SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA CHEMICKEJ A POTRAVINÁRSKEJ TECHNOLÓGIE

INTERNETOVÝ MODUL PRE MODELOVANIE A SIMULÁCIU PROCESOV

DIPLOMOVÁ PRÁCA

FCHPT-5414-40614

2013 **Bc. Rastislav Jenčík**

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA CHEMICKEJ A POTRAVINÁRSKEJ TECHNOLÓGIE

INTERNETOVÝ MODUL PRE MODELOVANIE A SIMULÁCIU PROCESOV

DIPLOMOVÁ PRÁCA

FCHPT-5414-40614

Študijný program: Automatizácia a informatizácia v chémii a potravinárstve
Číslo a názov študijného odboru: 5.2.14 Automatizácia
Školiace pracovisko: Oddelenie informatizácie a riadenia procesov
Vedúci záverečnej práce/školiteľ: Ing. Ľuboš Čirka, PhD.

2013 **Bc. Rastislav Jenčík**

Slovenská technická univerzita v Bratislave
Ústav informatizácie, automatizácie a matematiky

Fakulta chemickej a potravinárskej technológie
Akademický rok: 2012/2013
Evidenčné číslo: FCHPT-5414-40614

STU
FCHPT

ZADANIE DIPLOMOVEJ PRÁCE

Študent: **Bc. Rastislav Jenčík**
ID študenta: 40614
Študijný program: automatizácia a informatizácia v chémii a potravinárstve
Študijný odbor: 5.2.14 automatizácia
Vedúci práce: Ing. Ľuboš Čirka, PhD.

Názov práce: **Internetový modul pre modelovanie a simuláciu procesov**

Špecifikácia zadania:

Cieľom práce je upraviť a doplniť knižnicu modelov technologických procesov.

Úlohy:
1. Rešerš dostupných riešení virtuálnych laboratórií.
2. Naštudovať jazyky PHP a JavaScript (AJAX, jQuery), zoznámiť sa s databázou MySQL.
3. Naštudovať architektúru systému MODELTOOL.
4. Vykonať analýzu, špecifikovať požiadavky a navrhnúť štruktúru nových modulov.
5. Programovo realizovať moduly v PHP a MySQL.
6. Implementovať vytvorené moduly do MODELTOOL.
7. Overiť funkčnosť a vypracovať postup inštalácie.
8. Vypracovať dokumentáciu.

Rozsah práce: 50

Riešenie zadania práce od: 18. 02. 2013
Dátum odovzdania práce: 25. 05. 2013

L. S.

Slovenská technická univerzita v Bratislave Fakulta chemickej a potravinárskej technológie dekanát - 5 -

Bc. Rastislav Jenčík
študent

prof. Ing. Miroslav Fikar, DrSc.
vedúci pracoviska

prof. Ing. Miroslav Fikar, DrSc.
garant študijného programu

Poďakovanie

Touto cestou by som chcel poďakovať svojmu školiteľovi Ing. Ľubošovi Čirkovi, PhD. za cenné rady a odborné vedenie pri vypracovávaní tejto práce.

Abstrakt

Diplomová práca sa zaoberá rozšírením a úpravou simulačného nástroja realizovaného prostredníctvom internetovej stránky Modeltool. Teoretická časť sa venuje trendom súvisiacim s virtuálnymi laboratóriami, dostupným technológiám, metódam pri tvorbe webových stránok ako PHP, JavaScript, jQuery, Flot, SVG ako aj samotnej problematike simulácie modelov, riešením obyčajných diferenciálnych rovníc prvého rádu pomocou metódy Runge-Kutta štvrtého rádu. Práca obsahuje krátku analýzu jednotlivých modelov procesov, ktoré boli v rámci práce do Modeltool pridané. Hlavne z pohľadu vstupných formulárov pre simuláciu ako aj z pohľadu simulácie. Praktická časť opisuje vybraný príklad, na ktorom sa popisuje tvorba kompletného modulu. Zameriava sa najmä na problematiku súvisiacu s tvorbou zložitejších formulárov pre sériovo zapojené zariadenia. S využitím technológií HTML, SVG, PHP, JavaScript, knižnice jQuery a metódy AJAX. Opisuje spracovanie údajov z formulárov prostredníctvom PHP skriptu vytvoreného pre simulácie a následné uloženie získaných údajov do databázy. V práci sú opísané aj vytvorené PHP funkcie na export dát získaných zo simulácie do XML a txt súboru.

Kľúčové slová: virtuálne laboratórium, internetový modul, simulácia, PHP

Abstract

The aim of the diploma thesis is to modify and extend the web based simulation tool Modeltool of some new processes. The theoretical part is dedicated to present trends of virtual laboratories and technologies. It describes methods commonly used to create webpages such as PHP, JavaScript, jQuery, Flot or SVG. This part contents also the theory of the first order ordinary differential equations solution by using the method Runge Kutta fourth order. The thesis contents a brief analyse of input simulation forms and process model simulations at all, which were added to Modeltool. Practical part deals with all the specifications of the created simulation module using a sample. This part focuses on the issues related to the design of complex forms used to simulate serial connected devices. There are explained solutions by technologies such as HTML, SVG, PHP, JavaScript , jQuery and AJAX, . This part describes how the simulation data are processed by the form created by PHP script and after that saved by the same script. There are described also PHP functions invented to export data obtained from simulation to XML or txt format.

Keywords: virtual laboratory, Internet module, simulation, PHP

Obsah

Zoznam obrázkov

Zoznam tabuliek

Zoznam príloh

Príloha1 CD obsahujúce zdrojové súbory stránky a elektronickú verziu práce

Zoznam skratiek

DR	diferenciálna rovnica
VL	virtuálne laboratórium
US	ustálený stav

1 Úvod

Webové stránky už dávno nepredstavujú jednoduché HTML dokumenty ako to bolo v začiatkoch fungovania siete internet. Dnešné prostriedky a technológie, ktoré slúžia k tvorbe webových stránok ponúkajú široké možnosti a tieto možnosti sa stále rozširujú čo súvisí s veľkou popularitou internetu. Široké možnosti ktoré nové technológie a prostriedky ponúkajú, otvárajú cestu pre rôzne použitie webových stránok. Zároveň webové stránky nesú mnohé pozitíva z pohľadu užívateľa ako aj tvorcu. Z pohľadu užívateľa sa stále jedná len o výstupný dokument či internetovú aplikáciu, ktoré mu prostredníctvom webového prehliadača stránka poskytuje. Bez nutnosti inštalovať do počítača dodatočný softvér či priamo vlastniť stránku. Týmto je úžitok webovej stránky takpovediac veľmi dostupný - nezávislý od počítača. Pre tvorcu webových stránok je hlavný prínos v širokom výbere technológií a v prevažnej miere aj v ich bezplatnosti. To sú dobré vlastnosti ktoré viedli k použitiu webových stránok na najrôznejšie účely.

Jedným z možných účelov je použitie webovej stránky ako simulačného nástroja respektíve ako virtuálne laboratórium vybraných procesov. Čo predstavuje pojem virtuálne laboratórium a aké sú dôvody pre použitie ? Virtuálne laboratórium predstavuje nehmotné zastúpenie procesu alebo javu, ktorý chceme pozorovať či skúmať. Dôvody pre použitie sú rôzne, častým je že daný proces či jav je nebezpečný alebo finančne nákladný. Avšak dobré virtuálne laboratórium môže poskytnúť užitočné údaje a tak do istej miery nahrádzať reálny proces či jav pri skúmaní. Pre základné štúdium a pochopenie je virtuálne laboratórium dobrá voľba najmä pre študentov. Tí často s reálnym procesom nemajú možnosť prísť do styku alebo len v obmedzenej miere. Takto prostredníctvom dobre navrhnutého virtuálneho

laboratória s ohľadom na vzdelávací aspekt môžu dospieť k novým poznatkom či podnetom.

Virtuálne laboratória je možné realizovať prostredníctvom viacerých prostriedkov. A to buď pomocou softvéru, ktorý je na tieto a podobné účely určený ako MATLAB Simulink či GNU Octave, alebo tvorbou kompletného riešenia v niektorých z dostupných jazykov ako sú Java, C++, PHP, JavaScript, Flash a iné.

Cieľ prace je rozšírenie webovej stránky ako simulačného nástroja o ďalšie procesy a upraviť stránku s ohľadom na súčasné možnosti v oblasti tvorby webových stránok.

2 Teoretická časť

V teoretickej časti sa postupne budeme venovať každej časti problematiky ktorá súvisí s prácou. Prvá podkapitola teoretickej časti pojednáva o VL ktoré už boli realizované a snaží sa zanalyzovať aký je trend v oblasti VL. Vo všeobecnosti však netreba opomenúť čo laboratórium predstavuje. Laboratórium je pracovisko špecializované na výskum a experimenty s cieľom bádania a získavania údajov. Z pohľadu reality a dostupnosti ich môžeme rozdeliť na:

- reálne laboratória – sú hmotné a vyžadujú prítomnosť užívateľa
- reálne vzdialené laboratória – sú síce hmotné, ale užívateľ k ním môže pristupovať a manipulovať s nimi v istom rozsahu cez iný prostriedok napríklad webovú stránku
- virtuálne laboratória – simulované javy, procesy snažiac sa opísať reálny proces, jav

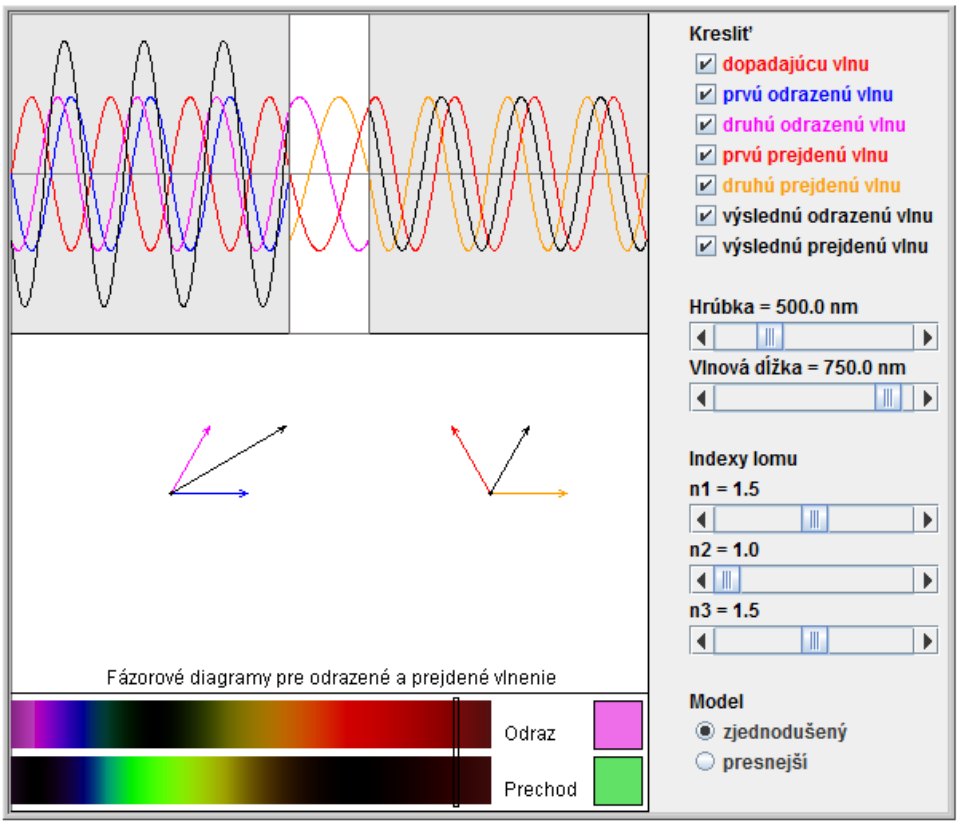
Práca sa venuje práve poslednému zo spomenutých. Druhá podkapitola opisuje použité modely slúžiace na opis procesu v rámci simulácie. Tretia podkapitola opisuje matematický postup pri riešení simulačného problému. Piata podkapitola sa venuje kompletnému súhrnu použitých web technológií a metód, ktoré sa použili v rámci práce.

2.1 Virtuálne laboratória, u nás i vo svete

Význam virtuálnych laboratórií zdôrazňuje ich veľké množstvo a široká škála obsahu a použitých riešení. Najčastejšie je možné prísť do kontaktu s virtuálnymi laboratóriami slúžiacimi na vzdelávacie účely. Krátky výber domácich ako aj

zahraničných virtuálnych laboratórií:

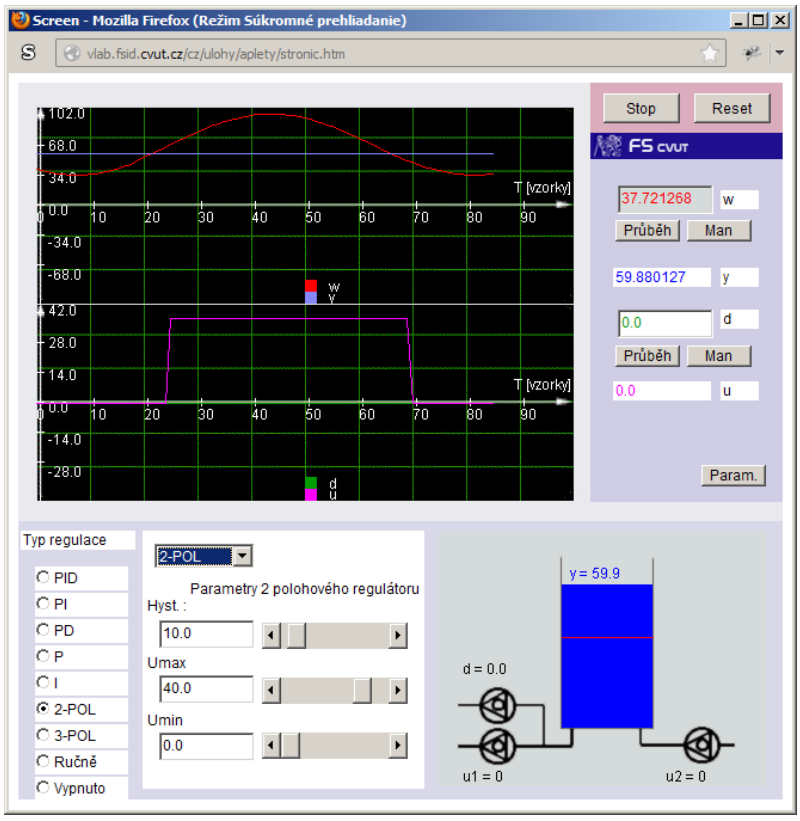
Virtuálne laboratórium fyziky a matematiky Slavomíra Tuleju – simulácie fyzikálnych a matematických javov ako napríklad interferencia svetla na tenkej vrstve (obr. 1) či pohyb družíc, realizované prostredníctvom jazyka Java. Projekt realizoval stredoškolský učiteľ fyziky RNDr. Slavomír Tuleja PhD. Jednotlivé javy a procesy sú možné k odskúšaniu na stránke http://www.stuleja.org/ v sekcii interaktívne učebné materiály [1].



Obr. 1 Aplikácia ilustrujúca interferenciu svetla z VL Slavomíra Suleju.

Virtuálne laboratórium automatického riadenia – laboratórium realizované na univerzite ČVUT v Prahe. Virtuálne laboratórium obsahuje šestnásť procesov ako napríklad vodnú a vzduchovú levitáciu, guličku na elipse, guličku na tyči a iné. Procesy sa zároveň nachádzajú v reálnom laboratóriu automatického riadenia na strojníckej fakulte ČVUT. Virtuálne laboratórium je vytvorené pomocou technológie

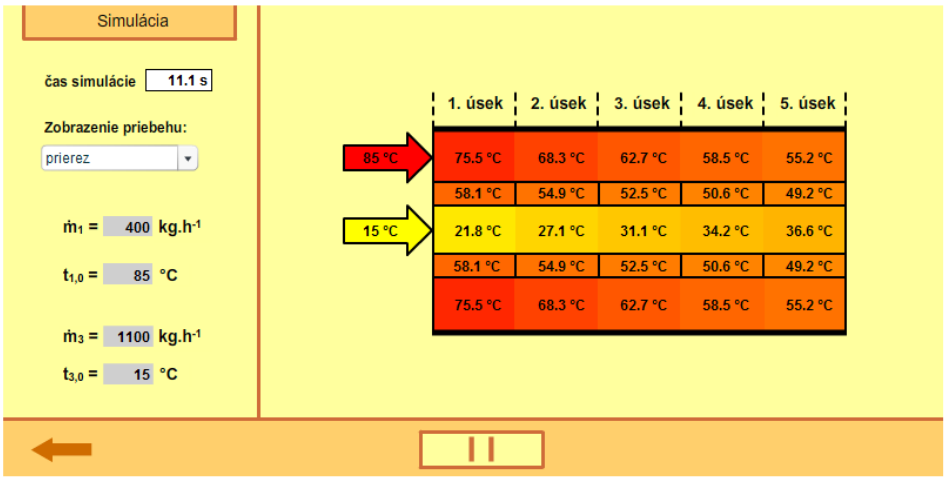
Java a zahŕňa aj riadenie simulovaných procesov pomocou rôznych typov regulátora a riadenia. Na obr. 2 možno vidieť aplikáciu simulujúcu riadenie zásobníka pomocou dvojpolohového regulátora. V dolnej časti, vľavo možno vidieť rovnako ďalšie možné typy regulátorov respektíve typov riadenia. Vpravo animáciu zásobníka. Stránka virtuálneho laboratória http://vlab.fsid.cvut.cz/cz/index.php [2].



Obr. 2 Ukážka z priebehu simulácie riadenia zásobníka

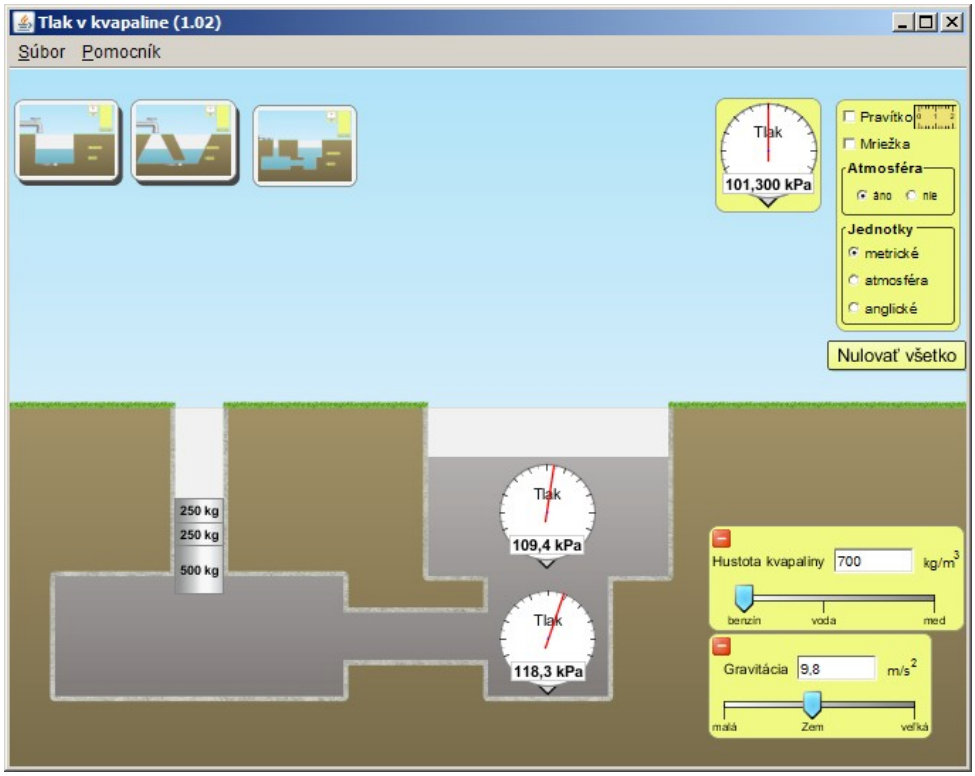
Virtuálne laboratórium, Ing. Martin Kalúz – diplomová práca v rámci STU FCHPT, realizovaná na Ústave informatizácie, automatizácie a matematiky. Virtuálne laboratórium realizované technológiou Adobe Flash zahŕňa chemicko-technologické procesy sú-prúdového rúrkového výmenníka tepla, sériovo zapojených zásobníkov kvapaliny s interakciou a prietokového chemického reaktora. Aplikácia ponúka k daným procesom zároveň animácie. Pre zásobníky sa v animácií mení výška hladiny na základe údajov zo simulácie. V prípade výmenníka tepla sa sleduje

teplota v jednotlivých úsekoch čo ilustruje meniaca sa farba počas animácie. Rozdielnu farbu úsekov možno vidieť na obr. 3. Celá aplikácia k odskúšaniu na e-learningu FCHPT http://www.kirp.chtf.stuba.sk/moodle/course/view.php?id=312 [3].



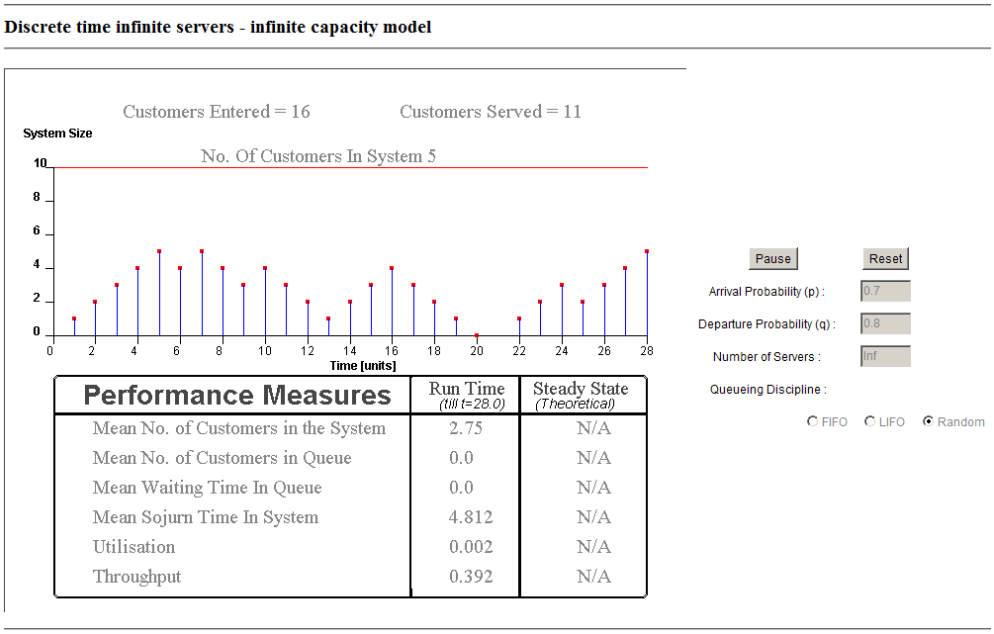
Obr. 3 Ukážka z priebehu simulácie trojkapacitného rúrkového výmenníka tepla, VL, Ing. Martin Kalúz

PhET - Physics Education Technology projekt – edukačný projekt realizovaný na štátnej univerzite Colorado v meste Boulder (CU Boulder). Zahrňuje simulácie z oblasti chémie, biológie, fyziky, matematiky a ďalších oblastí. Realizovaný je pomocou jazyka Java a Flash. Projekt má výraznú vizuálnu stránku ako dokazuje aj obr. 4. Pôsobí interaktívne, je vhodný skôr na ilustrácie ako na prácu so získanými údajmi. Celý projekt je realizovaný tak, aby vzniknuté aplikácie bolo možné jednoducho preložiť. Vďaka tomu sú mnohé simulácie preložené do viacerých jazykov medzi inými aj do slovenčiny. Stránka projektu http://phet.colorado.edu/ [4].



