

<body>
```

sem patří: publikovaný text, obrázky, tabulky,… znova skripty

```
</body>
</html>
```

Obr. 4.1.1-1 *Obecná struktura HTML dokumentu*

Poznámka: skript je vlastně další jazyk vložený do HTML, který jej dělá v rámci svých možností dynamičtější. Patří mezi ně například: JavaScript (neplést s Javou) a VBScript.

Dále se konečně zmíním o značkách, které jsou na stránkách použity.

1) část `<head></head>`

pojmenování okna stránky:

```
<title>název stránky</title>
```

nastavení znakové sady používané ve windows:

```
<META HTTP-EQUIV="Content-Type" CONTENT="text/html;
CHARSET=Windows-1250">
```

nastavení cesty k souboru s kaskádními styly styl.css:

```
<LINK href="styl.css" rel="StyleSheet" type="text/css">
```

nastavení stylů lokálně na aktuální stránce (velikost písma), stejně vypadají *css* soubory, vypustíme-li značky `<style></style>`:

```
<style>
  .nadpis {font-size: 30}
</style>
```

jednoduchý script (JavaScript & funkce na zavírání okna) [3]:

```
<script language="JavaScript">
  function ZavriOkno() {
        window.close();
  }
</script>
```

2) část `<body></body>`

základy:

```
nový řádek: <br>                vodorovná čára: <hr>
   kurzíva: <i>text</i>               ztučnění: <b>text</b>
    + nebo: <em>text</em>               + nebo: <strong>text</strong>
  potržení: <u>text</u>               hlavičky: <h1>text</h1>   ↓ menší
   odrážky: <li>text</li>                            ...        písmo
  paragraf: <p>text</p>                        <h5>text</h5>
                                  centrování: <center>text</center>
```

tabulky:

```
   vytvoření tabulky: <table></table>
         nový řádek: <tr></tr>
       nový sloupec: <td>text</td>
nová hlavička sloupce: <th>text</th>
```

v praxi se uzavírací část často vynechává, neboť výsledek je stejný

očíslované seznamy:

```
vytvoření seznamu: <ol></ol>
  položka seznamu: <li>sloupce tabulky</li>
```

V `<ol>` můžeme nastavit od kolika chceme číslovat: `<ol start="10">` nebo také jakým způsobem číslovat:

- `<ol type="1">` čísly
- `<ol type="a">` abecedně malými písmeny
- `<ol type="A">` abecedně velkými písmeny
- `<ol type="i">` římsky malými písmeny
- `<ol type="I">` římsky velkými písmeny

obrázky:

```
vytvoření obrázku: <img src="cesta_k_obrazku">
```

V `<img>` můžeme nastavit velikost okraje: `<img border="5">` nebo také jeho zarovnání:

- `<img align="left">` horizontálně doleva
- `<img align="right">` horizontálně doprava
- `<img align="middle">` vertikálně doprostřed řádku
- `<img align="bottom">` vertikálně naspod řádku
- `<img align="top">` vertikálně navrch řádku

odkazy:

```
vytvoření externího odkazu: <a href="cesta_k_cili">text odkazu</a>
vytvoření lokálního odkazu: <a href="#nazev_cile">text odkazu</a>
      cíl lokálního odkazu: <a name="nazev_cile">text cíle</a>
```

V `<a>` můžeme nastavit v jakém okně se má odkaz otevřít (implicitně je nastaveno v aktuálním) nastavením `<a target="jmeno">`. Chceme-li nezávislé okno, za jméno vybereme `_blanc`.

formuláře:

vytvoření formuláře: `<form></form>`

Ve `<form>` můžeme nastavit jakým způsobem bude formulář odeslán. Nejobvyklejší jsou dva způsoby `<form method="post">` nebo `<form method="get">`. Použijeme-li metodu post, nebudou odesílaná data součástí internetové adresy. Dále musíme zapsat, kam formulář budeme odesílat. Provedeme to parametrem `<form action="cil.php">` Odesíláme veškeré vstupy, které mají nastaven parametr `name` a odesílaná veličina je parametr `value`. Ve formuláři můžeme použít například tyto vstupy:

- tlačítko s nápisem STOP
  `<input type="button" value="STOP">`
- prázdné textové pole se jménem text1
  `<input type="text" value="" name="text1">`
- tlačítko pro odeslání formuláře
  `<input type="submit" value="Odešli">`
- tlačítko pro nastavení původních hodnot ve vstupech
  `<input type="reset" value="Vyčisti">`
- zaškrtávací políčko, implicitně zaškrtnuto; nechceme-li tak, checked vynecháme
  `<input type="checkbox" checked name="souhlas">`
- výběrová políčka svázaná k sobě stejným názvem (r1), odesíláme buď "a" nebo "b"
  `<input type="radio" name="r1" value="a">`
  `<input type="radio" name="r1" value="b">`

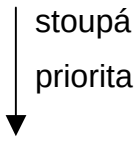
Poznámka: pro popis zaškrtávacích a výběrových políček je vhodné následný text uzavřít mezi značku `<label for="jmeno"></label>`*, kde do parametru* `for` *zvolíme stejný název jako u políček*

- roletový vstup, odesílá se buď "val1" nebo "val2", uživatel vybírá z názvů Kočka a Pes

```
<select name="roleta">
   <option value="val1"> Kočka
   <option value="val2"> Pes
</select>
```

Chtěl bych ještě doplnit, že velké možnosti nastavení vzhledu umožňuje vlastnost `style`. Ovlivňujeme ji pomocí:

- kaskádních stylů
- značkami `<style></style>` v části `<head></head>`
- vlastní definicí uvnitř značky, která ji využívá



Poznámka: kompletní seznam značek i s jejich vlastnostmi můžete zjistit z mnoha zdrojů. Např.: http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/reference/dhtmlrefs.asp.

Příklad. Chceme vytvořit stránku, která bude vypadat následujícím způsobem:

Dotazník

Dobrý den, jsme firma, která se zabývá průzkumem trhu. Byli bychom **rádi**, kdybyste nám odeslali následující dotazník.

S pozdravem, Jan Novák
vedoucí

Jméno: Název firmy:

Příjmení: Postavení ve firmě: vyberte

Jak se vám líbí naše produkty? Upřesňující text:

- jsme spokojeni
- jsme spokojeni
- je malý sortiment

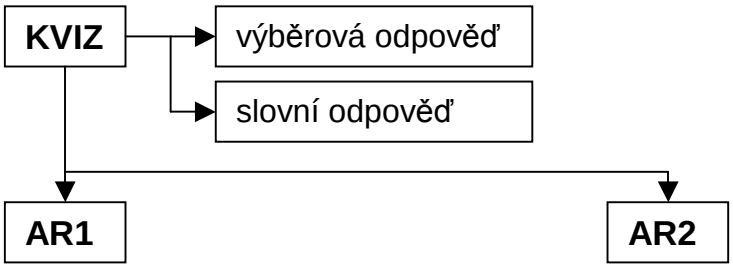
Odeslat Odešli

V příkladě vidíme jakýsi dotazník firmy Novák s. s r. o. Další potřebná data: Logo firmy je umístěné v souboru *logo.gif* a soubor pro zpracování odeslaného formuláře má jméno *dotaznik.php*. Počet odesílaných dat by mohl být velký zejména díky položce *Upřesňující text*. Použijeme tedy metodu *post* pro odesílání formuláře. (U skriptů PHP tento postup doporučuji přednostně používat, neboť internetová adresa zbytečně nenarůstá a navíc se nemusíme starat o to, zda není překročená její maximální délka a zda se odesílaný text správě zakóduje. K přijatým datům navíc přistupujeme zcela stejně jednoduše u obou způsobů.) Pro zarovnání vstupů je použita tabulka.

Kód by mohl vypadat například takto:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">

<html>
<head>
	<META HTTP-EQUIV="Content-Type" CONTENT="text/html;
		CHARSET=Windows-1250">
	<TITLE>Dotazník firmy: Novák, s. s r. o.</TITLE>
</head>

<body>
<img src="logo.gif" align="right" alt="Logo: Novak, s. s r. o.">
<div align="center"><h1>Dotazník</h1></div>
<p style="text-indent:0.5in">Dobrý den, jsme firma, která se zabývá průzkumem trhu. Byli bychom
	<b>rádi</b>, kdybyste nám odeslali následující dotazník.</p>
<div align="right">S pozdravem, Jan Novák<br>vedoucí</div>
<form action="dotaznik.php" method="post">
	<table>
	<tr valign="top"><td>
			<table>
				<tr valign="top">
					<td><u>Jméno:</u>
					<td><input type="text" name="firstname">
				<tr valign="top">
					<td><u>Příjmení:</u>
					<td><input type="text" name="surname">
			</table>
			<u>Jak se vám líbí naše produkty?</u>
			<br><input type="radio" name="satisfaction" id="good"><label
				for="good">jsme spokojeni</label>
			<br><input type="radio" name="satisfaction" id="sort"><label
				for="sort">je malý sortiment</label>
			<br><input type="radio" name="satisfaction" id="price"><label
				for="price">ceny jsou vysoké</label>
			<td>
			<table>
				<tr valign="top">
					<td><u>Název firmy:</u>
					<td><input type="text" name="firma">
				<tr valign="top">
					<td><u>Postavení ve firmě:</u>
					<td><select name="position" style="width:144">
							<option value="0" checked>vyberte
							<option value="1">ředitel
							<option value="2">náměstek
							<option value="3">vedoucí
							<option value="4">zaměstnanec
						</select>
			</table>
			<u>Upřesňující text:</u><br>
			<textarea name="addtext" cols="30" rows="4"></textarea><br>
			<input type="reset" value="Vyčisti"><input type="submit" value="Odeslat">
	</table>
</form>
</body>
</html>
```

4.2 WEBOVÉ STRÁNKY TYPU KVIZ



Tab.4.2-1 *Schéma kvizu v jednotlivých předmětech*

1. Modely, signály, šum
2. Minimalizace kv. krit., dynam. charakt.
3. Dopravní zpoždění, L-obraz
4. Bloková algebra
5. Geometrické místo kořenů
6. Frekv. charakt., Nyquistovo kritérium
7. PID regulátory
8. Seřízení PID reg. podle kvadr. krit.
9. Seřízení PID reg. podle minimální lineární reg. plochy a metodou opt. modulu
10. Seřízení PI a PD reg. frekv. přístupem
11. Filtry s fázovým zpožděním a předstihem
12. Rozvětvené reg. obvody
13. ... pokračování (viz. 12))
14. Počáteční podmínky a jejich transformace

1. Šum, diskrétní modely
2. Odhad parametrů modelu, korelační fce.
3. Diskrétní systém, Z-přenos
4. Seřízení diskr. PID reg., kvadr. krit.
5. Stav. popis, transf. do nových souřadnic
6. Estimace, návrh stav. reg. metodou Pole Placement
7. Stavový reg. s integrační složkou
8. Nelin. syst., linearizace, přepínací přímky
9. Fázové trajektorie, stabilita nelin. syst.
10. Relé, Ljapunovovo a Popovovo krit.
11. Úvod do fuzzy-řízení
12. Inferenční pravidla, deffuzyfikace
13. Fuzzy regulátor
14. MIMO systémy
15. ... pokračování (viz. 14))

Program kviz je vytvořen v HTML jazyce s použitím skriptovacího jazyka JavaSkript. Slouží jako autotest. Skládá se ze dvou částí. První vygeneruje formulář pro zadávání odpovědí a druhá informuje uživatele o správnosti zodpovězení otázek.

