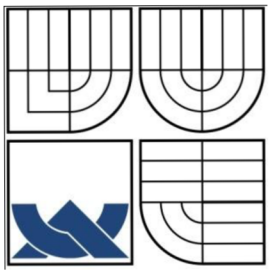
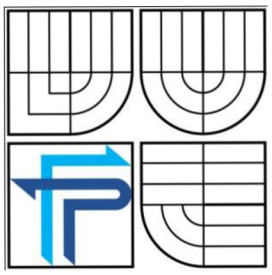


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA PODNIKATELSKÁ
ÚSTAV INFORMATIKY

FACULTY OF BUSINESS AND MANAGEMENT
INSTITUTE OF INFORMATICS

NÁVRH WEBOVÉ APLIKACE V ASP.NET PRO ZJIŠTĚNÍ TRENDU ČASOVÉ ŘADY

THE SUGGESTION ASP.NET WEB APPLICATIONS FOR DETECTION OF TIME SERIES TREND

BAKALÁŘSKÁ PRÁCE
A BACHLEOR'S THESIS

AUTOR PRÁCE — JANA CÍCHOVÁ
AUTHOR

VEDOUCÍ PRÁCE — Ing. Karel Doubravský, Ph.D.
SUPERVISOR

Brno 2010

Vysoké učení technické v Brně
Fakulta podnikatelská

Akademický rok: 2009/10
Ústav informatiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Jana Cíchová

Manažerská informatika (6209R021)

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách, Studijním a zkušebním řádem VUT v Brně a Směrnicí děkana pro realizaci bakalářských a magisterských studijních programů zadává bakalářskou práci s názvem:

Návrh webové aplikace v ASP.NET pro zjištění trendu časové řady

v anglickém jazyce:

The Suggestion ASP.NET Web Applications for Detection of Time Series Trend

Pokyny pro vypracování:

Úvod
Vymezení problému a cíle práce
Teoretická východiska práce
Analýza problému a současné situace
Vlastní návrhy řešení, přínos návrhů řešení
Závěr
Seznam použité literatury
Přílohy

Podle § 60 zákona č. 121/2000 Sb. (autorský zákon) v platném znění, je tato práce "Školním dílem". Využití této práce se řídí právním režimem autorského zákona. Citace povoluje Fakulta podnikatelská Vysokého učení technického v Brně. Podmínkou externího využití této práce je uzavření "Licenční smlouvy" dle autorského zákona.

Seznam odborné literatury:

ANDĚL, J. Základy matematické statistiky. 2. vyd. Praha : Matfyzpress, 2007. ISBN 978-80-7378-001-2.
CIPRA, T. Analýza časových řad s aplikacemi v ekonomii. 1. vyd. Praha : SNTL/ALFA, 1986. ISBN 99-00-00157-X.
HINDLS, R. a HRONOVÁ, S. a SEGER, J. Statistika pro ekonomy. 1.vyd. Praha : Professional Publishing, 2002. ISBN 80-86419-26-6.
KROPÁČ, J. Statistika B. 1. vyd. Brno : VUTFP, 2006. ISBN 80-214-3295-0.
RYAN, T. P. Modern Regression Methods. New York : John Wiley&Sons, Inc., 1997. ISBN 0-471-52912-5.
ZVÁRA, K. Regresní analýza. 1. vyd. Praha : Academia, 1989. ISBN 80-200-0125-5.

Vedoucí bakalářské práce: Ing. Karel Doubravský, Ph.D.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2009/10.

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ – Fakulta podnikatelská

L.S.

Ing. Jiří Kříž, Ph.D.
Ředitel ústavu

doc. RNDr. Anna Putnová, Ph.D., MBA
Děkanka

V Brně, dne 7. 2. 2010

ABSTRAKT

Cílem práce je vytvoření webové aplikace v prostředí ASP.NET, sloužící k regresní analýze časových řad. Výsledný produkt bude nabízen firmou vytvářející webové prezentace pro cestovní kanceláře, jako součást statistického balíčku, který již tato firma produkuje. Produktem bude webová prezentace, pomocí níž bude moci cestovní kancelář analyzovat například prodejnost jednotlivých destinací a na základě zadaných dat vytvořit prognózu pro další roky. Výsledky budou zpracovány jak textově, tak graficky a to do souboru typu *.xls nebo *.pdf.

KLÍČOVÁ SLOVA

Regresní analýza, časové řady, ASP.NET, webová aplikace, statistika, C#

ABSTRACT

The aim of this work is the creation of web applications in ASP.NET, for the regression analysis of time series. The resulting product will be offered by creating web presentations for travel agencies, as part of a statistical package that has the company produces. The product is a web presentation, by which the travel agency will be able to analyze the marketability of such destinations, and based on specific data to create a forecast for the next years. The results will be processed both text and graphic file type *. xls or *. pdf.

KEY WORDS

Regression Analysis, Time Series, ASP.NET, Internet Application, Statistics, C#

BIBLIOGRAFICKÉ CITACE

CÍCHOVÁ, J. *Návrh webové aplikace v ASP.NET pro zjištění trendu časové řady*. Brno: Vysoké učení technické v Brně, Fakulta podnikatelská, 2010. 59 s. Vedoucí bakalářské práce Ing. Karel Doubravský, Ph.D.

ČESTNÉ PROHLÁŠENÍ

Prohlašuji, že předložená bakalářská práce je původní a zpracovala jsem ji samostatně. Prohlašuji, že citace použitých pramenů jsou úplné, a že jsem ve své práci neporušila autorská práva (ve smyslu Zákona č. 121/2000 Sb., o právu autorském a o právech souvisejících s právem autorským).

V Brně dne: ……

Vlastnoruční podpis

PODĚKOVÁNÍ

Na tomto místě bych chtěla poděkovat svému vedoucímu bakalářské práce panu Ing. Karlu Doubravskému PhD., za jeho rady, odborné vedení a pomoc při zpracování mé bakalářské práce. Dále bych chtěla poděkovat vedení firmy Pear spol. s r.o. za poskytnutá data a podporu při této práci. V neposlední řadě bych také ráda poděkovala svojí rodině a všem, kteří mi vždy byli oporou a podporovali mě nejen při studiu.
Děkuji Vám.

OBSAH

ÚVOD 10
CÍLE PRÁCE 11
1 TEORETICKÁ VÝCHODISKA PRÁCE 12
1.1 Teorie z oblasti časových řad a regresní analýzy 12
1.1.1 Časové řady 12
1.1.2 Regresní analýza 15
1.2 Teorie k programování 20
1.2.1 .NET Framework 20
1.2.2 Vývojové prostředí ASP.NET 22
1.2.3 Programovací jazyk C# 22
1.2.4 Kaskádové styly 25
1.2.5 Databáze a jazyk SQL 26
1.2.6 JavaScript 27
2 ANALÝZA PROBLÉMU A SOUČASNÉ SITUACE 29
2.1 Funkčnost současného systému 29
3 POPIS IMPLEMENTACE APLIKACE 32
3.1 Řešení 32
3.1.1 Část přihlašovací 32
3.1.2 Část obsahová 32
3.1.3 Část Menu 48
3.2 Chyby v řešení 49
4 PŘÍNOSY APLIKACE A MOŽNOSTI ROZŠÍŘENÍ 50
4.1.1 Přínos ke stávajícímu řešení 50
4.1.2 Využitelnost firmou Pear spol. s r.o. 50
4.1.3 Přínosy řešení pro mě 51
4.1.4 Další možnosti využití 51
4.2 Možnosti rozšíření 52
4.2.1 Rozšíření výpočetních možností 52
4.2.2 Navázání na jednotlivé databáze cestovních kanceláří a poskytovatelů zájezdů 53
4.2.3 Import dat pro analýzu ze souborů 53
4.2.4 Design 53
4.2.5 Podpora vícejazyčnosti 53

4.2.6 Možná drobná vylepšení 54
ZÁVĚR 55
LITERATURA 56
SEZNAM OBRÁZKŮ 58
SEZNAM PŘÍLOH 59

ÚVOD

Pro svoji bakalářskou práci jsem si zvolila uplatnění statistických metod při zpracování dat díky tomu, že statistika mě ve škole baví a propojení s programováním se mi zdálo velmi vhodné vzhledem k oboru, který studuji.

Po absolvování odborné praxe se mi naskytla příležitost spojená s firmou vyrábějící pro cestovní kanceláře webové aplikace, rezervační a ekonomické systémy s podporou statistických aplikací, které ale neobsahují žádný nástroj pro zpracování a analýzu dat s možností výpočtu budoucího vývoje a tak jsem se rozhodla, že právě vytvoření tohoto nástroje bude předmětem mé bakalářské práce.

Po uvážení veškerých svých možností a také toho, s jakými nástroji pracuje firma Pear spol. s r.o. jsem zvolila jako výsledek své bakalářské práce webovou aplikaci, analyzující právě časové řady, což pro mě obnáší naučení se a nastudování potřebných prostředků jako je ASP.NET nebo jazyk C#.

CÍLE PRÁCE

Jak bylo výše uvedeno, cílem této práce je vytvořit webovou aplikaci, která bude analyzovat vývoj časových řad, jejich charakteristiky a na závěr také určovat jejich trend. Toho má být dosaženo ve vývojovém prostředí ASP.NET za použití jazyka C#.

Je tedy nutné nastudovat syntaxi a sémantiku jazyka C#, seznámit a naučit se pracovat s některou z aplikací, která interaktivně podporuje vývoj webových aplikací v jazyce C#. V aplikaci je také možné využít znalostí získaných při výuce z oblastí tvorby webových stránek nebo z databázových systémů. Samozřejmostí je využití poznatků ze statistiky zabývajících se právě časovými řadami a tedy i regresní analýzou.

Webová aplikace bude mít za úkol zjistit závislost analyzované veličiny na časové složce a podle této závislosti určit její trend pro další období. Pro určení trendu bude vybíraná optimální křivka pomocí indexu determinace a následně dopočítána prognóza pro vybraný počet období. Tato část práce je stěžejní, jelikož vyjadřuje podstatu celé práce.Aplikace bude nabízet i možnosti výpočtu různých rozšiřujících veličin, jako jsou například výběrový průměr časové řady nebo první diference. Veškeré možné výsledky se pokusím zobrazit v grafech, tak aby se v nich mohl uživatel jednoduše zorientovat.

Aby byla celá aplikace „user-friendly“ – tedy uživatelsky přívětivá, bude si sám uživatel moci zadat, co chce počítat a co počítat naopak nechce, které grafy se budou zobrazovat a které ne, a také, jaký typ grafu chce pro zobrazení hodnot využít. Dále bude vytvořená aplikace nabízet možnou registraci pro uživatele. Z této registrace bude vycházet jednoduchá výhoda – uživatel bude moci ukládat svoje již propočítané sestavy a později se k nim vracet. Dále bude aplikace uživateli nabízet možnost exportovat si již vypočítaná data do formátu užívaných programy Microsoft Excel a Adobe Acrobat – tedy *.pdf a *.xls.

Po vytvoření samotné webové aplikace zhodnotím její přínosnost ke stávajícímu řešení, přínos pro firmu a možnou využitelnost v praxi, dále jaké přínosy měla tato bakalářská práce pro mě a navrhnu další možnosti jejího rozšíření nebo vylepšení.

1 TEORETICKÁ VÝCHODISKA PRÁCE

V kapitole o teoretických východiskách práce rozeberu jak technologie využívané pro vytvoření aplikace, tak statistické metody, které bude výsledná webová aplikace zpracovávat.

1.1 Teorie z oblasti časových řad a regresní analýzy

Z oblasti statistiky jsem po konzultaci s vedením firmy Pear spol. s r.o. vybrala dvě „kapitoly“ z oboru statistky, které by bylo vhodné zpracovat tak, aby byla výsledná webová aplikace použitelná i v praxi. V následujících kapitolách tedy objasním pojmy z oblasti časových řad a regresní analýzy, které se vzájemně velmi dobře doplňují.

1.1.1 Časové řady

Pojmem časové řady rozumíme data, která mají konkrétní vývoj v čase. Analýzou dostupných dat, pak můžeme nejen získávat komplexní náhled na danou problematiku, ale i určovat prognózy dalšího vývoje sledované veličiny a na základě těchto prognóz vyvozovat závěry, které nám pomáhají například při manažerských rozhodováních. V dnešní době jsou tyto analýzy důležitou součásti každého nejen výrobního podniku, protože jejich výsledkem jsou důležitá provozní rozhodnutí například o tom, kolik kterého výrobku se další rok vyrobí nebo můžeme zjistit, jaké budou v dalším období náklady na výrobu nějakého výrobku.

Dalšími příklady těchto časových řad může být například vývoj průměrného platu vysokoškolsky vzdělaných lidí nebo nezaměstnanost v Jihomoravském kraji apod.

Pokud sestavujeme časovou řadu, je třeba dbát důraz na to, jaká data používáme. Není možné v jednom okamžiku používat jako měřítko časové složky roky a v následné části měsíce. Znamená to tedy, že data musí být seřazena chronologicky a jednotlivé časové kroky musí mít stejnou jednotku. Stejně tak obsah dat musí být ve stejném „formátu“.

Rozlišujeme časové řady intervalové a časové řady okamžikové. Rozdíl mezi nimi spočívá v tom, že sčítáním dat v intervalové časové řadě získáme hodnoty platné

pro sečtený časový okamžik, zatímco u okamžikové časové řady toto neplatí. Tato skutečnost se dá interpretovat na příkladu, kdy budeme brát v úvahu počet obyvatel v letech oproti přírůstkům obyvatel za rok. První zmiňovaný příklad je případem okamžikové časové řady. Pokud by byl střední stav obyvatelstva v roce 2007 10 322 689 a v roce 2008 10 429 692 (zdroj 7), pak sečtením obou položek nedostaneme stav v letech 2007-2008. Pokud ale budeme analyzovat pouze přírůstky v jednotlivých letech, pak sčítání těchto dat již je relevantní.

Vzhledem k tomu, že ve své bakalářské práci budu využívat výhradně časové řady intervalové a to s časovým měřítkem ročním, čtvrtletním a měsíčním, je třeba vyřešit problém, že každý měsíc není stejně dlouhý, což může lehce zkreslit výsledky analýzy časové řady. Musíme tedy převést data do jednoho formátu. Já si zvolila formu, kdy si stanovíme standardní délku měsíce 30 dnů. Měsíční hodnotu ukazatele pak vydělíme počtem dnů v konkrétním měsíci a vynásobíme 30. (1, str. 114 – 116).

Počet dní v měsíci d

Hodnota ukazatele h

Převedená hodnota ukazatele p

$$p = \frac{h}{d} \times 30$$

Vzorec 1: Převedení hodnot měsíčního ukazatele na hodnoty srovnatelné

Každou časovou řadu můžeme dekomponovat na několik složek. Tohoto postupu se používá z důvodu přesnějšího pospání závislostí v rámci jednotlivých složek časové řady než v časové řadě nerozložené. Hlavní složkou každé časové řady je složka trendová, která zahrnuje „surové“ hodnoty, které by vypovídali o průběhu celé časové řady bez působení jakýchkoliv okolních vlivů. Dalšími složkami jsou složka cyklická – kterou zle těžko předpovídat, protože její působení je ovlivňováno vnějšími vlivy, které jsou mnohdy špatně určitelné, dále složka sezónní, která reprezentuje pravidelné změny v rámci roku, které se periodicky opakují. Typickým příkladem vlivu sezónní složky je například klesání nezaměstnanosti v letních měsících, protože se objevují pracovní místa spojená s pěkným počasím – ať se jedná o práci na stavbách nebo více práce

v zemědělství. Poslední složkou každé hodnoty časové řady je pak hodnota náhodné složky, neboli složky reziduální, která zahrnuje vše, s čím se ještě v rámci konkrétní hodnoty časové řady nepočítalo. Do této složky můžeme zařadit například i lidskou chybu – při zaokrouhlování, při měření apod. Pokud tedy všechny složky sečteme, vyjde nám konkrétní hodnota pro zadané časové období.

Charakteristiky časových řad

Charakteristiky časových řad nám umožňují podívat se na řešenou problematiku důkladněji a získat komplexní náhled na věc. Za nejjednodušší charakteristiku můžeme považovat průměr časové řady, který vypočítáme následovně:

- pokud se jedná o časovou řadu intervalovou,

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$$

Vzorec 2: Průměr intervalové časové řady

- pokud se jedná o časovou řadu okamžikovou,

$$\bar{y} = \frac{1}{n-1}\left[\frac{y_1}{2} + \sum_{i=2}^{n-1} y_i + \frac{y_n}{2}\right]$$

Vzorec 3: Průměr okamžikové časové řady

kde n je počet dat a y hodnota zkoumané veličiny.