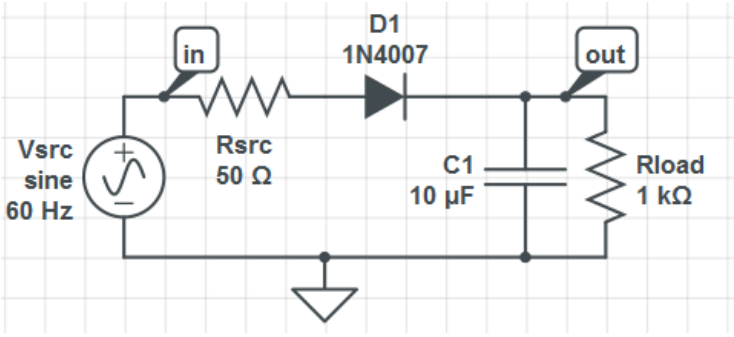
Obr. 4 Ukážka z aplikácie PhET pre sledovania tlaku v kvapaline

Virtual Labs, India MHRD – iniciatíva Ministerstva rozvoja ľudských zdrojov. Projekt realizovaný v Indii s cieľom zahrnúť virtuálne laboratória (obr. 5) do systému výučby. Jeden z cieľov je nadchnúť študentov k experimentom a vzbudiť v nich zvedavosť. Participuje na ňom dvanásť indických vzdelávacích inštitúcií z rôznych odborných oblastí. Jedná sa o kompletný edukačný nástroj, okrem samotných virtuálnych laboratórií – aplikácií, stránka obsahuje aj teoretické podklady k problematike, otázky na danú tému, videá k danému javu či procesu. Spracovanie, kvalita a použité riešenia závisia od inštitúcie ktorá danú problematiku riešila. Stránka projektu http://www.vlab.co.in/ [5].



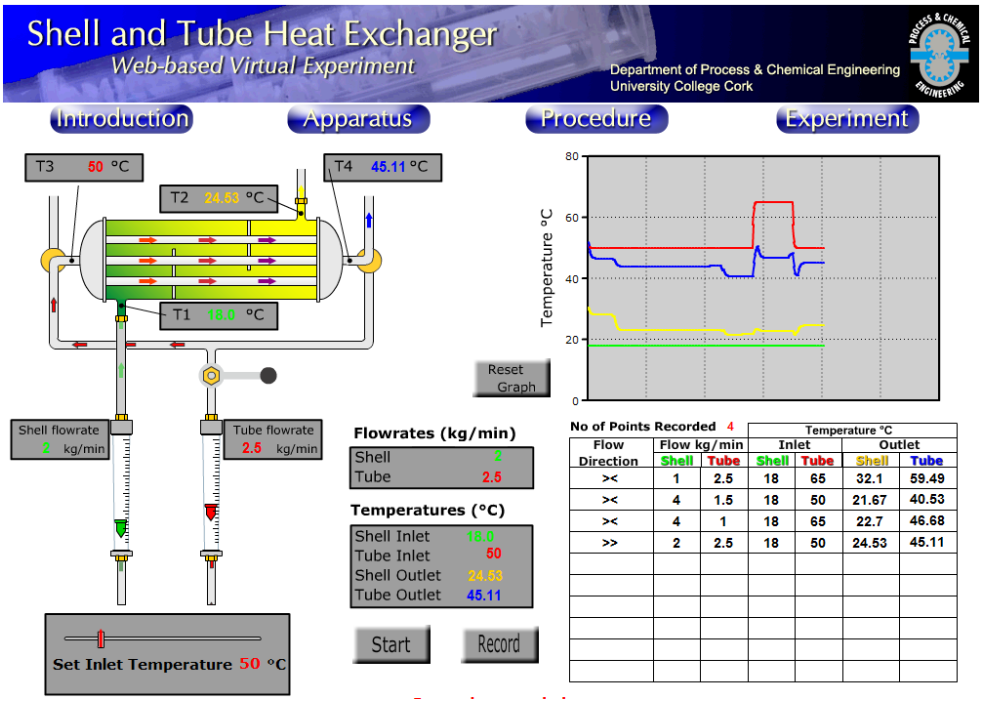
Obr. 5 Ukážka z aplikácie pre sledovanie diskrétnych modelov

CircuitLab – predstavuje aplikáciu ktorá funguje priamo na webovom prehliadači. Nebola vytvorená vzdelávacou inštitúciou a jej použitie je bezplatné. Umožňuje v prostredí webového prehliadača pomocou zadefinovaných blokov z oblasti elektrotechniky ako prepínače, relé, odpory, logické brány a iné poskladať kompletný elektronický obvod ako je na obr. 6. Jednotlivým blokom je možné priradiť rôzne parametre a hodnoty. S blokmi možno samostatne narábať otáčať ich a manipulovať s nimi. Kompletne poskladanú schému so zadanými parametrami je potom možné odskúšať – simulovať jej správanie. Výstupom zo simulácie je graf. V grafe môže byť rôzna závislosť v závislosti od toho čo sme sledovali. Schému rovnako ako aj graf je možné exportovať. Stránka projektu https://www.circuitlab.com/ [6].



Obr. 6 Detail schémy z aplikácie pre tvorbu a simuláciu elektrických obvodov

Shell and Tube Heat Exchanger – web based virtual experiment, dobrý príklad toho ako môže virtuálne laboratórium vyzerať. Projekt bol realizovaný na oddelení procesov a chemického inžinierstva na Univerzite Cork (University College Cork) v Írsku a publikovaný na štvrtom medzinárodnom sympóziu vzdelávania inžinierov v júli 2012 ktoré sa konalo vo Spojenom kráľovstve na Univerzite Sheffield. Je realizovaný v prostredí webovej stránky (obr. 7). Venuje sa simulácii rúrkového výmenníka tepla a má ilustrovať základné princípy, ktoré pri danom type zariadenia platia. Stránka obsahuje štyri kategórie, kde v prvých troch je teoretický základ a krátky popis reálneho zariadenia, ktoré virtuálne laboratórium simuluje, a manuál k ovládaniu simulačného nástroja. Simulačný nástroj umožňuje meniť vstupné prietoky kvapalín, vstupnú teplotu ohrevnej kvapaliny ako aj smer prúdenia tejto kvapaliny voči ohrievanej kvapaline. Výstupom je graf závislosti teplôt od času. Celý projekt je k dispozícii na stránke http://cs1.ucc.ie/~jb7/exch/index.html [7].



Obr. 7 Simulačná časť projektu UCC

To bolo niekoľko príkladov na realizované virtuálne laboratória. Realizácia je rôzna, závisí od rozsahu problematiky ako aj účelu. Ak sa jedná len o ukážku v malom rozsahu, ako to je v prípade PhET, tak sa VL zameriava najmä na názornosť, ktorá by sa optimálne mala užívateľovi premietnuť do reálnej predstavy o danom probléme. Aby užívateľ dokázal hlbšie pochopiť problém, jednoduchá simulácia nestačí. Vtedy je vhodné aby VL zároveň ponúklo grafy, prípadne iné širšie údaje o danom probléme. Dobré riešenie predstavuje názornosť spojenú s dostatkom dodatočných údajov, závislostí a možnosťou získania údajov, ktoré simulácia poskytla, teda ich exportom. Získané údaje môžu poslúžiť k ďalšiemu štúdiu problému. Tieto všetky znaky by VL malo mať, no najdôležitejším ostáva dostatočne presný opis objektu, (javu), ktorý VL opisuje (simuluje).

2.2 Matematické modely vybraných procesov

Matematické modely sú základom pre simuláciu. Úlohou modelu je vhodne

zachytiť, opísať vlastnosti, ktoré chceme sledovať. Pri modeloch sa uvažujú často zjednodušenia a predpoklady, ktoré je dobré poznať pri pozorovaní správania sa simulovaného procesu. Pre návrh simulačného nástroja je porozumenie modelov rovnako dôležité najmä ak chceme náš simulačný nástroj priblížiť skutočnosti alebo v určitom smere obmedziť.

[8, 9]

Diplomová práca sa nevenuje tvorbe modelov, či ich odvádzaniu preto sú v práci uvedené len vzťahy a niektoré zjednodušenia použité pri tvorbe simulačného nástroja. Použité modely možno nájsť v prednáškach k predmetu Modelovanie a k predmetu Riadenie procesov na portáli e-vzdelávania Moodle FCHPT. Nakoľko v procesoch sa opakujú veličiny, tak je dobré si ich vopred zadefinovať (tabuľka 1).

Tabuľka 1 Veličiny použité v modeloch

veličina	**rozmer**	**symbol**
aktivačná energia	J mol^{-1}	Ea
hustota	kg m^{-3}	ρ
koncentrácia	mol m^{-3}	c
konštanta ventilu	$m^{2.5}s^{-1}$	k
objem	m^3	V
objemový prietok	m^3s^{-1}	q
predexponenciálny faktor reakcie	1	k_∞
prierez	m^2	F
reakčná entalpia	J mol^{-1}	$\Delta_r H$
stechiometrický koeficient	1	ν
špecifická tepelná kapacita	J kg^{-1} K^{-1}	Cp
teplota	K	ϑ
teplo dodané elektrickým ohrevom	J s^{-1}	ω
teplo-výmenná plocha	m^2	A
univerzálna plynová konštanta	J K^{-1} mol^{-1}	g

úhrny koeficient prestupu tepla	W m^{-2} K^{-1}	α
výška hladiny v zásobníku	m	h

2.2.1 Zmiešavací zásobník

Pod týmto názvom sa na stránke nachádza proces miešania dvoch prúdov ako nelineárny systém. A to buď miešanie prúdov s rôznou teplotou alebo miešanie prúdov s rôznou koncentráciou. V prvom prípade sa jedná o entalpickú bilanciu, v druhom o materiálovú. Pre materiálovú bilanciu sme uvažovali dynamický matematický model

$$q_{01}(t)\cdot c_{01}(t)+q_{02}(t)\cdot c_{02}(t) \;=\; q_1(t)\cdot c_1(t)+V\cdot\frac{d\,c_1(t)}{dt} \tag{1}$$

so začiatočnou podmienkou $c_1(0)$. Koncentrácie v zmiešavacom zásobníku sú ustálenom stave v tvare

$$\frac{q^s_{01}}{q^s_{01}+q^s_{02}}\cdot c^s_{01}+\frac{q^s_{02}}{q^s_{01}+q^s_{02}}\cdot c^s_{02} \;=\; c^s_1 \tag{2}$$

kde $c_{01}(t)$, $c_{02}(t)$, $q_{01}(t)$, $q_{02}(t)$ sú vstupné veličiny a $c_1(t)$ je výstupná a stavová veličina. Podobne pre entalpickú bilanciu

$$q_{01}(t)\cdot\rho\cdot Cp\,\vartheta_{01}(t)+q_{02}(t)\cdot\rho\cdot Cp\,\vartheta_{02}(t) \;=\; q_1(t)\cdot\rho\cdot Cp\cdot\vartheta_1(t)+V\cdot\rho\cdot Cp\cdot\frac{d\,\vartheta_1(t)}{dt} \tag{3}$$

je začiatočná podmienka $\vartheta_1(0)$ a $\vartheta_{ref} = 0$

Teplota v zmiešavacom zásobníku má v ustálenom stave tvar

$$\frac{q^s_{01}}{q^s_{01}+q^s_{02}}\cdot\vartheta^s_{01}+\frac{q^s_{02}}{q^s_{01}+q^s_{02}}\cdot\vartheta^s_{02} \;=\; \vartheta^s_1 \tag{4}$$

kde $\vartheta_{01}(t)$, $\vartheta_{02}(t)$, $q_{01}(t)$, $q_{02}(t)$ sú vstupné veličiny a $\vartheta_1(t)$ je výstupná a stavová veličina.

Zjednodušenia:

hustota ρ, ako aj tepelná kapacita Cp sú rovnaké v celom zariadení, to platí aj pre vstupné a výstupné prúdy. Objem V v zásobníku je konštantný a zmes sa dokonale premiešava. Nakoľko uvažujeme, že hustota a tepelná kapacita sa v zariadení nemení, tak po úprave rovnica (3) prejde na tvar

$$\frac{1}{V}\cdot\frac{q_{01}(t)}{q_1(t)}\cdot\vartheta_{01}(t)+\frac{1}{V}\cdot\frac{q_{02}(t)}{q_1(t)}\cdot\vartheta_{02}(t)-\frac{1}{V}\cdot\vartheta_1(t)=\frac{d\,\vartheta_1(t)}{dt} \tag{5}$$

Do podobného tvaru upravíme aj rovnicu (3)

$$\frac{1}{V}\cdot\frac{q_{01}(t)}{q_1(t)}\cdot c_{01}(t)+\frac{1}{V}\cdot\frac{q_{02}(t)}{q_1(t)}\cdot c_{02}(t)-\frac{1}{V}\cdot c_1(t)=\frac{d\,c_1(t)}{dt} \tag{6}$$

Z toho vyplýva, že pri daných zjednodušeniach sú obe bilančné rovnice zhodné. Čo sa využilo pri návrhu modulu pre proces miešania.

2.2.2 Rekuperačný výmenník tepla

Pre rekuperačný výmenník tepla sme uvažovali konštantné objemové prietoky horúcej a studenej kvapaliny, teda materiálovú bilanciu sme nesledovali. Potom entalpická bilancia pre horúcu kvapalinu, stenu a studenú kvapalinu vyzerá nasledovne

$$q_1\cdot\rho_1\cdot Cp_1\cdot\vartheta_{01}(t)=q_1\cdot\rho_1\cdot Cp_1\cdot\vartheta_1(t)+A_{12}\cdot\alpha_{12}\cdot\left(\vartheta_1(t)-\vartheta_2(t)\right)+V_1\cdot\rho_1\cdot Cp_1\cdot\frac{\vartheta_1(t)}{dt} \tag{7}$$

$$A_{12}\cdot\alpha_{12}\cdot\left(\vartheta_1(t)-\vartheta_2(t)\right)=A_{23}\cdot\alpha_{23}\cdot\left(\vartheta_2(t)-\vartheta_3(t)\right)+V_2\cdot\rho_2\cdot Cp_2\cdot\frac{\vartheta_2(t)}{dt} \tag{8}$$

$$q_3\cdot\rho_3\cdot Cp_3\cdot\vartheta_{03}(t)+A_{23}\cdot\alpha_{23}\cdot\left(\vartheta_2(t)-\vartheta_3(t)\right)=q_3\cdot\rho_3\cdot Cp_3\cdot\vartheta_3(t)+V_3\cdot\rho_3\cdot Cp_3\cdot\frac{\vartheta_3(t)}{dt} \tag{9}$$

so začiatočnou podmienkou: $\vartheta_i(0)\quad i=1,2,3$ a ϑ_{ref} = 0,

kde $\vartheta_{01}(t)$, $\vartheta_{03}(t)$ sú vstupné veličiny, $\vartheta_1(t)$, $\vartheta_2(t)$, $\vartheta_3(t)$ sú stavové veličiny a $\vartheta_1(t)$, $\vartheta_3(t)$ predstavujú výstupné veličiny. Po úpravách z rovníc (7), (8) a (9) získame časové konštanty a zosilnenia

$$T_1 = \frac{V_1 \cdot \rho_1 \cdot Cp_1}{q_1 \cdot \rho_1 \cdot Cp_1 + A_{12} \cdot \alpha_{12}} \tag{10}$$

$$T_2 = \frac{V_2 \cdot \rho_2 \cdot Cp_2}{A_{12} \cdot \alpha_{12} + A_{23} \cdot \alpha_{23}} \tag{11}$$

$$T_3 = \frac{V_3 \cdot \rho_3 \cdot Cp_3}{q_3 \cdot \rho_3 \cdot Cp_3 + A_{23} \cdot \alpha_{23}} \tag{12}$$

$$Z_{101} = \frac{q_1 \cdot \rho_1 \cdot Cp_1}{q_1 \cdot \rho_1 \cdot Cp_1 + A_{12} \cdot \alpha_{12}} \tag{13}$$

$$Z_{12} = \frac{A_{12} \cdot \alpha_{12}}{q_1 \cdot \rho_1 \cdot Cp_1 + A_{12} \cdot \alpha_{12}} \tag{14}$$

$$Z_{21} = \frac{A_{12} \cdot \alpha_{12}}{A_{12} \cdot \alpha_{12} + A_{23} \cdot \alpha_{23}} \tag{15}$$

$$Z_{23} = \frac{A_{23} \cdot \alpha_{23}}{A_{12} \cdot \alpha_{12} + A_{23} \cdot \alpha_{23}} \tag{16}$$

$$Z_{303} = \frac{q_3 \cdot \rho_3 \cdot Cp_3}{q_3 \cdot \rho_3 \cdot Cp_3 + A_{23} \cdot \alpha_{23}} \tag{17}$$

$$Z_{32} = \frac{A_{23} \cdot \alpha_{23}}{q_3 \cdot \rho_3 \cdot Cp_3 + A_{23} \cdot \alpha_{23}} \tag{18}$$

Po zavedení časových konštánt a zosilnení platí pre ustálený stav

$$Z_{101} \cdot \vartheta^s_{01} + Z_{12} \cdot \vartheta^s_2 = \vartheta^s_1 \tag{19}$$

$$Z_{21} \cdot \vartheta^s_1 + Z_{23} \cdot \vartheta^s_3 = \vartheta^s_2 \tag{20}$$

$$Z_{303} \cdot \vartheta^s_{03} + Z_{32} \cdot \vartheta^s_2 = \vartheta^s_3 \tag{21}$$

2.2.3 Sériovo zapojené výmenníky tepla s elektrickým ohrevom.

Tento proces predstavuje prvý proces v rámci diplomovej práce, v ktorom vystupujú sériovo zapojené zariadenia a poslúžil najmä na testovanie a prvé pokusy so simuláciou sériovo zapojených zariadení. Uvažujme *n* zariadení, potom pre *n* zariadení nadobudne dynamický matematický model nasledujúci tvar

$$q \cdot \rho \cdot \vartheta_0(t) + \omega_1(t) = q \cdot \rho \cdot \vartheta_1(t) + V_1 \cdot \rho \cdot Cp \cdot \frac{\vartheta_1(t)}{dt} \tag{22}$$

$$q \cdot \rho \cdot \vartheta_1(t) + \omega_2(t) = q \cdot \rho \cdot \vartheta_2(t) + V_2 \cdot \rho \cdot Cp \cdot \frac{\vartheta_2(t)}{dt} \tag{23}$$

...

$$q \cdot \rho \cdot \vartheta_{n-1}(t) + \omega_n(t) = q \cdot \rho \cdot \vartheta_n(t) + V_n \cdot \rho \cdot Cp \cdot \frac{\vartheta_n(t)}{dt} \tag{24}$$

pri začiatočných podmienkach $\vartheta_i(0) \quad i = 1,2,\ldots,n$,

kde $\vartheta_0(t)$, $\omega_1(t)$, $\omega_2(t)$, ..., $\omega_n(t)$ sú vstupné veličiny, $\vartheta_1(t)$, $\vartheta_2(t)$, …, $\vartheta_n(t)$ sú výstupné a stavové veličiny. Po úpravách rovníc (22), (23) a (24) získavame vzťahy pre zosilnenie a časové konštanty

$$Z = \frac{1}{q \cdot \rho \cdot Cp}, \quad T_1 = \frac{V_1}{q}, \quad T_2 = \frac{V_2}{q}, \quad \ldots, \quad T_n = \frac{V_n}{q} \tag{25}$$

Po zavedení časových konštánt a zosilnenia platí pre ustálený stav

$$\vartheta^s_1 = \vartheta^s_0 + Z \cdot \omega^s_1, \quad \vartheta^s_2 = \vartheta^s_1 + Z \cdot \omega^s_2, \quad \ldots, \quad \vartheta^s_n = \vartheta^s_{n-1} + Z \cdot \omega^s_n \tag{26}$$

2.2.4 Prietokový chemický reaktor

Model prietokového chemického reaktora predstavuje aj pri zjednodušeniach zložitý systém, kde je nutné riešiť viacero diferenciálnych rovníc. Entaplickú bilanciu

tvoria dve DR rovnice pre entalpickú bilanciu – teplota reakčnej zmesi a teplota chladiaceho média. Materiálovú bilanciu tvoria DR zložiek. Ich počet závisí od počtu reaktantov, resp. zložiek celkovo ak bilancujeme aj produkty. Na zložitosti pridáva aj výpočet ustálených stavov, keďže matematický model ustáleného stavu je opísaný nelineárnymi algebraickými rovnicami. Ich počet závisí od počtu zložiek uvažovaných v bilancii, teda pre *n* uvažovaných zložiek je matematický model ustáleného stavu tvorený *n*+2 algebraickými rovnicami pričom *n*+1 je nelineárnych. Chemický reaktor možno opísať pomocou nasledovných rovníc

$$q(t)\cdot c_{v,i}(t)+\sum_{m}^{j=1}\nu_{i,j}\cdot\dot{\xi}_{V,j}(t)\cdot V \;=\; q(t)\cdot c_i(t)+V\cdot\frac{d\,c_i(t)}{dt} \tag{27}$$

pri začiatočných podmienkach $c_i(0) \quad i=1,2,...,n$

$$\xi_{V,j} \;=\; k_j(t)\cdot f(c_i(t), i=1,...,n) \tag{28}$$

$$k_j(t) \;=\; k_{j\infty}\cdot e^{-\frac{Ea_j}{R\cdot\vartheta(t)}} \tag{29}$$

$$\begin{aligned} &q(t)\cdot\rho\cdot Cp\cdot\vartheta_v(t)+\sum_{j=1}^{m}\dot{\xi}_{V,j}(t)\cdot V\cdot(-\Delta_r H)_j \;= \\ &q(t)\cdot\rho\cdot Cp\cdot\vartheta(t)+\alpha\cdot A[\vartheta(t)-\vartheta_c(t)]+V\cdot\rho\cdot Cp\cdot\frac{\vartheta(t)}{dt} \end{aligned} \tag{30}$$

$$q_c(t)\cdot\rho_c\cdot Cp_c\cdot\vartheta_{c,v}(t)+\alpha\cdot A\cdot[\vartheta(t)-\vartheta_c(t)] \;=\; q_c(t)\cdot\rho_c\cdot Cp_c\cdot\vartheta_c(t)+V_c\cdot\rho_c\cdot Cp_c\cdot\frac{\vartheta_c(t)}{dt} \tag{31}$$

pri začiatočných podmienkach $\vartheta(0),\ \vartheta_c(0)$.

