VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV MECHANIKY TĚLES, MECHATRONIKY A BIOMECHANIKY

FACULTY OF MECHANICAL ENGINEERING
INSTITUTE OF SOLID MECHANICS, MECHATRONICS AND BIOMECHANICS

NÁVRH A REALIZACE APLIKACE PRO ZPRACOVÁNÍ MĚŘENÝCH DAT Z MOTOROVÉHO STANDU

DESIGN AND IMPLEMENTATION OF APPLICATION FOR MEASURED DATA PROCESSING

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE — JOZEF BLAHÚT
AUTHOR

VEDOUCÍ PRÁCE — ING. JIŘÍ KOVÁŘ
SUPERVISOR

BRNO 2014

Vysoké učení technické v Brně, Fakulta strojního inženýrství

Ústav mechaniky těles, mechatroniky a biomechaniky
Akademický rok: 2013/14

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

student(ka): Jozef Blahút

který/která studuje v **bakalářském studijním programu**

obor: **Mechatronika (3906R001)**

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma bakalářské práce:

Návrh a realizace aplikace pro zpracování měřených dat z motorového standu

v anglickém jazyce:

Design and implementation of application for measured data processing

Stručná charakteristika problematiky úkolu:

Cílem práce je komplexní návrh a realizace aplikace pro zpracování naměřených dat na standu leteckých motorů. Aplikace musí být spustitelná na platformách PC se systémem Microsoft Windows a musí umožňovat různé režimy zobrazení naměřených dat.

Cíle bakalářské práce:

1. Prostudujte problematiku tvorby aplikací na Microsoft Windows.
2. Navrhněte architekturu aplikace pro zpracování dat.
3. Daný návrh realizujte v prostředí .NET Framework, C#
4. Aplikace musí umožňovat různé režimy zobrazení měřených dat, práci s osami a různé metody vykreslování os s přepočítaným rozsahem.
5. Implementujte výpočet základních statistických funkcí pro vybranou oblast grafu nebo celý graf (průměr, střední hodnota, plovoucí průměr, výpočet trendu, interpolaci různého druhu, atd)
6. Aplikace musí mít modulární architekturu pro snadnou rozšiřitelnost.

Seznam odborné literatury:

[1] Troelsen, A. Pro C# 2010 and the .NET 4 Platform, 2010, ISBN: 1430225491
[2] Ben-Gan, I. Microsoft® SQL Server® 2008 T-SQL Fundamentals (PRO-Developer), 2008, ISBN: 8178531046
[3] A. D. Kshemkalyani,M. Singhal, Distributed Computing: Principles, Algorithms, and Systems, Cambridge University Press, 2008
[4] B. Albahari, P. Drayton, B. Merrill, C# Essentials, Cambridge University Press, O'Reilly, 2008
[5] S. Cheng, Microsoft Windows Communication Foundation 4.0 Cookbook for Developing SOA, Packt Publishing, 2010

Vedoucí bakalářské práce: Ing. Jiří Kovář

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2013/14.

V Brně, dne 21.11.2013

L.S.

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ FAKULTA STROJNÍHO INŽENÝRSTVÍ DĚKANÁT

prof. Ing. Jindřich Petruška, CSc.
Ředitel ústavu

prof. RNDr. Miroslav Doupovec, CSc., dr. h. c.
Děkan

ABSTRAKT

Táto bakalárska práca sa zaoberá vývojom desktopovej aplikácie pre spracovanie meraných dát zo standu leteckých motorov. Jej hlavným cieľom je zjednodušiť spracovávanie nameraných údajov, ich vyhodnotenie a vzájomné porovnávanie. V úvode práce je rozobraná problematika tvorby aplikácií na Microsoft Windows. Následne je popísaná architektúra aplikácie, jej realizácia a v závere sú uvedené možnosti práce so získanými dátami.

KĽÚČOVÉ SLOVÁ

C#, C sharp, .NET, .NET Framework, zpracování dat, GUI, architektura

ABSTRACT

This bachelor thesis deals with the development of desktop application for the processing of measured data from aircraft engine stand. Its main aim is to simplify the analysis of obtained data, their evaluation and correlation. At the beginning of the work an issue of making applications on Microsoft Windows is discussed, followed by description of application´s architecture, its implementation and the conclusion deals with options for working with acquired data.

KEYWORDS

C#, C sharp, .NET, .NET Framework, data processing, GUI, architecture

BIBLIOGRAFICKÁ CITACE

BLAHÚT, J. *Návrh a realizace aplikace pro zpracování měřených dat z motorového standu.* Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2014. 29 s. Vedoucí bakalářské práce Ing. Jiří Kovář.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma Návrh a realizace aplikace pro zpracování měřených dat z motorového standu jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením tohoto projektu jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne 29. 5. 2014 …….

podpis autora

POĎAKOVANIE

Týmto by som sa chcel poďakovať vedúcemu bakalárskej práce Ing. Jiřímu Kovářovi za účinnú metodickú, pedagogickú a odbornú pomoc a ďalšie cenné rady pri spracovaní mojej bakalárskej práce. Ďalej ďakujem Ing. Lukášovi Ertlovi z firmy Unis a.s. za možnosť vypracovania bakalárskej práce na danú tému a za odborné vedenie a cenné konzultácie.

V Brně dne 29. 5. 2014 …….

podpis autora

OBSAH

ÚVOD

Táto práca je založená na dlhodobej spolupráci s firmou Unis a.s. a na konkrétnych požiadavkách s cieľom vyvinúť aplikáciu určenú pre spracovávanie dát meraných na stande leteckých motorov. Účelom vyvíjanej aplikácie je zjednodušiť spracovávanie nameraných údajov do prehľadnej grafickej podoby, umožniť používateľovi vyhodnotiť jednotlivé veličiny a zároveň ich porovnávať medzi sebou pomocou štatistických funkcií.

Tento dokument má taktiež slúžiť ako zhrnutie výsledkov jednotlivých konzultácií ohľadom vývoja aplikácie počas celého trvania pracovného pomeru s firmou Unis a.s.

Aplikácia bude vychádzať z konceptu desktopovej aplikácie pre operačné systémy rodiny Windows. Bude založená na myšlienke formulárovej aplikácie, ktorá bude vedieť spracovávať súbory z merania, graficky ich interpretovať a editovať s následným využitím štatistických funkcií pre vzájomné porovnávanie.

V tejto práci sa prvotne budem zaoberať problematikou tvorby aplikácií na platforme Microsoft Windows, kde rozoberiem aktuálne možnosti a nástroje pre tvorbu aplikácií. V jednotlivých podkapitolách rozoberiem typy programovacích jazykov, dostupné vývojové prostredia, grafické knižnice pre vizualizáciu dát ako aj matematické knižnice pre spracovanie meraných údajov.

V druhej časti postupne rozoberiem celkový návrh aplikácie z hľadiska výberu vývojových prostriedkov, návrhu modulárnej architektúry, použitých knižníc a bezpečnosti programu.

Tretia časť objasní realizáciu celkového návrhu s dôrazom na dizajn grafického užívateľského prostredia, rôzne režimy zobrazenia meraných dát a výpočet základných štatistických funkcií.

