VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA PODNIKATELSKÁ
ÚSTAV INFORMATIKY

FACULTY OF BUSINESS AND MANAGEMENT
INSTITUT OF INFORMATICS

UPLATNĚNÍ STATISTICKÝCH METOD PŘI ZPRACOVÁNÍ DAT

APPLICATION OF STATISTICAL METHODS FOR PROCESSING DATA

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. KAREL BALGA
AUTHOR

VEDOUCÍ PRÁCE — Mgr. VERONIKA NOVOTNÁ, Ph.D.
SUPERVISOR

BRNO 2014

Vysoké učení technické v Brně
Fakulta podnikatelská

Akademický rok 2013/2014
Ústav informatiky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Balga Karel, Bc.

Informační management (6209T015)

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 diplomovou práci s názvem:

UPLATNĚNÍ STATISTICKÝCH METOD PŘI ZPRACOVÁNÍ DAT

v anglickém jazyce:

APPLICATION OF STATISTICAL METHODS FOR PROCESSING DATA

Pokyny pro vypracování:

Úvod
Cíle práce, metody a postupy zpracování
Teoretická východiska práce
Analýza problému
Vlastní návrhy řešení
Závěr
Seznam použité literatury

Seznam odborné literatury:

HINDLS, R. Statistika pro ekonomy. 8. vyd. Praha: Professional Publishing, 2007, 415 s. ISBN 978-80-86946-43-6.
KROPÁČ, J. Statistika B. 2. dopl. vyd. Brno: Vysoké učení technické v Brně, Fakulta podnikatelská, 2009, 151 s. ISBN 978-80-214-3295-6.
KUBANOVÁ, J. Statistické metody pro ekonomickou a technickou praxi. 3. vyd. Bratislava: STATIS, 2008. 247 s. ISBN 978-80-85659-474.
RŮČKOVÁ, P. Finanční analýza: metody, ukazatele, využití v praxi. 3. rozš. vyd. Praha: Grada, 2010. 139 s. ISBN 978-80-247-3308-1.
SEDLÁČEK, J. Finanční analýza podniku. 1. vydání. Brno: Computer Press, 2007. 154 s. ISBN 978-80-251-1830-6.

Vedoucí diplomové práce: Mgr. Veronika Novotná, Ph.D.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2013/2014.

L.S.

doc. RNDr. Bedřich Půža, CSc. Ředitel ústavu	doc. Ing. et Ing. Stanislav Škapa, Ph.D. Děkan fakulty

V Brně, dne 25. 05. 2014

Abstrakt

Obsahem této diplomové práce je aplikace vybraných statistických metod, které budou využívány pro řízení zásob vybrané společnosti. Praktická část se zabývá analýzou a popisem aplikace, která bude vyhodnocovat jakým způsobem je vedena zásobovací politika společnosti.

Abstract

The content of this thesis is the application of selected statistical methods that will be used for inventory management selected companies. The practical part deals with the analysis and description of the application that will evaluate how the policy is guided by the supply company.

Klíčová slova

Regresní přímka, řízení zásob, bod objednávky, .NET, C#

Keywords

Regression line, stock management, order item, .NET, C#

BIBLIOGRAFICKÁ CITACE MÉ PRÁCE

BALGA, K. *Uplatnění statistických metod při zpracování dat.* Brno: Vysoké učení technické v Brně, Fakulta podnikatelská, 2014. 73 s. Vedoucí diplomové práce Mgr. Veronika Novotná, Ph.D.

ČESTNÉ PROHLÁŠENÍ

Prohlašuji, že předložená diplomová práce je původní a zpracoval jsem ji samostatně. Prohlašuji, že citace použitých pramenů je úplná, že jsem v práci neporušil autorská práva (ve smyslu Zákona č. 121/2000 Sb. o právu autorském a o právech souvisejících s právem autorským).

V Brně, dne 20. května 2014 ……...

Podpis studenta

PODĚKOVÁNÍ

Děkuji vedoucímu diplomové práce Mgr. Veronice Novotné, Ph.D., za cenné rady, připomínky a metodické vedení této diplomové práce

Obsah

Úvod

Diplomová práce pojednává o praktickém využití vybraných statistických metod pro vybranou společnost. Protože majitel společnosti si nepřál, abych ve své práci název společnosti zveřejňoval, použil imaginární název „ společnost XY“.

Zvolené statistické metody budou demonstrovány na vybraných vzorcích zásobovacích položek.

Výstupem této diplomové práce bude i nástroj, který bude schopný tyto úkony simulovat a ukazovat konkrétní výsledky, které se pak stanou rozhodujícím faktorem rozhodovávání podniku mnou zkoumané oblasti řízení zásob.

Cíle práce metody a postupy zpracování

Diplomová práce se bude zabývat praktickou aplikací vybraných statistických metod na zvolených položkách zásob podniku „XY“ tak, aby bylo umožněno managementu společnosti efektivně rozhodovat při jejich řízení.

Dílčím cílem této práce bude provést analýzu prostřednictvím vybraných statistických metod (regresní analýza) a na základě této analýzy pak optimalizovat řízení vybraných položek zásob podniku.

Hlavním cílem diplomové práce je pak z výstupu statistické analýzy navrhnout aplikaci, která bude vhodnou náhradou za plně automatizovaný software, který je dodáván specializovanými společnostmi, které se tímto vývojem zabývají. Aplikace bude fungovat tak, že nejdříve vyhodnotí vkládaná data a poté bude dávat podniku zpětnou vazbu, zda jsou zásoby řízeny optimálně. Samotná aplikace by poté měla poskytovat uživateli pohled na zásobovací politiku podniku, což by mělo šetřit nejen náklady za skladování, ale také omezit dobu vázanosti oběžných aktiv. Při implementaci aplikace bude také provedeno zaškolení uživatelů. Předpokládanou hodnotou této aplikace bude její jednoduchost a ušetřené náklady za její vytvoření.

Při zpracování analýzy zásob vycházím z modelu nezávislé poptávky. Součástí zpracování analýzy zásob podle tohoto modelu je regresní analýza. Prostřednictvím metod využívaných v modelu nezávislé poptávky pak budu následně schopný stanovit výši pojistné zásoby a také určit bod znovu objednávky.

1 Teoretická východiska práce

1.1 Časové řady

„Časovou řadou (někdy také „chronologickou řadou“) rozumíme řadu hodnot určitého ukazatele, uspořádaných z hlediska přirozené časové posloupnosti. Přitom je nutné, aby věcná náplň ukazatele i jeho prostorové vymezení byly shodné v celém sledovaném úseku.“ (1, str. 114)

Prostřednictvím posloupnosti dat, které bývají srovnávány časovými milníky, počínaje minulosti do nynějšího stavu můžeme namodelovat současný trend nebo predikovat budoucí vývoj. Pokud se provádí analýza vybraných dat, nesmí být opomenuto, že časové řady mají určité rozdělení a to:

- Intervalové - tyto řady odhalují určitý jev za určité období, které může být i v rámci kalendářního roka. (1)
- Okamžikové – tyto řady jsou využívány pro indikaci stavu v momentálním čase. (1)

Odlišností těchto mezi těmito typy ukazatelů časové řady spočívají v tom, že v případě intervalové časové řady lze v souvislosti se statistikou sčítat, kdežto sčítání u okamžikových časových řad nedává v podstatě žádný smysl. V případě, že se rozhodneme pracovat s intervalovou časovou řadou, je nutné detailně zkoumat rozdílnost jednotlivých délek časových řad. (1)

Pokud jsou délky intervalů rozdílné, pak tato skutečnost vede mystifikaci hodnot ukazatelů a potenciálního vývoje. Je tedy potřeba zkoumat délku doby a na základě toho srovnávat data. Tento proces lze zpracovávat následujícími způsoby.

1) Přepočet počátečních dat na hodnoty, který mají shodný časový interval. Tato situace může vzniknout například v případě u jednotlivých délek měsíců a to konkrétně tak, že některé měsíce mají 30 nebo 31 dní nebo variantní možnosti měsíce února, z těchto údajů pak musíme nastavit průměrnou měsíční výrobu dané společnosti. V popisovaném případě měsíční produkci následně podělíme počtem dní v měsíci a její výsledek je vynásoben 30. (1)

2) Získání průměrné délky měsíce můžeme získat i vydělením délky roku (365 resp. 366 dny) počtem měsíců tedy číslem 12. (1)

Pokud využíváme okamžikové časové řady, tak tento problém nemůže vzniknout, jelikož se hodnoty vztahují k určitému okamžiku. Tuto možnou skutečnost jsem již popisoval výše. (1)

Pro vhodnější interpretaci vybraných dat a výsledků analýzy je při nejmenším vhodné zobrazit data pomocí namodelovaných grafů, z kterých můžeme vyvodit současný vývoj a predikovat vývoj budoucí. (1)

Intervalové časové řady můžeme znázornit prostřednictvím těchto grafů:

- Sloupkovým grafem,
- Hůlkovým grafem,
- Spojnicovým grafem. (1)

Okamžikové časové řady lze vzhledem k jejich povaze zobrazit:

- Spojnicovým grafem. (1)

1.2 Charakteristiky časových řad

„Prvním úkolem při zpracovávání a analýze časových řad bývá získání rychlé a orientační představy o charakteru procesu, který tato řada reprezentuje. Mezi základní metody proto zcela běžně patří vizuální analýza chování ukazatele využívající grafů spolu s určováním elementárních statistických charakteristik.“ (3., str. 252)

„Uvažujeme časovou řadu okamžikového, resp. intervalového ukazatele, jejíž hodnoty v časových okamžicích resp. intervalech ti, kde i = 1, 2,…., n, označíme y_i. Budeme předpokládat, že tyto hodnoty jsou kladné. Při výpočtu charakteristik časových řad dále předpokládáme, že intervaly mezi sousedními časovými okamžiky, resp. středy časových intervalů jsou stejně dlouhé.“ (1, str. 117)

Do základních charakteristik časových řad zahrnujeme jejich průměry. Pokud hovoříme o intervalových časových řadách, pak se jedná o aritmetických průměr, který je dán tímto vztahem:

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \qquad (1.1)$$

Je potřeba vzít v úvahu také, že pokud hovoříme o okamžikových řadách, pak se jedná o tzv. chronologický průměr. V případě, že je zřejmá posloupnost mezi konkrétními délkami jednotlivých časových úseků nazýváme tento chronologický průměr neváženým chronologickým průměrem, jehož vztah je pak následující:

$$\bar{y} = \frac{1}{n-1}\left[\frac{y_1}{2} + \sum_{i=2}^{n-1} y_i + \frac{y_n}{2}\right] \qquad (1.2)$$

K provedení analýzy časových následně můžeme využít další charakteristiky popisu vývoje časových řad. K těm nejintenzivněji používaným charakteristikám patří tzv. *první diference*, které demonstrují přírůstek oproti předchozí hodnotě. Vztah pro diferenci získáme z následujícího vzorce:

$$_1d_i(y) = y_i - y_{i-1},\ i = 1,2,3,\ldots, \qquad (1.3)$$

Výslednou hodnotou je rozdíl dvou po sobě následujících hodnot časové řady. První diference je tedy rozdíl (přírůstek nebo úbytek) hodnoty časové řady k určitému časovému momentu (tedy lépe řečeno k předcházejícímu období). (1)

K vhodnějšímu znázornění se nabízí využití možnosti z prvních diferencí vypočíst průměrný přírůstek za sledované období, tzv. průměr prvních diferencí, který je dán následujícím vztahem:

$$_1\overline{d(y)} = \frac{y_n - y_1}{n-1} \qquad (1.4)$$

Dynamika, při níž hodnoty časové řady oscilují, jak k vyšším, tak k nižším hodnotám lze popsat koeficientem růstu získaným ze vzorce

$$k_i(y) = \frac{y_i}{y_{i-1}},\ i = 1,2,3,..., \qquad (1.5)$$

Koeficient růstu znázorňuje, kolikrát se zvýšila hodnota časové řady v určitém

okamžiku oproti bezprostředně předcházejícímu období. (1)

Důležitou charakteristikou při zkoumání časových řad je také tzv. *průměrný* koeficient růstu, který vypovídá o průměrné rychlosti růstu či poklesu hodnot. Tento průměrný koeficient růstu počítáme jako geometrický průměr dle následujícího vzorce:

$$\overline{\mathrm{k(y)}} = \sqrt[\mathrm{n-1}]{\frac{\mathrm{y_n}}{\mathrm{y_1}}} \tag{1.6}$$

Průměrná hodnota prvních diferencí a průměrná hodnota koeficientu růstu závisí explicitně jenom na první a poslední hodnotě ukazatele časové řady. Další vnitřní hodnoty ukazatele na tyto průměry nemají žádný důležitý vliv.(1)

1.2.1 Dekompozice časových řad

Model časové řady můžeme sestavit z několika složek, jejichž vlastnosti budou v této kapitole detailněji rozebrány.

„Hodnoty časové řady mohou být rozloženy na několik složek. Jestliže jde o tzv. aditivní dekompozici, lze hodnoty y_i časové řady vyjádřit pro čas ti, i =1,2,…, n, součtem:

$$y_\mathrm{i} = T_\mathrm{i} + C_i + S_i + e_i \tag{1.7}$$

kde jednotlivé sčítance vyjadřují:

- T_i – hodnotu trendové složky,
- $\mathrm{C_i}$ – hodnotu cyklické složky,
- $\mathrm{S_i}$ – hodnotu sezonní složky,
- $\mathrm{e_i}$ – hodnotu náhodné složky.“ (1, str. 122)

Pokud provedeme rozklad časové složky, naskytne se nám možnost identifikovat určité vazby mezi jednotlivými složkami, případně můžeme přijít na to, že některá část chybí úplně. Konkrétní náhled a popis na jednotlivé složky:

- Trendová složka – demonstruje tendenci sledovaného ukazatele v závislosti na časové veličině během dlouhodobého vývoje. V případě, že

trendový ukazatel během zkoumání časové řady kolísá k určité hodnotě nebo se snižuje, pak můžeme o takové charakteristice prohlásit, že se jedná o časovou řadu bez trendové složky.

