Portál pre odborné publikovanie ISSN 1338-0087

Využitie open technológií pre účely virtuálneho laboratória

Magyar Zoltán · Informačné technológie

18.03.2011

Cieľom projektu bolo vytvoriť web portál, ktorý by mohol slúžiť na on-line interaktívnu výučbu. Snahou je nahradiť proprietárny softvér, ktorý sa bežne používa vo výchovno-vzdelávacom procese, softvérom ktorý sa zakladá na otvorených technológiách. Pri tvorbe sme použili opensource matematické aplikácie (Scilab, OpenModelica, Maxima), ktoré tvoria jadro. Ďalej sme použili softvéry pre spojazdnenie servera (Ubuntu, JailKit, Denyhosts, Apache) na vytvorenie LiveDVD (Remastersys). Pre tvorbu webového používateľského rozhrania sa tiež využilo široké spektrum webových technológií (PHP, Python, Perl, XHTML, CSS, JavaScript, Jquery, AJAX, Flash, SVG, Flot, JsMath).

1. Opis vytvorenej webovej stránky

Pokiaľ si otvoríte v prehliadači stránku ktorá je momentálne na adrese http://147.175.125.77, ako prvé sa Vám objaví úvodná obrazovka, na ktorej máte 4 možnosti. Každá z týchto možností Vás presunie na stránku, ktorá umožňuje simulácie, prípadne matematické výpočty.

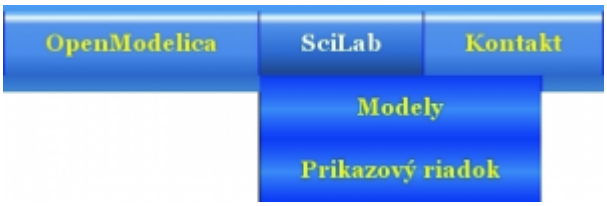
Obr. 1 – Ukážka úvodnej obrazovky webrozhrania obsluhujúceho Maximu, OpenModelicu a SciLab

Týmito možnosťami sú:

- Maxima

- OpenModelica
- Scilab

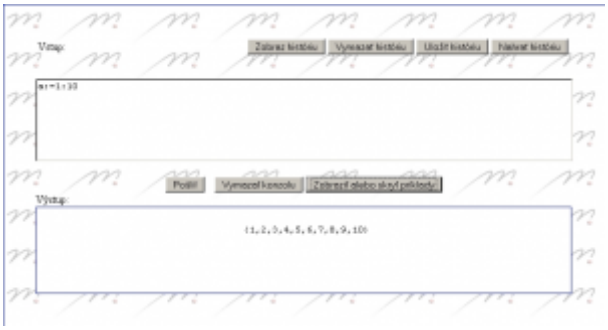
Keď sa rozhodnete prejsť na niektorú z týchto troch možností, objaví sa Vám dropdown menu.



Obr. 2,3,4 – Obrázky znázorňujúce možnosti výberu obsluhy jednotlivých aplikácií

Pri všetkých troch matematických softvéroch je možná voľba príkazového riadka, v ktorom sa do horného okna zadávajú postupne príkazy, pričom sa nám odozva vyobrazuje v spodnom okne. Takto má používateľ možnosť si vyskúšať naostro hociktorý z troch matematických softvérov bez toho aby si ich musel inštalovať doma na počítač. V prípade príkazových riadkov sú zakázané nebezpečné príkazy, ktoré môžu ovplyvniť bezpečnosť celej aplikácie (napr.: formátovací príkaz). Používateľ, ktorý nemá skúsenosti s použitými aplikáciami si môže otvoriť zbierku príkladov, pomocou ktorých ľahšie pochopí syntax jednotlivých matematických softvérov.