Pokud budeme chtít zkoumat průběh časové řady, použijeme první diference, průměr prvních diferencí, koeficient růstu a průměr koeficientu růstu.

První diference vyjadřují přírůstky či úbytky v jednotlivých létech oproti roku minulému. Je tedy jasné, že se zjišťují až pro druhé časové období či okamžik.

$$_1d(y) = y_i - y_{i-1}$$

Vzorec 4: Výpočet první diference

Pokud budeme mít zjištěny všechny první diference, můžeme z nich spočítat jejich průměr, který udává, jaký byl průměrný přírůstek nebo úbytek za časovou jednotku.

$$\overline{{}_1d(y)} = \frac{1}{n-1}\sum_{i=2}^{n} {}_1d_i\,(y) = \frac{y_n - y_1}{n-1}$$

Vzorec 5: Průměr prvních diferencí

Obdobným způsobem můžeme získat také koeficient růstu vyjadřující, kolikrát se hodnota časové řady zvýšila oproti období minulému.

$$k_i(y) = \frac{y_i}{y_i - 1}$$

Vzorec 6: Koeficient růstu

A následně spočítat průměrný koeficient růstu, který nám říká, kolikrát se průměrně zvýšila či snížila sledovaná veličina za jedno období.

$$\overline{k(y)} = \sqrt[n-1]{\prod_{i=2}^{n} k_i(y)} = \sqrt[n-1]{\frac{y_n}{y_1}}$$

Vzorec 7: Průměrný koeficient růstu

1.1.2 Regresní analýza

Pro popis vývoje trendu časových řad a prognóz pro následující období je nejpoužívanější právě regresní analýza. Regresní analýza skýtá mnoho možností využití ve všech oborech, kde dochází k měření veličin, které jsou na sobě závislé. Mezi těmito veličinami rozeznáváme nezávisle proměnnou x, která se bude konstantně měnit, v mém případě se bude vždy jednat o časovou složku a závisle proměnou y, která bude nabývat konkrétní hodnoty právě v závislosti na x.

V prvé řadě můžeme data pomocí regresní analýzy „normovat“ či jinak řečeno vyrovnat a tím se zbavit určitých odchylek, které mohou být vyvolány okolními vlivy.

V další fázi již určujeme prognózy pro další období a na základě těchto prognóz můžeme vyvozovat důsledky. (1, str. 124).

Regresní funkce

Při zvolené regresní funkci, kterou značíme $\eta(x;\beta_{1,}\beta_{2,\dots,}\beta_p)$, p ≥ 1, kde x značí vysvětlující proměnnou a β_1 až β_p označují regresní koeficienty, které se snažíme zvolit tak, aby vyrovnané hodnoty y_i (vysvětlované proměnné) se blížili co nejvíce původním hodnotám. Čím lépe se nám podaří odhadnout koeficienty, tím menší jsou vzdálenosti původních a vyrovnaných hodnoty, a tím větší máme pravděpodobnost, že se nám podaří správně vypočítat trend pro další období. (1, str. 79). V následujících oddílech popíši jednotlivé regresní funkce, které jsem pro svůj projekt zvolila.

Regresní přímka

Regresní přímka je nejjednodušším používaným modelem regresní analýzy. Pro určení regresní přímky používáme metody nejmenších čtverců. Její podstata spočívá v minimalizaci součtu druhých mocnin rozdílu naměřených hodnot a hodnot vyrovnaných.

Regresní přímka je udávána ve tvaru $\eta(x) = \beta_1 + \beta_2 x$. Odhady regresních koeficientů β_1 a β_2 značíme b_1 a b_2 pomocí kterých můžeme zapsat odhad regresní přímky ve tvaru:

$$\eta(x) = b_1 + b_2 x$$

Vzorec 8: Obecný tvar rovnice regresní přímky

Odhady koeficientů β pak vypočítáme podle vzorců (převzato z 1, str. 81), které vypadají následovně:

$$b_2 = \frac{\sum_{i=1}^{n} x_i y_i - n\bar{x}\bar{y}}{\sum_{i=1}^{n} x_i^2 - n\bar{x}^2}$$

Vzorec 9: Koeficient b_2 obecného tvaru regresní přímky

$$b_1 = \bar{y} - b_2 x$$

Vzorec 10: Koeficient b_1 obecného tvaru regresní přímky

Ve výše uvedených vzorcích používáme veličiny zvané výběrové průměry, jejich výpočet vypadá následovně:

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$$

Vzorec 11: Výběrový průměr vysvětlované proměnné

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Vzorec 12: Výběrový průměr vysvětlující proměnné

Nelinearizovatelné funkce nelineárních regresních modelů

Jedná se o modely, kdy regresní koeficienty závisí na zvolené funkci nelineárně. V tomto případě se je však nebudeme snažit linearizovat, jak bychom to dělali při linearizovatelných funkcích nelineárního regresního modelu, nýbrž podle vzorců vypočíst regresní koeficienty. Jedná se o tyto funkce ([1] str. 107).:

- Modifikovaný exponenciální trend s funkčním přepisem:

$$\eta(x) = \beta_1 + \beta_2 \beta_3^x$$

Vzorec 13: Funkční přepis rovnice modifikovaného exponenciálního trendu

- Logistický trend s funkčním přepisem:

$$\eta(x) = \frac{1}{\beta_1 + \beta_2 \beta_3^x}$$

Vzorec 14: Funkční přepis rovnice logistického trendu

- Gompertzova křivka s funkčním přepisem:

$$\eta(x) = e^{\beta_1 + \beta_2 \beta_3^x}$$

Vzorec 15: Funkční přepis rovnice modifikovaného Gompertzovy křivky

Odhady koeficientů výše zmíněných křivek a pomocné výpočty pro ně budeme realizovat podle následujících vzorců (1, str. 108):

$$b_3 = \left[\frac{S_3 - S_2}{S_2 - S_1}\right]^{\frac{1}{m \times h}}$$

Vzorec 16: Koeficient b_3 pro výše uvedené funkce

$$b_2 = (S_2 - S_1) \times \frac{b_3^h - 1}{b_3^{x_1} \times \left(b_3^{m*h} - 1\right)^2}$$

Vzorec 17: Koeficient b_2 pro výše uvedené funkce

$$b_1 = \frac{1}{m} \times \left[S_1 - b_2 b_3^{x_1} \frac{1 - b_3^{m \times h}}{1 - b_3^h}\right]$$

Vzorec 18.: Koeficient b_1 pro výše uvedené funkce

Veličiny značené písmenem S s indexy 1 – 3 udávají sumy vysvětlované proměnné v jednotlivých 3 skupinách o m prvcích. Tyto sumy získáme rozdílně pro jednotlivé funkce.

- Sumy S_1 až S_3 pro modifikovaný exponenciální trend:

$$S_1 = \sum_{i=1}^{m} y_i \; ; \; S_1 = \sum_{i=m+1}^{2m} y_i \; ; \; S_1 = \sum_{i=2m+1}^{3m} y_i$$

Vzorec 19 -21: Výpočet sum modifikovatelného exponenciálního trendu používaných ve vzorcích 15 – 17

- Sumy S_1 až S_3 pro logistický trend:

$$S_1 = \sum_{i=1}^{m} \frac{1}{y_i}; \; S_1 = \sum_{i=m+1}^{2m} \frac{1}{y_i}; \; S_1 = \sum_{i=2m+1}^{3m} \frac{1}{y_i}$$

Vzorec 22 -24: Výpočet sum logistického trendu používaných ve vzorcích 15 - 17

- Sumy S_1 až S_3 pro Gompertzovu křivku:

$$S_1 = \sum_{i=1}^{m} \ln y_i \; ; \; S_1 = \sum_{i=m+1}^{2m} \ln y_i \; ; \; S_1 = \sum_{i=2m+1}^{3m} \ln y_i$$

Vzorec 25 -27: Výpočet sum Gompertzovy křivky používaných ve vzorcích 15 - 17

Pro to, aby bylo možné výše uvedené sumy spočítat, je třeba docílit toho, aby se daly všechny prvky proměnné y rovnoměrně rozdělit do tří skupin. Pokud tedy nemáme počet prvků dělitelný třemi, je třeba toho dosáhnout odebráním hodnot dle vhodnosti buď zleva, či zprava, to však záleží na charakteru analyzované události. Je také nutné dodržet stejnou vzdálenost proměnné x mezi jednotlivými prvky. Z výše uvedeného pak vyplívá, že sumu S_1 budeme počítat z první skupiny prvků, která jich bude obsahovat právě m = n/3, S_2 z následující skupiny o stejném počtu prvků a sumu S_3 z poslední skupiny prvků.

Výběr regresní funkce

Pro srovnání dat bude třeba správně zvolit regresní funkci. Pro ověření správnosti volby funkce budu využívat reziduálního součtu čtverců, kterým se zjišťuje, jak těsně jsou data „osázena" kolem křivky zvolené regresní funkce, platí tedy pravidlo, že čím menší je reziduální součet čtverců, tím lépe byla regresní funkce zvolena.

Reziduální součet čtverců získáme jako sumu kvadrátů rozdílu původní hodnoty a hodnoty vyrovnané pro každé y_i (1, str. 85).

$$S_R = \sum_{i=1}^{n} \hat{e}_i^2 = \sum_{i=1}^{n} (y_i - \hat{\eta}(x_i))^2$$

Vzorec 28: Reziduální součet čtverců

Nevýhoda reziduálního součtu čtverců spočívá v tom, že sice dokáže určit optimální regresní funkci, není však schopen vystihnout, jak moc „trefně" byla funkce zvolena. Proto budu používat i index determinace, který nabývá hodnot $I^2 \in < 0; 1 >$,

přičemž hodnota 1 značí, že 100% rozptylu vysvětlované proměnné je vysvětleno modelem a 0 % zůstalo nevysvětleno a naopak. (zdroje 6 a 1, str. 102).

$$I^2 = \frac{S_{\hat{\eta}}}{S_y} \qquad I^2 = 1 - \frac{S_{y-\hat{\eta}}}{S_y}$$

Vzorec 29 a 30: Index determinace

Index determinace tedy získáme z průměru součtu druhých mocnin odchylek vyrovnaných hodnot od průměru původních dat, který podělíme průměrem ze součtu druhých mocnin odchylek zadaných hodnot od jejich průměru. (1, str. 103).

Jsou ovšem případy, kdy průběh funkce logicky neodpovídá vývoji dat. Jedná se například o případy, kdy se bude zkoumat prodejnost zájezdů. Jako časová jednotka bude zvolen 1 rok. Hodnota závisle proměnné y pak nikdy nemůže nabývat záporných hodnot, v tomto konkrétním případě nepřipadá v úvahu vyrovnání pomocí regresní přímky, spíše sáhneme mezi nelinearizovatelné funkce, jakými může být například logistický trend při stoupající tendenci.

1.2 Teorie k programování

Pro vytvoření mé bakalářské práce jsem zvolila, jako stěžejní kámen, prostředí ASP.NET, které je součástí .NET Frameworku a programovací jazyk C#. Jako podpůrné prostředky pro doladění detailů pak využiji:

- Pro vytvoření grafického prostředí kaskádové styly CSS.
- Pro interaktivní ovládací prvky na straně klienta JavaScript.
- Pro zlepšení použitelnosti projektu jazyk SQL, který umožní pracovat s databázemi.

1.2.1 .NET Framework

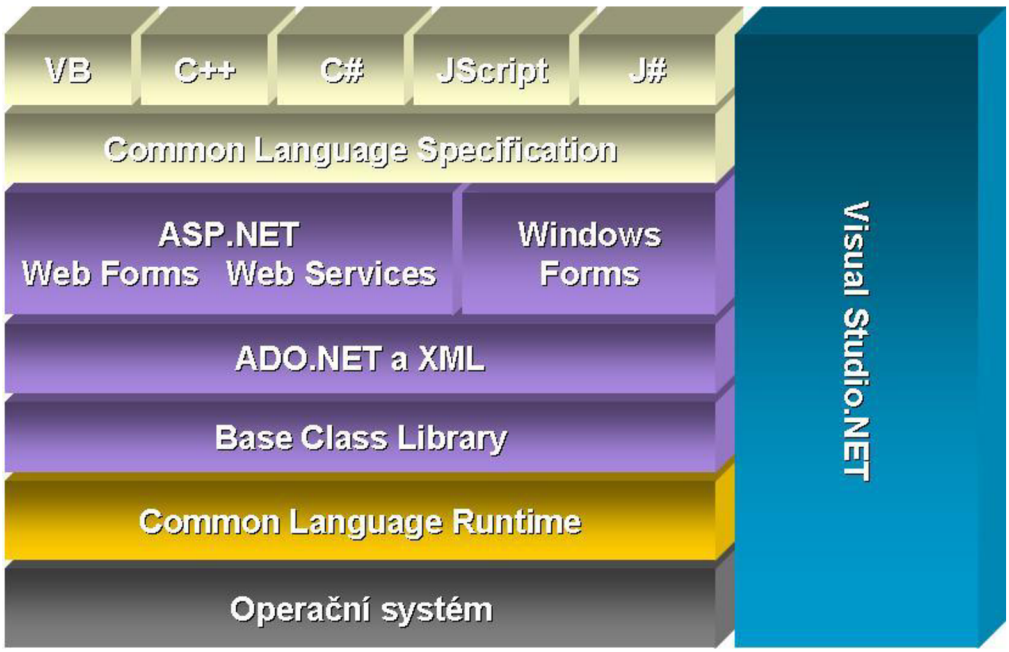
.NET Framework je systém vyvíjený společností Microsoft nejen pro tvorbu aplikací pro Microsoft Windows, Windows Mobile a jiné, ale i pro tvorbu webových aplikací pomocí prostředí ASP.NET.

Architektura .NET Frameworku se skládá ze základního Common Language Runtime (CLR), který např. spravuje paměť a provádí bezpečnostní kontroly a obsahuje mnoho užitečných funkcí, jako třeba možnost korektně ošetřovat chyby pomocí výjimek anebo garantuje, že veškeré operace budou prováděny nad korektními přípustnými datovými typy. Dalším charakteristickým rysem je správa paměti. O tu se stará tzv. garbage collector, který uvolňuje zdroj paměti tak, aby nedocházelo k jejímu nadbytečnému užívání.

Architektura .NET se dále skládá z BCL, neboli Base Class Library, což je základní knihovna pro přístup a využití funkcí operačního systému nebo také pro přístup k souborovému systému na discích.

Dalším prvkem architektury .NET je ADO.NET (ADO = ActiveX Data Object), který poskytuje funkce pro práci s databázemi, pracuje s objekty, které získá z databáze, tudíž není třeba při práci připojení ke konkrétní databázi.

Pro vytváření aplikací pak dělíme další vrstvu na Win Forms, pomocí které se vyvíjí aplikace pro operační systémy Microsoft, a ASP.NET, což je technologie, která se využívá pro vytváření webových aplikací.



Obrázek 1 : Architektura .NET - zdroj 3

Architektura .NET je architektura podporující mnoho programovacích jazyků, jako je například C#, C++, Visual Basic (jazyky deklarované firmou Microsoft) a např. Perl, jako jeden ze zástupců integrovaných jazyků třetích stran a mnohé další. Tato „multijazyčnost“ je umožněna díky překladu do universálního jazyka – intermediate language, ke kterému dochází v části nazývané CLS neboli Common language specification. Tato část architektury obsahuje pravidla, podle kterých se jednotlivé jazyky překládají do kódu intermediate language. (3, 5, str. 58, 10).

1.2.2 Vývojové prostředí ASP.NET

Jednoduchost programování webových aplikací spočívá také v možnosti naprosté separace kódu na jednotlivé funkční celky. Tato vlastnost je pro ASP.NET klíčová. Programátor má též velmi zjednodušenou práci tím, že mu ASP.NET poskytuje často užívané objekty od základních, jako jsou label nebo textbox přes jednoduché check boxy, buttons, a jejich listy až ke složitějším – např. wizard. Další možnosti pak představují doinstalovatelné moduly. Já budu například využívat komponentu umožňující jednoduchou práci s grafy.