- Cyklická složka – místo nebo oblast, kde můžeme pozorovat střídání růstu s poklesem. Délka pozorovaných časových period je variabilní. Prvky, jejichž působením dojde ke změnám v cyklické složce, je v některých případech složité rozpoznat; je potřeba podotknout, že tyto změny jsou zapříčiňovány vnějšími vlivy.
- Sezónní složka – ukazuje na změny v časové řadě; tyto změny jsou periodické a mohou se opakovat. Může se jednat např. o pokles nezaměstnanosti v letních měsících, což se stává pravidelně každým rokem
- Reziduální složka – tuto složku tvoří zejména nepravidelné výkyvy v průběhu sledování časové řady. Do reziduální složky jsou zahrnuty chyby, kterých se můžeme dopustit během pracování s daty (např. během chybného zaokrouhlování). (1)

1.3 Regresní analýza

„V ekonomice a přírodních vědách se často pracuje s proměnnými veličinami, kdy mezi nezávisle proměnnou x a závisle proměnnou y, kterou měříme či pozorujeme, existuje nějaká závislost. Ta je buď vyjádřena funkčním předpisem y = φ(x), kde ale funkci φ(x)neznáme nebo tuto závislost nelze „rozumnou“ funkcí vyjádřit. Při nastavené hodnotě x dostaneme jednu hodnotu závisle proměnné y.“ (1., str. 78)

Jako příklady z praxe lze uvést například zkoumání (měření):

- Jak velikost výdajů za potraviny závisí na počtu členů domácnosti,
- Jak velikost tržeb závisí na počtu obyvatelstva v místě, kde se prodejna nachází,
- Jak objem spotřebovaných pohonných hmot závisí na rychlosti jízdy automobilu. (1)

Pokud zaznamenáme vliv působení různých náhodných vlivů a činitelů získáme v případě, že měření provedeme s opakováním nastavenou hodnotou proměnné x rozdílné hodnoty závisle proměnné y. Z toho důvodu tedy můžeme konstatovat, že proměnná y se chová jako náhodná veličina označená Y. (1)

Na vazby mezi veličinami x a y působí šum. Šum je náhodná veličinu e, kterou lze obsahově vymezit jako vliv náhodných a neuvažovaných činitelů a budeme zároveň předpokládat, že její střední hodnota se rovná nule. (1)

Pro vyjádření závislosti náhodné veličiny Y na proměnné x využijeme podmíněnou střední hodnotu náhodné veličiny Y pro hodnotu x, označenou E (Y|x) a položíme ji rovnu zvolené funkci ŋ(x; β_1, β_2,.. β_p). Vzájemný vztah vyjádříme vzorcem:

$$E(Y|x) = \eta(x; \beta_1, \beta_2, .. \beta_p). \qquad (1.8)$$

Funkce ŋ(x; β_1, β_2,.. β_p) je funkcí nezávisle proměnné x a obsahuje neznámé parametry β_1, β_2,.. ßp, kde p≥1. Funkci ŋ(x) nazveme regresní funkcí a parametry β_1, β_2,.. β_p nazveme regresními koeficienty. Pokud funkci ŋ(x) pro zadaná data nalezneme, konstatujeme, že jsme zadaná data „vyrovnali regresní funkcí“. Prvořadou úlohou regresní analýzy je tak pro zadaná data zvolit vhodnou funkci ŋ(x) a následně odhadnout její koeficienty tak, aby se dosáhlo „co nejlepšího“ vyrovnání hodnot y_i. (1)

1.3.1 Regresní přímka

V běžném případě problému, kde je využíváno regresní úlohy je regresní funkce ŋ(x) vyjádřena přímkou ŋ(x) = $\beta_1 + \beta_2 x$. Tento vztah vyjádříme následovně:

$$E\,(Y|x) = \eta(x) = \beta_1 + \beta_2 x. \tag{1.9}$$

Primární úlohou je charakterizovat koeficienty regresní přímky (β_1 a β_2) pro konkrétní dvojice (x_i,y_i). Tyto odhady koeficientů označíme b_1 a b_2. Predikované hodnoty bychom měli navolit co nejoptimálněji. K docílení tohoto stavu budeme používat tzv. metodu nejmenších čtverců, která za „nejlepší“ odhady koeficientů považuje takové, které minimalizují funkci $S(b_1,b_2)$. Funkce $S(b_1,b_2)$ prezentuje součet kvadrátů odchylek naměřených hodnot y_i od hodnot $\eta_i = \eta\,(x_i) = b_1 + b_2 x_i$ na regresní přímce. (1)

Pro nalezení odhadů b_1 a b_2 koeficientů $ß_1$ a $ß_2$ budeme vycházet ze soustavy tzv. normálových rovnic:

$$n.\,b_1 + \sum_{i=1}^{n} x_i.\,b_2 = \sum_{i=1}^{n} y_i. \tag{2.0}$$

$$\sum_{i=1}^{n} x_i.\,b_1 + \sum_{i=1}^{n} x_i^2\,.\,b_2 = \sum_{i=1}^{n} x_i\,.\,y_i. \tag{2.1}$$

Hodnoty koeficientů b1 a b2 můžeme získat prostřednictvím metody pro výpočet soustavy dvou lineárních rovnic o dvou neznámých nebo lze k výpočtu využít následujících vztahů:

$$b_2 = \frac{\sum_{i=1}^{n} x_i\,y_i - n\bar{x}\bar{y}}{\sum_{i=1}^{n} x_i^2 - n\bar{x}^2};\ \bar{y} - b_2. \tag{2.2}$$

Výběrové průměry $\bar{x}$ respektive $\bar{y}$ určíme využitím vzorců:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i\ ;\ \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i. \tag{2.3}$$

K odhadu regresní přímky můžeme dojít následujícím vztahem:

$$\hat{\eta} = b_1 + b_2 x. \tag{2.4}$$

Koeficient b_2 popisuje odhad přírůstku střední hodnoty závisle proměnné při jednotkovém nárůstu nezávisle proměnné. (1)

1.3.2 Koeficienty regresní přímky a jejich vlastnosti

Hodnoty koeficientů regresní přímky (b_1, b_2) přiřadíme k získaným hodnotám y_i závisle proměnné. V případě, že bychom měření opakovali vícekrát, získané hodnoty y_i by se lišily. Také bychom došli k jiným výsledkům hodnot koeficientů b_1, b_2 a k jiné regresní přímce. Proto jsou vypočítané koeficienty regresní přímky i samotná regresní přímka náhodnými veličinami. Označíme je B_1, B_2 a $\hat{\eta}(x)$ a nazveme statistikami. Díky teorii regresních funkcí je pak možné dostat ze zadaných dat z těchto statistik více podkladů. Předpoklady o vlastnostech náhodných veličin e_i představující „šumy" vyjádříme následovně:

$$E(e_i) = 0,\ D(e_i) = \sigma^2,$$
$$C(e_i, e_j) = 0 \text{ pro } i \neq j, \text{ kde } i, j = 1,2,\ldots, n.$$

Výše stanovené předpoklady indikují, že náhodné veličiny e_i mají nulové střední hodnoty a týž rozptyl σ^2, což vypovídá o tom, že měření závislé proměnné není narušeno systematickými chybami a rozptyly chyb měření jsou nezávislé na jednotlivých hodnotách nezávisle proměnné. Další fakt, ze kterého budeme vycházet, vypovídá o kovarianci náhodných veličin e_i a e_j, kde $i \neq j$, jež je rovna nule. Z tohoto faktu vyplývá, že náhodné veličiny jsou tedy nekorelované, což znamená, že mezi nimi není lineární korelační vazba. (1)

Jsou-li výše uvedené předpoklady splněny, pak pro náhodné veličiny Y_i platí:

$$E(Y_i) = \beta_1 + \beta_2 x_i,\ D(Y_i) = \sigma^2,$$
$$C(Y_i, Y_j) = 0 \text{ pro } i \neq j, \text{ kde } i, j = 1, 2,\ldots, n. \qquad (2.5)$$

Z toho vyplývá, že můžeme přiřadit střední hodnoty náhodných veličin Y_i k hodnotám regresní přímky, jejich rozptyl je týž jako rozptyl náhodných veličin e_i a náhodné veličiny Y_i a Y_j, kde $i \neq j$, jsou nekorelované. Statistiky B_1, B_2 a $\hat{\eta}(x)$ jsou

nestrannými bodovými odhady koeficientů β_1, β_2 a regresní přímky $\hat{\eta}(x)$. Při výpočtu zmiňovaných koeficientů (resp. regresní přímky) pro více sérií měření veličiny y, pak průměry získaných regresních koeficientů (resp. získaných regresních přímek) přiřadíme k regresním koeficientům β_1 a β_2 (resp. regresní přímce $\hat{\eta}(x) = \beta_1 + \beta_2 x$). (1)

Pokud jsou splněny dříve stanovené podmínky, jsou rozptyly statistik B_1 a B_2 určeny vzorci

$$D(B_1) = \left[\frac{1}{n} + \frac{x^{-2}}{\sum_{i=1}^{n} x_i^2 - nx^{-2}}\right]\sigma^2, D(B_2) = \frac{\sigma^2}{\sum_{i=1}^{n} x_i^2 - nx^{-2}}. \quad (2.6)$$

a rozptyl statistiky $\hat{\eta}(x)$ je dán vzorcem:

$$D(\hat{\eta}(x)) = \left[\frac{1}{n} + \frac{(x-\bar{x})^2}{\sum_{i=1}^{n} x_i^2 - nx^{-2}}\right]\sigma^2. \quad (2.7)$$

Ve vzorcích pro výpočet rozptylů charakteristik B_1, B_2 a $\hat{\eta}(x)$ můžeme také zpozorovat hodnotu rozptylu σ^2, jenž charakterizuje přesnost měření. Pokud se s touto hodnotou nesetkáme, nikdo ji nestanovil, je nutné ji odhadnout. K tomuto odhadu se používá tzv. reziduální součet čtverců daný vzorcem:

$$S_R = \sum_{i=1}^{n} \hat{e}_i^{\,2} = \sum_{i=1}^{n} (y_i - \hat{\eta}(x_i))^{\,2}. \quad (2.8)$$

Reziduální součet čtverců vypovídá o stupni rozptýlení pozorovaných hodnot závislé proměnné kolem určené regresní přímky. (1)

Hodnota rozptylu σ^2 je dále určena vztahem:

$$\hat{\sigma} = \frac{S_R}{n-2}, \quad (2.9)$$

kde n udává počet naměřených dvojic hodnot x_i, y_i.

Pokud rozptyl není určen, dosadí se jeho odhad do vzorců (2.6) a (2.7). Tím získáme odhady rozptylů $\hat{D}(B_1), \hat{D}(B_2)$ a $\hat{D}(\hat{\eta}(x))$. (1)

1.3.3 Intervaly spolehlivosti pro koeficienty regresní přímky

Pokud k předpokladům zmíněným v předchozí podkapitole přidáme předpoklad tvrdící, že rozdělení náhodných veličin e_i je normální, pak statistiky

$$T_{\hat{n}} = \frac{\hat{\eta}(x) - \hat{\eta}(x)}{\sqrt{\widehat{D}(\hat{\eta}(x))}} \; a \; T_{B_1} = \frac{B_l - \beta_l}{\sqrt{\widehat{D}\,(B_l)}}, \text{ kde } l = 1,2, \qquad (3.0)$$

mají Studentovo rozdělení o n – 2 stupních volnosti.

Prostřednictvím těchto statistik lze mimo jiné zkonstruovat intervaly spolehlivosti pro koeficienty regresní přímky β_1, β_2. (1)

Pro koeficienty β_l, l = 1, 2, jsou 100(1-α) %-ní intervaly spolehlivosti dány předpisem:

$$\left(b_l - \; t_{1-\frac{\alpha}{2}}(n-2)\sqrt{\widehat{D}(B_1)}; b_1 + t_{1-\frac{\alpha}{2}}(n-2)\sqrt{\widehat{D}(B_l)}\right). \qquad (3.1)$$

Výraz $t_{1-\frac{\alpha}{2}}\,(n-2)$ představuje kvantil Studentova rozdělení, odhady rozptylů $\widehat{D}(B_l)$ koeficientů B_l se vypočítají pomocí vzorců (2.6). (1)

Výpočtem tedy získáme dolní a horní hranice intervalů, které pro zvolenou hladinu významnosti pokrývají parametry β_1, a β_2, dané regresní přímky. (1)

1.4 Řízení zásob

Tato kapitola bude pojednávat o řízení zásob a vzorce pro tuto problematiku jsem čerpal z (). Výběr modelu zásob je závislý na charakteristice dané poptávky tj., jaké má chování poptávka v daném čase t. Dle této konfigurace dělíme poptávku na závislou a nezávislou. (2)

1.4.1 Závislá poptávka

Poptávka, která indikuje potřebu doplnění určitého doplňkového množství výrobku (např. výrobku, který bude ještě součástí dalšího výrobního procesu) může být naproti tomu predikována z poptávky po konečném výrobku. (2)

Po vytvoření primárního výrobního plánu, který určuje velikost dávek a čas pro doplňování zásoby výrobků, které jsou dále určený ke konečnému prodeji, lze vypočítat čas a velikost potřeby všech doplňkových komponent a materiálů, které je třeba vyrobit či nakoupit pro výrobu konečného výrobku. (2)

Závislá poptávka (potřeba) se může vyskytnout pouze u dílů do výrobků zhotovovaných na sklad nebo montovaných na zakázku. Charakter závislé poptávky má také spotřeba materiálů a dílů pro plánované opravy v podniku, tato poptávka se odvíjí od toho, jaké budou představy o těchto opravách tzn. bude vycházet z nějakého plánu vyžadovaných oprav. (2)

1.4.2 Nezávislá poptávka

Nezávislá poptávka funguje na náhodné bázi – společnost neovlivňuje dobu uplatnění požadavků, ani na jejich požadované množství. Tato poptávka je použita v situacích, kdy zákazník požaduje konečný výrobek nebo když je kupříkladu potřeba materiál na náhradní díly, či samotné náhradní díly např. v situacích, kdy se vyskytne nějaká neplánovaná událost. (2)

Nezávislá poptávka po určité položce nemá přímou funkční závislost k potřebě jiných položek, musí být předpovídána, nelze ji vypočítat. Řízení zásob pro uspokojování nezávislé poptávky pracuje s pravděpodobnostními objednacími systémy, v nichž se regulací nejistoty odhadu budoucí poptávky vytváří pojistná zásoba. (2)

1.4.3 Závislá poptávka

Závislá poptávka po produktu (v mezi výrobním procesu) je odvozena z jeho předpovědi. Po sestavení primárního výrobního plánu, jehož obsahem je velikost dávek a doba, sloužící k doplnění zásoby konečných výrobků, lze vypočítat čas a velikost potřeby všech konkrétních dílů a materiálů, které je třeba vyrobit či nakoupit pro výrobu a montáž konečného výrobku. Závislá poptávka se může vyskytnout pouze u komponent do výrobků zhotovovaných na sklad nebo montovaných na zakázku. Charakter závislé poptávky má také potřeba materiálů a dílů pro plánované opravy v podniku – lze je totiž předem stanovit na základě přijatého plánu oprav. (2)

1.4.4 Modely řízení zásob položek při nezávislé poptávce

Tento model zadává signál, který upozorňuje, že je potřeba zadat objednávku k doplnění v případě poklesu dispoziční zásoby pod určitou výši, tzv. objednací úroveň. (2)

Objednací systémy realizují řízení materiálového na základě toku zásoby. Tyto systémy neumožňují předem zjistit ani budoucí okamžiky objednávání, ani budoucí okamžiky dodávek do skladu. Délky intervalů mezi jednotlivými dodávkami jsou proměnlivé a jsou závislé na množstevních i časových výkyvech skutečné poptávky od poptávky, která závisí na předpovědi. (2)

Objednací úroveň zásoby se dimenzuje tak, aby s požadovanou spolehlivostí pokryla skutečnou poptávku během očekávané délky intervalu od vydání pokynu k objednávce až po příjemku příslušné položky do skladovacích prostor. Tuto dobu nazýváme pořizovací lhůtou a označujeme ji t_p. (2)

1.4.5 Varianty objednacích systémů

Prostřednictvím objednacích systémů dostaneme odpověď na otázku, kdy a kolik objednat pro doplnění zásoby. Pokyn pro vydání signálu o objednávce a velikosti je možné zadávat ve dvou řešeních. (2)

Signál zalarmuje řídící složky v případě poklesu dispoziční zásoby objednací úrovně, kterou obvykle značíme B_0. Dispoziční zásoba se porovnává s objednací úrovní průběžně, tj. při každém výskytu požadavku na výdej, resp. při každém výdeji položky.

