MORAVSKÁ VYSOKÁ ŠKOLA OLOMOUC, o.p.s.

Martin Janský

Optimalizace procesů logistiky pomocí Visual Basic Applications

Optimizing logistics processes using Visual Basic for Applications

Bakalářská práce

Vedoucí práce: PhDr. Jan Lavrinčík, Ph.D.

Olomouc 2016

Čestné prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a použil jen uvedené informační zdroje.

V Olomouci 24. 3. 2016 …….. Martin Janský

Poděkování

Chtěl bych poděkovat PhDr. Janu Lavrinčíkovi, Ph.D. za vedení mé bakalářské práce, cenné rady a odborný dohled. Děkuji také Janě Janské za pomoc při gramatické kontrole práce.

Moravská vysoká škola Olomouc
Akademický rok: **2014/2015**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Martin Janský**
Osobní číslo:	**M13104**
Studijní program:	**B6208 Ekonomika a management**
Studijní obor:	**Podnikové informační systémy**
Název tématu:	**Optimalizace procesů logistiky pomocí Visual Basic Applications**
Téma anglicky:	**Optimizing logistics processes using Visual Basic for Applications**
Zadávající katedra:	**Ústav informatiky a aplikované matematiky**

Zásady pro vypracování:

- využívat aktuální informační zdroje (tištěné publikace, odborné časopisy, internet, aj.),
- dodržovat citační normu (podrobnosti na www.boldis.cz),
- sestavit si časový harmonogram vypracování závěrečné práce,
- pravidelně konzultovat (e-mail, telefon, osobní),
- u částí diplomové práce psaných v anglickém jazyce nechat provést korekturu.

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

LAURENČÍK, Marek a Michal BUREŠ. Programování v Excelu 2010: záznam, úprava a programování maker. 1. vyd. Praha: Grada, 2013, 198 s. Průvodce (Grada). ISBN 978-80-247-5033-0.
SHEPHERD, Richard a Michal BUREŠ. Access VBA: výukový průvodce. 1. vyd. Brno: Computer Press, 2012, 397 s. Průvodce (Grada). ISBN 978-80-251-3686-7.
KRÁL, Martin a Michal BUREŠ. Access VBA: výukový kurz. Vyd. 1. Brno: Computer Press, 2010, 504 s. Průvodce (Grada). ISBN 978-80-251-2358-4.
MILDEOVÁ, Stanislava a Michal BUREŠ. Manažer jako inteligentní uživatel I.: tabulkové aplikace. Vyd. 1. Praha: Oeconomica, 2002, 82 s. Průvodce (Grada). ISBN 80-245-0458-8.
WALKENBACH, John a Michal BUREŠ. Microsoft Excel 2000: programování ve VBA. Vyd. 1. Praha: Computer Press, 2002, 82 s. Průvodce (Grada). ISBN 80-722-6250-5.

Vedoucí bakalářské práce: **PhDr. Jan LAVRINČÍK, Ph.D.**
Ústav informatiky a aplikované matematiky

Datum zadání bakalářské práce: **10. února 2015**
Termín odevzdání bakalářské práce: **31. března 2016**

Podpis studenta: ..... Datum: 14.4.2015

Podpis vedoucího práce: ..... Datum: 14.4.2015

Mgr. Irena KOVAČIČINOVÁ
prorektorka

L.S.

Mgr. Veronika ŘÍHOVÁ, Ph.D.
manažer ústavu

V Olomouci dne 14. dubna 2015

Obsah

Úvod

Předkládaná bakalářská práce nese název Optimalizace logistických procesů pomocí Visual Basic for Application. Jak již samotný název napovídá, v práci se budeme zabývat minulostí, současností a vývojem logistiky.

V dnešní době logistika zastává důležité místo v managementu a řízení výrobních i nevýrobních podniků. Je důležitá pro maximální vytíženost dopravy při přepravě zboží, pro úsporné a nenákladné skladování zboží, skladování materiálů a pro ekonomiku všeobecně. S rostoucím vývojem společnosti rostou i požadavky a nároky zákazníků a tomu musí odpovídat i rychlost a kvalita jejich uspokojení. Logistika je jedním z mnoha kroků, kterými se musí řídit moderní hospodářství při cestě od výrobce ke spokojenému zákazníkovi.

Hlavní cíl bakalářské práce spočívá v navržení programu v aplikaci Visual Basic for Application na ovládání logistických procesů příjmu zboží. Vytvořený program dle návrhu optimalizuje procesy logistiky, zejména ty procesy, které se vztahují ke skladovému hospodářství a příjmu zboží. Abychom mohli lépe naplnit hlavní cíl práce, stanovili jsme si následující dílčí cíle. Prvním dílčím cílem bude systematicky popsat logistiku, její vývoj, a zejména se budeme soustředit na popis zásob. Druhý dílčí cíl bakalářské práce je analýza a možnosti vývojového prostředí Visual Basic for Application.

Teoretická část

1 Logistika

Logistika jako vědní disciplína je velmi mladá. Pro podniky je logistika velmi důležitá, protože se snaží využít v co největší míře všechny kapacity podniku a to zejména v oblasti zásobování a skladování. Logistika se neuplatňuje pouze ve sféře podniků a organizací, ale třeba i ve státních institucích (například ve školách, v nemocnicích). Logistika je nejen o přepravě a transportech výrobků, materiálů a zboží, ale logistika se snaží optimalizovat také procesy vedení zásob, toků zboží a tím snížit náklady spojené se skladovým hospodářstvím. Důležitým cílem logistiky je zajištění správné distribuce tak, aby materiály nebo zboží bylo vždy včas u odběratele, zákazníka a tím uspokojilo jejich požadavky. Logistika je v dnešní době již velice rozsáhlá disciplína, a proto je důležité se podívat na její historii a vznik. Definice logistiky jsou shrnuty v kapitole 1.1.

1.1 Definice logistiky

„*Logistika jako druh činnosti je doslova tisíce let stará, neboť její vznik můžeme spojovat již s nejranějšími formami organizovaného obchodu*.“[1]

„*Primát praktického uplatnění logistiky v hospodářské praxi patří Spojeným státům americkým. Vše se odvíjelo, ať to bylo v oblasti vojenské nebo hospodářské, od nutnosti překonat velké vzdálenosti.*“[2]

[1] LAMBERT, Douglas M, James R STOCK a Lisa M ELLRAM. *Logistika*. Vyd. 2. Praha: Computer Press, 2000, s. 5.

[2] SIXTA, Josef a Miroslav ŽIŽKA. *Logistika: metody používané pro řešení logistických projektů*. Vyd. 1. Brno: Computer Press, 2009, s. 14.

Jako definici logistiky jsem si vybral nejrozsáhlejší definici, v níž se autor snažil zahrnout velký záběr z tohoto pojmu. „*Logistika je řízení materiálového, informačního i finančního toku s ohledem na včasné splnění požadavků finálního zákazníka a s ohledem na nutnou tvorbu zisku v celém toku materiálu. Při plnění potřeb finálního zákazníka napomáhá již při vývoji výrobku, výběru vhodného dodavatele, odpovídajícím způsobem řízení vlastní realizace potřeby zákazníka (při výrobě výrobku), vhodným přemístěním požadovaného výrobku k zákazníkovi a v neposlední řadě i zajištěním likvidace morálně i fyzicky zastaralého výrobku.*“[3]

Pro celkový přehled a pochopení významu logistiky z definic je neméně důležité se také ohlédnout za vývojem logistiky.

Logistice se začíná dostávat mnohem větší pozornosti především po skončení druhé světové války, protože efektivnějšímu používání logistických operací spojených vojsk se připisoval velký podíl na vítězství.[4] Úkolem logistiky ve vojenských operacích je hlavně získávání, skladování, distribuování, rozmístění, přepravování vojenského materiálu a osob, zabezpečení lékařských a zdravotnických služeb.[5]

Pro naši bakalářskou práci je ovšem důležitější pohled na vývoj logistiky hospodářské než vojenské, proto se v kapitole 1.2 budeme zabývat vývojem hospodářské logistiky.

[3] SIXTA, Josef a Václav MAČÁT. *Logistika: teorie a praxe*. Vyd. 1. Brno: CP Books, 2005, s. 25.
[4] LAMBERT, Douglas M, James R STOCK a Lisa M ELLRAM. *Logistika*. Vyd. 2. Praha: Computer Press, 2000, s. 5.
[5] PERNICA, Petr. *Logistika pro 21. století: (Supply chain management).* Vyd. 1. Praha: Radix, 2005, s. 21.

1.2 Vývoj hospodářské logistiky

Vývoj hospodářské logistiky můžeme, také brát jako vývoj novodobý, protože začal až v půlce minulého století. Vývoj hospodářské logistiky jsem shrnul do pěti období pomocí poznatků autorů A. Stehlík a J. Kapoun z knihy *Logistika pro manažery*.[6]

Od roku 1950

Logistické myšlení, praxe a technologie jsou přebírány z vojenské sféry do civilní sféry. Ke změnám dochází především díky pokrokům vědy a techniky.

1955-1970

V tomto období dochází k formování logistické teorie a praxe. Rozšiřuje se také díky možnosti využití letecké přepravy. Objevuje se pojem „total-costs“[7].

K rychlejší implementaci logistiky pomohl:

- Vývoj a využití elektronického zpracování dat;
- Expanze technik marketingu;
- Tlak na logistické náklady a výdaje;
- Systémové teorie a teorie řízení;
- Intenzifikace konkurence;
- Růst distribučních nákladů – celkové náklady a zisky;
- Rozšíření sortimentu výrobků;
- Rozvoj v dopravě;
- Výzkum a výuka v oblasti distribuce.

[6] STEHLÍK, Antonín a Josef KAPOUN. *Logistika pro manažery*. Vyd. 1. Praha: Ekopress, 2008, s. 17-18.

[7] *„Celkové náklady, v mikroekonomické teorii zkráceně označované jako TC z anglického total costs, dostaneme jako součet fixních (*FC*) a variabilních (*VC*) nákladů. Z výše uvedeného vyplývá, že se celkové náklady se změnou velikosti produkce mění (*obsahují v sobě VC*). Při růstu výroby a tomu odpovídajících variabilních nákladů rostou i celkové náklady. Vlivem existence fixních nákladů však průměrné náklady na jeden výrobek klesají a naopak.“*

Zdroj: Celkové náklady. *Firemní slovník.cz* [online]. 2016 [cit. 2016-01-30]. Dostupné z: http://www.firemnislovnik.cz/c/celkove-naklady/

1970-1985

Dochází k úspěšnému rozvoji americké logistiky v Evropě. Velký důraz je kladem na fyzickou stránku objektů. Součástí distribučních systémů musí být i systémy informační.

1985-1995

Prosazuje se systém integrované logistiky, která je efektivnější. Uspokojení potřeb zákazníka při ekonomických pohledech na celkovou činnost firmy se klade na první místo.

Od roku 1995

Dochází k velkému rozvoji elektroniky a internetových technologií, které umožňují vytvoření velkých sítí. Řídí je koordinační SCM, tak aby účinnost logistiky byla optimální. (SMC -Supply Chain Management *„je oblast řízení, která zahrnuje všechny procesy komunikace s dodavateli v celém dodavatelském řetězci. SCM zahrnuje také jejich vzájemnou koordinaci, slaďování a řízení. SCM se obvykle vztahuje k výrobnímu sektoru pro řízení dodavatelů surovin nebo jiných subdodávek (výrobky). SCM zahrnuje procesy a oblasti: 1)Plánování, 2)Logistika, doprava a distribuce, 3) Skladování, 4) Výroba"*[8]).

Převážně díky novým dopravním možnostem nebylo překonávání přepravy na velké vzdálenosti již tak obtížným úkolem. Teorie a praxe logistiky se rozšířila ze severní Ameriky na další kontinenty, což vedlo nezbytně k dalšímu rozvoji. Nejdůležitější směry, které působily na logistiku a ovlivňují i logistiku dnešní doby jsou uvedeny v kapitole 1.3.

[8] SCM (Supply Chain Management). Managementmania [online]. 2013, 22.10.2015 [cit. 2015-11-29]. Dostupné z: https://managementmania.com/cs/supply-chain-management

1.3 Přehled megatrendů vývoje logistiky

K pochopení logistiky a její důležitosti v současné době nám pomůže objasnění hlavních směrů vývoje tzv. megatrendů, kterými se v 21. století logistika ubírá, jak shrnul P. Pernica ve své publikaci *Logistika pro 21. století*.[9]

Převaha tržního hospodářství a západního způsobu života

- Přechod od trhu prodávajícího k trhu kupujícího;
- Zvětšování sortimentu výrobků;
- Zkracování životního cyklu výrobků;
- Růst komplexnosti výrobků;
- Zkracování termínů dodání.

Globalizace

- Internacionalizace;
- Ekologizace;
- Deregulace;
- Standardizace.

Technická revoluce

- Rozvoj dopravy;
- Rozvoj telekomunikací;
- Informatizace.

