O PROBLÉMECH PARAMETRIZACE

Vilém Novák, Alexander Winkler, Svatopluk Zdebski

V příspěvku jsou zobecněny zkušenosti autorů s budováním parametrických systémů a jejich provozem. Ukazuje se, že parametrizace je výhodná, neboť minimalizuje nároky na údržbu programů, jejich počet a provoz. Plně parametrizované systémy se vyznačují značnou portabilitou a stávají se tak typovými projekty. Jsou diskutovány další obecné problémy parametrizace, provedena typizace parametrizovaných objektů a parametrů. Rozebírají se otázky programové analýzy parametrových struktur a problematika parametrizace automatizovaných systémů.

1. Hlavní problémy parametrizace

Parametrizace vznikla v podstatě na základě sdružování podobných problémů do určitých tříd, kdy určitou třídu se autor programu snažil řešit jedním programem.

Přínosem parametrizace programů je to, že program je externě ovladatelný v těch oblastech, v nichž dochází v závislosti na objektivní realitě k požadavku na změny algoritmu, které není možné zajistit pouze na úrovni příkazů a testování vnitřního stavu prováděného algoritmu. Snaha řešit tyto požadavky vedla v historii programování k tzv. „obecným" programům, které stejně nikdy nepostihovaly všechny možné požadavky a přitom mnoho času spotřebovaly na testování vnitřního stavu, v němž se nacházely. Požadavek, který nedovedly tyto programy řešit, znamenal vždy jejich modifikaci. Vhodnou parametrizací minimalizujeme změny programů a zvyšujeme jejich přizpůsobivost změně vnější reality. S tím je spojen problém účelné míry parametrizace.

Existuje zde velmi silný vztah mezi obecností parametrů a jejich složitostí. Zjednodušeně lze říci, že existuje-li jednoduchá a srozumitelná soustava parametrů pro 90% řešených problémů, zatímco s řešením zbývajících 10% by vzrostla její složitost, je výhodnější pro oněch 10% naprogramovat speciální programy. Velmi často se totiž stává, že autor programu, veden snahou po zvládnutí co nejširší třídy problémů, vybuduje systém parametrů s vysokými nároky

na zvládnutí. Navíc takto konstruované produkty vytvářejí novou profesi tzv. „kódovačů parametrů", u nichž se výrazně zvyšuje závislost na určité konfiguraci počítače, pro kterou byl tento produkt vytvořen.

Jako velmi efektivní se jeví pro řešení výše uvedeného problému použití tzv. uživatelských exitů. Pod tímto pojmem rozumíme speciální modul připojitelný v určitém místě k parametrizovanému objektu, který řeší speciální problémy nařešené parametrickým programem.

Druhým problémem parametrizace je co parametrizovat. Parametrizace musí vždy vycházet ze skutečné potřeby vnější reality, kterou daný program modeluje. Samoúčelná parametrizace vede spíše ke zvýšení složitosti programu a k vyšším nárokům na zadávání zpracování.

Pro parametrizaci je vhodné zvolit vždy nepříliš širokou třídu problémů, které se často opakují s poměrně malými odlišnostmi. Právě tyto odlišnosti řešíme parametrizací. Jedním z velmi úspěšných parametrizovaných programů je program pro třídění. Jeho úspěšnost je dána právě jednoduchostí funkce a parametrů. Je na čtenáři posoudit, jak by se zvýšila jejich složitost, pokud bychom chtěli souběžně řešit problémy konverze a výběru zpracovávaných dat.

Tyto dva základní problémy vedou k speciálnějším, jako je např. technika zpracování parametrů, unifikace parametrů v rámci systému a pod. Některým z těchto speciálních problémů jsou věnovány následující kapitoly.

2. Základní pojmy a typologie

Pro další úvahy je zapotřebí vymezit určité základní pojmy a provést kategorizaci jednotlivých typů parametrizovaných objektů a jejich parametrů.

Hledisko, pod nímž budeme definovat základní pojmy a provádět kategorizaci, je hledisko funkční.

Vyjděme z pojmu modul/podprogram/. Modul je objekt, který je jednotou formalizovaného zápisu algoritmu, vnitřních dat modulu a spojovací oblasti umožňující interakci modulů ve finálním produktu /programu/. Vnitřními daty modulu /řídící data algoritmu/ rozumíme data, jejichž počáteční nastavení je významné pro vlastní algoritmus /konstanty, výhybky, tabulky/. Program je takový modul, který již není součástí jiného modulu a je schopen samostatného zpracování.