1 PROBLEMATIKA TVORBY APLIKÁCIÍ NA MICROSOFT WINDOWS

Návrh aplikácie na platforme Windows zahŕňa radu špecifických problémov, ktoré je nutné preštudovať pred samostatným písaním kódu aplikácie. Ide o veľmi dôležitú prípravu, nakoľko v prípade zlého výberu jednej z častí sa môžeme dostať do stavu, kedy nebudeme vedieť daný problém vyriešiť s vybranými prostriedkami a budeme sa musieť vrátiť na úplný začiatok. Strata drahocenného času a predĺženie vývoju aplikácie preto nie je zdroj, ktorým môžeme plytvať. V nasledujúcich podkapitolách si rozoberieme popis platformy .NET Framework a jednotlivé problematiky pri tvorbe aplikácií v tomto prostredí.

1.1 .NET Framework

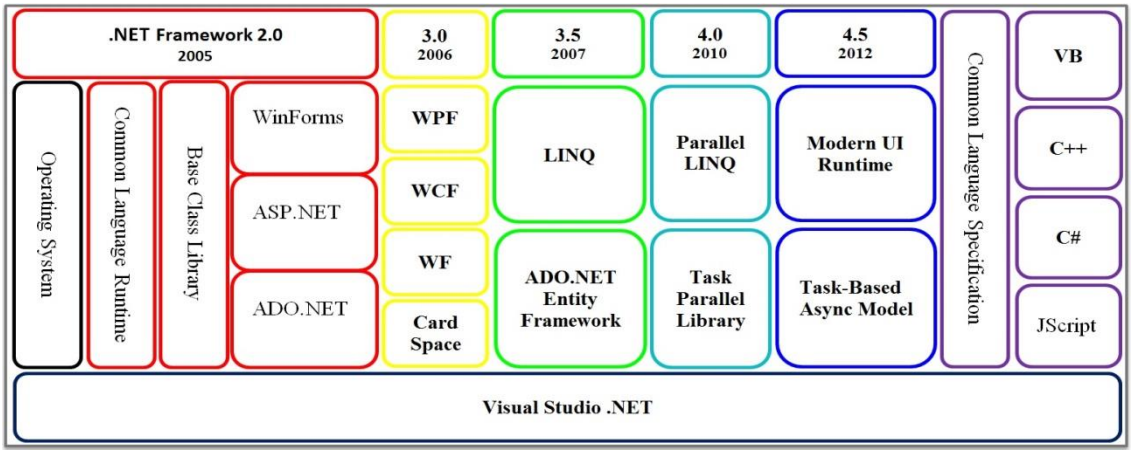
Jedná sa o najrozšírenejšiu vývojovú platformu pre Microsoft Windows. Počin od firmy Microsoft, vyvinutý v roku 2002, zoskupuje kolekciu technológií v softvérových produktoch určených pre RAD. Je to vízia ako by mal byť softvér písaný a sada nástrojov pre vývoj softvéru, ktoré túto víziu zrealizujú. [4]

1.1.1 História

Všetky verzie operačného systému Windows od verzie 3.1 až po Windows Server 2008 obsahujú rozhranie Windows API, ktoré sa s novou verziou OS nenahradzovalo, ale rozširovalo. To isté je možné povedať o ďalších technológiách a platformách používaných pri vývoji softvéru pre systém Windows. Spôsob rozširovania existujúceho rozhrania API namiesto jeho úplne novej verzie bol evolučný prístup k softvéru od spoločnosti Microsoft, ktoré zabezpečovalo spätnú kompatibilitu. Tento aspekt umožňoval v priebehu rokov iným dodávateľom vyvinúť rozsiahlu programovú základňu pre systém Windows, čo by inak bolo s použitím novej technológie v každej novej verzii OS nemožné. Avšak bola tu aj obrovská nevýhoda - integrácia novej technológie komplikovala celé rozhranie. A tak sa zrodila myšlienka, že sa musí niečo zmeniť. Touto zmenou bolo predstavenie technológie .NET Framework. [2]

1.1.2 Popis .NET Framework

Základnú architektúru platformy .NET Framework tvorí jej verzia 2.0, ktorá pozostáva z CLR(Common Language Runtime), BCL(Base Class Library), WinForms, ASP.NET, ADO.NET, CLS(Common Language Specifications) a Visual Studio .NET. Detailne zobrazená technologická architektúra .NET Framework je na obrázku 2.2. [3][4]



Obrázok 1 Prehľad platformy .NET Framework

Zdroj: prevzaté z [3][4]

Jadrom platformy sú bloky CLR a BCL. Tieto dve komponenty poskytujú exekučné prostredie a programovacie rozhranie pre vývoj .NET aplikácií. Aplikácie skompilované pre .NET Framework obsahujú medzivrstvu, tzv. MSIL(Microsoft Intermediate Language). Táto medzivrstva zaisťuje, že pri prvotnom spustení je kód MSIL preložený pomocou modulu CLR a jeho súčasti JIT(Just-In-Time compiler) do strojového kódu použitej platformy. [3] Okrem toho je CLR zodpovedný za správu exekučného prostredia .NET (tzv. riadeného kódu), vrátane alokácie pamäti a riadenie pracovných vlákien, rovnako ako vynútenie bezpečnostných pravidiel a taktiež správu garbage collector-a, ktorý sa automaticky stará o úniky pamäti. Všetky tieto činnosti sú automatické, čím sa skracuje doba vývoja a ostatné úlohy vývoja sú podstatne zjednodušené. [4]

Blok ASP.NET poskytuje pokročilú verziu Active Server Pages(ASP) určených pre vývoj webových aplikácií, blok ADO.NET pristupuje k dátam ActiveX Data Object(ADO), t.j. databázam, blok WinForms umožňuje tvorbu GUI a Visual Studio .NET je nástroj, ktorý umožňuje využiť všetky vlastnosti .NET Framework.

1.1.3 Výhody technológie .NET Framework

.NET Framework až po poslednú aktuálnu verziu č. 4.5 obsahuje množstvo implementovaných technológií, čo značne rozširuje jeho možnosti a taktiež vytvára rozsiahly počet výhod. Medzi základné prednosti patrí [2]:

- Objektovo orientované programovanie
- Knižnica základných tried
- Jazyková nezávislosť
- Podpora dynamických webových stránok
- Účinný prístup k dátam
- Zdieľanie kódu
- Vylepšené zabezpečenie
- Inštalácia s nulovým dopadom
- Podpora webových služieb
- Visual Studio (najnovšie verzia 2013)
- Programovací jazyk C#

1.2 Výber programovacieho jazyka

Pre naprogramovanie windows aplikácie máme v dnešnej dobe dostupných viacero programovacích jazykov kompatibilných s platformou .NET Framework, pričom každý z nich ponúka iné výhody aj nevýhody. Výber vhodného programovacieho jazyka závisí od skúseností samotného používateľa a konkrétneho zamerania aplikácie. Pri konkrétnom výbere zvažujeme najmä [1]:

