TECHNIKA LOGICKÉHO ŠTRUKTUROVANIA V PROGRAMOCH APLIKAČNÝCH SYSTÉMOV

Ing. František S e v e r a, Ing. Jana K o c ú r o v á

Štátna banka československá, Útvar mechanizácie a automatizácie, Bratislava

Úspešné riešenie problémov rastúceho rozsahu a zložitosti programových systémov vyžaduje neustále zdokonaľovanie metód programovania. V posledných rokoch sa u nás vďaka zásluhnej propagácii /aj na tychto seminároch/ začínajú uplatňovať prístupy, metódy a techniky širokého komplexu, označovaného spravidla súhrnným názvom štrukturované programovanie. V mnohých prípadoch sa však prax štrukturovaného programovania obmedzuje prevážne len na používanie klasických štruktúr riadenia pri vlastnom kódovaní programu, prípadne na používanie pseudokódu pri dokumentovaní programov. Pritom využívanie týchto nástrojov predovšetkým vo fáze návrhu programu podstatne rozširuje možnosti ďalšieho zdokonalenia kvality a efektívnosti návrhu, realizácie, údržby a rozvoja najmä väčších programových systémov. Účelom tohto príspevku je poskytnúť základné informácie o novom prístupe k logickému štrukturovaniu programov, metodicky rozpracovanému a prakticky overenému na našom pracovisku.

Podstata nového prístupu k návrhu programovej štruktúry nie je nová a spočíva v prioritnom členení funkcií programu na funkcie riadenia a funkcie spracovania abstraktných dátových štruktúr. Prvotným cieľom návrhu programu je určenie abstraktných dátových štruktúr, ktoré sú predmetom spracovania. Abstraktné dátové štruktúry súčasne slúžia ako modeľ korektných dát vrátane vzájomných väzieb ich jednotlivých prvkov. Pri návrhu abstraktných dátových štruktúr ide predovšetkým o postupnú identifi-

káciu a klasifikáciu dátových prvkov a ich vzťahov na jednotlivých úrovniach abstrakcie formou explicitne definovaných riadiacich premenných a ich možných hodnôt.

Algoritmus spracovania sa zostavuje na základe princípu korešpondencie štruktúry dát a štruktúry programu. Explicitne definované riadiace premenné dátových štruktúr považujeme pritom na každej úrovni návrhu súčasne za riadiace premenné v štruktúre procesov ich spracovania. Algoritmus spracovania sa zostavuje postupne podľa jednotlivých úrovní abstrakcie. Pri tomto postupe je na každej úrovni možná súbežná verifikácia návrhu tak z hľadiska korektnosti a úplnosti dát, ako aj z hľadiska správnosti riadenia procesov ich spracovania. Algoritmus spracovania je pri tomto postupe ponímaný ako proces, vyjadrujúci postup dynamickej rekonštrukcie navrhovaných abstraktných dátových štruktúr.

K tomu, aby uvedené abštraktné štruktúry dát a programu, okrem logickej prehľadnosti a náročnosti, mali súčasne aj praktický význam priamo realizovateľný logických konštrukcií, je však okrem uvedeného nevyhnutné koncipovať moduly obsluhy reálnych dátových súborov tak, aby komunikácia fyzických záznamov do vlastného programu zodpovedala logickej štruktúre dát. Tento moment, aj keď je tiež sám o sebe dobre známy, je v tejto súvislosti oproti Jacksonovej metóde novým prvkom štrukturovaného návrhu.

Dôsledným a súčasným uplatnením uvedených dvoch koncepčných prvkov - poprvé, explicitne definovaných riadiacich premenných - a podruhé, modelov logickej obsluhy fyzických súborov a logického riadenia internej komunikácie fyzických záznamov - sme dosiahli kvalitatívne novú úroveň v praktickej aplikácii metód štrukturovaného programovania.

Uvedená metóda umožňuje napr. dosiahnúť korešpondenciu logickej štruktúry problému /pohľad a jazyk užívateľa/ a fyzio-

kej štruktúry programu /pohľad a jazyk riešiteľa/. Pritom nie je nepodstatné, že obidva uvedené aspekty /v bežnej praxi často veľmi vzdialené a odlišné/ je možné nielen principiálne ale aj prakticky integrovať na jedinej schéme jednak pomocou modifikovaných Jacksonových diagramov a jednak pomocou pseudokódu. V tejto súvislosti je účelné zaregistrovať aj skutočnosť, že v bežnej praxi programovania sa doteraz využíva pseudokód prevažne jednostranne, t.j. iba ku logickému zápisu abstraktných procesov. Vtip je v tom, že abstraktné dátové štruktúry možno prakticky využiť pre štrukturovanie programu len v spojení s im zodpovedajúcimi procesmi ich obsluhy a riadenia. V opačnom prípade nenavrhujeme štrukturovaný program - iba používame základné štruktúry riadenia, ktoré samé o sebe v zložitejších prípadoch nepostačujú k zachovaniu logickej štruktúry programu.

