Vysoká škola logistiky o.p.s.

Využití VBA v práci dispečera

(Bakalářská práce)

Přerov 2023 Radek Šindelář

Vysoká škola logistiky o.p.s.

Zadání bakalářské práce

student	**Radek Šindelář**
studijní program	LOGISTIKA
specializace	Informatika pro logistiku

Vedoucí Katedry bakalářského studia Vám ve smyslu čl. 22 Studijního a zkušebního řádu Vysoké školy logistiky o.p.s. pro studium v bakalářském studijním programu určuje tuto bakalářskou práci:

Název tématu: **Využití VBA v práci dispečera**

Cíl práce:

Cílem práce je zhodnocení možnosti implementace VBA do práce dispečera sběrné služby a návrh jejího zjednodušení za pomoci maker v programu MS Excel.

Zásady pro vypracování:

Využijte teoretických východisek oboru logistika. Čerpejte z literatury doporučené vedoucím práce a při zpracování práce postupujte v souladu s pokyny VŠLG a doporučeními vedoucího práce. Části práce využívající neveřejné informace uveďte v samostatné příloze.

Bakalářskou práci zpracujte v těchto bodech:

Úvod
1. Teoretická východiska - zasilatelství, sběrná služba
2. IS v práci dispečera SBS
3. VBA a jeho možnosti
4. Analýza současného stavu práce dispečera SBS
5. Plánování pomocí VBA
6. Vyhodnocení implementace

Závěr

Rozsah práce: 35 – 50 normostran textu

Seznam odborné literatury:

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

LAURENČÍK, Marek. Excel - pokročilé nástroje: funkce, marka, databáze, kontingenční tabulky, prezentace, příklady. Praha: Grada, 2016. Průvodce (Grada). ISBN 978-80-247-5570-0.

GROS, Ivan a kol. Velká kniha logistiky. Vydání: první. Praha: Vysoká škola chemicko-technologická v Praze, 2016. ISBN 978-80-7080-952-5.

KAMPF, Rudolf, Václav CEMPÍREK a Rudolf KAMPF. Zasílatelství. Vyd. 1. Pardubice: Univerzita Pardubice, 2005. 101 s. ISBN 80-7194-745-8.

NOVÁK, Radek. Nákladní doprava a zasílatelství. 2., přeprac. vyd. Praha: ASPI, 2005. 412 s. ISBN 80-7357-086-6.

Vedoucí bakalářské práce: Ing. Libor Kavka, Ph.D.

Datum zadání bakalářské práce: 31. 10. 2022
Datum odevzdání bakalářské práce: 29. 4. 2023

Přerov 31. 10. 2022

Ing. et Ing. Iveta Dočkalíková, Ph.D.
vedoucí katedry

prof. Ing. Václav Cempírek, Ph.D.
rektor

Čestné prohlášení

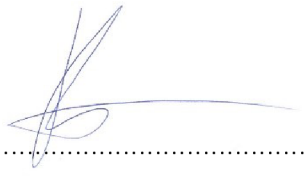
Prohlašuji, že předložená bakalářská práce je původní, a že jsem ji vypracoval samostatně. Prohlašuji, že citace použitých pramenů je úplná, a že jsem v práci neporušil autorská práva ve smyslu zákona č. 121/2000 Sb.; o autorském právu, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších předpisů.

Prohlašuji, že jsem byl také seznámen s tím, že se na mou bakalářskou práci plně vztahuje zákon č. 121/2000 Sb., o právu autorském, právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, zejména § 60 – školní dílo. Beru na vědomí, že Vysoká škola logistiky o.p.s. nezasahuje do mých autorských práv užitím mé bakalářské práce pro pedagogické, vědecké a prezentační účely školy. Užiji-li svou bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat předtím o této skutečnosti prorektora pro vzdělávání Vysoké školy logistiky o.p.s.

Prohlašuji, že jsem byl poučen o tom, že bakalářská práce je veřejná ve smyslu zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, zejména § 47b. Taktéž dávám souhlas Vysoké škole logistiky o.p.s. ke zpřístupnění mnou zpracované bakalářské práce v její tištěné i elektronické verzi. Souhlasím s případným použitím této práce Vysokou školou logistiky o.p.s. pro pedagogické, vědecké a prezentační účely.

Prohlašuji, že odevzdaná tištěná verze bakalářské práce a verze nahraná do informačního systému školy jsou totožné.

V Přerově, dne 18. 4. 2023



.....

podpis

Poděkování

Rád bych poděkoval své rodině za podporu a svému vedoucímu bakalářské práce panu Ing. Liboru Kavkovi, Ph.D. za profesionální přístup, návrhy a motivaci při vedení této práce.

Anotace

Cílem práce je zhodnocení možnosti implementace VBA do práce dispečera sběrné služby a návrh jejího zjednodušení za pomoci maker v programu MS Excel. Za tímto účelem jsou představena teoretická východiska sběrné služby a VBA, porovnány způsoby zpracování dat a představena jak jednoduchá, tak složitější makra pro práci s těmito daty. Dále porovnány metody pro dynamickou práci s oblastmi včetně časové náročnosti, a konečně představeno makro vhodné pro třízení zásilek včetně vývojového diagramu.

Klíčová slova

Sběrná služba, Dispečer, VBA, Excel

Annotation

The aim of the thesis is to evaluate the possibility of implementing VBA in the work of a dispatcher of a groupage service and to propose its simplification using macros in MS Excel. For this purpose, the theoretical background of the goupage service and VBA is presented, the methods of data processing are compared and both simple and more complex macros for working with this data are introduced. Furthermore, methods for dynamic work with areas are compared, including time requirements, and finally a macro suitable for sorting shipments is introduced, including a flow chart.

Keywords

Groupage service, Dispatcher, VBA, Excel

Obsah

Úvod

Žijeme v době digitalizace. S rozvojem informačních technologií se mění nejen naše každodenní životy, ale také způsob, jakým pracujeme. Digitalizace se stala nedílnou součástí pracovního procesu a přinesla s sebou mnoho příležitostí pro zjednodušení a optimalizaci práce. Lidských činností, které již bez digitalizace a informačních technologií dělat nelze, rychle přibývá, a ačkoliv se najdou tací, kteří se digitalizaci stále vyhýbají, většina pracujících již vnímá počítače jako prostředek ke zjednodušení a zefektivnění jejich práce. V dopravní logistice je důvodů k použití digitalizace mnoho. Již zmíněná rychlost, přesnost, efektivita, snížení chybovosti, rychlá přenositelnost dat mezi odděleními, přehlednost, variabilita výsledků a mnoho dalších, stejně jako v mnoha jiných odvětvích. Proto věřím, a doufám, že dispečer, v jedné ruce s tužkou a papírem a v druhé ruce počítačem s Excelem, nebude dlouze přemýšlet o své volbě.

Dispečeři musí být schopni efektivně pracovat s velkým množstvím dat a informací, a to v co nejkratším čase. Tato práce v tomto ohledu porovnává rozdíly mezi zpracováním fyzickým a digitálním. Také se ale zabývá každodenními pracovními činnostmi všude tam, kde lze využít VBA, hlavně však MS Excel a MS Outlook, protože se bezesporu jedná o nejpoužívanější kancelářské aplikace.

Na základě této práce se tak bude možné inspirovat při tvorbě maker a scriptů vedoucím k úspěšnému zjednodušení práce dispečera. Jsou zde k tomuto účelu blíže představeny konkrétní příklady maker včetně programových kódů využívaných v praxi dispečera sběrné služby, rozebráno využití tříd a příklady metodiky přístupu k dynamickým oblastem v MS Excel, včetně časového testování a porovnání. Všechny tyto prvky lze následně využít k představě, jak by bylo možné vhodně a efektivně zjednodušit práci dispečera tak, aby se již nikdy nemusel chtít vracet k oné tužce a papíru a bez velkých pochybností začal i on využívat výhod digitalizace.

1 Vymezení základních pojmů

Silniční doprava je u nás nejrozšířenějším druhem dopravy. Jak uvádí Stodola, v roce 2003 se jednalo o 68 % [1 str. 78]. Podle statistiky webu Eurostat představovala silniční doprava v ČR v roce 2021 77,1 % celkového počtu všech dopravních úkonů [2]. Její využití tedy stále roste. Stodola se obával vyčerpání kapacit [1 str. 78], ale vzhledem k neustálém růstu se do toho stavu nedostaneme alespoň do té doby, dokud silniční doprava nenalezne vhodného konkurenta, co se týče rychlosti a flexibility. [3]

1.1 Silniční nákladní doprava

Silniční nákladní doprava zahrnuje činnosti sloužící k přepravě zvířat a věcí vozidly, případně přemisťování vozidel samých po silnicích, dálnicích, místních a účelových komunikacích a volném terénu. V nákladní dopravě je tohoto přesunu využíváno k přidání hodnoty přepravovanému nákladu a takto je klíčovou složkou materiálových řetězců, propojující dodavatele surovin s konečným spotřebitelem. [4 str. 113] [5 str. 13] [1 str. 69]

Mezi základní charakteristiky, představující také důvody výsadního postavení silniční dopravy mezi přepravami vůbec, patří:

- nejnižší doba přepravy na krátké vzdálenosti,
- hustá síť silniční infrastruktury, která dává možnost dosáhnout míst dle požadavků zákazníka, například při přepravě zásilek,
- včasné a rychlé dodávky,
- možnost volby dopravního prostředku dle potřeby, podle velikosti zásilky,
- vyšší bezpečnost díky neustálému dohledu řidiče nad zásilkou, [4 str. 123]

kterou se pak rozumí přepravní jednotka, podaná k přepravě [1 str. 71]. Pro účely této práce je také velmi důležité definovat typy zásilek v silniční nákladní dopravě. Z komerčně – organizačního hlediska je lze rozdělit na 3 typy:

1 **Celovozová zásilka** – někdy také označovaná jako FTL = Full truck load, přeprava zboží pomocí plně naložených nákladních vozidel, která jsou určena

pouze pro jednoho zákazníka a jeden konkrétní náklad. Celá kapacita vozidla je vyčerpána jediným nákladem,

2 **Přikládka** – také LTL = Less than Truck Load, je kusová zásilka, přepravována společně se zásilkami určenými pro jiného příjemce za účelem lepšího využití kapacity vozidla,

3 **Nadrozměrná zásilka a speciální přepravy** – zásilky podléhající zvláštním předpisům nebo podmínkám. [6 str. 12] [7 str. 181]

1.2 Sběrná služba (SBS)

Do tohoto rozdělení se někdy také řadí sběrná služba, zastřešující celý systémový přístup k přepravě kusových zásilek. Velmi vhodnou definici uvádí Novák: „*SBS je obecně založena na maximálně efektivní a rychlé přepravě kusových zásilek. Jejím základem je svoz kusových zásilek od různých odesílatelů do tzv. sběrných center. Zde dochází ke konsolidaci (sdružování) zásilek do celovozových (celolodních atd.). Pak následuje celovozová přeprava do cílového sběrného střediska a následná dekonsolidace těchto zásilek a jejich rozvoz k příjemcům*“. [6 str. 320]

Mezinárodní sběrná služba je pak stejného charakteru s tím rozdílem, že sběrná střediska sídlí na území různých států. Principem sběrné služby je tedy ušetřit společnou vzdálenost, kterou malé zásilky mohou urazit společně a jinak by musely putovat odděleně na různých vozidlech.

1.2.1 Limity sběrné služby

Z toho důvodu je sběrná služba vždy vázána limity hmotnosti a objemu. Např. firma DHL má tyto limity do 3000 kg a 12 m3 [8], firma Geis 2 500 kg a 10 m3 [9]. Mnohdy se také využívají limity délkové, případně přímé omezení pouze na zásilky rozměru europalety. Je to z toho důvodu, že např. dlouhá zásilka o rozměrech 600 x 30 x 30 cm a hmotnosti 100 kg sice objemově do sběrné služby spadá, ale po jejím naložení již nelze prostor kamionu využít plnohodnotně.

1.3 Zasílatelství a SBS

Zasílatel je subjekt obstarávající dopravní služby. Toto provádí s vynaložením odborných znalostí ze všech oblastí přepravy a dalších souvisejících okruhů. Přepravu obstarává buď zprostředkovaně nebo s využitím vlastních kapacit. V prvním případě se jedná o tzv. **čistého zasílatele**, ve druhém o **zasílatele s vlastním vstupem**. [6 str. 18]. Zasílatel může konsolidovat malé zásilky do větších celků nebo přepravovat jednotlivě. Obvykle je schopen organizovat celý přepravní proces včetně zajištění dokladů, dokumentace, skladování nebo celního řízení. [10 str. 7]

Z těchto důvodů je sběrná služba běžně zajišťována právě zasílatelským způsobem. Je to výhodné, protože tím zasílatel přenáší riziko nízké poptávky na dopravce a svými vozidly zajišťuje pouze nejfrekventovanější linky s nízkou fluktuací zásilek. Také obvykle bývá majitelem sběrných středisek (HUBů), kde se provádí konsolidace zásilek.

1.4 LTL a SBS

LTL přeprava a sběrná služba nutně neznamená totožné termíny. **LTL přeprava** se používá pro přepravu menších nákladů, které nevyplní celou nákladní plochu, ale jsou přepravovány společně s náklady od dalších zákazníků. Tyto náklady jsou vloženy do stejného nákladního prostoru a doručovány společně, v ideálním případě ke stejnému příjemci, ale téměř nikdy se takový záměr nepodaří. Namísto toho je obvykle příjemce dokládky pouze na trase původní zásilky. Také u těchto druhů přeprav zcela chybí sběrné středisko. Dopravce se k hledání LTL přeprav uchyluje zejména v případě, kdy je nucen doplnit ložnou plochu kamionu. Může to být z různých důvodů na straně odesílatele, např. nebyl vyroben dostatek zboží pro FTL přepravu, nebo z důvodů ekonomických – LTL přepravy jsou obecně lépe ceněné a sestavit trasu kamionu z několika LTL přeprav je výhodnější než naložit jedinou FTL přepravu.

Zasílatel pracující na principu sběrné služby namísto toho prvoplánově operuje s co největším počtem LTL zásilek. Zásilky sdružuje na lokálních spádových depech, kde jsou následně buď doručovány rozvozovými vozidly k příjemcům nebo roztříděny a zpracovány pro převoz k bližšímu depu a přímému doručení.

