Dušan Streit
ŠLEZAN Frýdek-Místek

SELEKTIVNÍ SYNTÉZA SETŘÍDĚNÝCH SÉRIOVÝCH SOUBORŮ NA ÚROVNI VĚT O SHODNÝCH KLÍČÍCH PROSTŘEDNICTVÍM GENERÁTORŮ PROGRAMŮ

1. Strukturalizace a dekompozice systému zpracování dat z hlediska zajištění jeho skeletu uživatelským software.

Všeobecně jsou známy racionalizační snahy v oblasti programování. Různými prostředky (zdokonalené programovací techniky, vyšší programovací jazyky, rozhodovací tabulky, generátory, kompilátory, operační systémy) se skutečně ve všeobecném měřítku úroveň programování zvýšila a blíží se technickým možnostem počítačů 3. generace, k čemuž již několik let přispívá i seminář v Havířově. Celková úroveň systému zpracování dat (od ostatních částí ASŘ se budu abstrahovat) však nezávisí jen na úrovni programování, ale hlavně na úrovni projektování tohoto systému. Nic by nebyla platná dokonalá architektura programu, kdyby samotný jeho účel, vymezení jeho funkcí a jeho návaznost na ostatní prvky systému byly pochybné. Většina analytiků je ryze profesně a informačně orientována,a dochází proto k rozporu mezi organizovaností v rámci programu a anarchií celého systému zpracování dat z hlediska vnitřních vazeb. Je proto na nás, programátorech, kteří se nestydí za to, že jsou programátory (ne podle katalogu, ale podle zaměření) a kteří nepovažují svou profesi za přestupní stanici mezi analytiky, aby"přinutili" analytiky pracovat i v této

styčné oblasti racionálně, a to prostřednictvím uživatelského software.

Při tvorbě uživatelského software se programátor stává nejen programátorem, ale i projektantem - buduje potenciální, budoucí systém zpracování dat. Princip strukturovaného programování má možnost použít na vyšších dekompozičních úrovních - na úrovni programových chodů, na úrovni celého systému zpracování dat - a vtisknout mu tak charakter strukturovaného projektování. Na této úrovni je obecný programový chod modulem - je to programový panel, ze kterého je možno vystavět systém zpracování dat. Neobstojí námitka, že systém implementovaný z obecných chodů bude méně efektivní, než systém vybudovaný individuálně. Také např. COBOL je z tohoto pohledu méně efektivní než ASSEMBLER, a přesto je na určité úrovni znalostí a v určitém okruhu použití výhodnější: vyšší adaptibilita, portabilita, flexibilita, nižší pracnost a rozsah chyb. I když jedním z prostředků pro dosažení jmenovaných výhod u obecných programových chodů je právě použití ASSEMBLERu pro jeho programování.

V příloze č. 1 je alternativní schema obecného systému zpracování dat sériových souborů. Konverze dat, aktualizace kmenových souborů, kontroly a tisk jsou zajištěny obecnými typovými chody, sestavenými z interpretačních parametrických programů. Ze schematu je zřejmé, že vlastní výpočty jsou "očištěny" od práce se soubory, zbývá algoritmická část vhodná ke zpracování rozhodovacími tabulkami. To je umožněno právě funkcemi syntézy (sdružování) a selekce (výběru).

2. Metodika sdružování a výběru dat.

Při zpracování hromadných dat se nelze obejít bez třídění a následného vzájemného přiřazování dat z různých souhlasně setříděných souborů na základě shodných klíčů. U sériových souborů to představuje vzájemné "dotáčení"

(synchronizaci) souborů v jednotlivých programech. Tuto oblast běžně řeší normované programování. Při hlubší analýze však je patrné, že je neefektivní tento problém vždy znovu programovat, tedy i normovaně. Navíc práce se soubory a současné zpracování dat v jednom programu zpomaluje chod programu a omezuje multiprogramování. Výhodnější je vyčlenit a standardizovat potřebné funkce pro syntézu požadovaných souborů do jednoho výstupního souboru (funkce v zásadě vždy stejné) a teprve z něj provádět v samostatném programu algoritmické zpracování dat (individuální pro každý případ) s možností multiprogramování.

V programech nebude nutno uvažovat s tabulkami dat pro soubory, které nejsou souhlasně setříděny. Podstatně se sníží počet třídicích programů, protože odpadnou opakovaná třídění stejnými klíči, která se prováděla z titulu opakovaného používání kmenových souborů. Výhody nového řešení jsou dokumentovány na skupině úloh "úkolové mzdy", viz. příloha č. 2.

Celkový počet programů se snížil z 6 na 5, počet třídění se snížil ze 3 na 2, počet programů, které je nutno programovat ze 3 na 1. Do tohoto programu jak vstupuje tak vystupuje pouze 1 soubor. Takový program je přehlednější, výhodnější pro zpracování rozhodovacími tabulkami a pro multiprogramování. Většina práce (4 programy) je hotova vlastně už dříve, než začneme programovat.

Doplněním syntézy o možnost selekce - výběru vět a výběru položek je možnost použití ještě širší: např. ve spojení s tiskovým chodem bude tato cesta dobrým prostředkem všude tam, kde je potřeba rychle a bez nároku na programování (třeba i jednorázově) vystoupit s informacemi, které lze získat z existujících datových souborů.