Prezentovaná technika logického štrukturovania programov aplikačných systémov spája základné idey štrukturovaného programovania, abstraktných funkcií pseudokódu a abstraktných dátových štruktúr do kvalitatívne nového systému abstraktného programovania, otvárajúceho ďalší smer vývoja programovacích prostriedkov. Nakoľko základné informácie o nových aspektoch metódy abstraktného programovania sú z dôvodov obmedzeného rozsahu príspevku podané v teoretickej rovine, dokumentujeme uvedený postup v prílohe krátkym príkladom.

Uvedený príklad z Jacksona a porovnanie štruktúry pôvodného riešenia s naším, umožňuje posúdiť, o koľko je možné postúpiť v štruktúre programu už v tomto ešte relatívne jednoduchom prípade vstupom riadeného programu s dvomi súbormi na vstupe. S ohľadom na kľúčový význam modulov obsluhy a riadenia fyzických súborov doplňujeme uvedený príklad o charakteristiku týchto modulov, pričom sa pre názornosť obmedzíme na kategóriu vstupom riadených programov.

Moduly riadenia a obsluhy vstupných súborov predstavujú štrukturotvorný prvok programov s viacerými sekvenčnými súbormi. Umožňujú dosiahnuť vysokú úroveň korešpondencie štruktúry rie-

šeného problému /jeho funkcií/ a vlastnej programovej štruktúry. Vo väčších výpočtových strediskách môžu byť využívané vo viacerých programoch, čím sa znižujú nároky na rozvoj a údržbu programových systémov.

Ako centrálny a kľúčový prvok pri štrukturovaní metódou modulov riadenia a obsluhy vstupných súborov sa zavádza pojem skupiny a jej typu. Skupinu tvoria záznamy zo všetkých vstupných súborov s rovnakou hodnotou identifikátora. Moduly riadenia a obsluhy vstupných súborov potom vykonávajú

- sekvenčné čítanie vstupných súborov na úrovni skupín
- vyhodnotenie načítaných záznamov - viet s ohľadom na sekvenčnú postupnosť identifikátorov
- odovzdanie viet na spracovanie
- ošetrenie konca súborov; zatvorenie súborov.

Spôsob odovzdávania viet na spracovanie môže byť riešený viacerými spôsobmi v závislosti od počtu súborov a ich charakteru.

Pre lepšiu orientáciu uvádzame pseudokód hlavného užívateľského programu využívajúceho modul RDN.

```
hlavný program : PROC OPTIONS /MAIN/
          :          /* deklarácie premenných
          .           hlavného programu */
          :
          .
% INCLUDE dclrdn    /* deklarácie premenných pre RDN */
% INCLUDE DATA      /* programový kód DATA */
          :
          .
CALL RDN
DO WHILE /nie_koniec súborov/
          .
```

```
SELECT /TYP_DVOJICE/
      CASE/1/: spracovanie vety zo súboru Z
      CASE/2/: spracovanie vety zo súboru K
      CASE/3/: spracovanie vety zo súboru Z
               spracovanie vety zo súboru K
ENDSELECT
IF koniec_skupiny
      THEN spracovanie pre koniec skupiny - z hľadiska
           užívateľa
CALL DATA
END
END hlavný program
```

Už na tomto príklade pri porovnaní s pôvodným Jacksonovým riešením názorne vyniká prednosť našej metódy. V prípade viacerých vstupných súborov sú rozdiely oveľa markantnejšie.

1. Formulácia príkladu /viď [1] - odstraňujúci príklad/

Kmeňový súbor obsahuje záznamy o odberateľoch a je usporiadaný vzostupne podľa čísiel odberateľov. Vstupný sekvenčný súbor obsahuje jednotlivé položky faktúr, usporiadané vzostupne podľa čísiel odberateľov a čísiel faktúr. Pre daného odberateľa sa môže vyskytnúť niekoľko faktúr, pre iného nemusí byť žiadna. Z dôvodu chýb pri obstarávaní dát sa môžu vyskytnúť aj faktury, znejúce na odberateľa, číslo ktorého sa v kmeňovom súbore nenájde. Tieto položky je treba vypísať do protokolu, ostatné faktúry treba vytlačiť.