Chápeme-li programový objekt /program, modul/ jako realizaci funkce $y = f/x,\lambda/$, kde x jsou vstupní data, y výstupy objektu, f je algoritmus a λ jsou řídící data algoritmu, pak parametrizovaným objektem budeme chápat objekt, kde f,λ a formát x, y mohou být měněny na základě jistých externích deklarací. Parametrem pak rozumíme prvky těchto externích deklarací, které vstupují do objektu z jeho okolí a modifikují algoritmus, jeho řídící data a formáty vstupů a výstupů.

Přistoupíme nyní ke kategorizaci typů parametrizovaných objektů.

Objekty, u kterých dochází po zadání parametrů k vytvoření algoritmu i vnitřních dat nazýváme objekty s generovaným algoritmem. Do této třídy patří například některé dotazovací jazyky. Program je zde generován buď ve zdrojové formě některého z programovacích jazyků a pak dynamicky přeložen, nebo je generován přímo v proveditelné formě a odpadá fáze kompilace.

Druhou skupinou jsou objekty s modifikovaným algoritmem. Zde bývá zpravidla algoritmus modifikován na základě změny jeho řídících dat. Vlastní program je pak pouze manipulátorem, který je řízen strukturou vnitřních dat. Typickým představitelem jsou externě ovládané realizace tabulek /6/.

Další skupinou jsou objekty se stavebnicově konstruovaným algoritmem. Zde jsou na základě parametrů dynamicky kompletovány programy v době jejich provádění z připravených modulů uložených v proveditelné /load/ formě v některé knihovně.

Čtvrtou skupinou jsou objekty s inicializací předem připravených algoritmů. Zde jsou dílčí algoritmy trvalou součástí programů a jsou pomocí parametrů vyvolávány k provedení. Příkladem může být program, který má zabudováno podtrhávání řádků v tiskové sestavě a pomocí parametrů udáme číslo řádku, který chceme podtrhnout.

Poslední skupinou jsou objekty v nichž pomocí parametrů provádíme výběr dat ke zpracování a tato poskytujeme buď beze změny nebo pomocí formátových parametrů u nich měníme strukturu a vnější formu. Tyto programy nazýváme programy se selekcí dat. Příkladem mohou být konverzní programy se zabudovaným výběrem dat.

Programy, které se vyskytují ve skutečnosti, jsou obvykle různými kombinacemi uvedených typů.

Z uvedeného rozdělení programů vyplývá rozdělení parametrů.

Parametry pro programy s generovaným algoritmem nazveme generujícími parametry. Jde o relativně nejsložitější parametry, jejichž zadávání předpokládá poměrně vysokou odbornou úroveň zadavatele.

Parametry pro programy s modifikovaným algoritmem nazveme parametry modifikačními. Na základě poměrně jednoduché a uživatelsky čtivé formy mohou realizovat složité modifikace algoritmu programu, který je používá.

Parametry pro programy se stavebnicově konstruovaným algoritmem jsou zpravidla jednoduché, ale práce s nimi vyžaduje znalost algoritmů realizovaných jednotlivými moduly. Tyto parametry nazveme sestavujícími parametry.

Parametry pro čtvrtou skupinu programů nazveme parametry inicializačními.

Poslední skupinou parametrů pro programy se selekcí dat nazvene nazveme parametry selekční nebo formátující podle jejich funkce.

Každý program nebo modul může používat více typů parametrů. Vstup těchto parametrů je možný z různých zdrojů.

- vstup přes operačním systémem určenou oblast /např. JCL parametry/
- vstup přes běžná vstupní média /nejpoužívanější jsou děrné štítky nebo členěné soubory na magnetických discích/
- interaktivní vstup přes operátorské pracoviště např. u speciálních databázových programů, nebo přes terminál /např. interaktivní navrhování displejových panelů, výstupních sestav atd./

Každý z výše uvedených typů parametrů má svou externí a interní formu. Externí forma parametrů by měla být charakterizována vysokým stupněm srozumitelnosti pro uživatele /velmi často i neznalého programování/. S růstem uživatelského komfortu rostou však i nároky na převod externí formy na formu interní, která je zpracovávaná počítačem.