Analytik je tak orientován na práci se soubory, jejichž obsah by měl být jeho doménou a je odtržen od "projektování" programu.

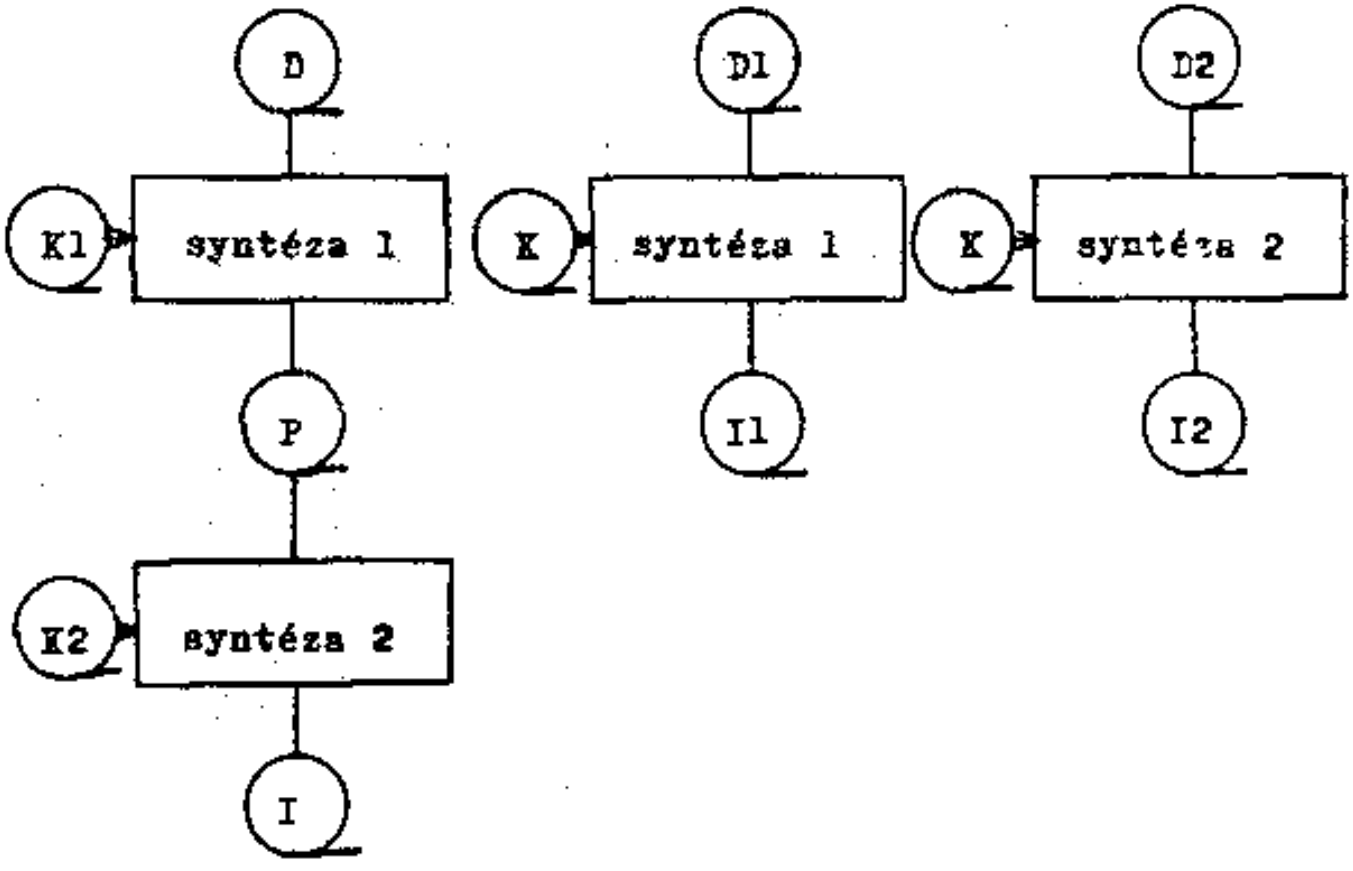
Obecný systém zpracování dat je kostrou, která drží úroveň systému i tam, kde jednotlivé obecné typové chody

svými možnostmi nepostačují a kde je třeba individuálně možnosti rozšířit. Každý prvek má však své místo a individuální systém se tvoří v intencích obecného systému.