Struktura zdrojového kódu je takováto:

1) hlavička (head) – deklarace proměnných, polí a načtení otázek
   a) *q* - pole pro ukládání odpovědí
   b) *pocet* - proměnná udávající počet otázek
   c) *otazky* - pole otázek skládajících se ze tří částí: vlastní otázka, typ vstupu odpovědi a správná odpověď (pro pohyb v něm tedy použijeme schématu: `3*a+b` kde a je z intervalu `<0,pocet>` - nese informaci o číslu otázky a číslem b z intervalu `<0,2>` - vybíráme chtěný údaj).
   d) *r* - proměnná udávající počet dosud vygenerovaných vstupů typu `radio`

e) *rad* - pole možností pro vstupy typu `radio` (vždy trojice možností)

2) tělo (body) – vlastní program, který tvoří část generující formulář odpovědí a funkce, která kontroluje jejich správnost, obojí formátované pomocí tabulky
   a) *program* - v cyklu běžícím podle počtu otázek testujeme, zda na právě zpracovávanou otázku odpovídáme do vstupu typu `text` nebo `radio`. V prvním případě vygenerujeme číslo otázky a značku `<input type=text id=qi>`. (Parametr `id` generujeme dynamicky, první písmeno q zůstává nezměněno a následující číslice je odvozena z čísla otázky. Pomoci tohoto parametru se při kontrole dovíme, jak uživatel odpověděl.) V druhém případě vygenerujeme také číslo otázky a podobnou značku `<input type=radio name=qi value=j onClick=q[i]=this.value>`. Ta je vygenerována 3x (máme tři možné odpovědi). Parametr `id` je zaměněn za `name` a tím si zaručíme, že budou svázány tři k sobě patřící tlačítka. Funkce pak do pole odpovědí q zapíše uživatelem vybranou. `Value` nyní nese údaj o zvolené možnosti. S ním se bude později kontrolovat správnost odpovědi. Nakonec vytvoříme tlačítka pro spuštění kontroly a vymazání testovacího formuláře.
   b) *funkce* - v cyklu procházíme pole odpovědí s `i`-tým indexem a porovnáváme je se správnými uloženými v poli se strukturou `otazky[i*3+2]`. Nejdříve spočítáme počet nesprávných a procento úspěšnosti. Poté cyklus provádíme znovu a v případě nesouladu zobrazíme otázku, odpověď uživatele a správné řešení. Celek ukončíme tlačítkem pro návrat k otázkám.

Tělo můžeme ponechat nezměněné pro všechny ostatní programy typu kviz. Budou se od sebe lišit pouze v hlavičce otázkami (`otazky`), jejich počtem (`pocet`) a polem možností pro vstupy typu `radio` (`rad`). Pro vytvoření nového programu typu kviz se nemusíme zabývat tím jak vlastně pracuje a stačí se držet následujícího návodu:

Zkopírujeme již existující soubor (např. *kviz1.html*) pomocí Ctrl + C a vytvoříme nový kombinací kláves Ctrl + V. Ten přejmenujeme (např. *kviz2.html*). Otevřeme jej a v menu zvolíme: **Zobrazit -> Zdrojový kód**. Otevře se nám textový soubor obsahující vlastní program. Za značku `<title>` v části `head` napíšeme nový název okna (např. Kviz č. 2). Stejně postupujeme za značkou `<h1 align="center">` v části `body`. (Tento druhý text se zobrazí jako nadpis kvizu.) Vrátíme se na začátek souboru a do proměnné nadeklarované pomocí `var pocet` uložíme počet otázek nového kvizu. Vymažeme obsah polí `otazky` a `rad`. (Vše co je mezi závorkami za slovem `Array`.) Jejich místo nahradíme novými otázkami. Vytvoříme je dle následujícího schématu, poté celý soubor uložíme a můžeme vyzkoušet jeho spuštěním. Celý systém by mělo ozřejmit následující schéma.

Pole otazky:

a) “[*text otázky pro slovní odpověď*]“, “text“,“[správná odpověď]“,

b) “[*text otázky pro výběrovou odpověď*]“, “radio“,“[číslo správně odpovědi (1 - 3)]“,

Pole rad (doplňujeme pouze v případě b) u pole otázky)**:**

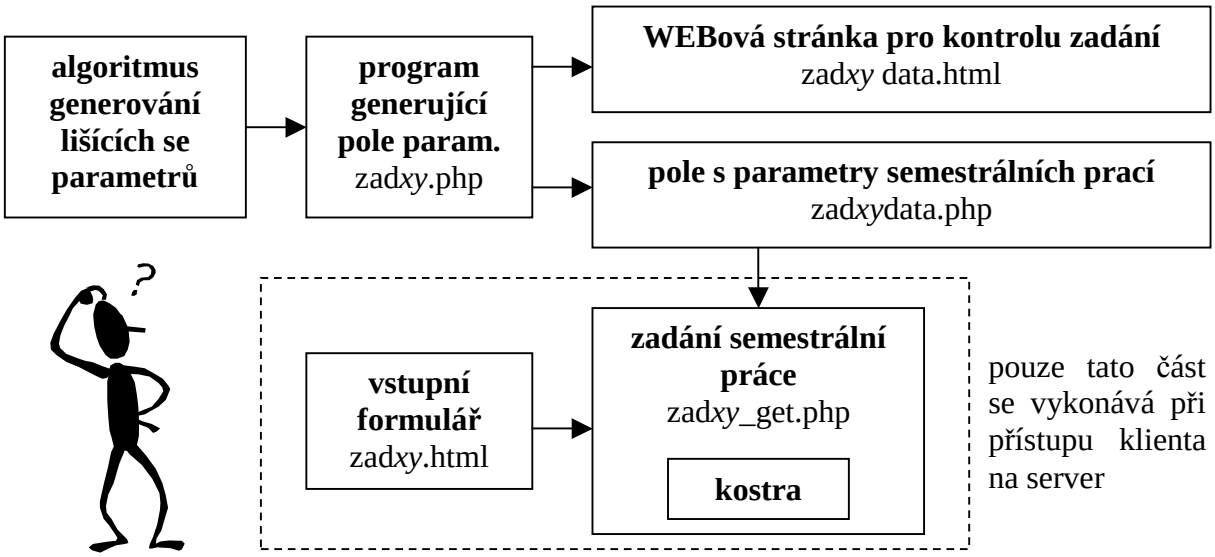
“[první možnost]“, “ [druhá možnost]“,“[třetí možnost]“,

Často se opakuje chyba, že za poslední položkou pole necháme čárku a naopak u nějaké předcházející na ní zapomeneme (zejména na konci řádku). V případě problémů je často užitečné prostudovat zdrojový kód již funkčního testu. Součástí textu otázek mohou být značky jazyka HTML. Chyba nastane, použijeme-li v textu uvozovky. Odstraníme ji použitím `\"` místo ". Celé pole můžeme napsat na jediný řádek, ale ztratili bychom pak přehlednost a proto doporučuji pro každou otázku nový řádek.

4.3 WEBOVÉ STRÁNKY PRO ZADÁVÁNÍ SEMESTRÁLNÍCH PRACÍ

Záměrem bylo vytvořit dynamické zadávání semestrálních prací pro každého studenta pokud možno odlišné. Proto před vytvořením zadání musí každý vyplnit jednoduchý vstupní formulář (sobory zad*xy*.html) obsahující dvě pole: označení studijního kroužku a pořadí v seznamu řazeném například dle abecedy (není důležitý způsob seřazení, ale aby každý student měl své odlišné číslo). Vlastní generování pak probíhá na nově zřízeném serveru (kde je v provozu i Matlab WEB Server) pomocí PHP. V některých případech toto nebylo zapotřebí a zadání se pro všechny studenty shodovala (místo vstupního formuláře se ihned objeví zadání). Dále tyto případy nebudu uvažovat.

Zadání semestrální práce (sobory zad*xy*.php) tvoří text (kostra) shodný pro všechny, lišící se pouze v některých bodech (např. jiné koeficienty v polynomu apod.). Tyto odlišnosti jsou zachyceny v poli (soubory zad*xy*data.php), které se při tvorbě vlastního zadání (soubory zad*xy*_get.php) do programu vloží funkcí `include ("zadxydata.php")`. Soubor zad*xy*data.php je vytvořen dle daného algoritmu programem zad*xy*.php společně s WEBovou stránkou pro kontrolu zadání (soubor zad*xy* data.html). Tento proces proběhne pouze jednou a později je využívané pouze pole dat. Výsledek si student může okamžitě vytisknout nebo uložit na disketu. Celý postup demonstruje následující obr.4.3-1.



Obr.4.3-1 *Schéma demonstrující zadávání semestrálních prací*

4.4 VSTUPNÍ A VÝSTUPNÍ FORMULÁŘE MWS A ZPRACOVÁNÍ NA STRANĚ SERVERU

V následujícím textu je popsán způsob, jak vytvořit aplikaci typu klient/server dávající uživateli bez nainstalovaného Matlabu využít jeho výpočetní možnosti a přiblížit problematiku řízení systémů daleko širší veřejnosti, než pouze lidem pohybujícím se v tomto oboru.

Jak už název kapitoly napovídá, budeme vytvářet tři soubory úzce se sebou spjaté. Bude tedy vhodné hned zpočátku zvolit způsob jejich pojmenovávání pro lepší orientaci a všechny je pak umístit do jediného adresáře. Získáme tím menší složitost při psaní přístupových cest k souborům. Tab.4.4-1 ukazuje jednu možnost.

vstupní formulář:	*BodeDg_in.html*
m soubor:	*BodeDg.m*
výstupní formulář:	*BodeDg_out.html*

Tab.4.4-1 *Pojmenovávání souborů*

4.4.1 Vstupní formulář MWS

Stránku vytváříme v HTML jazyce. V ní musíme vytvořit formulář způsobem popsaným v kapitole 4.1.1 Nejpoužívanější značky na stránkách studijních materiálů a doplněný o jedno uživateli skrytý vstup

```
<input type="hidden" name="mlmfile" value="BodeDg">,
```

kde vlastnost `value` obsahuje název matlabovského *m* souboru pro vykonání zadání. Vstupy můžeme použít všech druhů, ale nejpoužívanější bude asi typu `text` a nesmíme zapomenout na `submit` pro odeslání. Dále značka `form` bude upravena do následující podoby

```
<form action="../cgi-bin/matweb.exe" method="post">.
```

Celou stránku je vhodné doplit o zadání úlohy, schéma a další údaje pro správné pochopení.