2.2.5 Sériovo zapojené plášťové výmenníky tepla

Druhý proces so sériovým zapojením zariadení. Bilancuje sa len teplota v danom výmenníku, teplota steny ako aj plášťa sa nebilancuje. Sériovo zapojené výmenníky možno opísať nasledovnými rovnicami, pre prvý zásobník platí

$$q_{1,v}(t)\cdot\rho\cdot Cp\cdot\vartheta_{1,v}(t)+\alpha_1\cdot A_1\cdot[\vartheta_{p,1}(t)-\vartheta_1(t)] \;=\; q_1(t)\cdot\rho\cdot Cp\cdot\vartheta_1(t)+V_1\cdot\rho\cdot Cp\cdot\frac{d\,\vartheta_1(t)}{dt} \qquad (32)$$

pre druhý až *n*-tý výmenník platí

$$q_{i,v}(t)\cdot\rho\cdot Cp\cdot\vartheta_{i,v}(t) \;+\; q_{i-1}(t)\cdot\rho\cdot Cp\cdot\vartheta_{i-1}(t) \;+\; \alpha_i\cdot A_i\cdot[\vartheta_{p,i}(t)-\vartheta_i(t)] = q_i(t)\cdot\rho\cdot Cp\cdot\vartheta_i(t)+V_i\cdot\rho\cdot Cp\cdot\frac{d\,\vartheta_i(t)}{dt} \qquad (33)$$

pri začiatočných podmienkach $\vartheta_i(0) \quad i=1,2,\ldots,n$,

kde $\vartheta_{1,v}(t)$, $\vartheta_{2,v}(t)$, ..., $\vartheta_{n,v}(t)$, $q_{1,v}(t)$, $q_{2,v}(t)$, …, $q_{n,v}(t)$, $\vartheta_{p,1}(t)$, $\vartheta_{p,2}(t)$, ..., $\vartheta_{p,n}(t)$, sú vstupné veličiny, $\vartheta_1(t)$, $\vartheta_2(t)$, ..., $\vartheta_n(t)$, sú stavy a $\vartheta_n(t)$ je výstupná veličina.

Po úpravách platí pre ustálený stav prvého zásobníka

$$\vartheta_1=\frac{q_{1,v}\cdot\rho\cdot Cp}{\alpha_1\cdot A_1+q_1\cdot\rho\cdot Cp}\cdot\vartheta_{1,v} \;+\; \frac{\alpha_1\cdot A_1}{\alpha_1\cdot A_1+q_1\cdot\rho\cdot Cp}\cdot\vartheta_{p,1} \qquad (34)$$

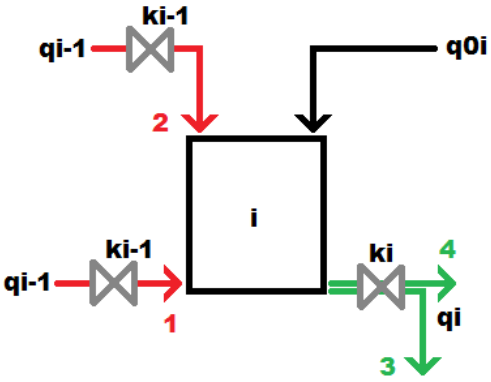
pre druhý až *n*-tý výmenník

$$\vartheta_i=\frac{q_{i,v}\cdot\rho\cdot Cp}{\alpha_i\cdot A_i+q_i\cdot\rho\cdot Cp}\cdot\vartheta_{i,v} \;+\; \frac{q_{i-1}\cdot\rho\cdot Cp}{\alpha_i\cdot A_i+q_i\cdot\rho\cdot Cp}\cdot\vartheta_{i-1} \;+\; \frac{\alpha_i\cdot A_i}{\alpha_i\cdot A_i+q_i\cdot\rho\cdot Cp}\cdot\vartheta_{p,i} \qquad (35)$$

2.2.6 Sériovo zapojené zásobníky

Sériovo zapojené zásobníky spolu s reaktorom, kde prebieha viacero reakcií, predstavujú najzložitejšie procesy pridané do Modeltool v rámci tejto diplomovej práce. K dispozícii je dva až päť zásobníkov. Medzi zásobníkmi môžu byť rôzne interakcie. Pre zvolenú konfiguráciu možno na základe obr. 8 pre *i*-tý zásobník odvodiť všeobecný model

$$\frac{d\,h_i(t)}{dt}=\frac{1}{F_i}\cdot q_{0i}+\frac{1}{F_i}\cdot\mathrm{IN}-\frac{1}{F_i}\cdot OUT \qquad (36)$$



Obr. 8 Možné prítoky (výtoky) do (zo) zásobníka

Do zásobníka *i* vchádzajú dva prúdy a jeden vychádza. Ak je zásobník *i* v interakcii so zásobníkom *i*-1, tak je aktívny prúd 1 symbolizujúci interakciu s predchádzajúcim zásobníkom a výraz IN nadobudne tvar $\mathrm{IN}=k_{i-1,i-1}\cdot\sqrt{h_{i-1}(t)-h_i(t)}$. Ak zásobník *i* v interakcii s predchádzajúcim zásobníkom nie je, tak je to prúd 2 a výraz IN nadobudne tvar $\mathrm{IN}=k_{i-1,i-1}\cdot\sqrt{h_{i-1}(t)}$. Výnimku pre vstupný prúd tvorí prvý zásobník, kedy na vstupe bude vždy iba objemový prietok q_0, ktorý nezávisí od interakcií a výraz IN nadobudne tvar $\mathrm{IN}=q_0(t)$. Druhý vstupný prúd q_{0i}, predstavuje len objemový prietok na vstupe a je nezávislý od interakcií. Podobne pre výstupný prúd zo zásobníka *i*. Ak je zásobník *i* v interakcií s nasledujúcim zásobníkom *i*+1, tak je aktívny prúd 4 symbolizujúci interakciu s nasledujúcim zásobníkom a výraz OUT nadobudne tvar $OUT=k_{ii}\cdot\sqrt{h_i(t)-h_{i+1}(t)}$, opačne to bude prúd 3, ktorý symbolizuje prúd bez interakcie, vtedy výraz OUT nadobudne tvar $OUT=k_{ii}\cdot\sqrt{h_i(t)}$. Výnimku pre výstupný prúd zo zásobníka *i* tvorí posledný zásobník v sérií, ktorý ma vždy výtok bez interakcie.

V súvislosti s interakciou sa pri zásobníkoch s interakciou uvažuje ešte aj možnosť výberu smeru toku v situácií, ak je rozdiel výšok hladín medzi zásobníkmi záporný. Teda, ak je výška v nasledujúcom zásobníku vyššia ako v aktuálnom, tak tlak kvapaliny v nasledujúcom zásobníku je vyšší (pre zásobníky uvažujeme rovnaký tlak nad hladinou), teda kvapalina nebude tiecť z aktuálneho zásobníka do nasledujúceho ale naopak. Tento jav budeme volať obojsmerný tok. Ak by sme takúto možnosť nechceli, tak povieme, že uvažujeme len jednosmerný tok a vtedy výtok z aktuálneho zásobníka do nasledujúceho je rovný nule a tok z nasledujúceho zásobníka do aktuálneho zásobníka je rovný tiež nule. Potom pri interakcií platí $\Delta h=h_i(t)-h_{i+1}(t)$, ak $\Delta h<0$ a uvažujeme iba jednosmerný tok, tak potom výraz

OUT vyjadrujúci výtok zo zásobníka je rovný nule, teda zásobník má len prítoky. Ak uvažujeme obojsmerný tok a $\Delta h<0$ tak výstupný prúd bude vyjadrený výrazom $OUT=-k_{ii}\cdot\sqrt{|(h_i(t)-h_{i+1}(t))|}$, teda zásobníky sa správajú ako jedno zariadenie. Podobne to platí pre vstupný tok z predchádzajúceho zásobníka $\Delta h=h_{i-1}(t)-h_i(t)$, ak $\Delta h<0$ a uvažujeme iba jednosmerný tok, tak potom výraz IN vyjadrujúci vtok z predchádzajúceho zásobníka, je rovný nule, ak by sme uvažovali obojsmerný tok, tak $\mathrm{IN}=-k_{i-1,i-1}\cdot\sqrt{|h_{i-1}(t)-h_i(t)|}$, čo znamená, že vstupný prúd sa bude správať ako výstupný. Celý spôsob výberu výrazu OUT a IN pre druhý až *n*-tý zásobník vyjadrujú pomocou stromov vetvenia obr. 9 a obr. 10.

Vstupné objemové prietoky $q_0(t)$, $q_{01}(t)$, $q_{02}(t)$, ..., $q_{0n}(t)$ predstavujú vstupy a výška hladiny $h_i(t)$ v zásobníkoch je výstupná a stavová veličina.

Keď sa v sérii nachádzajú všetky zásobníky bez interakcie, potom po úpravách z matematického modelu pre ustálený stav získame vzťahy na výpočet výšok hladín v ustálenom stave

$$h^s{}_1=\left(\frac{q^s{}_0+q^s{}_{01}}{k_{11}}\right)^2 \quad , \quad h^s{}_2=\left(\frac{q^s{}_0+q^s{}_{01}+q^s{}_{02}}{k_{11}}\right)^2 \quad ,\ldots, \quad h^s{}_i=\left(\frac{q^s{}_0+\sum_{j=1}^{n} q^s{}_{0j}}{k_{ii}}\right)^2 \tag{37}$$

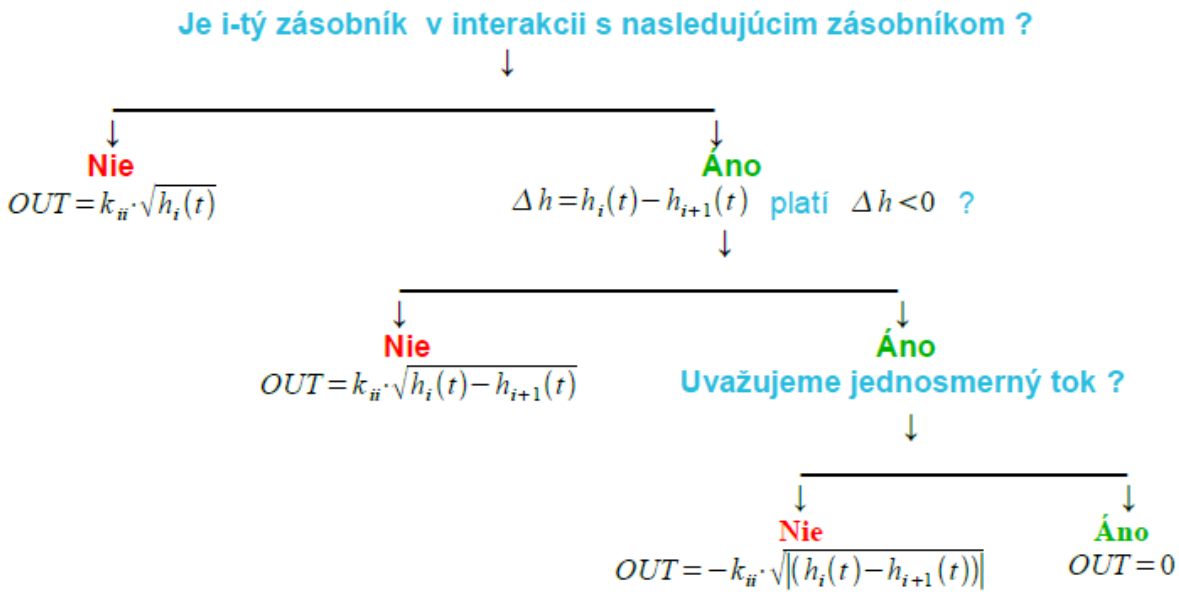
Podobne pre konfiguráciu, keď sú všetky zapojené zásobníky s interakciou, pre prvý až *n*-1. zásobník platí

$$h^s{}_i=\left(\frac{q^s{}_0+\sum_{j=1}^{i} q^s{}_{0j}}{k_{ii}}\right)^2+h^s{}_{i+1} \tag{38}$$

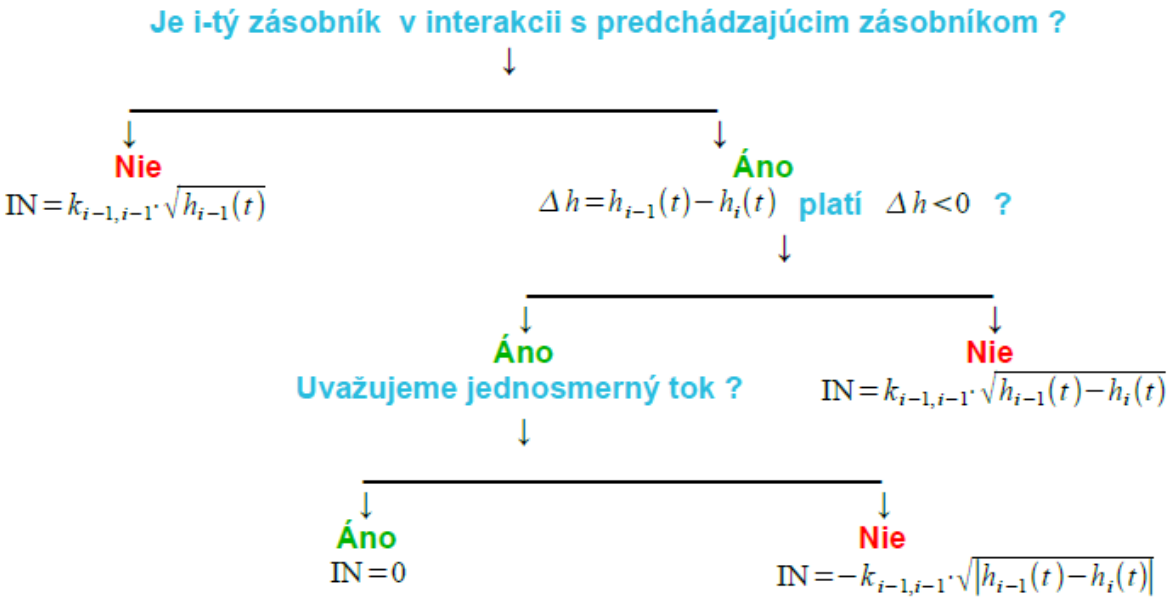
pre *n*-tý zásobník platí:

$$h^s{}_n=\left(\frac{q^s{}_0+\sum_{j=1}^{n} q^s{}_{0j}}{k_{nn}}\right)^2 \tag{39}$$

Pre inú konfiguráciu interakcií sú vzťahy rôzne a pre výpočet ustálených stavov v rôznej konfigurácií sa používa priamo simulácia.



Obr. 9 Strom vetvenia s pravidlami pre výraz OUT



Obr. 10 Strom vetvenia pre výber výrazu IN

2.3 Simulácia - jadro virtuálnych laboratórií

Simulácia predstavuje najdôležitejšiu časť virtuálnych laboratórií nakoľko sa jedná o matematické riešenie modelov, ktoré opisujú simulovaný objekt či jav. Od presnosti modelu, ktorý opisuje náš simulovaný objekt či jav a od správnosti, presnosti riešenia tohto modelu sa teda odvíja ako sa výsledkami približujeme k

reálnemu objektu či javu.

Práca sa venuje najmä procesom, ktoré sú opísané jednou alebo viacerými obyčajnými diferenciálnymi rovnicami prvého rádu [10]. Na riešenie obyčajných diferenciálnych rovníc prvého rádu sa vybrala metóda Runge-Kutta štvrtého rádu. Je to numerická metóda, ktorá hodnotu derivácie funkcie nahradzuje aproximáciou v určitých bodoch. Všeobecne, pre obyčajnú diferenciálnu rovnicu prvého rádu (39) so začiatočnou podmienkou (40), kde rovnice (41) až (44) predstavujú čiastkové funkcie a *h* je veľkosť kroku. Potom hodnota našej funkcie je daná vzťahom (45), kde x_n predstavuje hodnotu funkcie v predošlom bode funkcie a Δx je prírastok respektíve zmena hodnoty funkcie.

$$x' = f(t, x) \tag{39}$$

$$x(t_0) = x_0 \tag{40}$$

$$k_1 = f(t_n, x_n) \tag{41}$$

$$k_2 = f\left(t_n + \frac{h}{2}, x_n + \frac{h}{2} \cdot k_1\right) \tag{42}$$

$$k_3 = f\left(t_n + \frac{h}{2}, x_n + \frac{h}{2} \cdot k_2\right) \tag{43}$$

$$k_4 = f\left(t_n + h, x_n + h \cdot k_3\right) \tag{44}$$

$$x_{n+1} = x_n + \Delta x = x_n + \frac{h}{6} \cdot (k_1 + 2 \cdot k_2 + 2 \cdot k_3 + k_4) \tag{45}$$

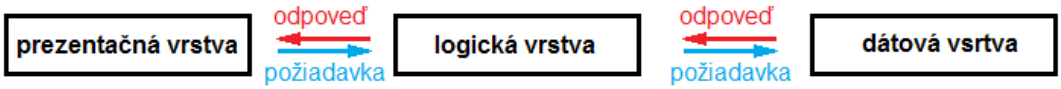
Pri riešení treba pamätať na to, že veľkosť kroku *h* priamo ovplyvňuje presnosť výpočtu.

2.4 Použité technológie

Výber použitých nástrojov, technológií súvisí priamo s účelom a funkciami

stránky. Základný výber je daný troj-vrstvovou štruktúrou stránky, ilustrovanou na obr. 11. Užívateľ je v styku s prezentačnou vrstvou, ktorá zaznamenáva vstupy od užívateľa a zobrazuje výstupy z logickej vrstvy. Logická vrstva spracuje vstupné údaje z prezentačnej vrstvy a zároveň ich uloží do dátovej vrstvy, ktorú predstavuje databáza.

Základom našej stránky je HTML, ktoré tvorí statickú časť a je na strane užívateľa. HTML je doplnené o štýly pomocou CSS a statickú časť doplna aj SVG, ktoré slúži na tvorbu obrázkov schém. O dynamickú časť na strane užívateľa sa stará JavaScript doplnený o metódu AJAX, ale primárne vykonáva dynamickú časť jazyk PHP, ktorý je na strane servera. Na ukladanie dát slúži databáza, ktorá je rovnako ako PHP na strane servera.



Obr. 11 Troj-vrstvová štruktúra stránky

2.4.1 HTML

[11]

HTML, Hypertextový značkový jazyk slúžiaci predovšetkým na tvorbu štruktúry stránky. Formátovanie štruktúry predstavuje rozlíšenie dokumentu – stránky pomocou značiek. Značky môžu byť párové alebo nepárové. Párové značky máju otváraciu značku napr. <html> a zatváraciu </html>. Časť stránky ktorá má byť zobrazená, štruktúrovaná pomocou HTML sa ohraničuje do párovej značky <html>, pred ktorou musí byť špecifikácia dokumentu <!DOCTYPE>, ktorá slúži webovým prehliadačom na určenie verzie HTML a teda správne zobrazovanie dokumentu HTML. HTML 4.01 pozná tri možné druhy tejto špecifikácie Strict, Transitional,

Frameset. Špecifikácie sa líšia požiadavkou na správnosť zápisu značiek, požadovaním CSS pre tvorbu štýlov a zakázaním niektorých značiek. Časť HTML dokumentu ohraničená párovou značkou <html> obsahuje dve párové značky. Prvá značka <head> definuje hlavičku HTML dokumentu a v rámci nej sa nachádza párová značka <title> ktorá slúži na definovanie názvu HTML dokumentu zobrazeného v prehliadači. Druhá značka <body> predstavuje samotné telo dokumentu a obsahuje zvyšný dokument, teda jeho obsahovú časť, ako aj značky, ktoré slúžia na formátovanie štruktúry obsahu. Prehľad najčastejšie používaných značiek použitých pri tvorbe webovej stránky v rámci diplomovej práce možno vidieť v tabuľke 2

Tabuľka 2 Prehľad HTML značiek

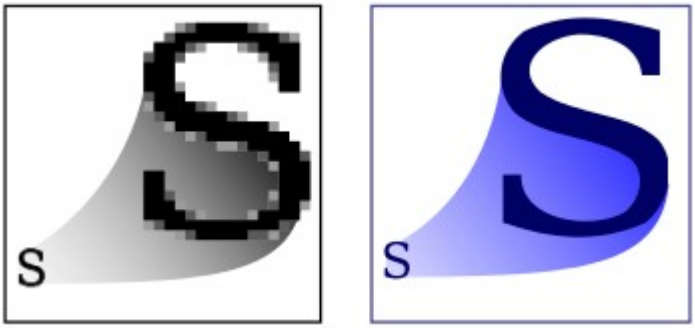
značka	**párová**	**najčastejší atribút**	**účel**
<a>	Áno	href	Tvorba odkazov
 	Nie	-	Zlomenie riadka
	Áno	id	Špecifikuje časť HTML dokumentu
<form>	Áno	action, id, method	Tvorba formulára
<h1>,..., <h6>	Áno	-	Tvorba nadpisov
<input>	Áno	Type, id	Tvorba formulárových prvkov
<option>	Áno	value, selected	Tvorba prvkov výberového menu
<p>	Áno	-	Definovanie odseku
<script>	Áno	type, src	Definovanie skriptu
<select>	Áno	id	Tvorba výberového menu
<style>	Áno	type	Definovanie štýlu
<sub>	Áno	-	Definovanie dolného indexu
<sup>	Áno	-	Definovanie horného indexu
<table>	Áno	-	Definovanie tabuľky
<td>	Áno	colspan, rowspan	Definovanie stĺpca v riadku tabuľky
<tr>	Áno	-	Definovanie riadku tabuľky

Problematika HTML je obsiahla. Jej plné pochopenie vyžaduje samostatnú prácu. Podrobnejší prehľad značiek, ich atribútov ako aj vzorové príklady použitia sú na stránke http://www.w3schools.com/html/default.asp.