1.5 Dispečer LTL a SBS

Akademický slovník cizích slov definuje dispečera jako odborného pracovníka řídícího ústředně chod provozu dopravy, přísunu surovin, energie apod. [11 str. 170]

Již konkrétního dispečera nákladní dopravy lze ale dále dělit na několik dalších profesí. Je proto nutné rozlišovat mezi:

a) **Dispečerem kamionů,**
b) **Dispečerem celovozových přeprav a**
c) **Dispečerem sběrné služby.**

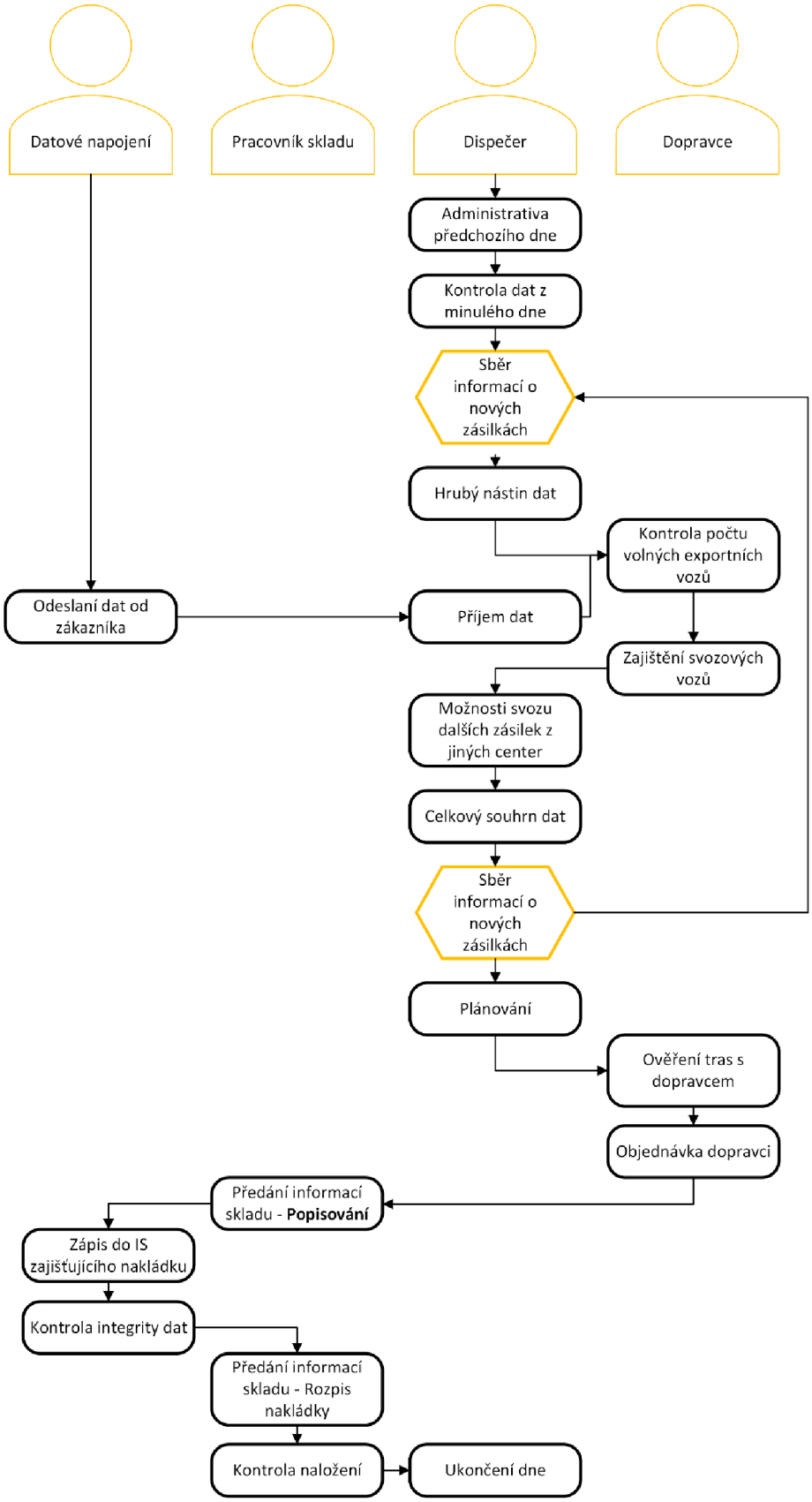
Ad a) **Dispečer kamionů** má obvykle na starost vozidla v počtu jednotek až nižších desítek, kterým na trhu s přepravami zajišťuje adekvátní vytížení. Jeho přesná pracovní náplň záleží na nastavení firmy. Podle počtu vozidel ve firmě bývají dispečeři rozděleni na ty, kteří zajišťují exportní přepravy, další, například s jiným jazykovým vybavením zajišťují importní přepravy anebo jsou v kontaktu s řidiči a představují prostředníka v komunikaci mezi dispečery a řidiči apod.

Ad b) **Dispečerem celovozových přeprav** se v některých firmách rozumí spíše disponent sjednaných pravidelných zakázek, který na tyto zakázky hledá odpovídající volné vozidlo z portfolia dopravců. V tomto případě se může jednat o čistého zasílatele. Takový dispečer poptává a následně uskutečňuje zadané hotové přepravy zboží za určitých podmínek o určitém objemu a termínovém zadání.

Ad c) Naproti tomu pro **dispečera SBS** tato činnost představuje pouze část jeho pracovní náplně. Základní rozdíl obou představuje fakt, že dispečer SBS má na počátku své pracovní činnosti k dispozici pouze souhrn informací o zásilkách připravených k expedici. Za využití těchto informací pak pomocí analytických metod a svých zkušeností ze zásilek nejprve sestaví jednotlivé trasy, a následně vytvoří podklady pro nakládku těchto tras na kamiony. Teprve poté přichází ona práce poptávací a zajišťovací, typická pro předchozí typ dispečera, kdy vytvořeným trasám vyhledá vhodné vozidlo z portfolia dopravců. K této analytické práci tvorby vlastních přeprav určených k prodeji, je v práci odkazováno jako k plánování.

2 Informační systém práce dispečera SBS

Už V. G. Korolev ve své příručce pro dispečera automobilové dopravy z roku 1954 uvádí [12 str. 6]: „*Dispečerská kancelář sestavuje plán přepravy tak, aby se co nejvíce využilo vozidel. Zejména věnuje pozornost součiniteli využití jízd. Při sestavování plánovaných úkolů jednotlivých vozidel dbá na to, aby jízdy na sebe navazovaly a aby se co nejvíce omezily prázdné jízdy.*“ Práce dispečera ve sběrné službě je v tomhle ohledu zaměřená na maximální využití nejen vozidel, ale i všech zásilek. Pro názornost je vzorový informační systém takového dispečera popsán na Obr. 2.1.



Obr. 2.1 - IS dispečera SBS

Zdroj: Vlastní zpracování 2023

Administrativa typická pro dispečera sběrné služby je standardní. Zahrnuje fakturaci, avizaci zákazníkům o nadcházejícím doručení, zpracovávání dat o doručení jednotlivých zásilek, zpracovávání reportů, atp.

Zcela klíčové pro maximální využití všech vozidel a hladkou práci dispečera sběrné služby je spolehlivé **datové napojení**, ať už zprostředkované systémem EDI nebo jakýmkoliv médiem. Účinnost dispečerovy práce je přímo úměrná době předstihu a kvalitě informací, které od zákazníka o přepravovaných zásilkách získá.

Tyto informace jsou všem dispečerům důvěrně známé a nezbytné k efektivní práci. Absenci některých informací lze za určitých okolností zákazníkovi prominout, např. kontakt na příjemce není nutný, dokud na místě doručení nedojde k problému. Jsou to např.:

- zákaznická reference (číslo zásilky, objednávky apod.),
- nadřazená kategorie (např. číslo přepravy),
- datum přijetí zásilky do přepravy (od kdy je zásilka k dispozici),
- stát příjemce, název firmy, ulice, město, směrovací číslo,
- váha zásilky, rozměr, typ balení, plátce přepravy a podobně.

Datové napojení dispečer neustále ověřuje, kontroluje, jestli nedošlo ke změně, navýšení objemů apod. Na základě toho průběžně rozhoduje o počtu vozidel nutných pro uspokojení požadavků v daném dni. Jak bylo uvedeno, dispečer operuje s nejvyšším možným počtem LTL zásilek. Proto je jeho snahou zajistit přístup k jakýmkoliv dalším vhodným zásilkám na ostatních depech firmy, popř. zákazníků a **zajištění jejich svozu**, pokud je to výhodné.

Jakmile je k dispozici pevný datový základ daného dne, dispečer plánuje zásilky do tras ve snaze o maximální využití časových a ekonomických kapacit řidiče. Hotové trasy přenáší na jednotlivé zásilky **předáním informace pracovníkovi skladu**, který zásilky třídí a připravuje je k nakládce. I jeho snahou je maximální využití kapacit nakládaného vozidla s ohledem na bezpečnost. Konečně probíhá **kontrola integrity dat** a stavu nakládky. Jakékoliv odchylky je nutno vyřešit rychle a efektivně.

2.1 Způsob předání a zpracování dat

Požadavek na míru efektivity předání dat roste s objemem zásilek. FTL dispečer dokáže celkem efektivně pracovat i s daty předanými v telefonickém hovoru. „*Odkud, kam, kdy, kolik toho je a kdo to bude platit.*“. Získané informace si zapíše a přepravu zajistí. Tento způsob je nicméně pro SBS nevhodný přímo úměrně počtem zásilek.

2.1.1 Data ve fyzické formě

Co se týče formy, v praxi je možno narazit na různé pokusy o práci s daty pro potřeby SBS ve fyzické, papírové, podobě.

22.03.2023 List nakládky 21.09.2020 [redacted] 1

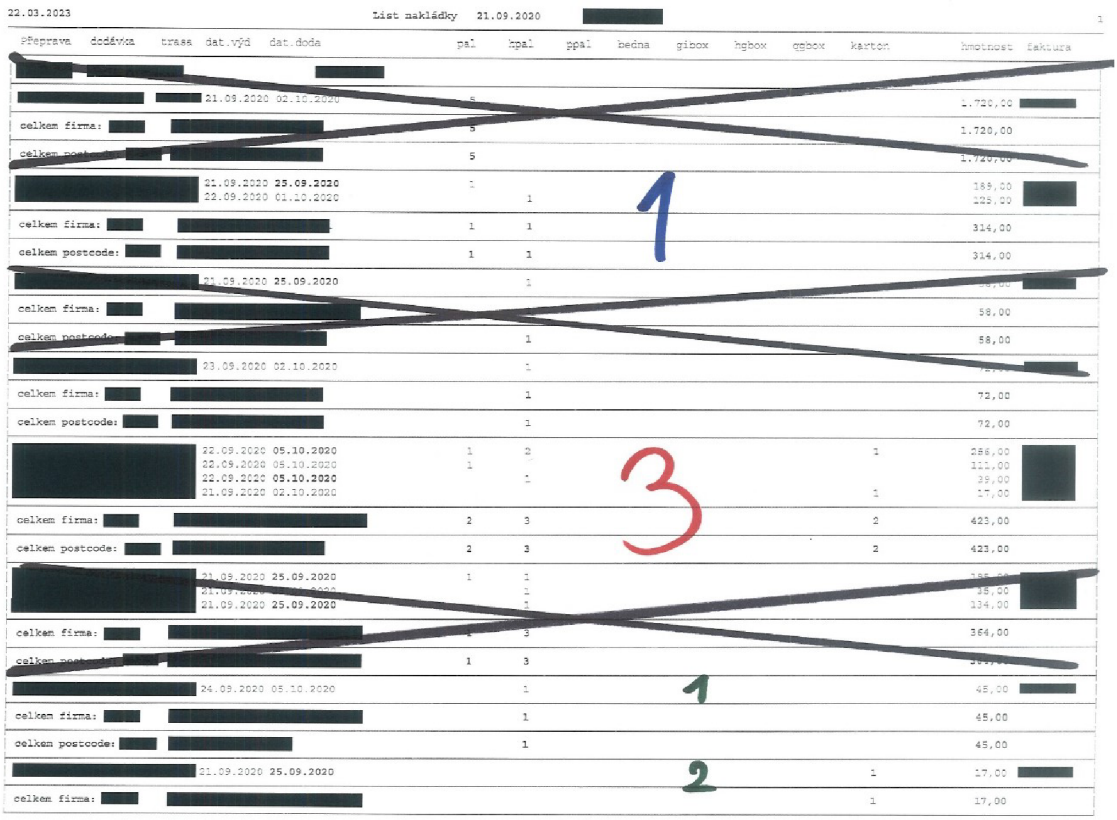
Přeprava	dodávka	trasa	dat.výd	dat.doda	pal	hpal	ppal	bedna	gibox	hgbox	qgbox	karton	hmotnost	faktura
[redacted]														
[redacted]			21.09.2020	02.10.2020	5								1.720,00	[redacted]
celkem firma:	[redacted]				5								1.720,00	
celkem postcode:	[redacted]				5								1.720,00	
[redacted]			21.09.2020	25.09.2020	1								189,00	[redacted]
			22.09.2020	01.10.2020		1							125,00	
celkem firma:	[redacted]				1	1							314,00	
celkem postcode:	[redacted]				1	1							314,00	
[redacted]			21.09.2020	25.09.2020		1							58,00	[redacted]
celkem firma:	[redacted]					1							58,00	
celkem postcode:	[redacted]					1							58,00	
[redacted]			23.09.2020	02.10.2020		1							72,00	[redacted]
celkem firma:	[redacted]					1							72,00	
celkem postcode:	[redacted]					1							72,00	
[redacted]			22.09.2020	05.10.2020	1	2						1	256,00	[redacted]
			22.09.2020	05.10.2020	1								111,00	
			22.09.2020	05.10.2020		1							38,00	
			21.09.2020	02.10.2020								1	17,00	
celkem firma:	[redacted]				2	3						2	423,00	
celkem postcode:	[redacted]				2	3						2	423,00	
[redacted]			21.09.2020	25.09.2020	1	1							195,00	[redacted]
			21.09.2020	25.09.2020		1							35,00	
			21.09.2020	25.09.2020		1							134,00	
celkem firma:	[redacted]				1	3							364,00	
celkem postcode:	[redacted]				1	3							364,00	
[redacted]			24.09.2020	05.10.2020		1							45,00	[redacted]
celkem firma:	[redacted]					1							45,00	
celkem postcode:	[redacted]					1							45,00	
[redacted]			21.09.2020	25.09.2020								1	17,00	[redacted]
celkem firma:	[redacted]											1	17,00	

Obr. 2.2 - List nakládky 1

Zdroj: [13]

Na Obr. 2.2 je názorně vidět použití fyzické formy dat. Tento objem zásilek představuje jeden celek, předem určený zákazníkem, v tomto případě se celek nazývá „Přeprava“. Tato konkrétní přeprava obsahuje přes sto zásilek tímto způsobem rozepsaných na 5 stranách A4, s daty seřazenými vzestupně podle směrovacího čísla příjemce, přičemž každá firma a každé směrovací číslo představuje blok, který je zvlášť sečten, počty kusů a váhy se tedy několikrát opakují, primárním klíčem je pouze číslo zásilky. Pokud je tedy

dispečerovým záměrem zadat příkaz k převezení těchto zásilek v rámci SBS na jiné, bližší depo, těchto pět stran podkladů vytiskne, na první stranu nadepíše číslo nebo kód vozidla a předá pracovníkovi nakládky. Ten podle podkladů zásilky na skladě najde a později nakládá. Při práci s přepravou obsahující desítky zásilek se jedná o rychlou a jednoduchou práci s relativně přehledně uspořádanými podklady.



Obr. 2.3 - List nakládky 2
Zdroj: [13]

Praxe sběrné služby však nezbytně obsahuje dělení velkého celku, běžně stovek zásilek, na menší kusy a jejich optimální analýzu a následné rozdělení na rozvozová vozidla podle výsledku této analýzy. Proto v naprosté většině případů nedochází k nakládce 1:1 tak, jak jsou zásilky naskladněny, ale k pečlivému výběru jednotlivých kusů. Tento výběr ilustruje Obr. 2.3. Dispečer tímto dává skladu najevo, že si přeje první zásilku vážící 1720 kg vynechat, druhou zásilku označit modrou jedničkou, pravděpodobně proto, že jde o první nakládkovou pozici kamionu, předem označeného jako modrý. Sklad palety vyhledá, dle předlohy je popíše a roztřídí. Jde o velmi levný způsob třídění zásilek bez nutnosti pořizování složitější techniky a je vhodný pro menší provozy.

Nepřímo to ale znamená, že někde na dispečinku existuje i negativ tohoto rozpisu, kde zásilka 1720 kg vynechaná není, aby se na ni nezapomnělo a aby byla později naplánována do jiného kamionu, např. následující den. Proto se tyto podklady k přepravám musí vytisknout minimálně tolikrát, kolik je v daný den k dispozici rozvozových tras a spotřeba kancelářského materiálu tím lineárně roste. Dispečer také z důvodu kontroly musí po vyškrtání zásilek z podkladů provést několik kontrolních součtů pozitivu i negativu a celkový součet porovnat s celkovým součtem přepravy, aby se ujistil, že na žádnou zásilku nezapomněl nebo ji nepřehlédl, a teprve poté začne přidělovat čísla kamionů. Pro názornost byla záměrně vložena chyba u zásilky č. 4, 72 kg (Obr. 2.3, řádek 14). Linka diagonálního přeškrtnutí je o jeden řádek posunuta dolů a je tak velmi špatně odhalitelné, že tato zásilka není ani přeškrtnutá, ani přidělená, k čemuž by nemělo docházet. Proto se jedná o metodu s vysokou potenciální chybovostí.