Pokud chceme shrnout dosavadní vývoj logistiky, tak je dle mého názoru zásadně ovlivněn informačními a komunikačními technologiemi. Informační a komunikační

[9] PERNICA, Petr. *Logistika pro 21. století: (Supply chain management).* Vyd. 1. Praha: Radix, 2005, s. 58.

technologie měly nemalý vliv nejen na logistiku, ale samozřejmě na vývoj celého světa. Troufám si říct, že informační systémy jsou již nezbytnou součástí všech logistických procesů. Informační a komunikační technologie jsou také hlavním faktorem růstu a rozvoje logistiky.

Jak už vyplývá z uvedené historie a vývoje logistiky, tak cíle logistiky se také postupně měnily. Jedním ze základních cílů logistiky je uspokojování potřeb zákazníků, protože zákazník je nejdůležitějším článkem celého řetězce. Od zákazníka vychází poptávka a u zákazníka končí celý řetězec logistiky. Cíle logistiky mohou být různě zaměřeny např. na výkonové, ekonomické, vnější, ale pro mou práci jsou nejdůležitější cíle vnitřní. Vnitřní cíle se totiž orientují na snižování nákladů a to například na zásoby, na dopravu, na manipulaci a skladování, na výrobu, na řízení a mnoho dalších.[10] V následujících bodech bakalářské práce se proto budeme věnovat problematice teorii procesů vedení zásob, skladování a manipulaci se zbožím, skladovému hospodářství (Obrázek 1 – první tři činnosti jsou barevně odlišeny, protože se budeme zabývat právě optimalizaci těchto bodů.).

Obrázek 1: Systém skladovacích a komisionářských činností.[11]

1.4 Zásoby

Zásoby jsou největší investicí firmy. Pokud se zásobami budeme nakládat efektivně, můžeme docílit zlepšení cash-flow[12] podniku i lepší návratnosti investic. Zásoby slouží

[10] SIXTA, Josef a Miroslav ŽIŽKA. *Logistika: metody používané pro řešení logistických projektů*. Vyd. 1. Brno: Computer Press, 2009, s. 19-20.

[11] Obrázek 1: Systém skladovacích a komisionářských činností – vlastní tvorba. Zdroj: SCHULTE, Christof. *Logistika.* Vyd. 1. Praha: Victoria Publishing, 1994, s. 91.

v rámci podniku k dosáhnutí co největších úspor, vyrovnání poptávky a nabídky, umožnění specializace výroby, zásoby poskytují ochranu před výkyvy v poptávce a poskytují ztlumení mezi spoji v distribuci. [13]

1.4.1 Klasifikace zásob

Klasifikaci zásob lze dělit dle mnoha kritérií například dle účetních přepisů, funkčního hlediska, hlediska použitelnosti, nebo podle stupně zpracování. Při optimalizaci stavu zásob vycházíme z funkční klasifikace zásob, která se dělí na běžnou zásobu, pojistnou, pro předzásobení, vyrovnávací, strategickou, spekulativní a technologickou.[14]

1.4.2 Řízení zásob

Věnovat se všem skladovým položkám, které tvoří zásobu v podniku, není vždy možné, protože se jedná většinou o počty v řádech tisíců. Je potřeba si stanovit skupiny, do kterých je možné artikly členit. Takové skupiny lze pak lépe analyzovat. Jednou z takových analýz může být například analýza ABC, která vychází z Paretova pravidla (80 % důsledků vyplývá z 20 % počtu možných příčin. Malá část z položek znamená většinu hodnoty spotřeby). Při použití analýzy ABC se vychází ze sestavených položek setříděných zásob dle sledovaného znaku, kterým může být například hodnota spotřeby v určité délce období.[15]

Pro řízení zásob ve firmách v 21. století určitě nestačí pouze použití ABC analýzy, proto existují sofistikovanější systémy a modely, které jsou pomocí výpočetní techniky

---

[12] *„Cash flow (CF), česky peněžní tok, je rozdíl mezi peněžními příjmy a peněžními výdaji za sledované období, ve výkazu CF jsou uvedeny tedy skutečné hotovostní toky.“*
Zdroj: Peněžní tok. In: *Management mania* [online]. 2015 [cit. 2016-02-14]. Dostupné z: https://managementmania.com/cs/penezni-tok

[13] LAMBERT, Douglas M, James R STOCK a Lisa M ELLRAM. *Logistika*. Vyd. 2. Praha: Computer Press, 2000, s. 112.

[14] SIXTA, Josef a Miroslav ŽIŽKA. *Logistika: metody používané pro řešení logistických projektů*. Vyd. 1. Brno: Computer Press, 2009, s. 62-63.

[15] SIXTA, Josef a Miroslav ŽIŽKA. *Logistika: metody používané pro řešení logistických projektů*. Vyd. 1. Brno: Computer Press, 2009, s. 66-67.

velice mocným nástrojem. Za zmínku stojí třeba Q-systém a P-systém řízení zásob a modely řízení zásob, které se dělí podle spotřeby a pořizovací doby (deterministické, stochastické, nedeterministické modely). Ve své bakalářské práci se věnuji problému skladování zásob a typům skladovacích prostor, kde se materiály /palety/ zboží uskladňují a tvoří tak jmenované zásoby podniku.

1.4.3 Skladování zásob

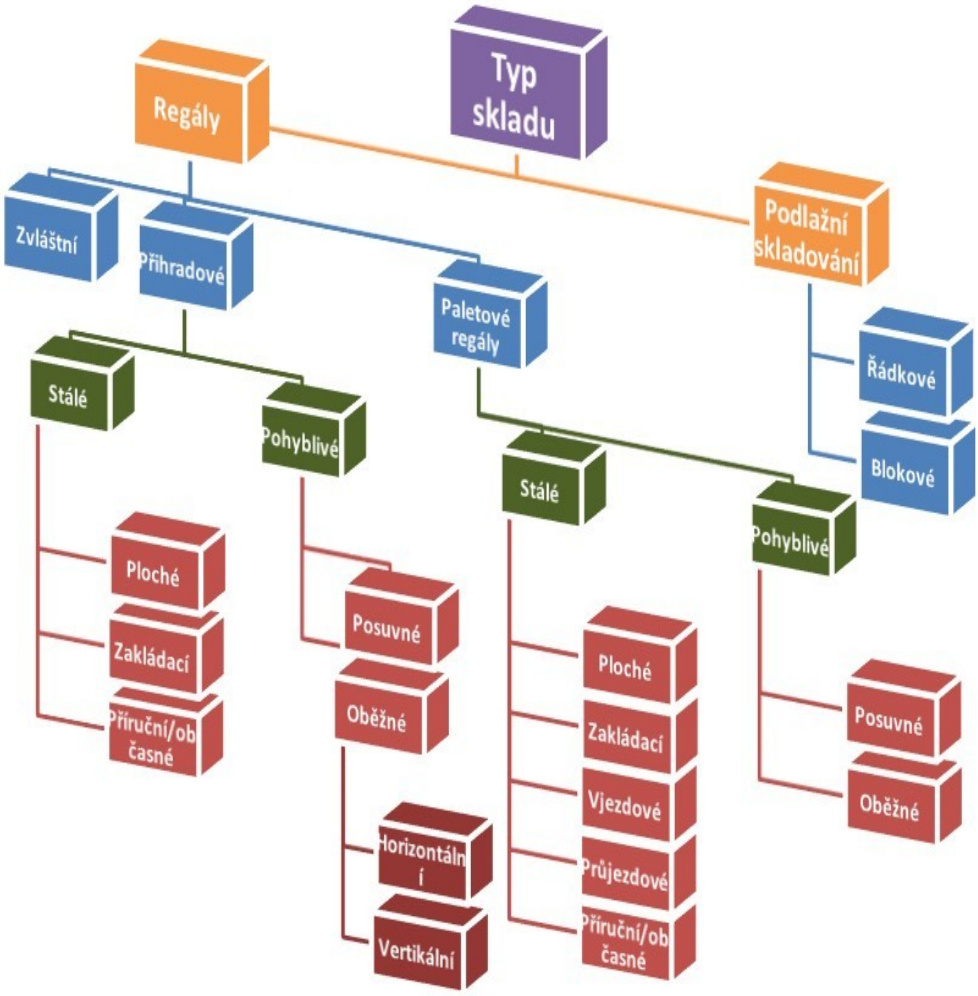
„Pokud chce podnik využít skladování jako prostředku k získání konkurenční výhody, pak musí tradiční pohled na skladování, podle kterého jde o pouhé uskladňování a řízení zásob, nahradit novým paradigmatem, které kromě zásob zahrnuje i veškeré související informační toky. V tomto novém pojetí skladování se základními předpoklady logistického úspěchu stávají: nasazení počítačové technologie, informace a automatizace.“[16]

1.4.4 Typy skladů

Přehled užívaných typů skladů, uskladnění a skladových míst zobrazuje Obrázek 2.

[16] LAMBERT, Douglas M, James R STOCK a Lisa M ELLRAM. *Logistika*. Vyd. 2. Praha: Computer Press, 2000, s. 310.

Obrázek 2: Typová struktura skladů.[17]



Blokové a řádkové sklady

Blokové skladování zboží se uskutečňuje na podlaze v blocích. Pokud je zboží uskladňováno na podlahu v řádkové formě, tak se jedná o řádkové skladování. U zboží,

[17] Obrázek 2: Typová struktura skladů - vlastní tvorba. Zdroj: SCHULTE, Christof. *Logistika.* Vyd. 1. Praha: Victoria Publishing, 1994, s. 93.

které je dostatečně pevné a necitlivé na tlak se může využít skladování stohové, kde se zboží stohuje do určité výšky. Výška stohování je závislá na výšce samotného skladovacího prostoru, nosnosti skladových jednotek a nosnosti podlaží. Blokové skladování je vhodné tam, kde se jedná o malý rozsah sortimentu a jde většinou o jedno-druhové zboží. Musí se počítat s tím, že bude zajištěn přístup pouze k horním skladovým jednotkám a jednotkám, umístěným v přední řadě blokového skladu. Mezi výhody blokového skladování můžeme zařadit hlavně menší investiční náklady a flexibilitu. Nevýhodou je prakticky nemožnost mechanizace a automatizace, velký důraz je kladen na uspořádané obsazování blokových míst. Stohovací skladování bez přeskládání může porušovat FIFO[18].[19]

Regálové sklady

Skladování ve skladech s regály se provádí na více rovinách nad sebou na uzavřených podlažích nebo nositelích uložení (nosnících, kde se ukládají palety). Výhodou oproti blokovému skladování je přímý přístup ke každé skladové jednotce, nebo sortimentu. Možnost dosažení vysoké obrátkovosti, možnost automatizace. Dobré možnosti komisionářství a kontroly stavu zásob. Mezi nevýhody můžeme počítat vyšší pracovní náklady, vyšší pořizovací investiční náklady a dle volby dopravní techniky velikost plochy nezbytná pro využívání dopravní techniky.[20]

[18] FIFO (First-in/ first-out) – nejdříve je obsloužen nejstarší požadavek. (například přijaté palety jsou vyskladněny dle pořadí příjmu). Mezi další typy přechodů patří LIFO (Last-in / first-out) – nejdříve je obsloužen nejnovější požadavek (například odebrání palet u stohového skladování, kde je odebrána horní paleta tzn. ta paleta, která byla uložena jako poslední). Dalším druhem přechodu fronty je PRI (priority) – nejdříve je obsloužena skladová jednotka s nejvyšší důležitostí (například zboží, které je akční). SIRO (selection in random order) – zde se jedná o náhodné pořadí. FEFO (First-expired / first-out) – znamená, že první je na řadě ten s nejbližším datem spotřeby.

Zdroj: SIXTA, Josef a Miroslav ŽIŽKA. *Logistika: metody používané pro řešení logistických projektů.* Vyd. 1. Brno: Computer Press, 2009, s. 118.

[19] SCHULTE, Christof. *Logistika.* Vyd. 1. Praha: Victoria Publishing, 1994, s. 94-95.

[20] SCHULTE, Christof. *Logistika.* Vyd. 1. Praha: Victoria Publishing, 1994, s. 96-98.