Obr. 5 – Príkazový riadok pre Maximu so zadaným príkazom a jeho výsledkom



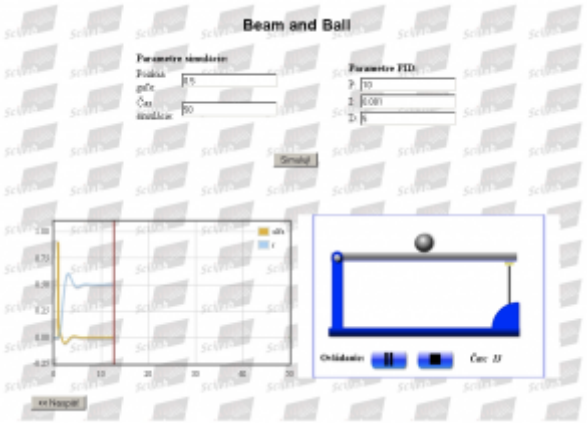
Obr. 6 – Príkazový riadok pre OpenModelicu

Obr. 7 – Príkazový riadok pre Scilab

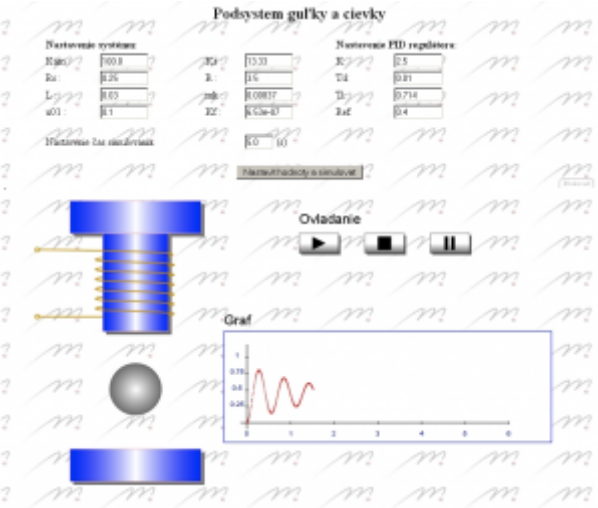
Používateľa môže potešiť možnosť zobrazenia histórie zadaných príkazov, tiež si ju môže uložiť do súboru, načítať či prípadne vymazať. Táto funkcia je užitočná pri nadefinovaných premenných, ktoré po skončení práce si môžeme odniesť so sebou.

OpenModelica a Scilab majú aj možnosť simulovania určitých predpripravených modelov, ktorým je možné upravovať parametre regulácie, pozorovať animáciu a grafický priebeh vývoja premenných týchto modelov počas procesu simulácie.

Obr. 8 – OpenModelica – model Beam and Ball – Príklad simulácie pohybu guličky na tyči na udržanie guličky na žiadanej pozícii

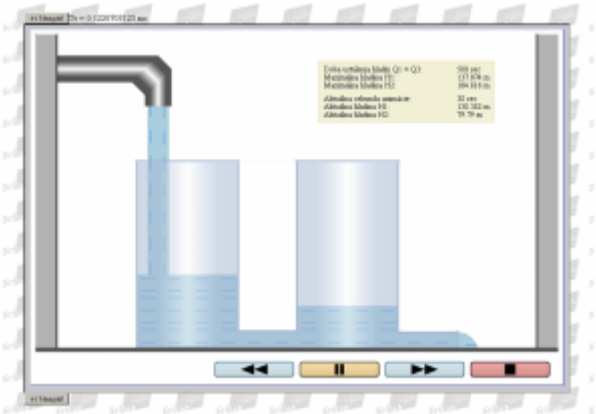


Obr. 9 – Scilab – Beam and Ball – Príklad simulácie pohybu guličky na tyči na udržanie guličky na žiadanej pozícií



