Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra kybernetiky

BAKALÁŘSKÁ PRÁCE

PLZEŇ, 2013 **LUBOMÍR KRISTEK**

Zadání bakalářské práce

Uchazeč: **Lubomír Kristek**

Téma: **Vývoj uživatelského rozhraní v HTML5 pro testovací systém řídicích tyčí automobilů**

Topic: **Development of HTML5 based human-machine interface for steering shaft test stand**

Pokyny pro vypracování:

- Seznamte se s novým standardem HTML5 [1] pro tvorbu webových aplikací, se skriptovacím jazykem JavaScript [2], se škálovatelnou vektorovou grafikou SVG [3], s knihovnou jQuery [4] v jazyku JavaScript a s řídicím systémem REX [5].
- Dále se seznamte s aktuálním stavem průběžně vyvíjených nástrojů pro vizualizaci a ovládání v HTML5 ve vývojovém týmu oddělení automatického řízení na katedře kybernetiky.
- Navrhněte uživatelské rozhraní testovacího systému řídicích tyčí automobilů pro nejrozšířenější internetové prohlížeče (Google Chrome, Internet Explorer, Firefox)
- Implementujte a ověřte navrženou aplikaci na virtuálním modelu testovacího systému řídicích tyčí pracujícím v reálném čase.

Doporučená literatura:

[1] HTML5 Working Draft, online: http://www.w3.org/TR/html5/
[2] JavaScript tutorial, online: http://www.w3schools.com/js/default.asp
[3] SVG tutorial, online: http://www.w3schools.com/svg/default.asp
[4] jQuery, online: http://jquery.com/
[5] REX Controls: Funkční bloky systému REX. Referenční příručka. 2012.

Prohlášení
Předkládám tímto k posouzení a obhajobě bakalářskou práci zpracovanou na závěr studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni.

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím odborné literatury a pramenů, jejichž úplný seznam je její součástí.

V Plzni dne 9.5.2013

.....

Poděkování

Na tomto místě bych rád poděkoval Pavlu Baldovi za neocenitelné připomínky k práci a čas strávený konzultacemi. Rovněž si velký dík zaslouží Ondřej Severa za pomoc s použitými technologiemi a trpělivost při vysvětlování webových aplikací.

Anotace

Tato práce se zabývá vývojem webové vizualizační aplikace. Cílem je vyvinout rozhraní pro ovládání modelu momentového standu a zobrazení výsledků ve webovém prohlížeči. Aplikace je vyvíjena ve formě tenkého klienta. Pro tyto účely byly využity možnosti HTML5 a SVG. Dále je popsána a implementována technologie pro přenos dat mezi serverem a internetovým prohlížečem prostřednictvím rozhraní WebSocket. Také jsou nastíněny možnosti využití systému REX pro simulaci dějů na reálném momentovém standu. Pro zpracování naměřených dat a řešení matematických úkolů je použit program MATLAB. V závěru jsou diskutovány možnosti využití matematických modelů pro modelování hysterezních křivek, odpovídajících naměřeným výsledků.

Klíčová slova

HTML5, SVG, tenký klient, WebSocket, REX, REXLANG, Fourierovy řady, MATLAB, hystereze, Bouc-Wen

Anotace v angličtině

This thesis deals with development of the web based visualization application. The goal is to develop human-mashine interface for a moment stand model in a web browser. The application is developed in a form of thin client. For this purpose, capabilities of HTML5 and SVG have been used. Further the technology for transferring data between server and web broswer via WebSocket is described and implemented. Also the possibility of using the REX control system for simulation of processes on a real machine is described. The software MATLAB is used for processing of measured data and solving mathematical tasks. The possibility of using mathematical models for modeling hysteresis curves corresponding to the measured data is discussed at the end of the thesis.

Keywords

HTML5, SVG, thin client, WebSocket, REX, REXLANG, Fourier series, MATLAB, hysteresis, Bouc-Wen

Obsah

1. Úvod

Cílem této práce je vytvořit vizualizaci průběhu testu řídicí tyče automobilu. Výsledná vizualizace bude simulovat uživatelské rozhraní, používané ve skutečném průmyslovém procesu. Testovací stand (Obr. 1) i původní vizualizační systém jsou popsány ve druhé kapitole.
Jedním z požadavků bylo, oddělit od sebe jednotlivé vrstvy na zdoj dat, přenos dat a jejich vizualizaci. Tento přístup je popsán ve třetí kapitole 'Architektura'.
Generování dat je zajištěno simulačním schématem v systému REX. O tomto programovém balíku, práci s ním a postupu generování dat pojednává kapitola čtyři.
Popis technologií pro přenos dat a způsob jejich použití je obsažen v páté kapitole.
Vlastní vizualizace a uživatelské rozhraní jsou vysvětleny v šesté kapitole. V této části práce bylo nutné splnit několik požadavků na vyvinutou aplikaci. Jedním z nich je požadavek na vývoj formou tenkého klienta. Jedná se o druh webové aplikace, který pro svou činnost nevyžaduje žádná rozšíření či instalaci doplňků ze strany koncového uživatele. Zájemcům o vizualizaci postačuje k jejímu spuštění webový prohlížeč s podporou HTML5 {značkovací jazyk pro hypertext - HyperText Markup Language}[1]. Tento nový standard má v budoucnu nahradit stávající koncept HTML 4.01, který byl vydán v roce 1999 a nadále nevyhovuje stávajícím trendům na webu. HTML5 vzniká za spolupráce institucí W3C {World Wide Web Consortium} a WHATWG {Web Hypertext Application Technology Working Group}. Hlavními cíli projektu je:

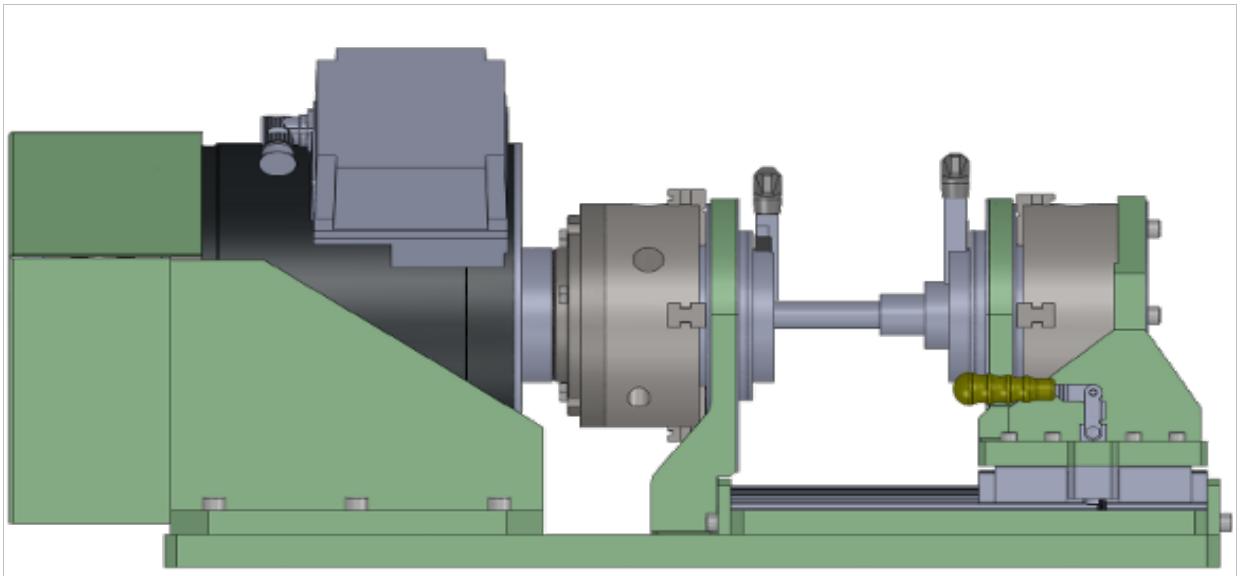
- zaváděné technologie by měly být postavené na HTML, CSS {kaskádové styly - Cascading Style Sheets}, DOM {objektový model dokumentu - Document Object Model} a jazyce JavaScript
- snížit množství potřebných externích doplňků (jako je Flash, Java)
- lepší vypořádání se s chybami
- HTML5 bude platformově nezávislé
- více nových elementů sníží nutnost současného skriptování

Jedním z těchto nových prvků je také SVG {škálovatelá vektorová grafika - Scalable Vector Graphics}. Používá se pro grafiku založenou na matematicky definovaných obrazcích, zobrazitelnou přímo v prohlížeči. Zajímavé prvky tohoto elementu a celé technologie jsou:

- výsledný obraz je definován v XML {rozšířený značkovací jazyk - eXtensible Markup Language} formátu
- grafika neztrácí nic ze své kvality, pokud je měněna její velikost
- každý prvek vystupuje jako objekt, který může být animován, případně je možné jinak řídit jeho vlastnosti (barvu, tvar..)

Z těchto důvodů se koncept SVG výborně hodí pro požadovanou vizualizaci.

Během vývoje simulačního schématu vznikla potřeba generovat data do vizualizace matematicky, nikoliv čtením z naměřených dat, jak tomu bylo zpočátku. Postup při získání matematického generátoru dat je popsán v sedmé kapitole.
Další možnost, jak získávat data pro vizualizaci, bylo vytvořit matematický model celého experimentu. Získání matematického modelu se věnuje kapitola osm.



Obr. 1: Momentový stand

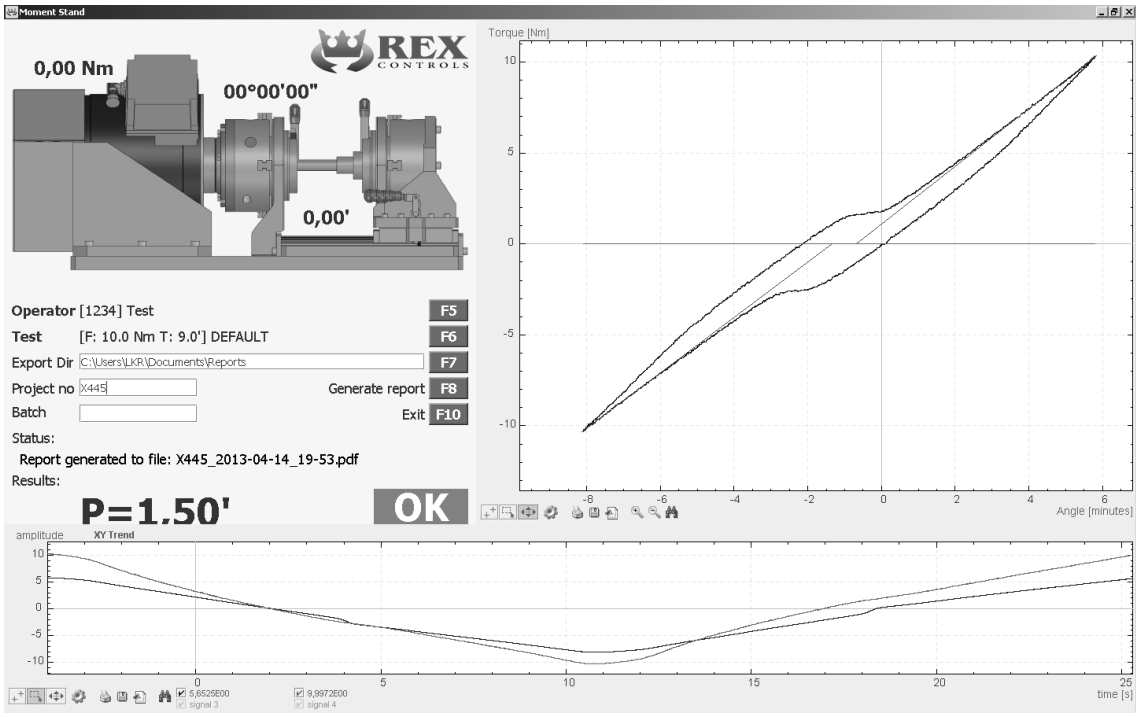
2. Popis úlohy

2.1. Momentový stand

Momentový stand, ke kterému měla být vizualizace vytvořena je stroj, vyvíjený pro firmu FujiKoyo. Cílem bylo vyvinout stand, který bude testovat vůli v řídících tyčích automobilů. Celý experiment probíhá tak, že motor, ke kterému je upnuta tyč působí známým momentem na tyč a na druhé straně je měřena odchylka v úhlových minutách.