1.4.5 Příjem zboží

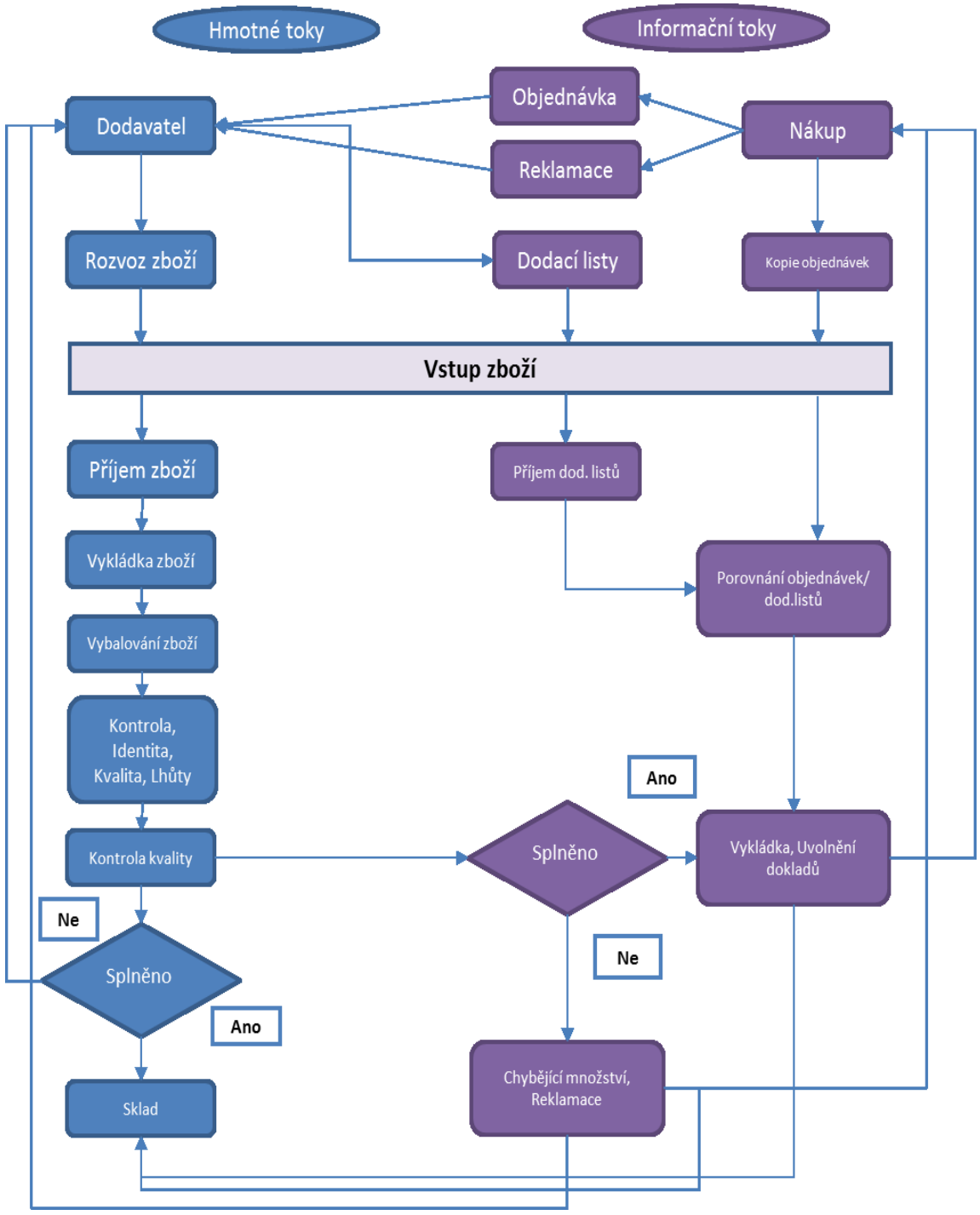
K zásobování a skladování patří nezbytně proces příjmu zboží. Při příjmu zboží je nutné provádět jednotlivé dílčí činnosti, které Christof Schulte shrnul v kapitole Hmotné a informační toky při příjmu zboží[21]. Pro lepší představu zachycuje tyto činnosti obrázek 3.

- Příjem dodávek materiálů a zboží;
- Prověření dodacích listů a objednávek, kontrola správnosti zboží a množství;
- Přesný odpočet (případně vážení, měření) a důraz na kontrolu množství;
- Zásilku je možno uvolnit k převzetí a deklaraci. Materiál/paletu je třeba označit a tím připravit pro další přepravu v rámci skladu.

Pro zajištění krátké doby tranzitu na úseku příjmu zboží je nutné počítat s celou řadou vlivů. Časové rozložení přijímaných transportů nám určuje potřebný počet personálu a dopravní techniky k obsluze. Přihlížíme na velikost a uspořádání skladů, skladovacích ploch a kancelářských ploch a ty jsou závislé na druzích přijímaných materiálů (například stav, tvar, velikost, rozměry, citlivost a vybalování materiálu). Dalším faktorem, ke kterému musíme přihlížet, je technické vybavení skladu. V neposlední řadě je důležitá organizace celého procesu (například vymezení postupů, pravidla informačních a dokladových toků, vymezení úkolů).[22]

[21] SCHULTE, Christof. *Logistika.* Vyd. 1. Praha: Victoria Publishing, 1994, s. 56-58.
[22] SCHULTE, Christof. *Logistika.* Vyd. 1. Praha: Victoria Publishing, 1994, s. 58-59.

Obrázek 3 : Hmotné a informační toky na příjmu zboží.[23]



[23] Obrázek 3: Hmotné a informační toky na příjmu zboží - vlastní tvorba. Zdroj: SCHULTE, Christof. *Logistika*. Vyd. 1. Praha: Victoria Publishing, 1994, s. 57.

2 VBA – Visual Basic for Application

Pro optimalizaci procesů logistiky jsem si zvolil program/jazyk Visual Basic for Application (VBA). VBA je součástí Microsoft Office balíčku, tudíž je také nedílnou součástí podnikových informačních systémů.[24] Microsoft Office je ve firmách využívám pro různé exporty a vyhodnocení (grafy, tabulky) z podnikových SCM a EDI[25] systémů.

2.1 Rodina Visual Basic

Takzvaná rodina Visual Basic je tvořena jazykem Visual Basic (VB), Visual Basic for Application (VBA) a VBScriptem, každý z nich má určitou roli pro vývoj programů a aplikací. Visual Basic je nadřízený systém celé skupiny a jako samostatný programovací produkt je nabízen v programu Microsoft Visual Studio. Úlohou studia je poskytnutí vývojářům software, pomocí kterého mohou vytvářet samostatné komponenty a aplikace. Visual studio se skládá z grafického vývojového prostředí, díky kterému vývojář může navrhovat uživatelské prostředí. Pomocí stejného programu může programátor také svůj program kompilovat, testovat a ladit. [26]

VBA a VB mají společný základ a jádro Visual Basicu je již součástí instalace Microsoft Office a jejich vzájemná komunikace probíhá přes formuláře a moduly. Formuláře vytvořené ve VBA se mohou lišit od klasického formuláře vytvořeného pomocí jazyka VB svou omezeností, ale pro většinu vytvářených aplikací ve VBA je to plně dostačující. Visual Basic je objektově orientovaný programovací jazyk. Objektový

[24] Jak uvádí Microsoft ve svých statistikách, tak MS Office využívá legálně přes 1.2 miliardy lidí a firem. Zdroj: Microsoft by the Numbers. In: *Microsoft.com* [online]. 2016 [cit. 2016-02-01]. Dostupné z: http://news.microsoft.com/bythenumbers/

[25] EDI (Electronic Data Interchange) – Elektronická výměna dat, systémy pro přenos informací. EDI má za cíl zajistit rychlejší a přesnější výměnu informací.
Zdroj: STEHLÍK, Antonín a Josef KAPOUN. *Logistika pro manažery*. Vyd. 1. Praha: Ekopress, 2008, s. 202

[26] GETZ, Ken a Mike GILBERT. *VBA developer's handbook*. 2nd ed. San Francisco: Sybex, 2000, s. 27-28.

model vytvořené aplikace ve Visual Basic jazyku se skládá z různých objektů, jejich metod, tříd[27] a vlastností. Metody a třídy používáme pro manipulaci s objekty a charakteristiky jednotlivých objektů stanovujeme a používáme pomocí jejich vlastností (například Excel obsahuje více než 100 tříd objektů – sešit, pracovní list, oblast buněk, graf a mnoho dalších).[28] Veškerý kód jazyka VBA je uložen v modulech. Moduly jsou součástí editoru, kde jsou fyzicky uloženy (například jednotlivý sešit Excelu, databáze Access). V jednotlivých modulech jsou obsaženy zmiňované procedury a funkce, které obsahují programový kód. Procedury a funkce obsahují příkazy, které jsou prováděny. Procedury a funkce se mohou volat z jiných procedur, nebo funkcí. Některé objekty jsou součástí kolekce, v které se vyskytují další objekty stejného typu.[29]

VBA startovala jako nástroj, který by umožnil Excelu, a pak dalším aplikacím sady Microsoft Office, řídit jejich vlastní prostředí programově a ve spolupráci s ostatními aplikacemi pomocí automatizace OLE[30]. Od roku 1996 je zadarmo VBA součástí každého kancelářského balíku Microsoft Office. Výhodou pro programátory jazyku VBA je, že získané dovednosti programovaní například v Excelu, mohou stejně použít i Accessu a ve Wordu, protože prostředí je stejné, ladící nástroje jsou stejné a syntaxe je stejná (výhodou také je, že je to odnož původního jazyku BASIC[31]). Rozdíl mezi programy sady MS Office je pouze v objektovém modelu dané aplikace. Pokud vývojář nebo zkušenější uživatel zná syntaxi jazyka VBA, tak pro něj není žádný problém tuto

[27] Třída (Class) – je to přepis/vzor pro vytvoření objektů. Digramy tříd zobrazují statickou stránku objektu, především vztahy mezi třídami. Vztahy, které jednotlivé třídy navzájem pojí, jsou asociace, agregace, kompozice, specializace/generalizace.
Zdroj: KANISOVÁ, Hana a Miroslav MÜLLER. *UML srozumitelně*. Vyd. 1. Brno: Computer Press, 2004, s. 51

[28] KRÁL, Martin. *Excel VBA: výukový kurz*. Vyd. 1. Brno: Computer Press, 2010, s. 13

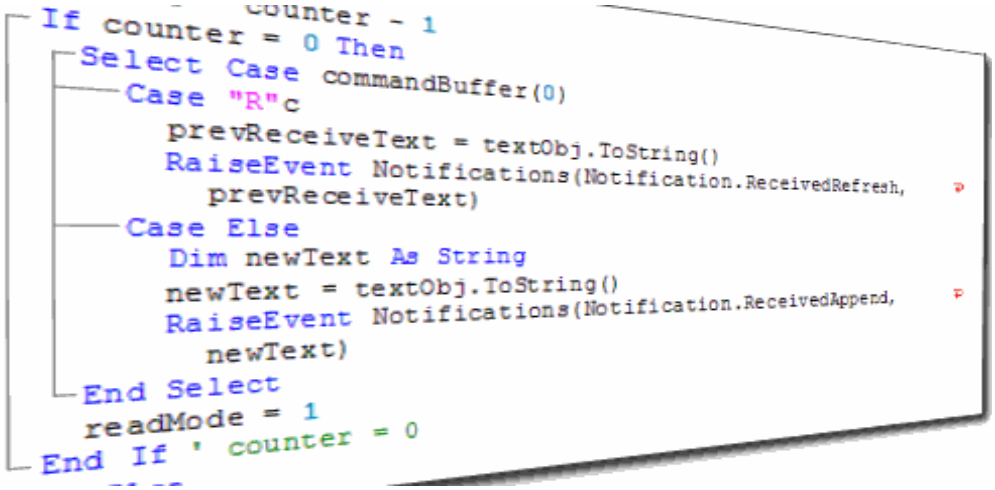
[29] ČERNÝ, Jaroslav. *Excel 2000-2007: záznam, úprava a programování maker*. 2., aktualiz. vyd. Praha: Grada, 2008, s 13-20.

[30] OLE (Object linking and Embedding) - Vkládání a propojování objektů. Protokol MS Windows.

[31] BASIC (Beginner's All-purpose Symbolic Instruction Code) - je rodina programovacích jazyků vysoké úrovně, která byla zavedena jako jednoduchý nástroj pro výuku programování.

syntaxi použít ve všech programech MS Office. Ve VBA můžeme psát aplikace pro koncové uživatele, kteří s programy mohou snadno pracovat, upravovat a dokonce si je sami rozšířit. Uživatelé, kteří programují ve VBA, většinou používají produkty z balíčku MS Office a chtějí si práci v editorech zautomatizovat. Pomocí programů vytvořených v jazyce VBA se dá snadno přejít i do jiných programů, exportovat výsledná data, poslat výsledky například emailem apod., možností je opravdu mnoho.[32]

Obrázek 4: Ilustrační obrázek části kódu vytvořeného pomocí VBA.[33]



Nespornou výhodou pro uživatele, ale i pro vývojáře v prostředí Microsoft Office je nahrávání vlastních maker. Editor má vestavěn zvláštní záznamník, přes který je snadné nahrávat všechny pohyby, kliknutí, stisknutí kláves a vyvolání dialogů v jazyce VBA. Tím můžeme docílit rychlého způsobu jak vytvořit základní makro. Toto nahrané makro

[32] GETZ, Ken a Mike GILBERT. *VBA developer's handbook.* 2nd ed. San Francisco: Sybex, 2000, s. 21.