- skúsenosti
  - jedna z najpodstatnejších vlastností pri výbere, avšak nepodstatná, ak nemáme predchádzajúce poznatky
- špecifickú platformu
  - dôležitý faktor, ktorý určuje, na ktorej platforme bude program používaný: Windows, Linux alebo Mac
- elasticitu jazyka
  - určuje náročnosť jazyka na vkladanie nových funkcií do už existujúceho kódu
  - túto vlastnosť môžeme jednoducho zistiť, ak si položíme tieto tri otázky:
    - Môžem začať používať novú funkciu bez použitia novej knižnice?
    - Ak nie, je táto funkcia dostupná v knižnici jazyka?
    - Ak nie je dostupná, koľko úsilia ma bude stáť naprogramovať novú funkciu?
- časové hľadisko
  - stanovuje čas potrebný pre naprogramovanie aplikácie do stavu, kedy splňuje zadané požiadavky a pracuje s pôvodným zámerom
- výkon
  - zohľadňuje sa predovšetkým pri aplikáciách nasadzovaných do prostredia, ktoré neponúka veľa priestoru pre škálovanie - typickým príkladom sú prenosné zariadenia
- podporu a komunita
  - akokoľvek dobrý programovací jazyk je, dôležité je, aby mal za sebou komunitu ľudí. Títo ľudia podporujú rozširovanie fóra, pridávanie návodov a to najdôležitejšie, tvorbu prídavný knižníc, ktoré rozširujú možnosti jazyka. Platí, že čím viac ľudí, tým väčšia šanca, že problém, s ktorým sa pri programovaní stretneme, niekto predo mnou už vyriešil.

Výberom toho správneho programovacieho jazyka môžeme vytvárať riešenia, ktoré budú jednoduché, prehľadné, ľahko rozšíriteľné, ľahko dokumentovateľné a jednoducho opraviteľné vzhľadom na požiadavky aplikácie. Avšak najpodstatnejším prvkom pri výbere jazyka je počet riadkov kódu. *Jasná voľba jazyka je tá, ktorá vie spraviť prácu na 10 riadkoch kódu namiesto 20 riadkoch kódu [1]*.

Spoločnosť Microsoft ponúka prekladače pre 4 komerčné jazyky: Visual C# .NET, Visual Basic .NET, Managed Extensions pre C++ a Visual J# .NET. Okrem týchto je podporovaných ešte viac ako 20 programovacích jazykov, ako napríklad Perl, Python, Cobol a iné. [3]

1.3 Programovacie prostredie

IDE (Integrated Development Environment) je programovacie prostredie, ktoré sa typicky skladá z editora kódu (editor), prekladača kódu (compiler), odlaďovača kódu (debugger) a rozhrania pre tvorbu grafického užívateľského prostredia (GUI). Jedná sa však len o základné prvky prostredia, ktoré môže byť rozšírené doplnkami (add-ons) ďalej uľahčujúc prácu. Medzi najznámejšie patria doplnky pre rozšírenú správu, pokročilé formátovanie alebo analýzu kódu. Vzhľadom na kvalitu Visual Studia od spoločnosti Microsoft sa na trhu nachádza pomerne málo programov, ktoré by ho vedeli zastúpiť. Príkladom môže byť SharpDevelop (#Develop) alebo MonoDevelop. Medzi ich výhodu môže byť pre niekoho fakt, že patria pod open-source programy s licenciou LGPL(MonoDevelop len čiastočne) a v prípade MonoDevelop aj multiplatformovou podporou(Linux, OS X). Sú zadarmo a to je ich najväčšia ponúkaná výhoda.

1.4 GUI

GUI(grafické užívateľské prostredie) je typ rozhrania, ktorý dovoľuje užívateľovi interakciu s elektronickým zariadením cez sériu ikon, vizuálnych indikátorov, textových štítkov a iných prvkov. Pri vývoji desktop aplikácií máme na výber z technológií ako je Windows Forms alebo WPF. Pri pohľade na obrázok 2.1 z kapitoly 2.1.2 je zrejmé, že technológia WPF je mladšia ako Windows Forms. WPF je tým pádom novšia, progresívnejšia, umožňuje užívateľovi vytvoriť pokročilejšie grafické prvky a tým pádom aj bohatšie grafické prostredie. Windows Forms zato ťaží z jednoduchosti, širšej podpore zo strany vývojárov vďaka dlhšiemu pôsobeniu a väčším množstvom návodov.

1.5 Knižnice

Široká komunita ľudí, ktorá tvorí nedeliteľnú súčasť platformy .NET Framework, rokmi vytvorila rozsiahlu kolekciu voľne dostupných knižníc s rozličnými funkciami. Práve táto možnosť dovoľuje ušetriť drahocenný čas pri vývoji koncovej aplikácie. Nakoľko táto práca nie je zameraná na vývoj knižníc, budú sa komplikovanejšie funkcie vykonávať práve pomocou voľne dostupných knižníc určených pre .NET Framework. Jedná sa hlavne o:

- vizualizáciu spracovávaných dát
- štatistické funkcie
- výpočet interpolácie

2 ARCHITEKTÚRA APLIKÁCIE UNISENSE

2.1 Označenie aplikácie

Z dôvodu existencie viacerých dokumentov popisujúcich vyvíjanú aplikáciu a zamedzenia nedorozumenia medzi nimi bolo potrebné označiť softvér kódovým označením *UniSense*, aby bolo jasné, že sa jedná o jednu a tú istú aplikáciu.

2.2 Výber programovacích prostriedkov

Pre návrh jednotlivých častí aplikácie UniSense boli vybrané nasledovné programové prostriedky s použitím poznatkov z kapitoly 2. :

- programovací jazyk C#
- platforma .NET Framework 4.0
- vývojové prostredie #Develop (SharpDevelop)
- GUI založené na formulároch Windows Forms
- knižnica ZedGraph pre grafické vykresľovanie dát
- knižnica Math.Net pre výpočet štatistických funkcií

2.3 Návrhový vzor

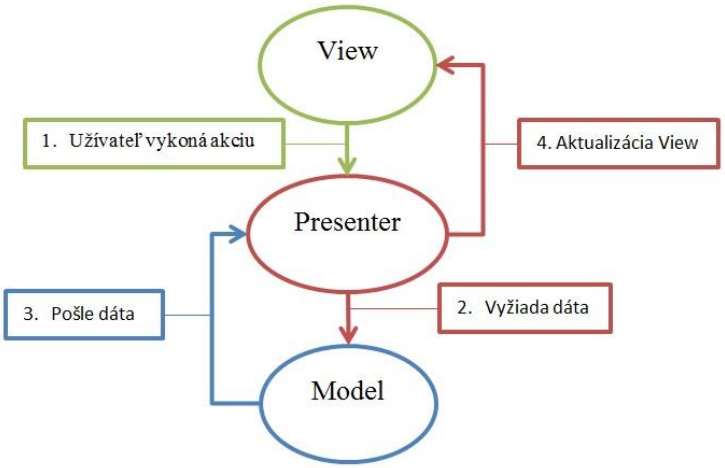
Základným stavebným kameňom celej architektúry aplikácie UniSense je použitie návrhového vzoru (angl. design pattern). Rozhodnutie zužitkovať návrhový vzor splňuje viaceré požiadavky vyplývajúce z charakteru projektu, kde tou najhlavnejšou potrebou je modulárna architektúra pre jednoduchú rozšíriteľnosť. Tohto cieľa dosiahneme aplikovaním návrhového vzoru Model-View-Presenter (ďalej len MVP).