2.1.2 Tabulková kalkulace

Tabulková kalkulace je naproti tomu pro SBS do určitého počtu zásilek velmi vhodná. Zejména z důvodu možnosti opakovaného slučování a rozlučování zásilek, čímž tento způsob poskytuje dispečerovi možnost lépe si představit kombinace různých tras, a tak lépe plánovat. Na Obr. 2.4 lze vidět výřez z tabulkové kalkulace. Zde řádek představuje jeden kus zásilky, číslo zásilky i číslo přepravy je dobře viditelné a možnosti různých zpracování jsou oproti papírové formě naprosto nesrovnatelné.

	A	B	C	D	E	F	G	H	I	
1	Přeprava	Datum odeslání	Dodávka	Závod	Typ balení	Rozměr	Barva	Stát	PSČ	Jméno
2	2000788107	08.02.2023	70028152	2	XXXXXX	XXX X XXX XX	ZELENÁ	XX	29	XXXXXXX XXXXX
3	2000854645	23.06.2023	70893050	1	XXXXXX	XXX X XXX XX		XX	84	XXX XXXXXXX X
4	2000303544	20.04.2023	70223539	1	XXXXXX	XXX X XXX XX		XX	101	XXXX XXX
5	2000159206	13.04.2023	70792241	1	XXXXX	XXX X XXX X XXX XX		XX	136	XXXXXXXXXXXX
6	2000067941	06.01.2023	70672002	2	XXXXXX	XXX X XXX XX	MODRÁ	XX	338	XXXXXXX XXXXX
7	2000115133	05.04.2023	70660221	1	XXXXX	XXX X XXX X XXX XX		XX	374	XXXXXXXXXXXX
8	2000472293	02.05.2023	70336153	1	XXXXX	XXX X XXX X XXX XX		XX	780	XXXXXXXXXXXX
9	2000532047	20.05.2023	70369004	1	XXXXX	XXX X XXX X XXX XX		XX	1371	XXXXXXXXXXXX
10	2000640645	07.05.2023	70172547	1	XXXXXX	XXX X XXX XX	ČERVENÁ	XX	1636	XXXXXXX XXXXX
11	2000952485	12.06.2023	70128904	1	XXXXXX	XXX X XXX XX	ZELENÁ	XX	1816	XXXX XXX
12	2000346825	07.05.2023	70148405	1	XXXXX	XXX X XXX X XXX XX		XX	1855	XXXXXXXXXXXX
13	2000237032	27.03.2023	70136449	1	XXXXXX	XXX X XXX XX	MODRÁ	XX	2009	XXXXXX XXXXX
14	2000994249	21.06.2023	70448140	1	XXXXXX	XXX X XXX XX	MODRÁ	XX	2506	XXXXXXX XXX X
15	2000005303	11.01.2023	70073954	2	XXXXXX	XXX X XXX XX	MODRÁ	XX	2769	XXXXXXX XXXXX
16	2000756216	18.03.2023	70609781	1	XXXXXX	XXX X XXX XX	ZELENÁ	XX	3247	XXXXXXX XXXXX
17	2000430756	10.04.2023	70151448	2	XXXXXX	XXX X XXX XX	ČERVENÁ	XX	3407	XXXXXXX XXXXX
18	2000380153	08.01.2023	70464302	2	XXXXXX	XXX X XXX XX	MODRÁ	XX	3452	XXXXXXX XXXXX
19	2000667259	08.04.2023	70161226	2	XXX	XXX X XXX XX	ČERVENÁ	XX	3588	XXXXXX XXXX X
20	2000419766	21.05.2023	70782887	1	XXXXXX	XXX X XXX XX	MODRÁ	XX	3831	XXXXXXXXXXX
21	2000547405	24.04.2023	70690499	2	XXXXXX	XXX X XXX XX	ČERVENÁ	XX	4165	XX XXXXXXXXX
22	2000010194	24.03.2023	70215870	1	XXXXXX	XXX X XXX XX	MODRÁ	XX	5194	XXXXXXX XXXXX
23	2000984236	04.04.2023	70998142	1	XXXXXX	XXX X XXX XX		XX	5537	XXXXXXXXXX X
24	2000610134	05.04.2023	70869495	1	XXXXXX	XXX X XXX XX		XX	5615	XXX XXXXXX XX
25	2000769415	25.03.2023	70499915	2	XXXXXX	XXX X XXX XX	ZELENÁ	XX	5629	XXXXXXX XXXXX
26	2000083641	26.05.2023	70888503	1	XXXXXX	XXX X XXX XX	MODRÁ	XX	5834	XXXX XXX

Obr. 2.4 - Tabulková kalkulace

Zdroj: vlastní zpracování 2023

2.2 Ruční zpracování

Jednoduchost práce s daty v MS Excel vychází už z jeho podstaty. Data lze jakkoliv řadit, filtrovat, aplikovat funkce apod. Výčet možných operací by byl přinejmenším vyčerpávající.

Je velmi složité exaktně změřit, kolik času vyžaduje vytvoření fyzických podkladů pro nakládku kamionu bez použití tabulkové kalkulace, protože se jedná o celodenní práci několika dispečerů. Nicméně v MS Excel taková sestava, ke které je později odkazováno jako k „Popisování“ zabere ručně asi 4 minuty včetně tisku a předání skladníkovi. Při použití VBA se pro stejnou operaci jednalo o 3,309 sekundy.

3 VBA a jeho možnosti

3.1 MS Excel

Historie Excelu sahá až do roku 1961, kdy byla poprvé vyslovena myšlenka o tabulkovém kalkulátoru profesorem Richardem Mattesichem. Jako první byl v roce 1978 na americké Harvard Business School vyvinut VisiCalc pro tehdy velmi úspěšný Apple II. Ve stejné době byl ale představen IBM PC a VisiCalc tak postupně ustoupil do pozadí za Lotus 1-2-3. Ten již zvládal práci s tabulkami, grafy a databázovými funkcemi. Také jako první představil záznam maker, čímž sklidil velký úspěch. Vývojáři Lotus 1-2-3 ale věřili, že operační systém OS/2 nakonec nahradí Windows. To se zřejmě nestalo, a ačkoliv proběhla snaha o migraci, Microsoft v té době už uvedl svou sadu MS Office a úspěch Lotusu byl nenávratně pryč. [14 str. 20]

3.2 VBA

VBA – Visual Basic for Applications byl vytvořen v roce 1993 a poprvé představen v programu Excel 5.0. Vznikl v roce 1993 jako nástupce jazyka Microsoft Basic (MS-BASIC) a byl součástí verze Microsoft Office, která byla vydána v roce 1995. V současné době prostupuje všemi aplikacemi MS Office a je velmi oblíbený hlavně kvůli své technické nenáročnosti. [15]

3.2.1 Základní vlastnosti jazyka VBA

VBA, jakožto objektově orientovaný jazyk, je primárně založen na objektech. Existuje ale i několik dalších základních vlastností, které je třeba zmínit [16] [17]:

- Objekty – např. soubory Excelu (`Workbook`), listy (`Worksheet`), buňky (`Range`), formulářové prvky (`Controls`), tabulky (`ListObject`) a další. Každý objekt má své vlastnosti a metody,
- Moduly, procedury a funkce – VBA kód je organizován do modulů, což jsou samostatné soubory kódu. Procedury a funkce pak provádějí určitou, přesně danou sekvenci kroků, daných uživatelem. Dále,

- Proměnné – VBA umožňuje pracovat s proměnnými, což jsou místa v paměti počítače, kam se ukládají data. Proměnné mohou mít různé datové typy, jako jsou čísla, text, datum a čas, true-false hodnoty atd.,
- Podmínky a cykly – VBA umožňuje používat podmínky (`If...Then`) pro rozhodování o tom, jaký kód se má vykonat na základě určité podmínky a cykly, tedy opakování určitého kódu, dokud, případně pokud je splněna určitá podmínka,
- Události – VBA umožňuje zachytávat události (`Event`), jako jsou kliknutí na tlačítko, změny hodnoty v buňce, načtení souboru atd., a provádět určitou akci na základě této události,
- Třídy – umožňují definovat vlastní objekty s vlastnostmi, metodami a událostmi. Tím umožňují organizovat kód do logických celků, což zvyšuje jeho čitelnost a znovupoužitelnost, a další.

3.3 Práce s daty pomocí VBA

3.3.1 Import a úprava dat

Předpokladem úspěšné práce s daty ve VBA je jejich normalizace. Pro účely této práce je předpokládáno, že dispečer má k dispozici jistý základní sešit MS Excelu s podporou maker (*.xlsm), který tak plní funkci určitého programového prostředí. Obsahuje datový list, ve kterém jsou shromažďována data o jednotlivých zásilkách, odehrává se zde samotné plánování a další listy podle zájmů a potřeb dispečera – ceníky, transportní časy, přehledy jednotlivých kamionů s jejich trasami apod. Přímo datový list bude pravděpodobně velmi připomínat zdrojová data. Jeho možné uspořádání lze najít v Příloha A, list Data.

V souvislosti s importem dat lze použít v pořadí třetí způsob předání a zpracování dat v rámci SBS, a to pomocí vestavěného nástroje v MS Excel, „Záznam maker“. Dispečer předepsanou rutinu zaznamená a bude mít možnost ji spouštět stále znovu a znovu dle potřeby. Tento způsob nevyžaduje žádnou znalost VBA a dá se použít celkem efektivně, pokud zdroj a výstup zůstávají neměnné a jedná se pouze o jednoduché procesy, jako je změna pořadí sloupců, úprava formátu apod.

Je ale potřeba mít na paměti, že uživatel tím dosáhne pouze statického zpracování. Pokud je přesouván sloupec, záznamem maker je označena určitá oblast, např. sloupec

a ta přesunuta, bez ohledu na to, jak se sloupec jmenuje, jestli jsou v oblasti vůbec data apod. Navíc je zaznamenáno i rolování v tabulce, aktivace listů apod., což je pro provedení programu zbytečné a pro jeho chod zatěžující. Z tohoto důvodu je vhodné využít čtvrtou, poslední metodu, a tou je kód VBA.

3.3.2 Řazení a formátování

Jedním z nejjednodušších procesů, které lze ve VBA naprogramovat a které dopomáhají plánování, je automatické řazení. Dispečer jednou potřebuje zboží seřadit podle směrovacího čísla, protože má zjistit základní strukturu zásilek daného celku, případně kolik aut bude zhruba potřeba vybavit na daný směr. Následně ale potřebuje data rychle seřadit podle zákaznické reference a jména příjemce, aby zjistil, že se někde nezmýlil nebo když potřebuje vypsat pořadí nakládky.

```
If Not ActiveSheet.Name Like "Data *" Then Exit Sub

    With ActiveSheet.Sort.SortFields
    .Clear
    .Add Key:=Columns(10), SortOn:=xlSortOnValues, Order:=xlAscending 'PSČ
    .Add Key:=Columns(11), SortOn:=xlSortOnValues, Order:=xlAscending 'Jméno příjemce
    .Add Key:=Columns(1), SortOn:=xlSortOnValues, Order:=xlAscending  'Číslo Přepravy
    End With

    With ActiveSheet.Sort
    .SetRange ActiveSheet.UsedRange
    .Header = xlYes
    .MatchCase = False
    .Orientation = xlTopToBottom
    .SortMethod = xlPinYin
    .Apply
    End With
```

Kód VBA 3.1 – Řazení
Zdroj: vlastní zpracování 2023

Kód VBA 3.1 je velmi jednoduchý a přehledný, navíc s pojistkou na začátku, že je makro spouštěno na správném listu. I tento kód je ale statického typu. Není nijak ošetřeno pořadí sloupců, programátor při psaní takového kódu nijak neošetřuje možnost, že ve sloupci 10 bude vždy opravdu poštovní směrovací číslo a ve sloupci 11 jméno příjemce. Proto je vhodnější makro zafixovat nikoliv na polohu, ale na název sloupce a vytvořit tím makro dynamické, jak ukazuje Kód VBA 3.2.

```
Public Sub tlRazeniPlanDyn(control As Object)

On Error Resume Next
If Not ActiveSheet.Name Like "Data*" Then Exit Sub

    With ActiveSheet.Sort.SortFields
    .Clear
    .Add Key:=sloupec("PSČ"), SortOn:=xlSortOnValues, Order:=xlAscending
    .Add Key:=sloupec("Jméno"), SortOn:=xlSortOnValues, Order:=xlAscending
    .Add Key:=sloupec("Přeprava"), SortOn:=xlSortOnValues, Order:=xlAscending
    End With

    With ActiveSheet.Sort
    .SetRange ActiveSheet.UsedRange
    .Header = xlYes
    .MatchCase = False
    .Orientation = xlTopToBottom
    .SortMethod = xlPinYin
    .Apply
    End With

End Sub
```

```
Private Function sloupec(zahlavi As String) As Range

Dim rngSloupec As Range
Set rngSloupec = ActiveSheet.Rows(1).Find(What:=zahlavi, LookIn:=xlFormulas, _
lookat:=xlWhole)

If Not rngSloupec Is Nothing Then
    Set sloupec = rngSloupec.EntireColumn
Else
    MsgBox "V záhlaví nebyl nalezen řetězec " & zahlavi & ".", vbCritical
End If

End Function
```

Kód VBA 3.2 - Řazení (dynamicky)

Zdroj: vlastní zpracování 2023

Nejprve je nutné vytvořit privátní funkci `sloupec` s parametrem `zahlavi`. Touto funkcí je prohledáno záhlaví listu a pokud není zaznamenána žádná chyba, resp. proměnná `rngSloupec` nenabývá hodnoty `Nothing`, výsledkem funkce je odpovídající sloupec. Tímto je ošetřena změna pořadí sloupců, stále je ale potřeba zachovat jejich neměnný název. V případě, že se v datech vyskytne jiný než očekávaný název sloupce, např. vinou neúmyslného smazání obsahu buňky v záhlaví nebo změny zdrojových dat, je zapotřebí oprava parametru funkce „sloupec“. Toho lze dynamicky dosáhnout také, např. analýzou buněk pod daným sloupcem, jestli jejich obsah skutečně odpovídá tomu, co je předpokládáno (např. PSČ má pro konkrétní státy určité charakteristiky, 5 čísel apod.). Popř. lze vytvořit zvláštní sešit s „nastavením“, ve kterém by byly předem vypsány platné názvy sloupců a tím pádem by uživatel nemusel zasahovat

do kódu. Pro účely této práce je ale předpokládáno, že dispečer je zároveň programátorem a není tedy nutné tvořit makra ve verzích pro zcela neznalé uživatele.

Ve chvíli, kdy jsou zásilky rozplánovány do tras, v datových listech je zaneseno určité rozdělení zboží do jednotlivých, předem daných kamionů na základě struktury zásilek (jako byla u fyzických podkladů zaznačena modrá jednička). Následně je tedy zapotřebí předat tyto informace pracovníkům skladu. Pracovník pak ke každé jednotlivé zásilce následně přistoupí a na základě přijatých dat paletu označí, aby bylo možno je začít třídit a naložit do kamionů v pořadí daném dispečerem. Pro tyto účely bylo vytvořeno makro **Popisování**.