2. Postup riešenia

Prostredkom oddelenia funkcií riadenia a spracovania dát je logická štruktúra abstraktného súboru VSTUP /viď. obr.2/. K jej znázorneniu sú použité modifikované Jacksonove štruktúrne diagramy /viď. obr.1/. Základom abstraktného súboru VSTUP sú abstraktné dátové jednotky vstupných dát na úrovni odberateľa. Každá jednotka predstavuje alebo /selekcia/ samostatný kmeňový záznam odberateľa bez faktúry, alebo záznamy faktúr bez odberateľa./t.j. pre fiktívneho odberateľa/ alebo záznam odberateľa s príslušnými faktúrami a ich položkami.

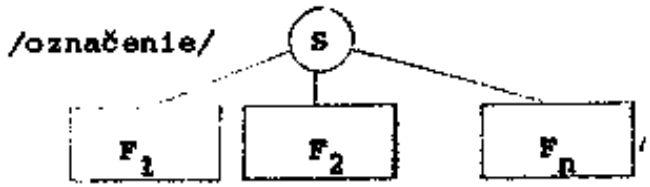
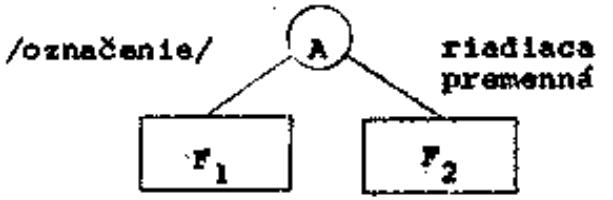
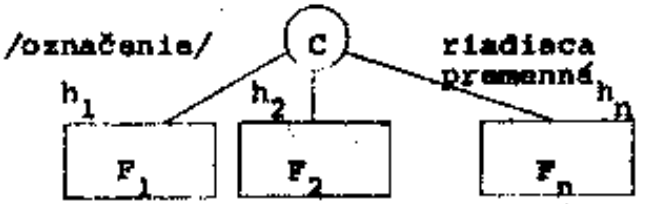
Dynamickú rekonštrukciu abstraktného VSTUPU v priebehu spracovania umožňujú programové moduly RDN a LEND. Modul RDN sekvenčne číta obidva vstupné súbory a po každom vyvolaní odovzdáva dátovú štruktúru, obsahujúcu riadiace premenné NEODAT a TYP-VSTUPU, KLUC a v súlade s hodnotou riadiacej premennej TYP-VSTUPU kmeňový a zmenový záznam. Vyhodnocovanie zmien úrovní spracovania vykonáva programový modul LEND. Tento modul pri zistení zmeny čísla odberateľa, resp. čísla faktúry, nuluje riadiace premenné NEOODB a NEOPAK.

Úloha vytvorenia týchto modulov môže byť riešená samostatne. Za predpokladu existencie modulov riadenia abstraktného súboru VSTUP /obr. 4/ dostávame základnú logickú štruktúru programu a tým aj základný algoritmus riešenia v troch krokoch. Ďalší postup návrhu, jeho prevod do pseudokódu a zostavenie konečného algoritmu špecifikovaním obsahu abstraktných funkcií vzhľadom na požiadavky výstupu je zrejmý a netreba ho rozvádzať. /viď. obr. 3/1-3/3/.