2.3.1 MVP

Model-View-Presenter je trojvrstvová softvérová architektúra, ktorá sa skladá z týchto častí [5]:

1. *Model:*
   Obsahuje informácie, ktoré sa zobrazujú užívateľovi. Aktualizuje sa vždy, keď užívateľ pridá nové informácie.
2. *View:*
   Ako názov hovorí, view je skupina grafických komponentov, v tomto prípade Windows Forms formulárov, ktoré zobrazujú informácie z modelu užívateľovi.
3. *Presenter:*
   Slúži ako medzi člen medzi model a view. Je zodpovedný za poskytovanie dát pre view kedykoľvek je to vyžiadané od užívateľa. Taktiež prijíma a overuje dáta z view, ktoré sú následne predané do modelu k spracovaniu a uloženiu.

Tento návrhový vzor teda využíva výhody princípu separácie záujmov, pričom každá časť vzoru má presne definované zodpovednosti. Ako príklad je možné uviesť prvok ListBox z technológie Windows Forms v časti view, ktorý zodpovedá za zobrazenie dát používateľovi na obrazovke, avšak nerieši už prístup k dátam ani biznis logiku [5]. Obrázok 2 vizuálne reprezentuje návrhový vzor MVP s jasným poradím vykonaných činností pri interakcii užívateľa s view.



Obrázok 2 Vizuálna ilustrácia vzoru MVP

Zdroj: vlastné spracovanie podľa [6]

Výhody:

- nezávislosť prvkov (voľná väzba medzi view a model)
- separácia zodpovedností
- rozšíriteľnosť
- testovateľnosť
- prenositeľnosť kódu
- flexibilita, adaptabilnosť

Nevýhody:

- časovo náročnejšia implementácia

2.4 Použité knižnice

Zvolené knižnice sú publikované pod licenciou LGPL (Lesser General Public License) a MIT/X11, čo spĺňuje predpoklady z kapitoly 1.5 o voľne dostupných knižniciach spolu so širokou podporou a početnými návodmi.

2.4.1 ZedGraph

ZedGraph je knižnica tried, napísaná v jazyku C#, určená pre desktopové Windows Forms a webové ASP aplikácie v prostredí .NET. Je určená pre vykresľovanie 2D čiarových, stĺpcových alebo koláčových grafov Obsahuje plne prispôsobiteľné prostredie pre tvorbu komplexných grafov s viacerými osami x a y a takmer každý prvok grafu je možné upraviť podľa osobných preferencií používateľa. Medzi ďalšie výhody patrí napríklad možnosť vytvorenia hlavného panela grafu s listom jednotlivých grafov a ich vzájomného prepínania, aktualizácie dát v reálnom čase alebo editácia dát pomocou kurzoru myši v hlavnom okne grafu. [11]

Z vlastných skúseností vyniká jednoduchou tvorbou programovacieho kódu a rýchlosťou prekresľovania dát, kde počet zobrazených bodov v celom grafe jednoducho presiahne hodnotu 100 000.

2.4.2 Math.Net

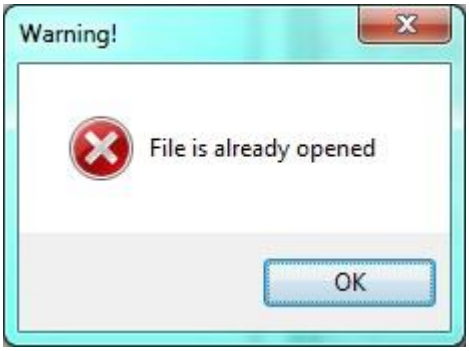
Knižnica je zameraná na poskytovanie metód a algoritmov pre numerické výpočty v oblasti vedy a techniky. Umožňuje riešiť náročné problémy ako špeciálne funkcie, lineárnu algebru, modely pravdepodobnosti, náhodné čísla, interpoláciu, regresiu a iné. Vznikla spojením dnAnalytics a Math.Net Iridium a je súčasťou projektu Math.NET. Podporuje .NET 4, .NET 3.5, Mono (Windows, Linux a Mac), SilverLight 5, Windows Phone 8, Windows 8/Store a zoznam uzatvára Android spolu s iOS. [12]

V aplikácii UniSense bude knižnica primárne využitá pre výpočet trendu a interpolácie.

2.5 Bezpečnosť programu

Pri práci s dátami je nutné zabezpečiť prevenciu pádu aplikácie a straty rozrobenej práce. Ochranu aplikácie počas celého procesu spustenia a interakcie s užívateľom zabezpečujú nasledujúce prvky:

- Deaktivácia ovládacích prvkov
  Slúži na zabránenie nekontrolovateľného správania aplikácie pri nevhodnom používaní, akým je napríklad viacnásobné stlačenie ovládacieho prvku. Pri použití ovládacieho prvku dôjde k jeho vypnutiu, následne sa vykonajú náležité funkcie a prvok sa znova aktivuje. Tento druh zabezpečenia je vhodný taktiež pri spúšťaní prvkov, u ktorých dochádza ku spusteniu náročnému výpočtu.
- Kontrola vykonávanej činnosti
  Overuje, či je možné vykonať užívateľom vyvolanú akciu. Príkladom je overenie, či je daný súbor s meranými dátami už v aplikácii otvorený. V prípade, že tomu tak je, vyvolá sa výstražná správa s popisom daného problému a nasledujúca činnosť je zastavená, viď obrázok 3.



Obrázok 3 Ukážka výstražnej správy

- Spracovanie výnimiek
  Jedná sa o úsek kódu pozostávajúceho z blokov *try-catch* za ktorým môže nasledovať blok *finally*. Úsek *try* skúša implementáciu použitého kódu. Ak sa v tejto časti vyskytne výnimka, nasledujúci blok *catch* ju zachytí a spracuje. Väčšinou sa jedná o upozornenie používateľa vhodnou textovou formou o výskyte chyby. Sekcia finally je vykonaná vždy, aj bez vygenerovanej výnimky, a umiestňuje sa do nej kód, ktorý vracia aplikáciu do pôvodného stavu pred výskytom výnimky. [8] Ukážka je na obrázku 4.

```
public class EHClass
{
    void ReadFile(int index)
    {
        // To run this code, substitute a valid path from your local machine
        string path = @"c:\users\public\test.txt";
        System.IO.StreamReader file = new System.IO.StreamReader(path);
        char[] buffer = new char[10];
        try
        {
            file.ReadBlock(buffer, index, buffer.Length);
        }
        catch (System.IO.IOException e)
        {
            Console.WriteLine("Error reading from {0}. Message = {1}", path, e.Message);
        }

        finally
        {
            if (file != null)
            {
                file.Close();
            }
        }
        // Do something with buffer...
    }

}
```

Obrázok 4 Ukážka try-catch-finally kódu

Zdroj: [7]

Je nutné avšak poznamenať, že tieto prvky nezabezpečujú celkovú ochranu aplikácie pred pádom, nakoľko nie je možné otestovať všetky možné scenáre interakcie medzi programom a užívateľom. Sú preto navrhnuté pre bežné a často sa vyskytujúce chyby.