[33] Obrázek 4: Ilustrační obrázek části kódu vytvořeného pomocí VBA. Zdroj: VBA Select Case Statement – Explained. In: *Excel Trick* [online]. 2015 [cit. 2016-02-01]. Dostupné z: http://www.exceltrick.com/formulas_macros/vba-select-case-statement/

si můžeme snadno upravit (absolutní adresace buněk apod.) a získat tak za pár minut svou vlastní proceduru, nebo funkci. [34]

2.2 Výhody užití VBA pro optimalizaci v praxi

Dle mého názoru je užití a práce s VBA velice vhodné pro účely optimalizace a vytváření důležitých reportů v logistice. Tím, že je VBA součástí MS Office, je také pro firmu nadále finančně nenáročné, protože programování a různé úpravy šité na míru přímo v systémech SCM jsou obvykle velmi časově a finančně náročné.

Autor publikace *Excel VBA* Martin Král[35] považuje za velmi důležitou výhodu VBA její úsporu času. Ve své knize popisuje příběh nového manažera ve velké firmě, v které je několik informačních systémů. A protože výstupy z těchto informačních systémů jsou mnohdy uživatelsky nepříznivé, tak si uživatelé vytváří samostatné tabulky v Excelu. Ale manažer naráží na to, že tabulky jsou vyplňovány ručně a mohou tak být náchylné na chybovost. Jako manažer potřebuje kvalitní informace od všech oddělení a hlavně úplná data. Jelikož manažer ví, jaká potřebuje data a informace může kontaktovat IT oddělení s požadavkem na úpravu stávajících informačních systémů. Jenže naráží na stejné obtíže, které jsem zmiňoval v úvodu této kapitoly (finanční a hlavně časová náročnost). Jako nejlepším řešením se manažerovi jeví automatické stahování dat z informačních systémů, převedení dat do Excelu, nebo Accessu. Tím pádem má veškerá data k dispozici a může vytvořit jednoduchý program ve VBA. Propojí buňky v sestavách, aby měl vždy aktuální přehled po stažení dat. Je to velice jednoduché řešení a až tak nenáročné na čas (pozn. záleží na tom, jakého rozsahu má být program a do jaké míry automatizovaný. Zvláště velmi propracované programy s různými

[34] ČERNÝ, Jaroslav. *Excel 2000-2007: záznam, úprava a programování maker*. 2., aktualiz. vyd. Praha: Grada, 2008, s 13-20.

[35] KRÁL, Martin. *Excel VBA: výukový kurz*. Vyd. 1. Brno: Computer Press, 2010, s. 14-16

exporty do emailu apod. mohou zabrat třeba i měsíce času.). Autor se snaží v podstatě říct, že investováním řádově desítek hodin do jednoduchého programu vytvořeného pomocí VBA si tak může manažer z příběhu zajistit účinně kvalitní a přehledná data potřebná třeba pro řízení podniku.[36]

[36] KRÁL, Martin. *Excel VBA: výukový kurz*. Vyd. 1. Brno: Computer Press, 2010, s. 14-16

Praktická část

Přes historii až po charakteristiku VBA jsme se dostali k praktické náplni bakalářské práce, v níž budeme řešit konkrétní procesy příjmů zboží do distribučního centra a jejich optimalizaci pomocí VBA.

3 Návrh vlastního systému řešení procesu příjmů zboží

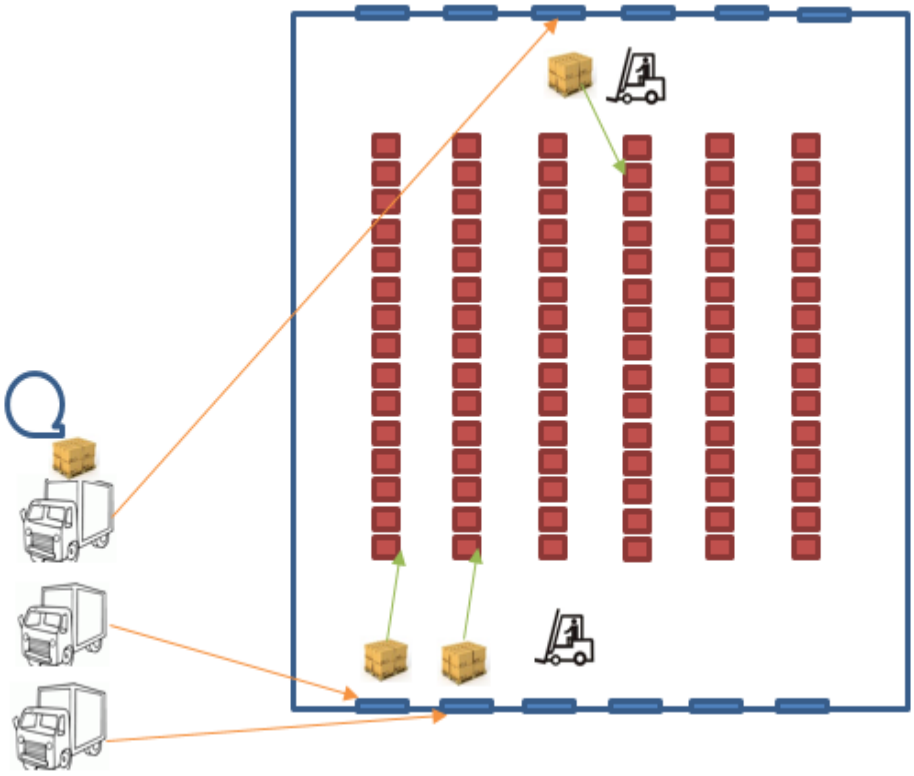
V návrhu optimalizace bychom mohli začít popsáním výchozího stavu ve firmě. Transport (nákladní automobil vezoucí zboží do DC[37]) se zaregistruje při vjezdu do objektu distribučního centra, následně PC operátor vygeneruje v ERP[38] systému dodávku z příslušné objednávky od dodavatele dle dodacích listů a přiřadí tuto dodávku zaregistrovanému transportu. Dále operátor přiřadí v systému daný transport na aktuálně volnou rampu/bránu dle odhadu podle příchozího zboží (customazing - nastavení rozdělení artiklů/zboží k skladovým místům).

Naším úkolem bude optimalizace procesu příjmu zboží (příchozích transportů), tak aby následně nedocházelo k velkým přejezdům vysokozdvižných vozíků od rampy/brány, až k příslušným skladovým místům, které náleží danému přijímanému artiklu/zboží. Obrázek 5 zobrazuje situaci příjmu zboží do skladu. Je zde znázorněna ukázka jedné haly se systémem regálových skladových míst a ramp/bran pro přistavení transportního vozidla.

[37] DC – Distribuční centrum

[38] ERP (Enterprise Resource Planning) - je informační systém pro plánování podnikových zdrojů. Systémy ERP mají za úkol umožňovat zapojení do logistického řetězce podnikům všech velikostí. ERP je základní jádrový software SMC. K tomu musí splnit celou řadu požadavků. Důležité je hlavně propojení ekonomické aplikace v podniku s řešeními pro manažerské informační systémy, pro sdružování zákaznických a prodejních údajů a pro dodavatelské řetězce. ERP umožní každému velkému, střednímu a malému podniku umožnit stát se skutečně článkem logistického řetězce a sítě SCM.
Zdroj: STEHLÍK, Antonín a Josef KAPOUN. *Logistika pro manažery*. Vyd. 1. Praha: Ekopress, 2008, s. 250

Obrázek 5: Ilustrační obrázek – proces příjmu LKW[39].[40]



Každé skladové místo má určitou vzdálenost k jednotlivým branám. Nejlepší pro další postup bude vytvoření matice vzdáleností všech bran k jednotlivým zónám (uličkám ve skladu) skladových míst (Tabulka 1). Matice vzdáleností bude sloužit k výpočtu nalezení nejvhodnější brány/bran pro příjem LKW z hlediska dalších přejezdů při uskladnění přijímaného zboží.

Každá rampa má také prostor k vyložení zboží a k provádění následné kontroly zboží (kontrola zboží probíhá nejčastěji pomocí ručních skenerů, do kterých zaměstnanec zadává data o zboží (jako mohou být například počty kusů v kartonu, počty kartonů na paletě, datum spotřeby, šarže od dodavatele a případně vratné obaly)

[39] LKW (Lastkraftwagen) - Nákladní automobil. Německá zkratka, užívaná i jinde v Evropě.
[40] Obrázek 5: Ilustrační obrázek procesu příjmu zboží - vlastní tvorba. Zdroj vlastní.

a příjmu zboží. Z tohoto prostoru se zkontrolované zboží zaváží do příslušných skladových pozic dle strategie a customazingu. Tento prostor, ale také může sloužit jako blokový sklad.

Tabulka 1 – Ukázka vytvořené matice vzdáleností (hodnoty jsou v metrech) od každé brány/rampy k jednotlivým zónám/ uličkám ve skladu (DC).[41]

Brána/zóna	101WA	102WA	103WA	104WA	105WA	106WA	107WA	108WA
100	33,125	39,625	45,875	51,625	56,625	62,875	68,625	73,625
101	37,375	35,375	41,625	47,375	52,375	58,625	64,375	69,375
102	41,625	35,375	37,375	43,125	48,125	54,375	60,125	65,125
103	45,875	39,625	33,125	38,875	43,875	50,125	55,875	60,875
104	50,125	44,625	37,375	34,625	39,625	45,875	51,625	56,625
105	54,375	48,875	43,375	38,875	35,375	41,625	47,375	52,375
106	58,625	53,125	47,625	42,125	35,375	37,375	43,125	48,125
107	62,875	57,375	51,875	46,375	39,625	33,125	38,875	43,875
108	67,125	61,625	56,125	50,625	45,125	37,375	34,625	39,625
109	71,375	65,875	60,375	54,875	49,375	43,875	38,875	35,375
110	75,625	70,125	64,625	59,125	53,625	48,125	42,625	35,375
111	79,875	74,375	68,875	63,375	57,875	52,375	46,875	39,625
112	84,125	78,625	73,125	67,625	62,125	56,625	51,125	45,625
113	100,625	95,125	89,625	84,125	78,625	73,125	67,625	62,125
114	104,875	99,375	93,875	88,375	82,875	77,375	71,875	66,375
115	109,125	103,625	98,125	92,625	87,125	81,625	76,125	70,625
116	113,375	107,875	102,375	96,875	91,375	85,875	80,375	74,875
117	117,625	112,125	106,625	101,125	95,625	90,125	84,625	79,125
118	121,875	116,375	110,875	105,375	99,875	94,375	88,875	83,375
119	126,125	120,625	115,125	109,625	104,125	98,625	93,125	87,625
120	130,375	124,875	119,375	113,875	108,375	102,875	97,375	91,875
121	134,625	129,125	123,625	118,125	112,625	107,125	101,625	96,125

Navrhovaný program musí stáhnout v reálném čase data z ERP systému (aktuální vytížení bran/ramp – také jejich dostupnost z hlediska probíhajícího příjmu a výdeje, aktuální vytížení skladu a obsazenost skladových míst, informace o již zaregistrovaném

[41] Tabulka 1 – ukázka vytvořené matice vzdáleností od bran k zónám – vlastní tvorba, zdroj vlastní.

LKW z jeho dodávek (objednávek). ERP systémem je v tomto případě SAP[42]. Stažení aktuálních dat ze SAPU do Excelu, v kterém budou probíhat následné výpočty lze dvěma způsoby. První způsob (Obrázek 8 - zde jsem vytvořil univerzální funkci pro stažení dat ze SAP) může být pomocí scriptingu[43] , druhý způsob je pomocí RFC[44] spojení.

Chtěl bych popsat vytvořenou funkci pro stažení dat ze SAP. Rozdíl mezi klasickou procedurou (Sub) a funkcí (Function) ve VBA je v tom, že funkci můžeme volat i v listu excelu za určitých podmínek (podobně jako již naprogramované funkce od výrobce například SUMA, KDYŽ). Na začátku své funkce (Obrázek 8), jsem si ošetřil případné chyby error handlerem, které mohou nastat například při nesplnění stanovených podmínek. `On Error GoTo chyba SAP` – pokud nastane chyba v běhu funkce, tak tok programu přeskočí do návěstí označeného chyba SAP. V části chyba SAP se funkce `Datas_from_SAP` nastaví jako `False`. Zde by bylo možné přidat další náležitosti jako je `MsgBox` (funkce vysklakujícího dialogového okna se zprávou). Na obrázku 6 jsem pro příklad ošetřené funkce vytvořil testovací proceduru s jednoduchým error handlerem a dialogovým oknem MsgBox (Obrázek 7, msgbox vypíše kód chyby, popis chyby a zadanou zprávu pro uživatele). [45]

[42] SAP (Systeme, Anwendungen, Produkte in der Datenverarbeitung) - Software, který je specifický obvykle pro logistiku. Skupina produktů SAP představuje kompletní řešení hlavně pro všechny interní oddělení podniku, ale současně i pokrývá procesy, které přesahují rámec podniku. MAASSEN, André. *SAP R/3: kompletní průvodce*. Vyd. 1. Brno: Computer Press, 2007, s. 14-15.