3.3.3 Dynamická oblast

Aby byla práce se sloupci rychlá a efektivní, je však nejprve vhodné správně zvolit metodu zpracování oblasti. Obecně při práci s daty uspořádanými jako seznam položek (1 řádek = 1 kus) jsou klíčové dvě základní struktury. První je dynamické určení oblasti buněk. Toho lze ve VBA dosáhnout několika způsoby:

1) Metoda `UsedRange` objektu `ActiveWorksheet`,
2) Metoda `CurrentRegion` objektu `Cells`,
3) Vytvořit a dále upravovat `ListObject` a jeho metodu `DataRange` nebo
4) VBA zcela vynechat a vytvořit Název v seznamu vzorců, odkazující na oblast za pomoci funkce `POSUN` a `POCET2`: `=POSUN („Název listu“! $A$1; 0; 0; POČET2 („Název listu“! $C:$C); „statický počet sloupců oblasti“)`

Každý z těchto přístupů má své chyby, které je potřeba vzít v potaz a každý jinak nakládá s nečekanými událostmi, například když list neobsahuje žádné buňky, uživatel smaže záhlaví, data z jednoho sloupce apod.

3.3.4 Konkrétní data na základě názvu sloupce

Druhou základní strukturou je přístup k jednotlivým buňkám dané oblasti v rámci běhu makra dynamicky, s proměnlivým názvem sloupce. Jinými slovy – jaká hodnota odpovídá aktuálně procházenému řádku, ale v jiném sloupci, např. „Číslo zásilky“? Toho se využívá zejména, pokud je cílem makra procházení hodnot, úprava formátu na základě nějaké podmínky, kontrola vstupních dat apod., což lze prakticky využít např. když:

- Podle toho, jestli zásilka přesahuje hmotnost 2,5t je potřeba ji zařadit do jednoho nebo druhého způsobu přepravy nebo
- Výpočet předpokládaného data doručení závisí na geografické pozici zásilky ke konečnému příjemci nebo
- Výpočet ceny přepravy zásilky, přičemž určení ceníku závisí na některém z dat v zásilce, např. váze nebo objemu.

Využití je tedy velmi široké a jednoduchost tohoto přístupu k jednotlivým hodnotám procházených řádku pak přímo ovlivňuje rychlosti průběhu celého makra, protože je zpravidla prováděno v cyklech.

Co do způsobu zpracování lze uvést následující 3 metody.

Metoda „Sloupec“

Metoda sloupec, jak bylo uvedeno výše, je vhodná pro dynamické určení sloupce podle jeho měnícího-se názvu. Po doplnění o metodu `Intersect` (Kód VBA 3.3) lze určit buňku odpovídající aktuálně zpracovávané buňce, ovšem v odlišném sloupci. Tato buňka je představována proměnnou `cell` s parametrem `optional`, aby byla metoda nadále funkční i pro vyhledávání sloupců.

```
Public Function sloupec(sh As Worksheet, zahlavi As String, _
Optional cell As Range) As Range

Dim rngSloupec As Range
Set rngSloupec = sh.Rows(1).Find(What:=zahlavi, _
LookIn:=xlFormulas, lookat:=xlWhole)

If Not rngSloupec Is Nothing Then
    If Not cell Is Nothing Then
        Set sloupec = Intersect(rngSloupec.EntireColumn, cell.EntireRow)
    Else
        Set sloupec = rngSloupec.EntireColumn
    End If
Else
    MsgBox "V záhlaví nebyl nalezen řetězec " & zahlavi & ".", vbCritical
    Set sloupec = Nothing
End If

End Function
```

Kód VBA 3.3 - Sloupec + Intersect
Zdroj: vlastní zpracování 2023

Metoda „ListObject“

Jak název napovídá, metoda je založena na objektu ListObject, česky Tabulka. Zde je tudíž klíčové zvolit pro zpracování dynamické oblasti dat tabulku (Kapitola 3.3.3 –

Dynamická oblast). Následně je pak v metodě využito toho, že na jednotlivé sloupce v tabulce se lze doptávat přímo podle jejich názvu, není nutné používat funkci Sloupec. Samotný kód pro další zpracování v makru pak probíhá pomocí dvou pomocných uživatelsky definovaných vlastností h a r („hodnota“ a „range“), viz Kód VBA 3.4.

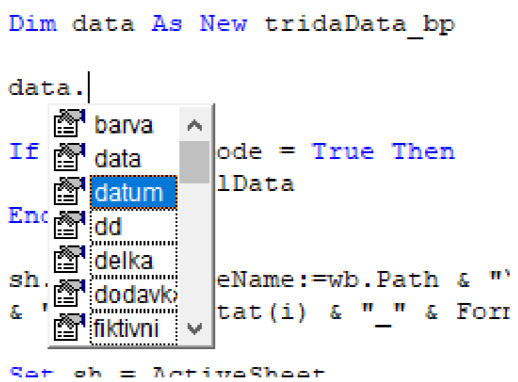
```
Private Property Get h(nazevSl As String) As String
    h = data.DataBodyRange(row.Index, data.ListColumns(nazevSl).Index).Value
End Property

Private Property Let h(nazevSl As String, ByVal vNewValue As String)
    data.DataBodyRange(row.Index, data.ListColumns(nazevSl).Index).Value = vNewValue
End Property

Private Function r(nazevSl As String) As Range
    Set r = data.DataBodyRange(row.Index, data.ListColumns(nazevSl).Index)
End Function
```

Kód VBA 3.4 - Vlastní funkce metody ListObject

Zdroj: vlastní zpracování 2023

Pomocí vlastnosti h lze tedy získat hodnotu typu `String`, nebo ji do buňky přímo zapsat. Je tak učiněno pomocí vlastnosti index nalezeného sloupce a předem určeného řádku `Row` typu `ListRow`, který je předpokládán jako privátně určený pro daný modul, lze ho ale samozřejmě doplnit do vlastností a určit spolu s názvem sloupce (`nazevSl as String`, `row as ListRow`). V tomto konkrétním nastavení je předpokládáno využití této metody uvnitř cyklu `For Each Row in (…) Next Row`.

Metoda „Třídy“



Obr. 3.1 - Třídy

Zdroj: vlastní zpracování 2023

Použití tříd vyžaduje již mírně pokročilejší znalost VBA a jeho naprogramování není zcela intuitivní pro uživatele, který se v minulosti ještě nesetkal s objektově orientovaným programováním. Nakonec je však uživatelsky příjemné i funkční. Třída je zadávána jako

nový objekt, který následně v kontextové nabídce uživateli nabízí předvyplněné vlastnosti dané třídy. Pokud jsou tedy takto vytvořeny jednotlivé vlastnosti, pro každý sloupec jedna, a proměnné typu `Range`, také jednotlivě, dochází k určení všech sloupců a přiřazení hodnot pouze jednou, při události `Class_Initialize` (při prvotním přístupu k třídě). V dalším zpracování už dochází pouze k vyvolávání hodnot těchto proměnných. Kód celé třídy k dispozici v Příloha B.

3.4 Časová náročnost metod

Každá z těchto metod přistupuje k datům jinak. Proto bylo provedeno testování, které pomůže určit, jaká z metod je vhodná, pro jaký počet dat. Zvolena byla jednoduchá časová metoda pomocí funkce `Timer` přímo ve VBA, jde tedy o testování časové náročnosti algoritmu.

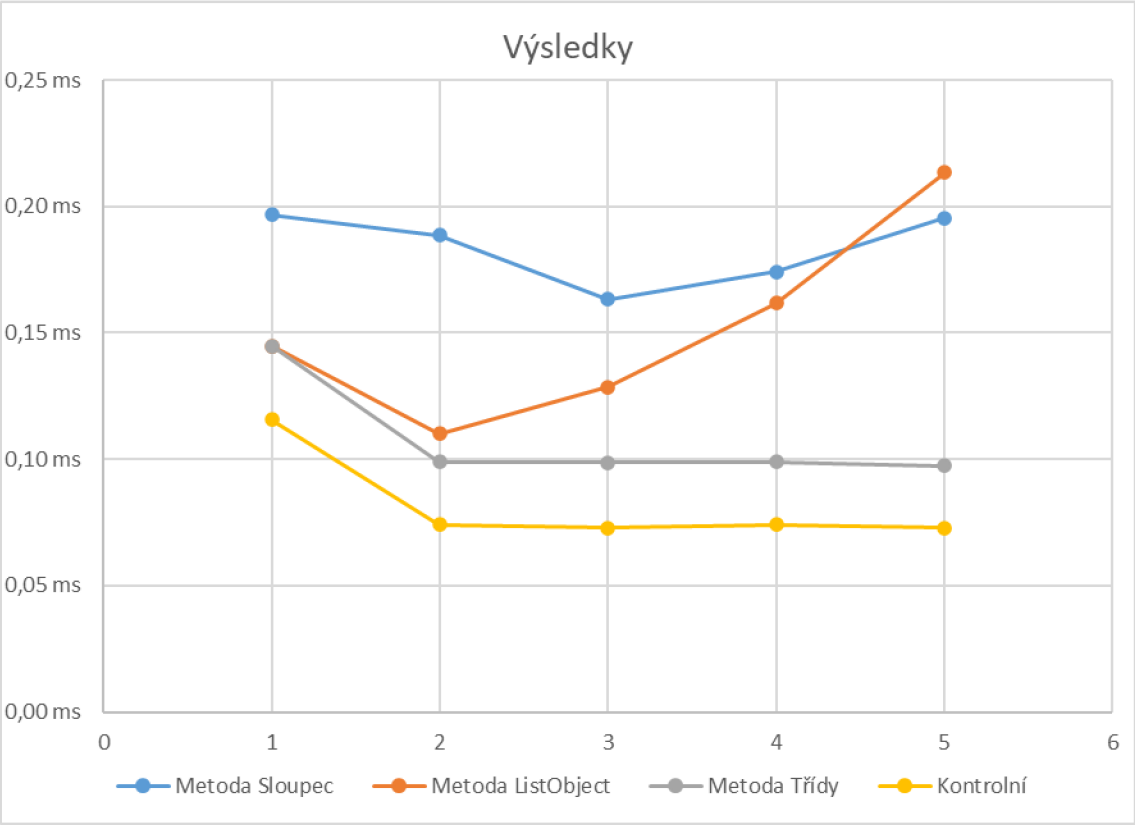
Ve zdrojích nalezneme definici, že ***Časová náročnost algoritmu*** *vyjadřuje, jak se čas potřebný k provedení výpočtu mění v závislosti na velikosti vstupních dat* [18]. Je tedy pochopitelné, že při exaktním časovém testování je nutné zvolit jinou metodu, protože bude probíhat na různých počítačích různě. Záleží na momentálním zatížení počítače, nastavení výkonu operačního systému, pomocných, například baterii šetřících, funkcích apod., které omezují výkon procesoru. Jak uvádí Mareš: „*Postup se stopkami v ruce je z teoretického hlediska nevhodný*“ [19 str. 92]. Zde ovšem není testován program jako takový, ale pouze porovnávány metody mezi sebou, a proto je pro tyto účely časové testování na jednom počítači dostačující.

3.4.1 Průběh testování

Pro testování třech výše zmíněných metod bylo nejprve vytvořeno 5 datových listů, pojmenovaných podle počtu zásilek v nich obsažených (100, 1000, 2500, 5000 a 10000). Aby byla nejlépe otestována rychlost zpracování, každou metodou byl dynamicky určen sloupec a hodnota každé jeho buňky v daném datovém listu nakopírována do listu DataZapis. Fakticky tedy byla jedna buňka po druhé nakopírována do jiného listu a sledována doba potřebná pro tento úkol.

Každé makro bylo spuštěno třikrát v rámci smyčky `For Each` a výsledky byly automaticky zapsány do tabulky. Z těchto výsledků byl následně určen medián. Čtvrtá metoda byla kontrolní, která pouze zapisovala hodnotu „1“ do každé buňky bez

jakéhokoliv čtení dat z jiného listu. Tento celý proces byl následně opět opakován třikrát a z této série třech kompletních výsledků opět určen medián. **Výsledky tedy reprezentují střední hodnoty času v milisekundách potřebných pro zápis jedné buňky zvolenou metodou**. Celý sešit testování společně s kódem metod, spouštěcí konzolí a zdrojovými daty je k dispozici v Příloha C.



Graf 3.1 - Výsledky měření
Zdroj: vlastní zpracování 2023

3.4.2 Interpretace

Na základě výsledků v Graf 3.1 lze konstatovat, že použití metody Sloupec je nejpomalejší a nejvíce zatěžuje chod excelu jak po výpočetní, tak po funkční stránce. Nutno zmínit, že při jejím zpracování také docházelo k nespecifikovaným náhodným prodlevám výpočtu při různých pokusech v řádech jednotek sekund. Naopak nejefektivnější je použití metody Třídy. Za zmínku stojí, že ačkoliv se může metoda `ListObject` jevit jako elegantní, protože fakticky ze sešitu nečte žádná data, pouze

nahlíží do vlastností již definovaného objektu, její náročnost roste úměrně počtem zpracovaných dat. V počtu odpovídajícím přibližně 190 tis. zpracovaných buněk dokonce v jednom z testování dobou zpracování překonala metodu Sloupec, která při každém čtení dat nahlíží do sešitu metodou `Find`. Zajímavostí také je, že kdyby metoda Sloupec nebyla spouštěna společně s příkazem `Application.ScreenUpdating = False`, který zabraňuje aktualizaci obrazovky při průběhu maker, doba zpracování by byla přibližně 4x delší. Zpracování 2500 řádků dosahovalo výsledků srovnatelných s předchozím zpracováním 10000 řádků. Výsledky viz Příloha C, list „Vysledky visible“.

3.5 Popisování

Jakmile je určena vhodná metoda pro dynamickou oblast a čtení dat, je ještě nutné pohovořit s pracovníky skladu a zjistit, jaké informace o zásilce potřebují vědět k jejímu efektivnímu nalezení a popsání, tedy, jak je potřeba, aby vypadal reálný výsledek. Následně už lze naprogramovat kompletní proces, díky kterému lze sloupce uspořádat do požadované podoby, data seřadit, upravit pro tisk, vyexportovat do PDF a odeslat emailem. Kód celé procedury viz Příloha D.