Signální sestavy je možné zaznamenávat prakticky denně. (2)

Dispoziční zásoba se porovnává s objednací úrovní, označovanou zde B_k, pouze periodicky v intervalech o pevné délce označené t_k, například týdně nebo měsíčně. Signální sestavy vznikají jen periodicky, jsou zpravidla obsáhlejší. Varianty objednacího množství

- Objednávka fixně určeného množství Q,
- Objednávka variabilního množství rovné rozdílu mezi předem určenou cílovou úrovní označovanou S a velikostí dispoziční zásoby v okamžiku vydání signálu. (2)

1.4.6 Analýza čerpání zásoby položky

Vzorce, které se týkají této kapitoly, byly čerpány ze zdroje () uvedeného v seznamu použité literatury. Stav zásob položky v jednotlivých okamžicích zjišťování její velikosti vyjádříme jako hodnoty časové řady z_1, z_2…, z_n, přičemž z_i označuje velikost zásoby položky v okamžiku i = 1, 2,…, n a číslo n značí počet okamžiků, v nichž byl sledován stav zásoby. (2)

Vhodnou metodou pro odhad průběhu čerpání zásoby během jednoho cyklu je regresní analýza. Při využití této metody proložíme zjištěná data v daném cyklu regresní přímkou o rov

$$z = b_1 + b_2 t \tag{3.2}$$

Proměnná *z* vyjadřuje, jak je velká položka zásoby v čase *t*, b_1 a b_2 představují parametry regresní přímky. (2)

Parametry regresní přímky jsou vyjádřeny následujícími vzorci:

$$b_2 = \frac{(\sum_{t=1}^{n} t_i z_i - n\bar{z}\bar{t})}{(\sum_{t=1}^{n} t_i^2 - nt^{-2})},\ b_1 = \bar{z} - b_2\bar{t}; \tag{3.3}$$

$\bar{t}\ a\ \bar{z}$ jsou výběrové průměry t a z, jež jsou dány vztahem

$$\bar{t} = \frac{1}{n}\sum_{i=1}^{n} t_i\ ,\bar{z} = \frac{1}{n}\sum_{i=1}^{n} z_i. \tag{3.4}$$

Získanou regresní přímku je možné dále využít. Hodnota parametru $\boldsymbol{b_2}$ regresní přímky udává průměrnou hodnotu velikosti čerpání položky během jednotkového časového intervalu. Z rovnice regresní přímky lze také určit okamžik, kdy dojde k vyčerpání zásoby položky. (2)

1.4.7 Analýza poptávky

Hodnoty poptávky neboli spotřeby položky během jejího čerpání, tvoří časovou řadu:

$$y_i, y_2, \ldots y_j, \ldots y_n,$$

kde hodnoty y_j vyjadřují velikost poptávky, během období (t_j), přičemž n označuje počet období, během nichž se poptávka odhaduje, ovšem za předpokladu, že tato období jsou fixně určené intervaly. (2)

V případě, že hodnoty nevykazují žádný trend, je možné o časové řadě prohlásit, že je časově ustálenou. Pak lze z hodnot y_j vyjádřit průměrnou velikost poptávky $\bar{y}_p$ a rozptyl velikosti poptávky S_p^2 s pomocí vzorců

$$\bar{y} = \frac{1}{n}\sum_{j=1}^{n} y_j,\ S_p^2 = \frac{1}{n-1} = \left[\sum_{j=1}^{n} y_j^2 - n * \overline{y_p^2}\right]. \qquad (3.5)$$

Pokud vezmeme do úvahy průměrnou dodací lhůtu $\bar{t_d}$, pak z výše vypočtených parametrů bod objednávky vyjde následovně

$$B_o = \overline{y_p} * \bar{t_d} \qquad (3.6)$$

1.5 Nástroje k realizaci aplikace

1.5.1 Platforma .NET

.NET je pojmenování strategie firmy Microsoft pro softwarové řešení, které je nezávislý na operačním systému a hardwaru. Pokud se bavíme o hardware, projekty .NET nejsou omezeny tradičními stolními počítači, ale je možné jejich prostřednictvím vyvíjet také aplikace pro kapesní počítače a telefony. Visual Studio je nástroj určený pro vytváření aplikací .NET. (6)

1.5.2 Architektura .NET

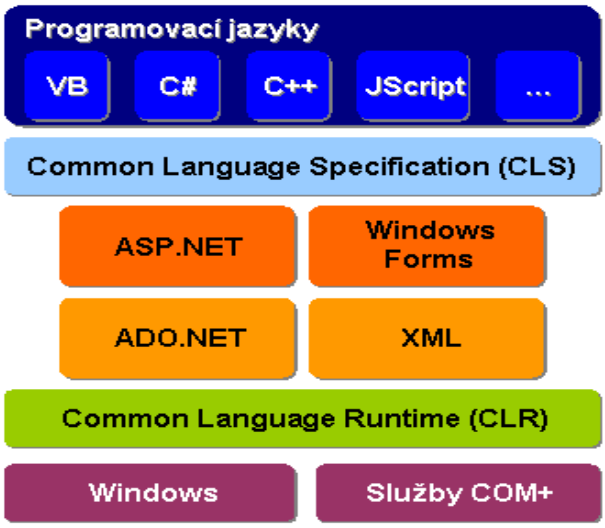
Úloha .NET framework je vnímána jako nadstavba nad operačním systémem. Tato nástavba v sobě zahrnuje CLR (Common Language Runtime), která má za úkol zajistit základní funkcionalitu celého .NET frameworku. Na tomto Common Language Runtimu jsou postaveny všechny další základní knihovny a objekty, jejichž účelem je například přístup k datům (ADO.NET, XML, SQL), k vícevláknovému zpracování aplikací (Threading) k přístupu do sítě a internetu (NET) a k zpracovávání, auditování a monitorování bezpečnosti aplikací (Security). Nad těmito knihovnami stojí už pouze:

a) Webové služby

- Samotné webové služby,
- Webové formuláře,
- ASP.NET. (9).

b) Uživatelské rozhraní

- Controls (tlačítka, seznamy, posuvníky),
- GDI+ (sloužící pro lepší manipulaci s grafikou, .NET framework totiž podporuje nativně souborové formáty JPG, GIF atd., které si již můžeme sami programově vytvářet),
- Služby pro tvorbu klasických desktopových aplikací. (9)



Obrázek č.1: Architektura .NET
Zdroj: Dle (8)

1.5.3 Vývojové prostředí

Visual studio je komplexně integrované vývojové řešení vhodné k procesu psaní, ladění a překladu kódu do výsledných sestavení (assemblies), která lze pak rozšiřovat na další počítače. V praxi to znamená, že Visual Studio poskytuje rozhraní MDI (rozhraní pro více dokumentů), v němž můžete dělat prakticky vše, co nějak s vývojem kódu souvisí. (6)

V prostředí je zahrnut textový editor, v němž můžete psát kód v jazyce C# (stejně jako kód v jazycích VB.NET či C++). Tento textový editor se během psaní automaticky stará o rozvržení kódu odsazením řádků, párováním složených závorek a barevným zvýrazňováním klíčových slov. Kromě toho kontroluje syntaxi již během psaní a okamžitě zvýrazňuje kód, který způsobí syntaktickou chybu nebo chybu při překladu. (6)

Také je využívána technologie IntelliSense, která automaticky zobrazí názvy tříd, datových složek nebo metod po zadání počátečních znaků jejich názvů. (6)

Prostředí taky zahrnuje designer, jehož úkolem je umístit do projektu prvky uživatelského rozhraní a datového přístupu. Při pohodlné manipulaci s grafickými reprezentanty ovládacích prvků se Visual Studio automaticky postará o tvorbu souvisejícího zdrojového kódu v jazyce C# pro vytvoření instancí požadovaných

objektů za běhu programu. Tento kód samozřejmě vloží do příslušných zdrojových souboru. (6)

Podpůrná okna, která umožňují prohlížet a upravovat různé vlastnosti vytvářených programů. V prostředí Visual Studio jsou okna určena pro zobrazení tříd použitých ve zdrojovém kódu nebo vlastností (a jejich počátečních hodnot) tříd definovaných v knihovnách Windows Forms a Web Forms. (6)

1.5.4 Programovací jazyk C#

Zdrojový kód jazyka C# je překládán do spravovaného kódu (managed code). Spravovaný kód je mezi jazykem (intermediate language – IL), jelikož je z poloviny jazykem vyšší úrovně (C#) a jazykem, který je řazen na úroveň nejnižší (jazyk symbolických adres nebo strojový kód). (4)

Prostředí Common Language Runtime za běhu program a průběžně je překládán zdrojový kód metodou Just In Time. Podobným způsobem, jako když jsou používány jiné techniky, má tato metoda své pro a proti. (4)

Jednoznačným záporem tohoto způsobu je neefektivní překlad zdrojového kódu za běhu aplikace. Tento proces se liší od interpretace, kterou obvykle využívají skriptovací jazyky Perl a Jscript. JIT – překladač nepřekládá libovolnou funkci při každém okamžiku, kdy je tato funkce volána; tuto akci udělá pouze při prvním jejím volání a při tomto úkonu vytvoří strojový kód nativní pro platformu, na jejímž řešení daný program běží. (4)

Jednoznačným kladem JIT – překladu je omezení množiny stránek aplikace, protože okamžité paměťové nároky zdrojového kódu jsou menší. Za běhu aplikace je totiž přeložen pouze potřebný zdrojový kód. Například pokud aplikace, která zrovna běží obsahuje zdrojový kód pro tisk, tak nepotřebuje tento zdrojový kód, pokud se uživatel nerozhodne tisknout nějaký dokument a není tedy JIT překladačem nikdy přeložen. (4)

CLR má schopnost optimalizovat způsob vykonání program za jeho běhu. Může tedy například určit způsob redukce chyb stránek ve správci paměti prostřednictvím reorganizace přeloženého kódu v paměti, a to za běhu program. (4)

Obrázek č.2: Jazyk C#
Zdroj: Dle (7)

2 Analýza současného stavu

2.1 Analýza zásob

Podnik ve své nabídce zahrnuje oleje, maziva a další automobilové kapaliny, používané pro osobní a nákladní automobily.

2.1.1 Nafta

První položkou zásoby, kterou jsem si zvolil k analyzování, je tzv. minerální olej - nafta. Po dobu 21 dnů jsem zaznamenával stav zásob této položky a tyto výsledky byly následně zapsány do následující tabulky č. 1. Množství zásoby nafty je v jednotkách litr.

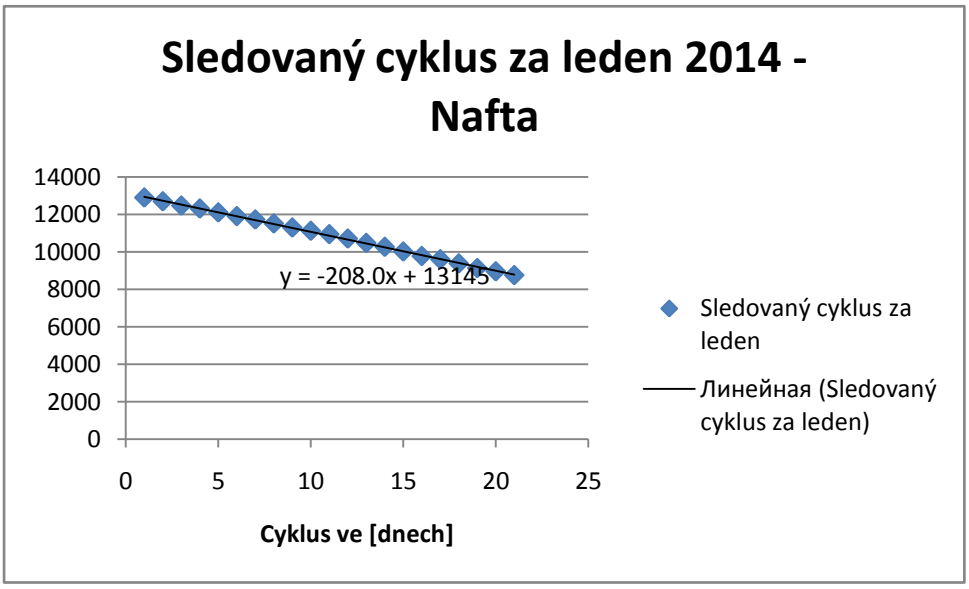
Datum	Den	Množství [l]
17.4.1935	Čtvrtek	12891
3.1.2014	Pátek	12688
6.1.2014	Pondělí	12461
7.1.2014	Úterý	12306
8.1.2014	Středa	12097
9.1.2014	Čtvrtek	11900
10.1.2014	Pátek	11720
13.1.2014	Pondělí	11508
14.1.2014	Úterý	11293
15.1.2014	Středa	11117
16.1.2014	Čtvrtek	10939
17.1.2014	Pátek	10706
20.1.2014	Pondělí	10473
21.1.2014	Úterý	10264
22.1.2014	Středa	10015
23.1.2014	Čtvrtek	9773
24.1.2014	Pátek	9601
27.1.2014	Pondělí	9381
28.1.2014	Úterý	9133
29.1.2014	Středa	8959
30.1.2014	Čtvrtek	8756

Tabulka č. 1: Zaznamenané statistické údaje pro položku nafta
Zdroj: Interní evidence společnosti XY
Zpracování: Vlastní

Z výše uvedené vypracované tabulky č. 1 plyne, že společnost nakupuje zásoby nafty ve větším množství a tento stav je jak deklaroval majitel společnosti doplněn jednou měsíčně.

Průměrná spotřeba zkoumané položky je cca 207 l za den. Tento způsob objednávky je pro společnost výhodnou variantou resp. volbou, protože získává od dodavatele množstevní slevy za nákup nafty.

Po podrobném sledování stavu zásob položky - nafta v intervalu měsíce ledna jsem ji následně vyznačil do grafu č. 1.



Graf č. 1: Průběh stavu zásob – položka nafta
Zdroje: Interní evidence společnosti XY
Zpracování: Vlastní

Graf č. 1 znázorňuje, že položka zásoby nafty má lineární trend, protože společnost objednává tuto položku jednou měsíčně, proto nebylo potřeba rozdělit tento měsíční cyklus na více částí a mohl jsem tedy znázornit regresi v měsíčním intervalu 21 dnů. Společnost výši zásoby a její zpětné objednávání stanovuje podle provozu a podle domluv, které má se svými dodavateli.

Cílem této diplomové práce bude striktně stanovit bod objednávky a to prostřednictvím aplikace, která jej bude dle uživatelských potřeb a zásobovacích vazeb vyhodnocovat, popř. bude uživateli sdělovat, že byla překročena mezní úroveň pod bod objednávky.

2.1.2 Stanovení matematického tvaru přímky položky nafty

Číselné vyjádření stavu zásob vyrovnáme pomocí regresní přímky. Vizuálně jsem tento stav znázornil již v grafu č. 1.

Pokud chceme vyrovnávat stavový trend zásob, budeme muset vypočíst koeficienty b_1 a b_2. Výsledky těchto dvou koeficientů získáme ze vzorce (3.30). Konečný tvar regresní přímky ve sledovaném cyklu, tedy v měsíci lednu má následující podobu:

$\hat{z}(t) = (-208{,}083t) + 13145{,}15$, kde $t \in N$, t zastupuje číslo příslušného dne.

Hodnota parametru b_2 udává průměrnou velikost čerpání položky během jednotkového časového intervalu. Pomocí tohoto koeficientu získáme bod znovu objednávky B_o, v takovém případě, kdy je známa pořizovací lhůta. (2)

Konkrétně v tomto případě to značí, že položka stavu zásob se snížila v průměru o 207 litrů. Nyní v tabulce zaznamenám konkrétní vyrovnané hodnoty položky nafta. K těmto hodnotám jsem dospěl tak, že do matematického tvaru trendu sledované položky jsem za neznámou t dosadil příslušný den sledování. V tabulce č. 3 jsou tedy vypočtené jednotlivé hodnoty po vyrovnání.

Příslušný den	Stav zásob	Hodnota po vyrovnání
1	12891	12937,067
2	12688	12728,984
3	12461	12520,901
4	12306	12312,818
5	12097	12104,735
6	11900	11896,652
7	11720	11688,569
8	11508	11480,486
9	11293	11272,403
10	11117	11064,32
11	10939	10856,237
12	10706	10648,154
13	10473	10440,071
14	10264	10231,988
15	10015	10023,905
16	9773	9815,822
17	9601	9607,739

18	9381	9399,656
19	9133	9191,573
20	8959	8983,49
21	8756	8775,407

Tabulka č. 2: Hodnoty zásob položky nafta po vyrovnání regresní přímkou
Zdroj: Interní evidence společnosti XY
Zpracování: Vlastní

Z matematického tvaru regresní přímky $\hat{\mathbf{z}}(t) = (-208{,}083t) + 13145{,}15$ můžeme také zjistit, kdy pravděpodobně budou vyčerpány zásoby a to následujícím způsobem:

$$t_v = \frac{\mathbf{13145{,}15}}{\mathbf{208{,}083}} \doteq 63{,}2$$

Tento výsledek můžeme interpretovat, že zásoby nafty budou vyčerpány asi za 63 dnů, což jsou 2 měsíce přibližně.