1.2.3 Programovací jazyk C#

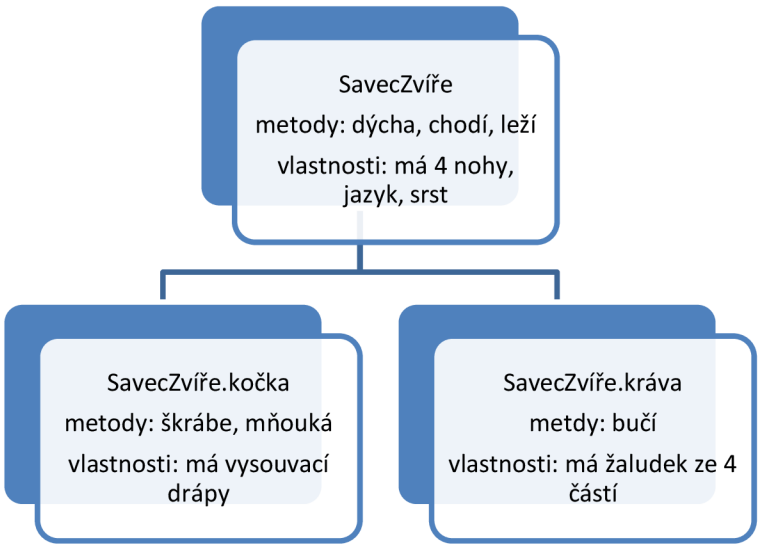
Programovací jazyk C# je objektově orientovaný jazyk vyvinutý jako odpověď společnosti Microsoft na jazyky C++ a Java, ze kterých se snaží brát to nejlepší, s „podporou jednoduchosti“ jazyka Visual Basic. Většina vlastností tohoto jazyka už je deklarována přímo samotnou .NET architekturou.

Základní myšlenkou OOP je nejen v jazyce C# to, že jednotlivé objekty reprezentují prvky reálného světa. Základem pro OOP (objektově orientované programování) v jazyce C#, jakožto i v ostatních objektově orientovaných jazycích jsou 3 následující vlastnosti:

- Dědičnost.
- Zapouzdřenost.
- Polymorfismus.

Dědičnost

Nejen v jazyce C#, ale napříč všemi objektově orientovanými jazyky platí, že každý objekt je zařazen do hierarchie tříd. Pokud bude objekt patřit do třídy, která má v hierarchii nad sebou další třídy, bude patřit i do tříd nadřazených. Každý objekt je definován svými vlastnostmi a metodami. Musí platit, že veškeré vlastnosti a metody definované nad objektem musí platit i na objekty pod tímto objektem. Stejně tak, pokud budeme mít instanci – konkrétního zástupce libovolné třídy, musí být možno ho správně zařadit v hierarchii. To se dá jednoduše ověřit otázkami „is –a?“ a „has –a?“, jak předvedu na níže uvedeném příkladu z běžného života.



Obrázek 2: Hierarchie objektů v rámci OOP – vlastní tvorba

Mějme instance – „Micka“ – zrzavá kočka a Kráva Milka – zástupce tříd objektů. Podle výše uvedeného můžeme tvrdit následující:

- Micka je instancí třídy SavecZvíře.kočka a tedy i třídy SavecZvíře.
- Kráva Milka je instancí třídy SavecZvíře.kráva a tedy i SavecZvíře.
- Jak pro Micku, tak pro Milku musí platit vlastnosti SavecZvíře a musí být umožněno použití metod definovaných pro třídu SavecZvíře.

„Is – a“: Je kočka zvíře? Ano.

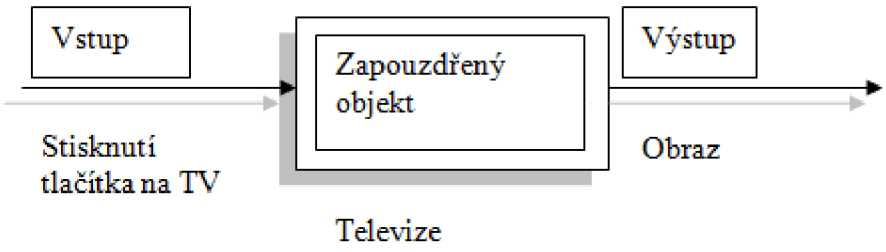
	Dýchá kočka?	Ano.
	Chodí kráva?	Ano.
„Has – a“:	Má kráva 4 nohy?	Ano.
	Má kočka srst?	Ano.

Dědičnost je tedy užitečná převážně pro zjednodušení a větší transparentnost při práci s objekty.

Zapouzdření

Velmi užitečná a oblíbená vlastnost objektově orientovaného programování, vzhledem k tomu, že před uživatelem „tají“ co všechno musí ta která metoda udělat, aby se stalo to, co uživatel očekává. Pokud tuto vlastnost aplikuji opět na situaci běžného života, může to vypadat následovně:

Modelová situace: Chci si pustit televizi – stisknu tlačítko ON na přístroji, dále mě vůbec nemusí zajímat, co se právě s televizí děje (z fyzikálního hlediska), zajímá mě až konkrétní výstup – tedy obraz a to je právě zapouzdření objektu.



Obrázek 3: Abstrakce zapouzdření - vlastní tvorba

A pokud budu chtít uvést konkrétní případ z programování na třídě Math[1]: Chci zaokrouhlit číslo na dvě desetinná místa – `ZaokrouhlenaHodnota = Math.Round(DeklarovanaPromenaJakoDouble,2);` – funkce s dvěma „vstupy“ – parametry, kterými jsou číslo, které chceme zaokrouhlit a počet desetinných míst a

[1] Jedná se o funkci zaokrouhlující číslo na daný počet desetinných míst

výstupem zaokrouhlenou hodnotou – nevím, jak to metoda Math.Round dělá, protože je zapouzdřená, podstatné je, že vždy udělá, co má.

Polymorfismus

O pojmu polymorfismus hovoříme například při dědění metod. Pokud máme pro nadřazenou skupinu objektů definovanou metodu, která má něco udělat, může mít pro objekty v hierarchii níže jiný výsledek. Pokud bychom u příkladu se zvířátky měli pro objekt SavecZvire definovanou metodu UdělejZvuk, pak při volání této metody dědicem, tedy SavecZvire.Kocka, by metoda "vydala" zvuk "mňau", při volání SavecZvire.Krava by tento zvuk byl "Bůů". Polymorfismus má však několik podob. Výše zmíněná byl základní polymorfismus. Další důležitou obdobou je parametrický polymorfismus, kde do předdefinované metody mohou vstupovat parametry různého typu a v různém počtu. Česky tuto vlastnost metod nazýváme přetížení metody.

Pro ukázku jazyka C# jsem vybrala drobnou část kódu, který obsahuje for-cyklus řešící načítání dat z tabulky a počítající některé potřebné veličiny pro určení Indexu determinace při použití regresní přímky, z nichž je jedna v každém kroku ukládána do vlastní vytvořené struktury.

```
for (int i = 0; i < pocetDat; i++)
{
    y = (Convert.ToDouble(((TextBox)TabulkaHodnot.Rows[i].FindControl("txtHodnota")).Text));
    Sy = Sy + ((y - vyberovyPrumerY) * (y - vyberovyPrumerY));
    SYMinusMiSeStriskou = SYMinusMiSeStriskou + ((y - (RPB1 + (RPB2 * (i + 1)))) *
                                (y - (RPB1 + (RPB2 * (i+1)))));
    StrukturaProPriimku HodnotaProPrimku = new StrukturaProPriimku();
    HodnotaProPrimku.RozdilOdZadane = y - (RPB1 + (RPB2 * (i + 1)));
    HodnotaProPrimku.VyrovnanaHodnota = RPB1 + (RPB2 * (i + 1));
    HodnotyProPrimku.Add(HodnotaProPrimku);
}
```

Obrázek 4: Ukázka kódu psaného v jazyce C# - zdroj: vlastní tvorba

1.2.4 Kaskádové styly

Z anglického Cascading Style Sheets pochází zkratka CSS, která značí v dnešní době běžně používané metody pro stylování internetových stránek. Kořeny těchto metod sahají někdy do poloviny devadesátých let, kdy vznikaly první internetové stránky, které byly zaměřeny čistě na obsah, ne na formu.

Moderní doba si však vyžádala pokrok a dnes je tomu mnohdy naopak. Design stránek má větší vliv, než kvalita jejich zpracování. Právě k designování stránek jsou kaskádové styly užitečným prostředkem. Umožňují nám:

- Striktně oddělit kód stránky od jejího designu – zpřehlednění celého kódu.
- Nadefinovat design každé skupiny elementů či každého elementu zvlášť v jednom kroku a tím sjednotit celkový vzhled stránky (například jednotný vzhled všech tabulek – namísto toho, abychom v kódu definovali vzhled každé tabulky zvlášť, nastavíme pro všechny tabulky jednotný vzhled jediným zápisem v souboru).
- Jednoduše a hlavně rychle změnit design určitých prvků či skupin prvků.

Momentálně je nejaktuálnější verze kaskádových stylů CSS 3. (8, 9).

Kaskádové styly jsou v ASP.NET využívány obdobným způsobem jak u klasických HTML a XHTML dokumentů, což vychází z toho, že výsledkem kódu napsaného pomocí technologie ASP.NET je vygenerovaná HTML stránka, tedy do kódu stránky je vygenerován odkaz na CSS soubor, který výslednou HTML stránku nastyluje. Stylovat ale lze i přímo značkami v kódu. Není nezbytné používat externí soubor.

1.2.5 Databáze a jazyk SQL

Každá databáze obsahuje tabulky, které nazýváme relace. Tyto relace slouží k uchovávání entit. Jako entitu si můžeme představit libovolný objekt z reálného světa – např. člověka. Pro každou relaci musí být zvolen primární klíč, který musí být jednoznačný a musí od sebe odlišit všechny entity v rámci relace. Relace jsou mezi sebou svázány vazbami, které mohou být v poměru 1:1, 1:N, N:1 nebo N:M, při čemž vazba 1:1 se v normálním životě téměř nevyskytuje, příkladem může být např. rodný list a člověk – každý člověk by měl mít pouze jeden. Nejčastěji se ovšem vyskytuje vazba N:M, kterou je třeba dekomponovat, aby ji bylo možné modelovat v databázi. Tyto vazby jsou normovány pomocí normálních forem. Databáze tedy slouží k zachytávání dat a vztahů mezi objekty v reálném světě.

Pro užívání databází se využívá mnoha odnoží jazyku SQL – Structured Query Language neboli v češtině strukturovaný dotazovací jazyk. Tento jazyk nese příkazy jak pro tvorbu databází, tak pro práci s nimi.

Jazyk SQL řadíme mezi tzv. deklarativní jazyky – nevyužíváme ho přímo samostatně, nýbrž jeho příkazy vkládáme do kódů jiných, procedurálních jazyků.

1.2.6 JavaScript

JavaScript je skriptovací programovací jazyk, jehož hlavním rysem je běh na straně klienta, na rozdíl např. od PHP nebo ASP, které běží na straně serveru. Tento rozdíl spočívá v tom, že pokud si uživatel prohlíží internetové stránky, tak v případě PHP je z klientského počítače odeslán požadavek na serverový počítač, tam je zpracován a zpět na klientské PC se vrací pouze výsledky, které vidíme zobrazené, právě na rozdíl od JavaScriptů, které pracují na straně klienta – tedy uživatelem je odeslán požadavek na server, na klientský počítač je serverem odeslána HTML stránka včetně JavaScriptů, které jsou následně zpracovány na klientském počítači browserem - prohlížečem.

```
<script type="text/javascript">
    function JSTrend() {
        if (document.getElementById('<%= CHBTrend.ClientID%>').checked) {
            document.getElementById('<%= LabelTrend.ClientID%>').style.display = 'inherit';
            document.getElementById('<%= TextBoxTrend.ClientID%>').style.display = 'inherit';
        }
        else {
            document.getElementById('<%= LabelTrend.ClientID%>').style.display = 'none';
            document.getElementById('<%= TextBoxTrend.ClientID%>').style.display = 'none';
        }
    }
```

Obrázek 5: Příklad javascriptu z bakalářské práce[2] - zdroj: vlastní tvorba

JavaScript tedy není plnohodnotný programovací jazyk, nýbrž skriptovací jazyk používaný jako doplněk k dalším jazykům. Využívá se převážně při psaní webových stránek, pokud chceme s nějakým prvkem na stránce dynamicky pracovat např. skrývat před uživatelem některé objekty, rozpohybovat obrázky nebo vytvořit interaktivní menu.

[2] Funkčnost tohoto javascriptu spočívá v tom, že pokud uživatel zatrhne určitý CheckBox označen jako CHBTrend na stránce, dojde k zobrazení prvků LabelTrend a TextBoxTrend, pokud CheckBox na stránce zatržen nebude, prvky zůstanou i nadále skryty

Pokud se na JavaScript podíváme z historického hlediska, tak jeho původ nalezneme u firmy Netscape v druhé polovině devadesátých let. V souvislosti s .NET je používán název JScript – verze pro firmu Microsoft. (4).

2 ANALÝZA PROBLÉMU A SOUČASNÉ SITUACE

V této kapitole popisuji momentální stav statistického balíčku firmy Pear spol. s r.o., který nabízí jako součást řešení komplexního souboru aplikací pro cestovní kanceláře a prodejce zájezdů.

2.1 Funkčnost současného systému

Produkt nabízený firmou Pear spol. s r.o. se v této době omezují pouze na statické zobrazování dat, uložených v databázích a jejich grafickou prezentaci. Protože poskytování statistických funkcí rozhodně není stěžejním produktem, je jejich úroveň prakticky bezvýznamná.

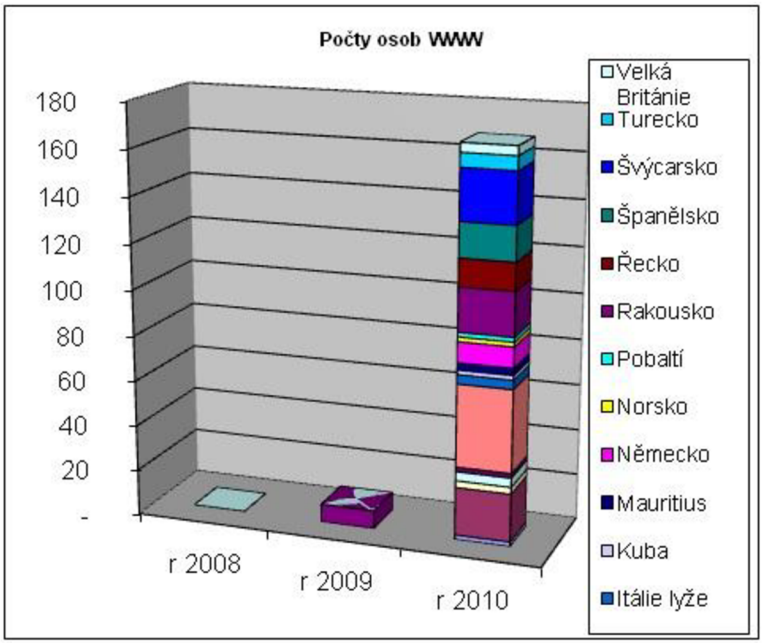
Pro analýzu současného řešení mi firma poskytla na otestování jejich vytvořený produkt, ve kterém byla zobrazena data od některých cestovních kanceláří. Běžně v provozu se tato data načítají z databáze, do které jsem ale neměla přístup.

Pod pojmem balíček či systém nástrojů je myšlen vytvořený soubor v aplikaci MS Excel 2003, který obsahuje několik listů s nástroji pro zobrazení dat z různých úhlů pohledu. Pro vytvoření těchto listů bylo pro ovládací prvky použito jazyka Visual Basic. Jedná se o listy s názvy:

- Země.
- Prodejci.
- Obsazenost.
- Slevy.
- Pracanti.
- Pohyby.
- Prodeje.
- Souhrn.
- Období země.
- Prodeje 2.

Na listu Země se uvádí objem prodejů rozčleněn jak podle jednotlivých navštěvovaných destinací, tak podle způsobu objednání zájezdu. Je možné si zobrazit zájezdy, které byly prodány pouze přes webové stránky cestovní kanceláře, nebo na jednotlivých pobočkách a kancelářích, nebo provizními prodejci. Poslední možností je zobrazení všech těchto skupin dohromady. Mimo sumárního číselného zobrazení tento

list nabízí i grafické znázornění po časových složkách – rozdělení celkového objemu prodeje podle zemí sloupcovým grafem, spojnicový graf pro jednotlivé destinace a sloupcový graf pro jednotlivé destinace.