Abychom se v budoucnu pro nové zadání nemuseli vracet v internetovém prohlížeči zpět, tak část kódu mezi párovou značkou `<form></form>` zkopírujeme do výstupního formuláře, ale o tom až později. Následuje kód určený pro Bodeho diagram (soubor *BodeDg_in.html*).

```
<form action="../cgi-bin/matweb.exe" method="post">
     <u>Zadej polynomy přenosu:</u>
     <input type="hidden" name="mlmfile" value="BodeDg">
     B: <input type="text" name="B" value="[1]" size="20"><br>
     A: <input type="text" name="A" value="[1 1]" size="20"><br>
     <br>
     <input type="submit" value="odeslat">
</form>
```

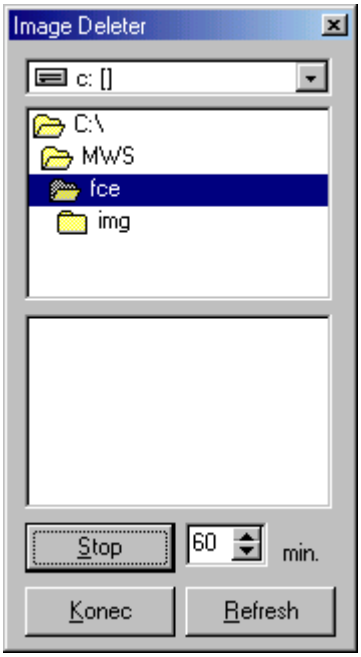
4.4.2 Zpracování na straně serveru

Data ze vstupního formuláře se odesílají klasickému *m*-souboru s názvem stejným jako je parametr `value` u vstupu jménem `mlmfile`. Uvnitř souboru je jediná funkce, která vrací výstupní řetězec prohlížeči. Na následujícím příkladu vykreslení Bodeho diagramu si nejjednodušeji vysvětlíme celou strukturu (soubor *BodeDg.m*).

```
function HTMLout=BodeDg(h)       %vytvoření funkce
mlid=getfield(h,'mlid');         %načtení vstupních dat do h
                                 %   cesta je v souboru matweb.conf
cd(h.mldir);                     %nastavení pracovního adresáře

B=h.B;     %načtení čitatele do proměnné B (vstup má jméno B)
A=h.A;     %načtení jmenovatele do proměnné A (vstup má jméno A)

s.B=B;     %načtení čitatele do výstupního řetězce s
s.A=A;     %načtení jmenovatele do výstupního řetězce s

sys=tf(eval(B),eval(A));   %vytvoření systému (musíme převést
                           %   proměnné z řetězce na polynomy
bode(sys);                 %vykreslení bodeho diagramu
savejpg1=strcat('img',num2str(round(cputime^cputime*1000000)));
savejpg1=strcat(savejpg1,'.jpg');

%vytvoříme jméno obrázku s náhodným číslem, aby prohlížeč klienta
%   nepoužil obrázek uložený v cache
```

```
s.img1=savejpg1;           %načtení názvu obrázku do výst. řetězce s
wsprintjpeg(gcf,savejpg1); %vytvoření obrázku ve formátu jpeg (jpg)
outfiletemp=which('BodeDg_out.html'); %název výstupního formuláře
HTMLout=htmlrep(s,outfiletemp);       %nahrazení ve výstupním
                                      %   formuláři výskyt $promena$
                                      %   daty z výstupního řetězce s
```

Pro vytvoření názvu obrázku s proměnným názvem nemůžeme užít `rand` (funkce generující pseudonáhodné číslo), neboť při každém spuštění Matlabu je vygenerováno stejné číslo. Můžeme ale použít dobu výpočetního času (funkce `cputime`), která není konstantní. Budeme-li však vytvářet obrázky s náhodnými jmény, potom by se zbytečně hromadili v adresáři serveru. Na jejich automatické mazání byl udělán jednoduchý program, který se o mazání obrázků stará automaticky v nastaveném intervalu (nesmí být moc krátký, aby byla malá pravděpodobnost, že by se soubor s obrázkem smazal dříve, než by jej klient obdržel) viz. obr.4.4.2-1. V následující tabulce tab.4.4.2-1 je uvedeno, jak získat data z různých vstupů.



Obr.4.4.2-1 *Program na mazání souborů s obrázky z MWS*

druh vstupu	**syntaxe pro získání hodnoty**
<input type="text" name="A">	`A=h.A;`

<select name="roleta">
<option value="val1"> 1000
<option value="val2"> 2000
	`switch h.roleta`
	`case 'val1'`
	`  number=1000;`
	`case 'val2'`
	`  number=2000;`
	`end`

OptionButton4	

<input type="radio" name="r1" value="a">
<input type="radio" name="r1" value="b">
	`switch h.r1`
	`case 'a'`
	`  moznost='a';`
	`case 'b'`
	`  moznost='b';`
	`end`

CheckBox1	

<input type="checkbox" name="ch">	`vyber='false';`
	`if (isfield (h,'ch'))`
	`  vyber='true';`
	`end`

Tab.4.4.2-1 *Způsob získání dat z různých vstupů (vstupní řetězec je v proměnné h)*

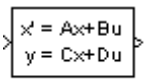
Pokud využíváme při zpracovávání požadavku simulinkový model, musíme k němu přistupovat trochu odlišným způsobem, než jsme byli doposud zvyklí. Veškeré proměnné ve funkci jsou totiž lokální a modely načítají data z workspace. Nejlépe bude opět řešení demonstrovat třeba na příkladu zadání polynomů jmenovatele a čitatele do bloku přenosu pojmenovaném `TF1` (proměnnými jsou řetězce `jmen` a `cit`). Předpokládejme dále, že soubor s modelem se jmenuje `simulace.mdl`.

```
load_system('simulace');                          %načtení modelu
set_param('simulace/TF1','Numerator',cit);        %uložení čitatele
set_param('simulace/TF1','Denominator',jmen);     %uložení jmenovatele
sim('simulace');                                  %provedení simulace
```

Získání druhého parametru do funkce `set_param` provedeme tak, že nad vybraný blok najedeme kurzorem myší, chvíli počkáme a poté se nám objeví popis bločku, obsahující veškeré názvy jeho parametrů. V tab.4.4.2-2 uvádím některé nejpoužívanější případy.

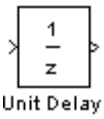
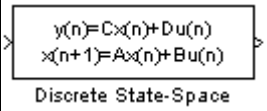
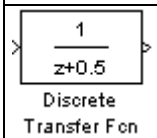
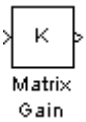
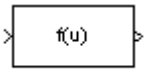
značka bloku	**názvy parametrů**	**ekvivalentní názvy použité Matlabem při ručním vyplňování parametrů bloku**
	`Value`	`Constant value`

	`Time`
	`Before`
	`After`
	`SampleTime`
		`Step time`
		`Initial value`
		`Final value`
		`Sample time`

	`A`
	`B`
	`C`
	`D`
	`X0`
		`A`
		`B`
		`C`
		`D`
		`Initial conditions`

	`Numerator` `Denominator`	`Numerator` `Denominator`

	`ExternalReset`
	`InitialConditionSource`
	`InitialCondition`
	`LimitOutput`
	`UpperSaturationLimit`
	`LowerSaturationLimit`
	`ShowSaturationPort`
	`ShowStatePort`
	`AbsoluteTolerance`
		`External reset`
		`Initial condition source`
		`Initial condition`
		`Limit output`
		`Upper saturation limit`
		`Lower saturation limit`
		`Show saturation port`
		`Show state port`
		`Absolute tolerance`

	Gain SaturateOnIntegerOverflow	Gain Saturate on int. overflow
	X0 SampleTime	Initial condition Sample time
	A B C D X0 SampleTime	A B C D Initial condition Sample time
	Numerator Denominator SampleTime	Numerator Denominator Sample time
	K	Gain matrix
	Expr	Expression
	UpperLimit LowerLimit	Upper limit Lower limit
	LowerValue UpperValue	Lower value Upper value
	OnSwitchValue OffSwitchValue OnOutputValue OffOutputValue	Switch on point Switch off point Output when on Output when off

Tab.4.4.2-2 *Názvy parametrů bloků Matlabu*

Místo bloku `scope` který nám v tomto využívání Matlabu není moc platný můžeme použít například blok `To Workspace`. Jeho parametr `Save format` ovšem změníme ze `Structure` na `Matrix`. Ve `Variable name` napíšeme jméno matice, ze které budeme čerpat odsimulovaná data.

4.4.3 Výstupní formulář MWS

Jedná se opět o stránku v HTML jazyce obohacenou o data získaná z MWS. Uživateli je zpřístupníme tím, že použijeme názvy proměnných použitých v *m* souborech a obklopíme je z obou stran symbolem „$". V tab.4.4.3-1 si uvedeme příklad:

***m* soubor**	**HTML soubor**
`s.img1 = savejpg1;`	`<img src="$img1$">`
`s.A = A;`	`<input name="A" type="text" value="$A$">`

Tab.4.4.3-1 *Příklad použití dat z MWS*

Jak jsem už dříve uved, je vhodné kód doplnit o vstupní formulář. Jednoduše jej sem zkopírujeme. Místo implicitních hodnot ve vlastnosti `value` ovšem je nyní lépe použít data vrácená z MWS, neboť v případě, že uživatel některé hodnoty změnil, objevili by se nám nyní původně nastavené.

`value` ve vstupním formuláři

```
<input name="A" type="text" value="[1 2 1]">
```

`value` ve výstupním formuláři

```
<input name="A" type="text" value="$A$">
```

Tab.4.4.3-2 *Parametr value ve vstupním a výstupním formuláři*

Následujícím kódem dokončíme příklad vracející uživateli Bodeho diagram (soubor *BodeDg_out.html*).

```
<img src="http://krt32.krt.vslib.cz/fce/$img1$"><br>
<form action="../cgi-bin/matweb.exe" method="post">
     <u>Zadej polynomy přenosu:</u>
     <input type="hidden" name="mlmfile" value="BodeDg">
     B: <input type="text" name="B" value="$B$" size="20"><br>
     A: <input type="text" name="A" value="$A$" size="20"><br>
     <br>
     <input type="submit" value="odeslat">
</form>
```

4.5 TVORBA SOUBORŮ TYPU PDF

Úvodem by bylo vhodné říct, že na rozdíl od běžně užívaných souborů (*.txt, *.doc, …) nezačínáme tvořit přímo *pdf* formátu, ale konečný soubor vznikne teprve importováním např. do námi používaného programu Adobe Acrobat. Jedná se o:

Adobe FrameMaker	**.fm, *.mk5*
Ascii Text	**.txt, *.text*
HTML	**.htm, *.html, *.shtml*
WordPerfect	**.wpd*
Microsoft Excel	**.xls*
Microsoft PowerPoint	**.ppt*
Microsoft Word	**.doc*
Image Files	**.gif, *.tif, *.png, *.jpg, *.jpe, *.bmp, *.pcx*

My se zaměříme na Microsoft Word (*doc*).