2.6 XML dokumentácia

Jedna z podstatných vlastností programátora je tvorba transparentného zdrojového kódu (angl. self-documenting code). Táto praktika umožňuje skoro úplne obmedziť používanie komentárov ohľadom zdrojového kódu. Pri použití modulárnej architektúry a možnosti rozširovania vyvíjanej aplikácie do budúcnosti je však vhodné zdrojový kód opatriť XML dokumentáciou.

Jedná sa o špeciálnu vlastnosť programovacieho jazyka C#, ktorá umožňuje vytvoriť dokumentáciu priamo v zdrojovom kóde. Iné programovacie jazyky túto možnosť nemajú. [9]

Medzi hlavné výhody patrí [10]:

- Export dokumentácie
  Exportovanie umožňuje využiť napísanú dokumentáciu pri ďalšom spracovaní, napríklad pri tvorbe sekcie *Help* v aplikácii.
- Našepkávanie pri programovaní
  Technológia IntelliSense umožňuje okrem automatického doplňovania zobraziť XML dokumentáciu., tak ako je to ukázané na obrázku 5.

```
/// <summary>
/// Ukážka XML Dokumentácie
/// </summary>
/// <param name="parameter1">Popis parametru 1.</param>
/// <param name="parameter2">Popis parametru 2.</param>
/// <returns>Vráti hodnotu výpočtu dvoch vstupných parametrov.</returns>
public double UkazkovaFunckia(double parameter1, double parameter2)
{
    // Výpočet
}
```

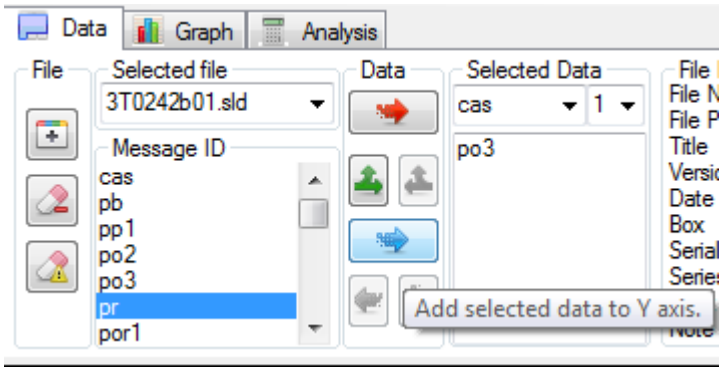
```
public double Model.MathLiB.UkazkovaFunckia(double parameter1, double parameter2)
 Ukážka XML Dokumentácie
parameter1: Popis parametru 1.
parameter2: Popis parametru 2.
Returns: Vráti hodnotu výpočtu dvoch vstupných parametrov.
```

Obrázok 5 Ukážka XML dokumentácie

2.7 Nápoveda

Všetky Windows Forms prvky s vlastnosťou *Tooltip* obsahujú nápovedu v anglickom jazyku. Týmto bolo možné dosiahnuť kompaktnejšieho prevedenia aplikácie a nezahltiť hlavnú obrazovku popisom jednotlivých komponentov a ich činnosti.

Na obrázku 6 je ukážka zobrazenej nápovede.



Obrázok 6 Nápoveda prvku Button

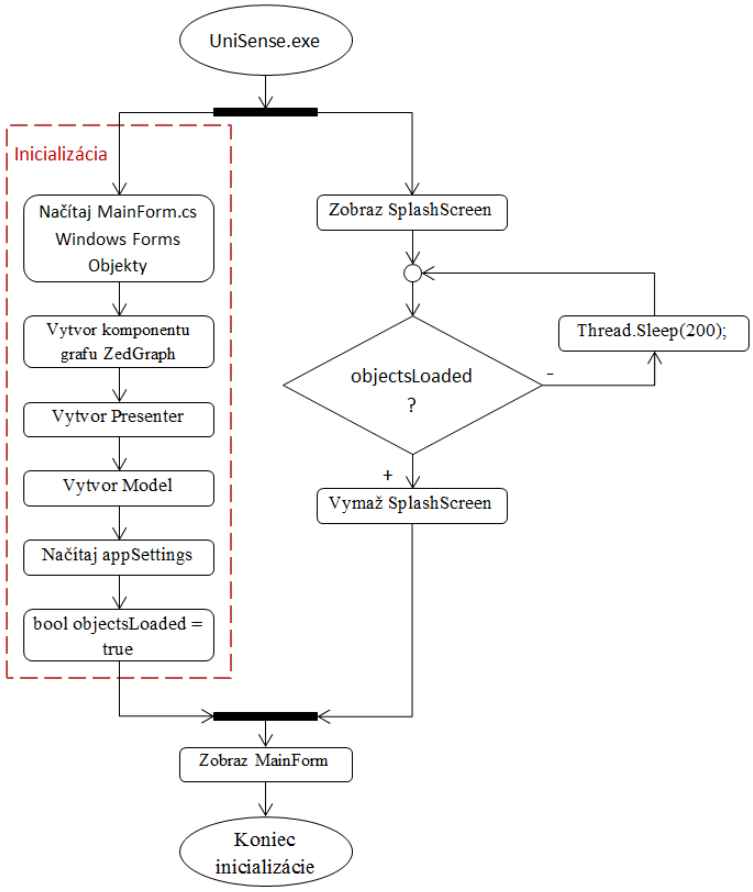
2.8 Lokalizácia

Kompletne celá aplikácia UniSense je lokalizovaná v anglickom jazyku. Jedná sa o všetky názvy premenných, prvkov Windows Forms, komentárov, ako aj všetka XML dokumentácia.

3 REALIZÁCIA APLIKÁCIE

3.1 Inicializácia

Pri použití modulárnej architektúry a možnosti rozširovania programu v budúcnosti bolo potrebné opatriť aplikáciu formulárom triedy *SplashScreen.cs*. Táto trieda sa spúšťa ešte pred vytvorením hlavného okna aplikácie v samostatnom vlákne. Zužitkovanie tejto metódy inicializácie umožňuje načítať všetky komponenty a moduly potrebné pre správnu funkčnosť ešte predtým, ako sa hlavné okno aplikácie zobrazí užívateľovi. Celý proces je zobrazený vo vývojovom diagrame na obrázku 7. Medzičasom je na obrazovke zobrazené logo programu s transparentným pozadím na obrázku 8.