[43] Scripting – součástí SAPU je „Script Recording and Playback“. Rekordér zaznamenává všechny kroky vykonávané v SAPU (kliknutí, transakce, dialogové okna, navigace). Pomocí této funkcionality je velice snadné nahrát si vlastní skript, který je pak velmi jednoduché upravit a implementovat do VBA. Zdroj vlastní.

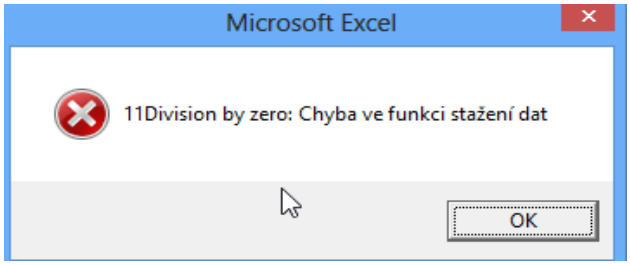
[44] RFC (Remote function call) - Komunikace mezi aplikacemi různých systémů v prostředí SAP zahrnuje spojení mezi systémy SAP, ale i dalšími systémy. RFC je standardní SAP rozhraní pro komunikaci mezi systémy SAP. RFC volá funkci, které má být provedena do vzdáleného systému. Zdroj: RFC. In: *The Best-Run Businesses Run SAP: Help potal* [online]. 2016 [cit. 2016-01-17]. Dostupné z: https://help.sap.com/saphelp_nw70/helpdata/en/6f/1bd5b6a85b11d6b28500508b5d5211/frameset.htm

[45] KRÁL, Martin. *Excel VBA: výukový kurz*. Vyd. 1. Brno: Computer Press, 2010, s. 55-60.

Obrázek 6 : Testovací funkce pro ošetření chyby s využitím MsgBoxu.[46]

```
Sub test()
    Dim cislo As Integer 'deklarace proměnných
    Dim popis_chyby As String
    On Error GoTo chyba
    cislo = 1 / 0  'vyvolání chyby dělení nulou
    Exit Sub ' ukončení procedury
chyba:
    popis_chyby = Err.Number & Err.Description & ": " 'zapsání čísla chyby a popisu chyby
    msg = MsgBox(popis_chyby & "Chyba ve funkci stažení dat", vbCritical) 'vyvolání Msgboxu
End Sub
```

Obrázek 7 : Vyskakovací okno MsgBox vyvolané po spuštění testovací funkce.[47]



V dalším kroku funkce pro stažení dat (Obrázek 8) následuje deklarace proměnných (`Dim`). Definováním proměnných si pojmenuji prakticky místo v paměti počítače. Proměnných typů ve VBA je celá řada například Boolen, Byte, Date, Double, Integer, Long, Object, String, Variant a další typy, které zabírají v paměti různou velikost. Dále pokračueme vytvářením objektů s použím knihoven SAP. V celém kódu využíváme funkce IF. Celý blok IF, Else, End If je jedním z nejdůležitějších ve VBA a má různé varianty. Příkazy, které jsou uvnitř bloku IF se budou vykonávat pouze pokud je splněna podmínka mezi slovy `IF` a `Then` (Když ….pak udělej). `Else` potom slouží, když podmínka splněná není. Jednotlivé podmínky se dají do sebe vnořovat a mohou se tak vytvářet různé podpodmínky. V podmínce můžeme také využít více sekcí a rozlišovat, tak více podmínek pomocí `ElseIf`.[48]

[46] Obrázek 6 : Testovací funkce pro ošetření chyby s využitím MsgBoxu – vlastní tvorba. Zdroj vlastní.

[47] Obrázek 7 : Vyskakovací okno MsgBox vyvolané po spuštění testovací funkce – vlastní tvorba. Zdroj vlastní.

[48] KRÁL, Martin. *Excel VBA: výukový kurz*. Vyd. 1. Brno: Computer Press, 2010, s. 22-41.

Obrázek 8: Funkce pro připojení do SAP pomocí VBA a scriptingu.[49]

```
Function Datas_from_SAP()
    On Error GoTo chyba_SAP 'nastavení error handleru pokud by nastala chyba, tak program nehodí chybu, ale přejde do "chyba_SAP"
    Datas_from_SAP = False 'nastavení funkce jako False na začátku. Pokud proběhne kód správně bude na konci nastaven na True
    '-------------------------začátek připojení do SAP----------------------------
    Dim App, SAP, Connection, Session As Object 'deklarace proměnných
    Dim system_name As String
    Dim end_trans_SAP As String
    system_name = nazev 'název systému SAP
    Set SAP = CreateObject("Sapgui.ScriptingCtrl.1")
    Set SapGuiAuto = GetObject("SAPGUI")
    Set App = SapGuiAuto.GetScriptingEngine()
    If App.Children.Count > 0 Then 'kontrola zda je spuštěný nějaký SAP?
        Dim system_SAP As Integer
        For system_SAP = 0 To App.Children.Count - 1
            If App.Children(0 + system_SAP).Description = system_name Then
                Set Connection = App.Children(0 + system_SAP)
                If Connection.Children.Count > 5 Then 'kontrola počtu otevřených oken systému SAP
                    MsgBox "Zavřete jedno okno SAP " & Left(system_name, 3) & ".", vbOKOnly + vbInformation, "příliš mnoho otevřených oken SAP
                    Exit Function
                End If
                Set Session = Connection.Children(0) 'nastavení 1.okna SAP
                end_trans_SAP = "/i"
                Exit For
            End If
        Next system_SAP
        If Session Is Nothing Then GoTo open_SAP
    Else
open_SAP:
        Set SAP = CreateObject("Sapgui.ScriptingCtrl.1")
        Set Connection = SAP.OpenConnection(system_name, True)
        Set Session = Connection.Children(0)
        end_trans_SAP = "/nex"
    End If
    If IsObject(WScript) Then
        WScript.ConnectObject Session, "on"
        WScript.ConnectObject SAP, "on"
    End If
    pocet_oken_pred_otevrenim_noveho = Connection.Children.Count
    'kontroluje zda je SAP spuštěn na více počítačích
    If end_trans_SAP = "/nex" And Session.Children.Count > 1 Then
        Session.findById("wnd[1]/usr/radMULTI_LOGON_OPT2").Select
        Session.findById("wnd[1]/usr/radMULTI_LOGON_OPT2").SetFocus
        Session.findById("wnd[1]/tbar[0]/btn[0]").press
    End If
    Session.findById("wnd[0]/tbar[0]/okcd").Text = "/o" 'zápis transakce /o vlevo nahoru
    Session.findById("wnd[0]").sendVKey 0 'enter
    Session.findById("wnd[1]/tbar[0]/btn[5]").press 'generování nového režimu SAPU (nové okno SAPU)
    Do 'změna session na nově vytvořené okno SAP
        If count_open_windows_SAP < Connection.Children.Count Then
            Set Session = Connection.Children(Connection.Children.Count - 1)
            Exit Do
        End If
    Loop
    '-------------------------zde vložit nahraný script ze SAP----------------------------

    '-------------------------konec připojení do SAP----------------------------
    Session.findById("wnd[0]/tbar[0]/okcd").Text = end_trans_SAP 'ukončení SAP
    Session.findById("wnd[0]").sendVKey 0 'enter v okně SAP
    Datas_from_SAP = True 'nastavení funkce na true
    On Error GoTo 0 'vypnutí error handleru
    Exit Function 'ukončení funkce
chyba_SAP:
    Datas_from_SAP = False
End Function
```

Ve funkci pro stažení dat používám cykly For a Do. Smyčka `For…Next` je běžným cyklem ve funkcích a procedurách VBA. Cyklus obsahuje čítač, který je automaticky při každém průchodu zvyšován o zvolený krok. Smyčka probíhá tak dlouho, dokud

[49] Obrázek 8: Funkce pro připojení do SAP pomocí VBA a scriptingu - vlastní tvorba. Zdroj vlastní.

vnitřní čítač nepřekročí definovanou hodnotu. Standardně se čítač zvyšuje o 1 právě přes klíčové slovo `Next`, ale můžeme použít v cyklu slovo `Step` a bude se zvyšovat o námi zvolenou hodnotu (`For i = 2 To 6 Step 2` – zde konkrétně proběhne cyklus 3krát při i = 2, i = 4 a i = 6). Cyklus můžeme za určitých podmínek předčasně ukončit použitím `Exit For` (tzn. dříve než překročí námi definovanou hodnotu – například pokud najde potřebná data, nebo dosáhne potřebné hodnoty ve vnořené podmínce). Smyčka `Do…Loop` se používá většinou v případech, kdy nemůžeme nadefinovat počet průchodů předem (`Loop Until` – pokračuj ve smyčce, dokud není podmínka splněná. `Loop While` -pokračuj, pokud není podmínka splněná). [50]

Volání funkce pro stažení dat z nějaké další procedury může vypadat takto:

```
If  Datas_from_SAP = True Then

    Zde můžeme vložit další volání funkcí, nebo další
    příkazy, které se mají vykonat. Podmínka je splněna
    a data jsou stažena.

Else

    Zde můžeme ošetřit například hlášení pro uživatele, že
    neproběhlo stažení dat, tudíž nebude vykonávat další
    kód (program by se mohl dostat do chyby, když chce
    pracovat se staženými daty, ale data stažená nejsou).

End If
```

Tím že jsme v proceduře použili takovou konstrukci volání funkce pro stažení dat uvedenou výše, je výsledkem funkce výstup `True` nebo `False`. Volání funkce nebo procedury se dá napsat více způsoby (například: `Call jmeno_funkce`).

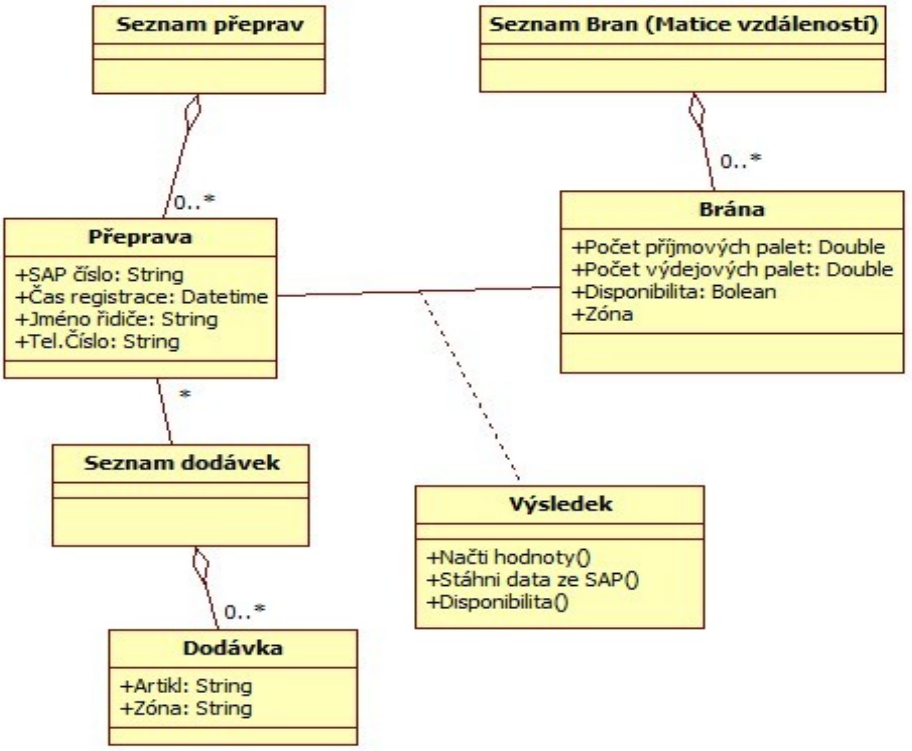
Funkce pro stažení dat je funkční a po vyplnění všech dalších náležitostí a zpracování nahraného scriptingu stahuje potřebná data. Teď nastává otázka jak

[50] KRÁL, Martin. *Excel VBA: výukový kurz*. Vyd. 1. Brno: Computer Press, 2010, s. 44-51.

uchopit a zpracovat tato stažená data (vytížení, disponibilitu bran, informace o dodávkách).

Pro lepší přehled a orientaci jsem vytvořil jednoduchý UML[51] diagram, popisující jak by mohly vypadat budoucí třídy programu (Obrázek 9).