2.1.3 Intervaly spolehlivosti pro regresní přímku čerpání zásob

Pro správné stanovení koeficientů intervalů spolehlivosti jsem si nejprve sestavil následující tabulku č. 3.

x_i	y_i	x_i^2	$x_i y_i$	Regrese	e_i	e_i^2
1	12891	1	12891	12937,067	46,067	2122,168489
2	12688	4	25376	12728,984	40,984	1679,688256
3	12461	9	37383	12520,901	59,901	3588,129801
4	12306	16	49224	12312,818	6,818	46,485124
5	12097	25	60485	12104,735	7,735	59,830225
6	11900	36	71400	11896,652	-3,348	11,209104
7	11720	49	82040	11688,569	-31,431	987,907761
8	11508	64	92064	11480,486	-27,514	757,020196
9	11293	81	101637	11272,403	-20,597	424,236409
10	11117	100	111170	11064,32	-52,68	2775,1824
11	10939	121	120329	10856,237	-82,763	6849,714169
12	10706	144	128472	10648,154	-57,846	3346,159716
13	10473	169	136149	10440,071	-32,929	1084,319041
14	10264	196	143696	10231,988	-32,012	1024,768144
15	10015	225	150225	10023,905	8,905	79,299025
16	9773	256	156368	9815,822	42,822	1833,723684
17	9601	289	163217	9607,739	6,739	45,414121
18	9381	324	168858	9399,656	18,656	348,046336

19	9133	361	173527	9191,573	58,573	3430,796329
20	8959	400	179180	8983,49	24,49	599,7601
21	8756	441	183876	8775,407	19,407	376,631649

Tabulka č. 3: Stanovení koeficientů pro výpočet intervalu spolehlivosti odebírané položky nafta
Zdroj: Interní evidence společnosti XY
Zpracování: Vlastní

V tabulce číslo č. 4 byly provedeny již konkrétní výpočty, pomocí kterých byly vyjádřeny intervaly spolehlivosti.

SR	31470,49008
Odhad rozptylu	1656,341583
Průměr(x)	11
Průměr (y)	10856,2381
$D(B_1)$	339,1556575
$D(B_2)$	2,151092965
Intervaly spolehlivosti A_1	-198,3982684
Intervaly spolehlivosti A_2	-217,7677316

Tabulka č. 4: Výpočet hodnot intervalů spolehlivosti – položka nafta
Zdroj: Interní evidence společnosti XY
Zpracování: Vlastní

Takto vypočítané výsledky je možné interpretovat tak, že interval (-217,77; -198,4) pokrývá parametr β_2 s 95% - ní spolehlivostí. Pokud se počet dní skladované položky zvyšuje, pak je střední hodnota poklesu denních zásob (v litrech) pokryta intervalem **(**-217,77**;** -198,4**)** s 95% - ní spolehlivostí).

2.1.4 Analýza čerpání zásob

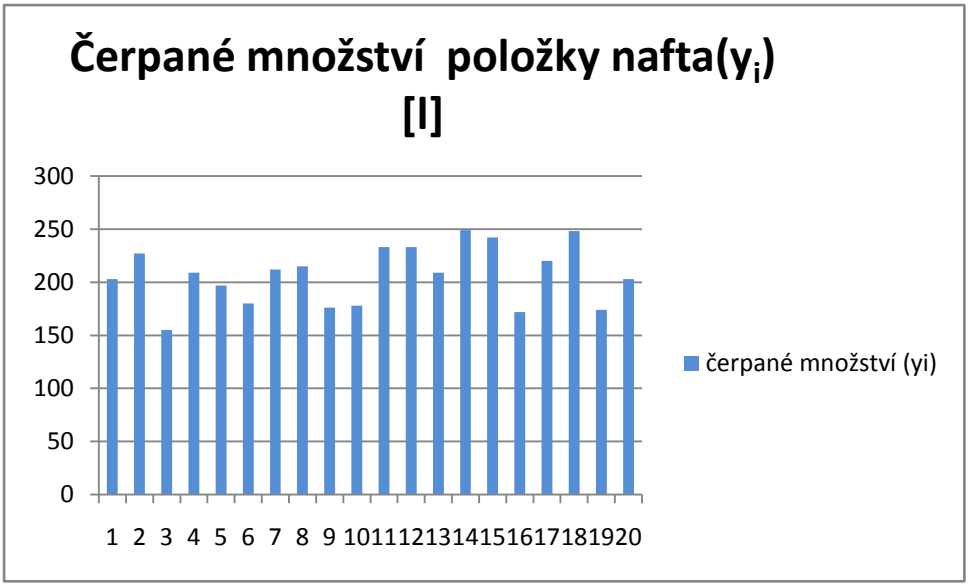
Součástí tzv. analýzy čerpání zásob je také monitorování denních prodejů ve sledovaném cyklu, v tomto případě v měsíci lednu. Smyslem této operace je takový, že bude odhadnutý trend poptávky po tomto zboží.

Pořadí (i)	Čerpané množství (y_i)	Hodnoty po vyrovnání
1	203	196,9238
2	227	197,9576
3	155	198,9914
4	209	200,0252
5	197	201,059

6	180	202,0928
7	212	203,1266
8	215	204,1604
9	176	205,1942
10	178	206,228
11	233	207,2618
12	233	208,2956
13	209	209,3294
14	249	210,3632
15	242	211,397
16	172	212,4308
17	220	213,4646
18	248	214,4984
19	174	215,5322
20	203	216,566

Tabulka č. 5: Zjištěné hodnoty odebíraného množství položky nafty
Zdroj: Interní evidence společnosti XY
Zpracování: Vlastní

Hodnoty zjištěné v tabulce č. 5 jsem zaznamenal do grafu a z těchto dat pomocí vlastního rozboru pak provedl další analýzu.



Graf č. 2: Odebírané množství – položka nafta
Zdroj: Interní evidence společnosti XY
Zpracování: Vlastní

Ze sestaveného grafu č. 2 je patrné, že odběr položky nafta se pohybuje přibližně na stejné úrovni. Z výše uvedených a analyzovaných informací lze také odvodit, že minimální odebírané množství nafty se pohybuje přibližně nad 150 litrů za den.

Další důležitou informací, která je z grafu zřejmá, je fakt, že jsem zde zaznamenal téměř konstantní trend.

Na základě vzorce (3.4) nyní vypočítáme výběrový průměr a výběrový rozptyl velikosti denního čerpání zboží tímto způsobem:

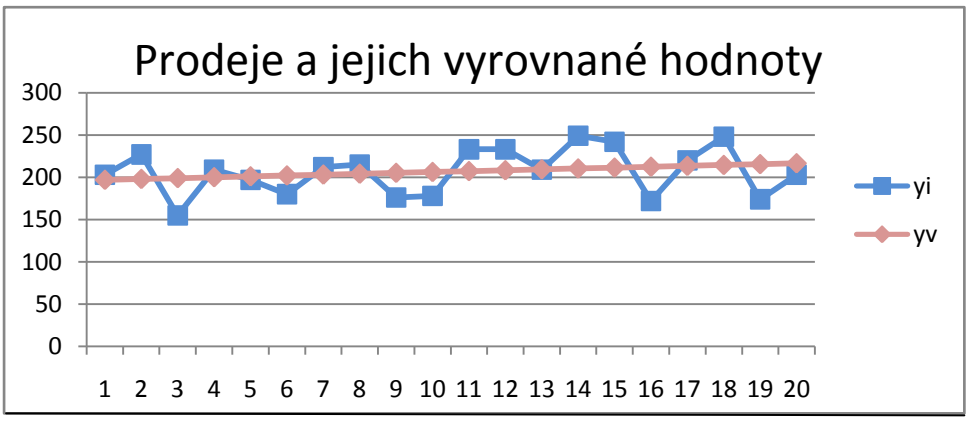
$$\bar{y} = \frac{4135}{20} = 206{,}75$$

$$\mathbf{S^2 = \frac{1}{20-1} * [869279 - 20 * 206,75^2] = 756,1974}$$

Z výsledku vyplývá, že velikost denního čerpání položky nafty se pohybuje okolo průměrné hodnoty 206,75 listrů a výběrový rozptyl činí přibližně 756,1974 litrů.

2.1.5 Vyrovnání hodnot denních prodejů

Pro denní spotřebu položky jsem stanovil rovnici regresní přímky ve tvaru y(t) = 1,0338x + 195,89. Vyrovnané hodnoty obsahuje výše uvedená tabulka č. 5 a následující graf č. 3.



Graf č. 3: Zjištěné a vyrovnané hodnoty spotřeby položky nafty
Zdroj: Interní evidence společnosti XY
Zpracování: Vlastní

Graf č. 3 nám dává náhled na hodnoty denní spotřeby položky nafty; z grafu lze konstatovat, že trend je téměř konstantní. Je také zcela patrné, že čerpání této položky neovlivňuje čas.

2.1.6 Subtotal

Další položkou zásoby, která mi byla poskytnuta k analýze, je položka snázvem subtotal. Po dobu 21 dnů jsem zaznamenával stav zásob této položky a tyto výsledky zapsal do následující tabulky č. 6. Množství zásoby subtotal je v jednotkách litr.

Datum	Den	Množství [l]
17.4.1935	Čtvrtek	5913
3.1.2014	Pátek	5787
6.1.2014	Pondělí	5648
7.1.2014	Úterý	5546
8.1.2014	Středa	5416
9.1.2014	Čtvrtek	5293

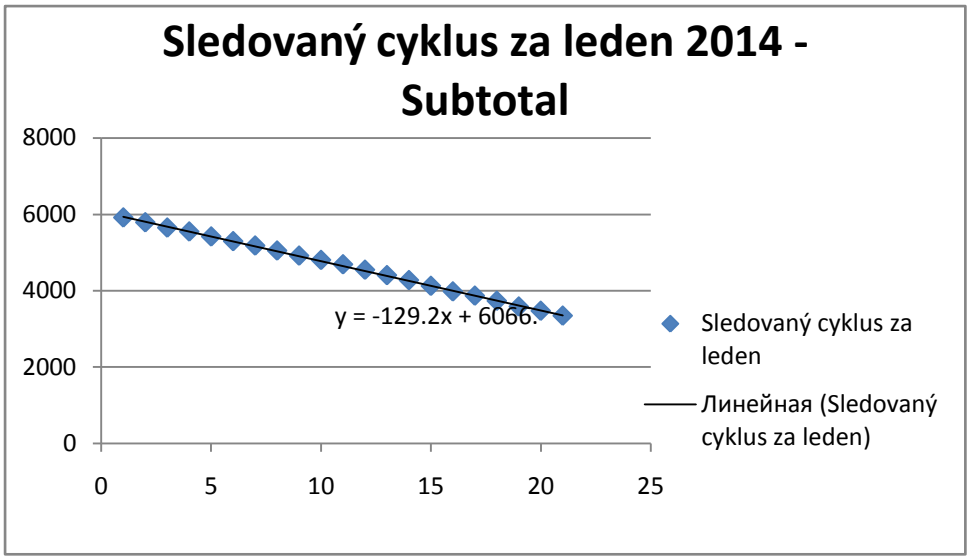
10.1.2014	Pátek	5178
13.1.2014	Pondělí	5047
14.1.2014	Úterý	4914
15.1.2014	Středa	4801
16.1.2014	Čtvrtek	4687
17.1.2014	Pátek	4545
20.1.2014	Pondělí	4403
21.1.2014	Úterý	4273
22.1.2014	Středa	4123
23.1.2014	Čtvrtek	3977
24.1.2014	Pátek	3866
27.1.2014	Pondělí	3731
28.1.2014	Úterý	3582
29.1.2014	Středa	3470
30.1.2014	Čtvrtek	3343

Tabulka č. 6: Zaznamenané statistické údaje pro položku subtotal
Zdroj: Interní evidence společnosti XY
Zpracování: Vlastní

Z výše uvedené tabulky č. 6 plyne, že společnost nakupuje tyto zásoby subtotalu v poměrně velkém množství a tento stav, jak opět deklaroval majitel společnosti, je doplněn jednou měsíčně, tedy stejným způsobem jako v předchozím případě.

Průměrná spotřeba zkoumané položky je cca 129 l za den. Tento způsob objednávky je pro společnost opětovně výhodnou volbou, protože získává od dodavatele na subtotal společně i s naftou množstevní slevy za nákup obou položek

Při sledování stavu zásob položky subtotal v intervalu 1 měsíce jsem následně toto zaznamenal do grafu č. 4.



Graf č. 4: Průběh stavu zásob – položka subtotal
Zdroj: Interní evidence společnosti XY
Zpracování: Vlastní

Graf č. 4 nyní znázorňuje, že položka zásoby subtotal má lineární trend a jako u předchozí sledované položky nafty společnost objednává položku jednou za měsíc. Proto nebylo potřeba rozdělit tento měsíční cyklus na více částí a mohl jsem tedy znázornit regresi v měsíčním intervalu 21 dnů. Společnost výši zásoby a její zpětné objednávání stanovuje podle provozu a podle předem stanovených podmínek, které má se svými dodavateli.

2.1.7 Stanovení matematického tvaru přímky položky subtotal

Číselné vyjádření stavu zásob vyrovnáme pomocí regresní přímky. Graficky jsem tento stav již znázornil v grafu č. 4.

Pokud chceme vyrovnávat stavový trend zásob, budeme muset vypočíst koeficienty b_1 a b_2. Výsledky těchto dvou koeficientů získáme ze vzorce (3.30), který již byl zmíněn u položky nafta. Konečný tvar regresní přímky ve sledovaném cyklu (měsíc leden) u zkoumané položky subtotal má následující podobu:

$\hat{z}(t) = (-129{,}2t) + 6066{,}1$, kde $t \in N$, t zastupuje číslo příslušného dne

Tento případ ukazuje, že položka stavu zásob se snížila v průměru o 207 litrů. Nyní v tabulce zaznamenáme konkrétní vyrovnané hodnoty položky subtotal. K těmto

hodnotám jsem opět přišel tím způsobem, že do matematického tvaru trendu sledované položky jsem za neznámou t dosadil příslušný den sledování. V tabulce č. 7 jsou také zaznamenány vypočtené hodnoty po vyrovnání.

Příslušný den	Stav zásob	Hodnota po vyrovnání
1	5913	5936,9
2	5787	5807,7
3	5648	5678,5
4	5546	5549,3
5	5416	5420,1
6	5293	5290,9
7	5178	5161,7
8	5047	5032,5
9	4914	4903,3
10	4801	4774,1
11	4687	4644,9
12	4545	4515,7
13	4403	4386,5
14	4273	4257,3
15	4123	4128,1
16	3977	3998,9
17	3866	3869,7
18	3731	3740,5
19	3582	3611,3
20	3470	3482,1
21	3343	3352,9

Tabulka č. 7: Hodnoty zásob položky subtotal po vyrovnání regresní přímkou
Zdroj: Interní evidence společnosti XY
Zpracování: Vlastní

Z matematického tvaru regresní přímky $\hat{z}(t) = (-129,2t) + 6066,1$ můžeme také zjistit, kdy pravděpodobně budou vyčerpány zásoby a to následujícím způsobem:

$$t_v = \frac{6066,1}{129,2} \doteq 46,95$$

Tímto výsledkem můžeme interpretovat, že zásoby nafty budou vyčerpány asi za 47 dnů, což je měsíc a 17 dní přibližně.

2.1.8 Intervaly spolehlivosti pro regresní přímku čerpání zásob

Pro stanovení koeficientů intervalů spolehlivosti byla sestavena tabulka č. 8, při níž byl použit obdobný postup výpočtu ke stanovení koeficientů intervalů spolehlivosti, jako to bylo u položky nafta.

x_i	y_i	x_i^2	$x_i y_i$	Regrese	e_i	e_i^2
1	5913	1	5913	5936,9	23,9	571,21
2	5787	4	11574	5807,7	20,7	428,49
3	5648	9	16944	5678,5	30,5	930,25
4	5546	16	22184	5549,3	3,3	10,89
5	5416	25	27080	5420,1	4,1	16,81
6	5293	36	31758	5290,9	-2,1	4,41
7	5178	49	36246	5161,7	-16,3	265,69
8	5047	64	40376	5032,5	-14,5	210,25
9	4914	81	44226	4903,3	-10,7	114,49
10	4801	100	48010	4774,1	-26,9	723,61
11	4687	121	51557	4644,9	-42,1	1772,41
12	4545	144	54540	4515,7	-29,3	858,49
13	4403	169	57239	4386,5	-16,5	272,25
14	4273	196	59822	4257,3	-15,7	246,49
15	4123	225	61845	4128,1	5,1	26,01
16	3977	256	63632	3998,9	21,9	479,61
17	3866	289	65722	3869,7	3,7	13,69
18	3731	324	67158	3740,5	9,5	90,25
19	3582	361	68058	3611,3	29,3	858,49
20	3470	400	69400	3482,1	12,1	146,41
21	3470	441	70203	3352,9	9,9	98,01

Tabulka č. 8: Stanovení koeficientů pro výpočet intervalu spolehlivosti odebírané položky subtotal
Zdroj: Interní evidence společnosti XY
Zpracování: Vlastní

V tabulce číslo č. 9 byly provedeny již konkrétní přesné výpočty a byly vyjádřeny intervaly spolehlivosti.