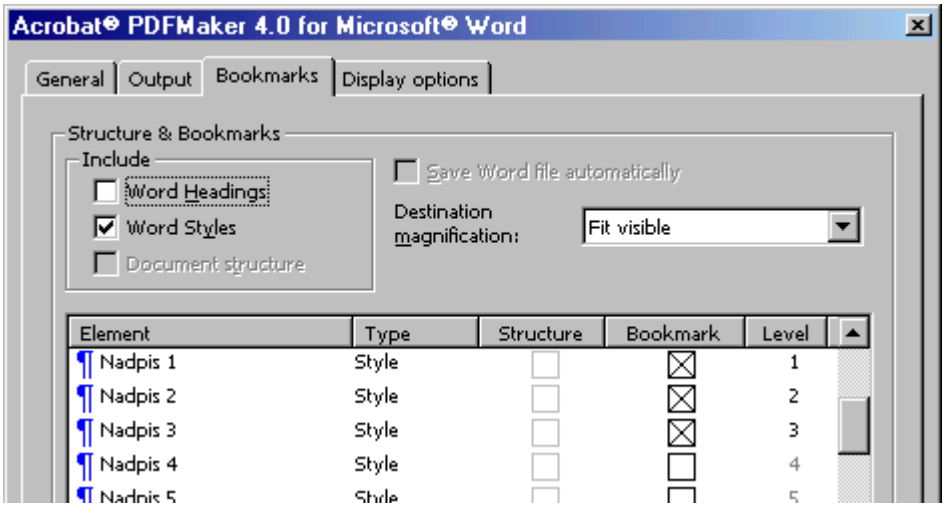
Po samotné instalaci Adobe Acrobata se ve Wordu automaticky nainstaluje ke stávajícím nástrojovým panelům nový, sloužící k importování do Acrobata a sám se též zobrazí v horní sadě ikonek viz. obr.4.5-1.

Obr.4.5-1 *Panel pro import z Wordu do Acrobata*

Při vytváření *doc* souboru dbáme na to, aby nadpisy a jiné důležité pasáže ke kterým budeme v chtít vytvořit záložky (bookmarky), měly nadefinovány správné styly a neměli jsme v budoucnu zbytečnou práci navíc s jejich ručním vytvářením viz. obr.4.5-2. (Používáme většinou: Nadpis 1, Nadpis 2, Nadpis 3.)

Obr. 4.5-2 *Roletové menu stylů ve Wordu*

Máme-li text vytvořen, stiskneme výše popisovanou ikonku. Zvolíme záložku Bookmarks a v tabulce zaškrtneme styly ze kterých chceme vytvořit záložky (viz. Obr. 4.5-1)



Obr. 4.5-1 *Panel pro nastavení automatického generování bookmarků (záložek)*

Za zmínění stojí i záložka Output, kde lze vybrat co všechno chceme, aby *pdf* soubor obsahoval. Poté stačí stisknout tlačítko **Create** a veškerou další práci za nás obstará program sám.

V případě použití diakritických znamének, je nutné záložky zkontrolovat a ručně opravit vzniklé chyby. Panel se otevře stiskem ikonky . Tyto problémy nastávají i v Adobe Acrobatu CE. Konečné úpravy na dokumentu lze provést nástroji na levé svislé liště. Jejich podrobnější popis jsem provedl v loňském projektu s názvem *Vypracování jednoduchého hypertextového dokumentu vybraného tématu pro potřebu výuky v předmětu TAŘ I, II*. Pomoc můžeme nalézt také na adrese:

http://www.it.rsad.edu/tools/acrobat/

Chceme-li ochránit dokument před cizím zásahem, postupujeme následovně. V menu zvolíme: **File -> Save As…** V otevřeném okně rozbalíme roletu **Security** a zvolíme Standard. Okno vyplníme a soubor uložíme. Můžeme tím zabránit:

- tisku
- změnám v dokumentu
- označování textu a grafiky
- přidávání či změnění popisů a formulářových polí

4.5.1 Problémy, které vznikly při importování z Wordu do Acrobata

- špatné znaky s diakritickými znaménky v záložkách

  řešení: ručně přepsat písmena, pokud na nějaké zapomenete, nelze je již druhým pokusem o opravu přepsat; musíme buď celý dokument importovat znova nebo stávající záložku smazat a ručně vytvořit znva, i v samotném textu nastávají tyto problémy u některých druhů písma, zde však nepomůže ani přepsání, musíme zvolit jiný font písma
- ztratil se mi obsah orámovaného vzorečku

  řešení: orámování jsem zrušil a rámeček jsem v Acrobatu dodělal ručně
- z úvodní stránky zmizel text

  řešení: stránku jsem importoval samotnou a nahradil s ní poškozenou
- chyba při importování

  řešení: stačí zrušit zaškrtnutí políček Internet `links` a `Cross-document links` v záložce `Output` na importovacím panelu a poté ručně dodělat veškeré odkazy
- zřejmě nevhodným nastavením velikosti stránky nebo přesahem okrajů stránky došlo roztržení textu a vytvoření několika stránek navíc

  řešení: v poškozených místech je nutné provést korekce obsahu, může také pomoct neimportovat text celý, ale například po skupinách deseti stránek a následně vše spojit dohromady
- obrázky které nejsou vytvořeny přímo ve Wordu ikonkami v panelu kreslení ztratili velice na kvalitě, zejména pokud se jedná o tenké čáry

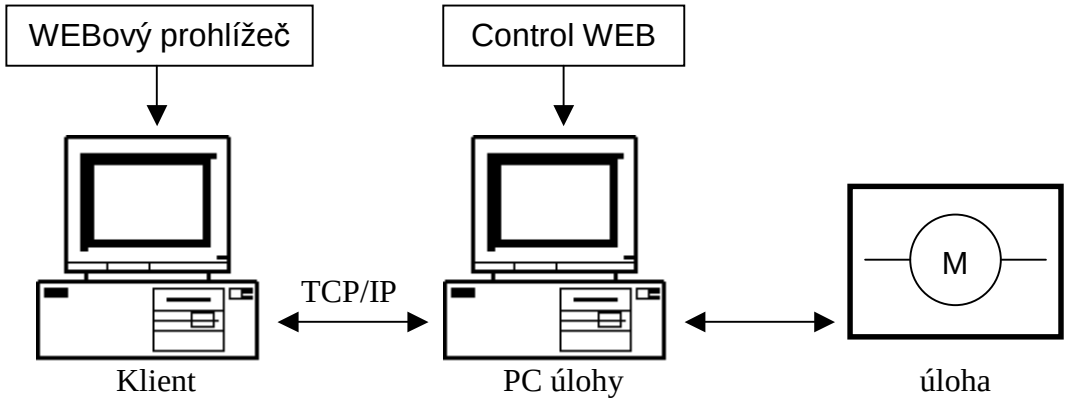
  řešení: na řešení jsem nepřišel, problém je zřejmě způsobem algoritmem převodu

4.6 PŘÍMÁ MĚŘENÍ Z WWW PROHLÍŽEČE

V tomto bodě bych se odvolal na diplomovou práci kolegy Libora Geislera. Původní myšlenka byla taková, že program Control WEB 2000 měl být umístěn společně s Matlab WEB serverem a klienti by si jej z něj spouštěli. Po neúspěšných pokusech takovýto systém zprovoznit jsme kontaktovali výrobce a ten nám sdělil, že to není možné. Pro samotný provoz tedy musíme mít k dispozici:

- zapnutou úlohu
- zapnutý počítač u systému (úlohy)
- nainstalovaný Control WEB na počítači u úlohy (odlišná licenční čísla)
- nainstalovaný WEBový prohlížeč na počítači u klienta

Obr.4.6-1 demonstruje schéma vzájemné spolupráce.



Obr. 4.6-1 *Schématické znázornění spolupráce jednotlivých částí při měření z WWW prohlížeče*

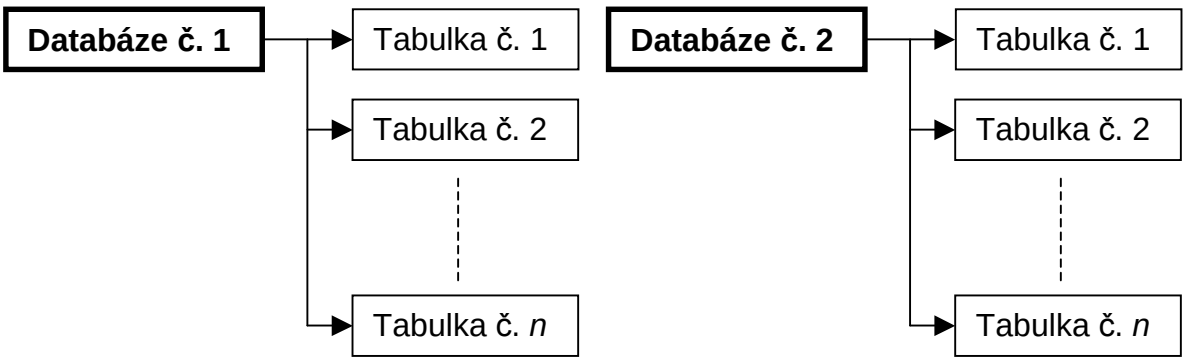
K ovládání úlohy je vytvořeno okno, s jehož pomocí zadáváme buzení, regulujeme soustavu a ukládáme obdržená data. Jeho vzhled si můžeme prohlédnout na obr.4.6-2.

obrázek od libora

Obr. 4.6-2 *Příklad okna uživatele ovládající systém z WWW prohlížeče*

4.7 ZÁKLADY JAZYKA SQL

SQL (Structured Query Language) je jazyk určený pro manipulaci a práci s daty umístěnými v databázi. Každá databáze se skládá z tabulek (viz. obr.4.7-1) ve kterých jsou uložená samotná data v předem definované struktuře vytvořené při



Obr.4.7-1 *Struktura databází*

zakládání tabulky. Před vytvořením vlastní databáze bychom měli pokud možno do největší podrobnosti analyzovat danou situaci a podle ní naložit s množstvím tabulek a jejich obsahem. Tato část se v literatuře označuje jako normalizace [14]. Skládá se ze sedmi stupňů, ale obvykle postačují první tři. V podstatě se jedná o to, aby se data zbytečně neopakovala a aby v jedné tabulce byli pouze ty informace, které k sobě nerozlučně patří. (Např. v nějaké firmě nebudeme míchat mezi sebou dodavatele s odběrateli nebo jeho kompletní data psát ke každému dodávanému zboží. Místo toho se odvoláme na přesně definovaný záznam v jiné tabulce, tak uspoříme místo v paměti a tím i rychlost.) Chceme-li databázi spravovat příkazy jazyka SQL, stačí nám velmi malé množství intuitivních instrukcí (je zvykem zapisovat je velkými písmeny, ale není to nutné) o kterých bych se nyní zmínil. Musí být ukončeny středníkem, jinak si program myslí, že příkaz není dosud kompletní. Můžeme je zapisovat přímo do konzole programu MySQL (zadáme-li v DOSovském příkazovém řádku: `C:\mysql\bin\mysql -p`) nebo jako parametr funkce PHP, čehož je v našem případě využíváno – viz. kap. 4.8.1.

4.7.1 Sloupce tabulky, datové typy, modifikátory polí, indexy

Každá tabulka se skládá ze sloupců (polí) a řádků. Řádky představují jednotlivé záznamy se kterými se pracuje a sloupce představující údaje každého záznamu (viz. obr. 4.7.1-1). Ty musíme při vytvoření tabulky nadefinovat a přiřadit jim vhodný datový typ. V následující tabulce tab.4.7.1-2 uvádím jejich výčet [14].