Přeprava	PSČ	Dodávka	Pořadí	Typ balen	Stát	Jméno	Místo	Váha	MJ
2000005303	2769	70073954	3	XXXXXX	XX	XXXXXXX XXXXXXX XXX	XXXXXXX	7	KG
2000010194	5194	70215870	3	XXXXXX	XX	XXXXXXX XXXXXXX XXX	XXXXX	30	KG
2000013174	18463	70437224	3	XXXXXX	XX	XXXXXXX XXXXXXX XXX	XXXXXXXXXXXXX	4	KG
2000027810	5855	70631924	3	XXXXXX	XX	XXXXXXXX XXXXXXX XX	XXXXXXX	23	KG
2000032967	9650	70557298	3	XXXXXX	XX	XXXXXXX XXXXXXX XXX	XXXXX	11	KG
2000067941	338	70672002	3	XXXXXX	XX	XXXXXXX XXXXXXX XXX	XXXXXXXXXX	2	KG
2000073401	14770	70858975	3	XXXXXX	XX	XXXXX XXXXX XXX	XXXXXX	79	KG
2000079828	7524	70576434	3	XXXXXX	XX	XXXXXXX XXXXXXX XXX	XXXXX XXXXXXXX	1	KG
2000083641	5834	70888503	3	XXXXXX	XX	XXXX XXX	XXXXXXXXXX	3	KG
2000125278	36975	70442653	3	XXXXXX	XX	XXXXX XXXXXXXXX X XXXXX	XXXXXX XXX XXXXXX XX	14	KG
2000142845	33272	70921243	3	XXXXXX	XX	XXXXX XXXXX XXXX	XXXXX	275	KG
2000172063	17050	70549735	3	XXXXXX	XX	XXXXX XXXXXX	XXXXXXXXXXX XXXXXX	0	KG
2000535603	13992	70367274	1	XXXXXX	XX	XXXXXXX XXXXXXX XXX	XXXXXXXXX	36	KG
2000547405	4165	70690499	1	XXXXXX	XX	XX XXXXXXXXX XXX	XXXXXX	14	KG
2000597250	9401	70418773	1	XXXXXX	XX	XXXXXXX XXXXXXX XXX	XXXXX	2	KG
2000609911	37193	70734808	1	XXXXXX	XX	XX XXXXXXXXX XXXXXX	XXXXXX XXXX	0	KG
2000635356	14871	70012159	2	XXX	XX	XXXXX XXXX XXX	XXXXXXXX	165	KG
2000640645	1636	70172547	2	XXXXXX	XX	XXXXXXX XXXXXXXXX XX	XXXXXXXXXX	1	KG
2000667259	3588	70161226	2	XXX	XX	XXXXX XXXX XXX	XXXXXXXX	28	KG
2000672300	19084	70425337	2	XXXXXX	XX	XXXXXXXXXXXXXXX XXXX	XXXXXXXXXXX	0	KG
2000744025	6068	70251316	3	XXXXXX	XX	XXXXXXX XXXXXXX XXX	XXXXX	1	KG
2000756216	3247	70609781	3	XXXXXX	XX	XXXXXXX XXXXXXX XXX	XXXXX	153	KG
2000769415	5629	70499915	3	XXXXXX	XX	XXXXXXX XXXXXXX XXX	XXXXXX	15	KG
2000788107	29	70028152	3	XXXXXX	XX	XXXXXXX XXXXXXX XXX	XXXXXXXXXXX	21	KG
2000800665	33331	70770907	2	XXXXXX	XX	XXXXX XXXXX	XXXXXXXX XXXXXXXX	40	KG
2000920304	14888	70547495	3	XXXXXX	XX	XXXXXXXXXXXXXX XXXX	XXXXXXXXXXX	4	KG
2000928638	33742	70092267	1	XXXXXX	XX	XXXXX XXXXXX XXXXXXX	XXXXXXX XXX	112	KG
2000952485	1816	70128904	1	XXXXXX	XX	XXXX XXX	XXXXX XXXXX XXXXX	25	KG
2000968830	8199	70222429	2	XXXXXX	XX	XXXXX XXXXXXX XXX	XXXXX XXXXXXX	1	KG
2000969505	29644	70703593	2	XXXXXX	XX	XXXX XXXXX	XXXXX XX XXXX XX	1	KG
2000979051	31693	70702559	2	XXXXXX	XX	XXXXX XXXXX	XXXXXXXX XXXXXXXX	0	KG
2000994249	2506	70448140	1	XXXXXX	XX	XXXXXX XXX XXX	XXXXX	7	KG

Celkem 55 colli / 1981 kg

Obr. 3.2 - Výstup Popisování

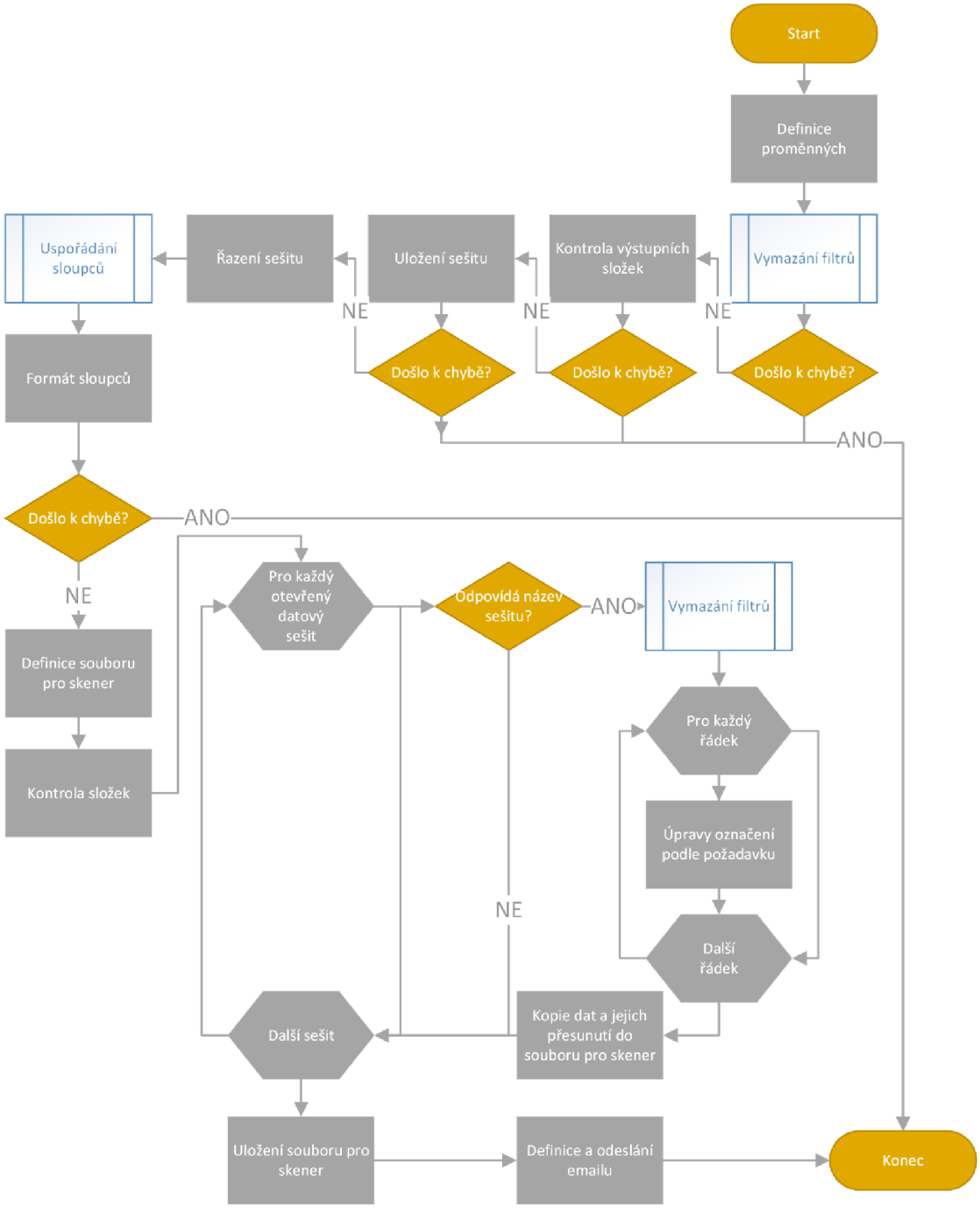
Zdroj: Vlastní zpracování 2023

Výsledkem Popisování je pak výstup ve formátu PDF, přiložený k emailové zprávě, seřazený a upravený podle požadavků pracovníka skladu (Obr. 3.2).

3.6 Skener

Toto provedení, v případě většího počtu zásilek, ale opět představuje výše zmíněné nevýhody práce s daty ve fyzické podobě – pracovník skladu dostane do rukou sice poněkud redukované, ale stále fyzické podklady, takže hledá konkrétní zásilky pouze vizuálně, zjišťuje, jakou barvou a číslem je potřeba je označit apod., což opět představuje potenciálně velké riziko lidské chyby. Proto byla ve snaze o eliminaci těchto nedostatků naprogramována druhá verze Popisování s názvem **Skener**, spočívající v úpravě dat do podoby vhodné pro funkci SVYHLEDAT a souboru tak, aby byl co nejmenší a jednoduše spustitelný na stanici například Zebra MC330M. Protože tato stanice obsahuje operační systém Android, dá se na do ní nainstalovat Microsoft Office. Pracovníci na ní následně otevírají výstupní soubor makra (Příloha E) a každou zásilku skenují za pomoci softwaru DataWedge od firmy Zebra, defaultně přítomný ve stanici. Protože ale Office pro Android nepodporuje makra, zobrazení konkrétní informace je provedeno pouze pomocí funkce SVYHLEDAT a podmíněného formátování, jak lze vidět na Obr. 3.3.

Obr. 3.3 - Popisování ve skeneru



Obr. 3.4 – Flow chart – Skener

Vývojový diagram na Obr. 3.4 ukazuje průběh makra Skener. Před každým blokem makra je vyhodnoceno, jestli při jeho běhu nedošlo k chybě a v takovém případě je ukončeno, přičemž informace o chybě je zobrazena v okně běhu programu, vytvořeném pomocí uživatelských formulářů. Kompletní kód makra Skener v Příloha D, výstupní soubor v Příloha E.

3.7 Funkcionality procesu Skener

Jak lze vidět v Příloha D, ve Skeneru byla použita již zmíněná makra pro řazení dat a také zpracování dynamické oblasti, konkrétně metoda ListObject, která nebyla vyhodnocena jako velmi vhodná pro velké množství dat, ale přesto byla použita. Je to z toho důvodu, že v makru je častěji přistupováno k oblasti sloupců než ke konkrétním buňkám a metoda ListObject je rychlejší než metoda Sloupec. Další funkcionality zmíněné ve vývojovém diagramu jsou:

3.7.1 Vymazání filtru

```
Public Sub vymazatFiltry(sh As Worksheet)

On Error Resume Next
sh.ShowAllData
sh.ListObjects(1).AutoFilter.ShowAllData
Err.Clear

End Sub
```

Kód VBA 3.5 - vymazatFiltry

Velmi jednoduchá a užitečná procedura k vymazání filtrů u konkrétního sešitu. Ve VBA je možné se na různé druhy chyb různě pojistit. Ověřováním dat, dotazováním se na typ proměnné apod. Mnohdy je ale užitečnější pracovat s příkazem `On Error Resume Next`. Procedura tak provede dva způsoby, jakými lze na daném listu zrušit automatický filtr dat, a to bez ověřování, který přístup je v daný okamžik správný. Programátor tak nemusí vytvářet podmínky, aby ověřil, který příkaz má aktuálně použít, aby běh programu nezpůsobil chybu. Chyba se jednoduše automaticky ignoruje.

3.7.2 Práce se složkami

Denně probíhajícím procesem, nutným zmínit ve snaze o zefektivnění práce dispečera, je automatická práce se složkami. V Kód VBA 3.6 je uvedeno, jak lze jednoduše vytvářet základní pracovní složky. Obvykle se jedná o základní složku s názvem aktuálního data a v ní několik pracovních složek pro daný den:

```
Dim cestazaklad As String
cestazaklad = ThisWorkbook.Path & "\" & Format(Now, "ddmmyyyy")

Dim cesta(1 To 4) As String
cesta(1) = cestazaklad & "\AVIZACE"
cesta(2) = cestazaklad & "\POPIS"
cesta(3) = cestazaklad & "\ROZPIS"
cesta(4) = cestazaklad & "\CELNI"

Dim i As Integer

If Dir(cestazaklad, vbDirectory) = Empty Then
        MkDir cestazaklad
    For i = 1 To 4
        MkDir cesta(i)
    Next i
Else
For i = 1 To 4
    If Dir(cesta(i), vbDirectory) = Empty Then
        If MsgBox("Složka " & Split(cesta(i), "\")( _
        WorksheetFunction.CountA(Split(cesta(i), "\")) - 1) & _
        " není vytvořena. Přejete si ji vytvořit?", vbYesNo) = vbYes _
        Then MkDir cesta(i)
    End If
Next i
End If
```

Kód VBA 3.6 - Základní složky

Zdroj: vlastní zpracování 2023

Důležitá je zde kontrola existence základní složky. Pokud složka neexistuje, je vytvořena nejprve ona a poté až jednotlivé pracovní složky, aby nedošlo k chybě. Pokud alespoň jedna z pracovních složek chybí, zobrazí se dotaz a teprve na základě potvrzení se složka dotvoří. Uživatel ji mohl smazat z nějakého důvodu a nemusí si ji přát vytvořit znovu. V makru Skener je tato funkcionalita použita pro kontrolu existence složky SKEN, do které se následně ukládají výstupní soubory.

3.7.3 Uspořádání sloupců

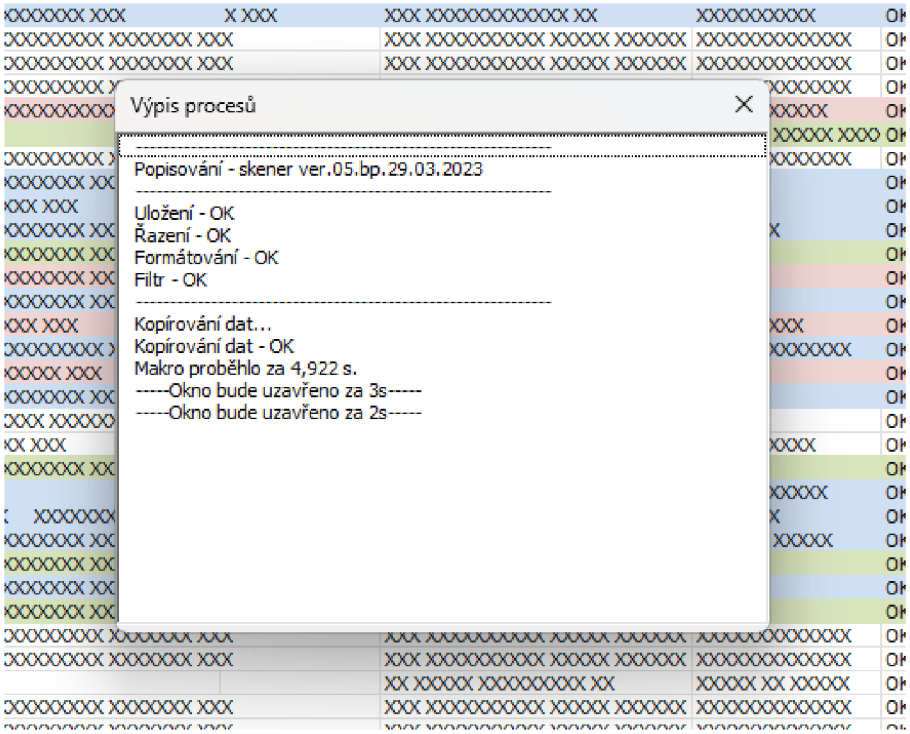
```
Public Sub usporadaniSloupcu(sh As Worksheet, poradi As Variant, _
deleteOstatnich As Boolean)

Dim sl As Range
Dim i As Integer, pocetSl As Integer

sh.UsedRange.EntireColumn.Hidden = False
pocetSl = UBound(poradi)

For i = pocetSl To 1 Step -1
    If poradi(i) = Empty Then
        Application.CutCopyMode = False
        sh.Columns(1).Insert
    Else
        Set sl = sh.Rows(1).Find(What:=poradi(i), LookIn:=xlFormulas, _
        lookat:=xlWhole).EntireColumn
        sl.Cut
        sh.Columns(1).Insert Shift:=xlToRight
    End If
Next i

If deleteOstatnich = True Then
    Range(sh.Columns(pocetSl + 1), sh.Columns(sh.Columns.Count)).Delete
End If

End Sub
```

Kód VBA 3.7 - usporadaniSloupcu

Tato procedura je použita hlavně ke zjednodušení kódu. Jak název napovídá, s její pomocí jsou řazeny sloupce určeného listu `sh` podle pořadí uloženém v proměnné `poradi`, u kterého se předpokládá, že se bude jednat o pole textových hodnot názvů uspořádávaných sloupců, proto typ `Variant`. A to tím způsobem, že je postupováno zprava doleva, kopírován poslední sloupec v seznamu a přesunován na první pozici v listu. Zároveň je proces doplněn i o boolean hodnotu rozhodující o ponechání nebo odstranění všech ostatních sloupců, kromě přenesených.