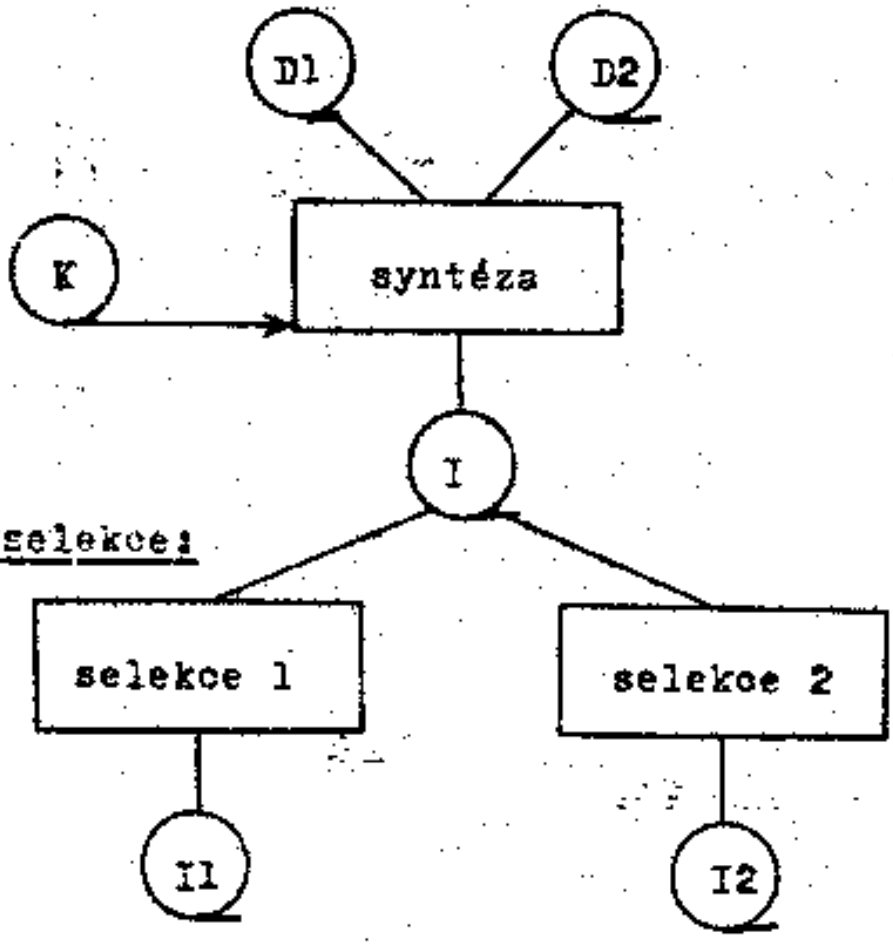
Jádrem celého obecného systému zpracování dat je informační soubor (jakási báze dat na úrovni skupiny úloh resp. subsystému), viz příloha č. 1. Úplný informační soubor nemusí fyzicky existovat, na základě syntézy je však vždy možno sdružit všechny potřebné (vybrané) položky (jednota syntézy a selekce - 2 stránky jednoho procesu).

Jestliže rozdílné klíčové údaje neumožňují získání takovéhoto souboru na 1 průchod syntézou, je možné ho získat sériovým způsobem na více průchodů s mezitříděním, viz. příloha č. 2. Naopak, chceme-li informační soubor rozvětvit, zpracujeme syntézu paralelním způsobem.

Syntéza: a) sériová b) paralelní

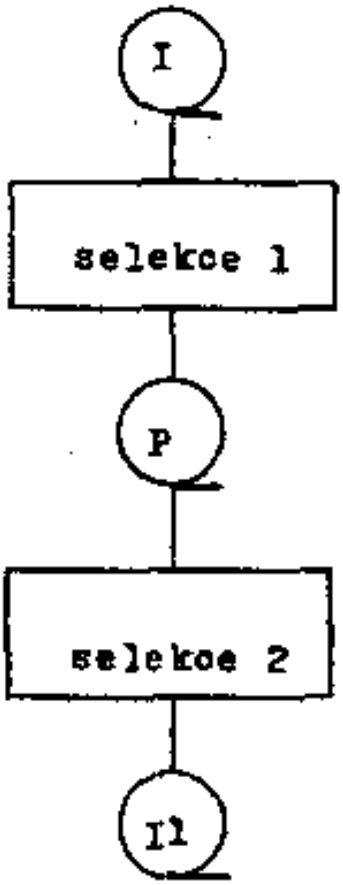


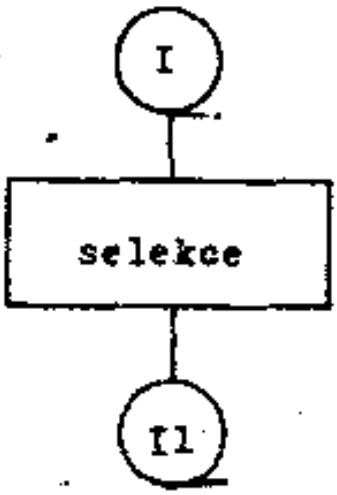
Paralelní syntéza je vlastně výběrem z fiktivního celkového informačního souboru:



paralelní selekce:

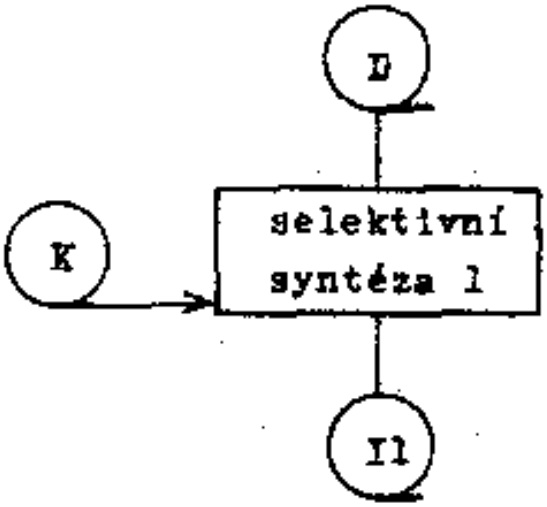
Skutečně další možnosti selektivní syntézy je výběr z existujícího souboru, a to <u>výběr vět</u> a <u>výběr položek</u>. <u>Paralelní selekcí</u> lze tak získat soubory, které jsou podmnožinami existujícího souboru. Výběr na úrovni položek je možno provádět z každého souboru, který vstupuje do <u>selektivní syntézy</u>, výběr na úrovni vět je možno provádět ze všech souborů, a to včetně výstupního; selekce a syntéza se provádějí implicitně. Selekce v čisté formě se tedy provádí tehdy, vstupuje-li do selektivní syntézy pouze 1 soubor. Čistá <u>selekce sériovým způsobem</u> nemá smysl, dá se provést na 1 průchod (i tehdy, jedná-li se o výběr vět a výběr položek současně):