Obrázek 9: UML diagram tříd navrženého programu pro vyhodnocení nejvhodnějších bran k daným transportům.[52]



Každé zboží má své skladové místo neboli zónu artiklu, je to především nejspodnější patro v regálovém systému, z tohoto místa probíhá komise/vyskladnění a na toto místo probíhá uskladnění, nebo doplnění z regálových pozic. Proto je důležité, aby uskladnění palet probíhalo v nejbližším okolí tohoto místa. Toto vyskladnění, uskladnění a doplnění probíhá v rámci customizace skladu v systému SAP. Program

[51] UML (Unified Modeling Language) – je modelovací jazyk a je výsledkem snažení a designerů, který v průběhu 80. a 90. let vytvářeli metody, které by uměli popsat objektově orientovanou analýzu a návrh. UML je souhrnem především grafických notací k vyjádření analytických a návrhových modelů. UML umožňuje modelovat jednoduché i složité aplikace pomocí jedné formální syntaxe. UML je také jazyk pro vizualizaci, specifikaci, stavbu a dokumentaci softwarových systémů. Zdroj: KANISOVÁ, Hana a Miroslav MÜLLER. *UML srozumitelně*. Vyd. 1. Brno: Computer Press, 2004, s. 13

[52] Obrázek 9: UML diagram tříd- vlastní tvorba. Zdroj vlastní.

musí být navržen tak, aby bylo možné uživatelsky rozhodnout, jakou využije variantu dle vytížení skladu. Varianta 1 – regálové pozice v celém skladu jsou zaplněny do 70 %, v této variantě budeme uvažovat o výpočtu pomocí vzdáleností od brány ke skladovému místu zboží (zóny artiklu).

Varianta 2 – regálové pozice jsou zaplněny z více jak 70 %, v této variantě se musí počítat pouze s volnými pozicemi v regálovém systému, protože pokud bude zboží, které se nachází na jedné straně skladu (umístění zóny artiklu), ale v tomto úseku skladu nebude žádné vhodné volné regálové skladové místo, tak vzniknou velké přejezdy, čehož se chceme vyvarovat. V druhé variantě je nutno stahovat také obraz všech regálových míst, což povede k velkému nárůstu času vyhodnocení při opakovaném obnovení vyhodnocení. Řešením tohoto problému bude vytvoření tzv. Back-Endu, který bude pro Front-end tato další data stahovat. Třeba v určitém časovém intervalu 1x za minutu.

Na Obrázku 10 je zobrazena procedura, která stáhne v tomto případě data o prázdných skladových pozicích z back-endu. V této proceduře je použito otevření sešitu, zkopírování potřebných dat do určitého listu. Proměnné (v tomto případě celá cesta k back-endu) jsou uloženy v pojmenovaném listu „help“. Je to z praktického hlediska mnohem efektivnější se odkazovat na soubor z jednoho místa v listu, protože i když soubor bude změněn, přejmenován, přemístěn, tak stačí pouze změnit hodnotu v buňce listu „help“ a program běží dále. Není tak nutné při každé z těchto změn měnit přímo zdrojový kód a procházet všechny procedury, které využívají stažení dat z tohoto sešitu.

Obrázek 10: Kopírování dat z jiného sešitu Excel (backend).[53]

```
Sub Copy_load()
    Dim path_to_load As String
    Dim file_to_load As String
    sheet_finder.Select
    Application.ScreenUpdating = False

    path_to_load = sheet_help.range("K2").Value 'zde jsou parametry umístění backendu (složka)
    file_to_load = sheet_help.range("L2").Value 'zde je název sešitu (BACKENDU)

    Workbooks.Open Filename:=path_to_load & "\" & file_to_load, ReadOnly:=True
    Workbooks(file_to_load).Worksheets("RXX").range("A:M").Copy 'kopírování potřebných dat z backendu

    'zrušení filtru ve skladových místech RXX
    With sheet_RXX
        If .AutoFilterMode = False Then
            .range("A:O").AutoFilter
        Else
            .AutoFilter.ShowAllData
        End If
    .range("A1").PasteSpecial xlPasteAll 'vložení dat do listu RXX
    End With
    Application.CutCopyMode = False
    Workbooks(file_to_load).Close
End Sub
```

V této proceduře je mimo jiné použita konstrukce `With…End With`. V tomto konkrétním případě `With sheet_RXX` znamená, že všechny řádky v bloku kódu až do ukončení `End With` podléhají listu RXX. Stačí pouze napsat „tečku“ a použít nějakou vlastnost pro objekt typu list. Jak jde vidět v kódu například `.Range("A1")`, nebo `.Autofilter`. Ukázka stejného kódu bez použití With je na Obrázku 11 (Na každém řádku kódu kde pracujeme s listem RXX, ho musíme mít v tomto případě jmenovaný/definovaný.). Konstrukce With…End With slouží hlavně k zjednodušení kódu, lepší přehlednosti a možnosti snadněji měnit například jméno listu. [54]

Obrázek 11: Kód bez With konstrukce.[55]

```
If sheet_RXX.AutoFilterMode = False Then
    sheet_RXX.range("A:O").AutoFilter
Else
    sheet_RXX.AutoFilter.ShowAllData
End If
sheet_RXX.range("A1").PasteSpecial xlPasteAll 'vložení dat do listu RXX
```

[53] Obrázek 10: Kopírování dat z jiného sešitu Excel (backend)- vlastní tvorba. Zdroj vlastní.
[54] KRÁL, Martin. *Excel VBA: výukový kurz*. Vyd. 1. Brno: Computer Press, 2010, s. 38-39.
[55] Obrázek11: Kód bez With konstrukce - vlastní tvorba. Zdroj vlastní.

V předchozí proceduře ale i dalších pracujeme s objektem `Range` (neboli oblast buňky/buněk). Objekt `Range` používá vlastnost `Range ("A1")`, která vrací odkaz na objekt `Range`. Kromě adresy buněk se může oblast buněk vyjádřit i dalšími způsoby. Pokud na listu bude pojmenovaná oblast buněk, nemusím se odkazovat přímo adresně, ale třeba pojmenováním oblasti (`.Range("pojmenovana_oblast")`). Vlastnost `Cells` je také možností jak se odkázat na nějakou oblast pomocí dvou parametrů (parametry určují číslo řádku a číslo sloupce - `.Cells(1,1)` je totéž jako `Range("A1")`). [56]

V Excel sešitu (v našem programu) jsou stažena potřebná data ze systému SAP a z back-endu, nyní se musí všechna data seřadit, přepočítat, a zajistit tak správný a srozumitelný výstup pro operátora, který přiřadí z vybraných ramp tu správnou k danému transportu. Na základě zvolené varianty bude program podle obsahu transportu vyhodnocovat skladová místa a jejich vzdálenosti k jednotlivým disponibilním branám.

[56] ČERNÝ, Jaroslav. *Excel 2000-2007: záznam, úprava a programování maker*. 2., aktualiz. vyd. Praha: Grada, 2008, s 113-117.

Obrázek 12: Procedura doplnění skladového místa ke každé paletě dle zóny zboží.[57]

```
Sub strategie1_dle_zony()
    With sheet_shipment
        sheet_finder.Select
        row_help_artikl = 2
        column_artikl_number = 4
        column_type_pal = column_artikl_number + 3
        column_place = column_artikl_number + 2
        column_zone = column_artikl_number + 16
        column_count_pal = column_artikl_number + 9

        Application.ScreenUpdating = False
        last_row_article = .range("A1").End(xlDown).Row

        'vymazat data na listu help
        sheet_help.range("A2:D1000").ClearContents
        sheet_help.range("M2:M1000").ClearContents

        For row_shipment = 2 To last_row_article
            'číslo artiklu
            pocet_radku = .Cells(row_shipment, column_count_pal).Value
            cislo_radku = 1
            Do
                artikl_number = .Cells(row_shipment, column_artikl_number).Value
                type_pal = .Cells(row_shipment, column_type_pal).Value
                zone = .Cells(row_shipment, column_zone).Value
                place = .Cells(row_shipment, column_place).Value
                If zone = "" Then
                    zone = Format(Left(place, 3), "000")
                End If
                sheet_help.range("A" & row_help_artikl).Value = artikl_number
                sheet_help.range("B" & row_help_artikl).Value = type_pal
                sheet_help.range("C" & row_help_artikl).Value = zone
                sheet_help.range("D" & row_help_artikl).Value = place
                sheet_help.range("M" & row_help_artikl).Value = place
                cislo_radku = cislo_radku + 1
                row_help_artikl = row_help_artikl + 1
            Loop Until cislo_radku > pocet

        Next row_shipment

    End With
End Sub
```

Procedura na Obrázku 12 prochází každý materiál ve dvou vnořených cyklech a vypisuje do pomocného listu „help“ skladová místa dle zóny materiálu ke každé paletě. Obdobným způsobem funguje i další procedura, která ale hledá pomocí customizace skladu a materiálů (velikost, vzdálenost, typ, zóna skladového místa) nejbližší prázdná skladová místa k přiřazené zóně materiálu ze stažených dat z listu RXX (v listu RXX jsou aktuální data stažená z back-endu).

[57] Obrázek 12: Procedura doplnění skladového místa ke každé paletě dle zóny zboží - vlastní tvorba. Zdroj vlastní.

Část další funkce pro hledání vzdáleností od zóny k bráně se nachází na Obrázku 13. Zde funkce hledá ve vytvořené matici (Tabulka 1) a počítá vzdálenosti od každého skladového místa (zóny, která byla stanovena, nebo nalezena v předchozí proceduře) ke všem branám. Výsledné vzdálenosti zapíše v ten moment k disponibilním branám a vydělí počtem palet. Tím získá nejlepší průměrný přejezd pro jednu paletu.

Obrázek 13: Část procedury k získání nejlepšího přejezdu (nejlepší brány).[58]

```
row_gates = 2
gates = 0
pocet_palet = Application.WorksheetFunction.CountA(sheet_help.range("A:A"))
Do While gates < 100 And sheet_distance.range("B" & row_gates).Value <> ""
    If sheet_distance.range("A" & row_gates).Value <> "" Then
        If ThisWorkbook.Sheets("help").range("AG1").Value = 0 Then

            sheet_help.range("R" & gates + 2).Value = sheet_distance.range("B" & row_gates).Value
            For p = 2 To pocet_palet
                najit = sheet_help.range("C" & p).Value
                Set najdi_zonu_new = sheet_distance.range("C1:EQ1").Find(what:=najit, lookat:=xlWhole)
                sheet_help.range("W" & gates + 2).Value = sheet_help.range("W" & gates + 2).Value _
                + sheet_distance.Cells(row_gates, najdi_zonu_new.Column).Value
            Next p
            sheet_help.range("W" & gates + 2).Value = sheet_help.range("W" & gates + 2).Value / sheet_help.range("Z2")
        Else
            If Left(sheet_distance.range("B" & row_gates).Value, 1) = "3" Then
                sheet_help.range("R" & gates + 2).Value = sheet_distance.range("B" & row_gates).Value
                For p = 2 To pocet_palet
                    najit = sheet_help.range("C" & p).Value
                    Set najdi_zonu_new = sheet_distance.range("C1:EQ1").Find(what:=najit, lookat:=xlWhole)
                    sheet_help.range("W" & gates + 2).Value = sheet_help.range("W" & gates + 2).Value _
                    + sheet_distance.Cells(row_gates, najdi_zonu_new.Column).Value
                Next p
                sheet_help.range("W" & gates + 2).Value = sheet_help.range("W" & gates + 2).Value / sheet_help.range("Z2")
            End If
        End If
        gates = gates + 1
    End If

    row_gates = row_gates + 1
Loop
```

Mezi další výhody užití VBA oproti VB, patří bezesporu užití již připravených funkcí v Excelu. Na Obrázku 13 využívám například vestavěnou metodu `Left` nebo `Find`. (Left vezme ze zadaného stringu počet zadaných znaků a Find hledá v zadané Range výraz).

[58] Obrázek 13: Část procedury k získání nejlepšího přejezdu (nejlepší brány)- vlastní tvorba. Zdroj vlastní.

V požadavku na vyhodnocení nejlepší brány k danému transportu se musí objevit i možnost zanechat přijímaný materiál v blokovém skladu (strana 26-27). Pro tyto účely nám poslouží již vytvořený back-end. Abychom mohli správně vyhodnocovat materiály, které se oplatí zanechat na blokových pozicích skladu, musíme si je rozdělit do skupin s prioritami dle setrvání zboží na DC (například PRIO1 = 7dní, PRIO2 = 14dní, PRIO3 = 21 dní). Back-end si musí stáhnout aktuální zásoby všech materiálů ve skladu, stáhnout data a vyhodnotit denní obrátku artiklů (počet všech vyskladněných palet za určité období děleno počtem dní v určitém období) a zahrnout i akční materiály u kterých se očekává brzké vyskladnění. Back-end si tak porovná, propočítá a rozdělí ze stražených dat materiály podle priorit (zjednodušeně přičte přijímané palety k aktuální zásobě a vydělí obrátkou nebo odečte akční palety, tak vypočítá setrvání a následně podle setrvání přiřadí materiálu prioritu). Program na straně uživatele si stáhne vyhodnocené materiály z back-endu a následně v běhu programu hledá materiály obsažené v transportu v seznamu materiálů s prioritami (uživateli nabídne příslušné blokové brány, které jsou vyhovující dle customizace a priority).