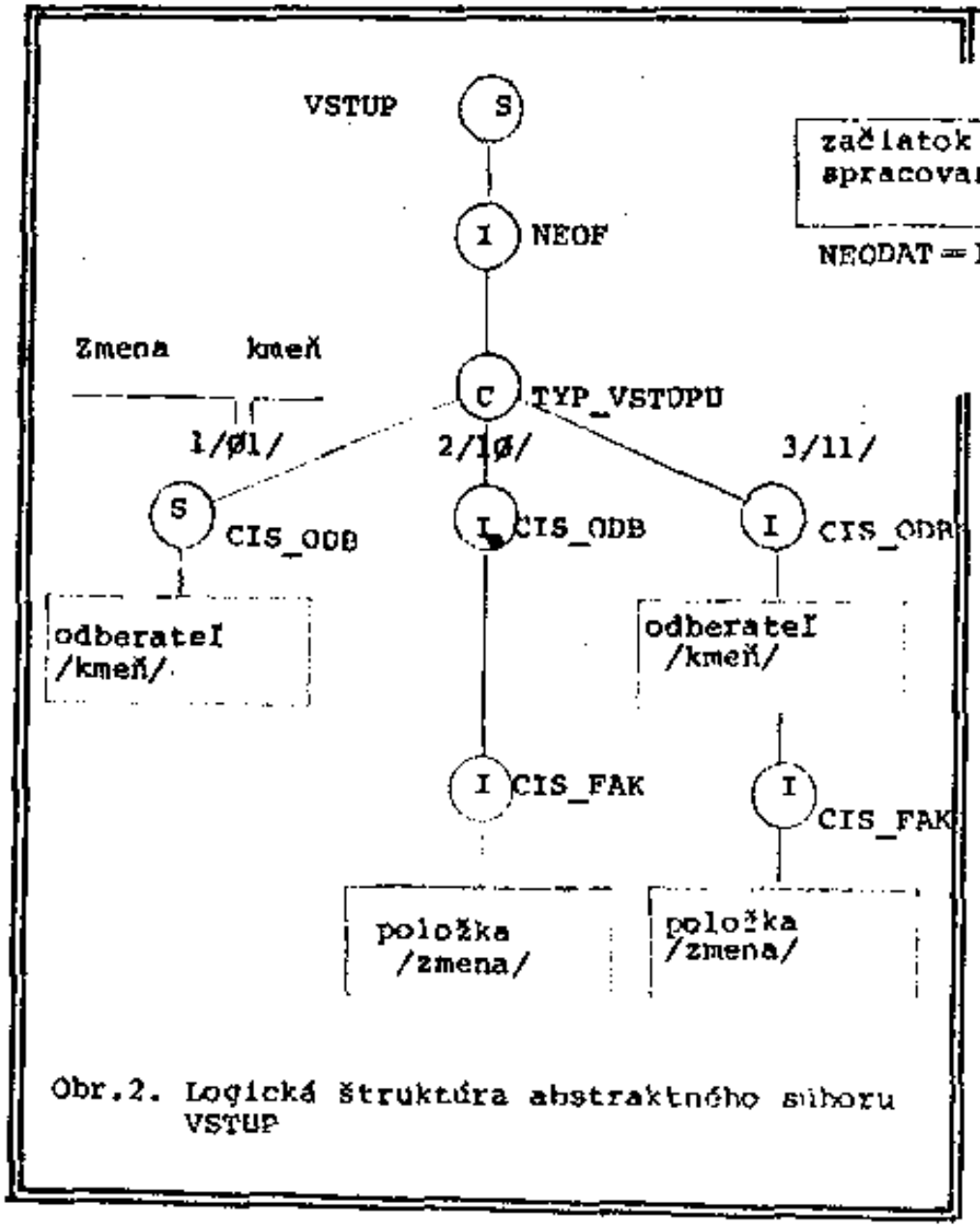
Základná štruktúra programu je určená riadiacimi premennými NEODAT - koniec vstupných dát, NEOODB - zmena čísla odberateľa, NEOFAK - zmena čísla faktúry a TYP-VSTUPU určujúceho dodávané záznamy pre danú hodnotu premennej KLUC /1- kmeňový záznam, 2-zmenový záznam, 3 - kmeňový záznam a zmenový záznam/.

Pri prvom čítaní súboru VSTUP sa vyvoláva iba procedúra RDN, lebo riadiace premenné príslušných úrovní spracovania CIS-ODB a CIS-FAK nemajú v tomto kontexte logický význam.