Obrázek 6: Ukázka zobrazení dat stávajícím produktem[3] - zdroj: Aplikace Excel printscrean

List prodejci poskytuje zobrazení objemu prodejů vyjádřených v peněžních jednotkách členěných podle jednotlivých prodejců. Tyto údaje můžou být stejně jako u předchozího listu zobrazeny ve stejně formátovaných grafech, jako na listu Země.

Využití objednaných kapacit je zobrazeno na listu Obsazenost, který se skládá z velkého množství grafů, přičemž každý graf náleží jedné destinaci. Tyto grafy jsou rozděleny podle dat, každému datu připadá jeden sloupec, který se skládá z kapacit volných, koupených, rezervovaných a propadlých rezervací.

V oddílu Slevy je analyzován objem slev oproti cenám beze slev. Slevy jsou rozděleny podle měsíců a podle jednotlivých let. Data jsou zobrazena tabulkou a spojnicovým grafem.

Posledním listem, který alespoň částečně analyzuje data je list Období země, který je kombinací prvních dvou listů. Zobrazuje data určená pravděpodobně pro

[3] rozdělení celkového objemu prodeje prodaných zájezdů přes webové stránky cestovní kanceláře působící na českém trhu podle zemí

provizní prodejce, protože umožňuje rozdělení prodejů jednotlivých destinací na základě let a jednotlivých cestovních kanceláří.

Na listech Pracanti, Pohyby, Souhrn a Prodeje 2 jsou zobrazeny pouze data o prodejích bez jakéhokoliv porovnání či práce s nimi. Jedná se pouze o určitou evidenci, která by mohla být zdrojem dat pro další listy.

3 POPIS IMPLEMENTACE APLIKACE

V této kapitole dopodrobna popíši webovou aplikaci, kterou jsem navrhla a jejímž hlavním účelem je určit nejlepší řešení pro vyrovnání dat časové řady, data vyrovnat, přepočítat, určit prognózy a také určit její charakteristiky a požadované výsledky graficky zobrazit.

3.1 Řešení

Webová aplikace se skládá z jediné webové stránky, která je rozdělena na 3 části:

- Část přihlašovací a registrační.
- Část obsahová.
- Část Menu.

3.1.1 Část přihlašovací

Část přihlašovací je řešena pomocí dvou oddílů, jednoho, který obsahuje popisky v labelech a druhého, kde jsou dva TextBoxy pro zadání jména a hesla. U pole, kde se zadává heslo je nastaven typ TextBoxu na „password“, což v praxi znamená, že vyplněný text se nebude zobrazovat, místo něj se zobrazí nahrazující znaky.

Po zadání jména a hesla a stisknutí tlačítka přihlásit jsou data odeslána na server k porovnání s údaji v databázi. Při úspěšném přihlášení se místo tlačítka přihlásit zobrazí tlačítko odhlásit.

3.1.2 Část obsahová

Tato část je řešena pomocí pokročilého ovládacího prvku ASP:NET zvaného Wizard, neboli průvodce. Jeho výhody spočívají v integrovaných funkcích, se kterými si programátor nemusí dělat starosti. Při jeho využívání jsou k dispozici mnohé ovládací prvky, jako například navigační tlačítka Další a Zpět, automatické odkazy v jednotlivých krocích, které uživateli umožní nedodržovat posloupnost mezi ovládacími prvky, mnohé předdefinované styly apod.

Wizard se využívá nejčastěji v situacích, kdy chceme po uživateli, aby vyplnil formulář – v té chvíli je lepší ho nezahltit informacemi a požadavky, ale rozdělit mu je do nějakých kroků. Jeho výhoda spočívá také v tom, že není třeba pro každý krok vytvářet vlastní webovou stránku. (2, str. 632 - 640).

Já jsem si požadavky na uživatele rozdělila do sedmi jednoduchých kroků, kde se mu snažím přesně objasnit, co po něm chci, aby nedošlo k nepochopení programu.

Krok 1: Vítejte

Prvního[4] kroku Wizardu jsem využila pouze k uvítání uživatele a letmému představení celé aplikace. Text jsem formátovala do Labelu – neboli popisku. Mimo představení aplikace tento Label obsahuje i návrh k registraci uživatele.

Krok 2: Zadání počtu dat časové řady

V tomto kroku uživatel pouze zadá počet dat, která obsahuje jeho časová řada. V informační poznámce, která se zobrazí při najetí na obrázek otazníku, jsou uvedeny podmínky pro data, která jsou uvedená v této práci na straně 10. Tyto pomocné otazníky se vyskytují v průběhu celé aplikace a usnadňují uživateli pochopení, co má dělat. Poznámky jsou řešeny vložením obrázku otazníku, který má nastaven text, který se má zobrazit. Toto řešení je aplikováno na všechny pomocné otazníky. Další alternativou řešení by bylo vytvořit JavaScript pro událost najetí nad obrázek. Já zvolila tu jednodušší metodu bez JavaScritpu.

Krok 3: Základní informace o časové řadě

Ve třetím kroku zjišťuji od uživatele základní informace o časové řadě, a to jakého formátu jsou data, kde má uživatel výběr z RadioButtonListu[5] z následujících možností: rok, pololetí, čtvrtletí, měsíc a jiné. Jako přednastavená hodnota je rok z důvodu, aby nedošlo, že žádná z alternativ nebude zvolena.

[4] Ve skutečnosti se nejedená o první krok, nýbrž o krok nultý. V ASP.NET je číslování prvků od nuly, na rozdíl třeba od Pascalu, kde na nulté pozici byly vždy uloženy informace o „objektu“ .
[5] RadioButtonList je prvek, který se využívá pro výčet prvků, ze kterého si uživatel může vybrat pouze jednu alternativu.

Další požadovanou informací je typ časové řady. Uživatel má opět k dispozici RadioButtonList s alternativami „intervalová“ a „okamžiková“. Předvolená je možnost „intervalová“, opět z důvodu, kdyby uživatel zapomněl a nezvolil z možností, aby bylo možné provést výpočet. Uživateli je opět vysvětleno, o co se jedná při najetí na obrázek otazníku.

Krok 4: Zvolení početních úkonů

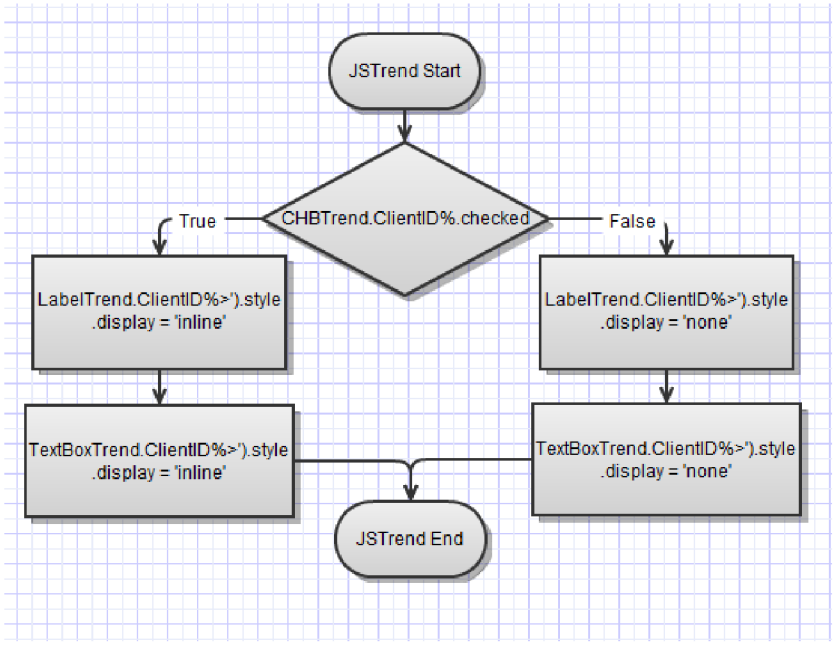
V tomto kroku již uživatel volí, co všechno se bude se zadanými daty dít, co bude požadovat od aplikace, aby počítala a zobrazovala. Tyto výběry jsou uživateli umožněny pomocí CheckBoxů[6]. V tomto kroku se tedy nachází CheckBoxy pro zvolení:

- Výpočtu charakteristik časové řady.
- Výpočtu trendu časové řady.
- Zvolení metody výpočtu Trendu časové řady.
- Zobrazení rovnice přímky.
- Zobrazení Indexu determinace.

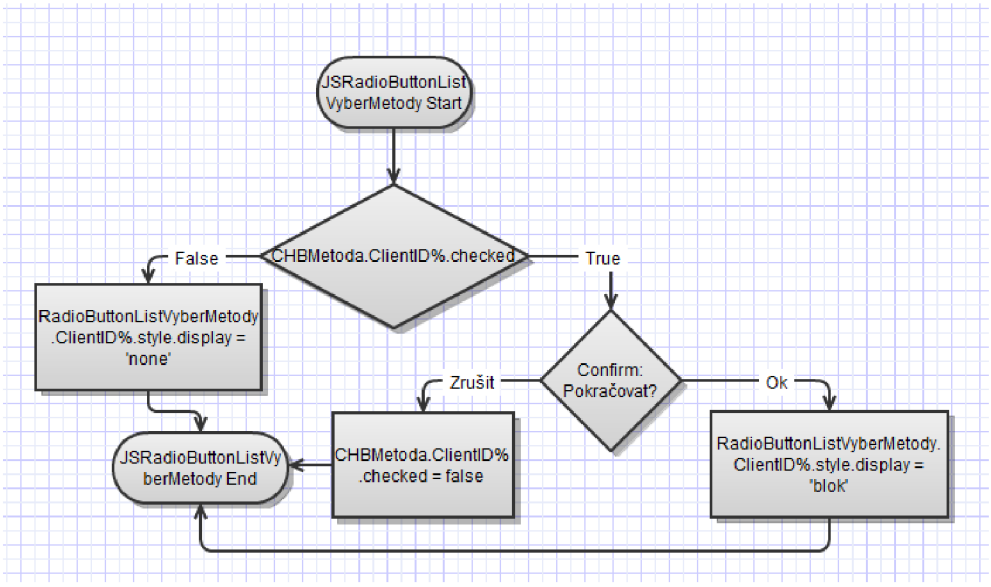
Při zatržení ChcekBoxu uvozujícího Výpočet trendu časové řady je uživatel dotázán pomocí JavaScriptu, pro kolik období, chce stanovit prognózu způsobem, který popisuje diagram na straně 33 nahoře (Obrázek 7).

Při zatržení CheckBoxu uvozujícího Zvolení metody výpočtu trendu je uživatel upozorněn na nutnou odbornost a pokud odsouhlasí pokračování výběru, je mu zobrazen opět RadioButtonList s možnostmi: „Regresní přímka“, „Modifikovaný exponenciální trend“, „Logistický trend“ a „Gompertzova křivka“. Řešení JavaScriptem je zobrazeno na obrázku č. 8.

[6] CheckBoxů v CheckBoxListu, na rozdíl od RadioButtonListu může být zatržen libovolný počet



Obrázek 7: Vývojový diagram Skriptu JSTrend - zdroj: Vlastní tvorba[7]



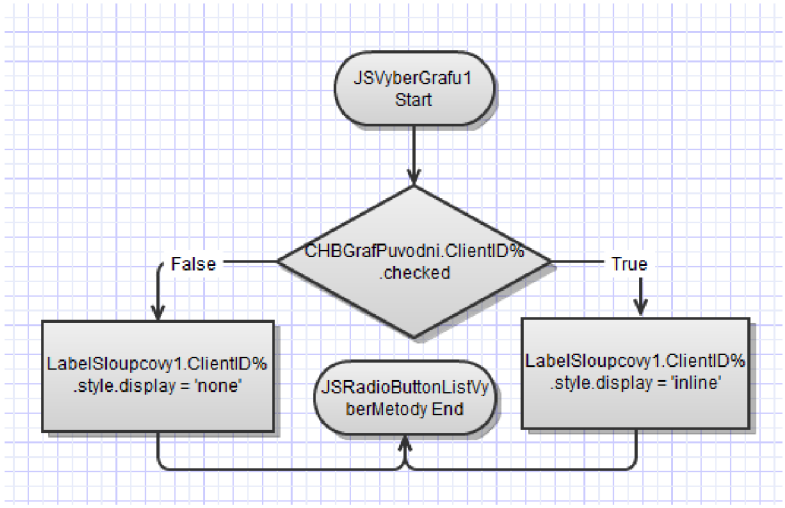
Obrázek 8: Skript pro výběr metody výpočtu trendu uživatelem - zdroj: Vlastní tvorba[7] str. 35

Krok 5: Zvolení grafů a jejich typů

V pátém kroku je uživatel dotázán, které grafy chce zobrazit. Při zatržení libovolné ze tří možností (Graf původních hodnot a vyrovnaných hodnot, Graf změn původních a vyrovnaných hodnot, Graf rozdílu mezi vyrovnanými a původními

[7] Všechny vývojové diagramy, včetně následujících byly vytvořeny zkušební verzí on-line programu Gliffy, dostupného na internetových stránkách www.Gliffy.com

hodnotami) se mu zobrazí další možnost výběru zatržení typu grafu. Typy grafů jsou sloupcový a spojnicový, je tedy možné celkem zobrazit až 6 grafů. Schéma JavaScriptu pro výběr grafu původních hodnot a vyrovnaných hodnot je zobrazen na obrázku (Schémata pro zbylé dva typy grafů nezobrazuji, neboť jsou z hlediska funkčnosti totožné).



Obrázek 9: Skript pro výběr grafu a jeho typu - zdroj: Vlastní tvorba[7] str. 35

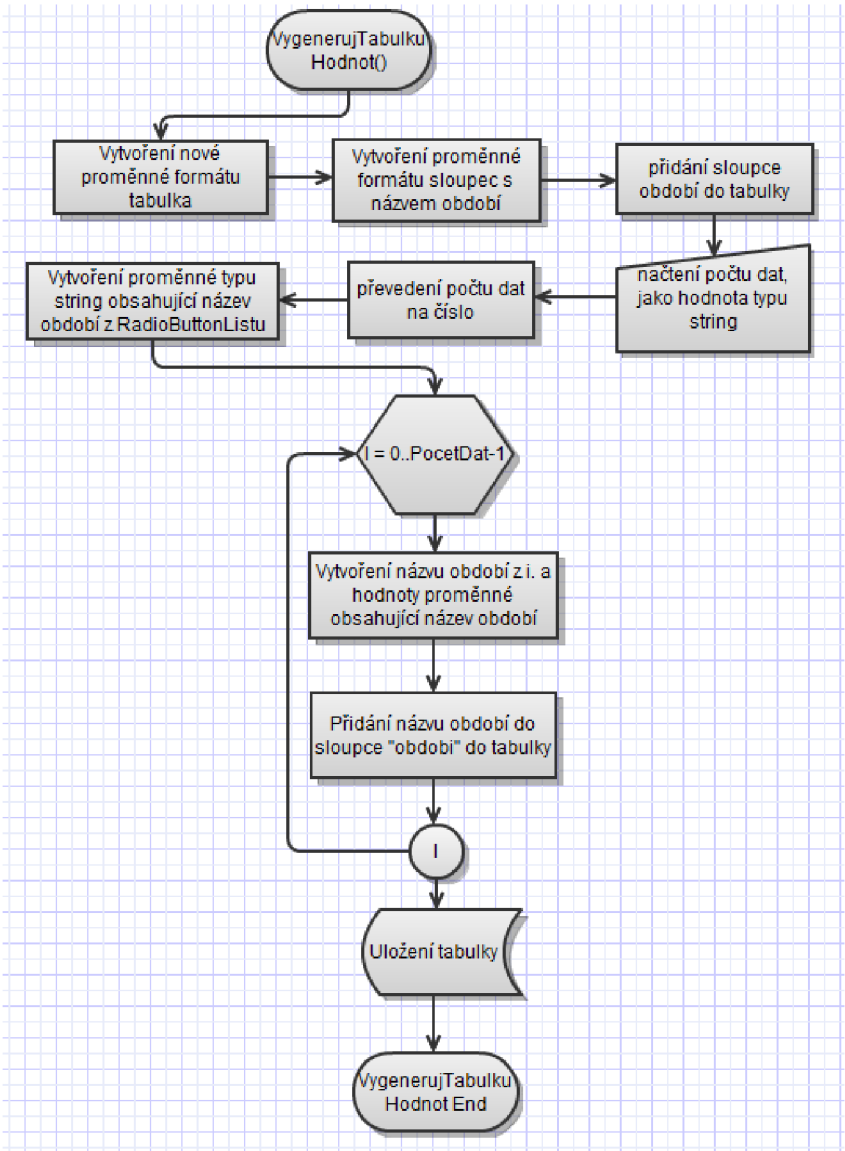
Krok 6: Zadání dat

Šestý krok je určen pro samotné zadání dat. Na základě počtu dat, zadaných ve druhém kroku se vygeneruje tabulka o dvou sloupcích. V první sloupci jsou uvedeny názvy období a jejich pořadové číslo, do druhého sloupce se pak zadávají samotná data. Generování a zobrazování tabulky je usnadňuje ASP.NET možností využití předdefinovaných ovládacích prvků pro provázání dat. Jedním z takových prvků je prvek GridView, který jsem používala. Nejprve je nutné si tento prvek vytvořit v HTML kódu a následně ho propojit s kódem programu. V mém případě k proceduře VygenerujTabulkuHodnot. Generování této tabulky a vytvoření názvů období je zobrazeno diagramem na obrázku č. 10. TextBoxy pro zadávání dat jsou přidány v šabloně pro tuto tabulku v části definující HTML.