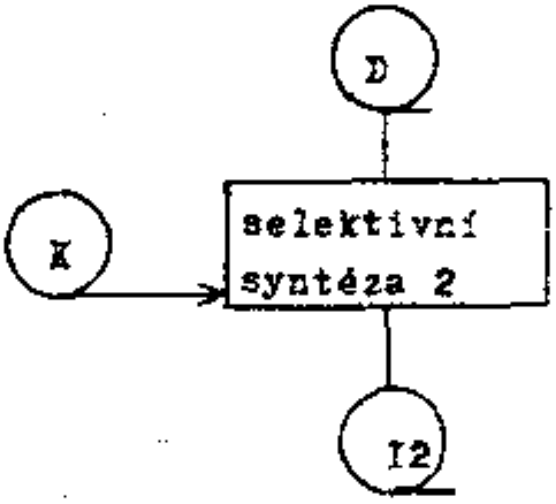




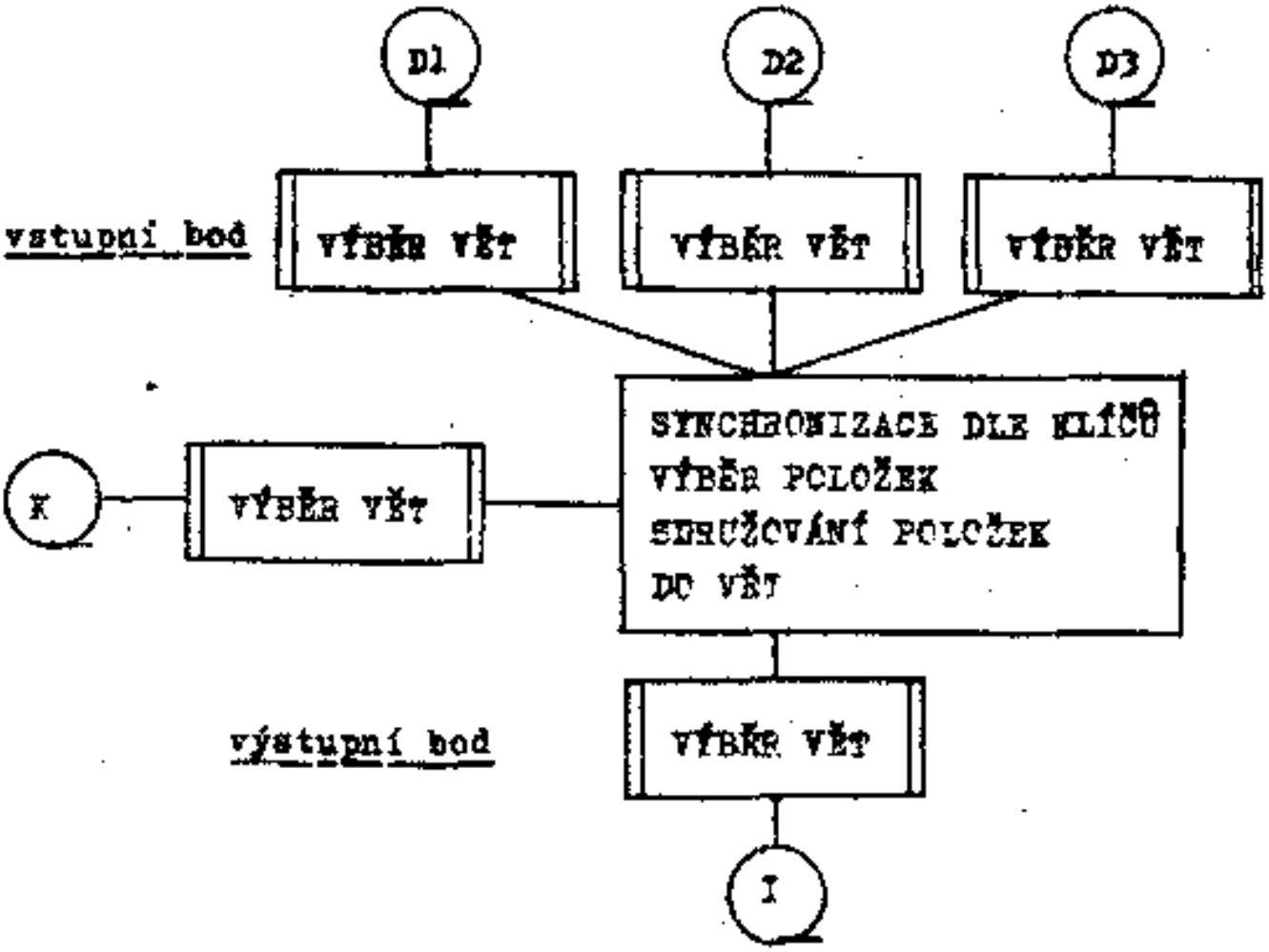
3. Selektivní syntéza

Je třeba si uvědomit, že použití selektivní syntézy pro stejné soubory nemusí vést ke stejným výsledkům na základě podmínek výběru:

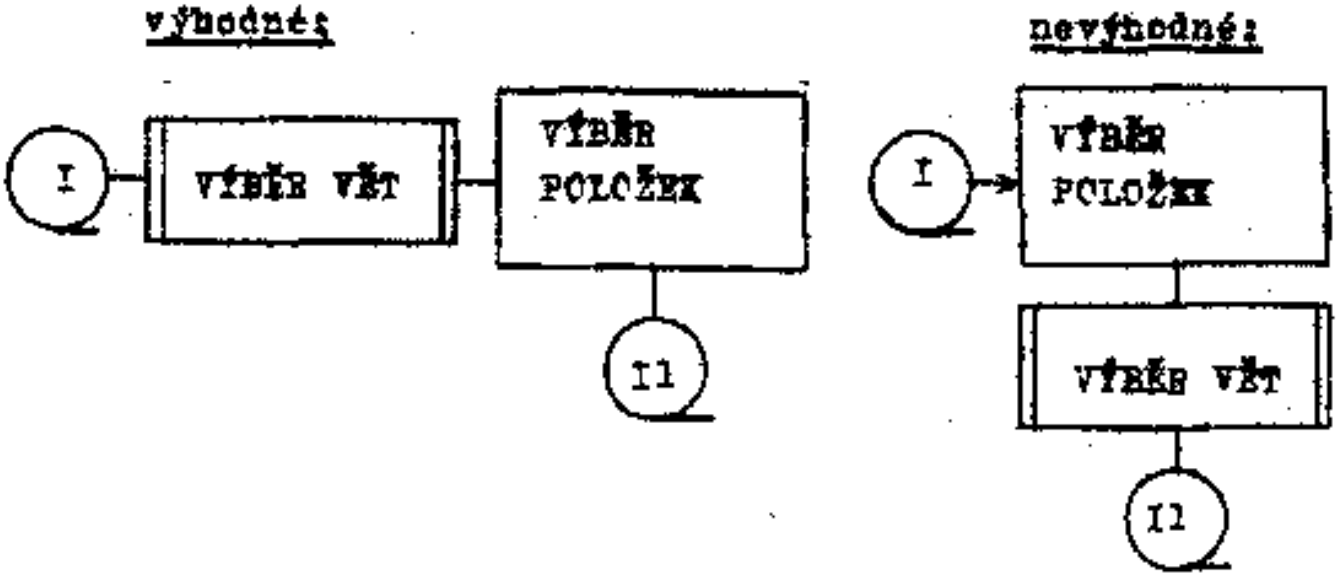




Celkové schema selektivní syntézy:



U čisté selekce je vzhledem k času CPU výhodnější zařazovat podmínky výběru vět do vstupního než do výstupního bodu:



Kombinací sériového a paralelního zpracování selektivní syntézy je možno zajistit jak sdružování údajů na vstupu a výběry pro výpočty, tak i agregaci a výběry údajů pro tisky, viz příloha č. 1.

Je možno provádět syntézu až 4 souborů dle klíčů v celkové délce max. 36 znaků (=9 slov). V 1 vstupním souboru je uvažováno fyzicky s 1 typem věty. Syntéza probíhá na úrovni vět; ze vstupních vět se tvoří výstupní věta o následujícím obsahu:

slovo 0	slovo 1 až slovo 9	úsek (sektor)	úsek (sektor)	úsek (sektor)	úsek (sektor)
čítač	klíčové pole	souboru 1	souboru 2	souboru 3	souboru 4

Výstupní věta má klíčové pole o pevné délce 36 znaků, do kterého se skládají od leva jednotlivé klíče dle priorit; zprava jsou doplněny nuly. Každý definovaný vstupní soubor má ve výstupní větě sektor, do kterého se přenášejí požadované položky ve stejném pořadí jako jsou ve vstupní větě. Popis výstupní věty je dán implicitně popisem vstupních vět. Klíčové údaje v popisech vstupních vět jsou označeny v souladu s prioritami zvláště volenými identifikátory. Popisy vstupních vět je možno získat pomocí příkazu COPY z katalogu souborů. Přesun ze vstupu na výstup se provádí vygenerováním instrukce MOVE CORRESPONDING, tzn. že výběr položek je zajištěn tím, že se nepřenášejí položky popsané FILLER. Položky v popisech je možno libovolně rozčleňovat a redefinovat. Do výstupní věty se tak dostanou vždy údaje max. z 1 věty z 1 vstupního souboru, tzn. celkově až ze 4 vět (mají-li věty ze všech 4 souborů shodné klíče). Na výstupu se utvoří za sebou tolik vět o shodných klíčích, kolik jich má vstupní soubor s největším počtem vět o shodných klíčích. Ve všech těchto větách jsou "taženy" údaje z věty souboru 1 (tzv. kmenového). Údaje ostatních souborů mají platnost v rámci 1 věty, a jestliže příslušná věta není frekventována, její sektor je na výstupní větě naplněn nulami.