SR	8138,21
Odhad rozptylu	428,3268421
Průměr (x)	11
Průměr (y)	4644,904762
D(B1)	87,70502005
D(B2)	0,556268626
Intervaly spolehlivosti A1	-128,552353
Intervaly spolehlivosti A2	-129,847647

Tabulka č. 9: Výpočet hodnot intervalů spolehlivosti – položka subtotal
Zdroj: Interní evidence společnosti XY
Zpracování: Vlastní

Vypočtené výsledky je možné interpretovat tak, že interval (-129,85; -128,55) pokrývá parametr β_2 s 95% - ní spolehlivostí. Pokud se počet dní skladované položky zvyšuje, pak je střední hodnota poklesu denních zásob (v litrech) pokryta intervalem
(-129,85; -128,55) s 95% - ní spolehlivostí.

2.1.9 Analýza čerpání zásob

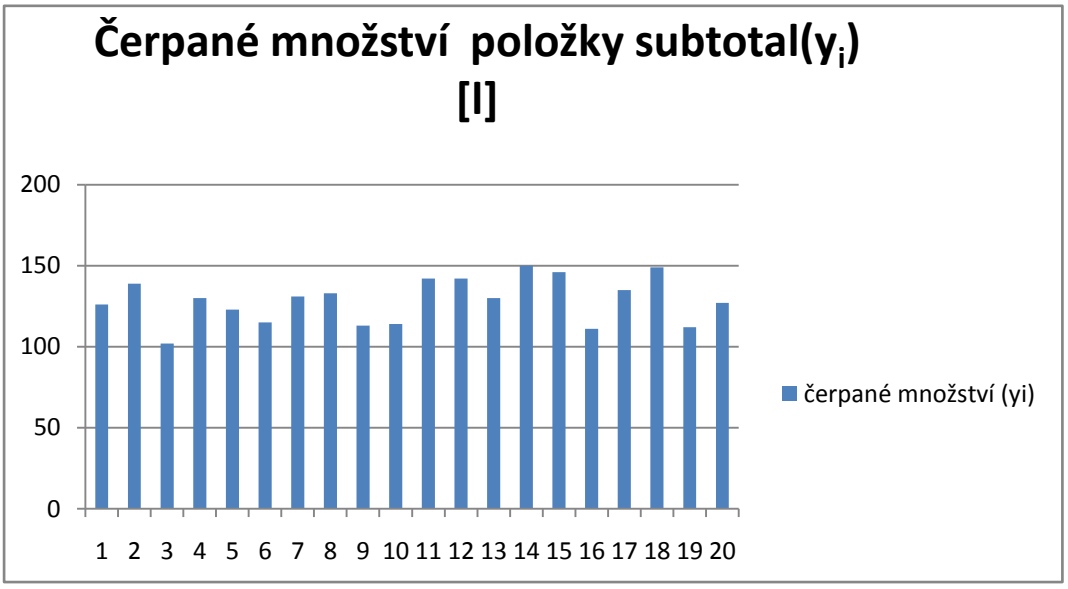
Dále jsem pokračoval časovou analýzou prodejů ve sledovaném cyklu (vybraný měsíc leden) pro odhadnutí trendu poptávky po tomto zboží.

Pořadí (i)	čerpané množství (y_i)	Hodnoty po vyrovnání
1	203	123,4238
2	227	123,9576
3	155	124,4914
4	209	125,0252
5	197	125,559
6	180	126,0928
7	212	126,6266
8	215	127,1604
9	176	127,6942
10	178	128,228
11	233	128,7618

12	233	129,2956
13	209	129,8294
14	249	130,3632
15	242	130,897
16	172	131,4308
17	220	131,9646
18	248	132,4984
19	174	133,0322
20	203	133,566

Tabulka č. 10: Zjištěné hodnoty odebíraného množství položky subtotal
Zdroj: Interní evidence společnosti XY
Zpracování: Vlastní

Hodnoty zjištěné v tabulce č. 10 byly zaznamenány do grafu a z těchto dat pak byla provedena další analýza.



Graf č. 5: Průběh stavu zásob – položka subtotal
Zdroj: Interní evidence společnosti XY
Zpracování: Vlastní

Ze sestaveného grafu č. 5 je opět patrné, že odběr položky nafta se pohybuje přibližně na stejné úrovni. Z výše uvedených informací lze také odvodit, že minimální odebírané množství se pohybuje přibližně kolem 100 litrů za den.

Další informace, která z grafu plyne, je fakt, že jsem i zde zaznamenal téměř konstantní trend jako v předchozím případě s položkou nafty.

Na základě již známého vzorce (3.4) byl vypočítán výběrový průměr a výběrový rozptyl velikosti denního čerpání zboží:

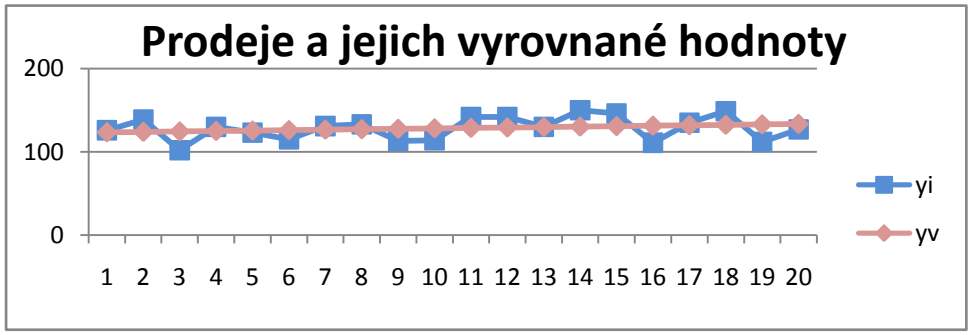
$$\bar{y} = \frac{2570}{20} = 128{,}5$$

$$\mathbf{S^2 = \frac{1}{20-1} * [333934 - 20 * 128,5^2] = 194,16}$$

Z uvedených výsledku vyplývá, že velikost denního čerpání položky nafty se pohybuje okolo průměrné hodnoty 128,5 litrů a výběrový rozptyl činí přibližně 194,16 litrů.

2.1.10 Vyrovnání hodnot denních prodejů

Pro denní spotřebu položky jsem stanovil rovnici regresní přímky ve tvaru y(t) = 1,0338x + 195,89. Vyrovnané hodnoty obsahuje tabulka č. 10 a následující graf.



Graf č. 6: Zjištěné a vyrovnané hodnoty spotřeby položky subtotalu
Zdroj: Interní evidence společnosti XY
Zpracování: Vlastní

Graf č. 6 znázorňuje hodnoty denní spotřeby položky - subtotal. Z grafu opět můžeme zaznamenat téměř konstantní trend. Je také zcela patrné, že čerpání této položky neovlivňuje časová proměnná.

3 Vlastní návrhy řešení

3.1 Analýza vybavení a potřeb společnosti

3.1.1 Hardware

Hardwarové vybavení, které má společnost XY ve svém vlastnictví je tvořeno poměrně staršími počítačovými sestavami. Přestože je tato výbava v dnešní době z mého pohledu poměrně zastaralou, pro potřeby této organizace je dostačující. Firma rovněž disponuje síťovou infrastrukturou, jejímž úkolem je koordinace uživatelských operací na bázi vyřizování prodeje a objednávek. Vybavení této společnosti popisuje tabulka č. 11.

Umístění	Procesor	RAM	Hardisk
Kancelář ředitele	iCeleron 2,66 GHz	1 GB	120 GB
Kancelář ředitele	Intel Core 1,8 GHZ	2 GB	120 GB
Účetní oddělení	iCeleron 2,4 GHz	512 MB	80 GB
Účetní oddělení	iCeleron 2,4 GHz	512 MB	80 GB
Prodejna	iCeleron 2,4 GHz	512 MB	40 GB
Kancelář pro administrativní pracovníky	iCeleron 2,4 GHz	512 MB	80 GB
Kancelář ekonomického ředitele	Intel Core 2,8 GHz	1 GB	80 GB

Tabulka č. 11: Položky pro navrženou datovou strukturu
Zdroj: Interní evidence společnosti XY
Zpracování: Vlastní

3.1.2 Software

Softwarové vybavení společnosti je nastaveno k manipulaci běžných podnikových operací, jako je účetní agenda k vyřizování objednávek v rámci odběratelských a dodavatelských vztahů. Tyto procesy jsou realizovány prostřednictvím elektronické pošty. Operační systém, který společnost využívá je Windows 7 a Windows XP. Působení těchto operačních systémů závisí na konkrétních hardwarových komponentách jednotlivých počítačových sestav.

Dalším softwarem, který je úzce propojen s problematikou této diplomové práce, je řešení, které se zabývá stavem zásob jednotlivých položek, které jsou v podniku uskladněny. Jedná se o software Horry, který je určen pro ekonomickou a logistickou oblast podnikání. Tento software využívá architekturu sklad - skladová evidence, kdy je

evidováno, jaké množství produktu je denně vyčerpáno. Na počátku nového dne majitel kontroluje stav této uvedené položky. Nevýhodou tohoto softwaru je, že dává pohled na stav zásob jen pro daný denní časový okamžik a nikde tyto záznamy neukládá.

Firma má také svůj elektronický obchod, avšak tato služba je vykonána prostřednictvím outsourcingu.

Co se týká zhodnocení softwarového řešení společnosti, domnívám se vzhledem k velikosti organizace, že pokud tyto starší programy stále zastávají svůj účel použití, není potřeba je měnit za novější a zbytečně tak plýtvat finančními prostředky.

3.1.3 Orgware

Současný informační systém samozřejmě nabízí možnosti přístupových práv, ale ve společnosti jsou přístupové práva a možnosti provádění různých operací s informačním systém administrátorsky nastavena na míru podniku. Ve společnosti není žádný specializovaný software, jehož špatným využíváním ze strany uživatelů by mohlo vzniknout riziko, které by mohlo potenciálně ohrozit chod společnosti.

Pokud by se však vyskytly potíže související s IS, tak je společnost řeší prostřednictvím svého IT oddělení. Musím podotknout, že podnik dodnes nestanovil žádné organizační normy, které by představovaly předpisy, které se zabývají činností v oblasti řízení problémů IS. Pokud by k neočekávané situaci opravdu došlo, je velmi důležité, aby se případné výpadky systému řešily v co nejkratší možné době, jelikož v případě dlouhotrvajícího výpadku systému by společnosti mohly vzniknout ztráty např. v oblasti financí.

3.1.4 Požadavky na daný software

Prostřednictvím strukturovaného rozhovoru s majitelem podniku „XY“ jsem vyhodnotil, že společnost nedisponuje softwarovým řešením, kde by byl pevně zabudován systém objednávání.

V případě jednotlivých objednávek podnik využívá svých zkušeností a znalostí, které vycházejí z provozu společnosti; do tohoto procesu je tedy více zapojena intuice a odborný odhad.

Dosavadní způsob a systém objednávky je tedy závislý na lidském faktoru, který

je jistě odborný, nikoliv však neomylný. Z tohoto důvodu a v případě velkého množství položek na skladě se ztrácí i při současném stavu softwaru přehled o momentální výši množství skladovaných položek.

Cílem softwarového řešení bude tedy návrh aplikace, která by usnadnila sledování stavu zásob a určila bod objednávky do stavu, kdy by tuto činnost mohl provádět i člověk nezkušený. Dané softwarové řešení by mělo také redukovat špatná rozhodnutí, což by mohlo ušetřit nemalé finanční prostředky za skladování.

Vedení by tímto nástrojem mohlo dosáhnout kontroly nad činností zaměstnanců této problematiky a následně přijat opatření, jak by tento proces mohl být zefektivněn.

Je potřeba také připomenout fakt, který je úzce svázán s počtem položek zboží či výrobků. Samotný výpočet této problematiky není složitý, jde však o to, že bude potřeba využívat velké množství dat, proto bude také kladen důraz na systém, který bude rychlý a přesný.

Majitel společnosti projevil intenci o využití tohoto systému osobně na svém počítači, protože rozhodování o množství objednaného zboží spadá pod jeho kompetenci a aplikace by tedy měla být autonomní vůči svému okolí. Měla by však vycházet z programu (Horry viz. kapitola HOS), který hlásí denní stav zásob položky a tyto numerické údaje by pak byly vloženy do navržené aplikace a poté vyhodnocovány.

.

3.1.5 Technické požadavky na aplikaci

Mnou navržená aplikace poběží na kterémkoliv PC a je primárně určena pro operační systémy Windows. Pro program není vyžadována instalace žádných speciálních programů do PC uživatele.

3.1.6 Požadavky na aplikaci

Postup, jak bude probíhat návrh a realizace aplikace je realizován prostřednictvím běžné konzultace se zákazníkem v kombinaci s agilním přístupem, kdy zpracovatel vytváří daný projekt na základě svých nápadů a představ.

Výsledky těchto metod by měl dát obecný náhled k tomu, jakou by měla mít aplikace filosofii a co by měla fakticky provádět.

Předpokladem těchto metod je také srozumitelnost pro obě strany – jak pro zákazníka, tak i pro programátora.

V případě, kterým se zabývá tato diplomová práce, je požadavek následující, uživatel bude do programu vkládat data a z těchto dat pak bude vyhodnocovat výše popisované statistické metody pro bod objednávky.

3.1.7 Členění problému

V této fázi se provede členění dané problematiky na související části. Na základě tohoto rozdělení se vývojář rozhodne, do jakých kategorií jednotlivé části členění rozdělí. Operace proběhne tak, že jsou navrženy třídy a poté jsou vytvořeny vztahy mezi těmito třídami

Mezi jednotlivými třídami se vytvoří vztahy a tím je připraven základ programu.

V mém případě jsem využil návrhový vzor MVC (Model-view-controller), který umožňuje explicitně oddělit část programu zabývající se vstupem a předzpracováním zadávaných příkazů od části reakcí na zadané příkazy a části mající na starosti výstup výsledků. V čisté verzi má každou z těchto činností na starosti samostatný objekt resp. skupina objektů. Každá složitější aplikace se principiálně skládá ze 3 částí, jenž spolu úzce spolupracují:

a) model - načítá data z datového zdroje (většinou databáze),

b) viewer - zobrazuje/formátuje data z modelu,

c) controller - provádí komunikaci s uživatelem, také obsahuje logiku aplikace

(například použije-li se události stisku tlačítka "ukládat data", controller spustí model, který data a následně zobrazí pomocí vieweru hlášení, že data byla uložena).

3.1.8 Nástroje pro vývoj aplikace

K vytvoření zdrojového kódu jsem zvolil programovací jazyk C#, který je součástí NET frameworku a databázi, která záznamy ukládala byla v XML formátu. Tento jazyk jsem zvolil z důvodu jeho rozšířenosti a jednoduchosti volně. V případě mojí aplikace jsem využil tyto konkrétní nástroje:

- viewer diagram zabudovaný přímo ve Visual Studiu
- visio na simulování vztahů mezi třídami

- k návrhu tříd, objektů a jejich metod jazyk C#
- k vytvoření uživatelského rozhraní knihovnu Windows Forms

Návrh byl pak realizován ve vývojovém prostředí Visual Studio 2012 od společnosti Microsoft.

3.1.9 Návrh aplikace

V samotném vývoji aplikace bylo nutné nejdříve vytvořit rozhraní, které bude komunikovat se samotným uživatelem. Základními kameny tohoto rozhraní byl návrh jednotlivých prvků ve vývojovém prostředí, kde jsem využil objekty, které jsou součástí knihovny Windows Forms. Protože se jedná o autonomní aplikaci, která neslouží k vizuální prezentaci, ale čisté k rozhodovacím účelům, nevyužil jsem žádné grafické prvky.