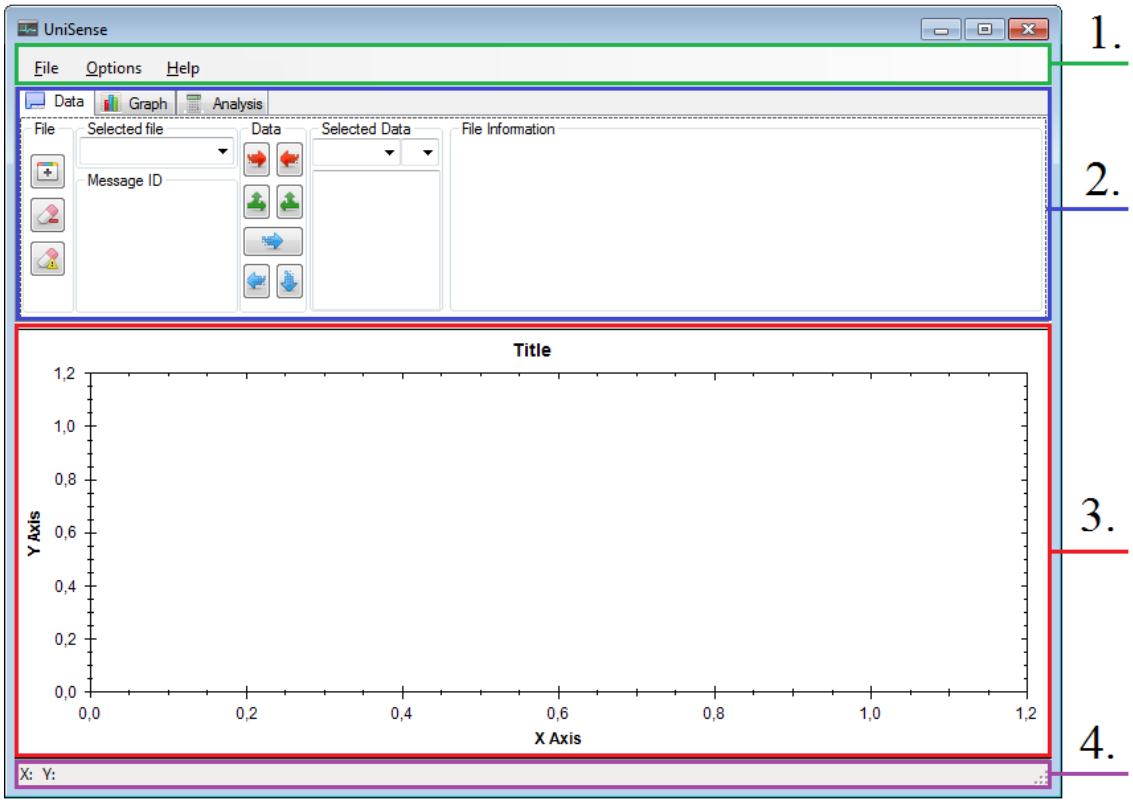


Obrázok 7 Diagram inicializácie aplikácie

Obrázok 8 Splashscreen logo aplikácie

3.2 Grafické užívateľské prostredie

Bolo vytvorené s ohľadom na jednoduchosť a intuitívnosť ovládania celej aplikácie. Grafické užívateľské rozhranie (GUI) je rozdelené na obrázku 9 na 4 hlavné časti.



Obrázok 9 Hlavná obrazovka aplikácie UniSense

3.2.1 Hlavné menu

- zelenou farbou orámovaná časť obrázku 9 označená číslom 1.

V momentálnej fáze vývoja aplikácia obsahuje malý počet ovládacích prvkov v hlavnom menu, nakoľko väčšina z nich je dostupná vo forme tlačidiel. Všetky prvky hlavného menu je možné spustiť klávesovou skratkou, ktorá je za názvom prvku v zátvorke.

Medzi ovládacie prvky hlavného menu patrí:

- Súbor/Koniec (Alt+F4)
- Options/Fullscreen (F3)
- Options/Always on top (F4)
- About/About (F1)

3.2.2 TabControl

- modrou farbou orámovaná časť obrázku 9 označená číslom 2.

Za účelom separácie ovládacích prvkov do logických skupín boli v prvku TabControl vytvorené 3 záložky.

1. Data

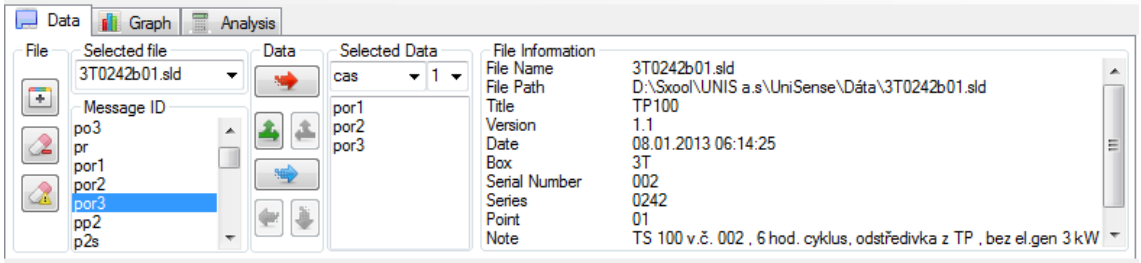
Prvá záložka *Data* obsahuje všetky ovládacie prvky vzťahujúce sa k práci so súbormi a ich dátami. Ovládacie prvky umožňujú:

- pridať/odobrať jeden/odobrať všetky súbory
- pridať/odobrať dáta do/z osi X
- pridať/odobrať druhú osu Y2
- pridať/odobrať /odobrať všetky dáta do/z osi Y1 alebo Y2
- zmeniť aktuálne spracovávaný súbor
- zmeniť aktuálne použitú os Y

Okrem týchto funkcií zobrazuje:

- dostupné premenné, ktoré je možné pridať do grafu
- premenné pridané do grafu
- hlavičku súboru

Na obrázku10 je ukážka práce so záložkou Data.



Obrázok 10 Ukážka práce so záložkou Data

2. Graph

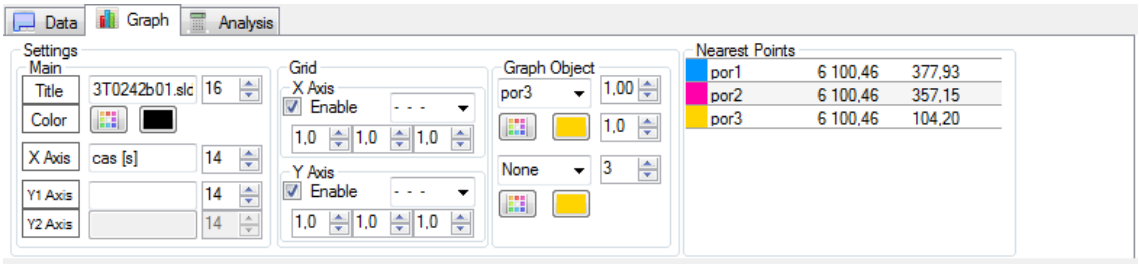
Druhá záložka *Graph* obsahuje všetky ovládacie prvky vzťahujúce sa k práci s grafom a jeho editovaním. Ovládacie prvky umožňujú:

- zmeniť názov grafu a osí X, Y1 a Y2
- zmeniť veľkosť písma názvov grafu a osí X, Y1 a Y2
- zmeniť farbu názvu grafu
- zapnúť/vypnúť mriežku v osi X/Y s vlastným nastavením parametrov
- zmena parametrov krivky (farba čiary, hrúbka čiary, typ symbolu, veľkosť symbolu, farba symbolu, multiplikátor)

Okrem týchto funkcií zobrazuje:

- aktuálne súradnice X, Y pre každú krivku v grafe spolu s farebnou interpretáciou pre jednoduchšie orientovanie

Na obrázku 11 je ukážka práce so záložkou *Graph*.