Výstup pro uživatele může vypadat jako na Obrázku 14. V prvním řádku budou zobrazena čísla zaregistrovaných transportů, v dalších řádcích budou informace o transportech a nejdůležitějším bodem bude zobrazení, zda je zboží v transportu vhodné pro uskladnění do bloku a pět nejlepších bran pro výběr a následné zavolání řidiče. Procedura také vyhodnotila a obarvila buňky v listu podle toho, zda jsou v daný moment jednotlivé brány disponibilní.

Obrázek 14: Zobrazení výsledku pro uživatele.[59]

	Transport	Transport	Transport	Transport	Transport	Transport
čas registrace	7:36:00	7:38:00	10:05:00	8:27:00	8:25:00	8:38:00
jméno řidiče	Novák1	Novák2	Novák3	Novák4	Novák5	Novák6
blok		PRIO 1				
počet pal	30	30	35	30	33	34
nejvhodnější Brána 1	345 / 43	345 / 43	138 / 47	125 / 40	306 / 56	325 / 116
nejvhodnější Brána 2	342 / 48	342 / 48	142 / 50	124 / 44	307 / 58	326 / 118
nejvhodnější Brána 3	341 / 52	341 / 52	137 / 52	129 / 46	311 / 70	332 / 119
nejvhodnější Brána 4	340 / 56	340 / 56	132 / 73	123 / 48	312 / 73	327 / 119
nejvhodnější Brána 5	339 / 60	339 / 60	131 / 77	130 / 50	104 / 76	333 / 120

Program je funkční, ukládá do dalšího sešitu logy, je opatřen error handlerem, který ošetří případné chyby a má možnost zaslat při určitých chybách informaci na příslušné oddělení (Příloha č. 1).

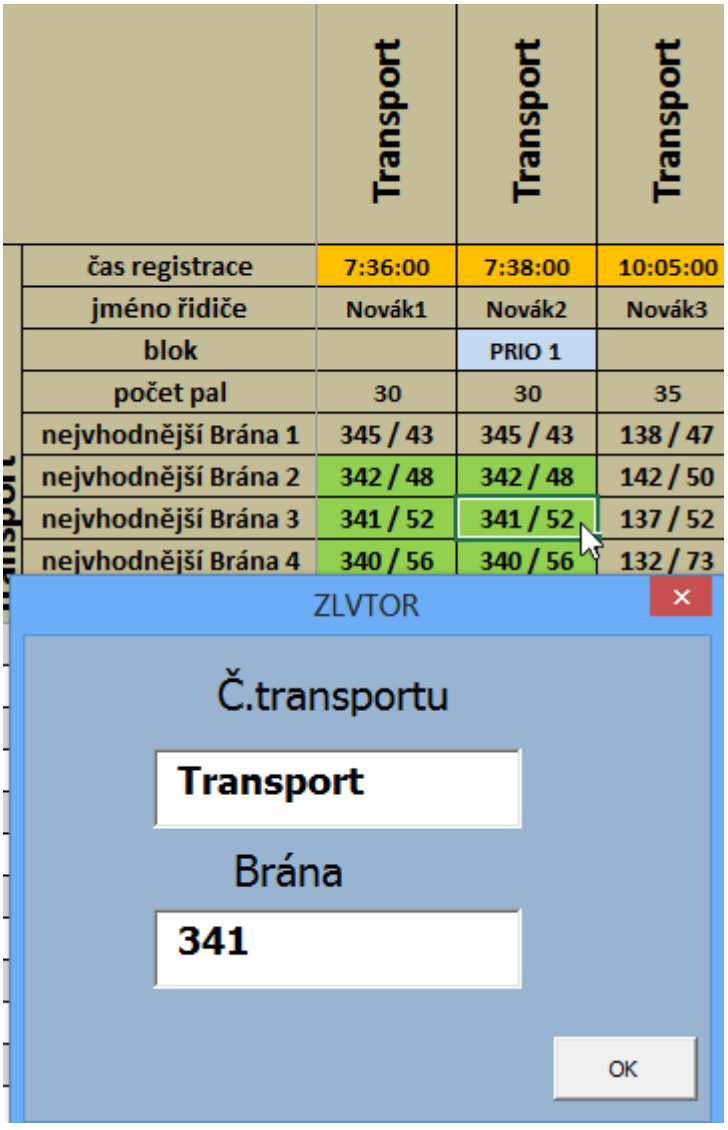
Vybranou nejlepší disponibilní bránu musí operátor/uživatel zadat do systému SAP a tím přiřadit bránu k danému transportu (program tyto informace mimo jiné také stahuje a v rámci další aktualizace programu již tuto bránu nevyhodnotí jako disponibilní pro další možné transporty vezoucí zboží).

Abychom mohli uživateli ještě více pomoci a práci ještě lépe zautomatizovat, vytvoříme jednoduchý formulář, který zadá vybranou bránu do systému (Obrázek 15). Formuláře jsou základní součástí uživatelského rozhraní aplikace. Jejich vzhled a užití může být různé. Do formulářů můžeme přidávat další prvky, jako jsou například Textbox, OptionButton, ListBox, ComboBox, Checkbox, CommandButton. V jednoduchém formuláři budu potřebovat pouze dva textboxy (číslo transportu, číslo

[59] Obrázek 14: Zobrazení výsledku pro uživatele - vlastní tvorba. Zdroj vlastní.

brány) a jedno tlačítko CommandButton na potvrzení výběru. Každé tyto prvky včetně formuláře mají spoustu možností nastavení tzv. Properties. Nastavení je možné měnit při vytváření návrhu formuláře, nebo i v průběhu vykonávání procedury pomocí příkazů.[60]

Obrázek 15: Zobrazení formuláře pro výběr brány a zadání do systému.[61]



[60] KRÁL, Martin. *Excel VBA: výukový kurz*. Vyd. 1. Brno: Computer Press, 2010, s. 416-422.

[61] Obrázek 15: Zobrazení formuláře pro výběr brány a zadání do systému - vlastní tvorba. Zdroj vlastní.

Na obrázku 15 lze vidět aktuální výběr buňky v listu s následným vyvoláním formuláře. Procedura vybrala data z aktuálního umístění kurzoru. Celá procedura výběru aktuálně označených dat v listu se nachází na Obrázku 16.

Obrázek 16: Procedura pro vyplnění dat dle aktuálního umístění v buňce a zobrazení formuláře.[62]

```
Dim poziceradek As Integer
Dim pozicesloupec As Integer
Dim brana As String
Dim transport As String
Dim oznaceni As String
Dim arr As Variant
With ThisWorkbook.Sheets("vyhodnoceni")
    poziceradek = ActiveCell.Row 'načte hodnotu řádku podle aktuálního umístění kurzoru
    pozicesloupec = ActiveCell.Column 'načte hodnotu sloupce podle aktuálního umístění kurzoru

    If poziceradek > 16 Then
        brana = .Cells(poziceradek, 6) 'zapíše bránu
        transport = .Cells(1, pozicesloupec) 'zapíše transport
    Else
        oznaceni = .Cells(poziceradek, pozicesloupec).Value
        If oznaceni = "" Then
            transport = .Cells(1, pozicesloupec)
            brana = "" 'zapíše prázdnou bránu
        Else
            arr = Split(oznaceni, " /") 'rozdělí pole
            brana = arr(LBound(arr)) 'zapíše bránu
            transport = .Cells(1, pozicesloupec) 'zapíše transport
        End If
    End If
End With
ZLVTOR.txt_brana = brana 'vypsání brány do formuláře
ZLVTOR.txt_transport = transport 'vypsání transportu do formuláře
ZLVTOR.Show 'otevření formuláře
```

Po stisknutí tlačítka „OK“ jak lze vidět na obrázku 15 následuje připojení do SAPU a vyplnění hodnot. Pro připojení do SAPU můžeme znovu použít již uváděnou funkci (Obrázek 8).

Celý vytvořený program lze ošetřit a zrychlit pomocí `Application.ScreenUpdating = False` (běh procedur/y se bude provádět na pozadí bez obnovování obrazovky), `Application.Calculation = xlCalculationManual` (v části programu můžeme průběžně vypínat a zapínat přepočet buněk a vzorců – automaticky totiž Excel přepočítává buňky v listu při každé změně v listu, takže pokud nebude mít prováděná změna procedurou vliv na výpočet,

[62] Obrázek 16: Procedura pro vyplnění dat dle aktuálního umístění v buňce a zobrazení formuláře - vlastní tvorba. Zdroj vlastní.

nebo pokud nepotřebujeme průběžně pro proceduru přepočítat hodnotu buněk v listu je to velmi jednoduchá cesta jak běh programu rapidně zrychlit).[63]

Pro ideální informování uživatele o stavu vyhodnocení je možné využít v Excelu StatusBar. V průběhu kódu se dá text ve StatusBar měnit. V cyklu procedury využíváme této funkcionality (viz níže).

```
Application.StatusBar = "zpracováno " & Format(radek / (celkem_radku * 0.01), "00.0") & "% přeprav, předpokládaný čas do konce: " & Format(((start_time - Now) / radek) * (celkem_radku - radek), "hh:mm:ss")
```

Na začátku celé procedury je uloženo do proměnné aktuální datum a čas (`start_time = Now`). Před spuštěním cyklu je spočítáno kolik řádků celkem mají stažená data v listě (proměnná `celkem_radku`). Vydělením řádku (proměnná `radek`, která je v cyklu zvyšována až do celkem řádků) od celkových počtů řádků získáme procento aktuálního vyhodnocení. Odhadovaný čas do konce vyhodnocení spočítá násobením odečteného původního času od času nyní vyděleným aktuálním řádkem a odečtem aktuálního řádku od všech řádků. Při velkém objemu dat je to užitečný a velmi dobře vypadající způsob, jak uživatele informovat o stavu aktualizace.

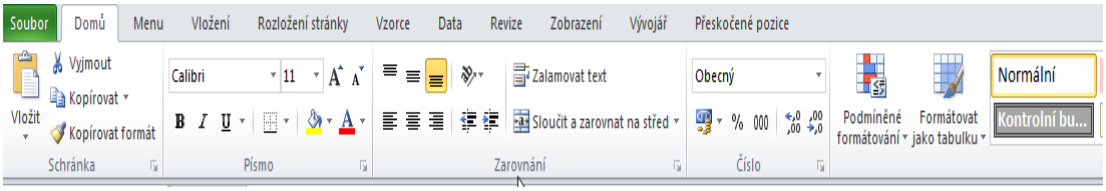
Aby program nejen správně, bez chyb a rychleji fungoval, musí být také uživatelsky příjemný. StatusBar nám pomohl s informacemi pro uživatele o stavu aktualizace, ale nesmíme zapomenout na tlačítka a ovládací prvky programu. Tlačítka potřebujeme pro spuštění aktualizace, také pro vyvolání vytvořeného formuláře a pro mnoho dalších naprogramovaných funkcí a procedur. V Excelu, ale i v dalších programech sady Microsoft Office, existuje spousta možností jak tlačítka zakomponovat do editoru.

[63] KRÁL, Martin. *Excel VBA: výukový kurz*. Vyd. 1. Brno: Computer Press, 2010, s. 323-333.

Můžeme například vložit do listu v Excelu obrázek a z toho obrázku následně vytvořit tlačítko a přiřadit k němu funkci, nebo proceduru. Nebo můžeme využít ovládací prvky formuláře a ovládací prvky ActiveX, které vložíme také do samotného listu. Takto vložená tlačítka, textová pole a další prvky stejné jako při vytváření formuláře (strana 39-40), nejsou dle mého názoru příliš elegantní a nepůsobí tak úplně profesionálně. Proto bych chtěl využít možnosti změny Ribbon menu.

Základní Ribbon Menu v produktech Microsoft Office je důležitý, protože obsahuje veškeré ovládací prvky pro práci v editorech (záložky - Soubor, Domů, Vložení, Rozložení stránky atd., tlačítka – Vložit, Graf, Obrazce apod., další různé prvky jako CheckBoxy, EditBoxy atd. Viz Obrázek 17.)

Obrázek 17: Ukázka klasického Ribbon Menu v Excelu.[64]



Veškeré menu je „uschované“ například uvnitř celého sešitu Excel v jazyce XML[65]. Upravovat toto menu v jazyce XML můžeme například pomocí různých editorů, které nám umožní vkládat vlastní ikony, validují správnost zapsaného kódu a třeba vygenerují část kódu, kterou můžeme umístit do našich modulů a tím přímo ovládat v běhu našeho programu vytvořené vlastní menu. Aby program vypadal

[64] Obrázek 17: Ukázka klasického Ribbon Menu v Excelu. Zdroj vlastní.