2.2. Operátorský software

Součástí standu bylo i softwarové vybavení (Obr. 2). To umožňuje obsluze standu vybrat operátora a typ experimentu. Pro generování souboru '.pdf' s výsledky experimentu je možné zvolit cestu k cílovému adresáři a název aktuálního projektu. Průběh testu je zobrazován ve dvou grafech, horní vykresluje závislost úhlu deformace na působící síle, spodní průběh obou hodnot v čase. Obrázek stroje je doplněn o dynamický text, kde jsou vypisovány aktuální hodnoty síly a deformace.Vygenerovaný report (Obr. 3) obsahuje informace o podmínkách a průběhu testu, zjištěné vůli v řídicí tyči, datum a čas vytvoření a informace o obsluhujícím operátorovi. Dokument je doplněn o grafické znázornění průběhu testu.



Obr. 2: Operátorský software

Fuji Koyo

Lash measurement protocol

X445

Details:

p_1 = 1.425'

p_2 = 1.503'

F_{max} = 10.0 Nm

Date: 2013-04-04

Shift: Day

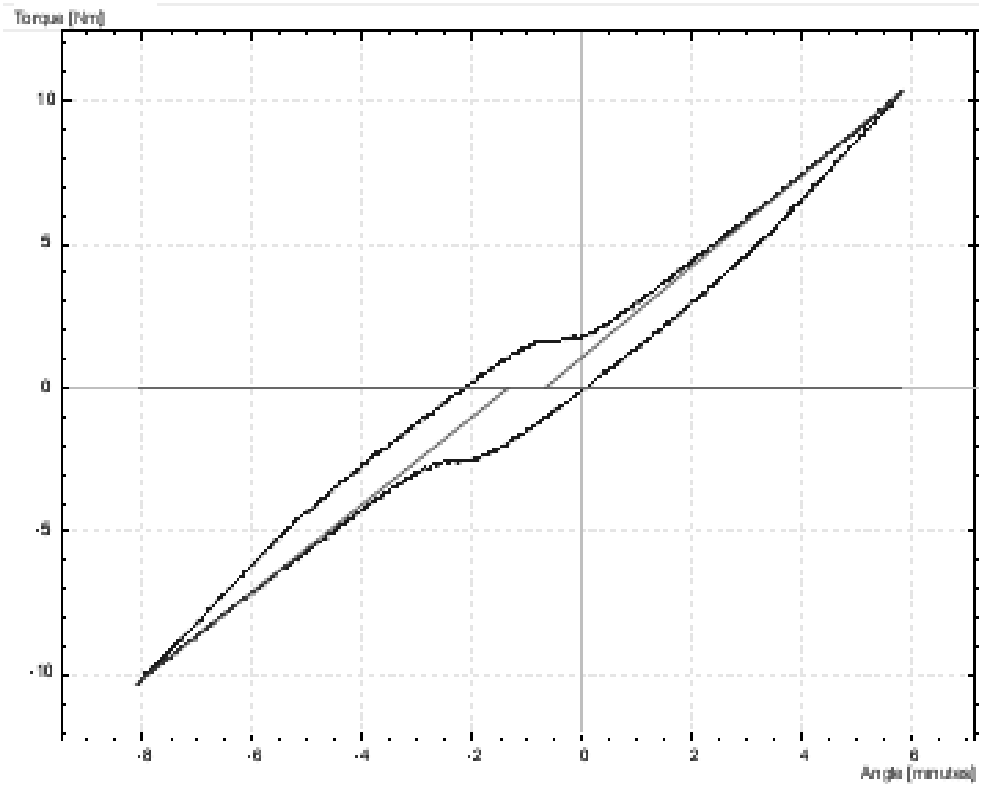
Time: 09:02

Op. name: Test

Op. ID: 1234

Batch:

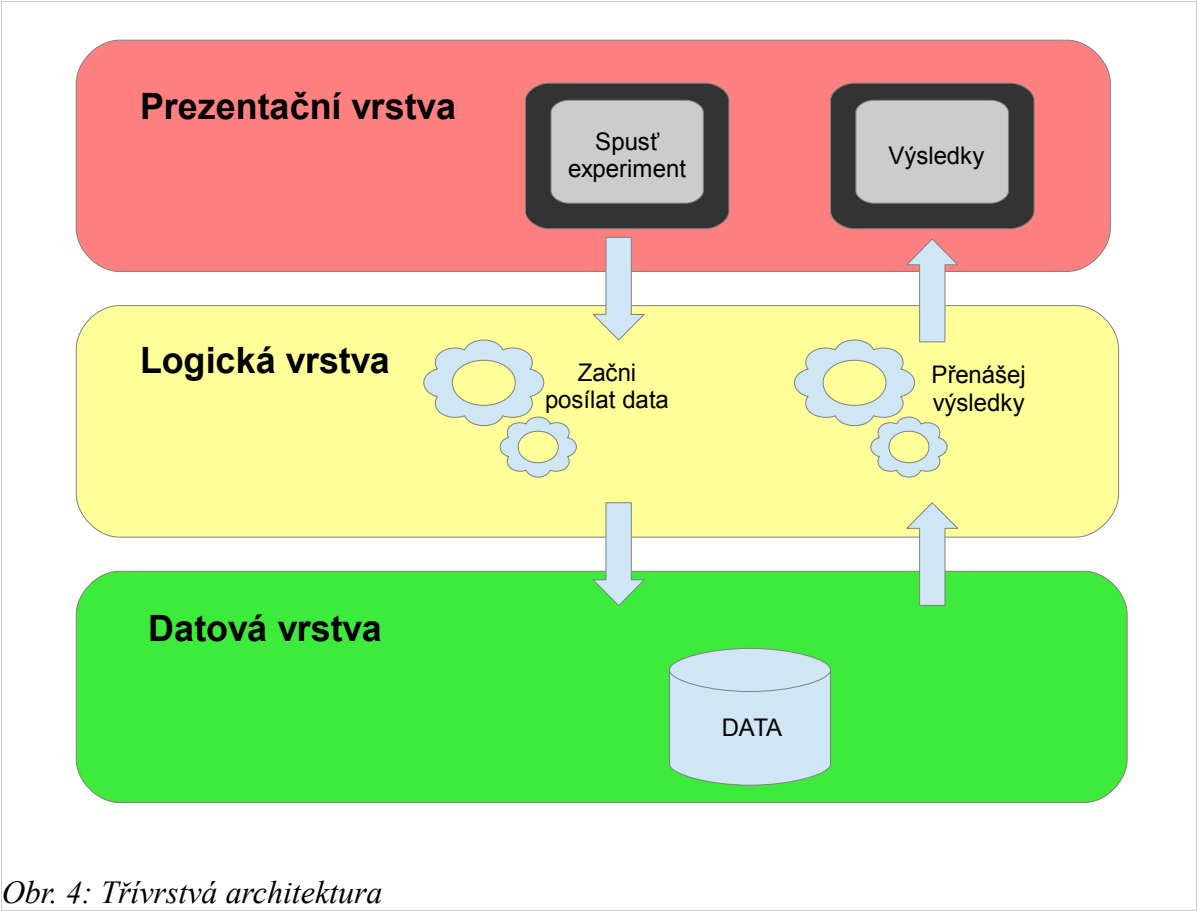
Results:	p	Tolerance	Status
	0° 1' 30.2"	0° 9' 0.0"	OK



Obr. 3: Ukázka reportu

3. Architektura

Řešení úlohy vychází z konceptu třívrtsvé architektury [2]. Tento přístup je v praxi hojně používán při vývoji webových stránek a aplikací. Skládá se ze tří nezávislých úrovní (Obr. 4). To například poskytuje výhody při výběru komponent, protože každá vrstva může běžet na samostatném stroji. Každá z vrstev může být vyvíjena a udržována samostatně. Navíc nemá uživatel přímý přístup k datům a nemůže je ovlivňovat nad rámec pravomocí, které jsou mu přiděleny.



Obr. 4: Třívrstvá architektura

Z ilustrace je patrné, že data jsou schromažďována v datové vrstvě a jsou oddělena od samotné vizualizace. V našem případě slouží jako zdroj dat simulační schéma v REXu. Podrobněji v kapitole '4 Datová vrstva'. S datovou vrstvou komunikuje logická (transportní) vrstva. Předává požadavky od klienta zdroji dat a získané výsledky intrepretuje a přenáší do prezentační vrstvy. Komunikaci v tomto případě zajišťuje rozhraní WebSocket. Prezentační vrstva se stará o vizualizaci dat na straně klienta a poskytuje uživatelské rozhraní pro ovládání experimentu. V souladu s požadavky na výslednou práci je vizualizace a GUI {Graphical User Interface - grafické uživatelské rozhraní} realizováno v HTML5 s využitím SVG a JavaScriptu. Tímto přístupem je

docíleno splnění požadavku na tenkého klienta. Zájemcům o ukázku práce momentového standu stačí ke shlédnutí příkladu pouze novější verze prohlížeče s podporou HTML5 (Chrome, Firefox), bez nutnosti instalace jakýchkoliv doplňků či rozšíření. Ostatní vrstvy jsou před ním skryty a fungují samostatně.