ID	**Jmeno**	**Prijmeni**	**RC**	**Pohlavi**
1	Jan	Novák	7901011234	M
2	Renata	Rychlá	7951011235	Z

Obr. 4.7.1-1 Struktura tabulky

Číselné typy:

Název typu	***Interval*** (od něj se přímo odvíjí i potřebný paměťový prostor)	
	se znaménkem	***bez znaménka***
TINYINT	-128 do 127	0-255
SMALLINT	-32768 do 32767	0-65535
MEDIUMINT	-8388608 do 8388607	0-16777215
INT	-2147483648 do 2147483647	0-4294967295
BIGINT	-9223372036854775808 do 9223372036854775807	0-18446744073709550615
FLOAT(M, D)	Liší se podle použitých hodnot.	
DOUBLE(M, D)	Liší se podle použitých hodnot.	
DECIMAL(M, D)	Liší se podle použitých hodnot.	

Textové typy:

Název typu	***Maximální velikost***
CHAR(X)	255 bajtů
VARCHAR(X)	255 bajtů
TINYTEXT	255 bajtů
TINYBLOB	255 bajtů
TEXT	65535 bajtů
BLOB	65535 bajtů
MEDIUMTEXT	1,6 MB
MEDIUMBLOB	1,6 MB
LONGTEXT	4,2 GB
LONGBLOB	4,2 GB

Poznámka: Datový typ CHAR se od VARCHAR liší tím, že má pevně danou délku. Je mu též v paměti přidělená paměť o jeden byte menší (u VARCHARU je zde umístěna jeho délka).

Různá pole:

Název typu	***Popis***	
	standardní formát	***nulová hodnota***
DATETIME	YYYY-MM-DD HH:MM:SS	0000-00-00 00:00:00
DATE	YYYY-MM-DD	0000-00-00
TIME	HH:MM:SS	00:00:00
YEAR	YYYY	0000
TIMESTAMP	mění se (viz. [14])	00000000000000 (nejdelší)
ENUM	výčtový typ (jediná možnost výběru)	
SET	výčtový typ (více možností výběru – až 64)	

Tab.4.7.1-2 *Výčet možných datových typů v MySQL*

K datovému typu můžeme dále přiřadit jeden nebo i více modifikátorů. Jejich výčet uvádím v následující tabulce tab.4.7.1-3 [14].

Název modifikátoru	***Lze použít u datového typu...***	***Popis***
AUTO_INCREMENT	všechny typy INT	automaticky zvyšovat o 1 (při překročení rozsahu začne od 1)
BINARY	CHAR, VARCHAR	binární data
DEFAULT	všechny kromě BLOB a TEXT	implicitní hodnota
NOT NULL	všechny typy	nenulový
NULL	všechny typy	nulový
PRIMARY KEY	všechny typy	primární klíč (lze použít pouze 1x)
UNIQE	všechny typy	unikátní
UNSIGNED	číselné typy	bez znaménka
ZEROFILL	číselné typy	vyplněný nulami

Tab.4.7.1-3 Modifikátory datových polí v MySQL

Pro zvyšování výkonu a tedy rychlejší vyhledávání a třídění dat se zakládají tzv. indexy. Index je datová struktura uložená v samostatném souboru a v každé tabulce jich můžeme použít max. 16 [14] (tomu bychom se v praxi při správném návrhu databáze měli vyhnout). Fungují jako abecední oddělovače. Hledáme-li informace například podle jména začínajícím na „Z“, nemusíme díky použití indexu ve sloupci jméno prohledávat zbytečně jména začínající zbývajícími písmeny. Naopak jeho nevhodným použitím můžeme dojít k opačnému efektu (např. neindexujeme pouze jediné písmeno, ale celé jméno). Dále přidáváme-li záznamy do indexované tabulky, je celý proces pomalejší. Tento nástroj tedy používáme především pro sloupce, na které se odvoláváme nejčastěji. Automaticky je indexován primární klíč tabulky. (Klíč je tabulka sloužící k dalšímu urychlení prohledávání tabulky a k jednoznačné identifikaci záznamů, viz. [14].)

4.7.2 Vybrané základní příkazy jazyka SQL

Přístup k databázi

`SHOW DATABASES;`	vypíše přístupné databáze
`USE Samoobsluha;`	použije databázi samoobsluha
`SHOW TABLES FROM Samoobsluha;`	vypíše tabulky databáze Samoobsluha
`SHOW COLUMNS FROM Dodavatele;`	vypíše sloupce tabulky Dodavatele

Vytvoření databáze, tabulky a indexu

`CREATE DATABASE OvoceAZelenina;`	vytvoří databázi

`CREATE TABLE Zbozi (`
`  ID INT NOT NULL PRIMARY KEY AUTO_INCREMENT,`
`  Nazev VARCHAR(30),`
`  Cena SMALLINT NOT NULL,`
`  IDDodavatel VARCHAR(30)`
`);`
	vytvoří tabulku

`CREATE INDEX ICena ON Zbozi (Cena);`	vytvoří index se jménem ICena v tabulce Zbozi pro sloupec Cena (do závorky můžeme uvést více polí oddělených čárkou)

Odstranění databáze, tabulky a indexu

`DROP DATABASE OvoceAZelenina;`	odstraní databázi OvoceAZelenina
`DROP TABLE Zbozi;`	odstraní tabulku Zbozi
`DROP INDEX ICena ON Zbozi;`	odstraní index ICena z tabulky Zbozi

Modifikace tabulky a indexu

`ALTER TABLE Zbozi RENAME TbZbozi;`	přejmenování tabulky Zbozi na TbZbozi
`ALTER TABLE Zbozi CHANGE Nazev` `  Jmeno VARCHAR(40);`	změna jména a datového typu sloupce Nazev v tabulce Zbozi na Jmeno VARCHAR(40)
`ALTER TABLE Zbozi DROP Cena;`	vymazání sloupce Cena z tabulky Zbozi
`ALTER TABLE Zbozi ADD Cena;`	přidání sloupce Cena do tabulky Zbozi

Tvorba dotazů

`SELECT * FROM Zbozi;`	vypíše všechny položky tabulky Zbozi se všemi údaji o každém záznamu
`SELECT Nazev FROM Zbozi;`	vypíše sloupec Nazev z tabulky Zbozi
`SELECT Nazev, Cena FROM Zbozi;`	vypíše sloupce Nazev a Cena z tab. Zbozi
`SELECT Nazev FROM Zbozi WHERE Cena>100;`	vypíše názvy výrobků s cenou vyšší 100
`SELECT Nazev FROM Zbozi WHERE Cena>100 AND Nazev LIKE "A%";`	vypíše názvy výrobků s cenou vyšší 100 a začínající na A
`SELECT * FROM Zbozi LIMIT 10;`	vypíše prvních deset záznamů
`SELECT * FROM Zbozi LIMIT 20,10;`	vypíše záznamy 20-30
`SELECT Nazev, CENA FROM Zbozi ORDER BY Nazev;`	vypíše názvy a ceny výrobků alfanumericky seřazené podle Nazev

Přidávání, aktualizace a mazání záznamů

`INSERT INTO Zbozi VALUES( NULL, "Grep", 10, "Fruit a.s." );`	přidá do tabulky Zbozi položku s uvedenými hodnotami
`INSERT INTO Zbozi(Nazev) VALUES ("Jablko");`	přidá do tabulky nový záznam, kde je vyplněno pouze pole Nazev
`INSERT DELAYED INTO Zakaznici (Zbozi) VALUES ("Hruška");`	přidá nový záznam až v případě, že se s tabulkou nepracuje
`UPDATE Zbozi SET Jmeno="Grepfruit" WEHRE Nazev= "Grep";`	změní název Grep na Grepfruit
`UPDATE LOW_PRIORITY Zbozi SET Jmeno="Grepfruit" WEHRE Nazev= "Grep";`	změní název Grep na Grepfruit v době, kdy se s tabulkou nepracuje
`DELTE FROM Zbozi WHERE IDDodavatel="Fruit a.s.";`	odstraní položky, kde je ve sloupci IDDodavatel uvedeno Fruit a.s.
`DELTE LOW_PRIORITY FROM Zbozi WHERE IDDodavatel="Fruit a.s.";`	odstraní položky v době, kdy se s tabulkou nepracuje
`DELTE FROM Zbozi WHERE IDDodavatel="Fruit a.s." LIMIT=10;`	odstraní 10 položek splňující kritérium

4.7.3 Demonstrující ukázka programu v MySQL

Jako jednoduchý příklad jsem vybral adresář osob. Databáze se bude skládat pouze z jedné tabulky a každý záznam bude obsahovat: ID, jméno, příjmení, pohlaví, adresu, mobilní telefon, ročník narození a poznámku.

```
CREATE DATABASE Adresar;            vytvoření databáze jménem Adresar
USE Adresar;                        použij databázi Adresar
CREATE TABLE Osoby (                vytvoření tabulky
   ID SMALLINT NOT NULL
      PRIMARY KEY AUTO_INCREMENT,
   Jmeno VARCHAR(15),
   Prijmeni VARCHAR(15),
   Pohlavi ENUM("M","Z") DEFAULT "M",
   Adresa VARCHAR(100),
   Mobil CHAR(10),
   Rocnik TINYINT,
   Poznamka TEXT
);
```

Nyní již můžeme do tabulky vkládat záznamy. Uvedu jeden z možných způsobů. Všimněte si, že neuvádíme ID (automaticky se zvyšuje) a že v případě neznalosti některého parametru jej jednoduše vynecháme.

```
INSERT INTO Osoby(Jmeno, Prijmeni, Pohlavi, Mobil, Rocnik, Poznamka)
   VALUES(
   "Jan", "Novak", "M", "0604123456", 70, "zajímá se o automobily"
);
```

Po naplnění tabulky můžeme začít pokládat dotazy dle potřeby.

```
SELECT * FROM Osoby;                vrátí kompletní záznamy
SELECT Prijmeni FROM Osoby          vrátí příjmení všech žen
   WHERE Pohlavi="M";
SELECT * FROM Osoby LIMIT 10;       vrátí prvních deset záznamů
```

Zastaralá data můžeme aktualizovat nebo mazat příkazy `UPDATE` a `DELETE`. V případě potřeby tabulku lze samozřejmě rozšiřovat o další záznamy či dokonce nové vlastnosti výše uvedenými příkazy a mnoha dalšími zde nezmíněnými.

4.8 ZÁKLADY JAZYKA PHP

Jak už je uvedeno výše, PHP je skriptovací jazyk na straně serveru. Klient tedy k němu nemá přístup k zdrojovému textu jako je tomu například u HTML nebo JavaScriptu a soubory se musí spouštět přes WEBovou adresu, nemohou se spouštět přímo z disku jako stránky HTML. Klient vidí pouze výsledek programu, nikoliv však vlastní algoritmus. Soubory pracující s PHP mývají přípony: `php`, `php3`, `php4`, `phtml`, ale může jej využívat i `htm` / `html`.