[65] *„XML je zkratka z anglického eXtensible Markup Language, rozšiřitelný značkovací jazyk. Ve skutečnosti je XML tzv. metajazyk, nadřazený značkovací jazyk, v rámci něhož je možné vytvářet vlastní jazyky. XML je zajímavý tím, že neobsahuje žádné konkrétní značky (elementy), kdokoliv si tedy může vymyslet vlastní značky, např. <barva>zelená</barva>. To umožňuje velice dobře definovat přesnou strukturu každého XML dokumentu podle aktuální potřeby. XML je tak jakýmsi předělem mezi databázovou strukturou a textovým dokumentem. „*

Zdroj: Co je XML. In: *Adaptic: Internetová řešení podle vašich potřeb* [online]. 2015 [cit. 2016-03-08]. Dostupné z: http://www.adaptic.cz/znalosti/slovnicek/xml/

profesionálněji, je možné úplně skrýt klasické vestavěné menu od Microsoftu a vytvořit si pouze vlastní záložky, vlastní tlačítka, vlastní ovládací prvky (Pozn. tím můžeme docílit i dalšího ošetření běhu programu, protože uživatel nebude moci bez vestavěných tlačítek dělat v Excelu různé změny, které mohou mít vliv na další průběh procedur a funkcí.).

Obrázek 18: Nadefinované menu pomocí XML.[66]

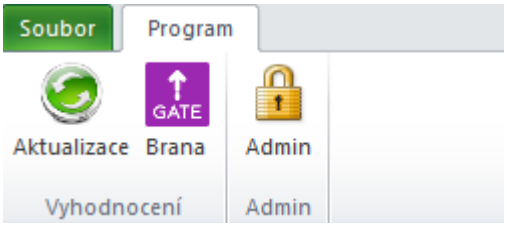
```
<customUI xmlns="http://schemas.microsoft.com/office/2009/07/customui" onLoad="Loadribbon1">
<ribbon startFromScratch="true">

<tabs>
<tab idMso="TabHome" getVisible="skryj" />
<tab idMso="TabReview" getVisible="skryj" />
<tab idMso="TabInsert" getVisible="skryj" />
<tab idMso="TabReferences" getVisible="skryj" />
<tab idMso="TabView" getVisible="skryj" />
<tab idMso="TabDeveloper" getVisible="skryj" />
<tab idMso="TabPageLayoutExcel" getVisible="skryj" />
<tab idMso="TabData" getVisible="skryj" />
<tab idMso="TabFormulas" getVisible="skryj" />

<tab id="MyCustomTab" label="Program" >
<group id="customGroup1" label="Vyhodnocení">
<button id="customButton2" label="Aktualizace" size="large" onAction="hromadne_aktualizace" image="refresh1" />
<button id="customButton3" label="Brana" size="large" onAction="gate" image="gate3" />
</group>
<group id="customGroup2" label="Admin">
<button id="customButton4" label="Admin" size="large" onAction="admin" imageMso="lock" />
</group>
</tab>
</tabs>
</ribbon>
</customUI>
```

V kódu, který lze vidět na Obrázku 18, jsme všem vestavěným prvkům editoru přidělili pomocí parametru getVisible=“skryj“. Pokud si definujeme v procedurách funkci „skryj“ a budeme ji předávat True nebo False, tak se dle toho budou skrývat nebo odkrývat položky v menu programu. Dále jsem si definoval vlastní záložku s popiskem „Program“, v které jsou dvě skupiny tlačítek a tři tlačítka. Skupinám i tlačítkům mohu nastavovat také různé parametry (mezi nejdůležitější patří onAction, který udává, co se bude dít po stisknutí tlačítka). Zvolil jsem tlačítkům i vlastní obrázky

[66] Obrázek 18: Nadefinované menu pomocí XML – vlastní tvorba. Zdroj vlastní.

pomocí parametru image. Tato tlačítka a ukázku vlastního menu lze vidět na Obrázku 19.

Obrázek 19: Výsledek nadefinovaného vlastního menu.[67]



Tlačítko „aktualizace“ se využije pro spuštění celkové aktualizace programu Tlačítkem „brána“ se zavolá již vytvořený formulář (Obrázek 15). Třetí tlačítko se může využít pro obnovu skrytého vestavěného menu například po zadání hesla.

Navrhovaný program nese známky profesionálně vytvořeného programu na zakázku, jako jsou uživatelská rozhraní, ošetření chyb, grafické zpracování, srozumitelný výstup, optimalizace běhu programu a informace o průběhu aktualizace.

Navrhovaný program by mohl velmi pozitivně ovlivnit procesy příjmu zboží. Výsledkem bude úspora času při přejezdech, při uskladňování skladových jednotek do regálových pozic, z toho plyne i finanční úspora pro celý podnik. Za stejný pracovní čas se stihne uskladnit více palet. Navrhovaný program se dotýká i procesů výdeje zboží a ušetření času i při vyskladnění například u zboží, které bude umístěno dle výpočtů do blokových pozic.

[67] Obrázek 19: Výsledek nadefinovaného vlastního menu – vlastní tvorba. Zdroj vlastní.

Závěr

Ve své bakalářské práci jsem se zabýval optimalizací procesů logistiky a snižování nákladů v oblasti skladového hospodářství.

Hlavním cílem bakalářské práce bylo navržení programu v aplikaci Visual Basic for Application na ovládání logistických procesů příjmu zboží. Cíl byl naplněn vlastním návrhem uživatelského programu pomocí Visual Basic for Application, který řeší a pozitivně ovlivňuje procesy příjmu zboží.

Pro dosažení hlavního cíle nám pomohly i stanovené dílčí cíle. Prvním dílčím cílem bylo popsání logistiky, její vývoj, popis příjmu zásob a skladového hospodářství. Dílčího cíle bylo dosaženo v teoretické části a následně v praktické části, v které byla provedena analýza teoretických informací, doplněná o vlastní poznatky z oblasti skladového hospodářství, získané po dobu mého působení ve velké nadnárodní společnosti. Druhý dílčí cíl byl zaměřen na analýzu aplikace Visual Basic for Application, v které probíhal samotný návrh vytvářeného programu pro optimalizaci procesů příjmu. Druhý dílčí cíl byl naplněn v teoretické části, ale i v praktické části, kde jsme využili nemalého potenciálu, kterým disponuje Visual Basic for Application.

Bakalářská práce i navržený program by se z mého pohledu mohl rozšířit o další možnosti optimalizace logistických procesů. Pro příklad by program mohl být v budoucnu rozšířen o management a monitor všech příjmových ploch a ramp. Program by také mohl v budoucnu řešit a optimalizovat práci příjmových kontrolorů, kteří přijímají zboží na příjmových rampách. A to tak, že by posílal pracovníky na nejbližší obsazené brány transportem dle určitých priorit. Stejně jako u navrženého programu pro příchozí transporty, který optimalizuje přejezdy, by se tak i v tomto případě zredukovaly přechody osob a zvýšila by se tak efektivita příjmu zboží.

Literatura a prameny:

Celkové náklady. *Firemní slovník.cz* [online]. 2016 [cit. 2016-01-30]. Dostupné z: http://www.firemnislovnik.cz/c/celkove-naklady/

Co je XML. In: *Adaptic: Internetová řešení podle vašich potřeb* [online]. 2015 [cit. 2016-03-08]. Dostupné z: http://www.adaptic.cz/znalosti/slovnicek/xml/

ČERNÝ, Jaroslav. *Excel 2000-2007: záznam, úprava a programování maker*. 2., aktualiz. vyd. Praha: Grada, 2008. Průvodce (Grada), 183 s. ISBN 978-80-247-2305-1.

GETZ, Ken a Mike GILBERT. *VBA developer's handbook*. 2nd ed. San Francisco: Sybex, 2000, xxx, 1073 p. ISBN 0782129781.

KANISOVÁ, Hana a Miroslav MÜLLER. *UML srozumitelně*. Vyd. 1. Brno: Computer Press, 2004, 158 s. ISBN 80-251-0231-9.

KRÁL, Martin. *Excel VBA: výukový kurz*. Vyd. 1. Brno: Computer Press, 2010, 504 s. ISBN 978-80-251-2358-4.

LAMBERT, Douglas M, James R STOCK a Lisa M ELLRAM. *Logistika*. Vyd. 2. Praha: Computer Press, 2000, xviii, 589 s. ISBN 80-7226-221-1.

MAASSEN, André. *SAP R/3: kompletní průvodce*. Vyd. 1. Brno: Computer Press, 2007, 733 s. Informační systémy. ISBN 978-80-251-1750-7.

Peněžní tok. In: *Management mania* [online]. 2015 [cit. 2016-02-14]. Dostupné z: https://managementmania.com/cs/penezni-tok

PERNICA, Petr. *Logistika pro 21. století: (Supply chain management)*. Vyd. 1. Praha: Radix, 2005, 3 sv. ISBN 80-86031-59-4.

RFC. In: *The Best-Run Businesses Run SAP: Help potal* [online]. 2016 [cit. 2016-01-17]. Dostupné z: https://help.sap.com/saphelp_nw70/helpdata/en/6f/1bd5b6a85b11d6b28500508b5d5211/frameset.htm

SCHULTE, Christof. *Logistika.* Vyd. 1. Praha: Victoria Publishing, 1994, 301 s. ISBN 80-85605-87-2.

SIXTA, Josef a Miroslav ŽIŽKA. *Logistika: metody používané pro řešení logistických projektů*. Vyd. 1. Brno: Computer Press, 2009, 238 s. Praxe manažera (Computer Press). ISBN 978-80-251-2563-2.

SIXTA, Josef a Václav MAČÁT. *Logistika: teorie a praxe*. Vyd. 1. Brno: CP Books, 2005, 315 s. Business books (CP Books). ISBN 80-251-0573-3.

STEHLÍK, Antonín a Josef KAPOUN. *Logistika pro manažery*. Vyd. 1. Praha: Ekopress, 2008, 266 s. ISBN 978-80-86929-37-8.

Zkratky.cz. Zkratky.cz [online]. [cit. 2016-01-14]. Dostupné z: http://www.zkratky.cz/

Seznam zkratek:

DC (Distribuční centrum) - logistické centrum (skladovací plochy).

EDI (Electronic Data Interchange) - Elektronická výměna dat, systémy pro přenos informací.

ERP (Enterprise Resource Planning) - je informační systém pro plánování podnikových zdrojů.

FIFO (First-in/ first-out) – druh fronty. Nejdříve je obsloužen nejstarší požadavek.

LKW (Lastkraftwagen) - nákladní automobil. Německá zkratka, užívaná i jinde v Evropě.

OLE (Object linking and Embedding) - vkládání a propojování objektů. Protokol MS Windows.

RFC (Remote function call) - komunikace mezi aplikacemi různých systémů v prostředí SAP.

SAP (Systeme, Anwendungen, Produkte in der Datenverarbeitung) - software, který je specifický obvykle pro logistiku.

SCM (Supply Chain Management) - je oblast řízení, která zahrnuje všechny procesy komunikace s dodavateli v celém dodavatelském řetězci.

UML (Unified Modeling Language) - je modelovací jazyk. UML je souhrnem především grafických notací k vyjádření analytických a návrhových modelů.

VB (Visual Basic) - programovací jazyk

VBA (Visual Basic for Application) - nástroj pro řízení produktů MS Office programově.

Seznam obrázků:

Seznam tabulek:

Přílohy:

Příloha č. 1	Funkce pro odesílání emailu pomocí programu Lotus Notes – vlastní tvorba. Zdroj vlastní.

Příloha č. 1 – Funkce pro odesílání emailu pomocí programu Lotus Notes

```
Function Send_Formatted_Rangedata(obsah As String, adresamail As String, hlavicka As String)
    'proměnné
    Dim noSession As Object, noDatabase As Object, noDocument As Object
    Dim prijemci() As String
    'příjemce
    prijemci = Split(adresamail, ",")
    On Error Resume Next
    'Inicializace Lotus Notes objektů.
    Set noSession = CreateObject("Notes.NotesSession")
    If noSession Is Nothing Then
        MsgBox "Nemáte otevřený Lotus Notes, požadavek nelze zadat.", vbCritical + vbOKOnly, "chyba"
        Send_Formatted_Rangedata = False
    Else
        Set noDatabase = noSession.GETDATABASE("", "")
        'Je LN otevřený a dostupný?
        If noDatabase.IsOpen = False Then noDatabase.OPENMAIL
        'Vytvoření nového emailu
        Set noDocument = noDatabase.CREATEDOCUMENT
        'Vložení dat do emailu
        With noDocument
          .Form = "Memo"
          .SendTo = prijemci
          .Subject = hlavicka
          .body = obsah
          .SAVEMESSAGEONSEND = True
        End With
        'Poslat email
        With noDocument
          .PostedDate = Now()
          .SEND 0
        End With
        'Smazání objektů z paměti
        Set embedobject = Nothing
        Set obAttach = Nothing
        Set noDocument = Nothing
        Set noDatabase = Nothing
        Set noSession = Nothing
        Send_Formatted_Rangedata = True
    End If
End Function
```