Navržená aplikace StoreStatistics je složena ze 3 formulářů:

- 1. formulář uživatele uvede do programu a dotazuje se jej, jakou akci chce zvolit. Uživatel si prostřednictvím objektu RadioButton zvolí požadovanou akci a je přesměrován na další formulář, kde se tato akce nachází
- 2. formulář slouží k přidání nového výrobku do databáze podle příslušného roku a také slouží k přidání množství k vybranému produktu, které je možné zvolit v objektu ComboBox
- 3. formulář slouží již k přímému výpočtu požadovaných výstupů dle roku a dle zvoleného výrobku

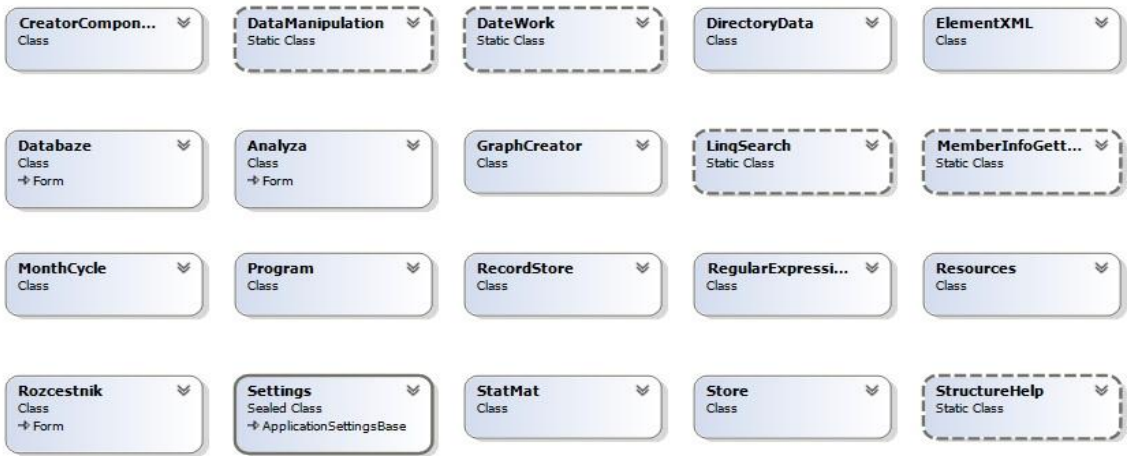
Bližší seznámení s funkčností aplikace bude představena v následující kapitole, kde nejdříve demonstruji zmíněný diagram tříd a podle něho pak budu dále popisovat jednotlivé části kódu, které byly použity při sestavení výsledného programu.

3.1.10 Vývoj aplikace

Třídy projektu

Pro navrženou aplikaci jsem vytvořil několik tříd, které byly navrženy tak, aby řešily jednotlivé oblasti problému, které se v programu vyskytly. Pro dané třídy jsem nevytvářel dědičnost nebo abstraktní třídy, ale řešil jsem konkrétní části už přímo

prostřednictvím navržené třídy a jejích metod. Následující obrázek č. 5 slouží k demonstraci a k náhledu jednotlivých tříd využívaných v programu StoreStatistics. V dalších příspěvcích prostřednictvím diagramu vyjádřím jejich závislosti a popíšu funkcionalitu jejich konkrétních metod.



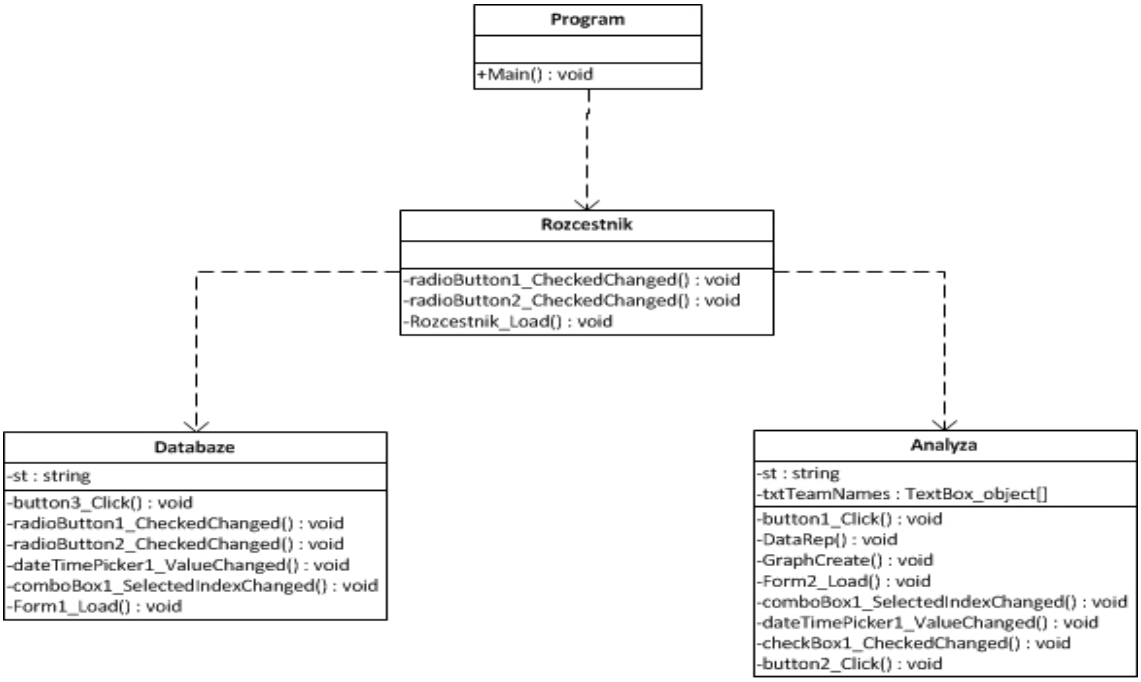
Obrázek č.5: Class Viewer
Zdroj: Vlastní

Závislosti mezi třídami

V této části budou demonstrovány jednotlivé závislosti mezi třídami, pod kterými spadají dílčí řešení problému v aplikaci, a to tím způsobem, že zde demonstruji, jakou mají platnost a jednotlivé vztahy mezi sebou. Např. implementace metody třídy, která abstrakci transformuje do vizuální podoby např. prostřednictvím GUI.

V první části, kde je řešena tato závislost mezi třídami jsem se rozhodl nejprve uvést závislost mezi třídami, které jsou stavebním kamenem aplikace StoreStatistics a poté na tyto formulářové třídy naváží třídy, které v těchto třídách řeší již konkrétní úlohy nezbytné pro běh aplikace.

Na obrázku č. 6 jsem vytvořil diagram čtyř základních tříd, na jejichž bázi celá aplikace tedy běží. Jedná se o formulářové třídy knihovny Windows Forms, jež dle hierarchického nákresu řídí celou aplikaci. Třída program je nejvýše v této hierarchii následována třídou rozcestník, která je druhá nejvyšší, a na poslední úrovni hierarchie jsou pak formulářové třídy Databaze a Analyza.



Obrázek č.6: Formulářové třídy
Zdroj: Vlastní

Funkce výše popsaných tříd:

Program - ve třídě Program se nachází funkce Main, která spouští a také je v ní vyvolána instance na formulářovou třídu Rozcestník, která je spuštěna jako první.

Rozcestník - ve třídě rozcestník jehož součástí jsou tři události, které jsou zaznamenány na obr. č.6, jsou vytvořeny instance na třídu Databaze a na třídu Analyza. Tyto instance, pak vyvolají metodu show příslušné třídy, jež je vyvolána prostřednictvím jedné z událostí radioButton.

Databaze - tato třída je navržena k tomu, aby zákazník mohl přidávat nový produkt do databáze anebo také může přidávat množství do již existujících výrobků. Toto množství pak následně slouží ke statistickým výpočtům, které jsou předmětem diplomové práce.

Analyza - tato třída slouží již k samotným výpočtům bodu objednávky a k vyjádření regresní přímky na základě počtu cyklů ve sledovaném období a vizualizaci grafu s lineárním trendem.

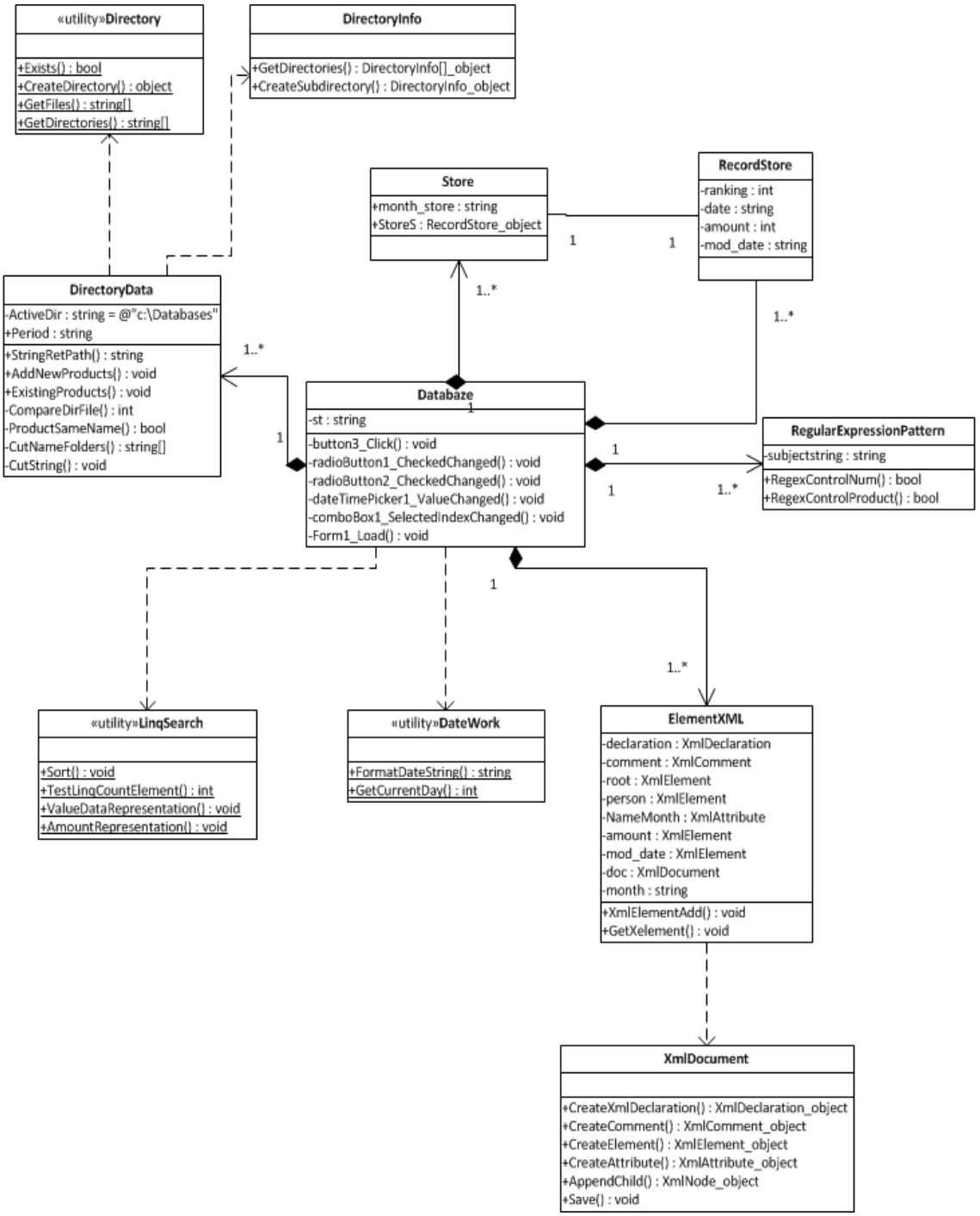
3.1.11 Třída Databaze

V další části bude popsána již dílčím způsobem třída Databaze. Tato třída je složena z několika dalších tříd, které řeší dané úkoly, jejichž výstupem je vytvořit databázi výrobků, ze kterých se poté provede statistická analýza dle třídy Analyza.

Na obrázku č. 7 je demonstrována závislost mezi třídami. Tato akce je opět provedena prostřednictvím UML diagramu.

Některé třídy, jež jsou implementovány ve třídě Databaze budou znovupoužity i ve třídě Analyza, a tudíž budu popisovat jejich funkcionalitu pouze v jednom z případů.

Koncept úložiště dat jsem zvolil ve formátu XML a to z toho důvodu, že je to nejlevnější řešení. Toto řešení je i přenositelné a multiplatformní oproti například relačním databázím, které jsou řešeny v prostředí MSSQL serveru.



Obrázek č. 7: UML diagram závislostí třídy Databaze
Zdroj: Vlastní

Funkce výše popsaných tříd:

Store - jedná se o veřejnou třídu, která je spolu se třídou RecordStore základním kamenem pro sestavení datového modelu, který pak následně slouží pro ukládání vstupních dat do databáze.

Původně tento sestavený datový model byl určen k procesu serializace, což byl prvopočáteční návrh jak veřejné vlastnosti a datové složky převést na prvky nebo atributy XML. Nakonec jsem tento převod vyřešil jiným způsobem.

V této třídě byl jeden atribut, který patřil v navržených XML databázích včetně kořenových značek mezi hierarchicky nejvyšší a tvořil základní strukturu dokumentu. Další člen této třídy byl odkaz na třídu RecordStore, která už definovala více atributů pro XML dokument.

RecordStore - tato veřejná třída tvoří společně se třídou Store (viz. výše) kompletní datovou strukturu, pro navržené úložiště. Vztah, který je však mezi těmito třídami neodpovídá dědičnosti.

Záměrem existence tříd Store a RecordStore byl důkaz, že jsem chtěl model datové struktury rozdělit na více elementárních složek.

V popisované třídě RecordStore jsou již konkrétní datové složky, které jsou základem struktury uložených dat v XML souborech. Pro používané databáze jsou to tyto datové složky vytvořeny z následující částí:

Název položky	Datový typ
Ranking	Int
Amount	Int
Date	String

Tabulka č. 12: Položky pro navrženou datovou strukturu
Zdroj: Vlastní

Popis těchto položek lze intuitivně odhadnout. Položka Ranking značí pořadí daného záznamu, dále položka Date obsahuje datum vložení záznamu a nejdůležitější část struktury položka Amount, která v sobě nese číselnou informaci stavu zásob. Prostřednictvím tohoto záznamu je následně provedena statistická analýza.

DirectoryData - úkolem této veřejné třídy je poskytnout metody, které po přidání nového výrobku, budou vytvářet souborový systém dle zadaného roku položky.

V konkrétním případě je to vyřešeno následovným způsobem – po prvním přidání zboží do databáze je v prostoru „@"c:\Databases"“ vytvořena složka a do ní jsou pak následně ukládány jednotlivé záznamy dle vybraného roku.

Třída obsahuje několik metod, které kupříkladu provádějí kontrolu, zda-li není již zadaný výrobek ve složce sledovaného období, pokud zjistí aplikace, že již k této události došlo upozorní uživatele, že tento úkon byl proveden. Metody této třídy jsou samozřejmě vybaveny i dalšími stavy, které informují uživatele o jeho nečekaných krocích.

Directory - statická třída Directory, která je přímo knihovnou, kterou vytvořili vývojáři od společnosti Microsoft, poskytuje metody pro práci s adresáři.

Prostřednictvím této třídy bylo pak snazší vytvořit, odstranit nebo přesunout operace adresářů a podadresářů.

Vzhledem k zmiňovanému statickému charakteru třídy Directory, nemůže tato třída vytvořit své instance. Tedy vyvolání metody třídy Directory lze přímo ze samotné třídy Directory. Účelem implementace této třídy bylo de facto zjednodušení práce při práci s adresářovou strukturou.

DirectoryInfo – je třída podobného charakteru jako třída Directory. I při tomto faktu jsem se danou třídu rozhodl ve svém projektu použít, protože ve své podstatě, dává prostor k rozšíření funkcionality navržené třídy DirectoryData.

Slouží i ke stejnému účelu jako třídy Directory, přičemž členové třídy DirectoryInfo jsou členy instance, na rozdíl od třídy Directory, která jak jsem již zmínil je statická.

Hlavní rozdílný znak mezi oběma třídami spočívá v samotném použití těchto tříd. Třída Directory lze použít, pokud chceme vyvolat jednoduše složkové operace všechny najednou. Například při odstranění složky.

Třída DirectoryInfo je pevně asociována s adresářem a dává uživateli k dispozici rozhraní, které nabízí velkou množinu metod, které jsou aplikovány přímo na tento adresář. Třída DirectoryInfo přijímá PATH (cesta příslušného adresáře) jako parametr

při vytváření instance a poskytuje uživateli možnost přímé manipulace. Je tedy možné vytvářet podadresáře, přesunout je, vyjmenovat atd.