3.7.4 Výpis procesů – AVLog

Jak si lze všimnout, v kódu se také vyskytuje několik samostatných řádků, jako například `AVlog.progress "Kopírování dat - OK"`. Jedná se o proces uživatelsky vytvořeného formuláře s prvkem `ListBox`, jehož účelem je zpravování uživatele o průběhu makra.



Obr. 3.5 – AVLog

Je to velmi užitečný nástroj, který rovněž makru přidává na jisté profesionalitě. Lze do jeho kódu také vhodně zakomponovat `Timer` a sledovat tak dobu průběhu makra v reálném čase, stejně jako průběh chyb, pokud je tomu uzpůsoben kód makra. Jeho užití je ovšem v MS Excel poměrně složité. VBA totiž obecně nedovoluje zasahovat do kódu v průběhu makra, ani ho jinak přerušovat nebo pozastavovat. Makro lze pozastavit, ovšem pouze na přesně definovaný čas a jediným způsobem, jak lze uživatelsky do kódu zasáhnout je pomocí okna `InputBox`, případně klávesovou zkratkou `Ctrl + Shift + Break`, pomocí které je makro zcela přerušeno v jeho průběhu. Proto je pro správnou funkčnost takového výpisu procesů nutné nejprve vytvořit veřejnou proměnnou, zde nazvanou `logSub`. Této proměnné je následně přiřazena hodnota názvu spouštěného makra a poté vyvolán AVLog pomocí příkazu `Show`. Při jeho inicializaci jsou spuštěny předem definované příkazy (Kód VBA 3.8), jež vyvolají `Timer`, požadované makro a v tomto makru pak následně pomocí procesu `progress` zápisy do `ListBoxu`. Žádné makro, jehož průběh je takto sledován, tedy není spouštěno standardní cestou, ale pouze přes AVLog a proměnnou `logSub`.

```
Private Sub UserForm_Activate()

Dim tStart As Double: tStart = Timer
    If logSub = Empty Then
        Unload Me
    Else
        Run logSub
    End If
Dim tPrubeh As Double: tPrubeh = Round(Timer - tStart, 3)
progress "Makro proběhlo za " & tPrubeh & " s."

Application.Wait (Now + TimeValue("0:00:01"))
progress "-----Okno bude uzavřeno za 3s-----"
Dim t As Integer: t = 2
    Do Until t = -1
        Application.Wait (Now + TimeValue("0:00:01"))
        progress "-----Okno bude uzavřeno za " & t & "s-----"
        DoEvents
        t = t - 1
    Loop
Unload Me

End Sub
```

```
Public Sub progress(text As String)

log1.AddItem text
log1.TopIndex = log1.ListCount - 1
DoEvents

End Sub
```

Kód VBA 3.8 - AVLog + progress

3.8 Další vhodná makra

VBA lze využít k velkému počtu i jiných kancelářských operací, které mohou být pro dispečera velmi vhodné. Níže uvedeny příklady takových.

3.8.1 Záloha

Jakákoliv forma zálohy aktuálního projektu se dříve nebo později prokáže být velmi užitečným nástrojem pro každého, kdo pracuje s MS Office.

```
Private Sub Workbook_BeforeClose(Cancel As Boolean)

    Dim cesta As String: cesta = ThisWorkbook.Path & "\ZÁLOHA\"
    Dim ArrCesta() As String: ArrCesta = Split(ThisWorkbook.Path, "\")

    If ArrCesta(WorksheetFunction.CountA(ArrCesta) - 1) = "ZÁLOHA" _
    Then Exit Sub

    If Dir(cesta, vbDirectory) = Empty Then
        MkDir (cesta)
    End If

    Application.DisplayAlerts = False
    ActiveWorkbook.SaveCopyAs Filename:=cesta & "PLAN_" & _
    Format(Now, "ddmmyyyy_hhmm") & ".xlsm"

End Sub
```

Kód VBA 3.9 - Záloha

Zdroj: vlastní zpracování 2023

Protože je makro prováděno před zavřením sešitu, je vloženo do modulu „`ThisWorkbook`“, události „`BeforeClose`“. Samotná automatická záloha je ovšem pouze poslední řádek s metodou `SaveCopyAs`. Předchází jí dvě důležité pojistky. První pojistkou je ověřeno, že sešit je hlavním sešitem. Pokud by byla otevřena samotná záloha, při svém uzavření by vytvářela zálohu sama na sebe, což není žádoucí. Funkcí Split je z tohoto důvodu rozdělena cesta spouštěného sešitu a zjištěno, zda se jedná o sešit ze složky záloh a v tom případě se program ukončí. Druhou pojistkou je ověřena existence kořenové složky záloh a v případě její neexistence je vytvořena funkcí `MkDir`.

Existenci nadřazené složky je možno potvrdit i jinými způsoby než metodou Split, např. metodou `GetParentFolderName` objektu `FileSystemObject`. Výsledek této metody je ale i nadále potřeba upravovat, případně dělit podle obrácených lomítek, takže způsob, jak provést výpis cesty bez vytváření nového objektu, je přinejmenším jednodušší.

3.8.2 Automatické emaily a úkoly MS Outlook

Každý dispečer, ať už FTL, LTL či SBS, pošle denně desítky emailových zpráv. A většinou velmi obdobných. Rozesílání administrativy, objednávek dopravcům, faktur, avíz zákazníkům apod. Proto je vhodné využívat možnosti VBA také v MS Outlook, který stejně jako MS Excel disponuje editorem kódu, bohužel ale ne záznamem maker.

```
Dim cesta As String: cesta = "C:\Users\public\Documents\"

If Dir(cesta & "obj.pdf") = Empty Then
    MsgBox "Soubor ""obj.pdf"" není vytvořen", vbCritical
    Exit Sub
Else
    Dim soubor As String: soubor = cesta & "obj.pdf"
End If

Dim zprava As MailItem: Set zprava = CreateItem(olMailItem)

With zprava
.To = "Email dopravce"
.Subject = "Objednávky " & Date
.HTMLBody = "Dobrý den,<br><br>posílám objednávky.<br><br>" & _
"<font color=""red"", size=4><b>Prosím o potvrzení do " & _
Time + TimeSerial(1, 0, 0) & ".</b></font>" & _
"<br><br>Děkuji za zpracování,<br><br>R.<br>"

.Attachments.Add (soubor)
.Display
End With

ukolComplete ("Objednávky")
```

Kód VBA 3.10 - Automatické emaily

Zdroj: vlastní zpracování 2023

Pomocí makra v Kód VBA 3.10 je vytvořen email s přílohou souboru s objednávkou. Nejprve je zkontrolována existence souboru ve zvolené složce, následně vytvořena emailová zpráva a v textu této zprávy požádáno o potvrzení objednávky za 1 hodinu od odeslání, což je zvýrazněno tučně a červeně. Zde je nutné zdůraznit, že formátování zprávy v MS Outlook vyžaduje elementární znalosti jazyka HTML a vkládá se do atributu `HTMLBody`.

Tento postup lze následně dobře kombinovat i s dalšími makry, např. pro vytváření úkolů.

```
Dim ukol As TaskItem: Set ukol = Application.CreateItem(olTaskItem)
ukol.Subject = "Objednávky"
ukol.StartDate = Date
ukol.Body = "Odeslat objednávky"
ukol.DueDate = Date
ukol.Save
```

Kód VBA 3.11 - Automatické úkoly

Zdroj: vlastní zpracování 2023

Takto je možno jednoduše vytvořit makro, pomocí kterého bude denně vytvářen např. seznam pravidelných úkolů. Pokud by bylo potřeba úkol splnit, kód by vypadal následovně.

```
Private Sub ukolComplete(ukolName As String)

Dim ukoly As Items
Dim ukol As TaskItem

Set ukoly = GetNamespace("MAPI").GetDefaultFolder(olFolderTasks).Items.Restrict( _
"[StartDate] = '" & Format(Date, "dd.mm.yyyy") & "'" & " And " _
& "[DateCompleted] <> '" & Format(Date, "dd.mm.yyyy") & "'")

For Each ukol In ukoly
    If ukol.Subject = ukolName Then
        ukol.MarkComplete
        MsgBox "Úkol " & ukolName & " je splněn", vbInformation
        Exit For
    End If
Next ukol

End Sub
```

Kód VBA 3.12 - ukolComplete

Zdroj: vlastní zpracování 2023

Tímto makrem (Kód VBA 3.12) jsou prolistovány všechny úkoly v daném intervalu vytvořeném metodou `Restrict`. Intervalem jsou úkoly omezeny, aby data načtená makrem nebyla příliš velká. V tomto případě se tedy jedná o úkoly dnes vytvořené a dnes ještě nesplněné. Následuje funkce `For Each`, kterou je prověřen název úkolu a pokud odpovídá hledanému, je označen jako splněný a uživateli se zobrazí okno s informací.

V Kód VBA 3.10, je viditelné, že je makrem zobrazen email a pokud existuje úkol s názvem „Objednávky“, je označen jako splněný. Toto provedení se ale po několika použitích může začít jevit jako nedokonalé. Pokud není email okamžitě odeslán metodou `Send`, ale uživatel ho chce nejdřív zkontrolovat, používá metodu `Display`. Úkol je tak následně označen jako splněný bez ohledu na fakt, jestli uživatel skutečně email odeslal nebo ne, což není zcela žádoucí. Úplné funkčnosti lze docílit pomocí uživatelsky definované události.

```
Private WithEvents Items As Outlook.Items

Private Sub Application_Startup()
  Set Items = GetNamespace("MAPI").GetDefaultFolder(olFolderOutbox).Items
End Sub

Private Sub Items_ItemAdd(ByVal email As Object)
    If email.Subject = "Objednávky " & Date Then
        ukolComplete ("Objednávky")
    End If
End Sub
```

Kód VBA 3.13 - Splnění úkolů

Kód VBA 3.13 je následně vkládán do hlavního modulu `ThisOutlookSession`, z toho důvodu je také potřeba proces `ukolComplete` převést na `Public`, aby byl přístupný i mimo daný modul. Tímto nastavením je dosaženo toho, že je proces vyvolán a úkol splněn až ve chvíli, kdy se email s požadovaným názvem skutečně objeví ve složce odeslané pošty.

3.8.3 Čtení PDF souborů

Užitečnou zajímavostí, které lze s pomocí VBA dosáhnout, je čtení obsahu PDF souborů v rámci makra. Na zajímavosti jí přidává to, že Adobe Reader nepatří do rodiny MS Office, a nemělo by s ním tedy VBA vůbec spolupracovat. Firma Adobe sice nabízí plugin pro spolupráci s VBA, který si lze nainstalovat, tento plugin je ale k dispozici pouze v placené verzi Adobe Acrobat Pro.

```
Public Function pdf_hledani(objWord As Object, hlStr As String, _
pdfCesta As String) As String

    Dim objDoc As Object
    Set objDoc = objWord.Documents.Open(Filename:=pdfCesta, _
    Format:="PDF Files", ConfirmConversions:=False)

    Dim pg As Object
    Dim wLine As String

    For Each pg In objDoc.Paragraphs
        wLine = pg.Range.text
            If InStr(wLine, hlStr) > 0 Then
                wLine = Mid(wLine, InStr(1, wLine, hlStr))
                wLine = Replace(wLine, hlStr, "")
                wLine = Trim(WorksheetFunction.Clean(wLine))
                pdf_hledani = wLine
                Exit For
            End If
    Next pg

    objDoc.Close False
    Set objDoc = Nothing

End Function
```

Kód VBA 3.14 - Hledání v PDF

V tomto kódu je využito skutečnosti, že soubory PDF lze otvírat v MS Word. Jde tedy o otevření PDF souboru a prohledání dokumentu po jednotlivých odstavcích. Jakmile dojde k nalezení, cyklus se ukončí. Výsledkem je slovo následující po hledaném textu, očištěné o netisknutelné znaky a mezery.

Takové makro lze využít při avizaci zásilek zákazníkovi, kde je například úkolem odeslat seznam zásilek směřující k příjemci ve formátu PDF a každý takový soubor obsahuje také

SPZ vozidla, které zásilky veze, o které si příjemce přeje být informován přímo v těle emailové zprávy, aby nemusel každý soubor otevírat a SPZ v něm hledat. V takovém případě lze podobným kódem PDF soubory otevřít automaticky a SPZ z nich vypsat přímo do emailové zprávy.

Lze si povšimnout, že objekt `objWord` se nenachází uvnitř funkce. Je to z toho důvodu, že je předpokládáno použití této funkce v nějaké formě cyklu, které prochází soubory např. ve složce a vytváření nového objektu Wordu je v rámci VBA relativně zdlouhavý proces (1-5 sekund). Proto je vhodnější `objWord` vytvořit mimo tento cyklus a uzavřít ho teprve, jakmile dojde ke zpracování všech požadovaných souborů s příponou PDF.

Závěr

Ačkoliv bylo mým hlavním cílem práce představit možnosti využití VBA v práci dispečera, nepřímo jsem poukazoval také na nedostatky nedigitální práce s daty. Snahou každého datového analytika, statistika nebo dispečera, i dalších povolání, by mělo být pracovat rychle, přesně a efektivně. Z toho důvodu jsem tuto práci strukturoval jako postupný přechod od tužky a papíru k dynamické tabulkové kalkulaci s využitím VBA. První den mé praxe dispečera SBS jsem totiž nedostal k dispozici počítač, nýbrž papír, tužku a kalkulačku. A jak jsem se každý pracovní den potýkal s chybovostí této metody, postupně jsem s nadějí začal vyvíjet první makra a přecházet k digitální práci s daty. Proto jsem nyní cítil morální povinnost se tímto tématem ve své první odborné práci zabývat a své zkušenosti předat, v naději o jejich praktické využití.

V této práci jsem proto nejprve přiblížil teoretický základ dopravní logistiky s ohledem na sběrnou službu, rozebral rozdíly mezi jednotlivými dispečery i typy přeprav a postupně představil různé metody i makra pro práci s daty tak, abych ilustroval vhodnost jejich užití. Z toho důvodu jsem také provedl testování, srovnávající různé přístupy pro čtení velkého množství dat a konečně představil několik maker pro práci uvnitř i vně rodiny MS Office pomocí VBA.

Možnosti VBA jsou samozřejmě ještě obsáhlejší, než je nastíněno v této práci. Pomocí zásuvných pluginů a knihoven lze přistupovat k objektům z dalších programů a vytvářet komplexní procesy, bez nadsázky plnohodnotné programové prostředí, jak ilustruje například zpracování známé hry Monopoly pomocí VBA. Do budoucna je však nutné zvážit některé trendy v oblasti Microsoft Office, například přechod na cloudové verze Office 365, čehož v některých firmách již dochází. Taková verze pak pro tvorbu scriptů využívá jazyk TypeScript, připomínající již spíše Javu. Prozatím stále ještě lze v Office 365 využívat VBA, pouze ale v desktopové verzi a zakomponování TypeScriptu se tak může zdát jako možná snaha společnosti Microsoft postupně ustupovat od využívání VBA a přecházet na jiný způsob práce s makry.

Nyní pro svou práci nicméně stále využívám VBA a obdobu datového sešitu z této práce. Makra používám denně k automatizovanému importu do datového sešitu, exportu do jiných sešitů, kontrole vstupních dat, výstupů pro kolegy, reportů pro vedení nebo přímo k plánování vozidel, kdy všechna důležitá data o jednotlivých vozidlech jsou soustředěna

strukturovaně na jednom listu a vytváří tak podobu uživatelského prostředí pro kompletní plánování importní i exportní dopravy jednoho sběrném střediska SBS. VBA tedy dalece přesahuje pouze automatické uspořádávání sloupců a vyplňování vzorců jedním kliknutím. Jedná se o plnohodnotný programovací jazyk, schopný vytvářet složité struktury, a i když lze narazit na jeho limity, stále platí, že jakákoliv práce, kterou lze dělat v Excelu ručně, lze ve VBA udělat rychleji a lépe.