Stránky se tvoří stejně jako ostatní s tím rozdílem, že vlastní PHP kód se odliší od zbytku uzavřením mezi značky `<?php` *program* `?>` nebo `<script language="php">` *program* `</script>`. Dále si musíme zvyknout na to, že veškeré proměnné musí začínat znakem dolaru "`$`", jinak se vyhodnotí jako funkce (např. `$a=$b+10;`) PHP je citlivé na velká a malá písmenka (key sensitive), tedy `$a` je jiná proměnná než `$A` a každý příkaz musí končit středníkem. Pokud je stránka využívaná pro zpracování odesílaných dat, tak k nim máme okamžitě přístup se jménem shodným s vlastností `name`, uvedeným ve nějakém vstupu odesílaného formuláře. Např. v něm máme `<input name="Jmeno" type="text">` a pak jsou přijatá data v proměnné `$Jmeno`. Poznámku vytvoříme tak, že na před text vložíme dvě lomítka (`//`) nebo blok textu uzavřeme mezi `/* text */`. Většina funkcí vrací číslo, které můžeme dále zpracovat. V případě neúspěchu 0 – nepravda, jiné číslo – pravda. Hlášení o případné chybě zabráníme vložením znaku zavináče "`@`" před funkci.

Potřebujeme-li změnit proměnnou typem `$a = $a + 1;`, lze tak učinit jednodušeji `$a++;`. Obdobně také `$a--;`. Potřebujeme-li změnit proměnnou typem `$a = $a + $b;`, lze tak učinit jednodušeji `$a += $b;`. Podobné zjednodušené zápisy mající teď již zjevný význam jsou: `$a -= $b; $a *= $b; $a /= $b; $a .= $b;`. Pro symbol negace se používá znak vykřičníku "`!`".

Proměnná může také reprezentovat výčtové pole (indexované od nuly). K jeho položkám přistupujeme pomocí hranatých závorek. (Např. na první položku pole `$a` se dostane takto: `$a[0];`.) Můžeme vytvořit i asociativní pole, viz. [5].

4.8.1 Vybrané základní příkazy jazyka PHP

Pro popis příkazů byla použita literatura [5] a kompletní manuál. Přesný popis všech funkcí (v některých případech i v češtině) můžete nalézt na WEBové adrese http://www.php.net/ nebo jeho zrcadlu http://www.php.cz/ ve formátu `html`, `txt`, `php` a `chm`. Budou uváděny konkrétní příklady, neboť daleko lépe objasní syntaxi příkazu než obecný zápis.

Výpis

`echo($text);` — vypsání textu (musíme mít na paměti, že vypsaný text se bude chovat podle pravidel zobrazování HTML, můžeme tedy s výhodou použít jeho formátovací značky; jednotlivé řetězce (texty) spojujeme pomocí tečky a proměnné můžeme přímo vkládat do textu bez jeho přerušení, např. je možné napsat: `echo("Výsledek součtu proměnné <i>a</i> a <i>b</i> je: ".($a+$b));`

`$a=crypt($a,11);` — šifrování textu metodou DES, druhým argumentem je libovolný dvojznak sloužící k založení klíče; nerozluštitelný

Vkládání souborů

`include("data.php");` `require("data.php");` — datový soubor se vyhodnocuje zvlášť jakoby podprogram příkaz je nahrazen obsahem datového souboru

Smyčky

```
for($i=0;$i<10;$i++) {
     // příkazy;
}
```

smyčka, která se 10x opakuje:
`$i=0` nám nastaví počáteční hodnotu testované proměnné
`$i<10` smyčka pokračuje, pokud tato podmínka je splněna
`$i++` třetí parametr nám říká, co se má provést v každém průběhu cyklu smyčky (ve smyčce je vhodné `$i` pouze číst)

```
while($i<10) {
     // příkazy;
}
```

smyčka se opakuje v případě, pokud platí podmínka `$i<10`

Větvení programu

```
if($a<10) {
     // příkazy1;
} elseif ($a<20) {
     // příkazy2;
} else {
     // příkazy3;
}
```

pokud je splněna podmínka `$a<10`, provedou se `příkazy1`, nevyhovuje-li, testujeme podmínku `$a<20` pro vykonání sady `příkazy2` a ve zbývajících případech se vykonají `příkazy3`; části `elseif` a `else` nemusíme použít, `elseif` lze opakovat v libovolném množství

```
switch ($a) {
     case "Ano": {
          // příkazy1;
          break;
     }
     case "Ne": {
          // příkazy2;
          break;
     }
     default: {
          // příkazy3;
          break;
     }
}
```

jako testovací podmínku bereme výraz uvedený v závorce za příkazem `switch` a program pokračuje tam, kde výraz u `switch` odpovídá výrazu u `case`; pokud ani jedna z možností nevyhovuje, je vybrána větev `default` (není povinná)

Práce se soubory – parametry pro otevírání souborů:

`a` otevře soubor pouze pro přidávání na konec souboru, neexistující soubor bude vytvořen

`a+` otevře soubor pro přidávání a čtení z něj, přidávání se bude provádět na konci

`r` otevře soubor pouze pro čtení

`r+` otevře soubor pro čtení a zápis, data budou zapsána na začátek souboru

`w` otevře soubor pouze pro zápis, obsah souboru bude vymazán, neexistující soubor bude vytvořen

`w+` otevře soubor pro zápis a čtení, obsah souboru bude vymazán, neexistující soubor bude vytvořen

```
$soubor=fopen("data.txt","r");
```

funkce otevře soubor s parametrem `r` a vrátí identifikátor souboru sloužící k výkonu dalších funkcí

```
fclose($soubor);
```

zavře soubor

```
unlink($soubor);
```

vymaže soubor

`fpassthru($soubor);`	vypíše obsah souboru
`fgetc($soubor);`	načtení jednoho znaku ze souboru
`fgets($soubor, $x);`	načtení `$x` znaků ze souboru
`fgetss($soubor, $x);`	viz. fgets, ale jsou odstranění HTML značky
`fputs($soubor, $text, $x);`	zapsání `$x` znaků řetězce `$text` do souboru
`rewind($soubor);`	přesun pozice v souboru na začátek
`fseek($soubor, $x);`	přesun v souboru na pozici `$x`
`$pos=ftell($soubor);`	vrací aktuální pozici v souboru
`feof($soubor);`	vrací pravdu, pokud jsme dosáhli konce souboru
`copy("zdroj.txt","cil.txt");`	kopírování souborů
`file_exists("data.txt");`	vrací pravdu v případě existence souboru
`filetype("data.txt");`	vrací atributy souboru „fifo“ FIFO „char“ speciální znakové zařízení „dir“ adresář „block“ speciální blokové zařízení „link“ symbolický odkaz „file“ normální soubor „uknown“ nelze určit
`$pole=fgetcsv($soubor,$x,";");`	načtení souboru typu *csv* s oddělovačem ";" a maximální délky `$x` do pole

Práce s adresáři

`chdir("C:/");`	změna aktuálního adresáře
`$adr=opendir("..");`	vrátí identifikátor aktuálního adresáře, případně zvolíme cestu k jinému
`readdir($adr);`	vrací název další položky v adresáři
`rewinddir($adr);`	vracení na začátek adresáře
`mkdir("C:/novy", 0700);`	vytvoří nový adresář, druhým parametrem specifikuje přístupová práva k unixovému

	adresáři (ve Windows ignorováno)
`rmdir($adr);`	odstranění adresáře

Regulární výrazy

Jedná se o daleko složitější vyhodnocovací operace, než funkce uvedené výše. Pro jejich pochopení bych opět odkázal na literaturu [5]. Budeme-li ovšem využívat jejich velmi silných nástrojů, můžeme velmi jednoduše testovat obsah řetězců (např.: zda odpovídají e-mailové adrese, obsahují-li alfanumerické znaky a to na počátku či na konci řetězce a v libovolné délce). Dále s nimi můžeme nahrazovat pasáže textu jinými, dělit řetězec na pole podle zadaného kritéria atd.

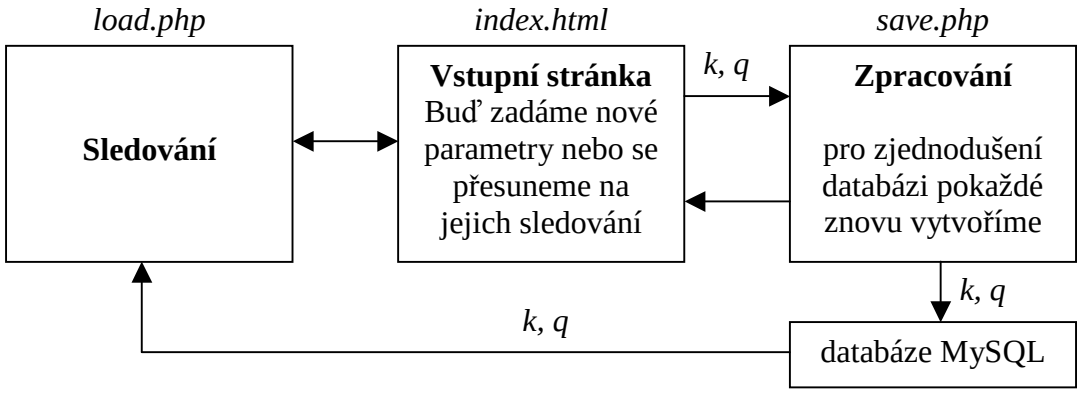
Spolupráce s MySQL

`$id_spojeni=mysql_connect( $hostitel,$jmeno,$heslo);`	vytváří spojení se serverem MySQL, spojení zaniká v okamžiku konce PHP kódu
`$id_spojeni=mysql_pconnect( $hostitel,$jmeno, $heslo);`	vytváří trvalé spojení se serverem MySQL
`mysql_close($id_spojeni);`	uzavření spojení se serverem MySQL
`mysql_select_db("pracovnici");`	výběr databáze nazvané pracovnici
`mysql_create_db("pracovnici");`	vytvoření databáze nazvané pracovnici
`mysql_drop_db("pracovnici");`	odstranění databáze
`$dotaz=mysql_query($sql);`	odešle dotaz `$sql` serveru MySQL
`mysql_errno();`	číslo případně vzniklé chyby
`mysql_error();`	popis případně vzniklé chyby
`$pocet=mysql_num_rows($dotaz);`	vrací počet řádků vráceného dotazu
`$v_pole=mysql_fetch_row($dotaz);`	převede dotaz na výčtové pole např.: `$v_pole[0];` ...
`$a_pole=mysql_fetch_array($dotaz);`	převede dotaz na asociativní pole např.: `$a_pole["Jmeno"];` ...
`$pocet=mysql_affected_rows($dotaz);`	vrátí počet ovlivněných řádků od dotazů INSERT, UPDATE nebo DELETE

Doinstalováním dalších modulů PHP rozšíříme jeho možnosti libovolným směrem.

4.8.2 Demonstrující ukázka programu v PHP

Předpokládejme, že by měl jeden člověk ukládat databáze MySQL například parametry přímky (*k*, *q* kde *y* = *kx* + *q*) a druhý člověk by měl tyto data opakovaně automaticky načítat v intervalu 5 sekund. Tento problém můžeme vyřešit následující strukturou uvedenou na obr.4.8.2-1.