2.5.2 SVG

[12,13]

Scalable Vector Graphics – škálovateľná vektorová grafika. SVG je značkový jazyk typu XML. V SVG sa grafika vytvára pomocou značiek, ktoré reprezentujú základné geometrické objekty, ktorým možno priradiť rôzne vlastnosti pomocou atribútov. Výhoda SVG je možnosť zväčšovania zobrazenej grafiky bez straty kvality, ako to je ilustračne zobrazené na obr. 12, kde vľavo je rastrová grafika a vpravo vektorová.



Obr. 12 Porovnanie zväčšovania rastrovej a SVG grafiky [12]

SVG element je vždy ohraničený párovou značkou <SVG>, ktorá obsahuje atribúty obsahujúce šírku a výšku tohto elementu. Potom v rámci ohraničenia touto značkou sú značky, ktoré slúžia na vykreslenie geometrických objektov. Základom každého objektu sú jeho súradnice, prípadne rozmery v *pixeloch*. Teda značky týchto geometrických objektov obsahujú jeden alebo viacero bodov definovaných *x* a *y* súradnicami, ktoré majú počiatok v ľavom hornom rohu SVG elementu. Teda hodnota súradnice *x* rastie zľava doprava, ale hodnota súradnice *y* rastie smerom z hora nadol vzhľadom na obrazovku. Prehľad najčastejšie používaných značiek použitých pri tvorbe webovej stránky v rámci diplomovej práce možno vidieť v tabuľke 3.

Tabuľka 3 Prehľad SVG značiek

značka	**základný atribút**	**účel**
<animate>	from,to,dur	Definovanie animácie
<circle>	cx,cy,r	Definuje kružnicu
<defs>	-	Element slúžiaci na definíciu špeciálnych elementov napr. gradientu
<line>	x1,x2,y1,y2	Definuje úsečku
<linearGradient>	Id,x1,x2,y1,y2	Definuje lineárny gradient
<path>	d	Definuje krivku
<polygon>	points	Definuje polygon
<polyline>	points	Definuje viacero spojených úsečiek
<rect>	x,y,width,height	Definuje pravouhlý štvoruholník
<text>	x,y	Definuje text

Základným geometrickým objektom v SVG môžeme upraviť vzhľad pomocou atribútov, ktoré upravujú štýl. Najčastejšie sa jedná o farbu a priehľadnosť čiar, ich hrúbku, prípadne farebnú výplň objektov, ktoré môžu byť vyplnené, napr. kružnica. Podobne je možné meniť aj štýl u písma. V tabuľke 4 je výber najčastejšie používaných atribútov v diplomovej práci.

Tabuľka 4 Prehľad SVG atribútov

atribút	**účel**
font-family	Definuje typ písma
font-size	Definuje veľkosť písma
begin	Definuje začiatok zmeny atribútu v čase
dur	Trvanie animácie
from	Definuje z akej hodnoty sa zmení atribút
to	Definuje do akej hodnoty sa zmení atribút
opacity	Definuje priehľadnosť objektu kde 0 je plne priehľadný a 1 je nepriehľadný
fill	Definuje farebnú výplň objektu
stroke	Definuje farbu čiary objektu
stroke-width	Definuje hrúbku čiary objektu
stroke-dasharray	Definuje prerušovanú čiaru

2.5.3 CSS

[14]

CSS – Cascading Style Sheets alebo kaskádové štýly. CSS sa primárne stará o vizuálnu stránku HTML dokumentu, resp. jeho obsahu. CSS predstavuje rozšírenie

k HTML, hlavným dôvodom pre zavedenie CSS je jednoduché formátovanie vizuálnej stránky HTML. Medzi hlavné výhody CSS patrí:

- štýl nie je nutné definovať pri každom elemente, štýl platí pre celý dokument
- možnosť špecifikácie použitia štýlu pomocou tried, pseudotried a identifikátorov
- štýl možno definovať vo vlastnom súbore - oddelenie CSS kódu od HTML dokumentu, teda väčšia prehľadnosť
- jednoduchšia manipulácia ako pri starom formátovaní pomocou HTML
- možnosť používania jedného štýlu v rámci celej stránky a pritom pracovať stále s jedným súborom
- širšie možnosti vo formátovaní

Základná syntax pre zápis štýlu pomocou CSS je selektor { vlastnosť: hodnota vlastnosti }. Selektor definuje, pre ktorý prvok element definujeme štýl. Vo vnútri zložených zátvoriek nasleduje vlastnosť a jej hodnota, ktorú chceme danému prvku, ktorý sme vybrali pomocou selektora priradiť. Pričom vo vnútri môže byť aj viacero vlastností ktoré formátujeme.

2.5.4 PHP

[15]

PHP v origináli Personal Home Page, neskôr PHP: Hypertext Preprocessor predstavuje skriptovací jazyk bežiaci na strane servera. Je určený pre tvorbu dynamických webových stránok. Jeho výhodou je, že komunikuje s relačnými

databázami. Medzi hlavné funkcie PHP patrí:

- spracovanie formulárov
- generovanie dynamického obsahu HTML dokumentu
- komunikácia s databázou

V diplomovej práci predstavuje PHP najpoužívanejšiu technológiu a prostredníctvom PHP sa zabezpečujú hlavné výpočtové operácie.

2.5.5 JavaScript

JavaScript je skriptovací jazyk používaný pri tvorbe webových stránok. Výhodou jazyka JavaScript oproti PHP je, že beží na strane užívateľa, vďaka tomu je možné použiť JavaScript na mnoho úloh ktoré PHP ako prostriedok na strane servera nemôže vykonať. Napríklad kontrola údajov formulára ešte pred odoslaním na spracovanie, zmena obsahu HTML dokumentu, zmena štýlu HTML dokumentu a to všetko bez nutnosti komunikácie so serverom.

2.5.6 AJAX

[16, 17]

AJAX (Asynchronous JavaScript + XML) predstavuje kombináciu technológií JavaScript a XML. Princíp spočíva vo vymedzení časti HTML dokumentu, ktorá sa pri určitej udalosti asynchrónne načíta – obnoví. To znamená, že obsah časti ošetrenej pomocou AJAX odošle údaje na server, server tieto údaje spracuje a na AJAX ošetrenej časti sa zo strany serveru vrátia už spracované údaje. Rozdiel medzi asynchrónnym odosielaním údajov a klasickým teda je, že pri klasickom odosielaní sa musí obnoviť celá stránka, kým pri asynchrónnym len určitá časť.

AJAX teda pridáva na komfortnosti pri práci s webovou stránkou. Jeho použitie je vhodné najmä pri rozsiahlych formulároch, ktoré obsahujú rôzne možnosti, ktoré formulár dynamicky menia a pridávajú či uberajú na jeho prvkoch. Postup celej akcie asynchrónneho načítania stránky možno zjednodušene opísať nasledovne, pri udalosti, ktorá sa týka špecifikovaného objektu napríklad formulárového prvku radio sa vytvorí požiadavka, ktorá sa odošle na server, na strane servera sa spracuje, vráti na stranu prehliadača, kde sa pomocou JavaScript údaje zo servera spracujú a na základe nových údajov sa špecifikovaná časť stránky obnoví.

2.5.7 jQuery

[18]

Predstavuje knižnicu funkcií v jazyku JavaScript. Pre jej použitie na stránke je nutné odkazovať sa na jej súbor, ten je možné si stiahnuť zo stránky http://jquery.com/, prípadne je možné odvolávať sa na verziu, ktorá je dostupná na internete. jQuery značne uľahčuje manipuláciu so stránkou a zároveň je ľahko použiteľná, čo je jej výhoda oproti samotnému jazyku JavaScript. Umožňuje jednoduché pridávanie rôznych animácií, manipuláciu s prvkami dokumentu, úpravu štýlov a rôzne iné úpravy. Základná syntax použitia jQuery je $(selector).action(), kde symbol dolár definuje použitie jQuery, v oblých zátvorkách nasleduje selektor čo predstavuje výber, pre ktorý sa akcia udeje a action() reprezentuje akciu ktorá sa má vykonať alebo udalosť, ktorá spustí ďalšiu akciu v rámci metódy jQuery. Ako príklad je vhodné uviesť si základnú funkciu, ktorá slúži na pozdržanie jQuery metód vo vnútri zápisu až do doby, kým celá stránka nie je úplne nahratá. Zápis vyzerá nasledovne:

```
$(document).ready(function(){
```

```
//jQuery metódy

});
```

Zo zápisu vidno, že selektor je dokument a akcia je pripravený, teda ak je dokument pripravený potom sa vykonajú metódy vo vnútri. Na tomto krátkom príklade vidno, ako ma jQuery prirodzene a dobre čitateľnú syntax.

Dobré možnosti jQuery potvrdzuje aj široké možnosti, ktoré ponúka výber pomocou selektora. V tabuľke 5 je krátky prehľad možností.

Tabuľka 5 Prehľad jQuery selektorov

Zápis selektora	**Príklad**	**Výber**
*	*	Vyberie všetky elementy
.class	.tabulka	Vyberie elementy s triedou tabulka
element	a	Vyberie všetky elementy a
#ID	#vstup	Vyberie všetky elementy s id=”vstup”
this	this	Vyberie aktuálny element
element.class	td.nazov	Vyberie všetky elementy td s triedou nazov
element:first	td:first	Vyberie prvý element td
:checked	:checked	Vyberie všetky vybraté vstupné elementy
:header	:header	Vyberie všetky elementy tipu nadpis (<h1>,<h2>, ...)
element1,element2,element3,...elementN	table,tr,td	Vyberie všetky elementy table, tr, td

Tento krátky výber ukazuje bohatosť pri výbere pomocou selektoru, avšak možnosti sú omnoho väčšie. Ďalšie možnosti sú k dispozícií na stránke http://w3schools.com/jquery/jquery_ref_selectors.asp

Podobné široké možnosti ponúka aj výber udalostí.

2.5.8 Flot

[19]

Flot predstavuje knižnicu funkcií vytvorenú v JavaScripte pre jQuery metódy. Slúži na tvorbu grafov. Ponúka veľké možnosti v nastaveniach grafov ako aj interakcie priamo v grafe ako je napríklad prepínanie zobrazených závislostí, zmenu

veľkosti zobrazeného grafu, približovanie, zobrazovanie x, y súradníc grafu a iné. Pre použitie Flot je potrebné mať knižnicu jQuery a samozrejme aj knižnicu Flot. Podobne ako jQuery ma Flot dobre pochopiteľnú syntax. Hlavný dôvod pre použitie knižnice Flot je jednoduchosť a dobré možnosti. V našom prípade sa jedná o možnosť približovania. Flot požaduje oblasť, v ktorej sa graf bude nachádzať. To sa realizuje priradením identifikátora do elementu v značke ako aj priradeniu rozmerov tohto elementu pomocou štýlu. Časť HTML dokumentu, kde sa bude tento element nachádzať, bude zároveň miestom, kde sa zobrazí výsledný graf s rozmermi, ktoré sa elementu priradili. Zároveň potrebuje údaje v tvare dvojíc x a y hodnôt. Napríklad štyri takéto body v premennej majú aj s deklarovaním premennej d a priradením do nej nasledovný zápis

var d = [[1, 8], [0, 9], [1, 9], [8, 9]]; kde prvé číslo z dvojice je x súradnica a druhé číslo je y. Potom pre vykreslenie grafu stačí zápis: $.plot("#identifikator", [d1]); pričom za symbolom mriežka nasleduj identifikátor zadefinovaného elementu.

2.5.9 MySQL

MySQL je relačný databázový systém pracujúci pomocou SQL dopytov, teda je to nástroj na správu databázy. Relačný databázový systém je systém, ktorý ukladá údaje do tabuliek ktoré sú navzájom prepojené.

3 Analýza problému

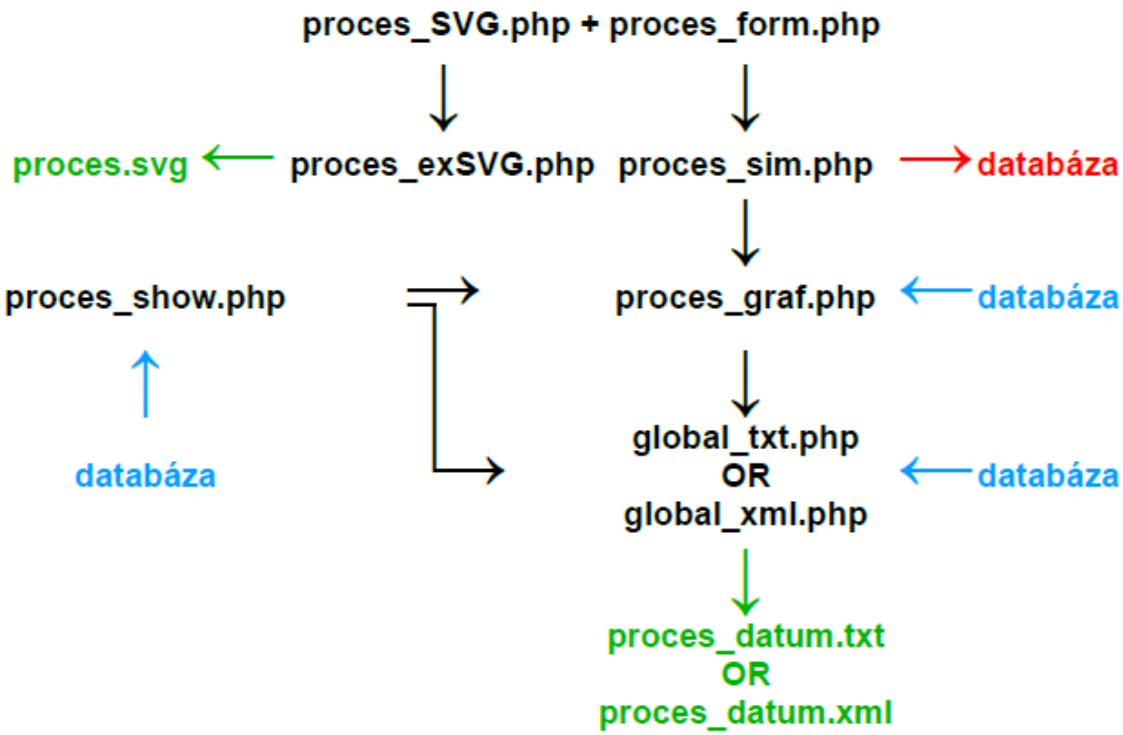
Hlavným cieľom analýzy bolo stanoviť problémy súvisiace s dosiahnutím cieľov práce, navrhnúť a zhodnotiť možné riešenia a na základe toho vybrať konkrétne riešenie, ktoré vyhovuje našim požiadavkám. V prípade úloh programovania existuje takmer vždy viacero riešení a líšia sa najmä zložitosťou a univerzálnosťou riešenia. Základný krok pri návrhu programového riešenia ostáva pochopenie riešenia vo všeobecnosti, či už ako logického alebo matematického. Samotný zápis programu býva už vec skúseností a praxe.

3.1 Štruktúra webovej stránky

Analýza základnej štruktúry webovej stránky vzhľadom na všeobecný proces nám ponúkne istý nadhľad, obraz o tom ako stránka vyzerá z pohľadu prepojenosti jednotlivých jej funkcií a pomôže riešiť problémy prirodzene ako nasledujú.

Úvodná stránka obsahuje menu. Menu obsahuje jednotlivé procesy a v rámci jednotlivých procesov je pod-menu. Pod-menu obsahuje výber novej simulácie, alebo zobrazenie zoznamu už realizovaných simulácií. Priblížime si prvú možnosť, teda možnosť novej simulácie pre všeobecný proces. Nová simulácia začína vstupným formulárom. Náš vstupný formulár je generovaný pomocou súboru *proces_form.php* a *súboru proces_SVG.php*. V prehliadači to je z pohľadu užívateľa HTML dokument. V hornej časti obsahuje schematické znázornenie procesu pomocou vektorovej grafiky a vo zvyšnej časti najmä textové a výberové formulárové prvky. Časť dokumentu, ktorá obsahuje SVG, umožňuje exportovať ilustračnú schému zariadenia do formátu SVG prostredníctvom súboru *proces_exSVG.php*. Po vyplnení údajov do formulára a odoslaní nasleduje spracovanie údajov pomocou súboru *proces_sim.php*. Súbor *proces_sim.php* predstavuje simulačnú časť v rámci

Modeltool. Po úspešnom spracovaní ukladá údaje do databázy. Hneď po spracovaní údajov zo simulácie nasleduje presmerovanie na súbor *proces_graf.php*. Tento súbor nahrá z databázy údaje k simulácií a na základe nich vykreslí graf. Zároveň pri voľbe užívateľa realizuje presmerovanie na export výstupov zo simulácie do txt súboru prípadne XML. Export sa realizuje pomocou súboru *global_txt.php*, alebo *global_xml.php*. Druhá možnosť v prípade pod-menu pre proces je zobraziť zoznam simulácií. Tento zoznam generuje súbor *proces_show.php.* Jeho výstupom je zoznam simulácií. Zoznam sa vytvára na základe údajov z databázy pre konkrétny proces. Teda výberom zo špecifickej tabuľky procesu. V rámci tohto zoznamu je možné exportovať údaje pre vybranú simuláciu zo zoznamu. Realizuje sa to rovnakými súbormi (*global_txt.php* a *global_xml.php*) ako v predošlom prípade. V zozname simulácií je zároveň odkaz na graf príslušnej simulácie. Celú štruktúru ilustruje obr. 13, kde modrou farbou je znázornené nahrávanie údajov z databázy, červenou zápis do databázy a zelenou export súborov. Túto základnú štruktúru majú všetky procesy realizované v Modeltool. Tu si treba uvedomiť, že každý proces má vlastný súbor na jednotlivé akcie, s výnimkou exportu údajov do xml a txt súboru.



Obr. 13 Základná štruktúra stránky pre všeobecný proces

3.2 Úlohy Formulárov

Formulár je prvé, čo užívateľ po výbere procesu, ktorý bude simulovaný vidí a s čím pracuje.

Jeho hlavné úlohy sú nasledovné:

- informačná úloha
  - jasné definovanie procesu
  - jasné definovanie vstupov vzhľadom na použitý model
- úloha zberu dát
  - zabezpečiť všetky potrebné údaje pre simuláciu
- úloha dizajnu

- jednoduchosť a prehľadnosť formulára
- užívateľské pohodlie

Nakoľko vstupné formuláre pre proces slúžia k získaniu údajov pre simuláciu, tak sa jedná o dôležitú časť v rámci celého VL. Ich obsah je závislý od matematického modelu procesu a parametrov simulácie. Všeobecne sú to:

- veličiny vystupujúce v diferenciálnych rovniciach opisujúcich proces
- veličiny potrebné k výpočtu ustálených stavov procesu
- skokové zmeny realizované na vstupoch procesu
- čas trvania simulácie
- krok simulácie h

Nakoľko procesy v kapitole 2.2 sú rôzne, rôzne budú aj ich formuláre. Pri sériovo zapojených zariadeniach sa vstupné údaje pre simuláciu odvíjajú od počtu zariadení. Rovnako je to pri reaktore s viacerými reakciami a zložkami. Aké sú riešenia ? Vytvorenie formulárov pre všetky kombinácie a možnosti. Takéto riešenie je pracné a komplikované, teda nevyhovujúce. Ďalšie možné riešenie je vytvoriť vstupné prvky vo vektorovom tvare a vstupy zadávať vo vektorovom tvare. Napríklad pre *n* zásobníkov zadávanie plochy prierezu by vyzeralo nasledovne [F_1 F_2 … F_n], kde F je hodnota prierezu zásobníka. Toto riešenie síce robí formulár nezávislí od počtu parametrov avšak je to menej prehľadné a vhodné skôr pre procesy kde je tých parametrov mnoho. Riešenie sú dynamické generované formuláre, generujúce sa už na základe údajov, ktoré užívateľ poskytuje napr. o počte zariadení.

3.2.1 SVG schémy v rámci Modeltool

SVG schémy pomáhajú naplniť informačnú úlohu formulárov. Ich úloha spočíva najmä v podpore predstavy o procese. Hlavne z hľadiska prúdov, teda toho čo vstupuje a čo zo zariadenia odchádza, prípadne parametrov zariadenia. Aby dobre plnili účel musia spĺňať tieto pravidlá:

- musia byť prehľadné
- značenie musí odpovedať formulárom
- konfigurácia musí odpovedať výberu užívateľa
- schéma musí vhodne opisovať proces, ktorý znázorňuje z pohľadu modelu
- mali by mať spoločný vizuálny štandard
- nesmú pôsobiť rušivo

Najvýraznejší vplyv majú v prípade zásobníkov, kde sa konfigurácia interakcií medzi zásobníkmi premieta aj do schémy.

3.3 Interpretácia metódy Runge-Kutta

Vo východzej verzii Modeltool sa realizovala simulácia prepisom rovníc (41) až (45) do PHP kódu s použitím funkcie modelu. Výsledný zápis bol vhodný na riešenie, kde bol vopred známy počet DR, ktoré sa budú riešiť. Nakoľko niektoré nové procesy, ktoré sa v rámci práce do Modeltool pridali, nemajú vopred určený počet DR, bolo nutné navrhnúť riešenie, ktoré by bolo nezávislé. Túto nezávislosť zaručí vytvorenie funkcie, ktorá bude reprezentovať metódu Runge Kutta. Zavolaním tejto funkcie pomocou cyklu v rámci jedného simulačného kroku toľko krát koľko máme DR na riešenie zabezpečí nezávislosť. Použitie funkcie zároveň redukuje kód

a uľahčuje pridávanie ďalších procesov. Funkcia bude vtedy dostatočne univerzálna, ak nebude viazaná k rovniciam modelu. Preto vytvorená funkcia nesmie obsahovať pevne napísaný model, iba sa naň prostredníctvom funkcie odvolávať. Vstupné parametre tejto funkcie musia potom byť stavy a vstupy, ktoré sa v rámci funkcie použijú.