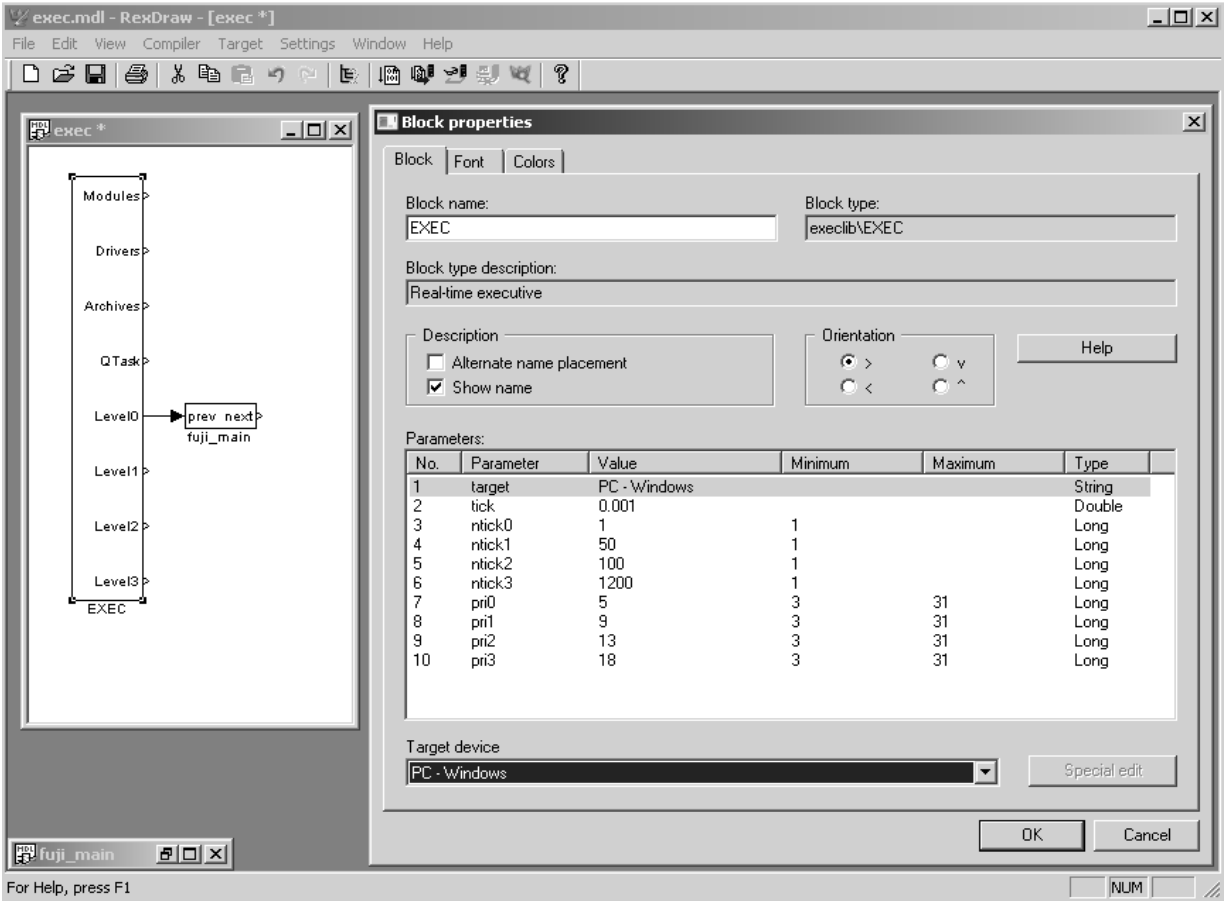
4. Datová vrstva

4.1. REX

Jak již bylo zmíněno výše, o generování potřebných dat se stará simulační schéma běžící v řídicím systému REX. Jedná se o nástroj, používaný v řízení průmyslových procesů a dalších oblastí, vyžadujících automatické řízení [3]. Samotný systém se skládá z několika částí.
Program RexDraw slouží pro návrh funkčních schémat systému. K dispozici je knihovna bloků, ze kterých může uživatel vybírat a zadanou úlohu postavit podle svých potřeb.
RexView umožňuje sledovat, co se v řídícím jádře programu odehrává za běhu. Využívá se při zavádění řídicích algoritmů do praxe a ladění vzniklých problémů.
RexComp je překladač, generující z výchozího souboru '.mdl', vytvořeného v RexDraw, konfigurační binární soubor '.rex' řídícího systému.
RexCore je jádro programu. Běží na cílovém zařízení a stará se o vlastní exekuci navrženého řízení.

4.2. Tvorba schématu

Každý projekt v systému REX musí obsahovat minimálně dva soubory '.mdl'. Jedním je hlavní soubor projektu, ve kterém se konfigurují jednotlivé úlohy, ovladače, priority. (Obr. 5) Další soubory obsahují jednotlivé úlohy, které se skládají z jednotlivých bloků z knihovny programu. Jedná se například o funkce pro generování a zpracování signálu, matematické operace, bloky vstupů a výstupů, práci s daty atd.. Pro použití v regulačních obvodech má program knihovny bloků speciálně určených pro regulaci a logické řízení [4].

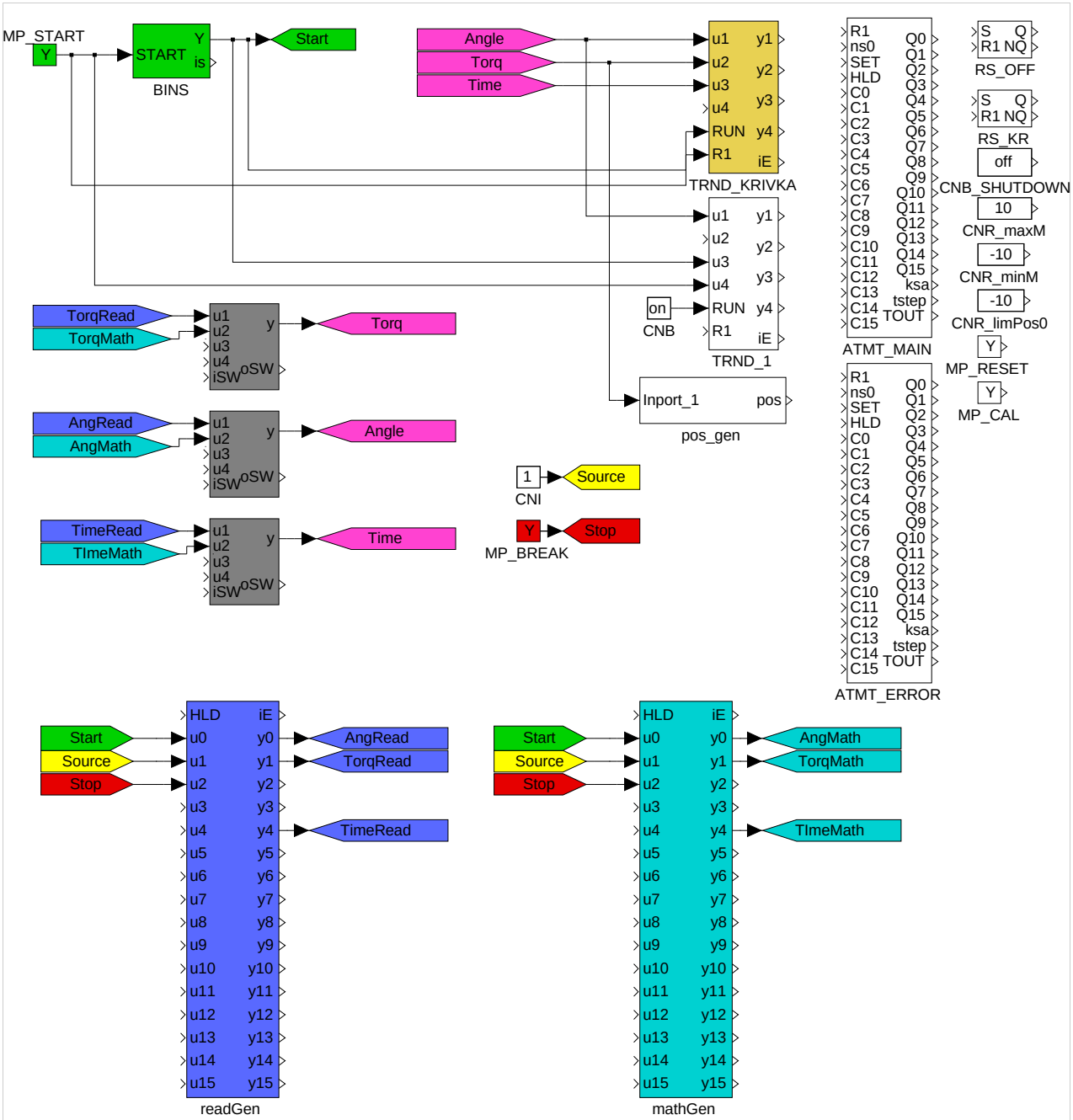


Obr. 5: Hlavní soubor a jeho konfigurace

4.3. Zdroj dat v systému REX

Simulační schéma, jež zajišťuje generování dat, muselo splňovat následující požadavky:

- Kompatibilita s obdobným schématem, obsaženém ve vlastní realizaci stroje. Při splnění tohoto požadavku bylo možné pro testování generovaných dat použít původní software, psaný v jazyce Java. To urychlilo celý proces vývoje generátoru dat.
- Umožnit základní ovládání na následující úrovni: možnost zvolit jeden z generátorů dat, spustit a zastavit proces. Generátory k dispozici jsou čtení z dat, naměřených při dvou experimentech, a jeden matematický generátor.
- Generování dat musí být rychlejší, než perioda načítání dat do stránky, aby byla zajištěna plynulá vizualizace.



Obr. 6: Schéma zdroje dat v programu RexDraw

V tomto simulačním (Obr. 6) jsou prvky barevně rozlišeny podle funkce. Zelené jsou pro realizaci funkce tlačítka 'Start', spouštějícího experiment. Žluté ovládají volbu zdroje. Červené slouží pro zastavení experimentu. Tmavě modré bloky jsou spojené s generováním dat čtením ze souboru. Světle modré bloky patří ke generování dat z matematicky. Šedivé bloky vybírají signál, jenž bude předán k vizualizaci. Růžové značí přenos výsledných dat a hnědý blok uchovává výsledná data k předání do vizualizace.

4.4. REXLANG

Oba generátory jsou realizovány blokem REXLANG. Jedná se o volně programovatelný blok, ve kterém je možné realizovat vlastní funkci ve skriptovacím jazyce podobném jazyku C nebo Java. Blok má povinné parametry pro nastavení jména a typu souboru se zdrojovým kódem, velikosti zásobníku, úrovně ladících kontrol a jména přidruženého datového souboru. Navíc může uživatel použít až 16 vlastních parametrů, které jsou přístupné ze skriptu. Dále je k dispozici 16 vstupů a výstupů, jejichž funkce je též ovládána ze skriptu (Obr. 7).

```
int input(0) start;
int input(1) source;
int input(2) stop;

double output(0) x;
double output(1) y;
double output(2) z;
double output(3) w;
double output(4) t;

int parameter(0) elements;
double parameter(1) period;
int parameter(2) columns;
```

Obr. 7: Ukázka práce se vstupy, výstupy a parametry

Skript je možné napsat v jakémkoliv textovém editoru. V tomto případě je využit PSPad, který nabízí zvýraznění syntaxe pro snažší orientaci v kódu.