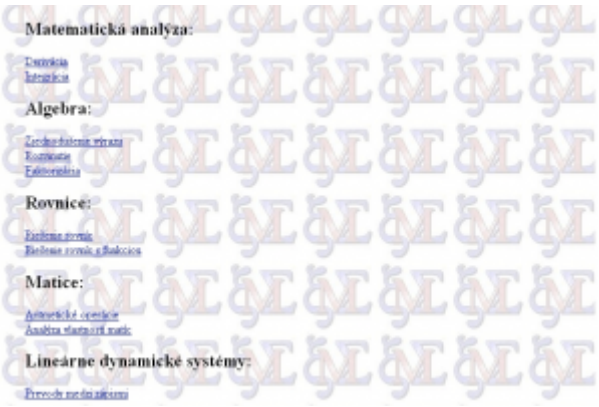
Obr. 10 – OpenModelica – Cievka a gulička – príklad simulácie udržania kovovej guličky v žiadanej pozícií nad povrchom za pomoci magnetického poľa

Obr. 11 – Scilab – graf priebehu výšky hladiny a výstupného toku pre dve spojené nádoby – príklad grafického priebehu regulácie žiadanej hodnoty výstupného toku spolu s grafickou interpretáciou výšky hladín oboch spojených nádob



Obr. 12 – Scilab – animácia simulácie prietoku kvapaliny cez sústavu dvoch spojených nádob – animácia príkladu simulácie dvoch spojených nádob z predchádzajúceho obrázka

Maxima má namiesto simulácie predpripravených modelov možnosť riešenia zložitejších typov matematických príkladov, pričom pri každom z voliteľných typov je možné si pozrieť aj vzor zápisu vybraného typu príkladu do okna.



Obr. 13 – Obrázok ponuky riešenia zložitých matematických príkladov pomocou Maximy

Možné voľby:

- Z oblasti matematickej analýzy môžeme riešiť derivácie a integrácie matematických výraz
- Z oblasti matematickej algebry je možné zjednodušovať výrazy, rozvinúť alebo faktorizovať matematické výrazy.
- Ďalej je možné riešiť rovnice, prípadne rovnice ktoré v sebe obsahujú ďalšiu funkciu.
- Z oblasti matíc je možné riešiť aritmetické operácie s maticami, prípadne zisťovať vlastnosti matíc. Ako poslednou možnosťou je realizovať prevody medzi jednotlivými zápismi lineárnych dynamických systémov.

2. Backend

Ako sme to už spomenuli, výpočtové schopností webovej aplikácie sú poskytnuté tromi aplikáciami (Maxima, OpenModelica a Scilab). Komunikácia medzi jednotlivými matematickými softvérmi a webovým rozhraním je odlišná.

Pri zahrnutí Maximy do nášho virtuálneho laboratória bolo potrebné vybudovať rozhranie, ktoré by umožnilo prístup k programovému jadru. Rozhodli sme sa implementovať grafické užívateľské rozhranie, ktoré ponúka užívateľovi využívanie skoro všetkých funkcií Maximy pomocou príkazového riadka na webovej stránke. Pre účely zobrazenia používateľského rozhrania v prehliadači sme využili technológie ako JQuery, JSMath a FLOT.

Užívateľ môže použiť aplikáciu Maxima nainštalovanú na serveri zadávaním príkazov Maximy do príkazového riadku zobrazeného na stránke. Po odoslaní formulára sa na serveri pre užívateľa vytvorí nová relácia, a prehliadač pošle príkazy do PHP skriptu na serveri. Tento skript overuje dáta a odovzdá ich druhému PHP skriptu, ktorý je zodpovedný za odosielanie dát do Maximy. Pred odoslaním používateľských dát do Maximy musí tento skript identifikovať, či sa jedná o príkaz na vykreslenie grafu, alebo

napríklad riešenie rovnice.

Ak je príkaz vykreslenie grafu, vygeneruje sa názov súboru pre súbor, ktorý bude obsahovať súradnice grafu. Po tomto úkone skript vytvorí reťazec príkazov pre shell, obsahujúci aj reťazec Maxima príkazov. Vytvorený reťazec dostane ako argument Perl skript, ktorý sa spustí cez PHP funkciu shell_exec(). Perl-skript spustí Maximu ako detský proces a predá mu príkazy na vykonanie.