Příklad:

klíče souboru 1: 1, 2, 3, 4, 5, 7, 9, 10, EOF
klíče souboru 2: 2, 2, 4, 6, 6, 7, EOF
klíče souboru 3: 2, 3, 4, 6, 8, 9, 11, EOF

	Klíč	Sektor 1	Sektor 2	Sektor 3
1)	1	X	0	0
2)	2	X	X	X
3)	2	X	X	0
4)	3	X	0	X
5)	4	X	X	X
6)	5	X	0	0
7)	6	0	X	X
8)	6	0	X	0
9)	7	X	X	0
10)	8	0	0	X
11)	9	X	0	X
12)	10	X	0	0
13)	11	0	0	X

U případu 3) je patrné, že údaje souboru 1 mají trvalou platnost, dokud se nezmění klíč (porovnej se souborem 3 u stejného případu). Nasazením klíče - přepínače je možno ze syntézy vyloučit ty případy, kdy ke kmenové větě ze souboru 1 není nalezena žádná věta o shodném klíči u jiných souborů, viz. případy 1), 6), 12). Automaticky není kontrolováno, že ke každé větě existuje kmenová věta ze souboru 1; případy 7), 8), 10), 13) jsou zapsány do výstupního souboru, pokud není určeno jinak v uživatelských podmínkách výběru ve výstupním bodě.

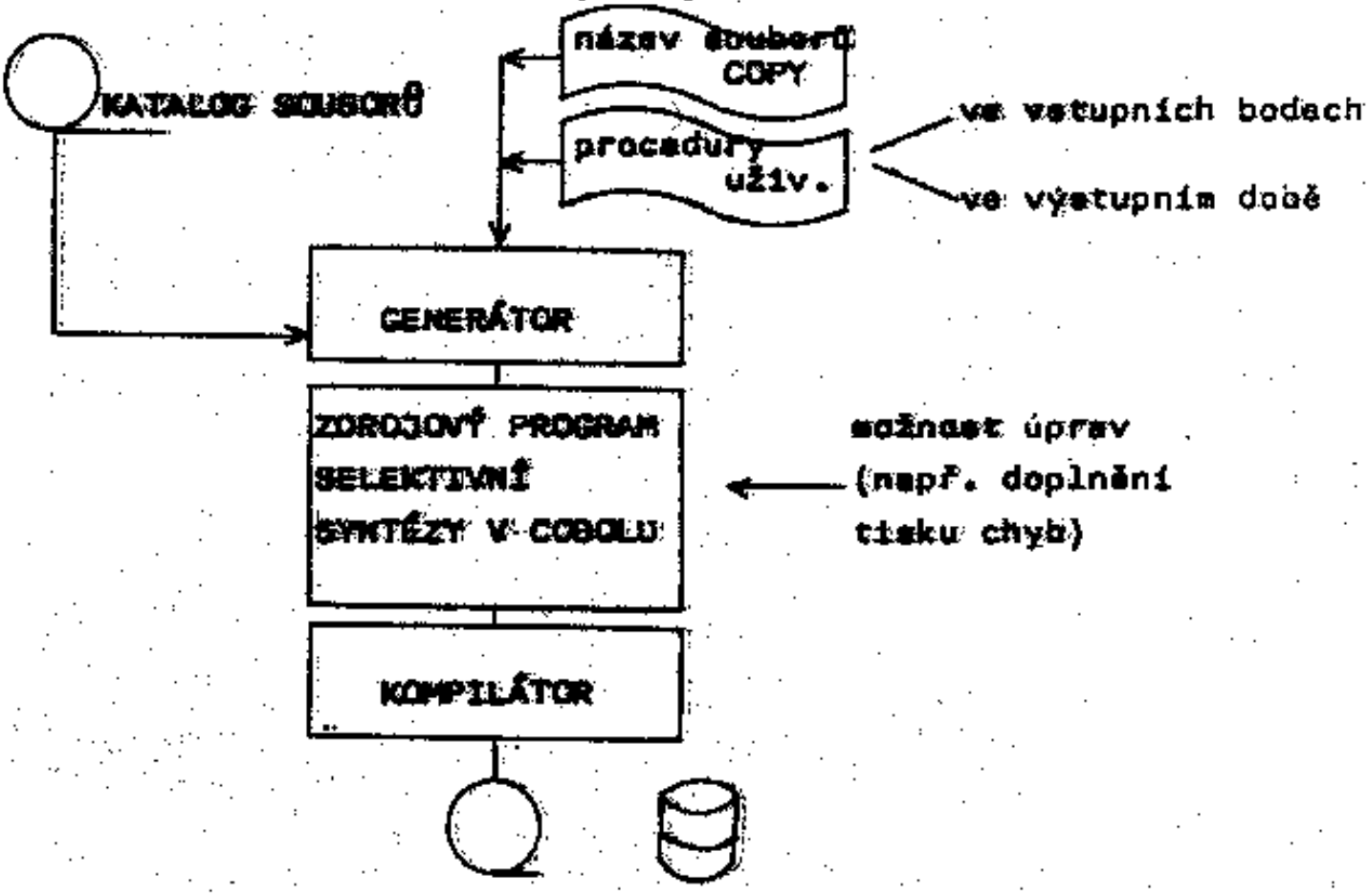
Při konkrétním zpracování syntézy se může stát, že neexistuje definovaný vstupní soubor (neexistuje např. soubor dobropisů, protože žádný dobropis nebyl v určitém období vystaven). Nasazením příslušných klíčů - přepínačů nebudou příslušné soubory na vstupu požadovány. Nepřítomné soubory budou mít však ve výstupní větě rezervovány

příslušné sektory, ty však budou naplněny nulami.