V tomto kroku zároveň dochází k tzv. validaci dat, kdy se kontroluje, zda jsou vyplněny hodnoty pro každé období. Druhý validátor kontroluje, zda jsou zadané hodnoty opravdu čísla. Pokud se uživatel pokusí odeslat neplatně vyplněnou tabulku,

aplikace ho na tuto skutečnost upozorní a nepustí ho dál, dokud toto neopraví. Validátory jsou poskytovány jako komponenta ASP.NET.

Na konci stránky je uživatel požádán, pokud počet dat nebyl dělitelný třemi (zjišťuje se algoritmem před zobrazením tabulky pro zadání dat) o zvolení, ze které strany má dojít k ořezání dat.



Obrázek 10: Vytvoření tabulky pro zadání dat - zdroj: Vlastní tvorba[7] str. 35

Krok 7: Kontrola zadání

V tomto kroku má uživatel možnost zkontrolovat si všechny svoje volby a na závěr je tlačítkem Další odeslat ke zpracování. Tlačítkem Zpět, které poskytuje, stejně

tak jak tlačítko Další, už výše zmíněný ovládací prvek Wizard, se uživatel dostane k úpravě údajů. Mezi jednotlivými kroky může zároveň přeskakovat pomocí hypertextových odkazů jednotlivých kroků.

Uživateli jsou zobrazeny všechny jeho volby, které zatrhl nebo označil v CheckBoxech a RadioButtonListech společně s tabulkou s vyplněnými daty, do které se nedají zapisovat změny. Pro změnu jakékoliv volby je třeba přejít do příslušného zadávacího kroku.

Pro kontrolu dat je opět vytvořen GridView, který je propojen s tabulkou nazvanou KontrolniTabulka. V této tabulce jsou uloženy názvy období stejným způsob, jako v tabulce pro zadání dat a ke každému období je vždy z tabulky pro zadání dat načtena příslušná hodnota.

Krok 8: Výpočty

Posledním a zároveň nejdůležitějším krokem, jsou samotné výpočty. V tomto bodě nejen, že dochází k vypočítání veškerých potřebných veličin, ale i k uložení sestavy do databáze nebo uložení výpočtů do souborů typu *.xls a *.pdf.

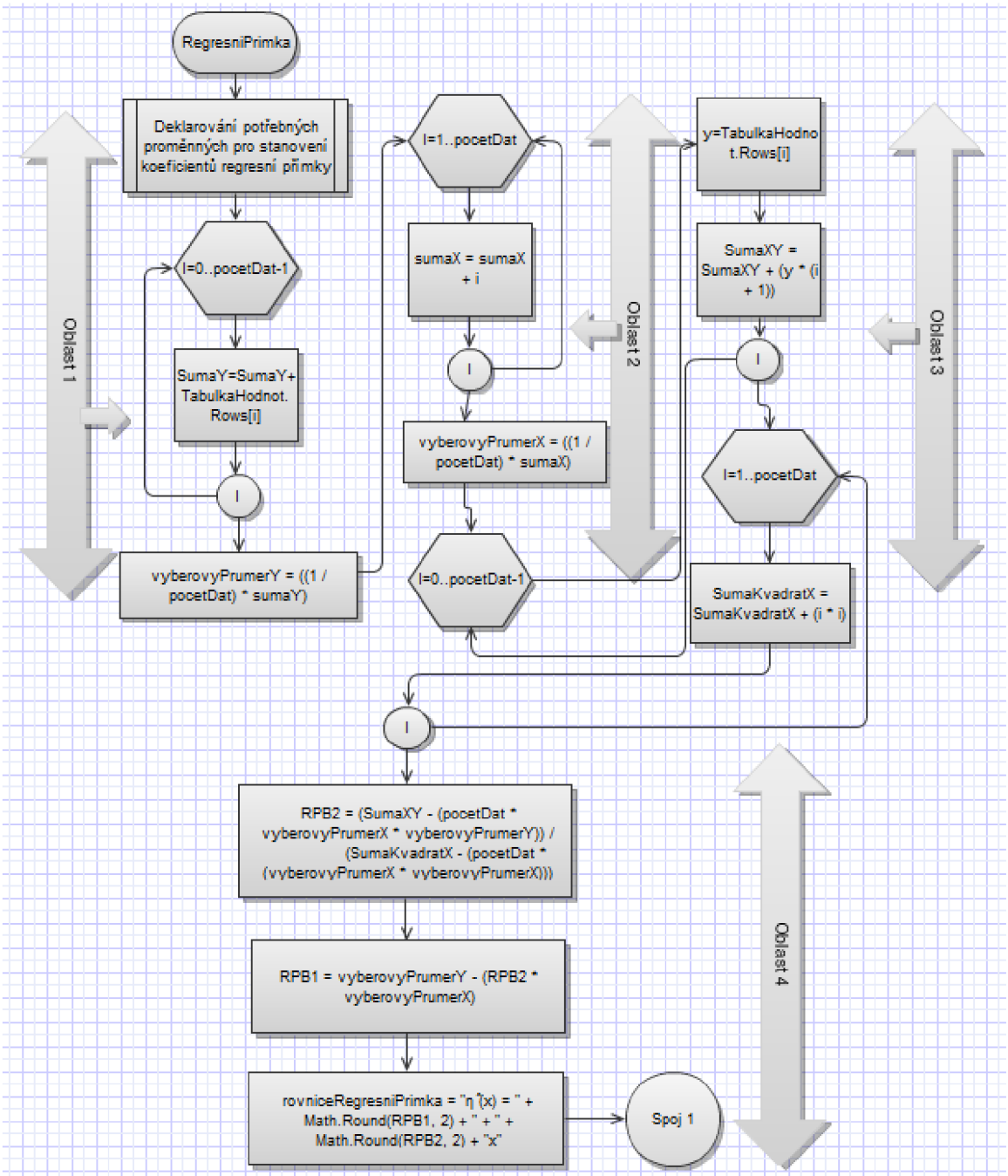
Výsledky se již nezobrazují v GridView, ale v rámci vytvořeném pro reporty v ReportViewveru.

Samotné výpočty jsou realizovány pomocí funkce UrceniKrivky, která má 4 vstupní parametry. Těmito parametry jsou výsledky 4 dalších funkcí a to RegresniPrimka, ModifikovanyExponencialniTrend, LogistickyTrend a GompertzovaKrivka. Tyto 4 funkce navracejí hodnotu, která je typu vytvořené struktury, kterou jsem si pojmenovala IndexAListHodnot. Každá z výše uvedených 4 funkcí tedy vstupuje do UrceniKrivky se strukturou obsahující proměnnou typu double, která v sobě nese hodnotu Indexu determinace pro danou křivku a proměnnou ListHodnot, což je opět struktura, která by se dala chápat, jako tabulka. Struktura ListHodnot je složena z hodnot dvou proměnných a to z hodnoty vyrovnané danou křivkou a z rozdílu mezi původní hodnotou a hodnotou vyrovnanou pro každé zadané období. Pokud si uživatel přál prognózu, nese v sobě část obsahující vyrovnané hodnoty i tyto prognózy.

Vybrání správné funkce se realizuje ve funkci UrceniKrivky, která má 4 vstupní parametry, což jsou struktury obsahující indexy determinace pro jednotlivé křivky. Pro vybrání správné funkce používám předdefinovanou funkci Math.Max, která vrací větší číslo ze dvou hodnot. Nejprve tato funkce porovná Index determinace pro Logistický trend s Indexem determinace pro Gompertzovu křivku, dále větší z těchto indexů porovná s indexem modifikovaného exponenciálního trendu a opět větší z těchto dvou indexů porovná s indexem determinace pro regresní přímku. Jedná se tedy o postupné určování maxima ze čtyř čísel. Pokud ale uživatel vybral sám křivku, kterou chce data vyrovnat, do funkce UrceniKrivky pak vstupuje pouze jeden parametr naplněn skutečným indexem determinace té křivky, kterou si uživatel vybral, jako zbylé tři parametry jsou poslány nulové hodnoty.

Pro určení správné křivky je samozřejmě potřeba stanovit vyrovnané hodnoty křivkou a určit jednotlivé indexy determinace. To se provádí ve čtyřech funkcích (RegresniPrimka, ModifikovanyExponencialniTrend, LogistickyTrend a GompertzovaKrivka), které mají téměř stejný průběh. V každé z funkcí nejdříve deklaruji proměnné zahrnující veškeré veličiny pro určení koeficientů regresní přímky, zahrnující y, výběrový průměr x, výběrový průměr y, dále suma x*y, suma x^2, suma x, suma y, proměnné pro koeficienty b a pro počet dat. Všechny tyto proměnné jsou typu double, poté následují samotné výpočty. Poté deklaruji proměnné potřebné pro výpočet indexu determinace pro danou křivku. Algoritmus pro výpočty jednotlivých koeficientů regresní přímky a indexu determinace je znázorněn na diagramech v obrázcích č. 11 a č. 12 níže. Pro výpočty koeficientů regresní přímky a Indexu determinace byly použity vzorce č. 8 - 12 a vzorec č. 30 z kapitoly 1.1.2 Regresní analýza.

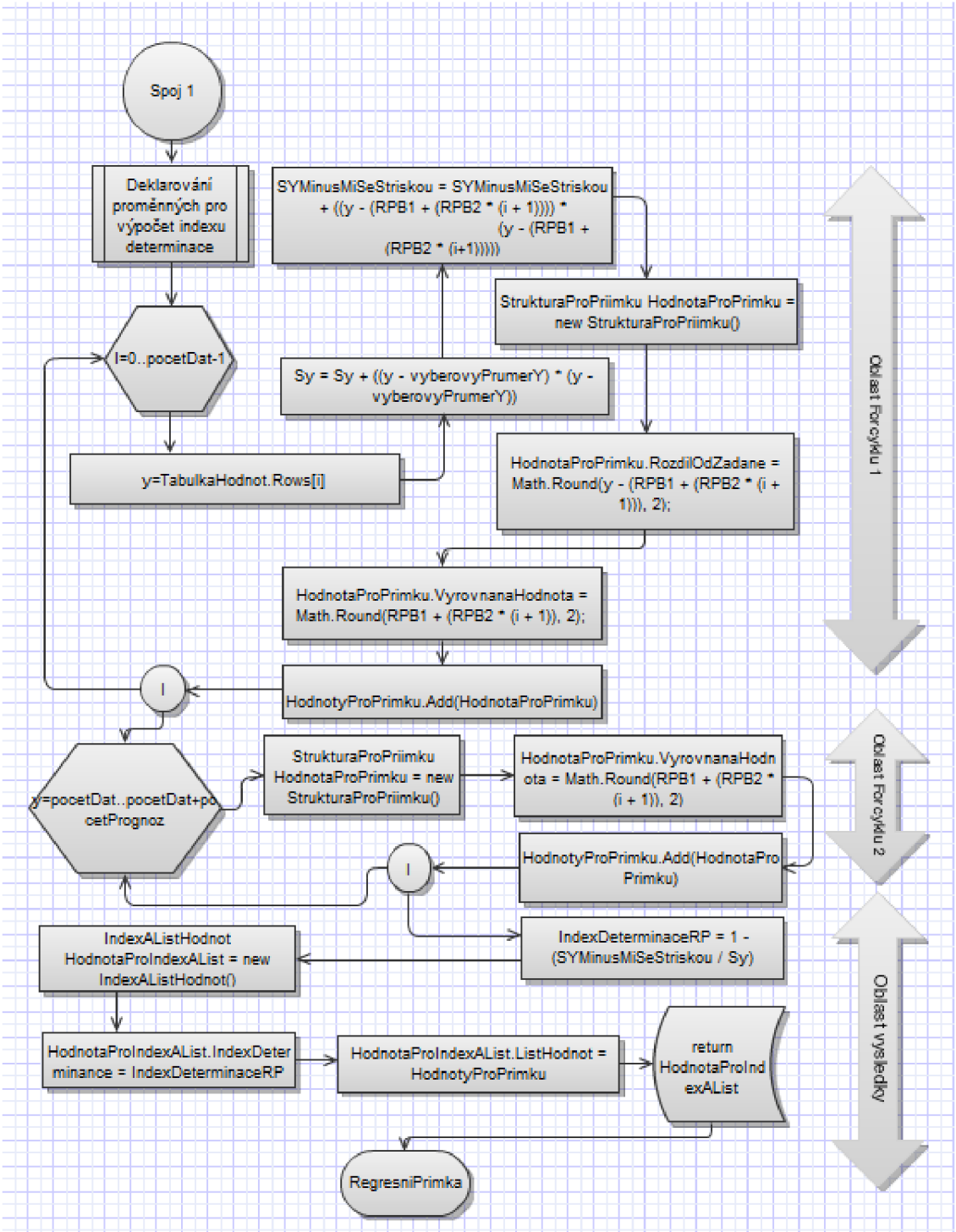
Diagram na obrázku č. 11 jsem rozdělila na 4 části. V první části je For cyklus, počítající celkovou sumu hodnot y, z tabulky vyplněné uživatelem. Z této sumy se po sečtení dat spočítá výběrový průměr dat. V oblasti 2 se počítá suma x z hodnoty řadící proměnné i. Třetí For cyklus v oblasti 3 počítá sumu x*y a čtvrtý For cyklus v téže oblasti počítá sumu x^2. Ve čtvrté oblasti se počítají už samotné koeficienty a vytváří se rovnice přímky. Math.Round(RPB1,2) je funkce, která zaokrouhlí číslo, v tomto případě koeficient RPB1 na dvě desetinná místa. Toto zaokrouhlení je využívané pouze pro zobrazení rovnice. Pro výpočty s rovnicí se používají přesné hodnoty koeficientů.



Obrázek 11: Část diagramu algoritmu vypočítávající koeficienty regresní přímky - zdroj: Vlastní tvorba[7] str. 35

Druhá část algoritmu zobrazena na obrázku č. 12 a je rozdělena na tři části. V první části se nejdříve spočítá suma druhé mocniny rozdílu mezi výběrovým průměrem a y, dále suma druhých mocnin rozdílu hodnoty y a jeho vyrovnané hodnoty. V dalším kroku je vytvořena proměnná typu struktura, nazvaná HodnotaProPrimku, do které se dále do oblasti RozilOdZadane a VyrovnanaHodnota přidávají příslušné

hodnoty. Nakonec jsou hodnoty z proměnné HodnotaProPrimku přidané do proměnné HodnotyProPrimku. Pro lepší pochopení by bylo možné vysvětlení, že máme tabulku, která se nazývá HodnotyProPrimku, každý řádek této tabulky se nazývá HodnotaProPrimku a hodnoty tohoto řádku pak RozdilOdZadane a VyrovnanaHodnota.