Obrázok 11 Ukážka práce so záložkou Graph

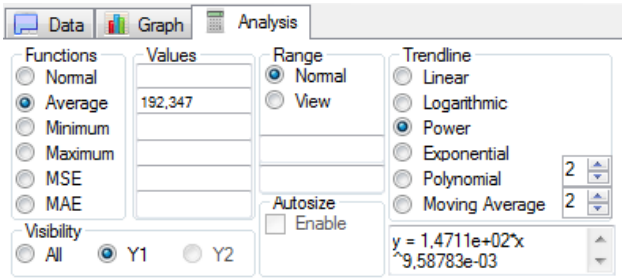
3. Analysis

Tretia záložka *Analysis* obsahuje všetky ovládacie prvky vzťahujúce sa k štatistickým funkciám a výpočtu interpolácie.

Ovládacie prvky umožňujú:

- nastaviť rozsah pre celý graf alebo vybranú oblasť
- vyhodnotiť štatistickú funkciu alebo interpoláciu
- nastaviť automatické nastavenie multiplikátora
- nastaviť viditeľnosť kriviek v ose pre obidve Y osi alebo jednotlivo pre Y1 a Y2

Na obrázku 12 je ukážka práce so záložkou Analysis.



Obrázok 12 Ukážka práce so záložkou Analysis

Obrázok 12 taktiež zobrazuje všetky implementované funkcie zo štatistiky v bloku *Functions* a následne aj možnosti interpolácie dát v bloku *Trendline*. Štatistické údaje a interpolácia sú počítané zo zvoleného rozsahu uvedeného na obrázku 11 v bloku *Range* pre premennú, ktorá je uvedená na obrázku 10 v bloku *Graph Object*.

3.2.3 ZedGraph

- červenou farbou orámovaná časť obrázku 9 označená číslom 3.

Komponent knižnice *ZedGraph* zodpovedný za vykresľovanie užívateľom vybraných dát.

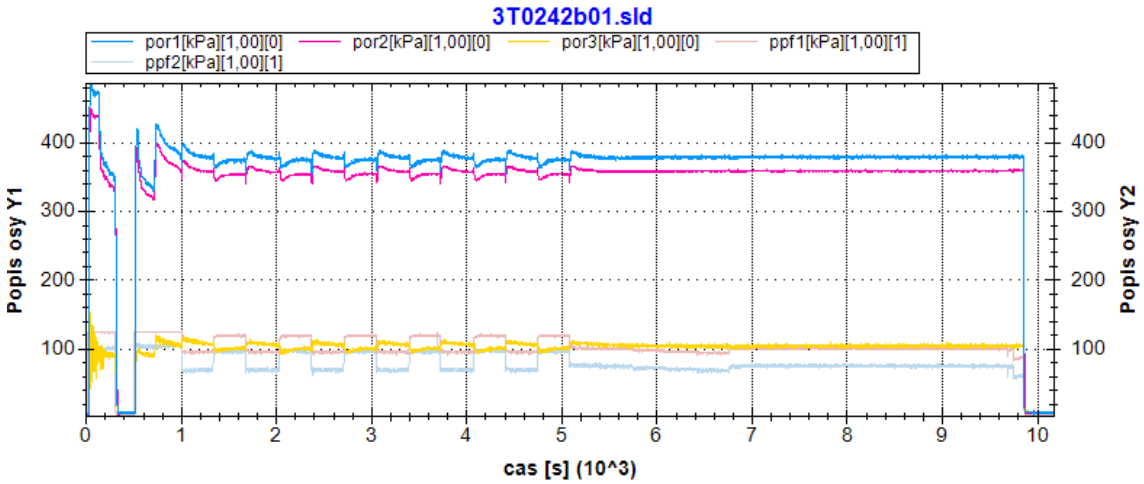
Obsahuje vlastné kontextové menu s možnosťami ako:

- skopírovať graf do schránky (angl. clipboard)
- uložiť graf do rôznych grafických formátov (*.emf, *.png, *.gif, *.jpg, *.tif, *.bmp)
- vytlačiť graf
- nastaviť pôvodné hodnoty rozsahu osi X, Y1 a Y2
- vrátiť sa o jeden krok vo funkcii zoom
- exportovať vykreslené dáta do separátnych súborov *.txt

Medzi ďalšie ponúkané funkcie patrí:

- inteligentný zoom
- zobrazenie osového krížu

Na obrázku 13 je ukážka práce s komponentom *ZedGraph*.



Obrázok 13 Ukážka práce s komponentom ZedGraph

Jednotlivé krivky v grafe sú označené podľa špeciálneho identifikátora v legende. Jedná sa o tvar „*XXX[uuu][m,mm][Y]*“ kde jednotlivé časti znamenajú nasledovné:

- „*XXX*“ obsahuje názov krivky
- „*[uuu]*“ je fyzikálny rozmer veličiny „XXX“
- „*[m,mm]*“ obsahuje hodnotu multiplikátora
- „*[Y]*“ indikuje index osi Y

3.2.4 StatusStrip

- fialovou farbou orámovaná časť obrázku 9 označená číslom 4.

Slúži pre zobrazovanie informácií z programu. Obsahuje dve časti, kde tou prvou sú súradnice kurzora na aktívnej ploche grafu X a Y. Druhá časť zobrazuje informačné alebo výstražné správy.

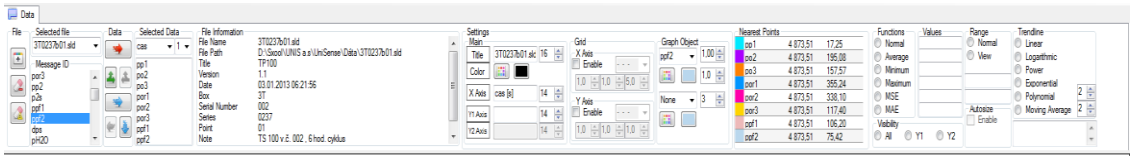
Na obrázku 14 je ukážka práce s komponentom StatusStrip.

Obrázok 14 Ukážka práce s komponentom StatusStrip

4 MOŽNOSTI PRÁCE S DÁTAMI

4.1 Celoobrazovkový mód

Aplikácia umožňuje prepnutie do celoobrazovkového režimu s využitím výhody monitorov s vysokým rozlíšením. Aktiváciu vyvoláme tlačidlom F3, čím dosiahneme roztiahnutie aplikácie na celú obrazovku s režimom *TopMost*, kedy sa aplikácia nachádza na úplnom vrchu a nie je možné prepnúť sa do inej aplikácie. Pre maximalizáciu využitia priestoru obrazovky boli odstránené tlačidlá pre minimalizáciu, maximalizáciu a vypnutie aplikácie. Pri tomto režime sa preskupia prvky *TabControl* do jednej záložky a je možné ich využívať súčasne. Mód má dva režimy vyplývajúce z rozlíšenia monitora. Ak užívateľ pracuje na monitore s rozlíšením širokým aspoň 1600 pixelov, preskupia sa mu dohromady prvky *Data* a *Graph*, záložka *Analysis* ostane samostatná. Ak má monitor rozlíšenie na šírku 1920 pixelov a viac, je mu umožnené využiť preskupenia všetkých troch záložiek *TabControl* do jednej. Na obrázku 15 je vidieť ukážka celoobrazovkového módu s rozlíšením obrazovky 1920x1080 pixelov.