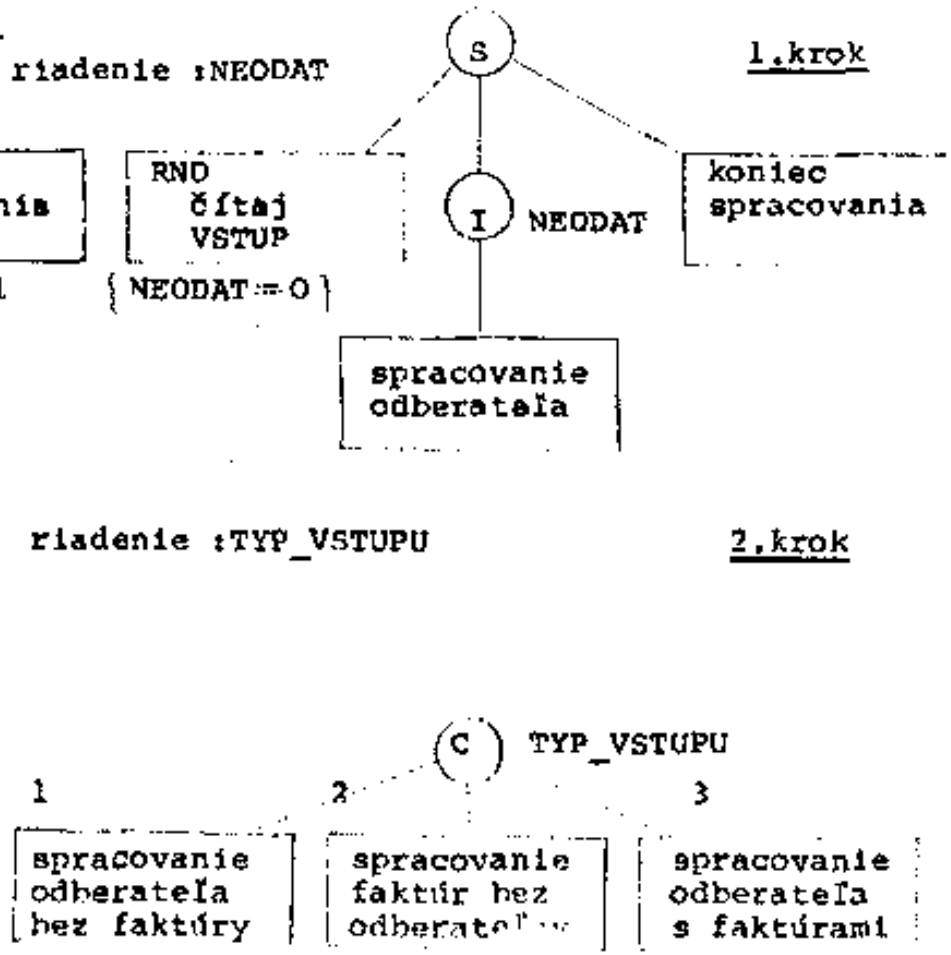
1/ M.A. Jackson, PRINCIPLES of program desing. Academic Press, New York 1975, str. 6 - 9

základné logické štruktúry	Schéma	Pseudokód
1. S e k v e n c i a /postupnosť/		/označenie/: F_1 F_2 . . F_n
2. S e l e k c i a /alternatíva/		/ označenie/: IF riadiaca premenná THEN F_1 ELSE F_2
3. S e l e k c i a /obecný výber/		/označenie/: SELECT riadiaca prem. CASE/h_1/ : F_1 CASE/h_2/ : F_2 : CASE/h_n/ : F_n EndSELECT
4. I t e r á c i a /opakovanie/		/označenie/: DOWHILE /riadiaca prem/ F END

Obr.1 Modifikované J a c k s o n o v e Štruktúrne diagramy /klasické riadiace štruktúry/