Obr.4.8.2-1 *Navržené schéma řešení s toky dat*

Výpis souboru *index.html*

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
	<title>Vstupní stránka</title>
</head>
<body>
<h3>y = kx + q</h3>
<form action="save.php" method="post">
	k: <input type="text" name="k"><br>
	q: <input type="text" name="q"><br>
	<br><input type="submit" value="odešli">
</form><br><br>
<a href="load.php">sledování výsledků</a>
</body>
</html>
```

Výpis souboru *save.php* (název databáze: RealTime, název tabulky: Data, pro zjednodušení případnou starou databázi smažeme a vytvoříme novou a netestujeme, zda *k* a *q* jsou opravdu číslice).

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
	<title>Zpracování</title>
	<META HTTP-EQUIV="Content-Type" CONTENT="text/html;
	CHARSET=Windows-1250">
```

```
</head>
<body>
<table><?php
      if($id_spojeni=mysql_connect("127.0.0.1","",""))
            echo("<tr><td>Připojení k databázi:<td>".$id_spojeni);

      $sql="DROP DATABASE RealTime";
      $vysledek=mysql_query("$sql");
      if($vysledek=mysql_query("$sql"))
            echo ("<tr><td>Odstranění staré databáze:<td>" .
            $vysledek);

      $sql="CREATE DATABASE RealTime";
      $vysledek=mysql_query("$sql");
      if($vysledek=mysql_query("$sql"))
            echo ("<tr><td>Vytvoření nové databáze:<td>" .
            $vysledek);

      if($id_nastaveni=mysql_select_db("RealTime"))
            echo("<tr><td>Výběr databáze:<td>".$id_nastaveni);

      $sql="CREATE TABLE Data (k INT, q INT)";
      $vysledek=mysql_query("$sql");
      if($vysledek=mysql_query("$sql"))
            echo ("<tr><td>Vytvoření tabulky:<td>" .
            $vysledek);

      $sql="INSERT INTO Data (k, q) VALUES ($k, $q)";
      if($vysledek=mysql_query("$sql"))
            echo ("<tr><td>Zapsání parametrů do tabulky data:<td>" .
            $vysledek);
      ?></table>
<br><br>...data zapsána<br><br>
<a href="index.html">zpět</a><br>
</body>
</html>
```

Výpis souboru *load.php*

```
<?php header("Cache-Control: no-cache, must-revalidate"); ?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
      <title>Sledování</title>
</head>
<body>
<h3>y = kx + q</h3>
<table><?php
      if($id_spojeni=mysql_connect("127.0.0.1","",""))
            echo("<tr><td>Připojení k databázi:<td>".$id_spojeni);
      if($id_nastaveni=mysql_select_db("RealTime"))
            echo("<tr><td>Výběr databáze:<td>".$id_nastaveni);
      $sql="SELECT * FROM Data";
      if($vysledek=mysql_query("$sql"))
      $pole=mysql_fetch_array($vysledek);
      echo("<tr><td>k: ".$pole["k"]."<td>q: ".$pole["q"]);
```

```
?></table><br>
<a href="index.html">Vstupní stránka</a>
<script language="JavaScript">

function Obnova() {
     window.location.replace('load.php?t=<?php echo(time()); ?>');
     setTimeout("Obnova()", 5000);
}
setTimeout("Obnova()", 5000);

</script>
</body>
</html>
```

Program pracuje následovně. Zadané parametry (v souboru *index.html*) odešleme ke zpracování (soubor *save.php*). Zde vymažeme starou databázi (pro zjednodušení netestujeme, zda nějaká stará vůbec existovala) a vytvoříme novou společně s tabulkou, do které zapíšeme přijaté parametry přímky. Vrátíme se k úvodnímu formuláři a spustíme okno pro načítání dat (soubor *load.php*). Po uplynutí 5 vteřin se stránka automaticky obnoví (je zajištěno, aby se pokaždé neuložila do cache paměti počítače) a proměnné *k* a *q* načte z tabulky. Protože se databáze po vložení nových údajů maže, nemusíme se starat o to, abychom načítali poslední vložené parametry.

Takovýto princip je aplikován i u chatu. Parametry všech uživatelů jsou uloženy v jediné tabulce. V momentě, kdy někdo vstupuje do místnosti, je pro něj vytvořena nová, do které se ukládá text určený jenom pro něj. Podle nastavení je zobrazováno pouze posledních x záznamů a v historii jsou přístupné všechny najednou.

5. MANUÁLY

5.1 UMÍSTĚNÍ HYPERTEXTOVÝCH DOKUMENTŮ

AR1: http://www.fm.vslib.cz/~krtsub/fm/tr1/materialy.htm
AR2: http://www.fm.vslib.cz/~krtsub/fm/tr2/materialy.htm
CR: http://www.fm.vslib.cz/~krtsub/fm/cir/materialy.htm
MWS: http://krt32.krt.vslib.cz/

5.2 TISK

Pro vlastní studium je zřejmě pro každého pohodlnější pracovat s materiály v tištěné podobě. Jednak nemusíme mít vždy po ruce počítač, můžeme si do něj zapisovat poznámky, rychleji listovat a i ze zdravotního hlediska není příliš dobré oči unavovat dlouhým pohledem na monitor, pokud obrazovka není vybavena LCD displayem.

5.2.1 Tisk semestrálních prací

V oknu vygenerovaného zadání zvolíme v menu **Soubor** -> **Tisk...** Můžeme též použít kombinaci kláves Ctrl+P.

5.2.2 Tisk studijních materiálů

Otevřeme stránku kliknutím na příslušný odkaz a dále stiskneme ikonku tiskárny Acrobata Readra obr.5.2.2-1. (Ne WEBového prohlížeče!)

Obr.5.2.2-1 *Ikonka tiskárny Acrobata Readra*

ZÁVĚR

Byl vytvořen provázaný systém zaměřený na vzdělávání. Můžeme jej rozdělit do několika částí:

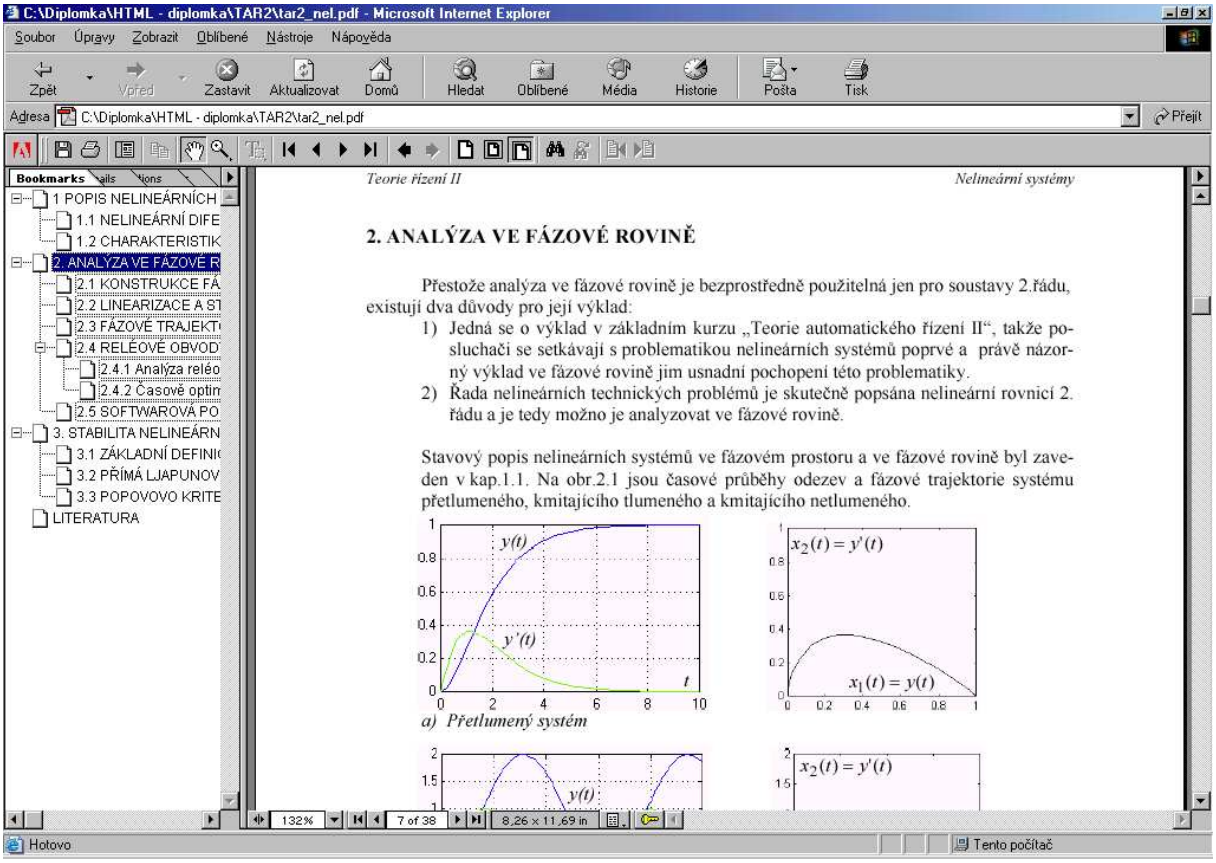
- základní jádro: studijní materiály, návody k laboratorním úlohám, autotesty (kvizy), kontrolní otázky, příklady z MWS,
- informace pro studenty: program přednášek a cvičení, chat a
- pedagogické pomůcky: zautomatizované zadávání semestrálních prací s kontrolou vlastního zadání.

Pro velký rozsah diplomové práce je naplněna struktura tab.2-1a pouze pro předměty AR1 a AR2. Nebude těžké zbývající část dodělat, neboť většina podkladů je před dokončením. Navíc je vytvořena celá struktura i se zázemím. V průběhu školního roku se navíc ukázalo na reakcích studentů, že koncepce je namířena správným směrem a v dalších letech bude dále vylepšována.