4.5. Čtení hystereze

Výsledná funkce pro čtení (Obr. 8) je vytvořena tak, aby umožňovala číst 2 až 4 sloupce dat ze souboru a přiřazovala je na výstupy. Čtení probíhá periodicky s volitelnou dobou periody. Využívám konfiguraci, jež čte 30 000 prvků 30 sekund. Blok umí číst ze dvou souborů, mezi nimiž je možné přepínat hodnotou na vstupu 'Zdroj'. Po detekci logické 1 jedna na vstupu 'Stop' přestane blok číst a na výstupy je přiřazena nula. Při aktivaci tlačítkem 'Start' začíná číst vždy od začátku souboru. Hodnoty v datových souborech jsou uloženy jako jeden sloupec čísel formátu double, střídavě pod sebou hodnoty 2-4 vektorů. Při použití takto připraveného souboru dosáhne

program nejvyšší rychlosti čtení ze souboru, protože je zde využita možnost sekvenčního čtení. Druhý přístup ke čtení z připravených dat bylo využít polí v samotném programu. Výhodou tohoto přístupu bylo, že celé pole se načetlo do paměti při inicializaci programu a čtení samotných dat poté probíhalo velmi rychle. Nepříjemná vlastnost byla, že při každé změně vstupních dat bylo nutné přepisovat zdrojový kód bloku. V případě tohoto přístupu by nebylo možné měnit vstupní data po kompilaci bloku a zavedení do jádra RexCore, proto nebyl ve výsledné práci použit.

```
int main(void){
  if(source == 1){
      if(stop == 1){hold = 1;}
      if(start == 1 && hold== 0){
        counter = counter + inTick;
        if(floor(counter) > floor(counter-inTick)){
         if((counter-inTick) == 0){
                  x=LoadValue(4,0);
         }
         if((counter-inTick) != 0){
           x=LoadValue(4,-1);
         }
         y=LoadValue(4,-1);
         if(columns > 2){ z=LoadValue(4,-1); w=0;
                  if(columns > 3){ w=LoadValue(4,-1);}
          }
        }
      }
       if(start == 0){
         hold = 0;
      }
       if(hold == 1 || start == 0){
         x=0;
         y=0;
         counter= 0;
       }
       t=counter;
  }
```

Obr. 8: Skript pro čtení ze souboru

4.6. Matematický generátor

Blokem REXLANG je realizován i matematický generátor hystereze. Vstupy bloku jsou hodnoty 'Start','Stop' a 'Zdroj', výstupem potom působící moment, úhel deformace a simulační čas. Získání matematického generátoru a modelu jsou věnovány kapitoly 7 a 8.

4.7. Výchozí data

Data poskytnutá k vizualizaci byla ve formě '.csv' souboru. Tato data byla získána měřením na reálném standu pro testování řídících tyčí automobilů. Původní soubory obsahovaly dva sloupce hodnot, v prvním sloupci moment, kterým působil motor standu, ve druhém naměřená odchylka na druhém konci stroje. Pro převod těchto hodnot do formátu, jenž bych mohl použít, bylo nutné vytvořit nástroj pro konverzi. Pro tyto účely jsem použil MATLAB, obsahující mimo jiné obsáhlý balík funkcí pro práci s textovými soubory [5]. Bylo nutné vytvořit funkci (Obr. 9), jež bude splňovat následující požadavky:

- Načtení dat z '.csv' souboru. V některých souborech byla použita čárka jako oddělovač desetinných čísel. U těchto souborů program jako první nahradí tyto čárky za tečky, kvůli kompatibilitě s reprezentací čísel v programu MATLAB.
- Bude možné zvolit počet řádků ve výsledném souboru. Uživatel si může zvolit, zda chce do výstupního souboru zahrnout všechny hodnoty, nebo jen určitý počet. Při počtu menším než původní jsou prvky rovnoměrně vynechávány přes celý soubor.
- Pokud byly prvky vynechávány, nabízí program funkci průměrovat přes vynechávané hodnoty.
- Lze zvolit ze dvou výstupů. Jedním z nich je soubor, obsahující hodnoty střídavě pod sebou pro čtení dat ze souboru. Druhým je skupina 2-4 souborů pro inicializaci interních polí. Z těchto souborů byla data přenášena do zdrojového kódu

```
% [M] = csvToArray(jmeno,typ, pocetPrvku, prumerovat)
% jmeno: jmeno puvodniho souboru
% typ:  1 vystupem jsou dva az ctyri soubory jez obsahuji pripravene
pole
%         hX a hY (hZ, hW)
%       2 vystup je soubor s nazvem puvodniho a priponou .dat,
obsahuje 1
%         sloupec ve formatu [x0 y0 (z0 w0) x1 y1 (z1 w1) x2..]'
%      12 aplikuje oba pristupy
% pocetPrvku: rika, kolik prvku bude zachovano, -1 pro vsechny
% prumerovat: 1 pokud chci pres vynechavane prvky prumerovat
% M je matice, ktera obsahuje cislo radku od 0, hodnoty t, x a y (z,w)
%
% skript vzdy prepise puvodni soubor, zmeni desetinne carky za tecky

function [M] = csvToArray(jmeno, typ, pocetPrvku, prumerovat)
```

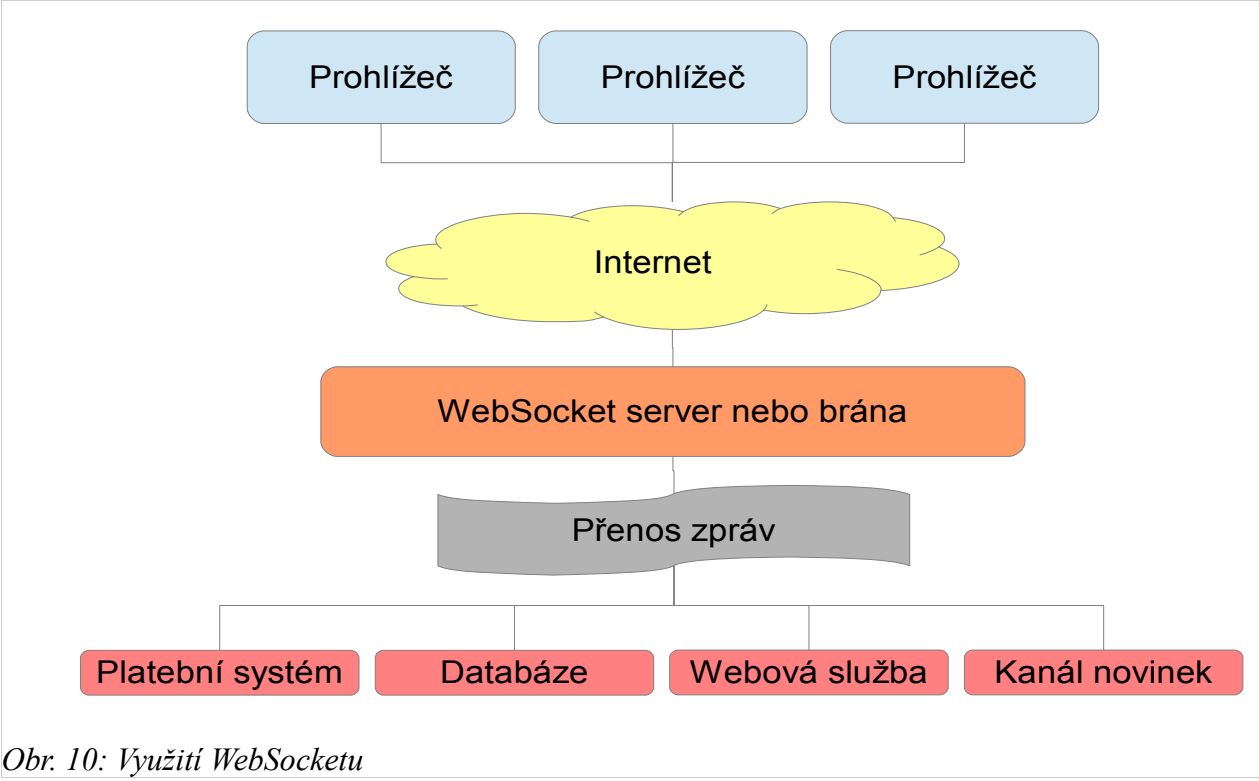
Obr. 9: Hlavička funkce pro převod souborů do požadovaného formátu

5. Přenosová vrstva

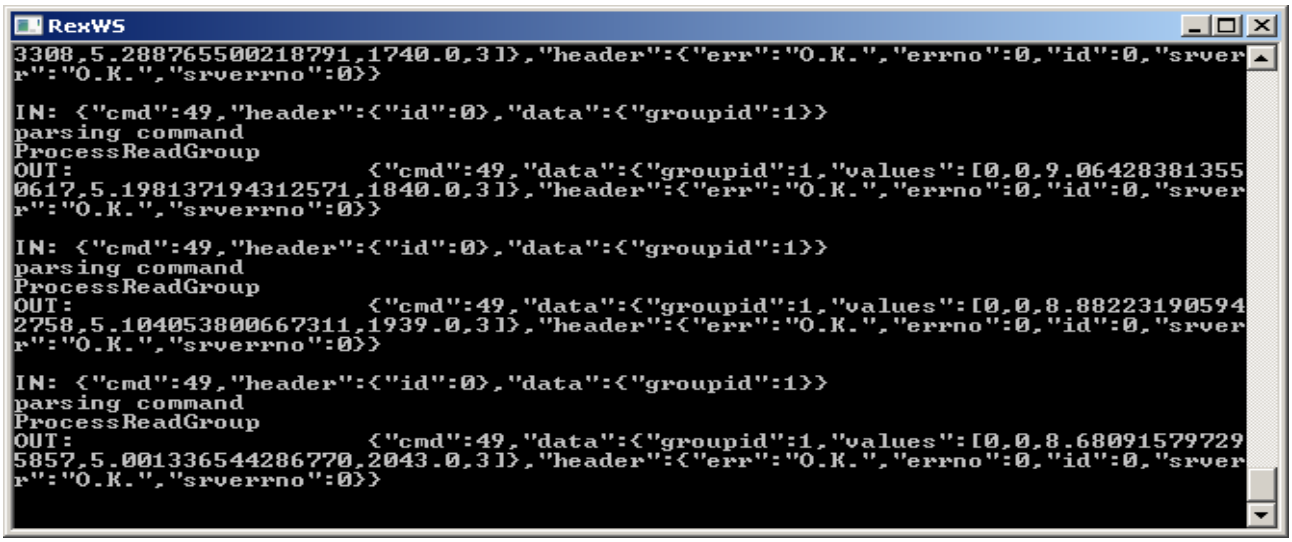
O obousměrný přenos dat mezi simulačním schématem a vizualizací ve webovém prohlížeči se stará REX WebSocket. Tento program je založen na WebSocket specifikaci [6]. Jedná se o rozhraní v jazyce JavaScript, vyvíjené v rámci nového standartu HTML5, které definuje spojení pro přenos zpráv mezi klientem a serverem (Obr. 10). WebSocket standard usnadňuje vytvoření a správu obousměrného webové komunikace.

5.1. Použití rozhraní WebSocket

Způsob použití tohoto rozhraní pro účely požadované vizualizace je velmi jednoduchý. Po načtení webové stránky je vytvořen přenosový kanál mezi webovým prohlížečem a simulačním schématem. Přes něj jsou periodicky posílány textové zprávy (Obr. 11). V nich jsou informace soustřeďovány v tzv. grupách. Jedná se o strukturovaný objekt pro předávání hodnot. V tomto případě jsou přenášeny z vizualizace pokyny 'Start', 'Stop' a 'Zdroj', simulace předává informace o působící síle, úhlu deformace a času vizualizace. Toto čtení a zápis probíhá desetkrát za vteřinu. Při každém úspěšném přečtení grupy je vyvolána funkce 'readCallback'. V ní jsou předána data do funkcí obsluhujících trend (poloha x a y) a aktualizujících dynamický text u obrázku standu a ilustrační grafické prvky.