RegularExpressionPattern - tato třída má za úkol řešit situace, jejichž charakter spadá spíše do technické oblasti a to konkrétně do oblasti regulárních výrazů.

Třída má nadefinovaný vzor dvou situací, kdy uživatel zadává vstupní hodnoty a to ve formě textu pro analyzovanou položku a ve formě hodnoty pro stanovené množství zásoby položky.

Pro numerické hodnoty je potřeba ověření, zda se opravdu jedná o hodnoty numerická, aby pak v databázi nebyly nelogické hodnoty, které by pak mohly bránit dalším výpočtům a stanovení výsledné analýzy.

Pro hodnoty řetězcové, jež udávají název výrobku, jde spíše o úpravu, která by měla uživatele směřovat k tomu, aby zadával logické názvy produktu.

ElementXML - účel existence této třídy je takový, že datovou strukturu ze tříd RecordStore a Store transformuje do podoby XML souborů. Metody této třídy se také starají o zadávání nových výrobků do XML souborů a také k zadávání analyzovaných dat.

Jak jsem se zmiňoval v předchozích kapitolách, k této transformaci by šlo využít i metody serializace, kterou jsem se nakonec pro tuto transformaci rozhodl nevyužít.

I přesto, že předmětem této diplomové práce není samotná analýza sestaveného kódu, chtěl bych alespoň u tohoto případu nastínit, jak jsem postupoval a navrženou funkci jsem sestavil prostřednictvím kombinace lambda výrazů a dalších technik používaných v jazyce C# a to následujícím způsobem:

1. Sestavil jsem si statickou třídu, jejíž funkce bude vracet název atributu. Tento název poté použiju k označení tagu v XML souboru.

```
public static class MemberInfoGetting
{
    public static string GetMemberName<T>(Expression<Func<T>> memberExpression)
    {
        MemberExpression expressionBody = (MemberExpression)memberExpression.Body;
        return expressionBody.Member.Name;
    }
}
```

Obrázek č. 8: Funkce pro transformační etapu do XML
Zdroj: Vlastní

2. Vyvolání statické funkce GetMemberName ze statické třídy MemberInfoGetting a následovná transformace do XML

```
public void XmlElementAdd(Store store,RecordStore recordstore,string textbox,string PATH)
{
    //XmlDocument doc = new XmlDocument();
    declaration = doc.CreateXmlDeclaration("1.0", "UTF-8", "yes");
    comment = doc.CreateComment("This is an XML Generated File");
    root = doc.CreateElement(recordstore.ToString());
    person = doc.CreateElement(MemberInfoGetting.GetMemberName(() => store.StoreS));
    NameMonth = doc.CreateAttribute("Month");
    amount = doc.CreateElement(MemberInfoGetting.GetMemberName(() => recordstore.Amount));
    mod_date = doc.CreateElement(MemberInfoGetting.GetMemberName(() => recordstore.Mod_date));

    NameMonth.Value = Month;
    amount.InnerText = textbox;
    mod_date.InnerText = " " + Convert.ToString(DateTime.Now);

    //Construct the document
    doc.AppendChild(declaration);
    doc.AppendChild(comment);
    doc.AppendChild(root);
    person.Attributes.Append(NameMonth);
    person.AppendChild(amount);
    person.AppendChild(mod_date);
    root.AppendChild(person);
```

Obrázek č. 9: Transformace do XML
Zdroj: Vlastní

XmlDocument - .NET framework poskytuje třídy, které mohou číst a manipulovat s XML dokumenty. Tyto třídy jsou umístěny v prostoru System.XML.*. NET a využívají ke svému účelu XML Document Object Model, což je sada tříd, které představují různé části XML dokumentu (i samotný dokument).

XML dokument je reprezentován třídou XmlDocument. Obsahuje aktuální XML tagy, které je možné číst nebo s nimi manipulovat.

Každý XML dokument by měl začínat s kořenovým elementem, který se také nazývá prvek dokumentu, který je reprezentován jako XmlElement. Všechno, co spadá hierarchicky pod tento prvek, se nazývá podřízené uzly. Jejich hierarchická důležitost je intuitivně nižší.

Tuto třídu jsem se rozhodl implementovat ve výše zmiňované třídě ElementXML, což mi zjednodušilo značně práci při vytváření metod, které svým charakterem spadaly do třídy ElementXML.

DateWork - tato třída slouží pro podporu rozhodování aplikace v situaci, kdy má být uloženo příslušný počet záznamů ve zvoleném měsíci do XML souboru.

Laicky řečeno, funkce vyhodnocuje kolik má měsíc v daném roce počet pracovních dní. Podoba této funkce vypadá následovně:

```
public static int GetCurrentDay(string year, string month)
{
    int workdays = 0;
    var weekends = new DayOfWeek[] { DayOfWeek.Saturday, DayOfWeek.Sunday };
    int daysInMonth = DateTime.DaysInMonth(Convert.ToInt32(year), Convert.ToInt32(month));
    IEnumerable<int> businessDaysInMonth = Enumerable.Range(1, daysInMonth)
                                .Where(d => !weekends.Contains(new DateTime(Convert.ToInt32(year), Convert.ToInt32(month), d).DayOfWeek));
    foreach (var day in businessDaysInMonth)
    {
        workdays++;
    }
    return workdays;
}
```

Obrázek č. 10: Funkce počet pracovních dnů ve zvoleném měsíci
Zdroj: Vlastní

LinqSearch - prostřednictvím této statické třídy jsem se snažil zajistit přístup a výběr k požadovanému tagu v XML souboru, aniž by bylo vyžadováno „všeobjímacího“ modelu.

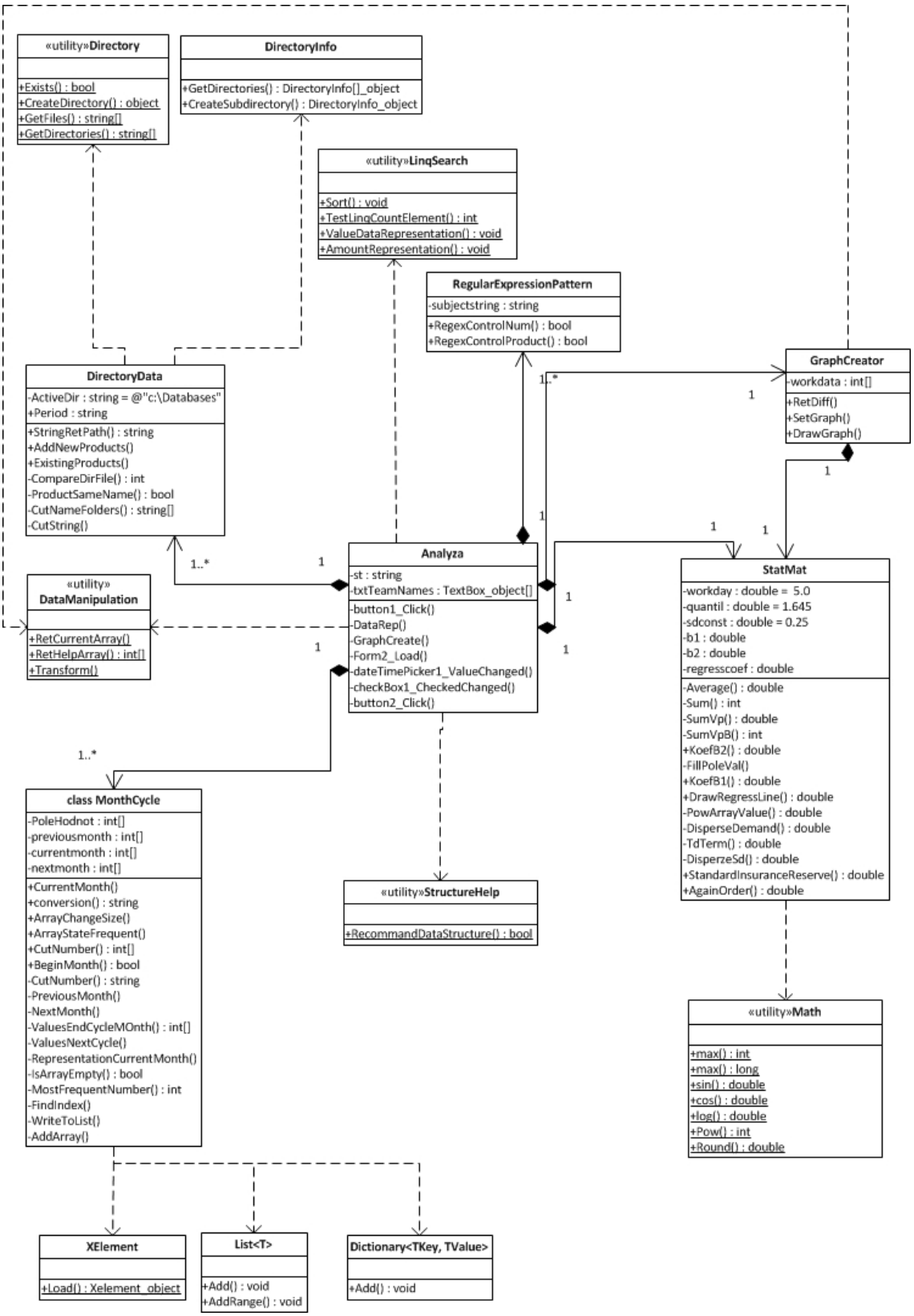
V metodách zakomponovaných v této třídě byly použity techniky pro práci s LINQ, který používá běžné možnosti operací nad různými datovými modely namísto sjednocování různých struktur v těchto modelech.

Jinými slovy při práci s LINQ byly uchovány nesourodě navržené datové struktury a byla vytvořena jednotná syntaxe na dotazování do všech těchto datových typů bez ohledu na jejich fyzickou reprezentaci.

3.1.12 Třída Analyza

V této kapitole bude popsána opět dílčím způsobem třída Analyza. Tato třída je také složena z několika dalších tříd, které řeší dané úkoly, jejichž výstupem je vytvořit graf na základě čerpání množství položek sledovaného výrobku a vyhodnotit, kdy je potřeba položku znovu objednat a jaké je doporučené nejzazší sledované množství výrobku.

Na obrázku č. 11 je taktéž jako to bylo u třídy Databaze demonstrována závislost mezi třídami. Tato akce je opět provedena prostřednictvím UML diagramu.



Obrázek č. 11: UML diagram závislostí třídy Analyza
Zdroj: Vlastní

DirectoryData - viz. předchozí kapitola.

Directory - viz. předchozí kapitola.

DirectoryInfo - viz. předchozí kapitola.

RegularExpressionPattern – viz. předchozí kapitola.

LinqSearch - viz. předchozí kapitola.

MonthCycle - veřejná třída MonthCycle byla vytvořena pro účel jakéhosi nástroje, který data transformuje do příslušných časových cyklů, které jsou základem pro stanovení výše pojistné zásoby.

Tento nástroj monitoruje během sledovaného období předchozí a na základě porovnání dat s předchozím obdobím zařazuje nebo nezařazuje příslušné hodnoty do vytvořených cyklů.

Tato operace je vykonána i s následujícím časovým obdobím. Tím pádem výsledná reprezentace dat je složena z údajů, které jsou v množině časově závislého cyklu.

Rozhodl jsem se, že pro účel této diplomové práce přidám demonstrativní příklad použití a manipulaci se slovníkovou datovou strukturou, kterou jsem implementoval v této třídě. Úkolem následné funkce, která je na obrázku č. 12 je stanovit frekvenci dat v jednotlivých cyklech a na základě tohoto faktu se pak dále rozhodnout, zda množinu dat je vhodné zakomponovat do předchozího cyklu resp. následujícího.

```
private int MostFrequentNumber(int[] arraystate)
{
    int FreqN = 0;
    Dictionary<string, int> freq = new Dictionary<string, int>();
    List<string> freqword = new List<string>();

    string[] result = arraystate.Select(x => x.ToString()).ToArray();

    foreach (string prvek in result)
    {
        if (!freq.Keys.Contains(prvek))
        {
            freq[prvek] = 1;
        }
        else
        {
            ++freq[prvek];
        }
    }
    FreqN = freq.Values.Max();
    foreach (string s in freq.Keys)
    {
        if (freq[s] == FreqN)
        {
            freqword.Add(s);
            FreqN = Convert.ToInt32(s);
        }
    }
    return FreqN;
}
```

Obrázek č. 12: Funkce MostFrequentNumber ve třídě MonthCycle
Zdroj: Vlastní

List<T> - pro transformaci dat jsem využíval generickou třídu List, která umožňuje prvky za běhu programu přidávat a mazat. Jedná se zcela o flexibilní a dynamický nástroj.

Důvodem, proč jsem se rozhodl použít právě tuto třídu a nevybral jsem si jinou třídu podobného charakteru, je rychlost přístupu k prvkům pomocí indexů. Nevýhodou je samozřejmě časová prodleva nutná k vytvoření nového pole a překopírování prvků, i když nastává jen občas.

Dictionary<TKey,TValue> - jedná se o třídu, která reprezentuje datovou strukturu. Já jsem funkcionalitu této třídy implementoval ve třídě MonthCycle.

Jednotlivé klíče třídy Dictionary vrací instanci typu (System.Collections.Generic.Dictionary<TKey,TValue>), který obsahuje všechny klíče ve slovníku. Pořadí resp. priorita klíčů není specifikována, ale jedná se většinou o stejné pořadí jako příslušné hodnoty v kolekci hodnoty vrácené System.Collections.Generic.Dictionary <TKey,TValue>. Values.Property. Pokud je základní slovník upraven, nebo hodnota klíče ve slovníku je upravena, chování klíčového kolekce je nespecifikované.

Xelement - tato třída je alfou a omegou třídy XmlDocument. Tato třída se objevila až v pozdější verzi frameworku, proto by měla disponovat modernějším řešením, a také Je provedena tak, že je vygenerován graf z uložených hodnot, které představují časový cyklus popř. časové cykly čerpání zásoby určené pro statistickou analýzu.

Poslední fází bylo výsledný graf protnout regresní přímkou a předat tak datům konečný formát, proto jsem se ji pokusil vcelku zdařile, implementovat ve třídě MonthCycle.

StructureHelp - statická třída StructureHelp z pracovních dat poskytuje jakési doporučení, zda je používaná struktura vhodná pro statistickou analýzu.

GraphCreator - veřejná třída GraphCreator poskytuje metody pro finální vizuální formát ve formě grafu.

StatMat - mezi základními stavebními kameny aplikace patří veřejná třída StatMat, ve které byly naprogramovány všechny potřebné výpočty, jež měly sloužit k samotné statistické analýzy a k jednotlivým výpočtům, které se zabývají problematikou zásob této diplomové práce.

Math - robustní statická knihovna, která je standardně zahrnuta .NET architektuře jsem využil v knihovně StatMat jako základ k některým statistickým výpočtům.

DataManipulation - tato statická třída má za úkol zpracovávat data z více datových zdrojů, které jsou například reprezentovány kolekcemi, které jsou typu seznamu, slovníku, ale i klasického pole.

Zpracování z více datových zdrojů pak v konečné fázi podléhá a stává se podpůrným nástrojem pro prezentační vrstvu dat, která zajišťuje jakousi vizualizaci výsledků, například prostřednictvím grafu nebo také může sloužit k dalším souvisejícím výpočtům, které jsou realizovány pomocí výše uvedené třídy StatMat.

3.1.13 Návrh grafického uživatelského rozhraní

Rozcestník

GUI bylo navrženo ze zmiňovaných tří formulářových tříd, aby bylo pro uživatele jednoduché a přehledné. Každá z těchto tříd řešila svůj specifický úkol.

V případě, že uživatel spustí aplikaci, otevře se následující formulářové okno s názvem rozcestník. Dole tohoto formulářového okna dle obrázku č. 13 jsou pak vyobrazeny možnosti, ze kterých uživatel vybírá na základě toho, co chce provádět.

V případě že bude zvolena možnost „Přidat nový výrobek resp.“ „množství“, spustí se událost, která aktivuje formulář s názvem Databaze, jehož funkcionalitu popíšu v dalším příspěvku. V případě, že uživatel aplikace zvolí možnost Analyzovat výrobky, pak bude jeho volba přesměrována na formulář Analyza, jehož funkcionalitou se budu opět zabývat samostatně.