3.4 Možnosti na výstupe zo simulácie

Východisková verzia Modeltool po zrealizovaní simulácie ponúkla užívateľovi k dispozícií závislosti vstupov a výstupov v podobe statických grafov. Tieto grafy sa na stránke zobrazili ako obrázky. Síce dobre plnili svoj účel, ale neponúkali širšie možnosti ako približovanie, alebo zobrazovanie súradníc, či rôzne iné nastavenia. Moderné grafy v prostredí webového prehliadača už ponúkajú tieto možnosti. Preto implementácia nového spôsobu pre vykresľovanie grafov s cieľom zlepšiť možnosti pre užívateľa, je jedným z hlavných cieľov vo výstupnej časti Modeltool. Pre užívateľa je často dôležité, aby získané údaje nielen videl, ale mal aj možnosť s nimi pracovať, napríklad získané údaje identifikovať. Vtedy je dobré mať možnosť exportu dát, Modeltool v základe ponúkal export dát do súboru XLS. Do tohto formátu sa exportovali kompletné informácie o simulácií. Každý proces však vyžadoval napísanie súboru ktorý by export vykonal, čo predstavovalo pracnú záležitosť. Aby sa export dát uľahčil a zároveň bol použiteľný pre ľubovoľný proces, tak bolo nutné navrhnúť všeobecnú funkciu ktorá zabezpečila export.

3.5 Špecifikácia požiadaviek

Analýzou jednotlivých častí Modeltool ako VL, z ohľadom na špecifikácie, ktoré prinášajú nové procesy, ako aj s ohľadom na skvalitňovanie nástroja z pohľadu užívateľa, sa už s ohľadom na možné technológie, metódy a riešenia vybrali tieto

čiastkové ciele a ich riešenia:

■ Každý proces bude doplnený o ilustračnú schému procesu, ktorá bude obsahovať značenie, ktoré odpovedá značeniu použitému pri vstupných formulároch. Schéma sa bude realizovať prostredníctvom jazyka SVG.

■ Vybrané procesy, pri ktorých môže byť schéma procesu rušivá vzhľadom na veľkosť, bude možnosť túto schému schovať. Tu sa využije knižnica jQuery

■ Každý formulárový prvok, popis alebo schéma zariadenia závislá od samotného formulára, sa bude meniť v závislosti od výberu užívateľa, tak aby užívateľ pracoval stále s verziou formulára, ktorá odpovedá zvolenej konfigurácií. To sa bude realizovať metódou AJAX.

■ V rámci simulácie sa bude dbať najmä na riešenia, ktoré budú nezávislé a budú použiteľné aj pri iných procesoch s cieľom zjednodušiť zavádzanie ďalších modulov do Modeltool

■ Každý proces bude mať graf s funkciou priblíženia a zobrazovaním x, y súradníc. Graf bude realizovaný univerzálnou funkciou tak, aby bol pri zmene vstupných parametrov použiteľný pre ľubovoľný proces. Pri samotnom vykresľovaní grafu sa použije knižnica Flot.

■ Export údajov sa bude realizovať pomocou jedného spoločného súboru s cieľom zjednodušiť pridávanie exportu pre procesy.

4 Praktická časť

V praktickej časti sa realizujú konkrétne riešenia jednotlivých cieľov. Praktickú časť možno rozdeliť do troch častí:

- Programovacia časť, týka sa všetkých častí súvisiacich s vytváraním modulu
- Verifikačná časť, v tejto časti sa výsledky, ktoré poskytujú nové moduly v Modeltool porovnávajú s výsledkami ktoré sú získané programom MATLAB/Simulink
- Manuál, obsahuje krátky popis vybraného modulu a postup inštalácie Modeltool

4.1 Programovacia časť

Zápis kódu bol realizovaný vo voľne šíriteľnom programe PSPad verzie 4.5.6. Stránka sa počas tvorby zobrazovala v prehliadači Mozilla Firefox. Úlohu virtuálneho servera s databázovým systémom MySQL plnil XAMPP verzie 1.7.7. Exportované schémy vo formáte SVG sa pri validácií zobrazovali pomocou programu Inkscape verzie 0.48

Všetky nové, ale aj staré procesy majú spoločné znaky z pohľadu tvorby modulov. Preto je na vybranom procese vhodné ukázať si celkový postup tvorby modulu. Ukázať si spoločné znaky všetkých modulov a prípadne spomenúť špecifikácie modulov.

4.1.1 Realizácia SVG schém

SVG schémy sa generujú pomocou PHP kódu. Celý zápis značiek je realizovaný pomocou PHP príkazu echo. Každý modul má svoj vlastný súbor ktorý generuje SVG schému. Pred začatím zápisu schémy sa vždy zvolila predloha, podľa

ktorej sa bude postupovať. Pri SVG schémach si treba uvedomiť, že sa vykresľujú do rozmerovo definovaného okna, ktoré má definovanú výšku a šírku, teda rozmery SVG obsahu nám tvoria hranice. Časti vykreslených objektov, ktoré presiahnu tieto hranice nie sú zobrazené. Pri vykresľovaní SVG schém sme ako prvý krok vhodne určili tieto rozmery. Pri procesoch, kde bola schéma statická, teda nemenila sa podľa nastavenia užívateľa, sa rozmery jednoducho číselne zadali podľa rozmerov schémy s ohľadom na istú rezervu. Pri procesoch, kde sa schéma mení podľa voľby užívateľa sa výška a šírka SVG obsahu musela meniť v závislosti od tejto voľby. Jednoduchým príkladom je schéma pre dva zásobníky kvapaliny. Pri tomto procese sa rozmery SVG obsahu menia v závislosti od toho či užívateľ zvolí možnosť interakcie alebo nie. Súbor, ktorý generuje schému, obsahuje na začiatku PHP kód, ktorý na základe podmienky posúdi, o ktorú možnosť sa jedná. Do premenných jazyka PHP priradí príslušné číselné hodnoty. Číselné hodnoty sú určené pre konkrétnu možnosť. Potom zadefinovanie rozmeru SVG obsahu vyzerá nasledovne:

```
echo '<svg id="SVG1" width="'.$sirka.'" height="'.$vyska.'" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink">';
```

V atribútoch *width* a *height* sú PHP premenné obsahujúce číselné hodnoty pre šírku a výšku SVG obsahu. Po správnom definovaní týchto hraníc, nasleduje zápis značiek, ktoré tvoria samotnú schému. Všetky tieto značky sú ohraničené značkou <svg>, v ktorej sme definovali rozmery SVG obsahu.

Vykreslenie schémy jedného, dvoch alebo N zásobníkov je identické z pohľadu kreslenia zariadenia. Rozdiel tvorí počet zariadení v schéme, ich značenie ako aj poloha. Preto je riešením tvorby schémy pre zásobníky PHP funkcia. Funkcia obsahuje SVG značky, pomocou ktorých sa vykreslí jeden zásobník. Potom celú schému získame opakovaným volaním rovnakej funkcie s rôznymi vstupnými

parametrami.

Naša funkcia pre vykreslenie zásobníka má názov *zasobnik01.* Má tieto vstupné parametre *$xref*, *$yref*, *$i*, *$wall*, *$width*, *$height*, *$color*, *$wallfill*, *$q*, *$q1n*, *$q2*, *$q2n*, *$F*, *$Fn*, *$kii*, *$qout*, *$interakcia*. Ich účel je nasledujúci:

- *$xref* – hodnota referenčnej súradnice osi x. O túto hodnotu sa zväčšujú hodnoty x súradníc všetkých značiek funkcie *zasobnik01*
- *$yref* – hodnota referenčnej súradnice osi y. O túto hodnotu sa zväčšujú hodnoty y súradníc všetkých značiek funkcie *zasobnik01*
- *$i* – číslo zásobníka, vyjadruje jeho poradie v sérií. Slúži k číslovaniu konštánt ventilov ako aj výstupných prúdov
- *$width* – hodnota určujúca šírku zásobník*a*
- *$height* – hodnota určujúca výšku zásobníka
- *$wall* – hodnota určujúca šírku steny zásobníka
- *$color* – farba prúdov z (do) zásobníka a ich farba textu
- *$wallfill* – výplň steny (farba alebo gradient)
- *$q1* – zobrazovania prvého vstupného prúdu. Pri hodnote 0 sa prúd nezobrazí
- *$q2* – zobrazovania druhého vstupného prúdu. Pri hodnote 0 sa prúd nezobrazí

- *$qout* – zobrazovania výstupného prúdu. Pri hodnote 0 sa prúd nezobrazí
- *$q1n* – popis pre prvý vstupný prúd
- *$q2n* – popis pre druhý vstupný prúd
- *$F* – zobrazovanie prierezu zásobníka. Pri hodnote 0 sa prúd nezobrazí
- *$Fn* – popis prierezu zásobníka
- *$kii* – zobrazovanie symbolu ventilu a jeho popisu
- *$interakcia – určuje zalomenie vystupujúceho prúdu zo zásobníka.* Pri hodnote 1 má prúd *zalomenie*

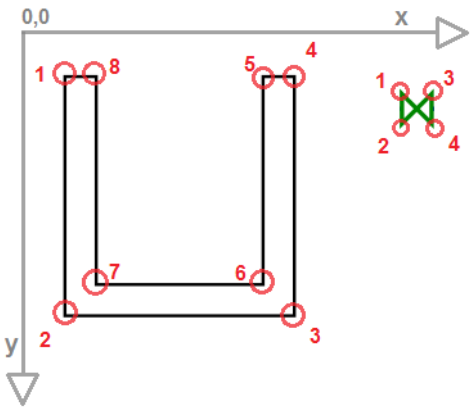
Samotná funkcia *zasobnik01* obsahuje sériu podmienok *if*, *else.* Podmienky sprístupňujú jednotlivé vetvy funkcie. Vetvy predstavujú možnosti, ktoré môže schéma nadobudnúť. V rámci týchto vetiev sú zapísané SVG značky pre jednotlivé geometrické objekty. Sú to predovšetkým značky *polygon*, *line*, *polyline* a *text*. Na značke *polygon* si opíšeme princíp akým sa určovali súradnice aj pri ostatných značkách. Túto značku sme v prípade funkcie *zasobnik01,* použili na vykreslenie tela zásobníka, symbol ventilu a trojuholníkov, ktoré predstavujú vrchol šípky, ktorá symbolizuje smer prúdu. Zápis pre vykreslenie tepla zásobníka je nasledovný:

```
echo '<polygon points="
'.($xref).','.($yref).'
'.($xref).','.($yref+$height+$wall).'
'.($xref+2*$wall+$width).','.($yref+$height+$wall).'
'.($xref+2*$wall+$width).','.($yref).'
'.($xref+$wall+$width).','.($yref).'
```

```
'.($xref+$wall+$width).','.($yref+$height).'
'.($xref+$wall).','.($yref+$height).'
'.($xref+$wall).','.($yref).'
" fill="none" stroke="black" stroke-width="1.5"/>';
```

Kde značka polygon obsahuje atribút points. Ten obsahuje jednotlivé súradnice, ktoré tvoria vrcholy mnohouholníka, ktorého grafiku značka tvorí. Pri použití značky polygon sa jednotlivé vrcholy automaticky spájajú úsečkou. Pravidlom aktuálny vrchol s nasledujúcim vrcholom. Výnimkou je posledný vrchol, ktorý sa spája s prvým. Dvojice súradníc pre vrcholy sme pre telo zásobníka zadávali tak, aby sa ich prepojenia nekrížili. Kríženie čiar, ale nepredstavuje chybu. V prípade symbolu ventilu sa využila práve táto možnosť. Poradie bodov pre telo zásobníka ako aj symbolu ventilu vidno na obr. 14.

Hodnoty všetkých x súradníc majú svoju hodnotu vyššiu o hodnotu referencie *$xref*, rovnako ako hodnoty y súradníc o referenciu *$yref*. Vzdialenosti medzi jednotlivými bodmi závisia od potreby. V prípade tela zásobníka sa použili parametre *$width*, *$height*, *$wall*, ktoré určovali jednotlivé vzdialenosti medzi bodmi jednoduchým prirátaním k danej súradnici. Napríklad vzdialenosť medzi vrcholom č.1 a č.4 na obr. 14 pre telo zásobníka je daná ako dvakrát šírka steny plus šírka zásobníka. Týmto jednoduchým spôsobom sme určili všetky súradnice vo všeobecnom zápise ako je uvedený v príklade pre telo zásobníka. Súradnice pritom môžu nadobúdať len celočíselné hodnoty.



Obr. 14 Porovnanie zadávania vrcholov pre rôzne geometrické objekty

Rovnakým spôsobom ako pri určovaní súradníc pre telo zásobníka sa určili aj ďalšie časti, ktoré tvoria kompletnú grafiku zásobníka. Celkovo bolo nutné všeobecne zadefinovať súradnice pre šestnásť častí tvoriacich výslednú grafiku zásobníka s popisom.

Okrem výplne jednou farbou je v SVG možné vytvoriť lineárny alebo radiálny gradient a ten použiť ako výplň. My sme si vytvorili funkciu pre lineárny gradient. Celý zápis tejto jednoduchej funkcie je nasledovný:

```
function grad($id,$from,$to,$x1,$y2,$x2,$y2)
    {
    echo "<defs>";
    echo '<linearGradient id="'.$id.'" x1="'.($x1).'%" y1="'.($y1).'%" x2="'.($x2).'%"
    y2="'.($y2).'%">';
    echo '<stop offset="0%" stop-color="'.($from).'" stop-opacity="1"/>';
    echo '<stop offset="100%" stop-color="'.($to).'" stop-opacity="1" />';
    echo "</linearGradient></defs>";
    }
```

Vstupnými parametrami tejto funkcie sú:

- *$id* – definuje identifikátor, pomocou ktorého gradient priradíme do výplne

- *$from – dolná hranica farebného rozsahu, v ktorom sa gradient mení. Zadáva sa farba.*

- *$to – horná hranica farebného rozsahu, v ktorom sa gradient mení. Zadáva sa farba.*

- *$x1, $x2* – určenie smeru zmeny gradientu, môžu nadobúdať hodnoty 0 až 100

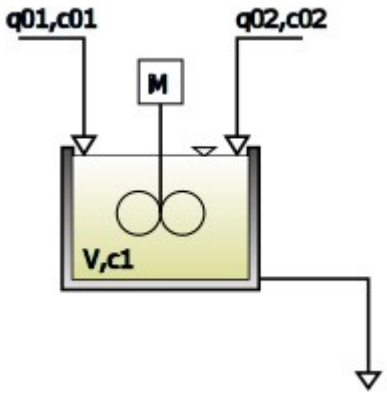
- *$y2, $y2* – určenie smeru zmeny gradientu, môžu nadobúdať hodnoty 0 až 100

Pričom ak $y1=$y2, $x1≠$x2 tak získame horizontálny gradient, mení sa v smere osi x. Naopak ak $y1≠$y2, $x1=$x2 tak získame vertikálny gradient, mení sa v smere osi y. Pre získanie uhlového gradientu treba zadať tieto parametre nasledovne $y1≠$y2, $x1≠$x2.

Vytvorený gradient priradíme do atribútu *fill* pomocou zadefinovaného identifikátora nášho gradientu v tvare *url(#identifikátor)*, napríklad pre našu funkciu *zasobnik01* a identifikátor gradientu pomenovaný ako *mojgradient1* bude použitie vo funkcii pre zásobník nasledovné:

zasobnik01(50,70,1,5,60,80,'blue','url(#mojgradient1)',1,'q01',1,'q02',1,'F',1,0,0);

prípadne všeobecne pre iný prípad je priradenie do atribútu fill="url(#mojgradient1)". Vytvorenie funkcií zjednodušilo vytváranie schém a výrazne zredukovalo kód. Schémy pre iné procesy vznikajú ako použitie viacerých funkcií, ktoré tvoria časť zariadenia. Napríklad funkciou pre zásobník, gradient a miešadlo získame schému pre proces miešania dvoch prúdov. Výsledok je na obr. 15.



Obr. 15 Schéma pre proces miešania získaná použitím kombináciou funkcií

4.1.2 Tvorba formulárov

Postup tvorby formulárov si vysvetlíme na najzložitejšom prípade a to na formulári pre sériovo zapojené zásobníky kvapaliny. Celý formulár ponúka viacero možností výberu, ktoré sú v niektorých prípadoch prepojené. Formulár obsahuje viacero vstupných prvkov, ktorých počet je premenlivý. Celkovo ide o tieto možnosti:

- voľba počtu zásobníkov
- voľba interakcií medzi jednotlivými zásobníkmi
- voľba jednosmerného toku, pre prípad ak máme interakciu medzi zásobníkmi
- voľba vyplnenia rovnakých parametrov pre všetky zásobníky
- voľba zobrazovania SVG schémy
- voľba vyplnenia parametrov zásobníka rovnakými parametrami ako má vybraný zásobník (len pre prípad voľby zásobníky nemajú rovnaké

parametre)

HTML dokument pre sériovo zapojené zásobníky kvapaliny je generovaný pomocou súboru *zasobniky_x_form.php*. V tomto súbore na začiatku zavoláme,uvedieme pomocou príkazu *include_onc*e súbor *index_head.php*. Pod výrazom generovanie budeme chápať výpis HTML značiek, prípadne JavaScriptu pomocou PHP príkazu *echo* s prípadným použitým podmienok *if*, *else* alebo cyklov *while* a *for*. Výrazom volanie súboru budeme chápať použitie príkazu *include_onc*e. Volaním funkcie budeme rozumieť zápis jej názvu aj s vyplnenými parametrami v oblej zátvorke v istej časti kódu. Súbor *index_head.php* predstavuje hlavný dokument stránky obsahujúci menu s výberom procesov ako aj hlavičku a všetky ostatné náležitosti stránky. Uvedenie súboru *index_head.php* je dôležité aj kvôli tomu, že v rámci tohto dokumentu sa volajú aj ďalšie funkcie a súbory, ktoré sú pre nás potrebné. Samotné generovanie formulára realizujeme funkciou *zasobniky_x*, tá je ohraničená HTML značkou *form* pre deklarovanie formulára. V značke *form* sme si zadefinovali do atribútu *action* odkaz na súbor *zasobniky_x_sim.php*, ktorý odoslané údaje z formulára spracuje. Formuláru sme pridelili do atribútu *id* identifikátor a určili v atribúte *method* metódu *post,* ktorou sa údaje odošlú.

Funkcia *zasobniky_x* obsahuje na začiatku kontrolu či pozná údaj z formulára o počte zásobníkov a interakcii medzi prvými dvoma zásobníkmi. V prípade ak tento údaj nepozná, tak počet zásobníkov je dva bez interakcie. Tieto údaje sa použijú na vygenerovanie formulára pri prvom zobrazení stránky. Nasleduje generovanie prvku výberové menu, ktoré slúži na výber počtu zásobníkov. Za týmto prvkom nasleduje formulárový prvok radio, slúžiaci na výber medzi možnosťou, kedy majú zásobníky rôzne parametre alebo ich majú rovnaké. Pod prvkom radio je uvedená SVG schéma *a* funkcia *zasobniky_rovnake.* Časť stránky, kde je SVG schéma a funkcia

zasobniky_rovnake je ohraničená značkou *div* s identifikátorom *id_vyber*. Nasledujú vstupné prvky typu text pre zadávanie času a kroku simulácie.

Pri udalosti *onchange* prvku pre výber počtu zásobníkov (na obr. 16 zvýraznený modrou šípkou) sa zavolá funkcia *js_vyber.* V tejto funkcii sa vytvorí asynchrónna komunikácia so serverom. Metódou *POST* sú odosielané aktuálne hodnoty údajov o počte zásobníkov, interakciách a type formulára (rovnaké alebo rôzne parametre zásobníkov) do súboru *js_vyber.php*, ktorý zabezpečuje obnovenie vymedzenej (identifikátorom *js_vyber*) časti stránky. Súbor *js_vyber.php* spracuje prijaté údaje a zobrazí nový obsah v danej časti dokumentu. Pri zmene ľubovoľného prvku, ktoré sú na obr. 16 v detaile zobrazené sa zavolá funkcia *js_vyber,* ktorá vykoná vyššie opísané zmeny.

Súbor *js_vyber.php* obsahuje volanie súboru SVG schémy a podmienku, podľa ktorej volá buď funkciu *zasobniky_rovnake* alebo *zasobniky_rozne.* Obe tieto funkcie na začiatku volajú funkciu *interakcie,* ktorá slúži na generovanie formulára pre voľbu interakcií medzi zásobníkmi. Nasleduje cyklus, v ktorom sa zrátajú hodnoty jednotlivých interakcií medzi zásobníkmi. Hodnota nula pre zásobníky, medzi ktorými interakcia nie je, opačne hodnota jedna. Ak je suma interakcií rovná nule, potom nie je ani v jednom prípade v celej sérií zvolená interakcia. V tom prípade sa na stránke nezobrazí formulárový prvok radio pre výber jednosmerného alebo obojsmerného toku (kedy tento jav môže nastať sme opísali v kapitole 2.2.6). V zvyšnej časti sa generujú ostatné časti formulára, najmä textové polia. Napríklad pre textové pole pre plochu prierezu *i-tého* zásobníka (voľba parametre zásobníkov rôzne) :

```
echo "<input type='text' name='F".($i+1)."' id='F".($i+1)."' value='' size='10' class='bgr' /><br />\n";
```

Funkcia *zasobniky_rovnake* predstavuje prípad formulára, kedy majú všetky zásobníky rovnaké parametre. Teda výsledný formulár obsahuje len všeobecné vstupné prvky pre zásobník *i*. Zadaním hodnoty do jedného textového poľa. Napríklad pre prierez, sa zadefinuje prierez pre všetky zásobníky v sérií. Podobne to je aj pre ostatné vstupné parametre. Interakcie ako aj skokové zmeny na vstupe do zásobníkov v čase sa definujú zvlášť pre každý zásobník.