Obr. 10: Využití WebSocketu



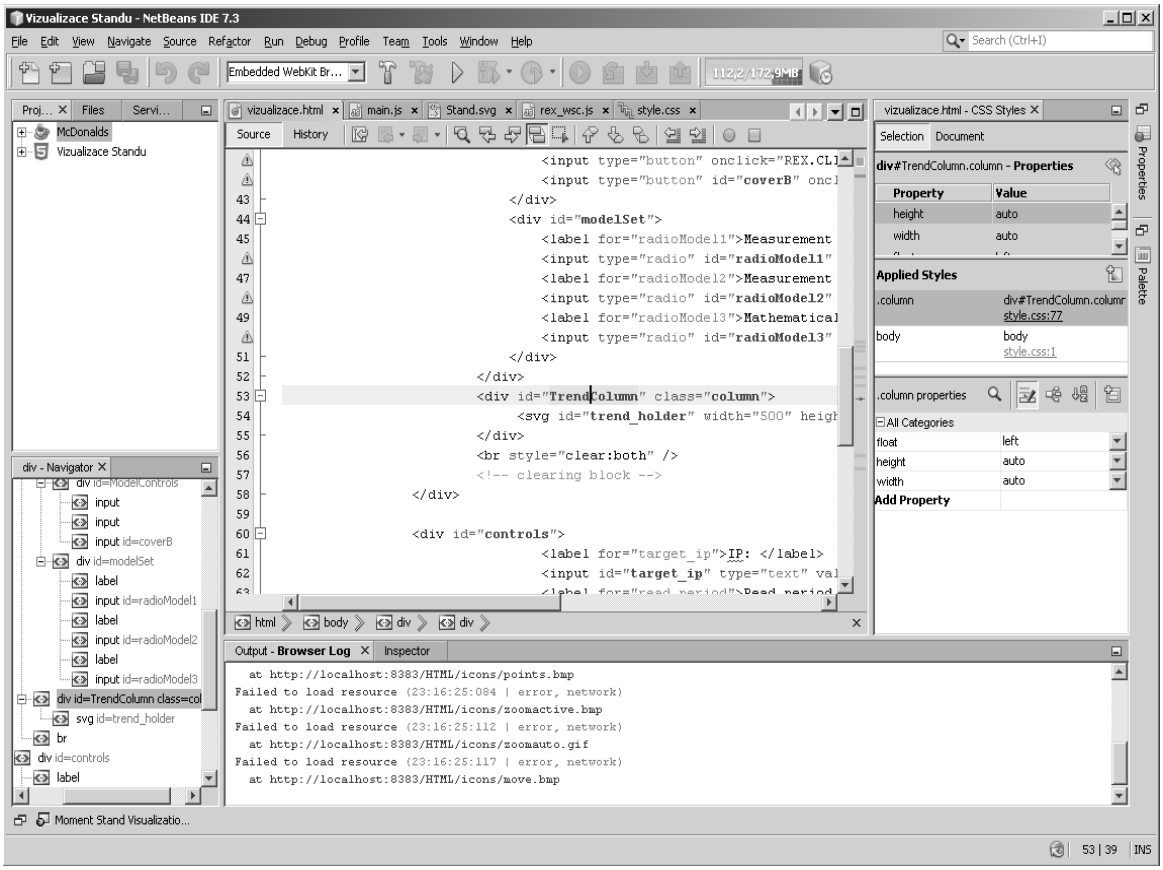
Obr. 11: Přenos zpráv WebSocketem

6. Vizualizace

Poslední vrstva je vlastní vizualizace. Je to jedinný uživatelův nástroj pro ovládání experimentu a zároveň i místo, kde jsou výsledky prezentovány. Grafická úprava a uživatelská přívětivost zde hrají nejvyšší roli. Zároveň zde pro úspěch aplikace klíčová podpora webových prohlížečů. Tato práce byla testována a optimalizována pro prohlížeče Chrome a Firefox.

6.1. Vývojové prostředí

Vývoj vizualizace proběhl v prostředí NetBeans (Obr. 12), doposud hojně využívaný pro vývoj v Javě. Tento nástroj poskytuje od verze 7.3 podporu pro vývoj HTML5 aplikací [7]. Součástí tohoto balíku je rozšířená podpora nejen HTML5, ale i testování JavaScriptu, vizuální editor CSS a WebKit integrovaný v prohlížeči Chrome.



Obr. 12: Prostředí programu NetBeans

6.2. Kompozice stránky

Aplikace je rozdělena do několika souborů. Tento přístup, častý při vývoji webových stránek. má tu výhodu, že jednotlivé součásti mohou být vyvíjeny a spravovány odděleně. Základem je vlastní

soubor '.html', který obsahuje jednotlivé prvky stránky ve struktuře podobné seznamu. O vzhled těchto prvků a celé stránky se starají soubory typu '.css'. V nich lze nastavovat jednotlivé atributy prvků a dosáhnout tak požadovaného vzhledu. O funkčnost stránky se na pozadí stará JavaScript, umožňující vývojáři definovat funkčnost a chování jednotlivých prvků.

6.3. HTML

Při vytváření stránky jsem vycházel z příkladu webové servisní vizualizace MTUNER, obsaženém ve standardní instalaci systému REX [8]. Výsledná stránka se skládá z několika oblastí. Nahoře začíná hlavičkou, která nese název úlohy. Následuje tělo, ve kterém jsou dva svg prvky. V prvním obrázek textu s dynamickými popisky, třemi ilustračními grafickými prvky a sadou tlačítek pro ovládání experimentu. Ve druhém pak graf, ve kterém je vykreslován na ose x aktuální úhel, na ose y působící moment. Následuje oddíl, kde jsou zobrazovány informace o připojení na simulaci a průběhu experimentu. Nakonec patička s doplňujícími informacemi.

6.4. Styly CSS

Pro následnou grafickou úpravu jsem využil styly css. Výsledná vizualizace obsahuje dva tyto soubory. Jeden pro samotnou stránku:

- V hlavičce je nastavena tmavě modrá barva pozadí a bílé písmo o velikosti 50 pixelů umístěné doprostřed.
- Tělo je podbarveno slabým odstínem modré.
- Pro obrázek standu a graf je vyčleněna třída slouec, díky které můžou být umístěny vedle sebe.
- Log je modrý, písmo je zde menší. Speciální vlastností této části je neměnná velikost, nové texty jsou zobrazovány nahoře a staré se 'ztrácí' pod spodní okraj.
- Následuje patička, se stejnou barvou jako hlavička, bílím menším písmem zarovnaným vpravo.

Druhý soubor stylů se stará o vizualizaci ovládajících prvků (tlačítek). Jedná se o styl vygenerovaný jako součást knihovny jQuery UI, která zajišťuje funkčnost uživatelského rozhraní [8]. Výsledná stránka (Obr. 13) kombinuje oba tyto styly.

6.5. JavaScript

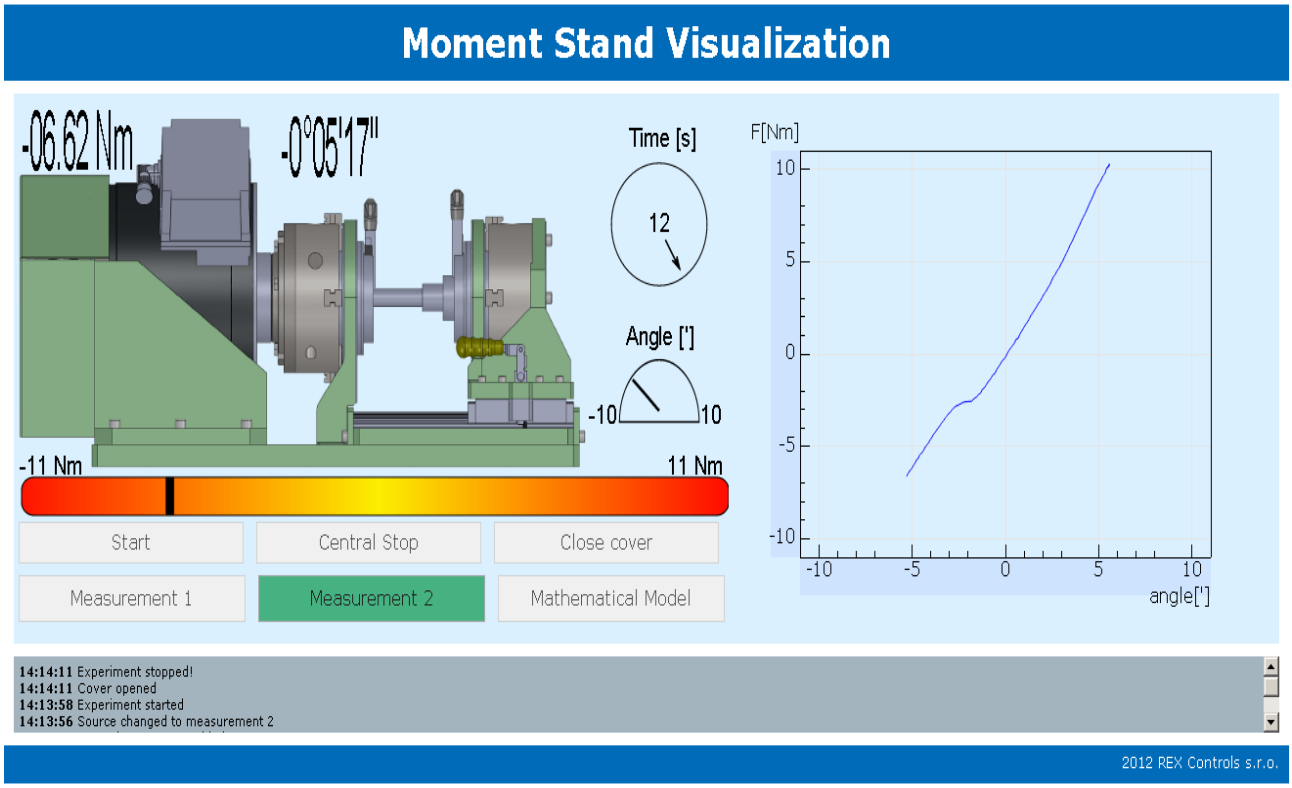
JavaScript, jak již bylo řečeno, zajišťuje funkčnost celé stránky. Vizualizace obsahuje několik souborů se skripty:

- *jquery-1.8.2.js* zajišťuje funčnost těch částí, které jsou napsány s využitím knihovny jQuery

[9]

- *jquery-ui.js* se stará o funkci ovládacích prvků, založených na jQueryUI [10]
- *rex_weblib.min.js* je určený pro komunikaci se systémem REX prostřednictvím WebSocketu a zároveň obsahuje nástroje pro vytvoření a práci s grafem
- *main.js* pro zajištění funkčnosti tlačítek a vizualizačních komponent, které jsou součástí obrázku standu

Poslední zmiňovaný byl vytvořen pro účely této vizualizace. Část opět vychází z již zmiňovaného příkladu. Jde o části, kde je navázáno spojení s WebSocketem a zobrazní trendu. Po načtení stránky začíná JavaScript pracovat. Naváže spojení se simulací a doplní připravený oddíl trendholder o graf. Při stisknutí tlačítka 'Start' je přenesen pokyn, který spustí buď čtení naměřených hodnot, případně matematický generátor v závislosti na volbě zdroje. 'Central Stop' simulaci zastaví. Obdobně funguje i tlačítko 'Open Cover'. To rovněž způsobí zastavení simulace v případě že probíhá, protože dojde k činnosti simulující otevření standu za běhu na reálném stroji. Text tohoto tlačítka se změní na 'Close Cover' a proměnná nesoucí informaci o krytu zůstává na hodnotě 'otevřeno', dokud uživatel kryt znovu nezavře. Za podmínky že je kryt otevřen není možné znovu spustit simulaci.



Obr. 13: Ukázka hotové stránky

7. Matematický generátor dat

Jedním z požadavků na výslednou práci bylo mít jeden generátor dat čtením z naměřených hodnot a další pro matematické získávání dat.