Obrázek 12: Část diagramu algoritmu vypočítávající index determinace - zdroj: Vlastní tvorba[7] str. 35

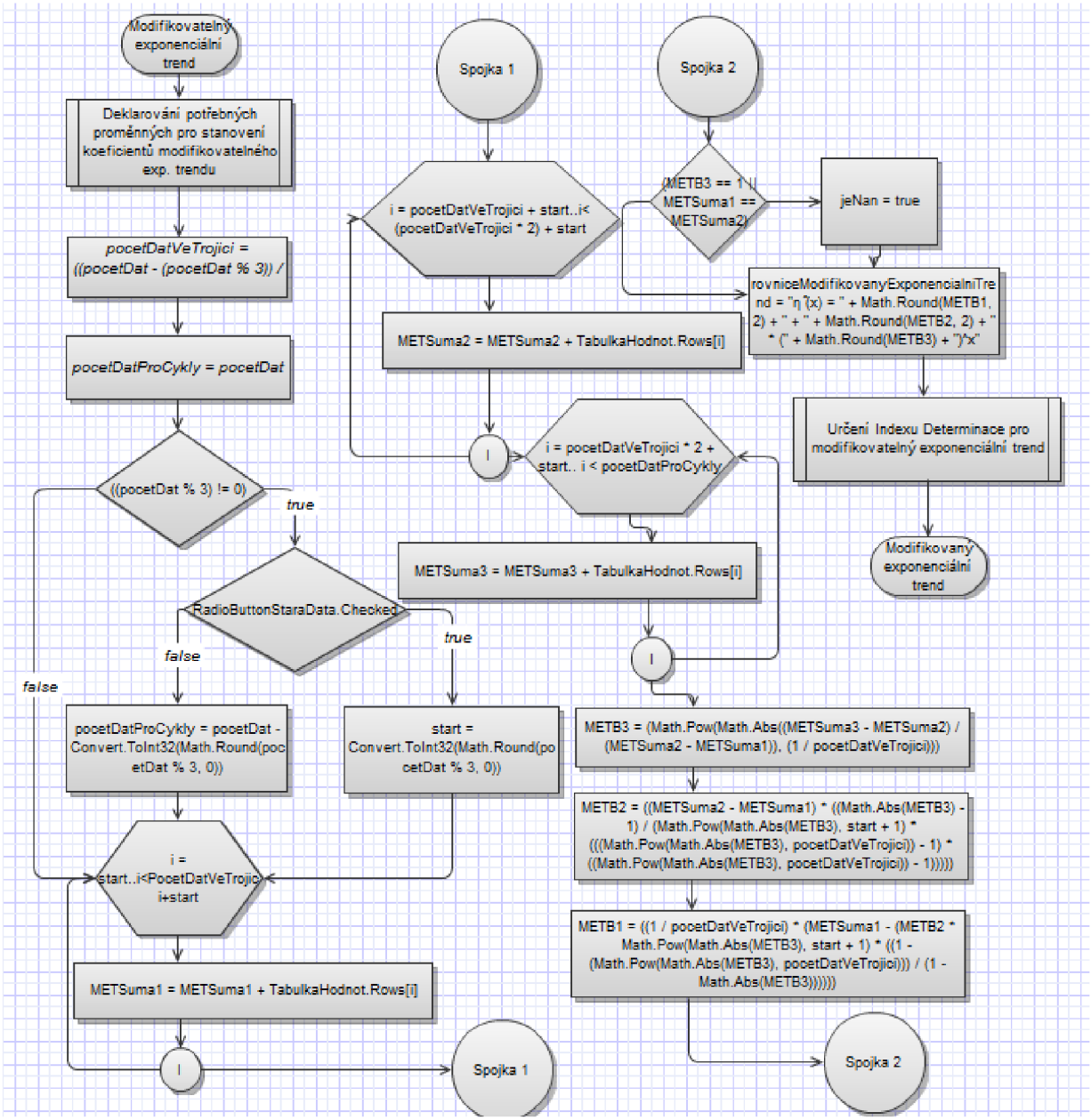
V druhé části je For cyklus, přidávající do proměnné HodnotyProPrimku vypočtené hodnoty prognózy pro tolik období, pro kolik si stanovil uživatel.

Ve třetí části se nejprve spočítá pomocí koeficientů index determinace, poté se do strukturovaného typu IndexAListHodnot nadefinuje proměnná HodnotaProIndexAList a do této proměnné uložené ve struktuře se vloží Index determinace pro regresní přímku. Celá tato proměnná je pak uložena do proměnné HodnotyProPřímku. Celá funkce navrací proměnnou HodnotaProIndexAList

Stejným způsobem jako regresní přímka jsou řešeny i nelinearizovatelné funkce. Rozdíl ve výpočtu je zobrazen obrázkem 13 níže, na kterém je diagram pro algoritmus počítající modifikovatelný exponenciální trend.

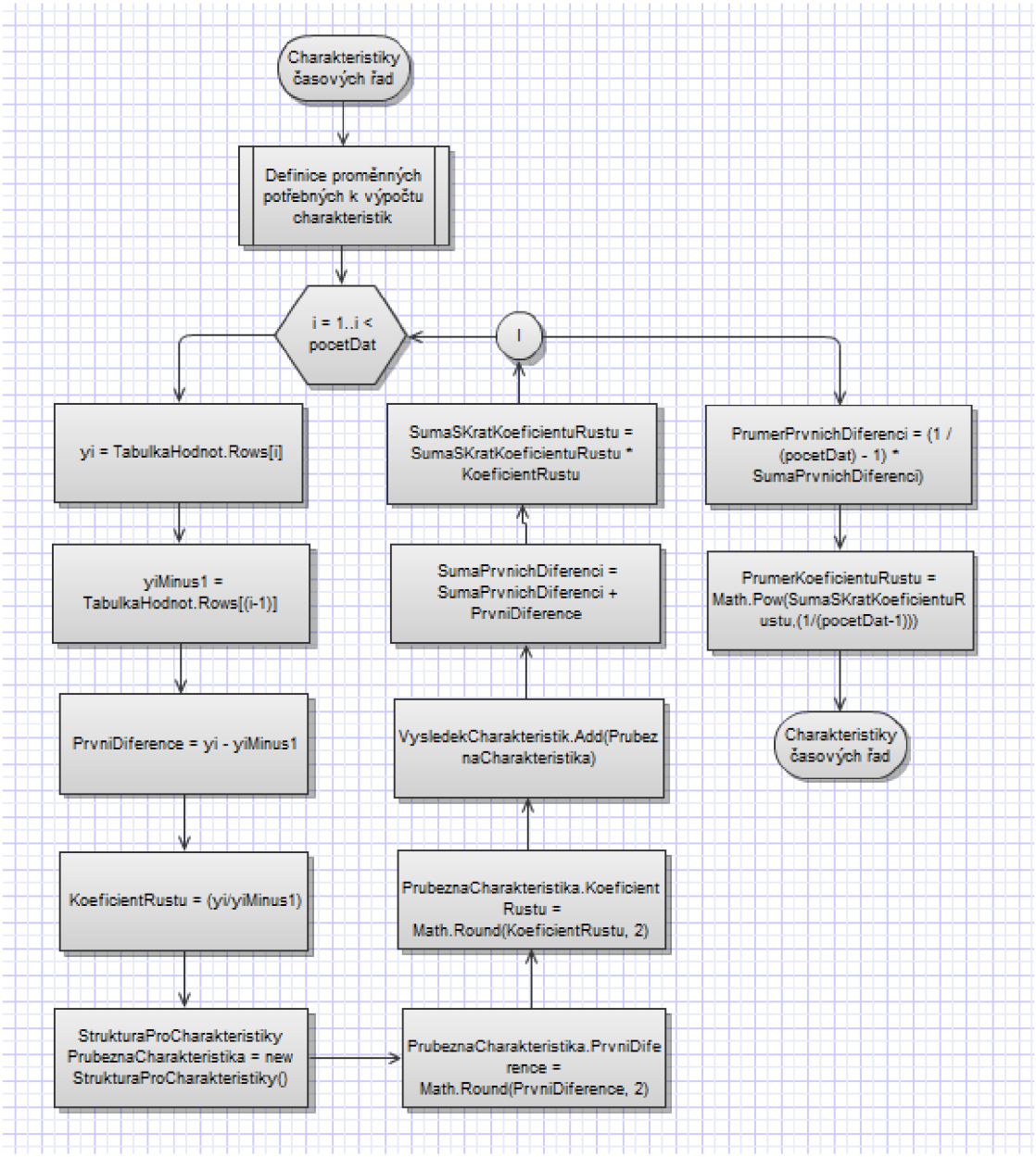
Nejprve je v části po Spojku 1 vypočítán počet dat ve trojici, a zadán rozsah počtu dat, dále je zjištěno, zda je počet zadaných dat dělitelný třemi beze zbytku, pokud ne, zjišťuje se, zda je vybrán RadioButton s možností stará dat – pokud ano, tak se dále pracuje s nejnovějšími daty, pokud ne, tak se pracuje s daty nejstaršími. Pokud byla dělitelnost beze zbytku potvrzena, pak se počítá s původním počtem dat ve trojici. Dále se pomocí tří For cyklů (jeden zobrazen v části po Spojku 1, další dva v části mezi Spojkou 1 a Spojkou 2) spočítají jednotlivé sumy, tak jak jsou popsány v kapitole 1.1.2 Regresní analýza vzorec 25 na straně 17[8]. V části po spojku 2 jsou ještě spočteny jednotlivé koeficienty modifikovaného exponenciálního trendu pomocí vzorců 16 - 18 na str. 17. Poslední část tohoto algoritmu, řešena pod spojkou dvě, byl problém, že při náhodném zadávání dat se občas podařila zadat taková kombinace, že vyšla nulová hodnota ve jmenovateli zlomku. Výpočet tedy nebyl proveditelný. Algoritmus provádí pro tyto případy kontrolu. Jedná se o možnost, kdy se Suma 1 a Suma 2 ve výpočtu křivky rovnají a když vyjde koeficient b3 roven jedné. Pokud dojde k jedné z těchto událostí, je tato skutečnost uložena do proměnné jeNan a této proměnné je pak využito při vypsání výsledků pro upozornění uživatele na neschopnost programu vypočítat jím požadované výpočty. Dalším krokem je již tvar samotné rovnice modifikovaného exponenciálního trendu a pak určení indexu determinace tak, jak už to bylo popsáno u obrázku č. 12.

[8] Sumy pro další funkce jsou popsány na straně 18, ty ale nezobrazuje tento diagram.



Obrázek 13: Modifikovatelný exponenciální trend - zdroj: Vlastní tvorba[7] str. 35

Dalším vykonávaným početním úkonem je výpočet charakteristik, pokud byl uživatelem vybrán. Charakteristiky časové řady zahrnují veličiny pospané v kapitole Časové řady a jejich výpočty se řídí vzorci č. 2 – 7 z téže kapitoly.

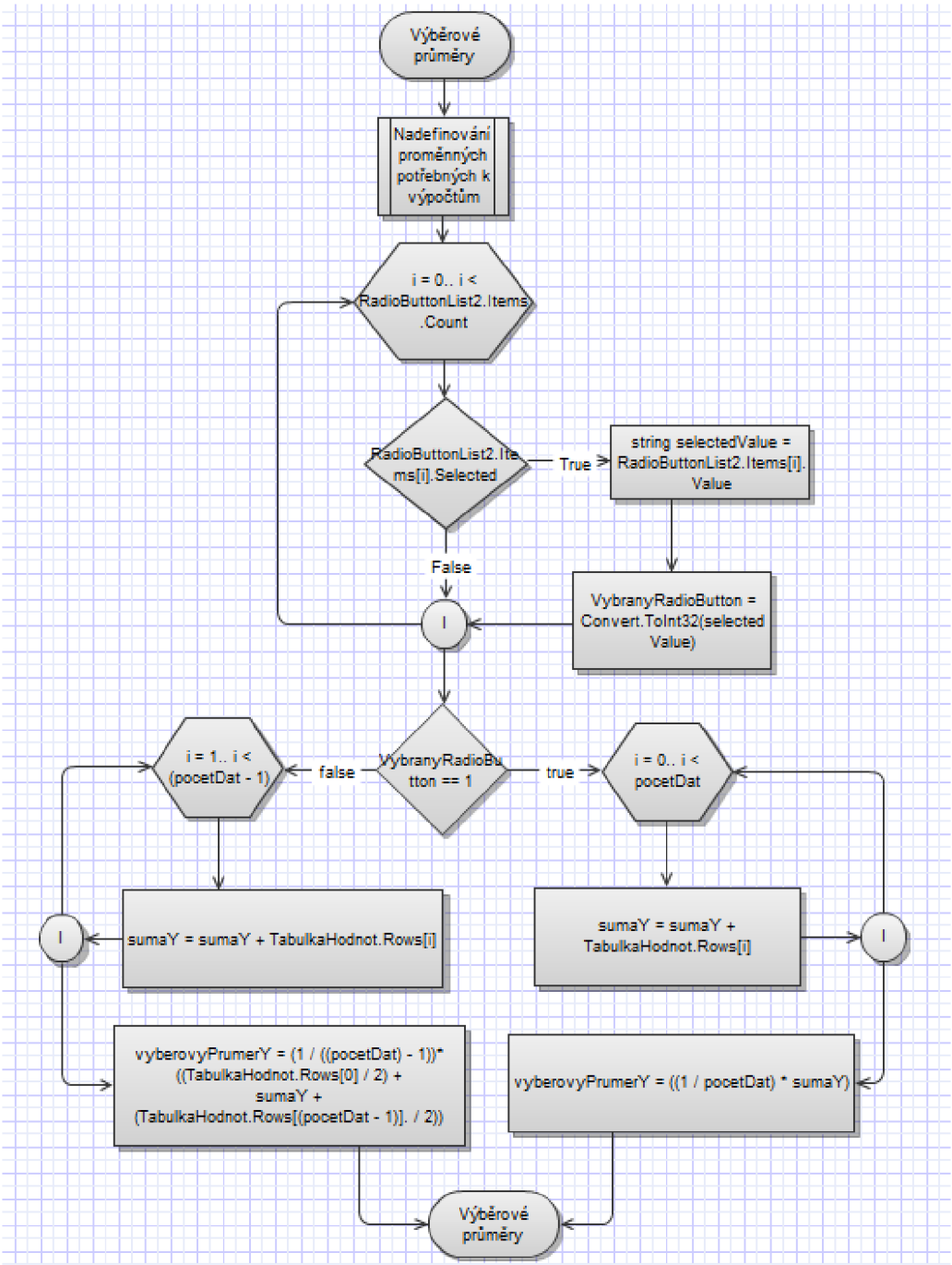


Obrázek 14: Diagram zachycující algoritmus počítající charakteristiky časové řady - zdroj: Vlastní tvorba[7] str. 35

Po nadefinování proměnných probíhá For cyklus, který nejdříve načte pořadovou hodnotu z tabulky, poté hodnotu předchozího období a z toho vypočítá první diferenci a koeficient růstu. Poté se definuje proměnná PrubeznaCharakteristika typu definované struktury, tento typ jsem pojmenovala StrukturaProCharakteristiky, do které se vkládají jak první diference, tak koeficient růstu zaokrouhlený na dvě desetinná místa. Poté se průběžná charakteristika musí uložit do proměnné typu list. Dalším krokem je spočítání sumy prvních diference a koeficientů růstu. V této chvíli zde for

cyklus, pokud splnil počet opakování, končí a ze sum se spočítají jejich průměry, které jsou uloženy v zaokrouhlené podobě na dvě desetinná místa do proměnné typu double.