Funkcia *zasobniky_rozne* na rozdiel od funkcie *zasobniky_rovnake* generuje pre každý zásobník vlastné textové polia. Zároveň im priradí správne indexy v popise ako aj v identifikátoroch podľa poradia v akom sú. Celé sa to realizuje pomocou cyklu *while*, ktorý trvá až kým PHP premenná *$i* nedosiahne hodnotu počtu zásobníkov. V každom kroku cyklu sa nám vygeneruje časť formulára odpovedajúca pre jeden zásobník. Premenná *$i* nám slúži na zmenu indexu číslovania ako aj číslovania identifikátorov.

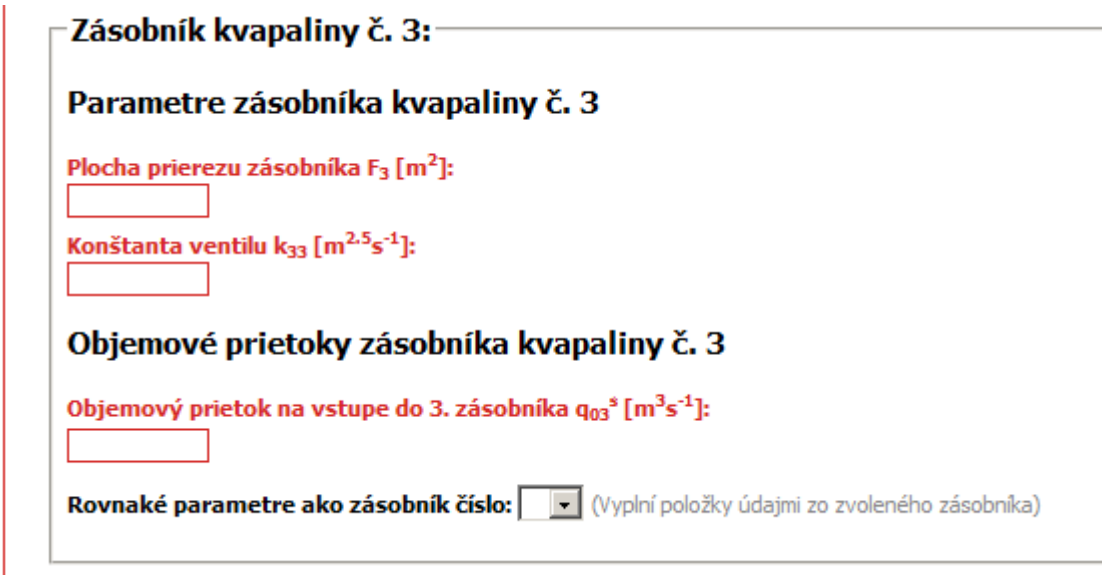
Funkcia *zasobniky_rozne* nám navyše pri každej časti formulára pre konkrétny zásobník, vygeneruje výberové menu slúžiace na vyplnenie položiek zásobníka hodnotami vybraného zásobníka z daného výberového menu. Tento vygenerovaný formulár pri zmene svojej hodnoty volá funkciu *presun*. Funkcia *presun* je napísaná v jazyku JavaScript a jej zápis je nasledovný:

```
function presun(j){
v=document.getElementById('zasobnik' +j).value  ;
document.getElementById('F' +j).value = document.getElementById('F' +v).value;
document.getElementById('k' +j+j).value = document.getElementById('k'
+v+v).value;
document.getElementById('q0' +j).value = document.getElementById('q0'
+v).value;  }
```

Jej vstupným parametrom je číslo (identifikátor) zásobníka, pre ktorý ideme

parametre priradiť. V rámci funkcie do premennej *v* priradíme vybranú hodnotu reprezentujúcu zásobník, ktorého údaje budeme priraďovať do zásobníka, ktorému údaje priraďujeme. Zvyšná časť funkcie obsahuje priradenie na základe identifikátorov zásobníkov. Na obr. 17 je detail výsledného formulára pre tretí zásobník, v dolnej časti výberové menu pre priradenie hodnôt iného zásobníka aktuálnemu.

Obr. 16 Detail formulára pre sériovo zapojené zásobníky kvapaliny.



Obr. 17 Detail časti formulára pre tretí zásobník v sérii .

Generovanie formulárov sa vo veľkej miere realizovalo pomocou cyklov a to najmä pri sériovo zapojených procesoch. Riešenie nie je univerzálne, procesy sa líšia. Hlavný rozdiel predstavuje použitie metódy AJAX tam, kde to je potrebné. Táto metóda sa použila najmä pri sériovo zapojených procesoch, ale aj pri reaktore.

Asynchrónne obnovenie časti stránky sa realizovalo principiálne rovnakým spôsobom ako pre zásobník aj pre ostatné procesy.

Pri formulároch sa okrem metódy AJAX použila aj knižnica jQuery. Pre jej použitie sme si ju museli v súbore *index_head.php* zadefinovať a to odkazom na jQuery knižnicu do značky *script* do atribútu *src,* kde sme priradili túto adresu :

//ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js

Knižnica jQuery sa použila v prvom prípade na zmazanie vstupných textových polí. Zápis pre túto akciu vyzerá nasledovne:

```
function formular_reset() { $(document).ready(function(){ $(':text').val(''); }); }
```

Funkcia sa vykoná až keď bude celý dokument plne načítaní. Potom vyhľadá všetky textové vstupné polia a ich hodnotu nastaví na prázdnu.

Druhá vytvorená funkcia slúži na prepínanie zobrazovania prvku označeného identifikátorom pomocou tlačidla. Samotnú jQuery funkciu máme uloženú v rámci PHP funkcie *schovaj*. Táto funkcia má tieto vstupné parametre:

- *$id* – slúži pre zadanie identifikátora, ktorý určuje na ktorú časť stránky bude funkcia aplikovaná
- *$popis* – text, ktorý bude mať tlačidlo slúžiace na prepínanie zobrazovania
- *$spomalenie* – čas trvania prechodového efektu pri zobrazovaní (skrývaní), udáva sa buď v milisekundách alebo slovne *slow* pre pomalý prechod a *fast*

pre rýchly prechod

- *$skryte – pri hodnote nula ostane výber nezobrazený, pri hodnote jedna sa zobrazí.*

Napríklad jQuery, časť ktorá pri prvom zobrazení stránky nezobrazí výber s identifikátorom *ID1*:

```
$(document).ready(function(){    $("#ID1").hide();});
```

a časť, ktorá prepína zobrazovanie pri kliknutí myšou na tlačidlo s triedou prepínač s prechodovým efektom tisíc milisekúnd.

```
$(document).ready(function(){  $(":button.prepinac").click(function(){    $
("#schema1").toggle(1000);  });});
```

4.1.3 Spracovanie vstupných údajov – simulácia

Po odoslaní údajov z formulára nasleduje spracovanie údajov v súbore, ktorý realizuje simuláciu. Postup si opíšeme pre sériovo zapojené zásobníky kvapaliny. Celý skript možno rozdeliť na tri časti:

- Vstupná
- Simulačná
- Výstupná

Prvá fáza realizácie simulácie je spracovanie údajov z formulára. To zahŕňa najmä ošetrenie údajov a ich priradenie do premenných. Premenné možno rozdeliť do dvoch skupín. Prvá sú premenné, ktorých počet je vopred známy. Takáto premenná je napr. koncový čas simulácie *$t_end*. Druhá skupinu tvoria premenné ktorých počet vopred nie je známy. V rámci spracovania údajov z formulára sa

vykonávajú dve úpravy. Prvou je odstránenie bielych (prázdnych) znakov pomocou PHP funkcie *trim.* Druhou úpravou je nahradenie symbolu čiarka za symbol bodka pomocou PHP funkcie *str_replace.* Týmto spôsobom ošetrujeme rôzne spôsoby zadávania desatinného čísla. Priradenie do premennej *$t_end* z formulára odoslaného metódou *POST* by aj s úpravami vyzeralo nasledovne:

```
$t_end= str_replace(',','.',trim($_POST["t_end"]));
```

V úvodzovkách je zadefinovaný identifikátor, pomocou ktorého sme určili, z ktorého formulárového prvku sme údaj vyberali. Úpravy na odstránenie bielych znakov, ako aj náhrada znaku čiarka za znak bodka, sa vykonáva len pre premenné, ktoré pochádzajú z formulárového prvku typu *text*.

U premenných, ktorých počet je vopred neznámy, sa realizuje priraďovanie pomocou *cyklov* for alebo *while.* Hodnoty sa priraďujú do polí. O dĺžke trvania cyklu rozhoduje obvykle počet zariadení prípadne reakcií či zložiek. V prípade sériovo zapojených zásobníkov rozhoduje počet zásobníkov v sérií. Napríklad cyklus, ktorý priraďuje hodnoty do poľa pre prierez zásobníkov, vyzerá nasledovne:

```
for($i=0;$i<$PZ;$i++)
{
 $F[$i]  = str_replace(',','.',trim($_POST['F'.($i+1)]));
}
```

Premenná *$PZ* je počet zásobníkov. Podobným spôsobom sa priraďujú z formulára aj ostatné premenné. V rámci jedného cyklu môže byť priraďovanie viacerých údajov do pola. Pre niektoré parametre jednorozmerné pole nestačí. Napríklad pre hodnoty skokových zmien vstupov do procesu. Počet zásobníkov v sérií určuje koľko vstupov do zariadenia máme a zároveň pre tieto vstupy môžeme realizovať ľubovoľný počet skokových zmien. Priradenie skokových zmien do viacrozmerného poľa sa realizuje

v cykle. Vonkajší cyklus určuje, pre ktorý vstup sa skok realizuje a vnútorný cyklus trvá podľa počtu skokových zmien a v rámci neho sa priraďujú konkrétne hodnoty skokových zmien do tohto poľa. Podobne to je aj pre prietokový chemický reaktor s viacerými reakciami a zložkami. Do dvojrozmerného poľa sa ukladá údaj o stereometrickom koeficiente zložky pre reakcie, ktoré v reaktore prebiehajú. Toto priradenie sa realizuje v nasledovnom cykle:

```
for($i=1;$i<=$zloziek;$i++) {
        for($j=1;$j<=$reakcii;$j++)
        {
                $sk[$i-1][$j-1] = str_replace(',','.',trim($_POST['sk'.$i.$j]));
        }
}
```

Výsledné pole má toľko riadkov, koľko je zložiek v reaktore a každý riadok má toľko stĺpcov, koľko reakcií beží v reaktore. V riadku sú vždy zapísané hodnoty stereometrických koeficientov zložky v jednotlivých reakciách.

Po spracovaní všetkých údajov z formulára nasleduje výpočet ustálených stavov. Realizuje sa na základe vzťahov pre výpočet ustálených stavov, ktoré sú uvedené v kapitole 2.2 pre jednotlivé procesy. Procesy, ktoré sú sériovo zapojené majú výpočet ustálených stavov realizovaný pomocou cyklu a výsledné hodnoty priraďujú do poľa. Výnimku tvorí prietokový chemický reaktor a zásobníky kvapaliny pri rôznej konfigurácií interakcií medzi zásobníkmi. Pre tieto prípady sa ustálené stavy získavajú priamo simuláciou. Simulácia trvá kým sa stavy neustália alebo simulácia skončí neúspešne, ak sa hodnoty príliš vzdialia od reálnej hodnoty.

Pri sériovo zapojených zásobníkoch zohráva veľkú rolu konfigurácia interakcií medzi zásobníkmi. V prípade, ak máme zapojenie bez interakcie vo všetkých

prípadoch, tak sa výpočet ustálených stavov realizuje pomocou funkcie *USBI,* ktorá používa na výpočet ustálených stavov vzťahy (37). Naopak pri konfigurácií ak sú všetky zásobníky s interakciou sa používa funkcia *USSI,* ktorá používa na výpočet ustálených stavov vzťahy (38), (39). Pre tretiu možnosť, kedy sa v zapojení vyskytujú aj zásobníky medzi ktorými interakcia nie je a aj zásobníky, ktoré medzi sebou interakciu majú sa používa pre výpočet ustálených stavov priamo simulácia. Simuláciu pre výpočet US reprezentuje cyklus *while.* Cyklus trvá až kým absolútna hodnota rozdielu medzi hodnotou hladiny získanou v kroku *k* a hodnotou hladiny získanou v kroku *k-1,* nie je menšia ako požadovaná presnosť. Sledujeme dva absolútne rozdiely. Hodnotu hladiny pre posledný a predposledný zásobník v konfigurácií.

Na riešenie DR sme použili funkciu *rk4*. Jej zápis je nasledovný:

```
function rk4($x,$u,$eqn){
        $xx = $x[$eqn];
        $f1 = model($x,$u,$eqn);
        $x[$eqn] = $xx + h/2*$f1;
        $f2 = model($x,$u,$eqn);
        $x[$eqn] = $xx + h/2*$f2;
        $f3 = model($x,$u,$eqn);
        $x[$eqn] = $xx + h*$f3;
        $f4 = model($x,$u,$eqn);
        $sys = $xx + h/6*($f1+2*$f2+2*$f3+$f4);
                if ($sys<0){$sys = 0;}
        return $sys;
}
```

Jej vstupné parametre sú:

- *$x* – pole stavov
- *$u* – pole vstupov, pri simulačnom výpočte US sa berú len ustálené hodnoty, skokové zmeny sa nerealizujú.
- *$eqn* – číslo DR použitej pre výpočet

V rámci funkcie sa na začiatku do premennej *$xx* priradí hodnota stavu v predchádzajúcom kroku, odpovedajúcu x_n v rovnici (45). Ďalšie časti funkcie realizujú zápis rovníc (41) až (45). Výsledný stav predstavuje hodnota v premennej *$sys.* V rámci funkcie je zároveň ošetrenie možnosti zápornej hodnoty výsledku. Vtedy sa výsledok bude rovnať nule, nakoľko záporná výška hladiny sa neuvažuje. Funkcia *rk4* sa pri výpočte US v rámci hlavného cyklu nachádza vo vnútornom cykle *while.* Vnútorný cyklus zabezpečuje výpočet DR pre potrebný počet stavov. Hlavný – vonkajší cyklus zabezpečuje krok simulácie a sleduje či sú hodnoty už ustálené.

Funkcia *rk4* zabezpečuje riešenie DR metódou *Runge-Kutta*, aby bola čo najviac univerzálna nie je vytvorená pre konkrétny model. Preto sa v rámci tejto funkcie vola funkcia *model*. Funkcia *model* používa rovnaké vstupné parametre ako funkcia *rk4.* Musí navyše poznať aj konštanty vystupujúce v DR, prípadne iné špecifikácie modelu. V prípade sériovo zapojených zásobníkov sú navyše (pomocou funkcie *global*) zadefinované tieto údaje:

- *$interakcia* – pole obsahujúce údaje o interakciách medzi zásobníkmi
- *$k* – pole obsahujúce konštanty ventilov
- *$F* – pole obsahujúce prierezy zásobníkov

- *$jednosmernytok* – premenná určujúca smer toku pri špecifickom prípade záporného rozdielu výšok hladín v zásobníkoch (opísané v kapitole 2.2.6)

Vo funkcii *model* sa DR opisujúca zásobník generuje na základe údajov, ktoré boli zvolené užívateľom. Vo funkcii sa pomocou série podmienok *if*, *else* podľa pravidiel, ktoré sú na obr. 9 a obr. 10. získa model pre konkrétny zásobník. Premenná *$eqn* určí, o ktorý zásobník v sérií sa jedná a poslúži n a výber údajov z potrebných polí. Napríklad zápis modelu pre druhý až *n*-tý zásobník vyzerá v PHP nasledovne

```
$sys=(1/$F[$eqn])*($u[$eqn]+(($signIN)*($k[$eqn-1]*sqrt($IN)))-
(($signOUT)*($k[$eqn]*sqrt($OUT))));
```

pre prípad, ak nie je v interakcii s predchádzajúcim zásobníkom, výraz *$IN* a *$signIN* nadobudne tvar

```
$IN = $x[$eqn-1];
$signIN = 1;
```

a ak je hladina v nasledujúcom zásobníku vyššia ako v aktuálnom a uvažujeme obojsmerný tok, tak výraz *$OUT* a *$signOUT* nadobudne tvar

```
$OUT = abs($x[$eqn]-$x[$eqn+1]);
$signOUT = -1;
```

Podobne sa DR rôzne menia podľa pravidiel opísaných v kapitole 2.2.6 a užívateľom zadaných parametrov. Pre iné procesy je výber DR popisujúcej proces realizovaný jednoduchým výberom z možností konkrétnych rovníc zapísaných v kóde. Výnimkou je reaktor s viacerými zložkami a reakciami, pri ktorom sa vyčísľuje suma v rovniciach (27) a (30) pomocou cyklu.

Po získaní ustálených stavov nasleduje simulácia. Simulácia sa realizuje rovnakým postupom ako výpočet US simuláciou pre sériovo zapojené zásobníky s rôznymi interakciami. Rozdiel spočíva v meniacich sa vstupoch a v trvaní hlavného cyklu.

Trvanie hlavného cyklu je dané koncovým časom simulácie zadaným užívateľom a krokom simulácie rovnako zadaným užívateľom. Počas simulácie sa sleduje premenná *$t,* ktorá sa v každom kroku simulácie zväčší o krok simulácie. Simulácia trvá až kým *$t* nedosiahne hodnotu koncového času zníženú o krok simulácie. Počas simulácie sa v každom kroku simulácie do polí ukladali hodnoty stavov a vstupov.

Posledná časť v rámci spracovania údajov v simulácií, je ich ukladanie do databázy. Do databázy sa ukladajú ako vstupné údaje tak aj údaje získané simuláciou. V prípade polí sa realizuje úprava poľa do reťazca spojeného znakom rúra |. Táto úprava sa realizuje pomocou PHP funkcie *implode.* Táto funkcia má ako prvý parameter znak, ktorým spája prvky poľa. Druhým parametrom je samotné pole. Po úprave polí sa realizuje zápis do databázy. Napríklad pre sériovo zapojené zásobníky kvapaliny je zápis pre vkladanie údajov nasledovný:

```
$sql = "INSERT INTO zasobnikyn VALUES(null,'".$_SESSION['id']."','$PZ',...);";
mysql_query($sql);
```

Údaje sa uložia do tabuľky s názvom *zasobnikyn.* V oblých zátvorkách sa nachádzajú údaje (príklad neobsahuje všetky, kvôli veľkému množstvu vkladaných premenných), ktoré majú byť vložené do tabuľky v databáze. Po vložení údajov do databázy sa do premennej *$id* priradí identifikátor, ktorý má hodnotu identifikátora, ktorý sa nachádza v tabuľke pri práve vloženom zázname. Tento identifikátor sme získali pomocou funkcie *mysql_insert_id*. Identifikátor sa odošle pomocou metódy *GET* cez presmerovanie na stránku zobrazujúcu výstupy zo simulácie.

4.1.4 Výstupy zo simulácie – grafy a export

Po spracovaní údajov v simulácii sa užívateľovi zobrazí stránka obsahujúca

grafy, v našom príklade zásobníkov ju generuje súbor *zasobniky_x_graf.php*. Stránka na začiatku nahrá všetky údaje z databázy do premennej *$row* pre danú simuláciu na základe identifikátora, ktorý sa poslal metódou *GET* zo súbora, ktorý realizoval simuláciu. Zmenou oproti pôvodnej verzii Modeltool je nová metóda pre vykresľovanie grafov s použitím jQuery knižnice Flot.

Pri použití knižnice Flot sme vopred vedeli, ktoré súbory knižnice budeme potrebovať. Potrebné súbory sme v súbore *index_head.php* zadefinovali pomocou značky *script* s uvedením cesty k jednotlivým súborom knižnice v atribúte *src*. Použili sme tieto súbory:

- excanvas.min.js
- jquery.js
- jquery.flot.js
- jquery.flot.navigate.js

Flot pre svoje použitie zároveň potrebuje knižnicu jQuery, spôsob jej pridania na stránku sme si vysvetlili v kapitole 4.1.2. Pre pridanie grafu na stránku sme si vytvorili dve funkcie. Prvá funkcia *spracuj_udaje* nám spracuje údaje, ktoré chceme zobraziť do takej podoby s akou dokáže Flot pracovať. Druhá funkcia *graf_flot* slúži už na samotné vykreslenie grafov so vstupnými údajmi, ktoré tejto funkcii zadefinujeme, ktoré slúžia na nastavenie vlastností grafu. Funkcia *spracuj_udaje* má dva vstupné parametre. Prvý parameter je reťazec obsahujúci názvy stĺpcov v tabuľke. Názvy stĺpcov sú v reťazci spojené znakom rúra. Pomocou PHP funkcie *explode*, sa tieto znaky odstránia a získame pole priradené do premennej *$nazvyStlpcov. Toto pole* obsahuje názvy stĺpcov tabuľky. Na začiatku funkcie

spracuj_udaje zadefinujeme pomocou príkazu *global* premennú *$row* ktorá obsahuje údaje nahrané z databázy. Hlavnú úlohu funkcie *spracuj_udaje* vykonávajú dva cyklusy. Vonkajší trvá na základe počtu údajov v premennej *$nazvyStlpcov*. V rámci neho sa vyberajú údaje z konkrétneho stĺpca tabuľky, zároveň pomocou PHP funkcií *min* a *max* nájdeme z údajov maximálnu a minimálnu hodnotu. V rámci vonkajšieho cyklu beží vnútorný cyklus. Vnútorný cyklus trvá podľa počtu údajov, ktoré sú vo vybranom poli. V rámci tohto cyklu sa vytvorí jeden reťazec, ktorý má všeobecný tvar: *[x0,y0], [x1,y1], ..., [xn,yn]*, tento reťazec sa priradí do premennej. Získaná premenná sa v rámci vonkajšieho cyklu priraďuje do poľa. Na konci vonkajšieho cyklu sa zrealizuje výpočet pre maximálnu a minimálnu hodnotu údaja, ktorý predstavuje y súradnicu. Tieto dva údaje sa použijú ako maximálna a minimálna hodnota y súradnice v grafe. Druhý vstupný parameter funkcie *spracuj_udaje* slúži na prepočet údajov na stupne Celzia. To sa použije v prípade, ak sú vstupné údaje v Kelvinoch. Pri hodnote tohto parametra jedna sa údaje prevedú na stupne Celzia. Návratová hodnota funkcie *pracuj_udaje* je pole obsahujúce koncový čas simulácie, maximálnu hodnotu y, minimálnu hodnotu y a pole, ktoré obsahuje reťazce dát.