7.1. Fourierova řada

První uvažovaná metoda, jak získat matematický generátor, bylo použít rozvoj vstupního a výstupního signálu do Fourierovy řady.

Fourierovy řady vychází z myšlenky, že jakýkoliv periodický signál je možné vyjádřit sumou sinusoidů [11].

$$v(t)=\sum_{k=0}^{\infty} a_k \cos\frac{2\pi k t}{T}+b_k \sin\frac{2\pi k t}{T}$$

, kde T je perioda signálu. Koeficienty a_k a b_k lze získat ze vztahu

$$a_0=\frac{1}{T}\int_0^T v(t)\,dt$$
$$a_k=\frac{2}{T}\int_0^T v(t)\cos\frac{2\pi k t}{T}\,dt$$
$$b_k=\frac{2}{T}\int_0^T v(t)\sin\frac{2\pi k t}{T}\,dt$$

koeficient b_0 je vždy nulový. Toto je předpis pro nekonečnou řadu a aby tato konvergovala, musí platit, že absolutní hodnota koeficientů a_k a b_k se blíží k nule, pokud jdeme s k do nekonečna. Jinými slovy musí existovat takové K, že pro k>K jsou hodnoty koeficientů a_k a b_k zanedbatelné. Původní předpis se změní do podoby

$$v(t)\approx\sum_{k=1}^{K} a_k \cos\frac{2\pi k t}{T}+b_k \sin\frac{2\pi k t}{T}$$

To znamená, že původní funkci lze aproximovat konečnou řadou.

Základní frekvence f_0 je definována jako převrácená hodnota periody. Za použití této frekvence můžeme přepsat tvar Fourierovy řady jako

$$v(t)=\sum_{k=0}^{\infty} a_k \cos(2\pi k f_0 t)+b_k \sin(2\pi k f_0 t)$$

Je zřejmé, že sinusoidy, které tvoří Fourierovu řadu, mají frekvenci, jež je celočíselným násobkem základní frekvence. Tyto frekvence jsou známé jako harmonické frekvence.

7.2. Získání Fourierovy řady

Pro získání takového generátoru jsem z naměřených dat vytvořil dva vektory hodnot, jeden pro vstup, jeden pro výstup. Tato data jsem dále zpracovával v nástroji 'cftool' programu MATLAB, který využívá pro nalezení koeficientů řady metodu nejmenších čtverců.

7.3. Metoda nejmenších čtverců

Tato metoda vychází z myšlenky, že chceme aproximovat funkci, která je zadána tabulkou, v níž se nacházejí hodnoty x a y [12]. Zvolíme tedy aproximaci ve tvaru

$$\varphi(x)=c_0\varphi_0(x)+..+c_m\varphi_m(x)$$

kde φ_i jsou zadané bázové funkce a c_i jsou hledané parametry.
Naším cílem je tedy minimalizovat odchylku funkce φ od naměřených dat. Aproximace metodou nejmenších čtverců musí splňovat podmínku, že norma chyby je minimální, tedy

$$R(f,\varphi)=\sum_{i=0}^{n}\left[f(x_i)-\sum_{j=0}^{m}cj\,\varphi_j(x_i)\right]^2$$

je minimální. Snažíme se tedy minimalizovat funkci $R(f,\varphi)$. Nutná a postačující podmínka minima je

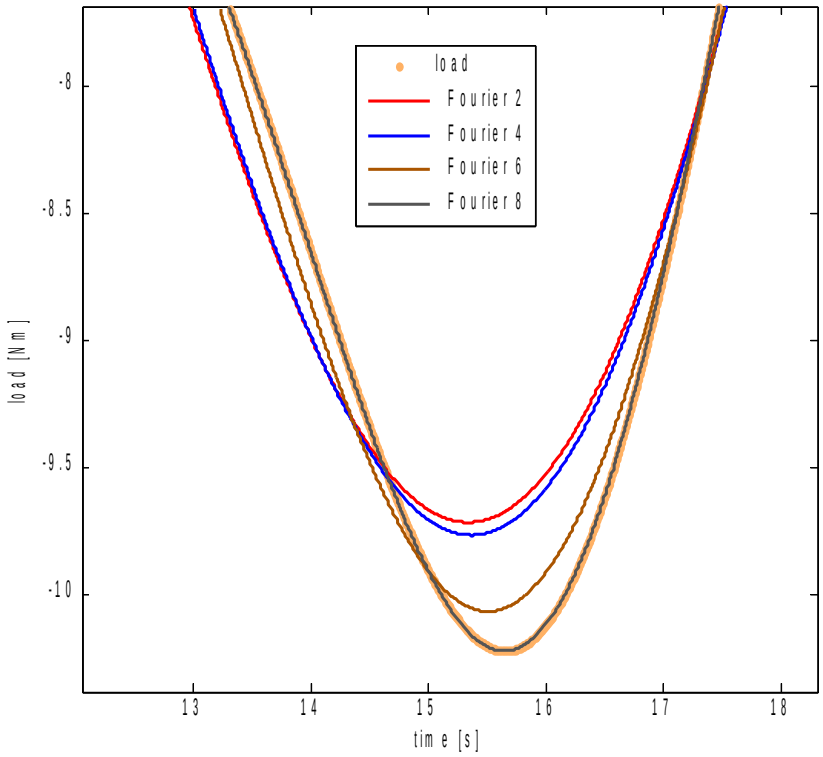
$$\frac{\partial R}{\partial c_i}=0,\forall c_i$$

Koeficienty c_i nalezneme jako řešení soustav takto získaných rovnic.

7.4. Aproximace vstupních dat

Jako první krok bylo nutné zvolit řád Fourierovy řady. V nabídce tohoto nástroje je možné vybrat 1. až 8. řád. Kvalita aproximace těmito řadami je znázorněna níže (Obr. 14). V grafu lze pozorovat, že až nejvyšší nabízený řad rozvoje nabízí dostatečnou aproximaci k působící síle. Za indikátor kvality aproximace jsem považoval hodnotu SSE {suma kvadrátů chyby - sum of squared error}. V následující tabulce se nachází srovnání jednotlivých aproximací a jejich SSE hodnot. Model Fourier 8 lze považovat za přijatelnou aproximaci pro generování působícího momentu.

Aproximace	SSE
Fourier 2	22504.83
Fourier 4	1641.97
Fourier 6	474.91
Fourier 8	68.56



Obr. 14: Porovnání Fourierových řad řádu 2,4,6,8

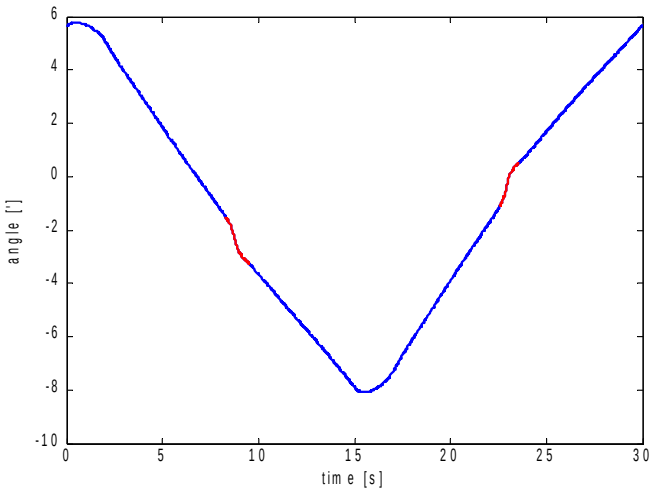
7.5. Aproximace výstupních dat

Obdoný postup byl použit při aproximaci výstupních dat. Kvůli průběhu křivky, kde zaznamenáváme na intervalech $t \in \langle 8.5, 9 \rangle \cup \langle 22.8, 23.3 \rangle$ (Obr. 15) podsatně rychlejší změnu úhlu deformace, byla hodnota SSE podstatně vyšší, než při zpracovávání vstupního signálu. Aproximace v místě minima (Obr. 16) a na kritickém intervalu (Obr. 17) je přiložena níže.

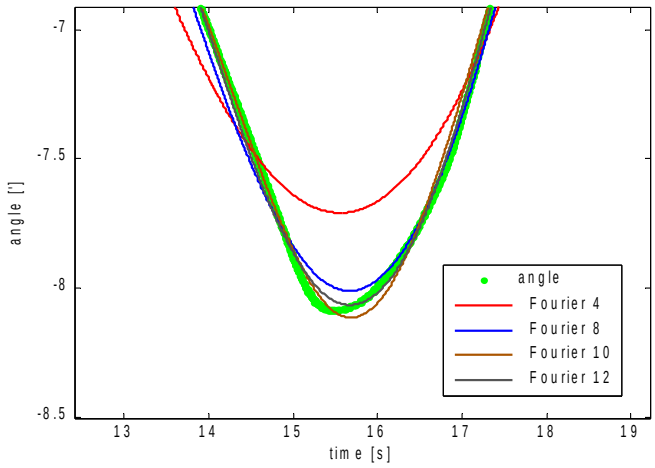
Hodnoty SSE:

Aproximace	SSE
Fourier 4	1009.87
Fourier 8	168.23
Fourier 10	87.42
Fourier 12	67.64

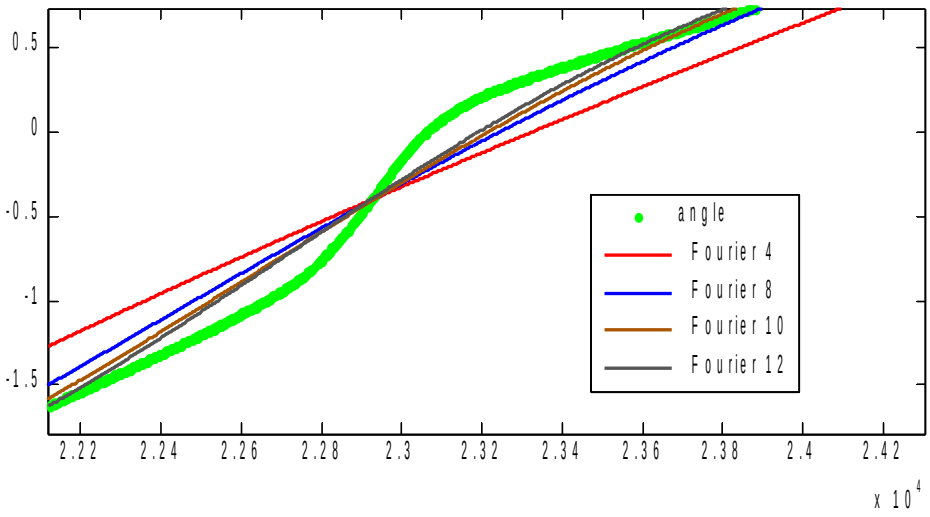
V tomto případě byla i chyba nejvyššího nabízeného řádu (osmého) stále dost vysoká, proto bylo nutné nadefinovat řády 10 a 12 jako vlastní rovnice.