Ak reťazec obsahuje príkaz na vykreslenie grafu, potrebné výstupy sú Maximou vygenerované a zavolá sa program gnuplot pre 2D vykreslenie. Gnuplot vygeneruje DAT súbor z výstupu Maximy s názvom súboru, ktorý PHP skript dopredu vytvoril. Potom Maxima pošle svoj výstup vo forme textového reťazca späť do Perl skriptu, ktorý ho prepošle do PHP skriptu, ktorý ho volal. PHP skript dostane dáta ako návratovú hodnotu shell_exec() funkcie. V tomto kroku sa odstránia zbytočné znaky z reťazca a tento reťazec sa pošle do PHP skriptu, ktorý generuje výstup pre užívateľa. Po týchto krokoch má používateľ výsledok v prehliadači.

Obr. 14 – Bloková schéma princípu komunikácie programu Maxima s webovou stránkou

V prípade OpenModelicy realizácia komunikácie je vlastne rozšírením komunikácie použitej v bakalárskej práci Internetom podporované riešenie úloh v programovom balíku OpenModelica[10]. Z tejto práce bol na našom serveri využívaný aj jeden model, ktorý znázorňuje simuláciu magnetickej levitácie. Vytvorenú komunikáciu sme spracovali do jednej triedy, ktorá má predefinované metódy pre jednoduchšiu komunikáciu.

V tejto časti webovej aplikácie na strane servera požiadavky koncového užívateľa sú spracované Pythonom. Python pretransformuje údaje tak, aby boli zrozumiteľné pre OpenModelicu a odošle jej ich v podobe reťazca. OpenModelica beží v móde Interactiv CORBA. Pri komunikácii sa používa ID programu, ktoré je vytvorené po jeho spustení.

Pythonom generované reťazce odpovedajú takým príkazom alebo modelom, ktoré sa bežne zadávajú pomocou klávesnice do konzoly lokálne spustenej OpenModelicy. OpenModelica tieto príkazy vykoná, v prípade zadaného modelu tento model vytvorí a výsledok prepošle do Pythonu. V prípade simulácie posiela späť len správu o úspešnom vykonaní. Údaje sa zapíšu do externého súboru. V tomto kroku nie je rozdiel medzi lokálne a interaktívne bežiacim módom, pretože aj v prípade lokálneho módu sa dáta tiež ukladajú do externého súboru. Ak sa spracováva len jednoduchý výpočet, tak sa vráti očakávaný výsledok.

V prípade simulácie Python musí vykonať aj nasledovné úkony:

1. Získať výsledky simulácie.
2. Vykresliť grafické objekty. V tejto časti sa uskutoční vyhodnotenie výsledku, ktorý sa vykresľuje ako graf alebo animácia.

Vykreslením výsledku sa ukončí komunikácia. Užívateľ má možnosť odoslať nové

údaje, ktoré server môže znova prijať a spracovať.

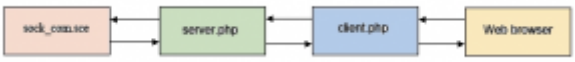
Vytvorená aplikácia vykoná aj kontrolu nesprávne zadaných údajov, ktoré nie je možné nasimulovať. Zlyhaniu systému sa snažíme zabrániť tak, že v Pythone neustále kontrolujeme vracajúce sa výsledky, ktoré ešte pred samotným vykreslením najprv vyhodnotíme. Týmto riešením môže server bežať ďalej, a je pripravený na ďalšie výpočty. Komunikačná bloková schéma sa nachádza na obrázku 15.