Funkcia *graf_flot,* ktorá slúži na tvorbu grafu, má pomerne veľký počet vstupných parametrov. Sú to tieto parametre:

- *$pocetGrafov* – slúži na určenie počtu vykreslených závislostí v grafe. Tento parameter určuje trvanie cyklov ktoré generujú premenné pre graf. Zápis pre tvorbu týchto premenných v cykle vyzerá nasledovne:

  ```
  for($i=0;$i<$pocetGrafov;$i++)
  {
  echo  'var d'.$i.' = {label: "'.$legend[$i].'",data:['.$data[$i].']};';
  ```

```
}
```

- *$legend* – pole obsahujúce popis pre jednotlivé závislosti
- *$data* – pole, ktoré obsahuje údaje pre vykreslenie, pole sa získava z funkcie *spracuj_udaje* ako návratová hodnota
- *$idplot* – identifikátor grafu
- *$xlim* – maximálna hodnota pre os x, hodnota sa získava z funkcie *spracuj_udaje*
- *$ymin* – minimálna hodnota pre os y, hodnota sa získava z funkcie *spracuj_udaje*
- *$ymax* – maximálna hodnota pre os y, hodnota sa získava z funkcie *spracuj_udaje*
- *$points* – pri hodnote *true* zobrazí body grafu funkcie, naopak *false*
- *$steps* – pri hodnote *true* bude graf schodíkového typu, naopak *false* pre hladký graf
- *$width1* – šírka grafu
- *$height1* – výška grafu
- *$idhoover* – identifikátor pre element v ktorom sa zobrazujú súradnice x a y
- *$popis* – text zobrazený nad grafom (popis)
- *$popisY* – popis osi y

Funkciu *graf_flot* možno rozdeliť na štyri časti. Prvá je časť priradenia dát, kde sa v cykle generujú premenné. Tie obsahujú legendu a údaje pre vykreslenie. Ďalší cyklus generuje premennú *data* obsahujúcu premenné reprezentujúce jednotlivé závislosti. Zápis tohto cyklu aj s priradením do premennej je nasledovný:

```
echo 'var data = [';
for($i=0;$i<$pocetGrafov;$i++)
{
if($i!=$pocetGrafov-1){echo 'd'.$i.',';}
else {echo 'd'.$i;  }
}
echo '];';
```

Ďalšou časťou funkcie je definovanie vlastnosti grafu do premennej *options*. V premennej *options* je veľké množstvo nastavení. Ide hlavne o nastavenia štýlu jednotlivých častí grafu ako aj o nastavenia súvisiace s interaktívnou stránkou grafu. Flot nevyžaduje zadefinovanie všetkých vlastností, tie ktoré nie sú zadefinované budú mať štandardné hodnoty, ktoré im priradí Flot. Pre nás boli dôležité tieto nastavenia:

- *zoom* – pri hodnote *true* povoľuje približovanie
- *pan* – pri hodnote *true* povoľuje manipulovať - pohybovať s grafom v rámci špecifikovanej oblasti
- *hoverable* – v nastaveniach pre mriežku (*grid*) pri hodnote *true* s použitím funkcie umožní zobrazovanie súradníc aktuálnej pozície myši v rámci grafu.
- *zoomRange* – v rámci nastavenia pre os x a y (xaxis, yaxis), nastavenie

rozsahu priblíženia

- panRange – v rámci nastavenia pre os x a y (xaxis, yaxis), určuje hranice, v ktorých je možné manipulovať, pohybovať s grafom.

Poslednou časťou funkcie *graf_flot* je zápis html značiek. Nachádza sa tu zápis značky *div* s prideleným identifikátorom pre sekciu, v ktorej sa zobrazí graf. Rovnako značka *div* s identifikátorom pre časť, v ktorej sa zobrazujú súradnice, na ktorých sa nachádza myš. Funkcia je uložená v priečinku *flot* v súbore *functiom_plot_1.php*.

Vykreslenie grafu pre sériovo zapojené zásobníky sa s použitím funkcií *spracuj_daje* a *graf_flot* realizuje nasledovne opísaným spôsobom. Vytvorí sa pomocná premenná obsahujúca reťazec s názvami stĺpcov tabuľky. Názvy stĺpcov sa vyberajú podľa toho, aké údaje chceme vykresliť. Napríklad pre vykreslenie hladín štyroch zásobníkov vyzerá reťazec nasledovne *x1|x2|x3|x4*, tento reťazec sa v rámci stránky generuje v nasledovnom cykle:

```
$vyber ='';
for ($i=1;$i<=$row['pocetzas'];$i++){
if ($i!=$row['pocetzas']) {$vyber =  $vyber.'x'.$i."|";}
else{$vyber =  $vyber.'x'.$i;}
}
```

Pomocná premenná *$vyber* obsahuje náš reťazec. Reťazec v premennej *$vyber* použijeme ako vstupný argument do funkcie *spracuj_udaje.* Návratovú hodnotu z tejto funkcie priradíme do premennej týmto zápisom `$vysledok=spracuj_udaje($vyber,0);`. Druhý parameter je nula, lebo nerealizujeme prepočet na stupne Celzia, nakoľko údaje sa týkajú výšky hladiny a nie teploty. V cykle si vytvoríme text pre legendu. Spracované údaje následne použijeme ako

vstupy do funkcie *graf_flot.* Zápis pre sériovo zapojené zásobníky vyzerá potom nasledovne:

```
graf_flot($row['pocetzas'],$legend,$vysledok[3],'graf1',$vysledok[0],
$vysledok[1],$vysledok[2],'false','true','740','540','hoover1','Priebeh výšok
hladín v zásobníkoch počas simulácie','h(t)');
```

Kde jednotlivé vstupné parametre sú popísané pri opise funkcie *graf_flot*.

Ďalšou možnosťou v prípade výstupov zo simulácie je ich export. V rámci stránky s grafom sa zároveň zobrazujú odkazy pre export údajov do textového alebo xml súboru. Odkaz pomocou metódy *GET* odosiela do súborov pre export informáciu o čísle procesu, identifikátore pre simuláciu a názvy stĺpcov v tabuľke, ktoré majú byť exportované. Odosielaný údaj je v tvare reťazca. Potrebné údaje v tomto reťazci sú spájané znakom rúra. Napríklad pre sériovo zapojené zásobníky kvapaliny, pri počte zásobníkov päť a identifikátore 112 je výsledný odosielaný reťazec nasledovný: *3|112|x1|x2|x3|x4|x5|u0|u1|u2|u3|u4|u5*. Export sa realizuje pomocou súborov *global_txt.php* a *global_xml.php* v závislosti od výberu užívateľa či chce textový alebo xml súbor. Oba súbory obsahujú pole s názvami tabuliek v databáze pre jednotlivé procesy. Reťazec sa v rámci týchto súborov pomocou funkcie *explode* zmení na pole. Prvý prvok v poli určuje, o ktorý proces sa jedná, z ktorej tabuľky v databáze údaje vyberáme. Druhý prvok nám určuje identifikátor použitý pre výber riadku v tabuľke. Zvyšné prvky poľa určia, ktoré stĺpce v rámci riadka a tabuľky vyberáme. Získané údaje zapisujeme pre textový súbor pomocou dvoch cyklov. Prvý vonkajší cyklus zapisuje riadky, druhý vnútorný cyklus stĺpce. Export xml súboru sa realizuje tiež pomocou dvoch cyklov. Vonkajší cyklus je daný počtom premenných a vnútorný počtom údajov pre danú premennú. Samotnú štruktúru xml, tvoria značky. Štruktúra xml súboru je nasledovná:

```
<udaje>
    <premenna>
        <bod>
            <hodnota></hodnota>
            <cas></cas>
        </bod>
    </premenna>
</udaje>
```

Značka *udaje* predstavuje hlavný element. V rámci neho sú premenné v značke *premenna.* Každá premenná obsahuje svoje údaje v značke *bod,* ktorá obsahuje údaje o čase a hodnote v značkách *hodnota* a *cas*.

4.2 Verifikačná časť

Zaoberá sa porovnaním údajov získaných zo simulácie pomocou Modeltool s údajmi získanými simuláciou v programe MATLAB/Simulink. Pre simuláciu v Simulinku sme nastavili fixný krok simulácie rovnaký aký sme použili pri simulácií v Modeltool. V programe Simulink sme použili riešiteľ (solver) ode4 (Runge-Kutta). Verifikáciu si ukážeme pre dva vybrané procesy. Vybrali sme sériovo zapojené procesy, kde jeden získava ustálené stavy priamo z DR. Rovnakým spôsobom akým sa validácia robila pre vybrané procesy uvedené v práci, sa robila aj pre ostatné procesy v rámci Modeltool. Pre prietokový chemický reaktor s viacero reakciami a zložkami, predstavovali vhodné vstupné údaje ťažkú úlohu, a preto všetky možnosti neboli odskúšané.

4.2.1 Verifikácia výsledkov sériovo zapojených zásobníkov kvapaliny

Tento proces zahŕňa viacero možných nastavení, pre overenie sme vybrali

konfiguráciu troch zásobníkov, kde prvé dva sú v interakcii. Simulácia mala čas simulácie 100 sekúnd a krok simulácie 0.1 s. Uvažoval sa jednosmerný tok. Vstupné údaje sú uvedené nižšie.

Tabuľka 6 Vstupné parametre pre sériovo zapojené zásobníky

		Zásobník:		
Parameter	**Rozmer**	**č.1**	**č.1**	**č.1**
Prierez zásobníka F_i	m^2	2.4	2.4	2.4
Konštanta ventilu k_{ii}	$m^{2.5}s^{-1}$	1.4	1	1.4
Objemový prietok v ustálenom stave q_0^s	$m^3 s^{-1}$	0.1	*	*
Druhý vstupný objemový prietok v ustálenom stave q_{0i}^s	$m^3 s^{-1}$	1	1	1

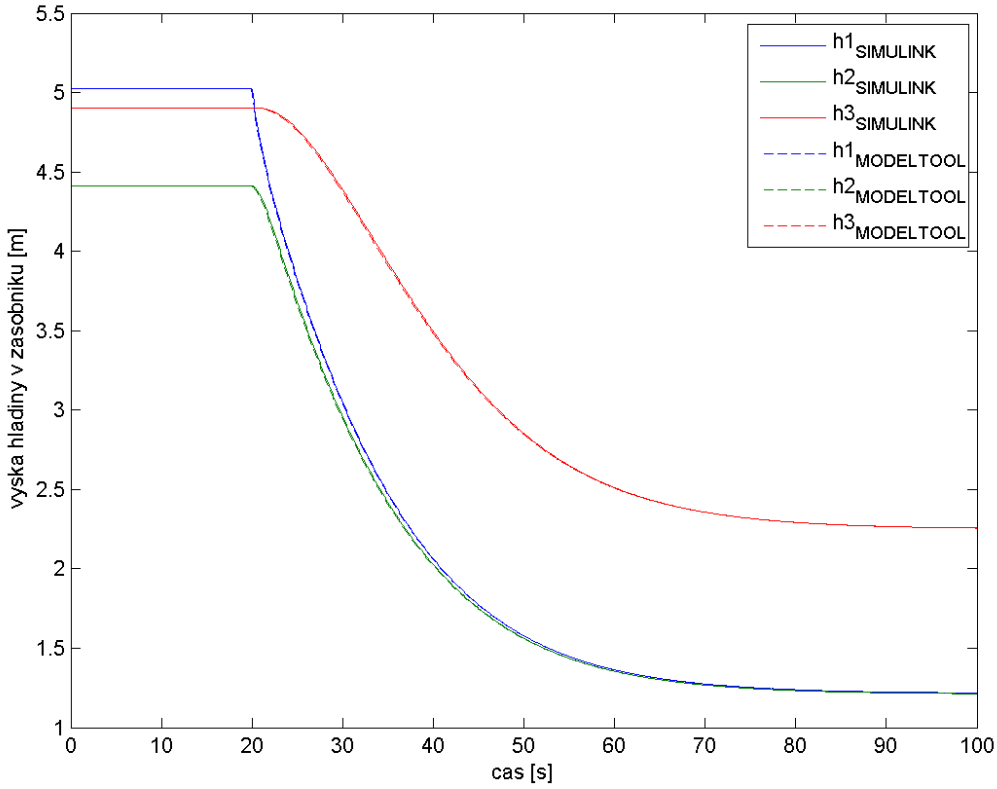
* vstupný prietok je výtok z predchádzajúceho zásobníka

V čase dvadsať sekúnd sa realizovala skoková zmena druhého objemového prietoku na vstupe do prvého zásobníka z hodnoty 1 m^3s^{-1} na hodnotu 0 m^3s^{-1}.

Tabuľka 7 Porovnanie ustálených stavov pre sériovo zapojené zásobníky

	h_1^s [m]	**h_2^s [m]**	**h_3^s [m]**
Modeltool	5.0271	4.4098	4.9028
MATLAB	5.0273	4.4100	4.9031
Rozdiel (US_{MATLAB}-$US_{Modeltool}$)	0.0002	0.0002	0.0003

Ustálené stavy sa v prípade Modeltool získavali priamo z výpočtu DR. Ustálené stavy získané v programe MATLAB sa získali priamo z algebraických vzťahov.
Sumy štvorcov odchýlok medzi hodnotami získanými simuláciou v programe Simulink a hodnotami získanými simuláciou v Modeltool sú:
$ss(h_1) = 0.035089519560167$
$ss(h_2) = 0.054353555897342$
$ss(h_3) = 0.051314925750679$
Porovnanie priebehov oboch simulácií možno vidieť na obr. 18



Obr. 18 Porovnanie priebehov výšok hladín

4.2.2 Verifikácia sériovo zapojených plášťových výmenníkov tepla

Zvolili sme konfiguráciu troch výmenníkov v sérií s časom simulácie 100 sekúnd a krokom simulácie 0.1 sekundy. Vstupné údaje sú uvedené nižšie.

Tabuľka 8 Vstupné parametre pre sériovo zapojené zásobníky

		Výmenník		
Parameter	**Rozmer**	**č.1**	**č.2**	**č.3**
Koeficient prestupu tepla α_i	W $m^{-2}K^{-1}$	72	72	72
Objem ohrievanej kvapaliny V_i	m^3	1	1	1
Plocha prestupu tepla A_i	m^2	1.5	1.5	1.5
Objemový prietok v ustálenom stave q_{0i}^s	m^3 s^{-1}	0.05	0.05	0.05
Teplota v plášti v ustálenom stave $\vartheta_{p,i}$	K	400	400	400
Teplota vstupného prúdu v ustálenom stave ϑ_{0i}	K	300	300	300
Hustota ohrievanej kvapaliny ρ	kg m^{-3}	900	900	900

Špecifická tepelná kapacita ohrievanej kvapaliny	$J\,kg^{-1}\,K^{-1}$	3.84	3.84	3.84

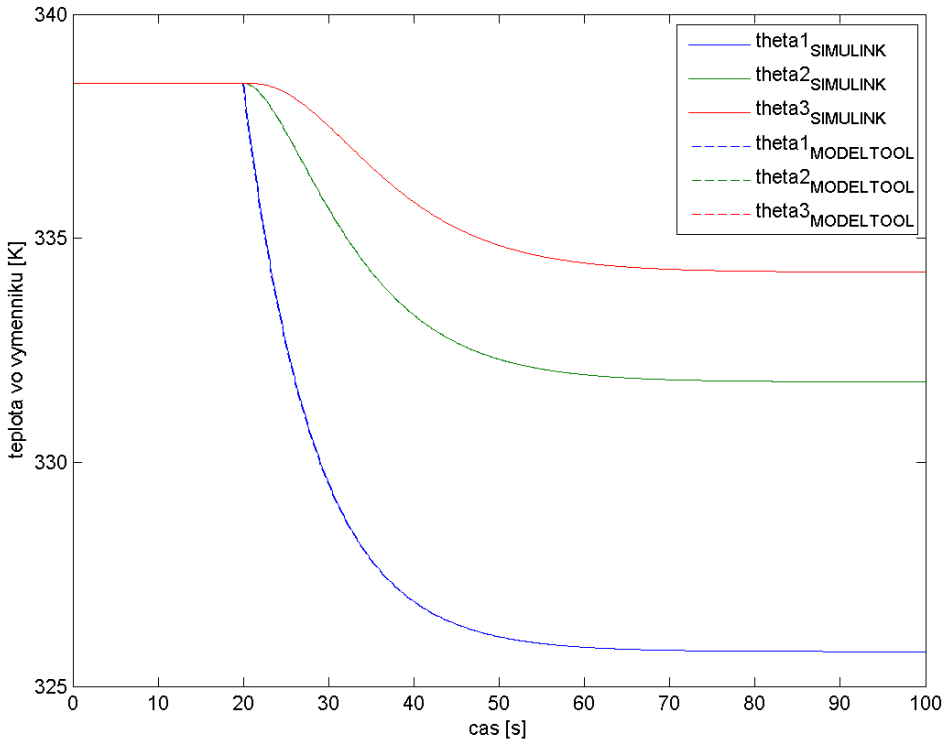
V čase 20 s sa realizovala skoková zmena objemového prietoku ohrievanej kvapaliny na vstupe do prvého výmenníka tepla z hodnoty 0.05 $m^3\,s^{-1}$ na hodnotu 0.09 $m^3\,s^{-1}$

Sumy štvorcov odchýlok medzi hodnotami získanými simuláciou v programe MATLAB Simulink a hodnotami získanými simuláciou v Modeltool sú:

ss(ϑ_1) = 0.97601

ss(ϑ_2) = 0.039077

ss(ϑ_3) = 1.1114e-006



Obr. 19 Porovnanie priebehov teplôt

4.3 Modeltool

4.3.1 Inštalácia

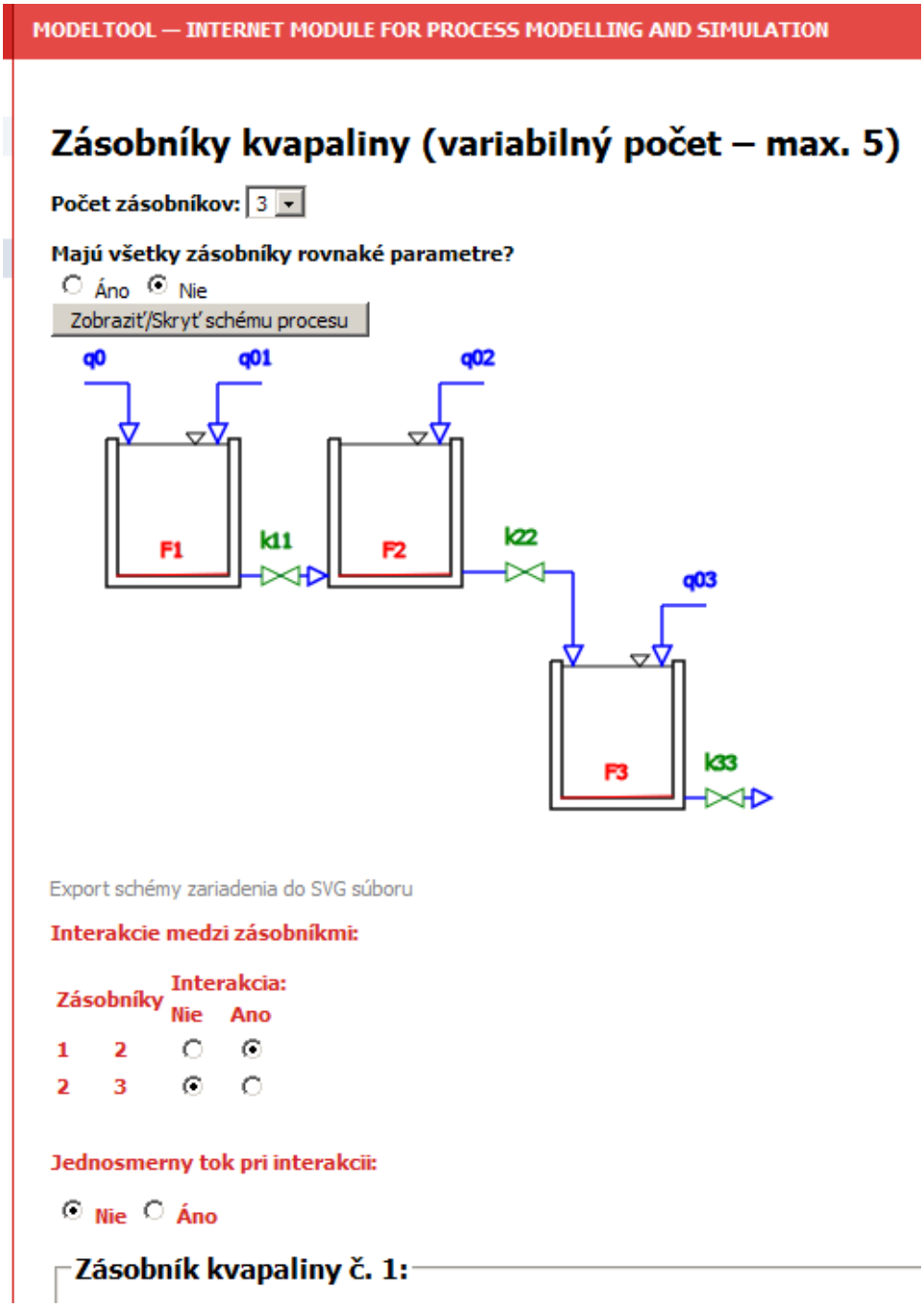
Inštalácia Modeltool predstavuje skopírovanie priečinku Modeltool na server.

V priečinku Modeltool sú všetky súbory stránky. Zároveň je nutné vytvoriť databázu a v rámci nej importovať jednotlivé tabuľky, ktoré Modeltool potrebuje. Každá tabuľka má vytvorený svoj súbor pre import. Prihlasovacie údaje do databázy je nutné zmeniť vo funkcii *connect* v súbore *connect.php*

4.3.2 Popis práce s vybraným modulom

Popis všetkých modulov by bol zdĺhavý. Preto si ukážeme popis na vybranom procese sériovo zapojených zásobníkov kvapaliny. K ostatným procesom si uvedieme isté dôležité špecifikácie.