Obr. 15: Kritické intervaly



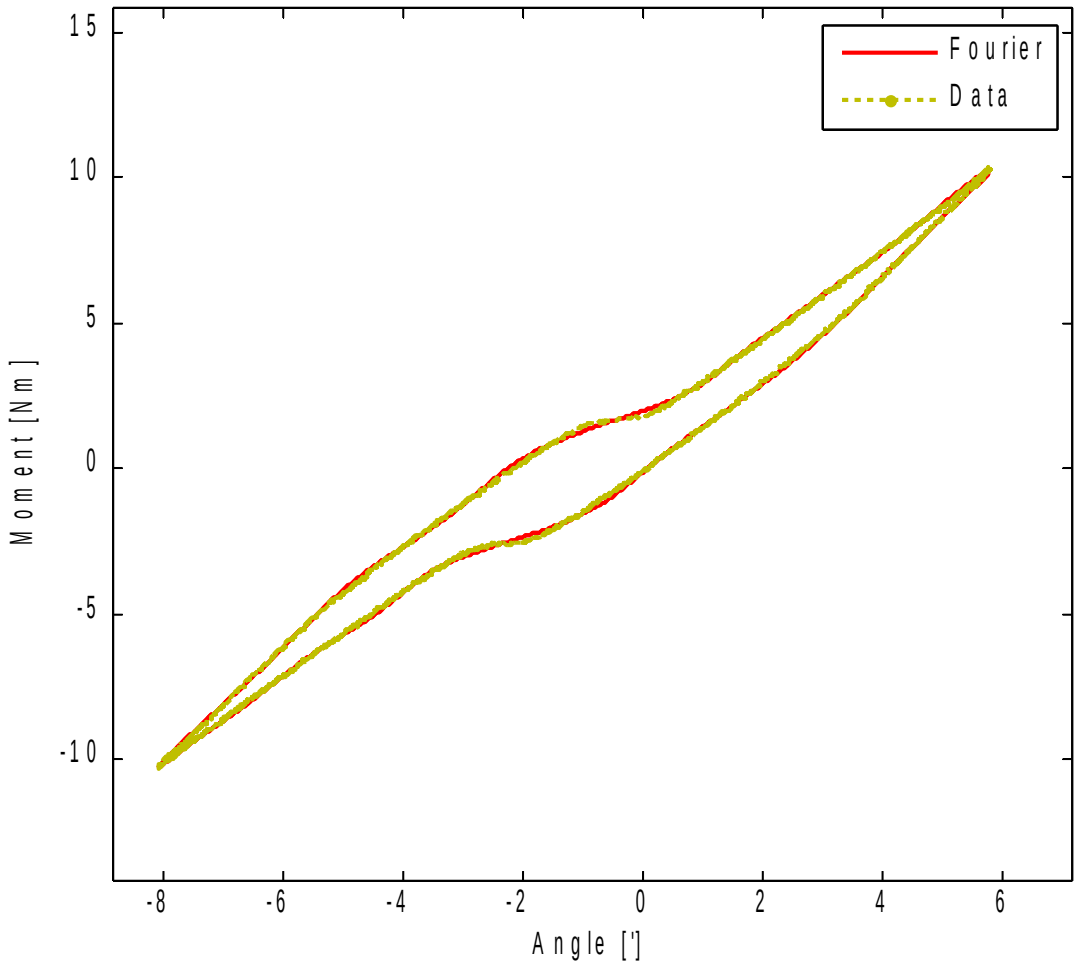
Obr. 16: Porovnání Fourierových řad v místě minima



Obr. 17: Porovnání Fourierových řad na kritickém intervalu

7.6. Výsledná aproximace

Výstup obou generátorů se na většině průběhu překrývá s původním signálem. (Obr 18.) Proto lze získané rovnice považovat za přijatelný generátor dat.



Obr. 18: Porovnání původního a aproximovaného průběhu síly a deformace

7.7. Implementace získané řady

Takto získané řady bylo nutné v poslední fázi implementovat jako skript pro blok REXLANG (Obr. 19). Generátor času je řešený interním čítačem. Aktuální hodnoty momentu a úhlu jsou vypočtené dosazením hodnoty na čítači do předipsu Fourierových řad pro moment a úhel. Vstupy jsou 'Start','Stop', 'Zdroj' a výstupy: velikost působícího momentu, odchylka v úhlových minutách a čas.

```
int input(0) start;
int input(1) source;
int input(2) stop;

double output(0) x;
double output(1) y;
double output(4) t;

int hold;
double counter;

int init(void)
{
 counter=0;
 hold=0;
 return 0;
}

int main(void){
  if(source == 3){
      if(stop == 1){hold = 1;}
      if(start == 1 && hold== 0){
        counter = counter + 1;
        x=sin(counter*(8.69118347038255E15/4.611686018427388E18))..
        y=cos(counter*(9.65687052258695E14/4.611686018427388E18))..
      }
       if(start == 0){ hold = 0; }
       if(hold == 1 || start == 0) {
          x=0;
          y=0;
          counter= 0;
       }
       t=counter;
  }
      return 0;
}
```

Obr. 19: Generátor signálu v REXLANGu

8. Matematický model hystereze

Další možností jak získat generátor dat bylo nalézt k úloze odpovídající matematický model. Jako zdroj při hledání byl použit přehled v praxi používaných modelů [13]. Zmíněny byly například hysterony, Preisachův model či Bouc-Wenův model hystereze. Po srovnání možností realizace a optimalizace jednotlivých modelů se jevil jako nejvhodnější model Bouc-Wen.

8.1. Bouc-Wen model hystereze

Bouc-Wen model hystereze nabízí působivou matematickou jednoduchost, přesto je díky velkému množství parametrů použitelný na velké skupině jevů, od mechanických deformací až po magnetické jevy [13].

Máme $T=[t_{0,}t]\in R$, stavy $x(t), z(t): T\to R$, vektorovou funkci $f:(R^m, R^m, R, R)\to R^m$ a vstup $u(t): T\to R$, Bouc-Wen model je pak popsán

$$\ddot{x}(t)=f(x,\dot{x},z,u), \qquad \ddot{x}(t_0)=x_{00,}$$
$$\dot{z}=A\dot{x}-\beta\dot{x}|z|^n-\gamma|\dot{x}||z|^{n-1}z, \qquad z(t_0)=z_0$$

Zde x a z znamenají pozici či deformaci tělesa a hysterezní sílu. Reálné parametry n, A, β, γ řídí velikost a tvar hysterezní křivky. Protože je z nefyzikální veličina, její použití jako výstupní veličiny je možné pouze z matematického hlediska. V praxi ovšem může být dopočítávána z reálných výstupních veličin systému.

8.2. Implementace Bouc-Wenova modelu

Jako první pro implementaci byl zvolen upravený Bouc-Wenův model, jak ho popsal ve své disertaci Christian Heine [14]. Ten popisuje rovnici pohybu oscilujícího (či namáhaného) objektu takto:

$$\ddot{u}(t)+2\cdot\xi_0\cdot\omega\cdot\dot{u}(t)+H(u,z,t)=f(t)$$

kde je celková obnovující síla H, složená z lineární a obnovující hysterezní síly, popsána:

$$H(u,z,t)=f_R(u,t)+f_h(z,t)=\alpha\cdot\omega^2\cdot u(t)+(1-\alpha)\cdot\omega^2\cdot z(t)$$

Na základě Bouc-Wenova modelu je rovnice pro z vyjádřena:

$$\dot{z}(t)=A\cdot\dot{u}(t)-(\beta\cdot|\dot{u}(t)|\cdot|z(t)|^{n-1}\cdot z(t)+\gamma\cdot\dot{u}(t)\cdot|z(t)|^n)$$

V takto zadaných rovnicích bylo nutné určit parametry $\xi, \omega, \alpha, A, \beta, \gamma, n$. Pro tyto účely byl použit 'System identification toolbox' pro určování matematických modelů z naměřených dat, konkrétně oddíl pro určování parametrů uživatelsky definovaných modelů, sestavených z nelineárních diferenciálních rovnic. Takzvaných 'Grey boxů'.

Výše uvedené rovnice byly přepsány do stavového popisu chování systému s hysterezí, se stavy: $x_1(t)=u(t), x_2(t)=\dot{u}(t), x_3(t)=z(t)$ a vstupem $u(t)=f(t)$. Stavový popis ve tvaru použitelném v modelu 'Grey box':

$$\begin{bmatrix}\dot{x_1}\\ \dot{x_2}\\ \dot{x_3}\end{bmatrix}=\begin{bmatrix}x_2\\ u-2\xi\omega x_2-\alpha\omega^2 x_1-(1-\alpha)\omega^2 x_3\\ A x_2-(\beta|x_2||x_3|^{n-1}x_3+\gamma x_2|x_3|^n)\end{bmatrix}$$

Výstupní rovnice je pak $y=x_1$

Tato soustava rovnic byla dle vzoru pro vytváření modelů typu Grey box implementována do funkce v MATLAB. (Obr. 20) Tato metoda umožňuje vytvořit stavový popis systému s volnými parametry jako funkci, kde vstupními argumenty jsou stav, vstup a parametry, na výstupu se objeví výstup a derivace vstupu.

```
function [ dx, y ] = BoucWen(~,x,u, xi, omega, alpha, A, beta,
gamma, n,k, varargin)
%funkce pro realizaci Bouc-Wen modelu hystereze
y=x(1);
dx = [
    x(2);
    u(1)*k-2*xi*omega*x(2)-alpha*(omega^2)*x(1)-(1-
alpha)*(omega^2)*x(3);
    A*x(2)-(beta*abs(x(2))*(abs(x(3))^(n-
1))*x(3)+gamma*x(2)*abs(x(3))^n)
 ];
end
```

Obr. 20: Zápis modelu Grey box jako funkce v MATLAB

Pro určení parametrů byla použita metoda PEM, která je součástí knihovny programu MATLAB pro identifikaci nelineárních systémů.

8.3. Metoda pro určení parametrů PEM

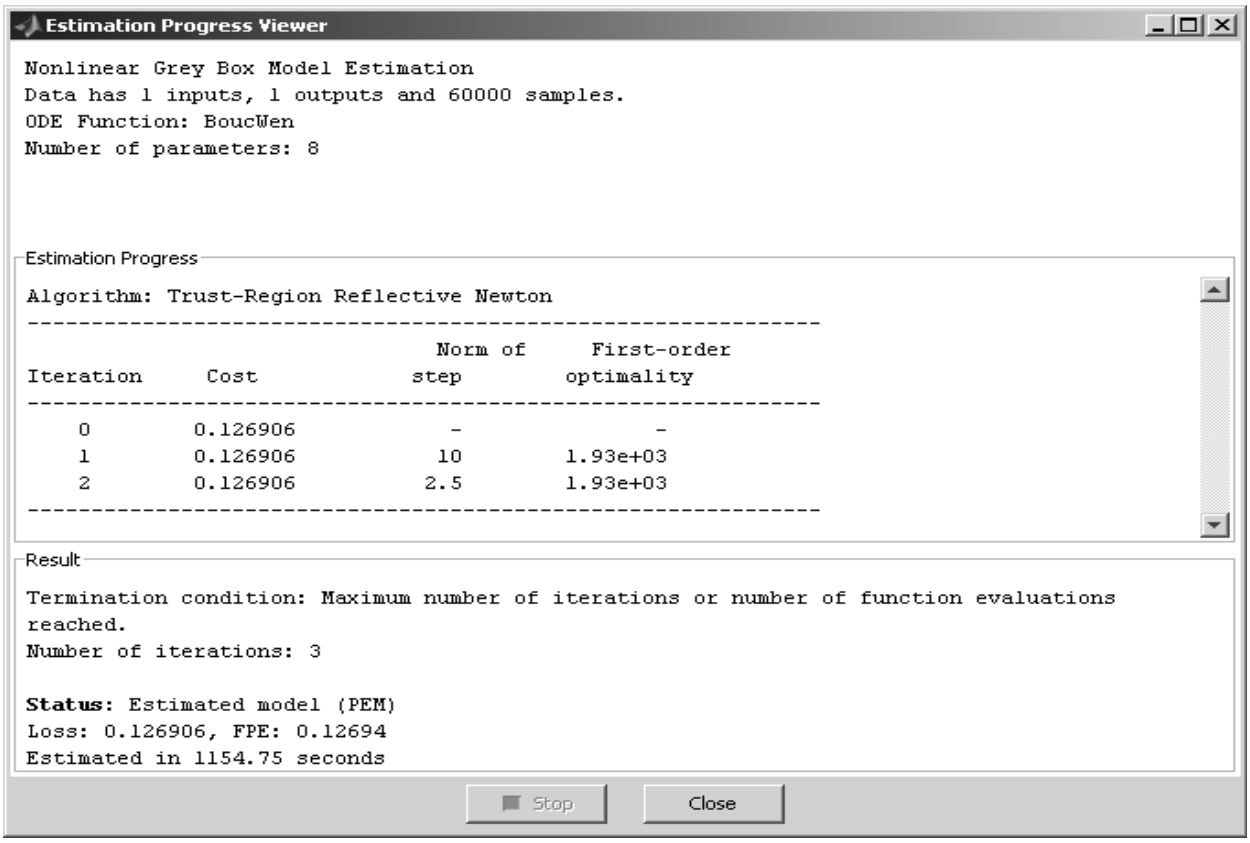
PEM {metoda odhadu chyby - Prediction Error Method} narozdíl od metody nejmenších čtverců pracuje s přesností predikce, vypočtené z reálných vstupních dat a snaží se minimalizovat odchylku výpočtu od naměřených výstupních dat. [15] Její implementace v matlabu využívá numerickou optimalizaci ke snížení nákladové funkce [16].