Obr. 15 – Bloková schéma komunikácie OpenModelica s prehliadačom užívateľa

V časti našej webovej aplikácie, ktorá využíva SciLab pre výpočty a simulácie sa využíva rozšírenie SciLabu s názvom TCP Socket Toolbox. Tento toolbox bol pôvodne vytvorený na riadenie reálnych zariadení, ktoré sa dajú pripojiť k počítaču pomocou klasickej sieťovej karty. Síce nie je to štandardnou súčasťou balíku, ale je to voľne dostupná na stránke autora [13]. Tento nástroj nám umožňuje pripojiť sa na počúvajúci TCP soket a vymieňať informácie cez tento soket. Jediná nevýhoda tohto nástroja je, že nám neumožňuje vytvárať sokety, len pripájať sa na ne. V našom prípade to ale neznamená problém, pretože väčšina webových programovacích resp. skriptovacích jazykov prácu so soketmi (vrátane ich vytvárania a zrušenia) umožňuje.

Po odoslaní požiadavky koncového užívateľa (kliknutím na tlačidlo Pošli v prípade webového rozhrania príkazového riadku SciLabu resp. Simuluj v prípade modelov) sa na serveri spustí PHP skript client.php, upraví vstupné parametre užívateľa do formy SciLab-ovských príkazov a prostredníctvom skriptu server.php pošle tieto príkazy vo forme textového reťazca SciLabu. Hlavná úloha skriptu server.php je vytvorenie TCP spojenia, posielanie inštrukcií, prijímanie výsledkov a zrušenie spojenia.

V prostredí SciLabu je spustený skript sock_com.sce, ktorý po prijatí údajov vykoná požadované inštrukcie a výsledok vráti tiež vo forme textového reťazca. Tento reťazec je potom spracovaný skriptom client.php, ktorý vizualizuje výsledky napr. vo forme grafu, alebo animácie vo webovom prehliadači koncového užívateľa. Ilustrácia tohto procesu je vidno na nasledovnom obrázku.



Obr. 16 – Bloková schéma komunikácie Scilab-u s prehliadačom koncového užívateľa

Podrobnejšie informácie o tohto princípu komunikácie si môžete dočítať v diele Internetom podporované riešenie úloh v programovom balíku SciLab [15].

3. Použitá Literatúra

1. Stránka JavaScriptovskej knižnice JQuery
   http://jquery.com/
2. Stránka JavaScriptovskej knižnice FLOT
   http://code.google.com/p/flot/
3. Stránka JavaScriptovskej knižnice JSMath

http://www.math.union.edu/~dpvc/jsMath/
4. Stránka Maximy naprogramovanej v CLisp
   https://launchpad.net/~blahota/+archive/wxmaxima
5. PHP knižnica vytvorená na komunikáciu s aplikáciou MAXIMA
   http://maximaphp.sourceforge.net/
6. Oficiálna stránka matematickej aplikácie MAXIMA obsahujúca všetky dôležité informácie o softvéri
   http://maxima.sourceforge.net/
7. OpenModelica home page
   http://www.openmodelica.org/
8. Modelica and the Modelica Association
   http://www.modelica.org/
9. OpenModelica releases
   http://www.ida.liu.se/labs/pelab/modelica/OpenModelica/releases/
10. Ladislav Szolik. Internetom podporované riešenie úloh v programovom balíku OpenModelica
11. http://www.openmodelica.org/index.php/home/userdocumentation
12. Scilab consortium. Scilab
    http://www.scilab.org/
13. Reveyrand, Tibault. The SOCKET Toolbox for Scilab
    http://www.reveyrand.fr/
14. Nikoukhah, Ramine. Scicos presentation
    http://www.scicos.org
15. Magyar, Zoltán. Internetom podporované riešenie úloh v programovom balíku Scilab

---

Spoluautormi projektu sú Bc. Ladislav Szolik, Bc. Tomáš Starý, Bc. Ľudovít Vörös. Vedúca tímu je doc. Ing. Katarína Žáková, PhD.