Výběry vět jsou zajištěny uživatelskými procedurami, které je možno vztahovat ke vstupním i výstupnímu souboru.

Program selektivní syntézy vzniká prostřednictvím generátoru programů v jazyce COBOL na principu modulárního programování.

Schema tvorby selektivní syntézy:

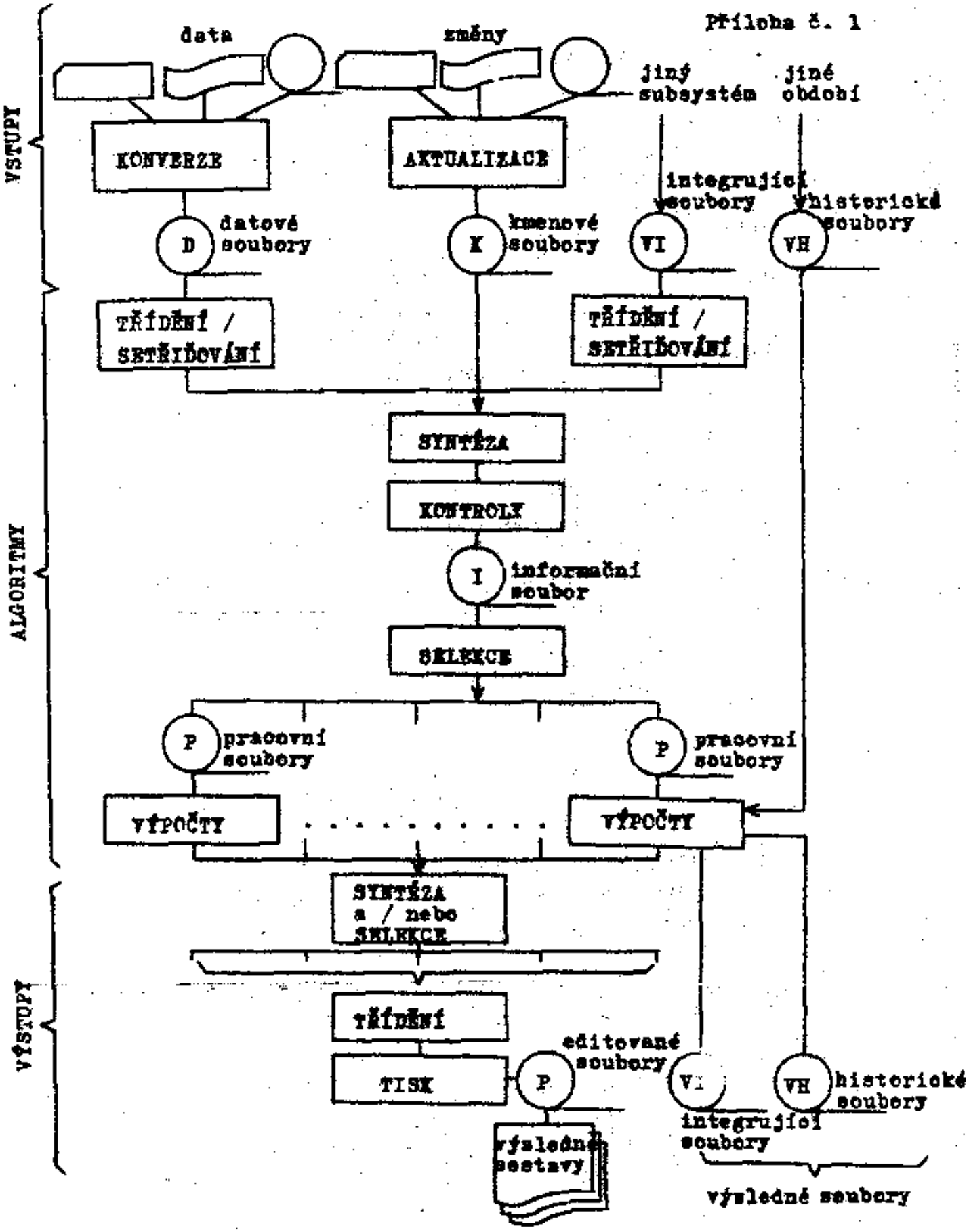


Uvedený postup se dá výhodně kombinovat s běžnými programovacími technikami. Jeho flexibilita je možným nástrojem zvyšování kvality programů.