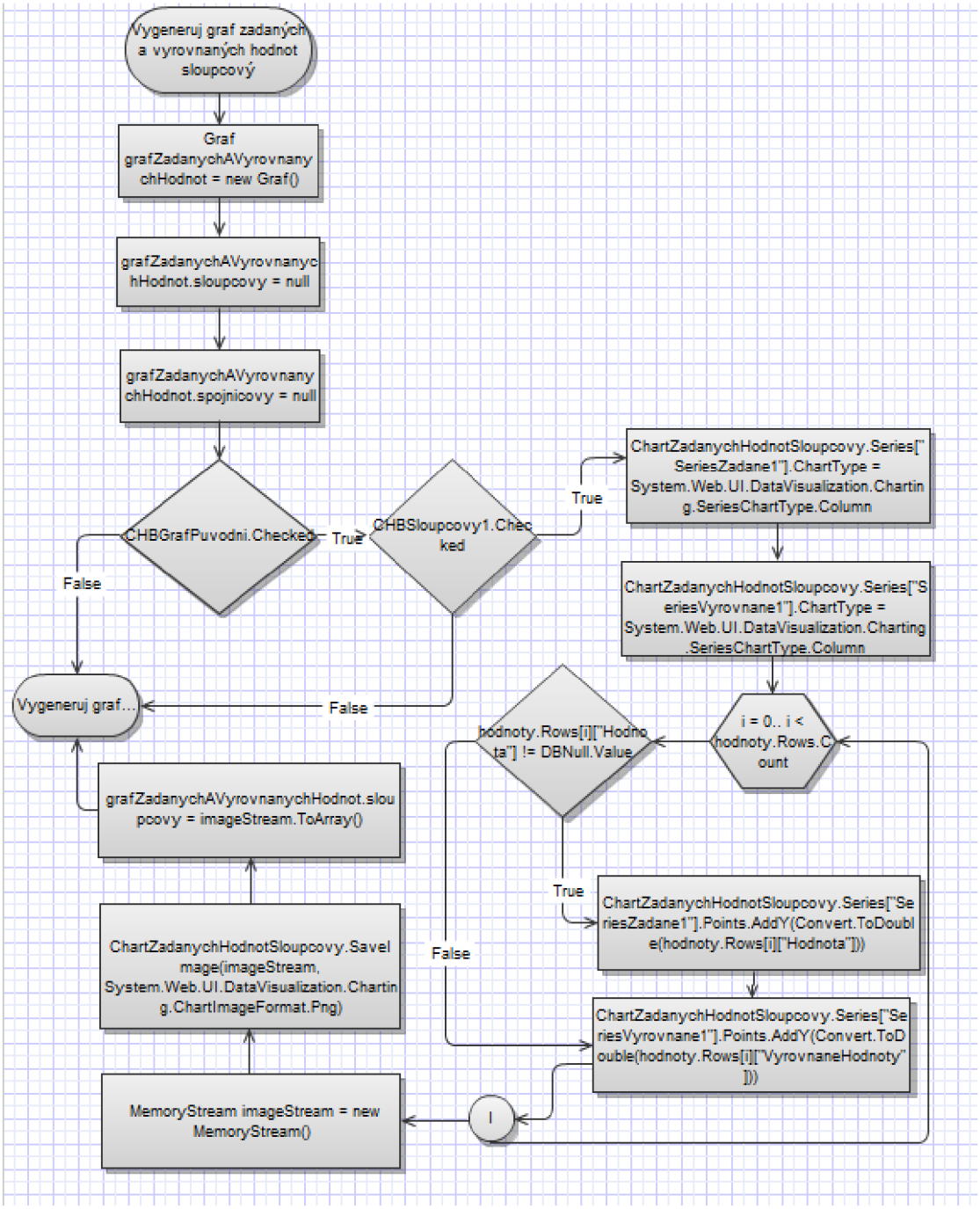
Další charakteristikou je výběrový průměr. Tento průměr má rozdílné výpočty pro typ časové řady intervalové a okamžikové. Průběh výpočtu je zobrazen na obrázku č. 15.



Obrázek 15: Diagram algoritmu počítající výběrový průměr - zdroj: Vlastní tvorba[7] str. 35

Nejprve jsem si opět nadefinovala všechny proměnné, které budu využívat. První for cyklus řeší, který RadioButton je při výběru typu časové řady zatržen. Pokud narazí na zatržený RadioButton, pak převede jeho hodnotu, která byla pro každou položku zadaná v HTML části, na číslo typu integer, pokud se nebude jednat o zatržený RadioButton, vrátí se cyklus na začátek nebo pokud byly zkontrolovány všechny položky RadioButtonListu, pak for cyklus skončí. Pro časovou řadu intervalovou byla v HTML části hodnota i nastavena na 1, pro okamžikovou na hodnotu 2. Pokud se vybraná hodnota rovná jedné, pak se provede for cyklem výpočet sumy hodnot y a po ukončení for cyklu výpočet aritmetického průměru. Pokud se hodnota nerovná jedné, spočítá se for cyklem hodnota sumy y, ale bez první a poslední zadané hodnoty a po jeho ukončení je spočítám chronologický průměr.

Další části k řešení bylo zobrazení výsledků v grafech. Nejprve bylo třeba si pomocí grafové komponenty, která je volně ke stažení na stránkách firmy Microsoft, vložit do HTML stránky oblasti pro grafy. V této části jsem nastavila pro každý graf velikost, popisky, název a design, také bylo potřeba docílit, aby se graf nezobrazoval přímo v HTML části, ale v části reportů. Grafy jsem tedy skryla nastavením hodnoty atributu „Visible“ na „false“. Dále jsem musela grafy naplnit daty. Tato část se definuje již v programové části. Nejdříve dojde ke kontrole, zda byl daný graf vybrán pro zobrazení uživatelem, pokud ano, tak se přidá řada grafu, které se nastaví typ na „column“ nebo „line“, tedy sloupcový či spojnicový, podle toho, který byl vybrán uživatelem. Poté se pomocí for cyklu prochází tabulka hodnot, což je proměnná typu vytvořená struktura, která se naplnila hodnotami předchozími výpočty, až do doby než se projde celá. Každá hodnota z příslušného sloupce je přidána jako bod grafu. Pro vložení grafu do repotu bylo zapotřebí uložit ho do paměti, tedy vytvořit pro něj místo v paměti, vygenerovat obrázek ze zadaných dat, v tomto případě ve formátu *.png a nakonec tento obrázek vložit do paměti. Úsek programu (funkce se vstupním parametrem, kterým je již zmiňovaná tabulka hodnot) řešící zobrazení sloupcového grafu vyrovnaných a původních hodnot je zobrazen diagramem na obrázku níže.

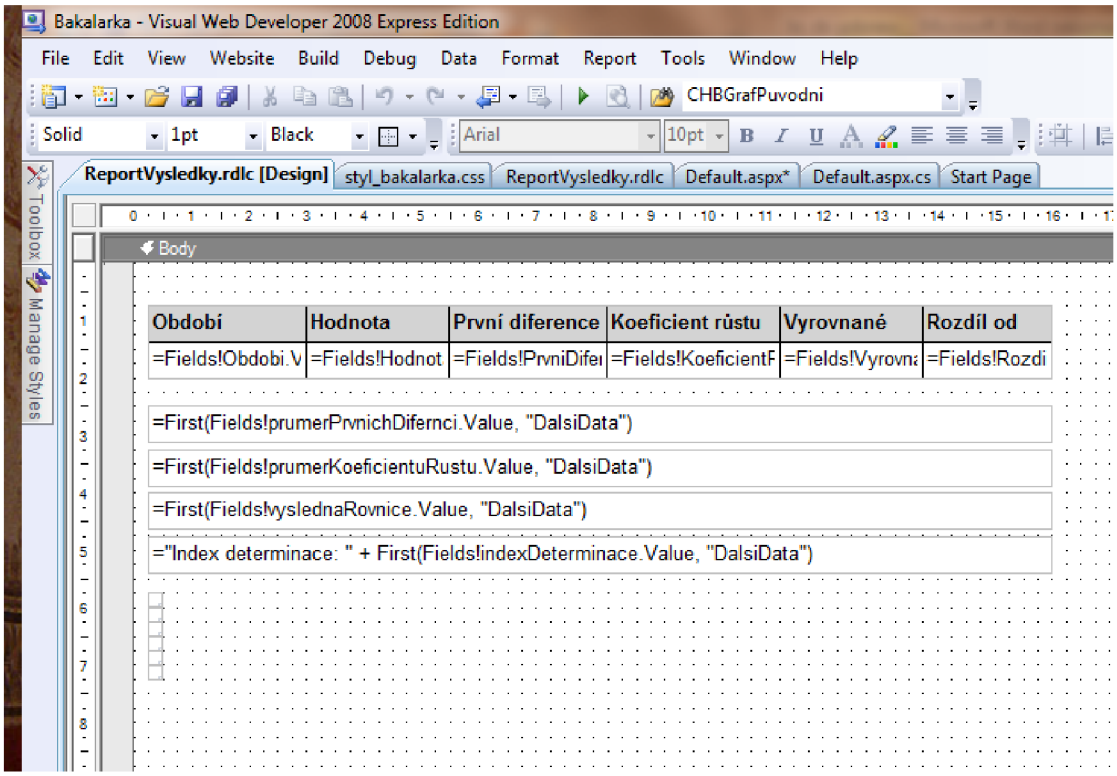


Obrázek 16: Diagram generování grafů - zdroj: Vlastní tvorba[7] str. 35

Poslední částí, kterou popíši, je export dat do souborů. Export dat do souborů je přímo podporován architekturou .NET pomocí reportů. Pro vytvoření reportů je třeba si založit soubor typu *.rdlc obsahující XML tagy, které definují budoucí vzhled reportu, zdroje dat pro jednotlivé oblasti a způsoby načítání těchto dat. Předdefinování vzhledu je zjednodušeno možností spustit si tento xml soubor v tzv. design módu, který je svým způsobem WYSIWYG (What you see is what you get – Co vidíte, to dostanete)

editorem. Pomocí tohoto „editoru“ jsem si vytvořila oblast tabulky, kam se načítají výsledky z předpočítané tabulky, dále oblast pro zobrazování dalších dat, ve které, pokud to uživatel vyžaduje, se zobrazují charakteristiky časové řady, rovnice křivky vystihující trend a index determinace a nakonec oblasti pro obrázky, které nesou grafy.

Všechny tyto oblasti v design módu jsou vidět na obrázku č. 17.



Obrázek 17: Úprava reportu v design - módu - zdroj: Print screan z aplikace Visual Web Developer 2008 Express Edition

3.1.3 Část Menu

Tato část je k dispozici pouze přihlášeným uživatelům. Data do této části jsou načítána z databáze. Nejprve je z databáze načteno datum uložení a název sestavy, a vedle těchto hodnot zobrazena ikonka pro soubor v MS Excel a Acrobat (pdf). Ke stažení dokumentu dochází až při kliknutí na jednu z ikonek z důvodů urychlení aplikace – kdyby se měli stahovat všechny uživatelovi soubory, mohlo by to aplikace zatížit a zpomalit.

3.2 Chyby v řešení

Veškeré chyby v řešení vycházeli z mojí programátorské nezkušenosti. Nejedná se přímo o chyby ve funkčnosti, ale spíše o usnadnění a manipulaci s jednotlivými mnou naprogramovanými metodami tedy, funkcemi a procedurami, kde jsem porušila jedno z nepsaných pravidel – a to každou metodu se snažit rozdělit na atomické kroky.

Moje funkce a procedury jsou velmi robustní a tudíž těžkopádné. Obsahují mnoho úkonů, které pak nejde využívat v dalších metodách, ve kterých by mohli být využity. Příkladem této chyby může být například výpočet výběrového průměru y, kde by bylo rozhodně lepším řešením vytvoření funkce pro tento výpočet, který by vracela jako výstup právě hodnotu výběrového průměru, než počítání výběrového průměru y v každé další funkci. Dalším takovým příkladem je výpočet koeficientů jednotlivých funkcí využívaných pro určení trendu. Základem řešení byl pouhý výpočet těchto koeficientů, avšak v průběhu vývoje aplikace se začaly vynořovat další potřeby hodnot z této funkce. Namísto atomizace řešení výpočtu jsem všechny další výpočty přiřadila do té samé funkce, ze které přejímám dopočítané hodnoty. V jedné funkci se tedy počítají jak koeficienty pro funkci, tak i Index determinace pro vyrovnání danou funkcí, charakteristiky časové řady, funkce dále předává hodnoty do tabulky, předává hodnoty pro graf atd. …a to vše i tehdy, pokud si tyto výpočty uživatel nepřeje. Výsledkem je nejen nepřehledný, ale i těžko upravovatelný kód, a pokud by došlo k nějakému většímu rozšíření kódu, mohlo by to ohrozit i rychlost funkčnosti celé aplikace.

4 PŘÍNOSY APLIKACE A MOŽNOSTI ROZŠÍŘENÍ

Celkové řešení je nakonec koncipováno spíše jako demo-projekt, které na to, aby mohlo být samostatně nabízeno jako produkt, potřebuje minimálně níže uvedené úpravy.

Bohužel dnes, kdy nejen na trhu cestovního ruchu vládne poměrně velká nejistota, a stále doznívají důsledky celosvětové hospodářské krize, které pocítili i cestovní kanceláře, nikdo nemůže zajistit či zaručit, že předpovědi takovýchto analytický nástrojů, jako je právě tato webová aplikace budou alespoň přibližně odpovídat možné skutečnosti. Je tedy nutné brát v úvahu, že veškeré výsledky poskytované mojí aplikací slouží prozatím pouze orientačně. Pokud by se při dalším vývoji podařilo zahrnout počítání i s těmito fakty, aplikaci by to rozhodně prospělo a zvětšilo její využitelnost. Otázkou zůstává, nakolik je vůbec možné tyto faktory předpovídat a jakým způsobem je dále zahrnout do výpočtu.

4.1.1 Přínos ke stávajícímu řešení

Stávající řešení firmy Pear spol. s r.o. spočívá hlavně v zobrazování dat. Moje aplikace by mohla s těmito daty dále pracovat a analyzovat je. K tomu by bylo zapotřebí buď předělat stávající řešení na formát webové aplikace, nebo sjednotit zdroj dat pro oba nástroje. Hlavní přidanou hodnotu vidím převážně v možnosti predikce budoucnosti, ač jenom orientační. Tyto odhady by mohli pomoct při objednávání kapacit dopravy, ubytování, při odhadu rozpočtů - tržeb a výdajů na další období nebo počet objednávek pro jednotlivé destinace apod.

4.1.2 Využitelnost firmou Pear spol. s r.o.

Přínos tohoto demo-projektu pro firmu Pear spol. s r.o. a jeho nynější možnosti využití spočívají převážně v těchto oblastech:

- Firma se ještě nerozhodla, zda nakonec bude investovat do dalšího vývoje tohoto projektu – může svým zákazníkům ilustrativně demonstrovat několik

příkladů využití a pokud by byl o tento analytický nástroj zájem, může pokračovat v jejím vývoji.

- Pokud by některý z jejich zákazníků projevil zájem o služby, které tato výsledná aplikace poskytuje, budou mít základ, na kterém se dá dále pracovat a určité minimální statistické know-how, v této bakalářské práci popsané.

- I v této podobě je produkt schopen základních analýz, tudíž by mohl být distribuován, jako základní manažerský nástroj pro podporu rozhodování, toto však nedoporučují minimálně do doby, než budou dodělány některé z níže popsaných rozšiřujících možností.

- V podniku mohou otestovat funkčnost tohoto softwarového řešení sami na sobě například při určování výdajů na příští rok.

4.1.3 Přínosy řešení pro mě

Tato bakalářská práce mi poskytla dostatek prostoru, pro lepší pochopení objektově orientovaného programování a získala jsem cenné znalosti jak o architektuře .NET a jejím využívání v dnešní době, tak i o programovacím jazyku C#, který je dnes velmi rozšířený a často využívaný v podnicích působících na trhu s vývojem softwaru.

Jako výhodu pro mne ještě musím uvést, že jsem s firmou Pear spol. s r.o. domluvená, při zájmu zákazníků, na pokračování vývoje.

4.1.4 Další možnosti využití

Aplikace je využitelná i jako nástroj pro výuku statistických metod ve škole, buď jako kontrolní prvek pro studenty a jejich spočítané příklady nebo při samotných výpočtech – to je dáno tím, že aplikace nejen že zobrazuje celkové výsledky, ale i poskytuje informace o mezivýpočtech – záleží pouze na tom, co si uživatel zvolí, že zobrazit chce a co vidět nechce. Pro tyto potřeby by aplikace mohla sloužit i v této podobě, v jaké ve stávající chvíli je.

Možnosti dalšího využití jsou teoreticky možné i v dalších odvětvích, nejen v cestovním ruchu a vzdělávání. Problémem však bude konkurenceschopnost tohoto produktu vyvíjeným prozatím jediným člověkem s profesionálně tvořenými softwarovými aplikacemi, které nejen že nabízí komplexnější analýzu dat, ale i více možností pro samotnou predikci trendů časových řad a případně i více prostoru pro práci s daty. Takovéto komplexní řešení nabízí například společnost TriloByte Statistical Software za 8280 Kč.[9]

4.2 Možnosti rozšíření

Pro využití aplikace v běžné praxi, by bylo třeba udělat následné opatřetní a dodělky, tak aby celá aplikace byla 100% funkční a její výpočty byly spolehlivé.

4.2.1 Rozšíření výpočetních možností

Při tvorbě aplikace jsem předpokládala, že časová řada je tvořena pouze hodnotami trendu a jejich rezidui (mezi rezidua řadíme nejen výkyvy, které nelze žádným způsobem předpokládat, ale i například lidskou chybu – v měření, v zaokrouhlování). Časovou řadu však můžeme rozkládat na více složek, jednou z nich, kterou by bylo vhodné při dekompozici zohlednit, a kterou by bylo možné přidat do této aplikace je sezónní složka, která reprezentuje pravidelné změny v rámci roku, které se periodicky opakují. O dekompozici časové řady více výše, str. 11 – 12.