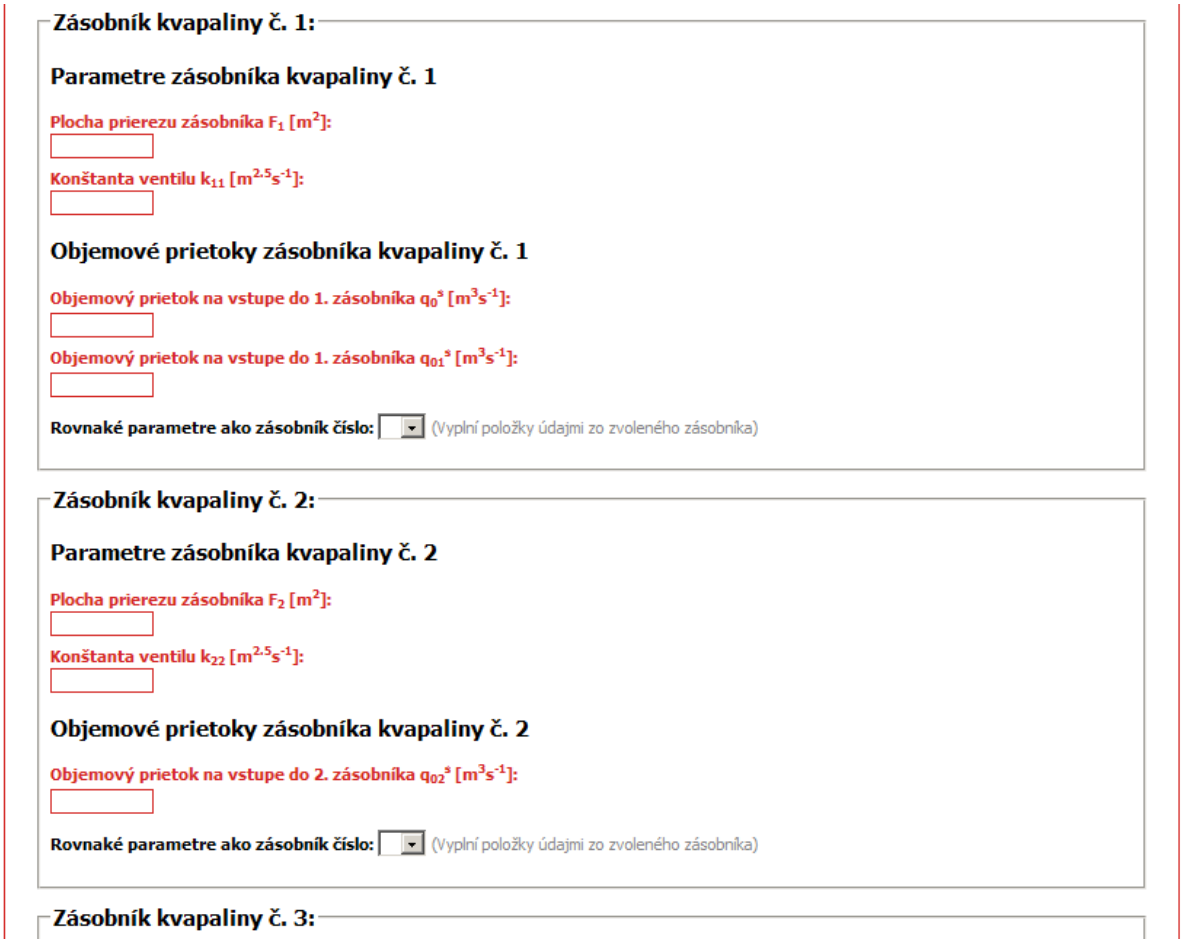
Výber procesu sa realizuje výberom v menu, ktoré obsahuje jednotlivé procesy. Po výbere procesu a zvolení novej simulácie sa užívateľovi zobrazí formulár pre zadávanie parametrov simulácie. Detail formulára pre tri zásobníky s rôznymi parametrami je na nasledujúcom obrázku.



Obr. 20 Detail z formulára pre výber parametrov pre simuláciu.

Vrchná časť formulára predstavuje úvod k procesu. Obsahuje nastavenia počtu zásobníkov, ich interakcií a podobne. Obsahuje aj schému zariadenia, ktorú je možné tlačidlom schovať prípadne zobraziť. Schému je možné aj exportovať kliknutím na odkaz pod schémou. Po výbere počtu zásobníkov sa voľby prejavia na stránke zmenou dokumentu. Podľa počtu zásobníkov sa nám zobrazí formulár pre

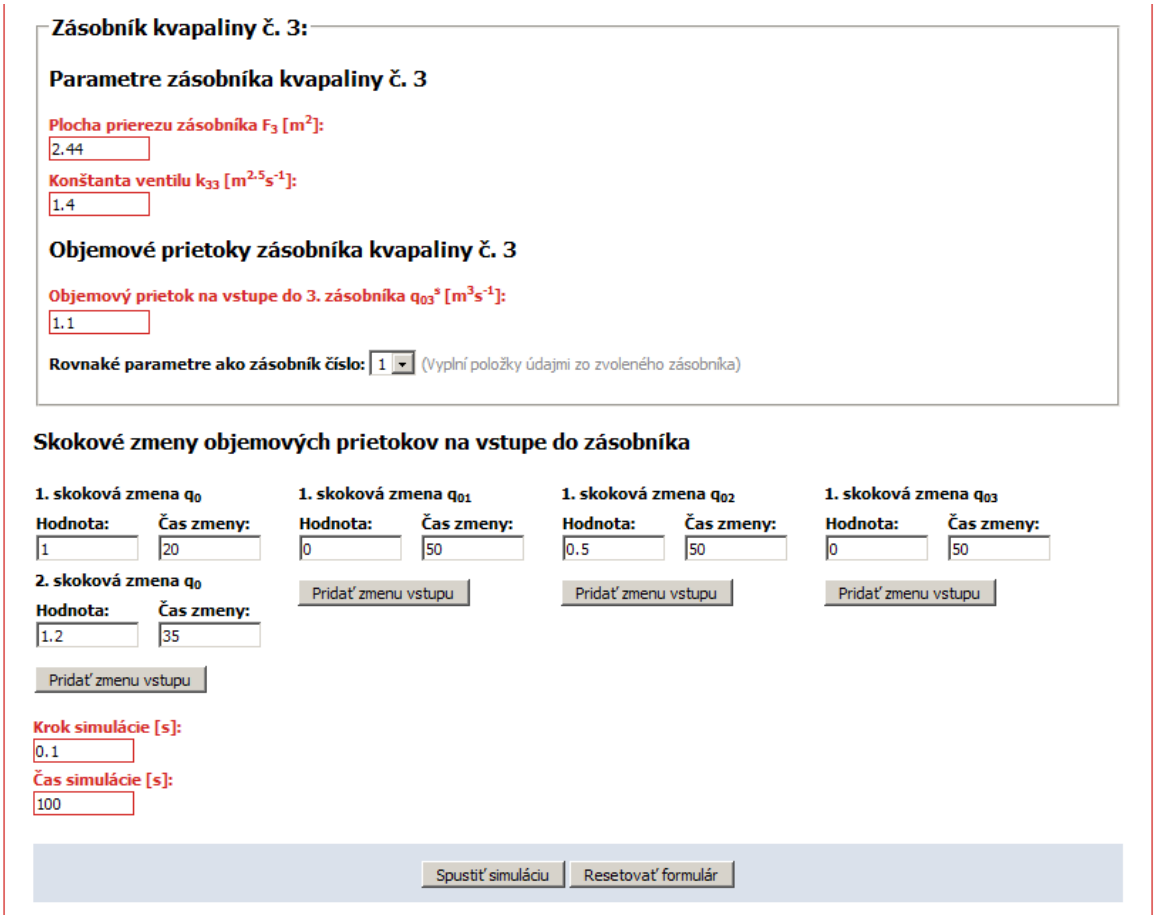
zadávanie parametrov pre jednotlivé zásobníky. V detaile na nasledujúcom obrázku pre prvé dva zásobníky v sérií.



Obr. 21 Detail z formulára pre zadávanie parametrov pre jednotlivé zásobníky.

Do jednotlivých polí zadáme parametre, prípadne využijeme možnosť priradenia hodnôt zásobníka, pre ktorý sme už hodnoty zadali. Na tento účel slúži výberové menu, v dolnej časti sekcie formulára pre každý zásobník. Po vyplnení údajov k jednotlivým zásobníkom, nasleduje časť formulára venovaná skokovým zmenám na vstupe do procesu. V tejto časti môžeme zadávať skokové zmeny pre príslušné vstupy. Môžeme zadať žiaden alebo ľubovoľný realizovateľný počet týchto skokových zmien. Po vyplnený všetkých údajov odošleme údaje k simulácií pomocou tlačidla spustiť simuláciu. Pre prípad ak chceme vyplniť textové polia formulára nanovo, je v dolnej časti formulára tlačidlo *Resetovať formulár*. Jeho

stlačením sa údaje vymažú. Táto časť formulára v detaile na obr. 22.

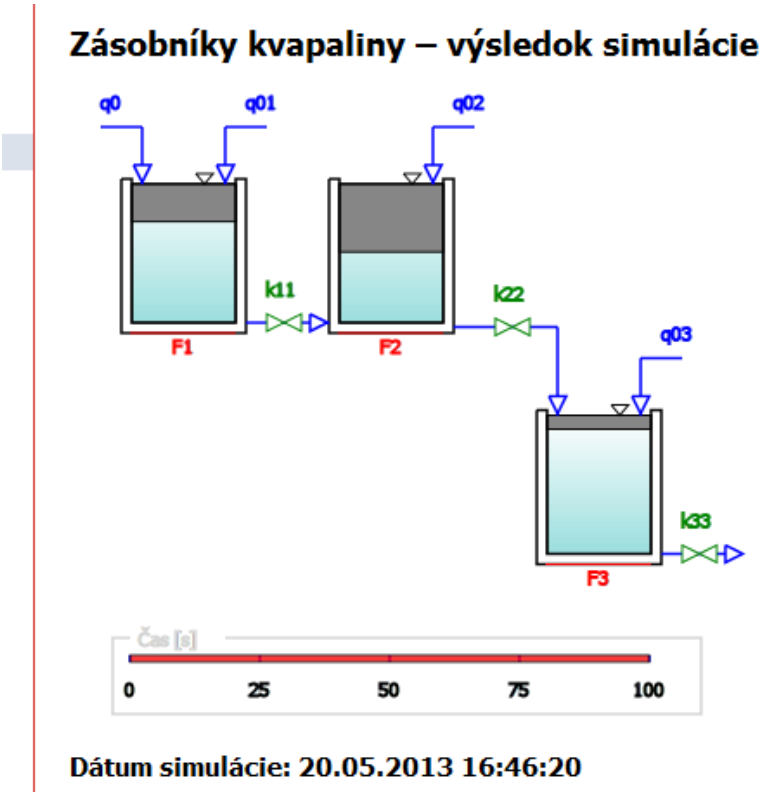


Obr. 22 Detail z formulára v časti zadávanie skokových zmien.

Pre niektoré procesy je v časti pod tlačidlami pre spustenie simulácie a vymazanie textových polí, vzorový príklad údajov pre simuláciu.

Po spustení simulácie, odoslaní údajov z formulára na spracovanie, je užívateľ presmerovaný na stránku zobrazujúcu výsledky simulácie. V prípade simulácií zásobníkov sú výsledky simulácie doplnené o animáciu hladiny v zásobníkoch pomocou SVG. Animácia nezohľadňuje veľkosť prierezu zásobníkov, sleduje len výšky hladín. Najväčšia hodnota výšky hladiny získanej zo simulácie sa berie ako 90% výšky zásobníka. Celá animácia je zrýchlená. Trvanie animácie neodpovedá trvaniu simulácie. Ukážka časti stránky zobrazujúcej túto animáciu je na

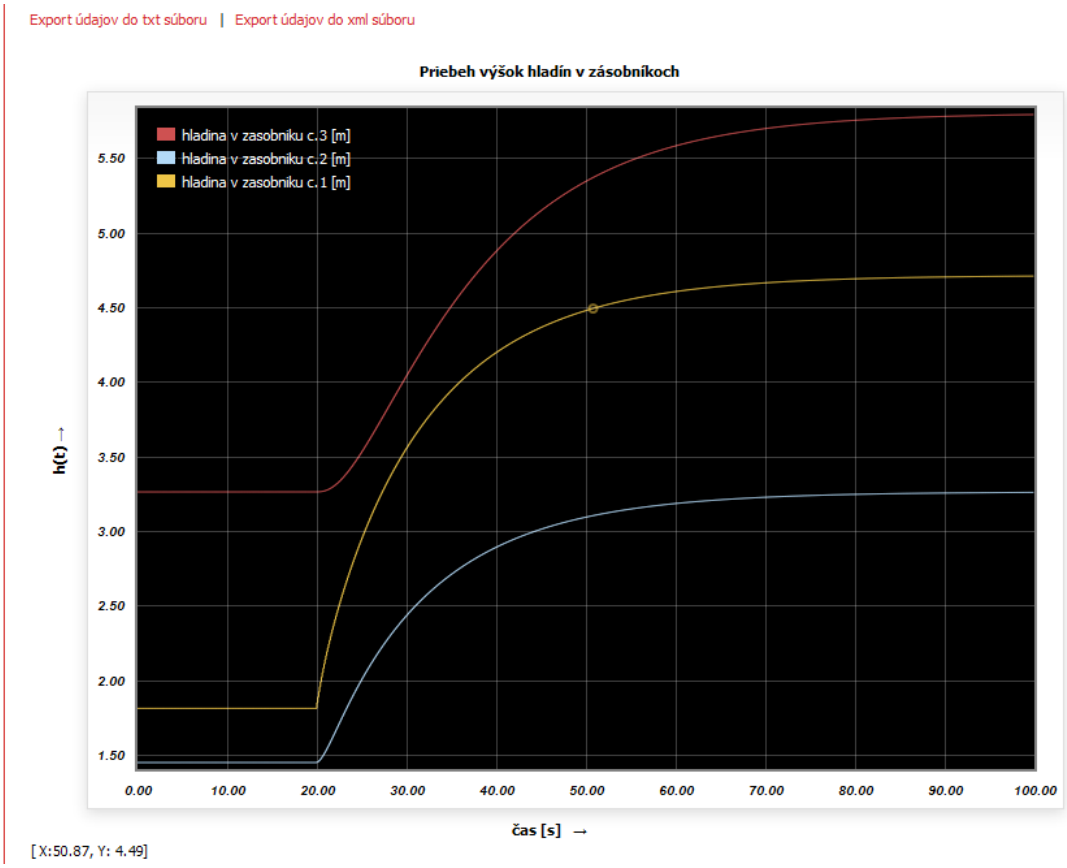
nasledujúcom obrázku.



Obr. 23 Detail stránky zobrazujúci animáciu výšok hladín v zásobníkoch

Pod animáciou výšok hladín sa na stránke nachádzajú odkazy na stiahnutie údajov zo simulácie do txt alebo xml súboru. Pri kliknutí na jeden z týchto odkazov vybehne užívateľovi ponuka pre stiahnutie súboru alebo jeho otvorenie. Pod týmito odkazmi sa nachádzajú grafy. Prvý graf reprezentuje priebeh výšok hladín počas simulácie. Druhý graf zobrazuje priebeh vstupných objemových prietokov do zásobníkov. Pri pohybe v každom z grafov sa užívateľovi zobrazujú súradnice x a y, na ktorých sa aktuálne v rámci grafu myš nachádza. Rolovaním kolieska myši je možné si graf priblížiť. Podržaním ľavého tlačidla myši je možné sa v rámci grafu pohybovať. Všetky grafy vykreslené v Modeltool majú prednastavené vykresľovanie grafu schodiskového typu. Na obr. 24 graf zobrazujúci priebehy výšok hladín v

zásobníkoch. V ľavom dolnom rohu obrázku možno vidieť súradnice polohy myši v rámci grafu .



Obr. 24 Detail stránky s interaktívnym grafom

To bol krátky opis modulu sériovo zapojených zásobníkov kvapaliny. Ostatné procesy majú rovnakú štruktúru a postupnosť pri celom procese simulácie. Pri exporte údajov zo simulácie do súborov txt či xml treba dodať, že pri txt súbore je obsah údajov v stĺpcoch nasledovný: čas, výstupy vstupy. Pri súbore xml je postupnosť sérií údajov nasledovná: výstupy potom vstupy.

5 Záver

Cieľom práce bolo rozšírenie ponuky procesov, ktoré sú k dispozícií na simuláciu vo webovom simulačnom nástroji Modeltool a zároveň upraviť možnosti tohto nástroja súčasným možnostiam. V rámci práce bolo do Modeltool pridaných šesť nových modulov. Bol zmenený systém pre vykresľovanie grafov. Nový systém umožňuje jednoducho pridávať interaktívne grafy s rozšírenými možnosťami nastavení pri tvorbe ďalších modulov. Každý proces bol doplnený o schému procesu s možnosťou exportu tejto schémy. V každom module bola pridaná možnosť exportu vybraných údajov získaných zo simulácie do textového alebo xml súboru. Jednotlivé moduly sa verifikovali porovnávaním výsledkov s výsledkami získanými pomocou programu MATLAB/Simulink. Porovnaním sa ukázala dobrá presnosť týchto výsledkov vzhľadom na MATLAB/Simulink.

Vytvorenie modulov s použitím metód, ktoré umožňujú asynchrónne načítanie stránky umožňuje užívateľovi poskytnúť k práci taký nástroj, ktorý nielen ponúka dobrý komfort, ale aj možnosti. Použitie schém podporilo tvorbu obrazu o virtuálnom procese, ktorý sa pozoruje. Pridaný export údajov umožňuje študovať a pracovať s výsledkami aj po simulácií.

Práca nastolila mnoho otázok a problémov, ktoré však boli mimo rozsah a zadanie práce, preto sme sa im nevenovali. Jedným z problémov v súvislosti s webovými stránkami je kompatibilita stránky s internetovými prehliadačmi. Tento stále aktuálny problém sa týka aj Modeltool.

Problém, ktorý vyplynul priamo pri tvorbe modulov pre Modetool je popis k početným častiam stránky. Popis k vzorovým príkladom, popis k údajom, ktoré boli použité pri simulácií a podobne. Popis sa často opakuje, kde sa obmieňajú indexy či

kľúčové slová. Tvorba tohto obsahu je často zdĺhavá a zaberá veľké množstvo času pri tvorbe nového modulu. Aké sú možnosti pri tvorbe čo najviac univerzálneho riešenia?

Posledným špecifickým problémom predstavuje modul pre simuláciu reaktora s viacerými zložkami a reakciami. Tento modul predstavoval z hľadiska verifikácie výsledkov náročnú úlohu. Pre úspešné získanie výsledkov bolo treba veľmi opatrne voliť parametre pre simuláciu. Tomuto modulu by bolo preto treba venovať viac priestoru. Zistiť aký počet reakcií a zložiek dokáže v rámci simulácie PHP spracovať.

Z pohľadu Modeltool, ako prostriedku určenému k štúdiu, by bolo priaznivé rozšírenie jeho možností viacerými smermi. Doplniť moduly tak, aby bolo možné simulovať ich riadenie pomocou PID regulátora, prípadne takéto riadenie rozšíriť o viaceré možnosti. Napríklad kaskádové riadenie výmenníka tepla, či pomerová regulácia pri miešaní. Možnosťou je doplniť Modeltool o nástroj na identifikáciu z nameraných charakteristík.

6 Zoznam zdrojov

[1] TULEJA S., Virtuálne laboratórium fyziky a matematiky[online], [cit. 2013-05-07]. Dostupné na internete: <http://www.stuleja.org/>

[2] ČVUT Praha Strojnícka fakulta, Virtuálne laboratórium automatického riadenia [online], [cit. 2013-05-07] Dostupné na internete: <http://vlab.fsid.cvut.cz/cz/index.php>

[3] KALÚZ M., Virtuálne laboratórium, Diplomová práca, FCHPT STU v Bratislave, 2010

[4] CU Boulder,PhET [online], [cit. 2013-05-08] Dostupné na internete: <http://phet.colorado.edu/>

[5] India MHRD,VLAB [online], [cit. 2013-05-08] Dostupné na internete: <http://www.vlab.co.in/>

[6] CircuitLab, Inc., CircuitLab [online], [cit. 2013-05-08] Dostupné na internete: <https://www.circuitlab.com/>

[7] University College Cork, Shell and Tube Heat Exchanger web-based virtual experiment [online], [cit. 2013-05-08] Dostupné na internete: <http://cs1.ucc.ie/~jb7/exch/index.html>

[8] Bakošová, M., Fikar M., - Kurz Riadenie proceov v systéme Moodle (pdf súbory prednášok) <http://www.kirp.chtf.stuba.sk/moodle/course/view.php?id=214>

[9] Bakošová, M. - Kurz Modelovanie v systéme Moodle (pdf súbory prednášok), <http://www.kirp.chtf.stuba.sk/moodle/enrol/index.php?id=160>

[10] Wikimedia Foundation,Inc. Free encyklopedia - Runge–Kutta methods [online], [cit. 2013-05-09] Dostupné na internete: <http://en.wikipedia.org/wiki/Runge%E2%80%93Kutta_methods>

[11] W3School,HTML [online], [cit. 2013-05-12] Dostupné na internete: <http://www.w3schools.com/html/default.asp>

[12]Wikimedia Foundation,Inc. Free encyklopedia – Scalable Vector Graphics [online], [cit. 2013-05-12] Dostupné na internete: <http://en.wikipedia.org/wiki/Scalable_Vector_Graphics>

[13] W3School,SVG [online], [cit. 2013-05-12] Dostupné na internete:<http://www.w3schools.com/svg/default.asp>

[14] Wikimedia Foundation,Inc. Free encyklopedia – Kaskádové štýly [online], [cit. 2013-05-12] Dostupné na internete:<http://sk.wikipedia.org/wiki/Kaskádové_štýly>

[15] Wikimedia Foundation,Inc. Free encyklopedia – PHP [online], [cit. 2013-05-12] Dostupné na internete:<http://en.wikipedia.org/wiki/PHP>

[16] W3School,AJAX Tutorial [online], [cit. 2013-05-13] Dostupné na internete:<http://www.w3schools.com/ajax/>

[17] W3School,AJAX Introduction [online], [cit. 2013-05-13] Dostupné na internete:<http://www.w3schools.com/ajax/ajax_intro.asp>

[18] W3School,jQuery Tutorial [online], [cit. 2013-05-13] Dostupné na internete: <http://w3schools.com/jquery/default.asp>

[19] IOLA and Ole Laursen l,JavaScript plotting for jQuery [online], [cit. 2013-05-14] Dostupné na internete:<http://www.flotcharts.org/>