Obrázek č. 13: Formulář Rozcestník
Zdroj: Vlastní

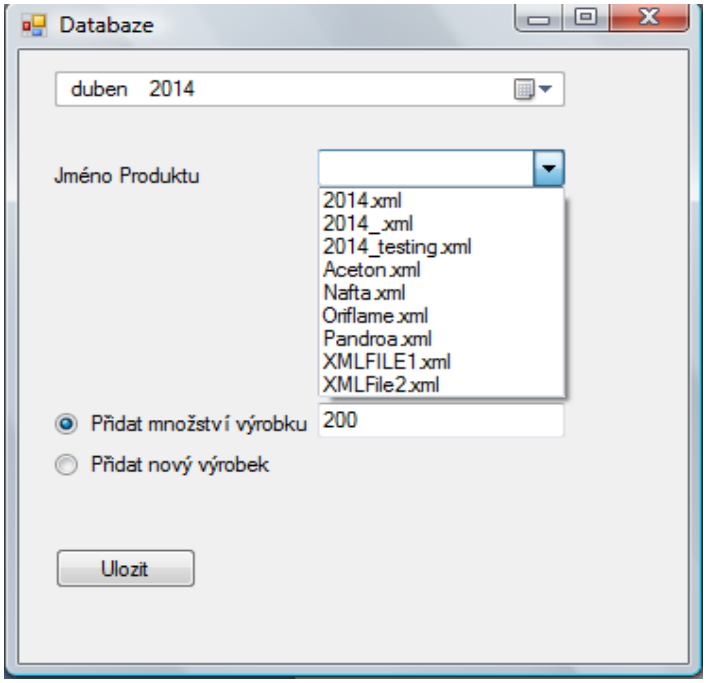
Databaze

Pokud je uživatel přesměřován z Rozcestniku do Databaze, může volit ze dvou povolených akcí.

Když se uživatel rozhodne přidat množství položky, je nutné zaškrtnout tuto volbu a poté vybrat časový okamžik, tedy dobu, jež udává momentální množství a stav vybrané položky.

Dále uživatel zvolí jméno upravovaného výrobku dle obrázku č. 14 a následně tuto volbu potvrdí tlačítkem Uložit.

Po této akci bude informován o stavu úspěšného uložení výrobku. V případě, že uživatel zadává množství výrobku, musí jej definovat číselnou hodnotou, pokud by se rozhodl napsat jiné vstupy než číselné, bude o tomto stavu aplikací informován a zadaný vstup tedy nebude akceptován.

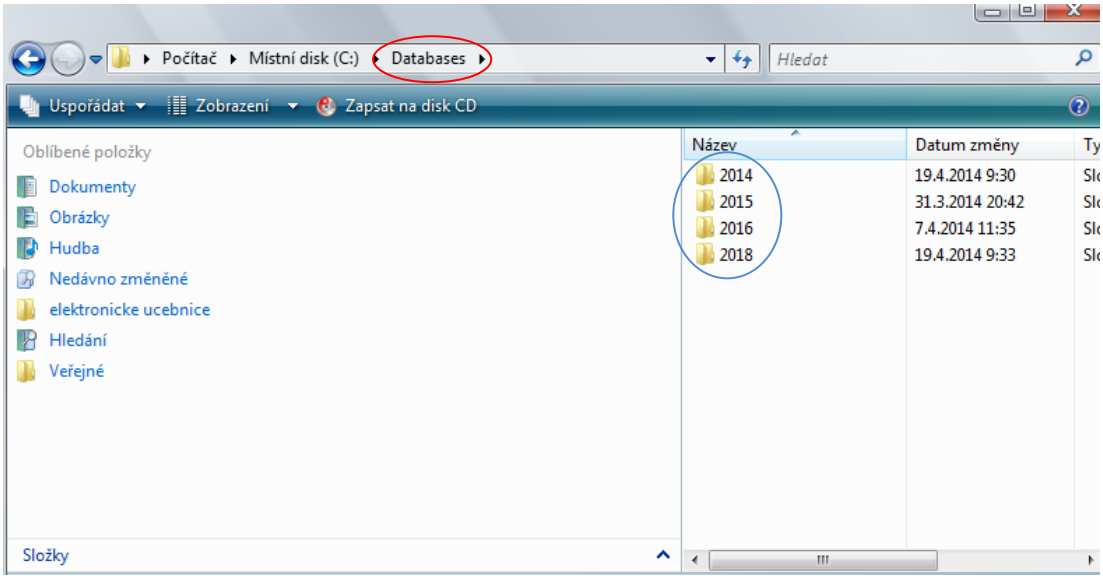


Obrázek č. 14: Formulář Databaze
Zdroj: Vlastní

V situaci, kdy se rozhodl uživatel vybrat možnost přidat výrobek, bude vyzván opět k tomu, aby definoval časové období, do kterého má analyzovaná položka spadat. Pokud je tato akce potvrzena a uživatel vybral období, tak se vytvoří příslušný adresář, do kterého se následně položka uloží.

V okamžiku, kdy již adresář časového období existuje, je vytvořen pouze XML soubor v příslušném adresáři. Uspořádání adresářů je možné si představit jako na obrázku č. 15 (modré zvýraznění).

V každé složce se nacházejí příslušné položky dle stanoveného roku. Na obrázku je možné si tak všimnout, že i umístění složek souhlasí již s deklarovanou cestou „@"c:\Databases"“ (označeno červeným zvýrazněním).



Obrázek č. 15: Přidání nového výrobku
Zdroj: Vlastní

Analyza

V situaci, kdy se uživatel rozhodne analyzovat data, bude jeho aktivita přesměrována na formulář analýza.

V první fázi je uživatel vyzván k tomu, aby zadal opět časovou veličinu, která je základem k tomu, aby byla vybrána správná položka ze složky zvoleného časového období.

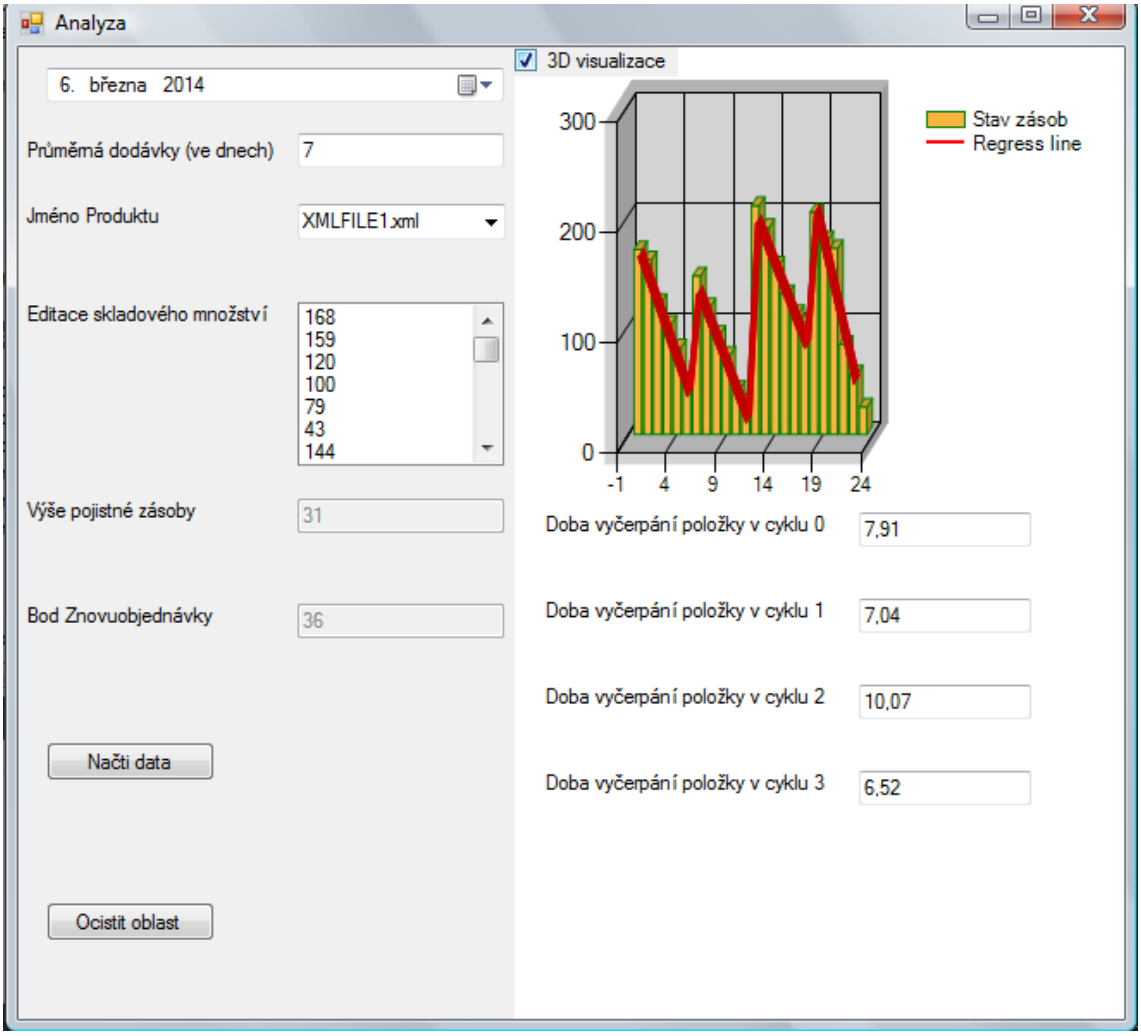
Poté je potřeba doplnit průměrnou dodávku, která se stanoví dle smluvních vztahů mezi dodavatelem a příslušnou společností, která produkt odebírá; až je tento úkon splněn uživatel musí zvolit výrobek, který chce v daném časovém horizontu analyzovat.

Následně je uživatel intuitivně naveden k tomu, aby stiskl tlačítko „Načti data“ a tato událost se pak vygeneruje na základě uložených dat ze zvoleného XML souboru

graf, který demonstruje cyklus resp. cykly na základě načtených dat. Možností grafu je také i 3D vizualizace. K tomuto grafu je také vygenerována informace o době vyčerpání položky v závislosti na vyhodnocení počtu cyklů zkoumaného produktu.

S vygenerovaným grafem a dobou vyčerpání zásoby, jsou také vypočteny sledované hodnoty a to výše pojistné zásoby a bod, který indikuje, kdy má být provedena znovu objednávka.

Pokud uživatel zamýšlí provést další analýzu, zvolí tlačítko „Očistit oblast“ a postupuje v analýze stejným způsobem, jako to již bylo popsáno.



Obrázek č. 16: Formulář Analyza
Zdroj: Vlastní

3.1.14 Přínosy

Užitečnost mé aplikace zatím nelze přesně kvantitativně vyčíslit.

Co lze však vyčíslit je, že tento vývoj nebude stát žádné finanční prostředky. Pokud by tento projekt společnost zadala profesionálnímu programátorovi, mohla by při nynějších mzdových tarifech, které činí cca 200 Kč/hod a přibližně deseti dnech práce (pokud by programátor pracoval 8 hodin), zaplatit kolem 16 až 20 tis. plus další náklady v případě, že by chtěla aplikaci upgradovat nebo by požadovala rozšíření její funkcionality.

Pokud se osvědčí nasazení této aplikace, může společnost ušetřit náklady za skladování nebo také může své vázané aktiva mít delší dobu ve formě cash - flow.

Majitel předpokládá nasazení technologie, jejího užívání a rozšiřování jejich funkcionalit dle potřeb podniku.

3.1.15 Shrnutí

V této kapitole byla popsána a komplexně demonstrována filosofie, návrh, funkcionalita a metody, které jsem aplikoval pro vytvoření programu.

Tento popis mně pomohl vytvořit a více seznámit s charakterem aplikace StoreStatistics.

Nejdříve jsem se rozhodl udělat analýzu tříd, abych dostal komplexní pohled na to, jakou cestou se mám vydat při vytváření aplikace. Tato analýza byla provedena pomocí UML diagramů. Poté jsem se zaměřil na samotnou implementaci aplikace prostřednictvím jazyka C# a jeho technologií, které využívá, aby vývojářům zjednodušil práci a vytvářel dynamické a flexibilní aplikace.

Je důležité zdůraznit, že se tedy jednalo o program řízení zásob, jehož účelem bylo také stát se pro uživatele flexibilním, jednoduchým, přístupným a levným řešením, což se podle mého názoru povedlo.

Závěr

Hlavním cílem této diplomové práce bylo využití vybraných statistických metod k řízení zásob, které jsou chemického charakteru. Využívané statistické techniky, zejména regresní analýza, byly aplikovány na vybrané položky, které tvořily provozní zásoby v modelu nezávislé poptávky, což umožní vedení této malé společnosti monitorovat jejich stav.

Analýzou poptávky obou sledovaných produktů (nafta, subtotal), které byly předmětem zpracování v praktické části, bylo zjištěno, že se během sledovaného období nijak jejich charakter zásadně nezměnil. Dospěl jsem k závěru, že trend zjištěné poptávky nevykazoval změnu charakteristiky ve svém vývoji. Ze zjištěných faktů lze tedy vyvodit závěr, že ve zkoumaném období výrazně nestoupla nebo nepoklesla potřeba po produktech vybraného charakteru.

Při zpracování analýzy obou produktů jsem zjistil, že ve vybrané společnosti není žádný systém objednávání zásob a objednávky bývají podány při různých hladinách předmětných položek. Tento proces je tedy otázkou znalosti provozu a zkušenosti personálu.

Cílem této práce tedy bylo navrhnout autonomní aplikaci, která neměla být určena pro on-line používání, ale její účel byl pouze pro analýzu stavu zásob za vybrané období. Aplikace ze zadaných dat identifikuje pokles stavu zásob pod bod objednávky a zjištění výše pojistné zásoby sledované položky.

Při implementaci aplikace bylo také provedeno zaškolení uživatelů. Povedlo se rovněž splnit požadavek na jednoduchost a přehlednost aplikace a také se ušetřily nemalé finanční náklady za její realizace. Jak přesně vysoké náklady společnost ušetří, bude zjištěno za delší časové období, tzn. až budou informace, které aplikace bude poskytovat, plně využity a pochopeny a výsledek bude mít vypovídající hodnoty o jeho úspěšnosti.

Seznam použité literatury

1) KROPÁČ, Jiří. *Statistika B*. 2. dopl. Brno: Vysoké učení technické v Brně, Fakulta podnikatelská, 2009, 151 s. ISBN 978-80-214-3295-6.

2) KROPÁČ, Jiří. *Statistika C*. 1. vyd. Brno: Vysoké učení technické v Brně, Fakulta podnikatelská, 2008, 151 s. ISBN 978-80-214-3591-9.

3) HINDLS, R. *Statistika pro ekonomy*. 8. vyd. Praha: Professional Publishing, 2007, 415 s. ISBN 978-80-86946-43-6.

4) SHARP, John. *Visual C#: krok za krokem*. Brno: Computer Press, 2010, 696 s. ISBN 978-80-251-3147-3.

5) NASH, Trey. *C# 2010: Rychlý průvodce novinkami a nejlepšími postupy*. Brno: Computer Press, 2010, 624 s. ISBN 978-80-251-3034-6.

6) *Objektově orientované programování* [online]. Brno, 2011 [cit. 2013-12-01]. Dostupné z: https://www.vutbr.cz/www_base/priloha.php?dpid=68410. Skripta. VUT.

7) C# Rolling file logger. In: *C# Rolling file logger* [online]. [cit. 2013-12-01]. Dostupné z: http://haykmanucharyan.net/Default.aspx?mode=2&id=21 Architektura .NET. In: *J2EE, .NET a vývoj rozsáhlých systémů 2.* |

8) *Interval.cz* [online]. [cit. 2013-12-01]. Dostupné z: http://interval.cz/clanky/j2ee-net-a-vyvoj-rozsahlych-systemu-2/

9) Architektura .NET frameworku | Interval.cz. In: *Architektura .NET frameworku | Interval.cz* [online]. [cit. 2013-12-01]. Dostupné z: http://interval.cz/clanky/architektura-net-frameworku/

Seznam obrázků, tabulek a grafů

Seznam obrázků

Seznam tabulek

Seznam grafů

Seznam příloh

Příloha č. 1: CD s aplikací StoreStatistics