Další možnosti rozšíření by měli spočívat v rozšíření možností výpočtů trendu. Jednou z alternativ by bylo přidání podpory linearizovatelných funkcí, kde by byla uživateli nabídnuta možnost zadání si vlastního tvaru funkce, kterou chce zadaná data vyrovnat. To by však vyžadovalo poměrně vysokou odbornost uživatele, se kterou nemůžeme dopředu počítat. Proto jsem tuto možnost sama do aplikace neimplementovala. Za další možnost bych zvažovala přidání takových metod výpočtů, které nevyužívají, jako hlavní myšlenky minimalizace reziduí. Takovou metodou jsou například metody klouzavých průměrů, které je vhodné využívat v případech, kdy se data vyrovnávají metodami nejmenších čtverců nepřesně. Metoda klouzavých průměrů

[9] Aktuální cena pro období duben 2010. Více informací o produktu na http://www.trilobyte.cz/QC-Expert/Linearni-regrese.html

je vhodná spíše k určení trendu, než pro prognózy do dalších let, protože její výsledky bývají často nepřesné.

4.2.2 Navázání na jednotlivé databáze cestovních kanceláří a poskytovatelů zájezdů

Vzhledem k tomu, že cestovní kanceláře mají informace vhodné k analýze například o prodejích a objednávkách uloženy v databázích, bylo by dobré umožnit budoucím uživatelům načítat data do aplikace přímo z těchto databází. V tomto případě je zadávání dat do formuláře v aplikaci neobratné a vytváří možnost pro vytvoření lidské chyby z nepozornosti, která pak může znehodnotit výsledky celé aplikace.

4.2.3 Import dat pro analýzu ze souborů

Dalším krokem pro vylepšení funkčnosti celé aplikace by bylo přidání načítání neboli import dat nejen z databáze, ale i ze souborů, například typu *.xls. Toto načítání by mohlo být využito v případě dalších subjektů, které nemají data uložena pouze v databázích.

4.2.4 Design

Momentální stav grafické úpravy neodpovídá dnešním trendům v designu internetových stránek. Proto bych doporučila buď grafikem vytvořit reprezentativní design na míru, tak aby byla vytvořená aplikace prezentovatelná budoucím potenciálním zákazníkům nebo alespoň navázat na firemní design a barvy, tak aby aplikace měla i prezentační význam pro firmu a zákazník si například pomocí firemního loga hrušky dokázal aplikaci asociovat s firmou Pear spol. s r.o. Optimálním řešením by mohla být kombinace obou způsobů, která by však pro firmu neměla být obtížná, vzhledem k tomu, že se v tomto odvětví sami pohybují.

4.2.5 Podpora vícejazyčnosti

Vzhledem k firemní klientele je pravděpodobné, že by aplikace našla uplatnění u uživatelů, kteří by nemuseli ovládat češtinu. Jednalo by se převážně o manažery cestovních kanceláří, které nejsou ve vlastnictví Čechů. Z tohoto důvod bych navrhla přidání modulu vícejazyčné podpory, kdy by si uživatel mohl vybrat jazyk. Doporučila

bych přidat minimálně angličtinu a slovenštinu (značná část zákazníků firmy Pear spol. s r.o. je ze Slovenska), popřípadě němčinu. Tento vícejazyčný modul by mohl být přidán až při zájmu o produkt takovým zákazníkem, kterým by byl požadován.

4.2.6 Možná drobná vylepšení

V této kapitole nastíním několik drobností, které jsem sama chtěla udělat, ale v důsledku nedostatku času mi na ně nezbyl prostor.

Názvy časových období

Prozatím jsou časová období tvořena pouze číslovkou určující pořadí v časové řadě a názvem období, která je bráno z vytvořeného Radiobutton Listu.

Období	Hodnota	Období	Hodnota
1. Rok	90953	1. Měsic	38576
2. Rok	71973	2. Měsic	29365

Obrázek 18: Generování názvů období - zdroj: Vytvořená aplikace

Je možné vytvořit algoritmu, který by se po vybrání formátu dat zeptal na hodnotu prvního období a na základě této hodnoty a ze zadaného počtu dat by vygeneroval přesné názvy období.

Volba počtu míst, na které se budou výsledky zaokrouhlovat

Dále mě napadlo umožnit uživateli volbu, na kolik desetinných míst chce mít výsledky zokrouhleny. Toho by bylo dosaženo tak, že by byl uživatel dotázán na počet desetinných míst, tato hodnota by byla uložena do proměnné a proměnná se předávala jako parametr funkce pro zaokrouhlování Math.Round.

Vylepšení grafů

Bylo by vhodné nastavit grafy tak, aby zobrazovali nejen průběh hodnot pro orientaci, ale i konkrétní hodnoty.

ZÁVĚR

Výsledkem této bakalářské práce je v praxi, s drobnými omezeními, využitelná webová aplikace, která používá základních principů analýzy trendů v časových řadách. Pro tato omezení jsem navrhla opatření, která jsou rozepsána v kapitole Možnosti rozšíření. Podařilo se mi vytvořit produkt, který se vyznačuje svoji jednoduchou ovladatelností a možností využívání i uživateli, kteří nemají žádné znalosti o využívaných statistických metodách. Toho bylo dosaženo jak automatizací veškerých výpočetních kroků, tak i vytvořením rozsáhlé nápovědy, která zahrnuje popisky pro každé požadavky na uživatele ze strany aplikace. Splnila jsem tedy svůj požadavek na „user-friendly“ prostředí aplikace, se kterou je jednoduché pracovat.

Při zpracování daného tématu se mi podařilo dosáhnout mnou stanovených cílů ve všech bodech, ve kterých jsem chtěla. Základním požadavkem, který jsem si pro sebe stanovila, bylo naučit se základům programovacího jazyka C# a pochopit, jak funguje architektura .NET. Pro výslednou aplikaci jsem vytvořila velmi jednoduchý definiční soubor CSS stylů, základní databázi umožňující zaměření se na konkrétního uživatele a jeho data a celá aplikace zpracovává data uspořádaná v časových řadách, která analyzuje, určuje jejich trendovou složku a stanovuje prognózy pro další období. Uživateli je také umožněno tyto výsledky exportovat do souborů ve formátech *.pdf a *.xls a následně si tyto výsledky uchovat u sebe na počítači.

Aplikace tvoří základní rámec, kterým by se mohla dál ubírat cesta pro vytvoření komplexního souboru manažerských analytických nástrojů pro podporu rozhodování. V bakalářské práci jsou uvedeny další návrhy a možnosti rozšíření, pomocí kterých by mohlo být dosaženo větší komplexnosti a hlavně přesnosti výpočtů a předpovědí a možnosti momentálního využití, které jsou, jak už jsem zmínila, značně omezené hlavně tím, že obsahují malé množství metod výpočtů a částečně také jednoduchým vzhledem, který nevypadá reprezentativně.

LITERATURA

Knižní tituly (řazeno abecedně podle autora):

[1] KROPÁČ, Jiří. *Statistika B*. 2. dopl. vyd. Brno : Fakulta podnikatelská, VUT v Brně, 2009. 151 s. ISBN 978-80-214-3295-6.

[2] MACDONALD, Matthew, SZPUSZTA, Mario. *ASP.NET 2.0 a C#*. Brno : ZONER software, s.r.o. 2006. 1376 s. ISBN 80-86815-38-2.

Elektronické dokumenty (řazeno abecedně podle názvu):

[3] BĚHÁLEK, Marek. *1. Architektura .NET Framework.* [online]. c2007 [cit. 2010-1-29]. Dostupný z WWW:http://www.cs.vsb.cz/behalek/vyuka/pcsharp/text/ch01s01.html.

[4] JANKOVSKÝ, Dušan. *Jakpsatweb.cz* [online]. Rok vydání: Neuvedeno, Poslední změna 2.5.2010 [cit. 2010-05-17]. Javascript - úvod.
Dostupný z WWW:
http://www.jakpsatweb.cz/javascript/javascript-uvod.html.

[5]MICROSOFT. *Přehled architektury .NET* [online]. [cit. 2010-1-29]. Dostupný z WWW:
http://www.google.cz/url?sa=t&source=web&ct=res&cd=2&ved=0CA8QFjAB&url=http%3A%2F%2Fdownload.microsoft.com%2Fdownload%2F8%2F6%2Fc%2F86c09926-affc-4e14-bec0-3c45cd989436%2FPrehled_architektury_NET.pdf&rct=j&q=architektura+.NET&ei=JRhkS8LNIZCanwPz_40a&usg=AFQjCNEiBHwoYVJgOyh3LxCU-l_3fdrhpw.

[6] *Regrese a korelace* [online].[cit. 2009-12-30]. Dostupný z WWW: http://iastat.vse.cz/regrese/Regrese9.htm.

[7] *Obyvatelstvo - Roční časové řady* [online]. c2009 [cit. 2009-11-22]. Dostupný z WWW: http://www.czso.cz/csu/redakce.nsf/c5cfebca9de6e905c125723a004180a6/70205e505233c01cc12570820040b7e7/$FILE/oby01.xls.

[8] JANOVSKÝ, Dušan. *Úvod do CSS* [online]. c2009-12-3 [cit. 2010-1-2]. Dostupný z WWW: http://www.jakpsatweb.cz/css/css-uvod.html.

[9] *Úvod do CSS* [online]. c2004 [cit. 2010-1-2]. Dostupný z WWW http://www.webtvorba.cz/css/uvod-do-css.html#kcemu.

[10] *Začínáme s ASP.NET 2.0* [online]. 26.9.2005. [cit. 2010-1-29] Dostupný z WWW: http://www.zive.cz/clanky/zaciname-s-aspnet-20--1-dil-uvod-instalace/sc-3-a-126787/default.aspx

SEZNAM OBRÁZKŮ

Obrázek 1 : Architektura .NET - zdroj [3] 21
Obrázek 2: Hierarchie objektů v rámci OOP – vlastní tvorba 23
Obrázek 3: Abstrakce zapouzdření - vlastní tvorba 24
Obrázek 4: Ukázka kódu psaného v jazyce C# - zdroj: vlastní tvorba 25
Obrázek 5: Příklad javascriptu z bakalářské práce - zdroj: vlastní tvorba 27
Obrázek 6: Ukázka zobrazení dat stávajícím produktem - zdroj: Aplikace Excel printscrean 30
Obrázek 7: Vývojový diagram Skriptu JSTrend - zdroj: Vlastní tvorba 35
Obrázek 8: Skript pro výběr metody výpočtu trendu uživatelem - zdroj: Vlastní tvorba str. 36 35
Obrázek 9: Skript pro výběr grafu a jeho typu - zdroj: Vlastní tvorbastr. 33 36
Obrázek 10: Vytvoření tabulky pro zadání dat - zdroj: Vlastní tvorba str. 33 37
Obrázek 11: Část diagramu algoritmu vypočítávající koeficienty regresní přímky - zdroj: Vlastní tvorba str. 33 40
Obrázek 12: Část diagramu algoritmu vypočítávající index determinace - zdroj: Vlastní tvorba str. 33 41
Obrázek 13: Modifikovatelný exponenciální trend - zdroj: Vlastní tvorba str. 36 43
Obrázek 14: Diagram zachycující algoritmus počítající charakteristiky časové řady - zdroj: Vlastní tvorba str. 35 44
Obrázek 15: Diagram algoritmu počítající výběrový průměr - zdroj: Vlastní tvorba str. 35 45
Obrázek 16: Diagram generování grafů - zdroj: Vlastní tvorba str. 36 47
Obrázek 17: Úprava reportu v design - módu - zdroj: Print screan z aplikace Visual Web Developer 2008 Express Edition 48
Obrázek 18: Generování názvů období - zdroj: Vytvořená aplikace 54

SEZNAM PŘÍLOH

Příloha č. 1: Uživatelská příručka

Příloha č. 2: CD se zdrojovými soubory

Příloha č. 1: Uživatelská příručka

Produkt slouží jako podpůrný nástroj pro manažerská rozhodování, při kterých je třeba stanovit prognózu pro další období.

Webové stránky jsou v době svázání této bakalářské práce k dispozici na adrese:

http://www.Pear.cz/AnalyzaTrendu/

Po načtení webové aplikace vidíme panel, kde je možno se přihlásit. Přihlášení proběhne po zadání již registrovaného uživatelského jména do kolonky „Zadej jméno:“ a tomu příslušného hesla do kolonky „Zadej heslo“. Protože je aplikace určená pro malé množství uživatelů, není umožněna registrace. Pouze administrátor může přidat nového uživatele. Veškeré funkce jsou ale k dispozici i nepřihlášeným uživatelům. Nepřihlášený uživatel si pouze nemže ukládat své výpočty.

V levém panelu vidí uživatel, který je přihlášen své již vypočtené sestavy. Je zde uvedeno datum vytvoření, název a ikonka pro zobrazení a stažení souboru v MS Excel nebo ve formátu *.pdf.

V hlavním panelu probíhá samotné zadání dat a výpočet v krocích, které jsou zobrazeny v levé straně tohoto panelu. Tyto kroky jsou:

- Nové zadání.
- Počet dat.
- Formát dat.
- Volby výpočtů.
- Předvolby grafů.
- Zadání hodnot.
- Kontrola zadání.
- Výpočty.

Nové zadání

Zde je vidět jednoduchý popis aplikace, k čemu slouží a základní pokyny k používání, jako například nutnost povolených JavaScriptů. Po kliknutí na tlačítko další se dostáváme k dalšímu kroku.

Počet dat

V kroku počet dat se zadává do kolonky „Zde zadejte počet dat“ počet dat, které bude mít aplikace k dispozici. Po kliknutí na tlačítko další se dostaneme k dalšímu kroku.

Formát dat

V tomto kroku jsou zobrazeny dva seznamy, ve kterých se v prvním vybírá z přednastaveného seznamu, jakou má délku jednotlivé období. Je nabízen výběr z roku, pololetí, čtvrtletí, měsíci a jiné. V druhém seznamu se volí, jestli se bude jednat o časovou řadu intervalovou nebo okamžikovou.

Volby výpočtů

Zde je seznam pro výběr výpočtů, obsahující možnosti Charakteristiky časové řady – po zatržení aplikace počítá průměr hodnot, první diference, průměr prvních diferencí, koeficient růstu a průměrný koeficient růstu, dále je možné zatrhnout Trend časové řady, při této volbě je nutno zadat, pro kolik období se bude počítat prognóza. Je nutné zadat alespoň nulovou hodnotu, pokud by se měl počítat pouze trend. Další možností je zvolení metody výpočtu trendu, při zatržení je nutné odsouhlasit upozornění, že volba je pouze pro pokročilé uživatele a pak už stačí jen vybrat jednu z metod. Další dvě volby spočívají pouze v zobrazení rovnice trendu a indexu determinace. Kliknutím na tlačítko další se přechází do dalšího kroku.

Předvolby grafů

V tomto kroku je možné zvolit, které grafy se budou zobrazovat. Po zatržení libovolné z voleb je nutné vybrat typ grafu, zda se bude jednat o sloupcový či spojnicový. Kliknutím na tlačítko další se opět přechází do dalšího kroku.

Zadání hodnot

Nyní je třeba zadat samotné hodnoty, se kterými bude aplikace pracovat. Pokud byl zadán počet dat, který není dělitelný třemi, je zobrazen pokyn pro výběr, z které strany se budou data ořezávat.

Kontrola zadání

Zde by si měl uživatel zkontrolovat veškeré svoje předvolby, a pokud v nich nenajde žádné nesrovnalosti, odeslat data k výpočtu stisknutím tlačítka další.

Výpočty

V tomto kroku jsou zobrazeny výsledky ve formuláři, který umožňuje export dat. Po výběru formátu z rozvinovacího seznamu je možné vybrat uložení do formátu MS Excel nebo Acrobat (pdf). Po výběru formátu je nutné kliknout na tlačítko Export, poté vyskočí tabulka operačního systému pro uložení nebo otevření souboru. Tlačítkem uložit může přihlášený uživatel uložit svoji sestavu do databáze. Tlačítkem Nové zadání je možné se přesunout zpátky do prvního uvítacího kroku.

Příloha č. 2: CD se zdrojovými soubory