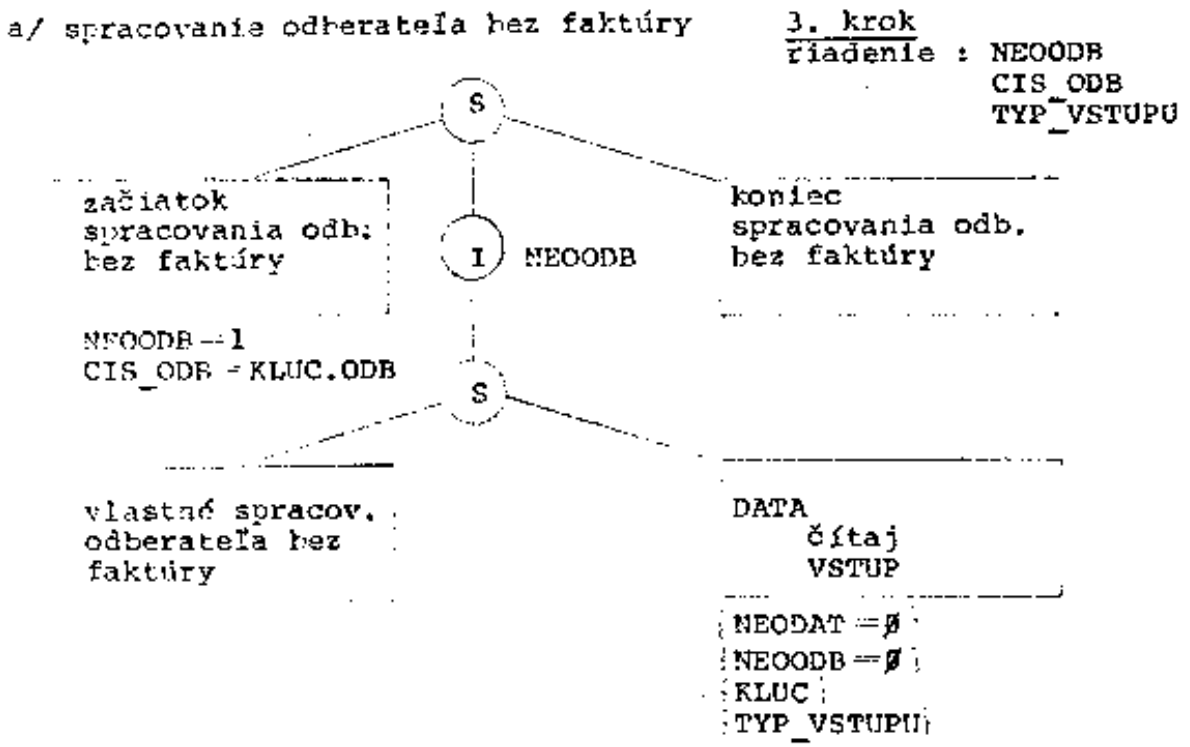
Obr.2. Logická štruktúra abstraktného súboru VSTUP



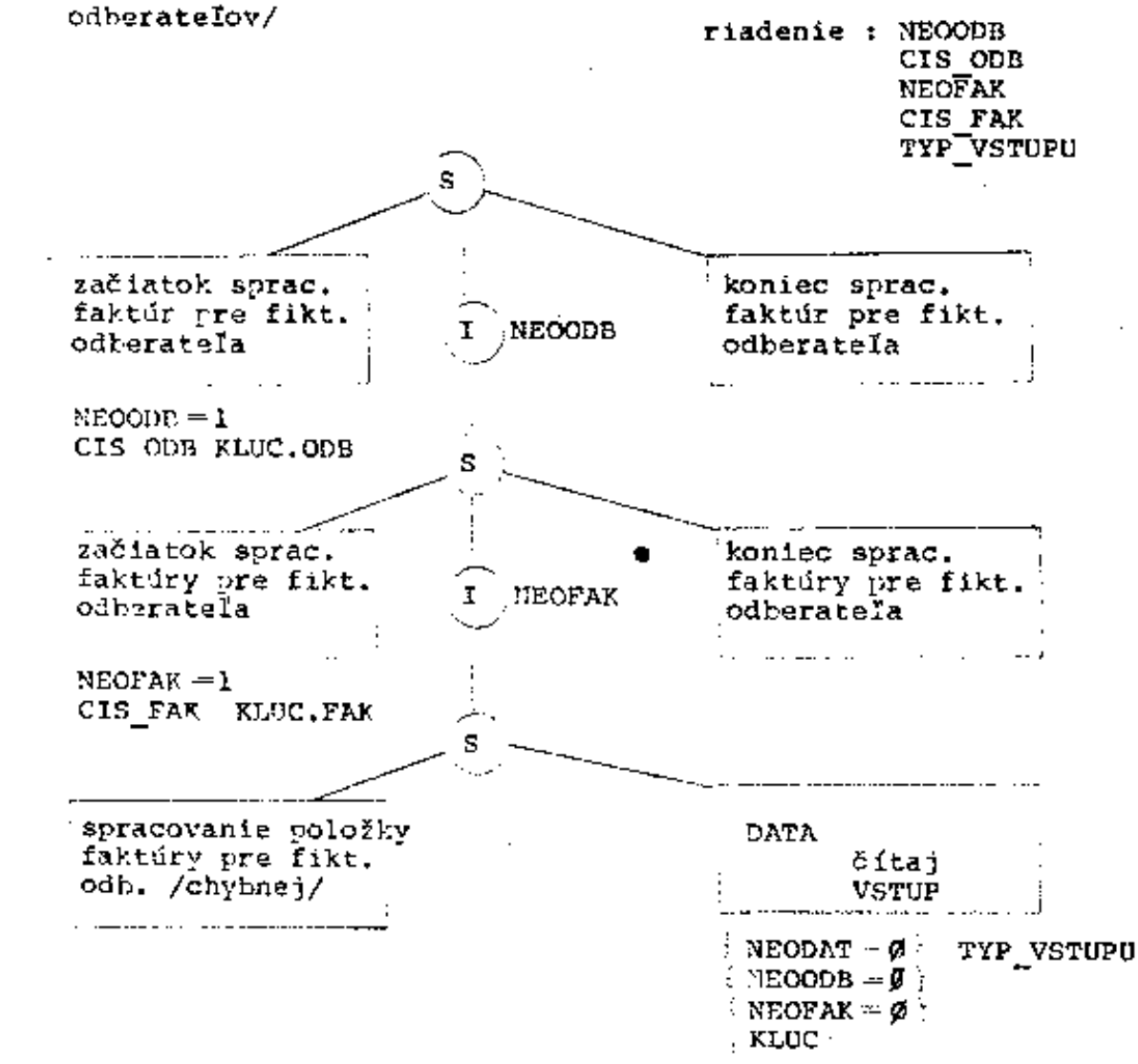
Obr. 3/1 Postupný vývoj logickej štruktúry programu