Obrázok 15 Ukážka práce vo Fullscreen móde

4.2 Zmena multiplikátora

Každej krivke v grafe je umožnené meniť svoje pôvodné nastavenia, medzi najväčšiu výhodu patrí zmena multiplikátora, ktorý prenásobí údaje v ose Y zvolenou hodnotou. Týmto je lepšie umožnené vzájomne porovnávať jednotlivé krivky medzi sebou.

Táto funkcia je dostupná aj v automatickom móde v záložke *Analysis*, blok *Autosize*, prvok *Enable*. Pre nedostatočné otestovanie je však nevhodné ju používať.

4.3 Druhá osa Y

Graf je možné rozšíriť o druhú osu Y2. Krivky je možné pridávať do ľubovoľnej z nich. V prípade, že chce užívateľ osu Y2 odstrániť, je mu ponúknutá možnosť krivky s indexom osi Y2 presunúť do osi Y alebo ich z grafu odstrániť. Práca s osou Y2 je znázornená na obrázku 13 v kapitole 3.2.3.

4.4 Filtrovanie kriviek

Do aplikácie bola pridaná možnosť filtrovať zobrazené krivky v grafe podľa indexu osi Y. Pri využití tejto funkcie sa v grafe ponechajú len krivky s indexom vybranej osi, legenda grafu sa upraví aby obsahovala len zobrazené prvky a nepotrebná os Y sa skryje. Taktiež dochádza k úprave minimálneho a maximálneho rozsahu osi X a vybranej osi Y aby korešpondovali so zobrazenými údajmi v grafe.

ZÁVER

Cieľom tejto bakalárskej práce bolo vytvoriť aplikáciu pre spracovávanie meraných dát zo standu leteckých motorov. Aplikácia bude využitá primárne vo firme Unis a.s, kde sa zaoberajú problematikou simulácie leteckých motorov v reálnom čase. Aplikácia umožňuje rôzne režimy spracovania meraných dát a zjednodušuje ich vzájomné porovnávanie medzi dvomi simuláciami.

Aplikáciu, ktorá je výsledkom tejto bakalárskej práce, som vytvoril samostatne podľa požiadaviek firmy Unis a.s. pod vedením Ing. Lukáša Ertla, s ktorým boli jednotlivé návrhy a prípadné doplňujúce požiadavky konzultované počas celej doby vývoja.

Úvod práce tvorila rešerš, ktorá zahŕňala jednotlivé problematiky tvorby aplikácií pre operačný systém Microsoft Windows a platformu .NET Framework.

V tejto bakalárskej práci boli ďalej zhrnuté jednotlivé konzultácie ohľadom návrhu modulárnej architektúry aplikácie, ktorá bola vytvorená na základe modelu Model-View-Presenter. To umožnilo rozdeliť aplikáciu na tri oddelené vrstvy, pričom každá z nich mala na starosti presne definované zodpovednosti. Týmto bola docielená separácia záujmov a oddelenie závislosti grafického používateľského prostredia od biznis logiky a modelu.

V ďalšej časti bolo predstavené grafické prostredie aplikácie za účelom jeho logického rozdelenia do 4 hlavných častí, z nich každá bola popísaná z hľadiska možností ovládania aplikácie a názornej ukážky. Prostredie bolo navrhnuté tak, aby umožnilo štatistickú prácu s dátami, ako napríklad výpočet interpolácie alebo smerodajnej odchýlky.

Posledná kapitola zahŕňala možnosti práce s dátami na celoobrazovkovom režime, využitím dvoch osí y, ich vzájomného filtrovania a zmeny multiplikátora za účelom efektívnejšieho porovnávania dát.

Samotná aplikácia je momentálne v ďalšej fáze vývoja, ktorá zahŕňa rozšírenie aplikácie napríklad o export dát do užívateľom zadanej formy alebo automatické generovanie reportov.

5 ZOZNAM POUŽITÝCH ZDROJOV

[1] REGHUNADH, Jerry a Neha JAIN. Selecting the optimal programming language. *IBM* [online]. 13.09.2011 [cit. 2014-02-07]. Dostupné z: <http://www.ibm.com/developerworks/web/library/wa-optimal/index.html?ca=dat>

[2] NAGEL, Christian et al.: *C# 2008: Programujeme profesionálně*. Vyd. 1. Brno: Computer Press, 2009, 772 s. ISBN 978-80-251-2401-7.

[3] Moving Java Applications to .NET. MICROSOFT. [online]. [cit. 2014-05-5]. Dostupné z: <http://msdn.microsoft.com/en-us/library/ms973842.aspx>

[4] Přehled architektury.NET. MICROSOFT. [online]. 2001.[cit.2014-05-15] Dostupné z:<http://download.microsoft.com/download/8/6/c/86c09926-affc-4e14-bec0-3c45cd989436/Prehled_architektury_NET.pdf>

[5] D. AGUILAR, Luis. *UI Design Using Model-View-Presenter* [online]. 18.03.2013 [cit. 2014-05-25]. Dostupné z: <http://www.codeproject.com/Articles/563809/UIplusDesignplusUsingplusModel-View-Presenter>

[6] FOSSMO, Pål. Model View Presenter explained. [online]. 26.12.2007 [cit. 2014-05-25]. Dostupné z: <http://blog.fossmo.net/post/Model-View-Presenter-explained.aspx>

[7] Try-catch-finally (C# Reference). MICROSOFT. [online]. [cit. 2014-05-26]. Dostupné z: <http://msdn.microsoft.com/en-us/library/dszsf989.aspx>

[8] ROSSEL, Sander. Using Try... Catch..., Finally!. [online]. 5.02.2011 [cit. 2014-05-26]. Dostupné z: <http://www.codeproject.com/Articles/154121/Using-Try-Catch-Finally>

[9] XML Documentation. MICROSOFT. [online]. [cit. 2014-05-26]. Dostupné z: <http://msdn.microsoft.com/en-us/library/b2s063f7(v=vs.71).aspx>

[10] D. ELLIOTT, Michael. C# and XML Source Code Documentation. [online]. 02.04.2010 [cit. 2014-05-26]. Dostupné z: <http://www.codeproject.com/Articles/11082/C-and-XML-Source-Code-Documentation>

[11] CHAMPION, J. A flexible charting library for .NET. [online]. 06.06.2007 [cit. 2014-05-29]. Dostupné z: <http://www.codeproject.com/Articles/5431/A-flexible-charting-library-for-NET>

[12] Math.net numerics. [online]. [cit. 2014-05-30]. Dostupné z: <http://numerics.mathdotnet.com/>

6 ZOZNAM POUŽITÝCH SKRATIEK A SYMBOLOV

ADO	ActiveX Data Object
API	Application Programming Interface
ASP	Active Server Page
BCL	Base Class Library
CLR	Common Language Runtime
CLS	Common Language Specifications
GUI	Graphical User Interface
IDE	Integrated Development Environment
JIT	Just-in-time
LGPL	Lesser General Public License
MIT/X11	Massachusetts Institute of Technology
MSIL	Microsoft Intermediate Language
OS	Operating System
OS	Operating System
RAD	Rapid Application Development
WPF	Windows Presentation Foundation
XML	Extensive Markup Language

7 ZOZNAM OBRÁZKOV

8 ZOZNAM PRÍLOH

Prílohy na CD:

Program aplikácie UniSense.exe