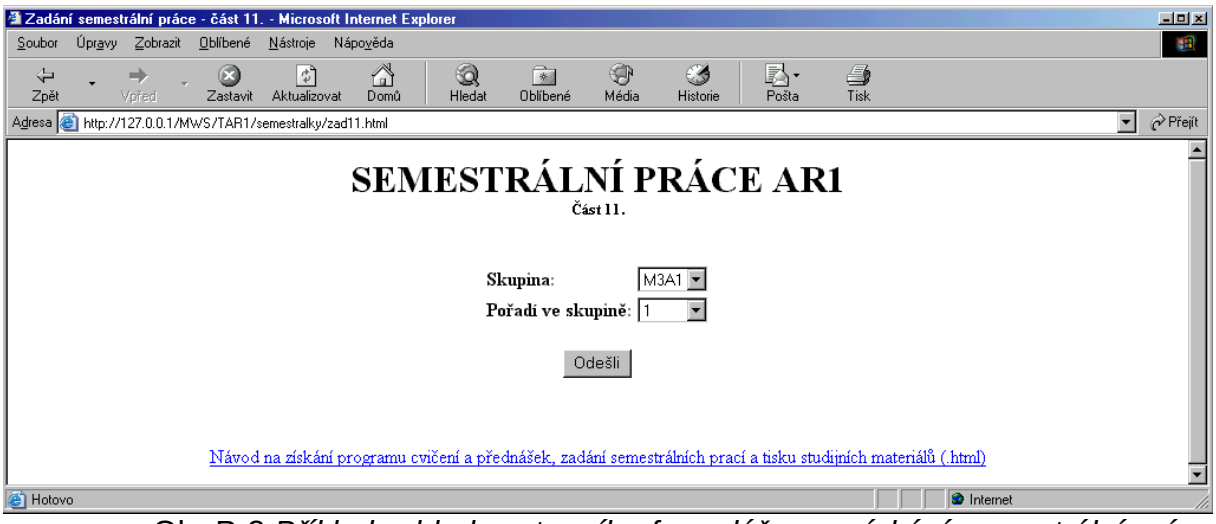
LITERATURA

[1] FRONTZ, T.: Výuka podporovaná počítačem TkATEX V1.2 – Závěrečná práce. [Diplomová práce]. Liberec: TU, Fakulta Pedagogická, 1998.
[2] SLAVÍČEK, P., MAZÁK, E.: Základy pedagogicko-psychologických znalostí pro uživatele počítačové výuky a didaktické techniky. Praha: ČVUT, Výzkumný ústav inženýrského studia, 1990.
[3] EISENMENGER, R.: JavaScript kompletní kapesní průvodce. Praha: Grada Publishing, 1999. ISBN 80-7169-383-9.
[4] SCOTT, I.: Dynamické HTML. Praha: Computer Press, 2000. ISBN 80-7226-268-8.
[5] CASTAGNETTO, J., RAWAT, H., SCHUMANN, S., SCOLLO CH., VELIATH, D.: PHP Programujeme profesionálně, Praha: Computer Press, 2002. ISBN 80-7226-310-2
[6] KYRAL, A.: Vypracování jednoduchého hypertextového dokumentu vybraného tématu pro potřebu výuky v předmětu TAŘ I, II. [Projekt]. Liberec: TU, 2001.
[7] MODRLÁK, O.: Úvod do diskrétní parametrické identifikace. Studijní materiál [online]. Liberec: TU, 2001 [cit. květen 2002]. Dostupné na WWW: http://www.fm.vslib.cz/~krtsub/fm/tr2/tar2_did.pdf.
[8] MODRLÁK, O., KYRAL, A.: Úvod do identifikace. Studijní materiál [online]. Liberec: TU, 2001 [cit. květen 2002]. Dostupné na WWW: http://www.fm.vslib.cz/~krtsub/fm/tr1/tar1_zid.pdf.
[9] MODRLÁK, O., KYRAL, A.: Fuzzy řízení a regulace. Studijní materiál [online]. TU Liberec, 2001 [cit. květen 2002]. Dostupné na WWW: http://www.fm.vslib.cz/~krtsub/fm/tr2/tar2_fuz.pdf.
[10] MODRLÁK, O.: Základy řízení ve stavovém prostoru. Studijní materiál [online]. Liberec: TU, 2001 [cit. květen 2002]. Dostupné na WWW: http://www.fm.vslib.cz/~krtsub/fm/tr2/tar2_zas.pdf.
[11] MODRLÁK, O., VOTRUBEC, R.: Laboratorní návody pro předmět Teorie řízení I. Studijní materiál [online]. Liberec: TU, 2001 [cit. květen 2002]. Dostupné na WWW: http://www.fm.vslib.cz/~krtsub/fm/tr1/In-Out.pdf.
[12] GRACE, A., LAUB, J.A., LITTLE, J.N., THOMPSON, C.M.: Control Systém Toolbox. For Use with MATLAB. User‘s Guide. The Math Works Inc., 1995.
[13] RUSSELL Ch.: Macromedia Flash 5 pro pokročilé. Praha: Computer Press, 2001. ISBN 80-7226-516-9
[14] MASLAKOWSKI M.: Naučte se MySQL za 21 dní. Praha: Computer Press, 2001. ISBN 80-7226-448-6

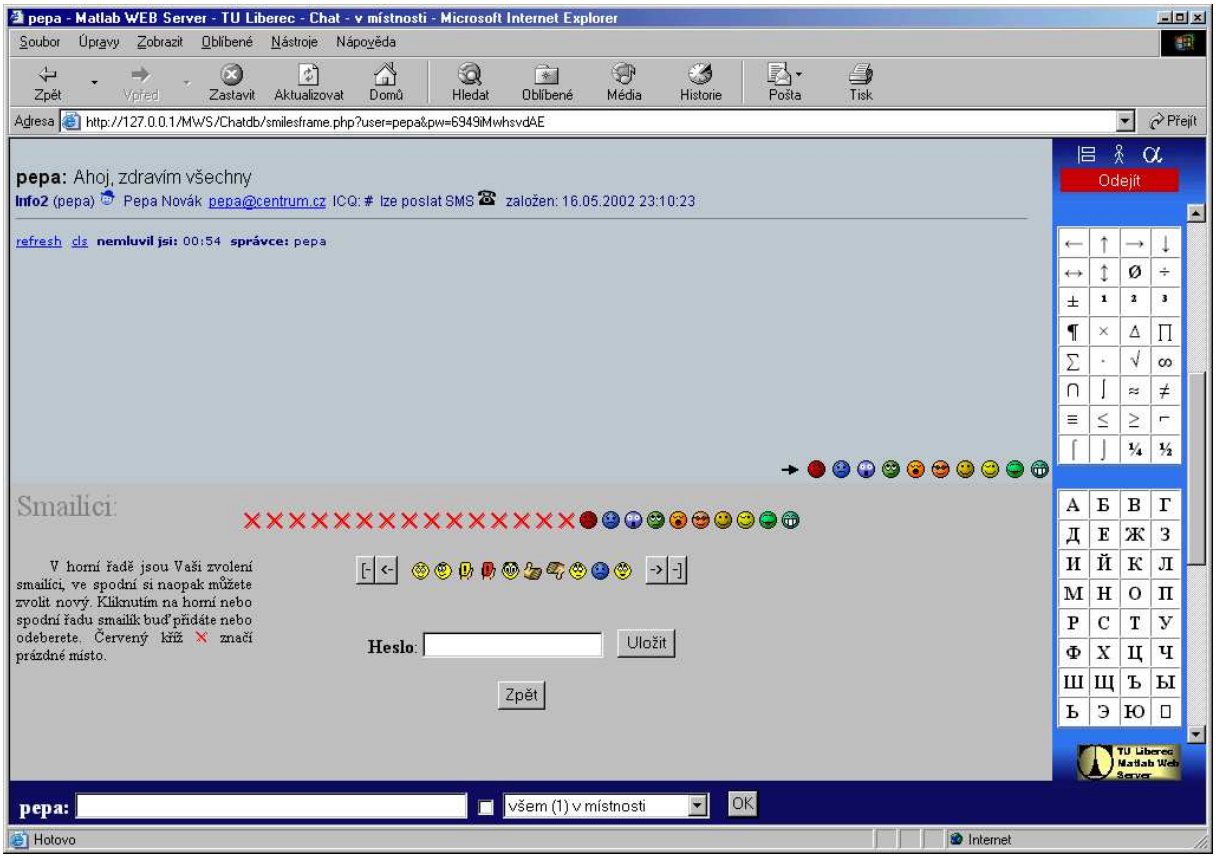
PŘÍLOHY



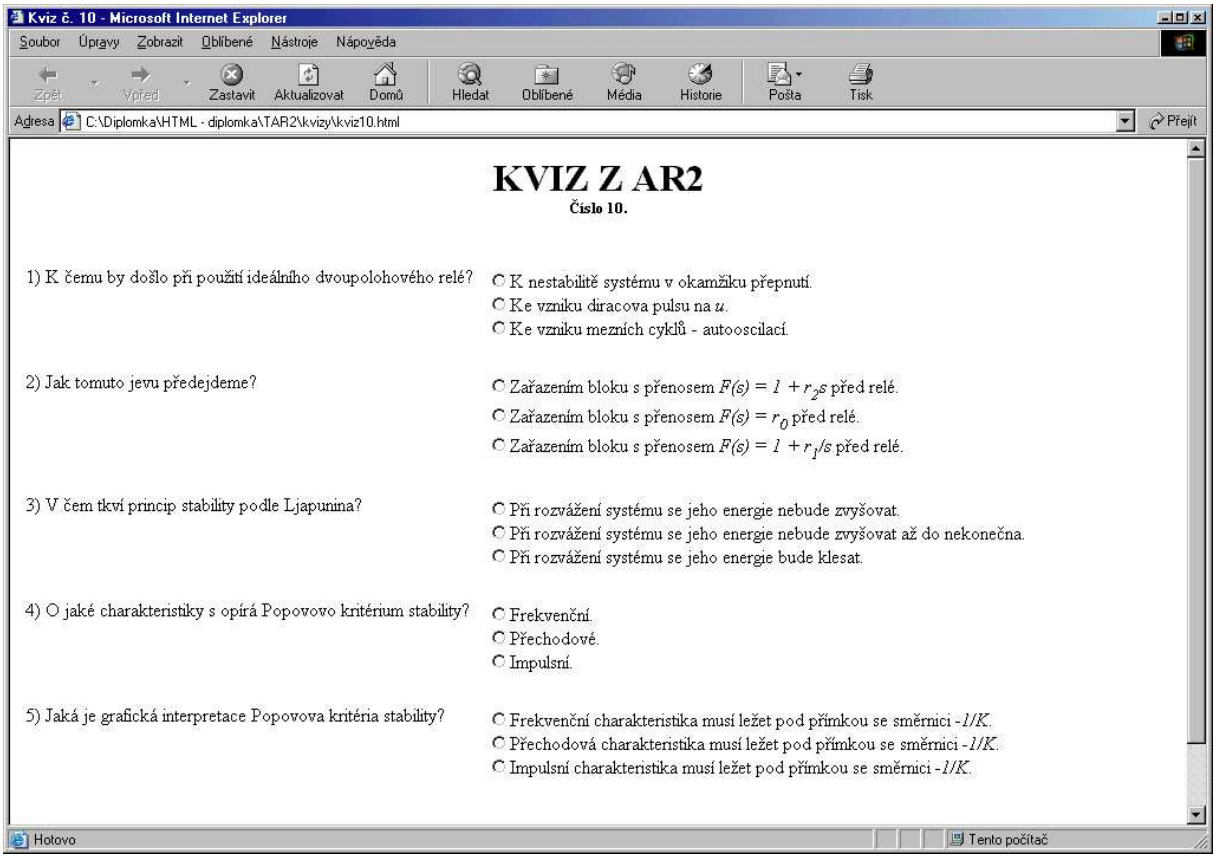
Obr.P-1 *Příklad vzhledu studijních materiálů*



Obr.P-2 *Příklad vzhledu vstupního formuláře pro získání semestrální práce*



Obr.P-3 *Vzhled chatu*



Obr.P-4 *Vzhled autotestu (kvizu)*