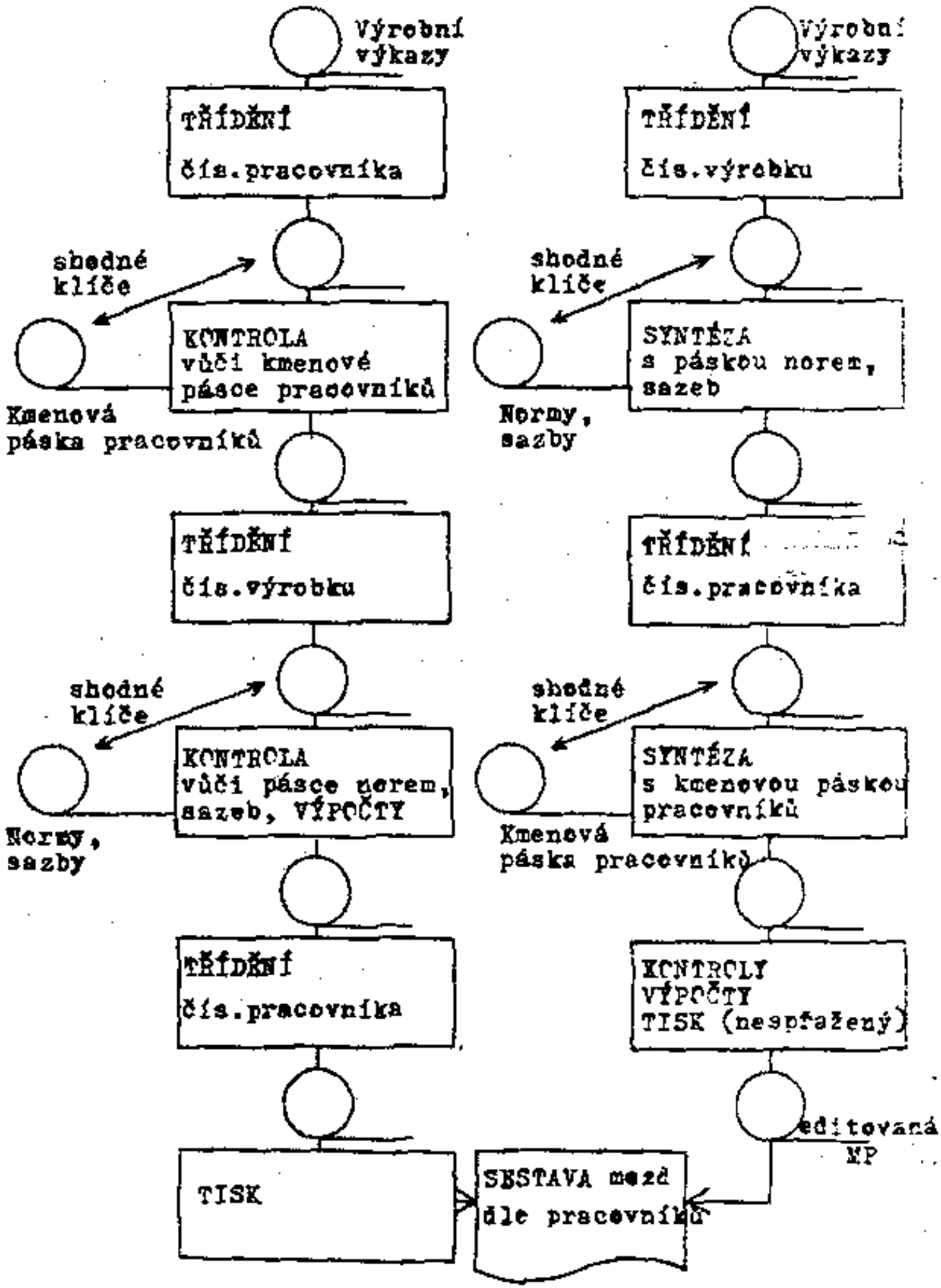
Literatura:

(1) Doc. ing. Adamec, CSc.: Systémová analýza a projektování ASŘ
(2) Streit: Standardizace v oblasti konverzí a prvotních kontrol dat, sborník Programování 77, Havířov
(3) Zbytek: Standardní chod pro zakládání a údržbu kmenových souborů, sborník Programování 79, Havířov
(4) Syst. manuál ICL: Data Management System

Příloha č. 1


VSTUPY
ALGORITMY
VÝSTUPY
data
změny
jiný subsystém
jiné období
KONVERZE
AKTUALIZACE
integrující soubory
historické soubory
D datové soubory
K kmenové soubory
VI
VH
TŘÍDĚNÍ / SETŘIĎOVÁNÍ
TŘÍDĚNÍ / SETŘIĎOVÁNÍ
SYNTÉZA
KONTROLY
I informační soubor
SELEKCE
P pracovní soubory
P pracovní soubory
VÝPOČTY
VÝPOČTY
SYNTÉZA a / nebo SELEKCE
TŘÍDĚNÍ
TISK
P editované soubory
VI
VH historické soubory
integrující soubory
výsledné sestavy
výsledné soubory

Příloha č. 2


Konvenční přístup
Výrobní výkazy
TŘÍDĚNÍ
čís.pracovníka
shodné klíče
KONTROLA vůči kmenové pásce pracovníků
Kmenová páska pracovníků
TŘÍDĚNÍ
čís.výrobku
shodné klíče
KONTROLA vůči pásce norem, sazeb, VÝPOČTY
Normy, sazby
TŘÍDĚNÍ
čís.pracovníka
TISK
Nové řešení
Výrobní výkazy
TŘÍDĚNÍ
čís.výrobku
shodné klíče
SYNTÉZA s páskou norem, sazeb
Normy, sazby
TŘÍDĚNÍ
čís.pracovníka
shodné klíče
SYNTÉZA s kmenovou páskou pracovníků
Kmenová páska pracovníků
KONTROLY VÝPOČTY TISK (nespřažený)
editovaná MP
SESTAVA mezd dle pracovníků