Ve všech případech optimalizace jsem pro určení parametrů využíval dvě periody výstupního signálu. Za vstupní signál jsem volil aproximaci vstupního signálu Fourierovou řadou, získanou z aproximace vstupních dat. Tento postup jsem zvolil, aby bylo možné vytvořit generátor vstupu pro získaný model.

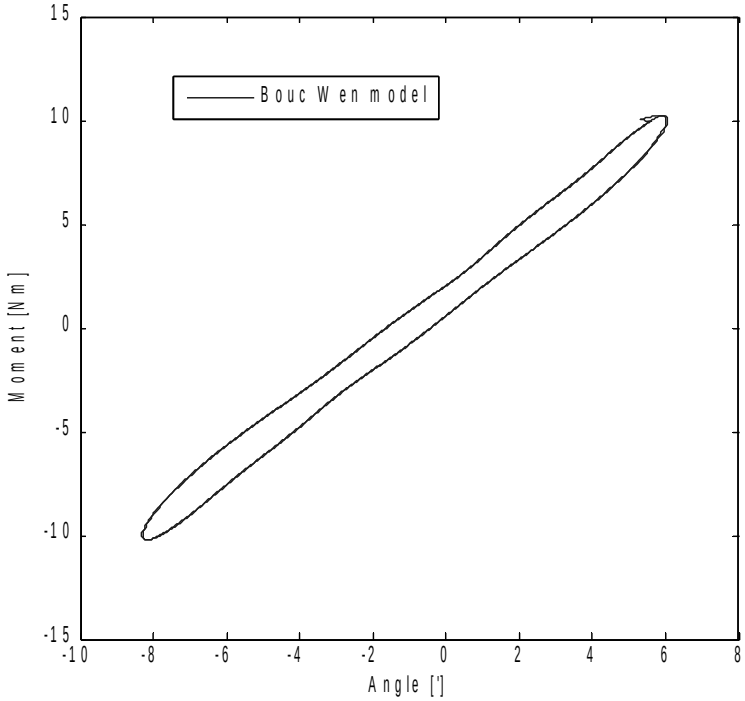
8.4. Výsledek optimalizace Bouc-Wenova modelu

Po několika stovkách iterací metody PEM, klesla hodnota ztrátové funkce na 0.126906 (Obr. 21)**.** Výsledný model (Obr 22.) neodpovídal hledanému modelu. Z nalezených parametrů vyplývá, že vliv hystereze na celý průběh je velmi malý. Tento poznatek odpovídá skutečnému průběhu

deformace řídicí tyče. V případě, že by se jednalo o samotnou ocelovou tyč, byl by průběh velikosti deformace na působené torzní síle lineární, za předpokladu, že by jsme sílu dále nezvyšovali a nepřekročili meze dané Hookeovým zákonem. Při úloze s řídicí tyčí se do experimentu dostane fáze, kdy je síla přenášena přes kloubové spojení a nedochází k plastické deformaci oceli. Tento jev zanáší do experimentu hysterezi, která se ovšem projeví pouze na krátkém časovém úseku. Z tohoto důvodu dochází při optimalizaci parametrů ke zkreslení a výsledný model neodpovídá hledanému.



Obr. 21: Průběh odhadu parametrů



Obr. 22: Výsledek simulace Bouc-Wen modelu

8.5. Další uvažované modely hystereze

Přestože byla hledání vhodného modelu hystereze věnována značná část času vymezeného této práci, nalezení použitelného modelu pro simulaci průběhu experimentu přesahovalo její rámec. V průběhu vytváření matematického modelu byly zvažovány například tyto modely: Bouc-Wen-Baarber-Noori hysterezní model [14] používaný pro modelování pokročilých mechanických jevů jako je degradace tuhosti, degradace elasticity či povolování materiálu. Tento model však přinesl jen minimální zlepšení výsledků simulace. Jako užitečná se ukázala práce popisující super-elasticitu NiTi šroubovice [17]. Ačkoliv výsledky práce s tímto modelem byly doposud nejlepší a optimalizace tohoto modelu po prvních experimentech přinesla zajímavé výsledky, nepodařilo se dalšími výpočty ztotožnit průběh simulovaného procesu s naměřenými hodnotami.

9. Závěr

Cílem práce bylo vytvořit vizualizační rozhraní k průmyslovému procesu. Tohoto cíle bylo dosaženo a výsledná práce splňuje všechny zadané požadavky. Data jsou generována na simulačním schématu, vytvořeném v řídícím systému REX. Přenos těchto dat zajišťuje rozhraní

WebSocket. Výsledná vizualizace je zobrazitelná na základní instalaci prohlížečů Chrome a Firefox a využívá možnosti HTML5, SVG a jazyku JavaScript. Je tedy splněn i požadavek na tenkého klienta, což mimo jiné zaručuje i využití na různých platformách počínaje stolními počítači, přes tablety až po chytré telefony s webovým prohlížečem. Numerické generování dat je vytvořeno prostřednictvým aproximace vstupních a výstuních dat Fourierovými řadami. Doplňující požadavek na matematický model hystereze se v průběhu práce ukázal jako komplikovanější, než se při předběžném zkoumání zdálo a jeho vyřešení přesahuje rámce této práce.

Zdroje

[1] Refsnes Data. HTML5 Introduction. *W3Schools Online Web Tutorial* [on-line]. Refsnes Data, © 1999-2013 [cit. 17.4.2013]
Dostupné z: http://www.w3schools.com/html/html5_intro.asp

[2] Manos Papagelis. WEB APP ARCHITECTURES: MULTI-TIER (2-TIER, 3-TIER) & MVC. In: *Computer Science UNIVERSITY OF TORONTO* [on-line]. August 2012 [cit. 13.4.2013]. Dostupné z : http://queens.db.toronto.edu/~papaggel/courses/csc309/docs/lectures/web-architectures.pdf

[3] REX Controls. Řídicí systém REX. *REX Controls* [on-line]. REX Controls s.r.o., © 2000-2013 [cit. 13.4.2013]
Dostupné z: http://www.rexcontrols.cz/rex

[4] REX Controls. Funkční bloky systému REX, Referenční příručka.In: *REX Controls* [on-line]. Plzeň, 10.4.2013 [cit. 14.4.2013]
Dostupné z: http://www.rexcontrols.cz/media/documents/manuals/cz/BRef_CZ.pdf

[5] MathWorks. Text Files – MATLAB & Simulink. *MathWorks – MATLAB & Simulink* [on-line]. The MathWorks, Inc, © 1994-2013 [cit. 15.4.2013]
Dostupné z: http://www.mathworks.com/help/matlab/text-files.html

[6] Kaazing Corporation. What is WebSocket?. *WebSocket.org - A WebSocket community* [on-line]. Kaazing Corporation, ©2013 [cit. 20.4.2013]
Dostupné z: http://www.websocket.org/index.html

[7] Oracle Corporation. NetBeans IDE 7.3 Release Information. *NetBeans IDE* [on-line]. Oracle Corporation, ©2013 [cit. 16.4.2013]
Dostupné z: https://netbeans.org/community/releases/73/

[8] REX Controls. Vývojové prostředí systému REX. *REX Controls* [on-line]. REX Controls s.r.o., © 2000-2013 [cit. 3.4.2013]
Dostupné z: http://www.rexcontrols.cz/rex

[9] The jQuery Foundation. jQuery API Documentation. jQ*uery* [on-line]. The jQuery Foundation, ©2013 [cit. 13.4.2013]
Dostupné z: http://api.jquery.com/

[10] The jQuery Foundation. jQuery UI API Documentation. jQ*uery UI* [on-line]. The jQuery Foundation, ©2013 [cit. 12.4.2013]
Dostupné z: http://api.jqueryui.com/

[11] Kundert Kenneth S. Introduction to the Fourier Series. In: *The Designer's Guide Community* [on-line]. 31 October 2010 [cit. 10.4.2013]
Dostupné z: http://www.designers-guide.org/theory/fourier.pdf

[12] Daněk Josef. Numerické metody. In: *trial.zcu.cz* [on-line]. 13.2.2013 [cit. 14.4.2013]
Dostupné z: http://trial.zcu.cz/?page=predmet&volba=TH&tlink=12-080-040&vlink=8.4.

[13] Sain P. M., Sain M. K. a Spencer B. F. Models for Hysteresis and Application to Structural Control. *Proceedings American Control Conference*, June 1997

[14] Heine Christian P. Simulated Response of Degrading Hysteretic Joints With Slack Behavior. *Wood Science and Forest Products* [on-line]. URN etd-08092001-100756. [cit. 8.4.2013]
Dostupné z: http://scholar.lib.vt.edu/theses/available/etd-08092001-100756/Ch5.pdf

[15] Pelckmans Kristiaan. System Identification. *Uppsala Universitet* [on-line]. 16 May 2012 [cit. 7.4.2013]
Dostupné z: http://www.it.uu.se/edu/course/homepage/systemid/vt12/ch6.pdf

[16] MathWorks. Prediction error estimate for linear or nonlinear model - MATLAB pem. *MathWorks – MATLAB & Simulink* [on-line]. The MathWorks, Inc, © 1994-2013 [cit. 15.4.2013]
Dostupné z: http://www.mathworks.com/help/ident/ref/pem.html

[17] Awrejcewicz Jan, Dzyubak Larisa, Lamarque Claude-Henri. Modelling of hysteresis using Masing–Bouc-Wen's framework and search of conditions for the chaotic responses.*Technical University of Łódź, National Technical University "Kharkov Polytechnic Institute",Ecole Nationale des Travaux Public de l'Etat*. 30 October 2006 [cit. 20.4.]
Dostupné z: http://test.abm.p.lodz.pl/awrejcewicz/publikacje/publ_pdf/PC221.pdf