Seznam zdrojů

[1] **STODOLA, Josef, MAREK, Josef a FURCH, Jan.** *Logistika.* Brno : Mendelova zemědělská a lesnická univerzita, 2007. ISBN 978-80-7375-071-8.

[2] **Eurostat.** Statistics | Eurostat. *Eurostat Data Browser.* [Online] Eurostat, 15. Březen 2023. [Citace: 26. Březen 2023.] Cesta: All data; Transport; Multimodal data; Modal split of transport. https://ec.europa.eu/eurostat/databrowser/view/TRAN_HV_FRMOD/default/table?lang=en&category=tran.tran_hv_ms.

[3] **Gros, Ivan.** *Velká kniha logistiky.* Praha : Vysoká škola chemicko-technologická v Praze, 2016. ISBN 978-80-7080-952-5.

[4] **ŠIROKÝ, Jaromír.** *Technologie dopravy. Páté doplněné vydání.* Pardubice : Univerzita Pardubice, 2020. ISBN 978-80-7560-309-8.

[5] **CEMPÍREK, Václav a KAMPF, Rudolf.** *Logistika.* Pardubice : Institut Jana Pernera, 2005. ISBN 80-86530-23-x.

[6] **NOVÁK, Radek.** *Nákladní doprava a zasílatelství, 2. přeprac. vydání.* Praha : ASPI, 2005. ISBN 80-7357-086-6.

[7] —. *Mezinárodní silniční nákladní přeprava a zasílatelství.* Praha : C.H. Beck, 2018. ISBN 978-80-7400-041-6.

[8] **DHL Freight CZ s.r.o.** Přepravní podmínky a rizika paletové přepravy. *Evropská nákladní přeprava | DHL Freight | Česká republika.* [Online] 5. Prosinec 2019. [Citace: 26. Březen 2023.] https://www.dhl.com/content/dam/dhl/local/cz/dhl-freight/documents/pdf/domestic/cz-freight-packing-instructions-ecd-cs.pdf.

[9] **Geis CZ s.r.o.** Všeobecné Obchodní podmínky. [Online] 1. Listopad 2016. [Citace: 27. Březen 2023.] https://www.geis-group.cz/data/images/_orig/6/406.pdf.

[10] **KAMPF, Rudolf a CEMPÍREK, Václav.** *Zasílatelství.* Pardubice : Univerzita Pardubice, 2005. ISBN 80-7194-745-8.

[11] **PETRÁČKOVÁ, Věra a KRAUS, Jiří.** *Akademický slovník cizích slov.* Praha : Academia, 1995. ISBN 80-200-0497-1.

[12] **KOROLEV, V. G.** *Příručka pro dispečera automobilové dopravy.* Praha : SNTL, 1954. cnb000511147.

[13] **SAP ČR, spol. s r.o.** *SAP GUI 7.70.* [software] Praha : autor neznámý, 2023.

[14] **WEBER, Monika a BREDEN, Melanie.** *Excel VBA: velká kniha řešení.* Brno : Computer Press, 2007. ISBN 978-80-251-1453-7.

[15] **Microsoft.** Visual Basic Blog. *Visual Basic Blog.* [Online] Microsoft. [Citace: 26. Březen 2023.] https://devblogs.microsoft.com/vbteam/.

[16] **KRÁL, Martin.** *Excel VBA: výukový kurz.* Brno : Computer Press, 2010. ISBN 978-80-251-2358-4.

[17] **Laurenčík, Marek.** *Excel - pokročilé nástroje : funkce, marka, databáze, kontingenční tabulky, prezentace, příklady.* Praha : Grada, 2016. ISBN 978-80-247-5570-0.

[18] **České vysoké učení technické v Praze.** *CourseWare Wiki.* [Online] 2009. [Citace: 26. Březen 2023.] https://cw.fel.cvut.cz/old/_media/courses/y36alg/2009_sumperk_p10.pdf.

[19] **MAREŠ, Martin a VALLA, Tomáš.** *Průvodce labyrintem algoritmů.* Praha : CZ.NIC, z.s.p.o., 2017. ISBN 978-80-88168-19-5.

Seznam grafických objektů

Obrázky

Kódy VBA

Grafy

Seznam zkratek

EDI	Electronic data interchange, elektronická výměna dat
FTL	Full truck load, celovozová zásilka
HUB	Logistické centrum
LTL	Less than truck load, přikládková zásilka
MS	Microsoft
PDF	Portable document format, formát souborů programu Adobe Reader
SBS	Sběrná služba
VBA	Visual Basic for Applications

Seznam příloh

Příloha A

Datový sešit se všemi procedurami, „datovy_sesit.xlsm“

K dispozici na přiloženém CD.

Příloha B

Zdrojový kód třídy tridaAuta

```
Option Explicit

Private mPreprava As Range
Private mDatum_odeslani As Range
Private mDodavka As Range
Private mZavod As Range
Private mTyp_baleni As Range
Private mRozmer As Range
Private mBarva As Range
Private mStat As Range
Private mPSC As Range
Private mJmeno As Range
Private mPoznamka As Range
Private mUlice As Range
Private mMisto As Range
Private mUI As Range
Private mAP As Range
Private mPM As Range
Private mDatum_dodani As Range
Private mVaha As Range
Private mNetto As Range
Private mMJ As Range
Private mPoradi As Range
Private mDelka As Range
Private mSirka As Range
Private mVyska As Range
Private mMJ2 As Range
Private mObjem As Range
Private mMJ3 As Range

Private sh As Worksheet

Private Sub Class_Initialize()

Set sh = Module1.shD

Set mPreprava = sloupec("Přeprava")
Set mDatum_odeslani = sloupec("Datum odeslání")
Set mDodavka = sloupec("Dodávka")
Set mZavod = sloupec("Závod")
Set mTyp_baleni = sloupec("Typ balení")
Set mRozmer = sloupec("Rozměr")
Set mBarva = sloupec("Barva")
Set mStat = sloupec("Stát")
Set mPSC = sloupec("PSČ")
Set mJmeno = sloupec("Jméno")
Set mPoznamka = sloupec("Poznámka")
Set mUlice = sloupec("Ulice")
Set mMisto = sloupec("Místo")
Set mUI = sloupec("UI")
Set mAP = sloupec("AP")
Set mPM = sloupec("PM")
```

```
Set mDatum_dodani = sloupec("Datum dodání")
Set mVaha = sloupec("Váha")
Set mNetto = sloupec("Netto")
Set mMJ = sloupec("MJ")
Set mPoradi = sloupec("Pořadí")
Set mDelka = sloupec("Délka")
Set mSirka = sloupec("Šířka")
Set mVyska = sloupec("Výška")
Set mMJ2 = sloupec("MJ2")
Set mObjem = sloupec("Objem")
Set mMJ3 = sloupec("MJ3")

End Sub

Private Function sloupec(zahlavi As String) As Range

    On Error Resume Next
    Dim hledani As Range: Set hledani = sh.Rows(1).Find(What:=zahlavi,
LookIn:=xlFormulas, lookat:=xlWhole)

    If hledani Is Nothing Then
        MsgBox "V záhlaví nebyl nalezen řetězec " & zahlavi & ".",
vbCritical
    Else: Set sloupec = hledani.EntireColumn
    End If

End Function

Public Property Get Preprava(Optional cell As Range, Optional
dataRange As Boolean) As Range

    If Not cell Is Nothing Then
        Set Preprava = Intersect(mPreprava, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Preprava = Range(sh.Cells(2, mPreprava.Column),
sh.Cells(sh.UsedRange.Rows.Count, mPreprava.Column))
    Else
        Set Preprava = mPreprava
    End If

End Property

Public Property Get Datum_odeslani(Optional cell As Range, Optional
dataRange As Boolean) As Range

    If Not cell Is Nothing Then
        Set Datum_odeslani = Intersect(mDatum_odeslani,
cell.EntireRow)
    ElseIf dataRange = True Then
        Set Datum_odeslani = Range(sh.Cells(2,
mDatum_odeslani.Column), sh.Cells(sh.UsedRange.Rows.Count,
mDatum_odeslani.Column))
    Else
        Set Datum_odeslani = mDatum_odeslani
    End If

End Property
```

```
Public Property Get Dodavka(Optional cell As Range, Optional dataRange
As Boolean) As Range

    If Not cell Is Nothing Then
        Set Dodavka = Intersect(mDodavka, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Dodavka = Range(sh.Cells(2, mDodavka.Column),
sh.Cells(sh.UsedRange.Rows.Count, mDodavka.Column))
    Else
        Set Dodavka = mDodavka
    End If

End Property
Public Property Get Zavod(Optional cell As Range, Optional dataRange
As Boolean) As Range

    If Not cell Is Nothing Then
        Set Zavod = Intersect(mZavod, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Zavod = Range(sh.Cells(2, mZavod.Column),
sh.Cells(sh.UsedRange.Rows.Count, mZavod.Column))
    Else
        Set Zavod = mZavod
    End If

End Property
Public Property Get Typ_baleni(Optional cell As Range, Optional
dataRange As Boolean) As Range

    If Not cell Is Nothing Then
        Set Typ_baleni = Intersect(mTyp_baleni, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Typ_baleni = Range(sh.Cells(2, mTyp_baleni.Column),
sh.Cells(sh.UsedRange.Rows.Count, mTyp_baleni.Column))
    Else
        Set Typ_baleni = mTyp_baleni
    End If

End Property
Public Property Get Rozmer(Optional cell As Range, Optional dataRange
As Boolean) As Range

    If Not cell Is Nothing Then
        Set Rozmer = Intersect(mRozmer, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Rozmer = Range(sh.Cells(2, mRozmer.Column),
sh.Cells(sh.UsedRange.Rows.Count, mRozmer.Column))
    Else
        Set Rozmer = mRozmer
    End If

End Property
Public Property Get Barva(Optional cell As Range, Optional dataRange
As Boolean) As Range

    If Not cell Is Nothing Then
```

```
        Set Barva = Intersect(mBarva, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Barva = Range(sh.Cells(2, mBarva.Column),
sh.Cells(sh.UsedRange.Rows.Count, mBarva.Column))
    Else
        Set Barva = mBarva
    End If

End Property
Public Property Get Stat(Optional cell As Range, Optional dataRange As
Boolean) As Range

    If Not cell Is Nothing Then
        Set Stat = Intersect(mStat, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Stat = Range(sh.Cells(2, mStat.Column),
sh.Cells(sh.UsedRange.Rows.Count, mStat.Column))
    Else
        Set Stat = mStat
    End If

End Property
Public Property Get PSC(Optional cell As Range, Optional dataRange As
Boolean) As Range

    If Not cell Is Nothing Then
        Set PSC = Intersect(mPSC, cell.EntireRow)
    ElseIf dataRange = True Then
        Set PSC = Range(sh.Cells(2, mPSC.Column),
sh.Cells(sh.UsedRange.Rows.Count, mPSC.Column))
    Else
        Set PSC = mPSC
    End If

End Property
Public Property Get Jmeno(Optional cell As Range, Optional dataRange
As Boolean) As Range

    If Not cell Is Nothing Then
        Set Jmeno = Intersect(mJmeno, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Jmeno = Range(sh.Cells(2, mJmeno.Column),
sh.Cells(sh.UsedRange.Rows.Count, mJmeno.Column))
    Else
        Set Jmeno = mJmeno
    End If

End Property
Public Property Get Poznamka(Optional cell As Range, Optional
dataRange As Boolean) As Range

    If Not cell Is Nothing Then
        Set Poznamka = Intersect(mPoznamka, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Poznamka = Range(sh.Cells(2, mPoznamka.Column),
sh.Cells(sh.UsedRange.Rows.Count, mPoznamka.Column))
    Else
```

```
        Set Poznamka = mPoznamka
    End If

End Property
Public Property Get Ulice(Optional cell As Range, Optional dataRange
As Boolean) As Range

    If Not cell Is Nothing Then
        Set Ulice = Intersect(mUlice, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Ulice = Range(sh.Cells(2, mUlice.Column),
sh.Cells(sh.UsedRange.Rows.Count, mUlice.Column))
    Else
        Set Ulice = mUlice
    End If

End Property
Public Property Get Misto(Optional cell As Range, Optional dataRange
As Boolean) As Range

    If Not cell Is Nothing Then
        Set Misto = Intersect(mMisto, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Misto = Range(sh.Cells(2, mMisto.Column),
sh.Cells(sh.UsedRange.Rows.Count, mMisto.Column))
    Else
        Set Misto = mMisto
    End If

End Property
Public Property Get UI(Optional cell As Range, Optional dataRange As
Boolean) As Range

    If Not cell Is Nothing Then
        Set UI = Intersect(mUI, cell.EntireRow)
    ElseIf dataRange = True Then
        Set UI = Range(sh.Cells(2, mUI.Column),
sh.Cells(sh.UsedRange.Rows.Count, mUI.Column))
    Else
        Set UI = mUI
    End If

End Property
Public Property Get AP(Optional cell As Range, Optional dataRange As
Boolean) As Range

    If Not cell Is Nothing Then
        Set AP = Intersect(mAP, cell.EntireRow)
    ElseIf dataRange = True Then
        Set AP = Range(sh.Cells(2, mAP.Column),
sh.Cells(sh.UsedRange.Rows.Count, mAP.Column))
    Else
        Set AP = mAP
    End If

End Property
```

```
Public Property Get PM(Optional cell As Range, Optional dataRange As
Boolean) As Range

    If Not cell Is Nothing Then
        Set PM = Intersect(mPM, cell.EntireRow)
    ElseIf dataRange = True Then
        Set PM = Range(sh.Cells(2, mPM.Column),
sh.Cells(sh.UsedRange.Rows.Count, mPM.Column))
    Else
        Set PM = mPM
    End If

End Property
Public Property Get Datum_dodani(Optional cell As Range, Optional
dataRange As Boolean) As Range

    If Not cell Is Nothing Then
        Set Datum_dodani = Intersect(mDatum_dodani, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Datum_dodani = Range(sh.Cells(2, mDatum_dodani.Column),
sh.Cells(sh.UsedRange.Rows.Count, mDatum_dodani.Column))
    Else
        Set Datum_dodani = mDatum_dodani
    End If

End Property
Public Property Get Vaha(Optional cell As Range, Optional dataRange As
Boolean) As Range

    If Not cell Is Nothing Then
        Set Vaha = Intersect(mVaha, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Vaha = Range(sh.Cells(2, mVaha.Column),
sh.Cells(sh.UsedRange.Rows.Count, mVaha.Column))
    Else
        Set Vaha = mVaha
    End If

End Property
Public Property Get Netto(Optional cell As Range, Optional dataRange
As Boolean) As Range

    If Not cell Is Nothing Then
        Set Netto = Intersect(mNetto, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Netto = Range(sh.Cells(2, mNetto.Column),
sh.Cells(sh.UsedRange.Rows.Count, mNetto.Column))
    Else
        Set Netto = mNetto
    End If

End Property
Public Property Get MJ(Optional cell As Range, Optional dataRange As
Boolean) As Range

    If Not cell Is Nothing Then
        Set MJ = Intersect(mMJ, cell.EntireRow)
```

```
    ElseIf dataRange = True Then
        Set MJ = Range(sh.Cells(2, mMJ.Column),
sh.Cells(sh.UsedRange.Rows.Count, mMJ.Column))
    Else
        Set MJ = mMJ
    End If

End Property
Public Property Get Poradi(Optional cell As Range, Optional dataRange
As Boolean) As Range

    If Not cell Is Nothing Then
        Set Poradi = Intersect(mPoradi, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Poradi = Range(sh.Cells(2, mPoradi.Column),
sh.Cells(sh.UsedRange.Rows.Count, mPoradi.Column))
    Else
        Set Poradi = mPoradi
    End If

End Property
Public Property Get Delka(Optional cell As Range, Optional dataRange
As Boolean) As Range

    If Not cell Is Nothing Then
        Set Delka = Intersect(mDelka, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Delka = Range(sh.Cells(2, mDelka.Column),
sh.Cells(sh.UsedRange.Rows.Count, mDelka.Column))
    Else
        Set Delka = mDelka
    End If

End Property
Public Property Get Sirka(Optional cell As Range, Optional dataRange
As Boolean) As Range

    If Not cell Is Nothing Then
        Set Sirka = Intersect(mSirka, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Sirka = Range(sh.Cells(2, mSirka.Column),
sh.Cells(sh.UsedRange.Rows.Count, mSirka.Column))
    Else
        Set Sirka = mSirka
    End If

End Property
Public Property Get Vyska(Optional cell As Range, Optional dataRange
As Boolean) As Range

    If Not cell Is Nothing Then
        Set Vyska = Intersect(mVyska, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Vyska = Range(sh.Cells(2, mVyska.Column),
sh.Cells(sh.UsedRange.Rows.Count, mVyska.Column))
    Else
        Set Vyska = mVyska
```

```
    End If

End Property
Public Property Get MJ2(Optional cell As Range, Optional dataRange As
Boolean) As Range

    If Not cell Is Nothing Then
        Set MJ2 = Intersect(mMJ2, cell.EntireRow)
    ElseIf dataRange = True Then
        Set MJ2 = Range(sh.Cells(2, mPreprava.Column),
sh.Cells(sh.UsedRange.Rows.Count, mMJ2.Column))
    Else
        Set MJ2 = mMJ2
    End If

End Property
Public Property Get Objem(Optional cell As Range, Optional dataRange
As Boolean) As Range

    If Not cell Is Nothing Then
        Set Objem = Intersect(mObjem, cell.EntireRow)
    ElseIf dataRange = True Then
        Set Objem = Range(sh.Cells(2, mObjem.Column),
sh.Cells(sh.UsedRange.Rows.Count, mObjem.Column))
    Else
        Set Objem = mObjem
    End If

End Property
Public Property Get MJ3(Optional cell As Range, Optional dataRange As
Boolean) As Range

    If Not cell Is Nothing Then
        Set MJ3 = Intersect(mMJ3, cell.EntireRow)
    ElseIf dataRange = True Then
        Set MJ3 = Range(sh.Cells(2, mMJ3.Column),
sh.Cells(sh.UsedRange.Rows.Count, mMJ3.Column))
    Else
        Set MJ3 = mMJ3
    End If

End Property
```