Obr. 3/2

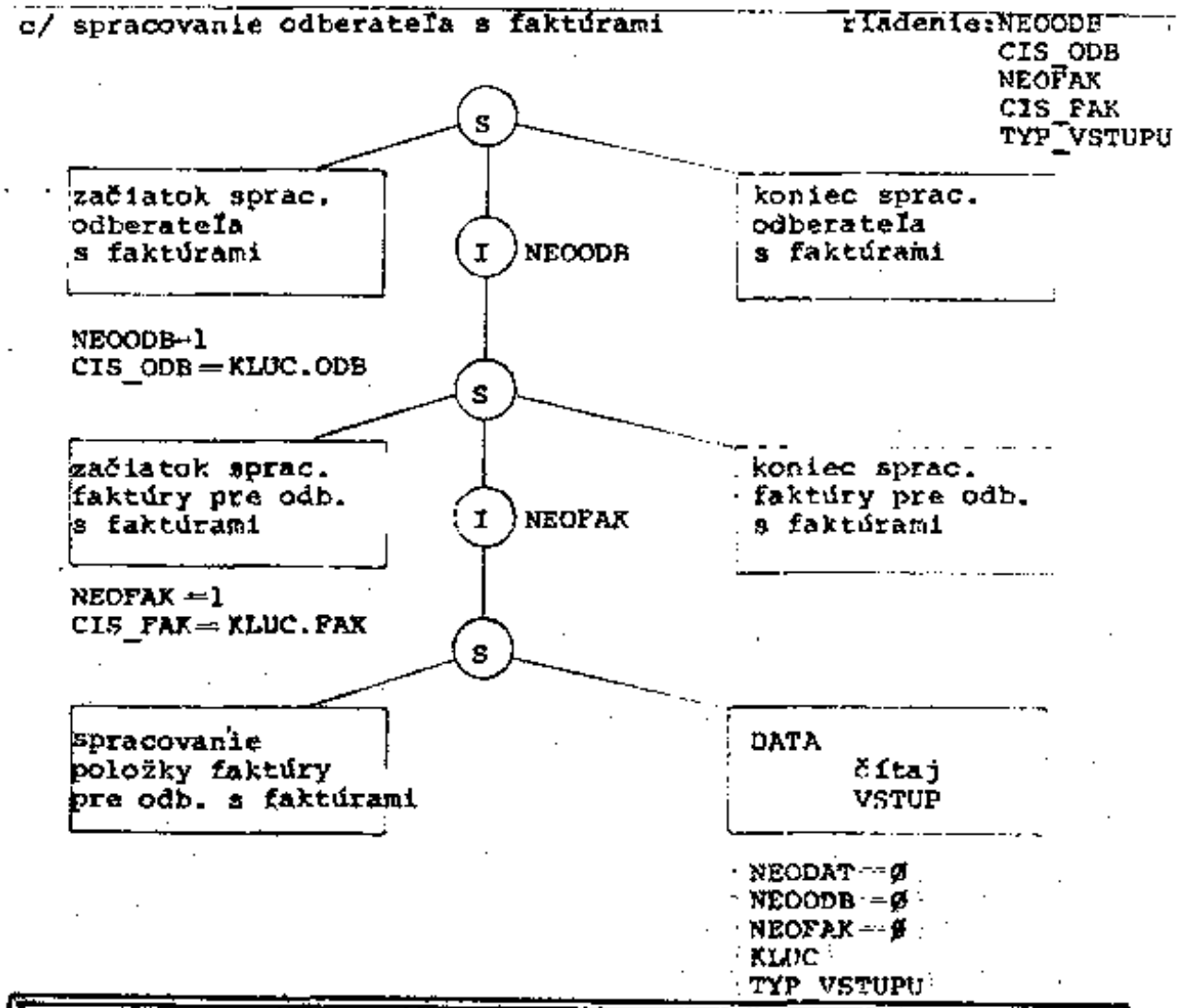
a/ spracovanie odberateľa bez faktúry

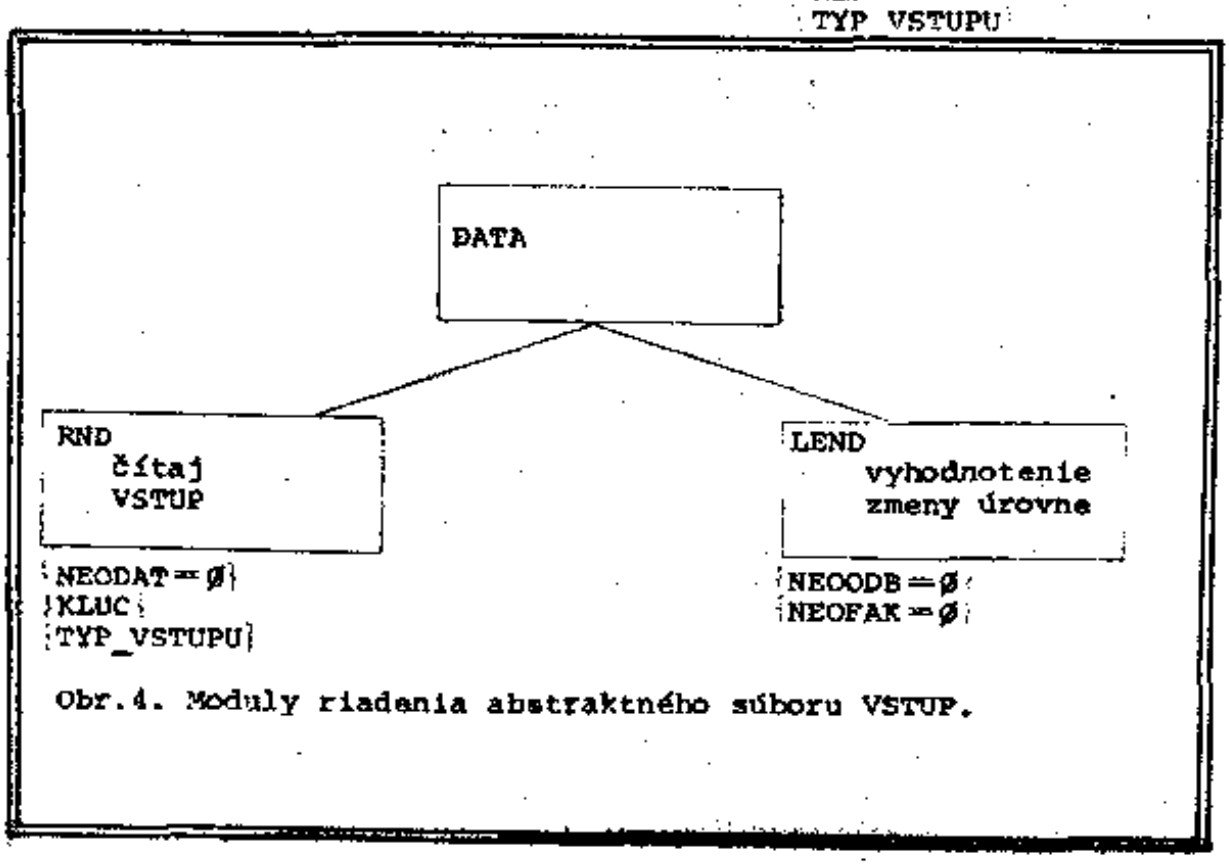


b/ spracovanie fiktívneho odberateľa /t.j. faktúr bez odberateľov/



Obr. 3/3





Obr.4. Moduly riadenia abstraktného súboru VSTUP.