Příloha C

Testovací soubor „TEST_RYCHLOSTI.xlsm“

K dispozici na přiloženém CD.

Příloha D

Zdrojový kód celého modulu - Makra Popisování a Skener

```
Option Explicit
Option Base 1

Public shP As Worksheet
Private wbP As Workbook
Private data As ListObject
Private row As ListRow

Private ok As Boolean

Private cell As Range
Private cesta As String, jmeno As String
Private fullCesta As String

Public Sub tlSken(control As Object)

AVlog.logSub = "skener"
AVlog.Show

End Sub

Public Sub tlPopis(control As Object)

AVlog.logSub = "popis"
AVlog.Show

End Sub


Private Sub popis()

Application.ScreenUpdating = False

AVlog.progress "-------------------------------------------------------
-------"
AVlog.progress "Popisování - ver.05.bp.29.03.2023"
AVlog.progress "-------------------------------------------------------
-------"

Set shP = ActiveSheet
Set wbP = shP.Parent
Set data = shP.ListObjects(1)

cesta = ThisWorkbook.Path & "\POPIS\"

procesy.vymazatFiltry shP

ok = True

If ok Then ulozeni
If ok Then razeni
If ok Then formatovani
```

```
If ok Then filtr
If ok Then konec_ulozeni
If ok Then odeslani

End Sub

Private Sub skener()

AVlog.progress "-------------------------------------------------------
-------"
AVlog.progress "Popisování - skener ver.05.bp.29.03.2023"
AVlog.progress "-------------------------------------------------------
-------"

Set shP = ActiveSheet
Set wbP = shP.Parent
Set data = shP.ListObjects(1)

cesta = ThisWorkbook.Path & "\POPIS\"

procesy.vymazatFiltry shP

ok = True

If ok Then ulozeni
If ok Then razeni
If ok Then format_skener
If ok Then filtr
If ok Then exp_skener

End Sub


Private Sub ulozeni()

On Error Resume Next

    wbP.Save

    If Dir(cesta, vbDirectory) = Empty Then MkDir cesta

    jmeno = "popis_" & Format(Now, "ddmm")

    shP.Copy

    Set shP = ActiveSheet
        Application.ActiveWindow.WindowState = xlMinimized

    Set wbP = shP.Parent
    Set data = shP.ListObjects(1)

        Application.DisplayAlerts = False
    If funkce.bOtevreny(jmeno & ".xlsx") = True Then Workbooks(jmeno &
".xlsx").Close False

    shP.SaveAs Filename:=cesta & jmeno & ".xlsx", FileFormat:=51
```

```
If Err.Number > 0 Then
    AVlog.progress "V části ""Uložení"" zaznamenána chyba č." &
Err.Number & ", " & Err.Description
    Err.Clear
    ok = False
    Exit Sub
Else
     AVlog.progress "Uložení - OK"
End If

End Sub

Private Sub razeni()

With data.Sort.SortFields
    .Clear
    .Add Key:=data.ListColumns("Přeprava").Range,
SortOn:=xlSortOnValues, Order:=xlAscending,
DataOption:=xlSortTextAsNumbers
    .Add Key:=data.ListColumns("PSČ").Range, SortOn:=xlSortOnValues,
Order:=xlAscending, DataOption:=xlSortTextAsNumbers
    .Add Key:=data.ListColumns("Dodávka").Range,
SortOn:=xlSortOnValues, Order:=xlAscending,
DataOption:=xlSortTextAsNumbers
End With

With data.Sort
    .Header = xlYes
    .MatchCase = False
    .Orientation = xlTopToBottom
    .SortMethod = xlPinYin
    .Apply
End With

    AVlog.progress "Řazení - OK"

End Sub

Private Sub formatovani()

On Error Resume Next

funkce.usporadaniSloupcu shP, Array("Barva", "Přeprava", "PSČ",
"Dodávka", "Pořadí", "Typ balení", "Stát", "Jméno", "Místo", "Váha",
"MJ"), True

data.Resize shP.Cells(1).CurrentRegion

Dim sl As ListColumns: Set sl = data.ListColumns

sl("Přeprava").Range.ColumnWidth = 11
sl("PSČ").Range.ColumnWidth = 6
sl("Dodávka").Range.ColumnWidth = 20
sl("Pořadí").Range.ColumnWidth = 10
sl("Typ balení").Range.ColumnWidth = 8
sl("Stát").Range.ColumnWidth = 4
sl("Jméno").Range.ColumnWidth = 25
```

```
sl("Místo").Range.ColumnWidth = 22
sl("Váha").Range.ColumnWidth = 8
sl("MJ").Range.ColumnWidth = 3
sl("Pořadí").DataBodyRange.HorizontalAlignment = 3

shP.Cells.FormatConditions.Delete

For Each row In data.ListRows

        Select Case h("Barva")
        Case "MODRÁ"
        row.Range.Interior.Color = 15849925
        Case "ČERVENÁ"
        row.Range.Interior.Color = 12040422
        Case "ZELENÁ"
        row.Range.Interior.Color = 12379352
        End Select

Next row

With data.Range
.Borders(xlEdgeBottom).LineStyle = xlContinuous
.Borders(xlEdgeLeft).LineStyle = xlContinuous
.Borders(xlEdgeRight).LineStyle = xlContinuous
.Borders(xlEdgeTop).LineStyle = xlContinuous
.Borders(xlEdgeBottom).Weight = xlMedium
.Borders(xlEdgeLeft).Weight = xlMedium
.Borders(xlEdgeRight).Weight = xlMedium
.Borders(xlEdgeTop).Weight = xlMedium
.Borders(xlInsideVertical).LineStyle = xlContinuous
.Borders(xlInsideHorizontal).LineStyle = xlContinuous
.Borders(xlInsideVertical).Weight = xlThin
.Borders(xlInsideHorizontal).Weight = xlThin
End With

If Err.Number > 0 Then
    AVlog.progress "V části ""Formátování"" zaznamenána chyba č." &
Err.Number & ", " & Err.Description
    Err.Clear
    ok = False
    Exit Sub
Else
    AVlog.progress "Formátování - OK"
End If
End Sub

Private Sub format_skener()

funkce.usporadaniSloupcu shP, Array("Dodávka", "BARVA", "Pořadí"),
True

data.Resize shP.Cells(1).CurrentRegion

Dim sl As ListColumns: Set sl = data.ListColumns

sl("Dodávka").Range.ColumnWidth = 21
sl("Pořadí").Range.ColumnWidth = 10
```

```
shP.Cells.FormatConditions.Delete

If Err.Number > 0 Then
    AVlog.progress "V části ""Formátování"" zaznamenána chyba č." &
Err.Number & ", " & Err.Description
    Err.Clear
    ok = False
    Exit Sub
Else
    AVlog.progress "Formátování - OK"
End If
End Sub

Private Sub filtr()

    data.Range.AutoFilter
Field:=data.ListColumns("BARVA").Range.Column, _
    Criteria1:=Array("MODRÁ", "ČERVENÁ", "ZELENÁ"),
Operator:=xlFilterValues

    data.ListColumns("Barva").Range.EntireColumn.Hidden = True

    AVlog.progress "Filtr - OK"


End Sub

Private Sub konec_ulozeni()

On Error Resume Next

If WorksheetFunction.Subtotal(3, data.ListColumns("Dodávka").Range)
Mod 36 < 4 Then
With shP.PageSetup
        .LeftMargin = Application.InchesToPoints(0.7)
        .RightMargin = Application.InchesToPoints(0.7)
        .TopMargin = Application.InchesToPoints(0.75)
        .BottomMargin = Application.InchesToPoints(0.75)
        .HeaderMargin = Application.InchesToPoints(0.3)
        .FooterMargin = Application.InchesToPoints(0.3)
        .CenterHorizontally = True
End With
End If

Dim vysledek As Integer: vysledek = WorksheetFunction.Subtotal(9,
data.ListColumns("Váha").DataBodyRange)
Dim soucet As Integer: soucet = WorksheetFunction.Subtotal(3,
data.ListColumns("Váha").DataBodyRange)

With shP.Cells(data.ListRows.Count + 3, data.ListColumns.Count - 4)
.Value = "Celkem " & soucet & " colli / " & vysledek & " kg"
.Font.FontStyle = "Calibri"
.Font.Size = 11
.Font.Bold = True
End With
```

```
cesta = cesta & "\Export"

If Dir(cesta, vbDirectory) = Empty Then MkDir cesta

fullCesta = cesta & "\" & jmeno & ".pdf"

shP.ExportAsFixedFormat Type:=xlTypePDF, Filename:=fullCesta,
Quality:=xlQualityStandard, IgnorePrintAreas:=False,
OpenAfterPublish:=False

AVlog.progress "Uložení - OK"

wbP.Close (True)

If Err.Number > 0 Then
    AVlog.progress "V části ukončení zaznamenána chyba č." &
Err.Number & ", " & Err.Description
    Err.Clear
    ok = False
    Exit Sub
End If

End Sub

Private Sub odeslani()

Dim appOut As Object: Set appOut = CreateObject("Outlook.Application")
Dim zprava As Object

Set zprava = appOut.CreateItem(0)

    With zprava
        .To = "SKLAD"
        .CC = "DISPECINK"
        .Subject = Date & " Popisování"
        .htmlbody = "Ahoj,<br><br>v příloze posílám popisování na
dnešní nakládku.<br><br>Děkuji<br><br>"
    End With

zprava.Attachments.Add (fullCesta)
zprava.Display

Set appOut = Nothing

End Sub


Private Sub exp_skener()

Dim wbSken As Workbook: Set wbSken = Workbooks.Add(ThisWorkbook.Path &
"\" & "SKENER.XLSX")
Application.ActiveWindow.WindowState = xlMinimized

Dim dataSken As ListObject: Set dataSken =
wbSken.Sheets(1).ListObjects(1)
cesta = ThisWorkbook.Path & "\SKEN"
If Dir(cesta, vbDirectory) = Empty Then MkDir cesta
```

```
Dim soubor As Object
AVlog.progress "-----------------------------------------------------
-------"
AVlog.progress "Kopírování dat..."

For Each wbP In Workbooks
    If Not wbP.Name Like "popis*" Then GoTo dalsiSesit

    Set shP = wbP.Sheets(1)
    Set data = shP.ListObjects(1)
    If data.DataBodyRange Is Nothing Then GoTo dalsiPopis

    procesy.vymazatFiltry shP

    For Each row In data.ListRows
        If h("Pořadí") = "N" Or h("Pořadí") = Empty Then h("Pořadí") =
"X"
    Next row

    data.DataBodyRange.Copy

    If dataSken.DataBodyRange Is Nothing Then
        wbSken.Sheets(1).Cells(dataSken.Range.Rows.Count,
dataSken.ListColumns("Dodávka").Range.Column).PasteSpecial
    Else
        wbSken.Sheets(1).Cells(dataSken.Range.Rows.Count + 1,
dataSken.ListColumns("Dodávka").Range.Column).PasteSpecial
    End If

    dataSken.Resize wbSken.Sheets(1).Range("A1:C" &
wbSken.Sheets(1).UsedRange.Rows.Count)

dalsiPopis:
    wbP.Close SaveChanges:=False
dalsiSesit:
Next wbP

AVlog.progress "Kopírování dat - OK"

Dim Nazev As String

If Dir(cesta & "\Popisovani " & Format(Date, "dd.mm.") & ".xlsx") =
Empty Then
    Nazev = "Popisovani " & Format(Date, "dd.mm.")
Else
Dim i As Integer: i = 1
    Nazev = "Dohoz " & i & " " & Format(Date, "dd.mm.")

    Dim konec As Boolean
    Do Until konec
        If Not Dir(cesta & "\" & Nazev & ".xlsx") = Empty Then
            i = i + 1
            Nazev = "Dohoz " & i & " " & Format(Date, "dd.mm.")
        Else
            konec = True
        End If
```

```
    Loop

End If

wbSken.Sheets(2).Activate
wbSken.SaveAs cesta & "\" & Nazev & ".xlsx"
wbSken.Close True

Dim email As Object

Dim appOut As Object: Set appOut = CreateObject("Outlook.Application")

Set email = appOut.CreateItem(0)
With email
.To = "SKLAD"
.Subject = Nazev
.Attachments.Add cesta & "\" & Nazev & ".xlsx"
.Body = "Ahoj," & vbCrLf & vbCrLf & "posílám dnešek." & vbCrLf &
vbCrLf
.Display

End With

Set appOut = Nothing

End Sub

Private Property Get h(nazevSl As String) As Variant
    h = data.DataBodyRange(row.Index,
data.ListColumns(nazevSl).Index).Value
End Property

Private Property Let h(nazevSl As String, ByVal vNewValue As Variant)
    data.DataBodyRange(row.Index,
data.ListColumns(nazevSl).Index).Value = vNewValue
End Property

Private Function r(nazevSl As String) As Range
    Set r = data.DataBodyRange(row.Index,
data.ListColumns(nazevSl).Index)
End Function
```

Příloha E

Výstupní soubor makra SKENER – „Popisovani 19.03..xlsx“

K dispozici na přiloženém CD.

Autor/ka BP	
Název BP	
Studijní program	
Rok obhajoby BP	**2023**
Počet stran	
Počet příloh	
Vedoucí BP	
Anotace	
Klíčová slova	
Místo uložení	ITC (knihovna) Vysoké školy logistiky v Přerově
Signatura	