3. Programové analýza parametrových struktur

Jedním ze stěžejních problémů, se kterým se setkáváme při realizaci parametrizovaných objektů, je otázka vhodného způsobu jejich ovládání. Věnujme proto nyní zvýšenou pozornost problematice návrhu, analýzy a konverze tzv. parametrových struktur.

4.1. Základní pojmy

V kapitole 2 jsme definovali parametrizovaný objekt jako objekt, jehož složky mohou být měněny na základě externích deklarací.

Parametrovou strukturou budeme dále rozumět souhrn všech externích deklarací vstupujících z jednoho zdroje /např. obsah PARM-pole, parametrový člen členěného souboru, děrné štítky atp./. Nebudeme si zde ovšem všímat všech možných typů parametrových struktur, nýbrž pouze takových, které jsou nástrojem člověka /uživatele programového objektu/ k definování jeho požadavků na zpracování. Za parametrovou strukturu lze totiž obecně považovat i seznam parametrů určitého modulu, který je podřízen jinému programovému objektu. Budeme pak hovořit o programové parametrové struktuře, vyjadřující požadavky programových objektů a uživatelské parametrové struktuře /přídomek uživatelská budeme vynechávat, jestliže nemůže dojít k omylu/.

Každá parametrová struktura má svou externí a interní formu. Externí forma slouží uživateli, nebývá však vhodné pro vlastní programové zpracování. Jedním z hlavních problémů realizace parametrizovaných objektů je problém přechodu k interní formě parametrové struktury. Každá z uvedených forem má své specifické způsoby zobrazování, kterými se budeme zabývat v následujících paragrafech.

4.2. Externí forma parametrové struktury

Parametrová struktura obráží jistý výsek reality, který lze dekomponovat v řadu reálných entit, kterým odpovídají parametrové entity - parametry /popis atributů souboru, záznamu, tiskového řádku, formátu obrazovky atp./. Obecně je parametr definován svým jménem /lze „zamlčet", je-li parametrová struktura homogenní vzhledem ke kvalitě parametrů/ a definuje řadu atributů - subparametrů /popis záznamu::=formát záznamu, délka záznamu, poloha klíčové položky atp./.

Subparametr je určen opět jménem a určuje jistou hodnotu. Lze je klasifikovat dle

a/ typu identifikace /jména/

- klíčové subparametry - mají jméno i hodnotu a jejich postavení v parametru je dáno jménem /BLKSIZE=7200, BLKSIZE IS 7200 BYTES/
- poziční subparametry - jejich postavení je dáno pozicí a jméno je zamlčeno

- samodefinující subparametry - jsou definovány svou hodnotou, která je jednoznačně určuje. Lze je opět chápat jako klíčové se zamlčeným jménem /DECK a NODECK jsou hodnoty parametru PUNCH - jméno se neuvádí/

b/ typu hodnoty

- jednoduché subparametry: 1/ skalární - hodnota má jednu složku
  2/ vektorové resp. seznamové - posloupnost pevného nebo proměnného počtu složek téže kvality a formátu
- strukturované subparametry: hodnota je dána posloupností složek různé kvality a formátu.

Příklady: MEMBER=KAKNIH/MODUL/ ... strukturovaný subparametr /hodnota určuje jméno knihovny a jméno členu/

MEMBERS ARE MOD1, MOD2 ... seznamový subparametr /hodnotou jsou jména dvou členů/

LENGTH IS 54 BYTES ... skalární subparametr

Parametrovou strukturu lze rozčlenit do bloků /nezávislých/ parametrů. Mezi bloky existují vztahy podřízenosti a nadřízenosti /obsah nadřízeného bloku vymezuje obsah podřízeného bloku/.

Příklad:

a/	b/	
FUNKCE=SELECT	FUNKCE=CHANGE	blok 1
MEMBERS=M1,M2	NAME=M1/N1/	blok 2

Blok FUNKCE /výběr nebo změna názvu/ určuje tvar následných parametrů /seznam vybíraných členů nebo seznam dvojic „nové jméno - staré jméno"/. V rámci bloku lze hovořit o typu výskytu parametru:

a/ parametry s jedinečným výskytem - např. popis souboru

b/ parametry s mnohonásobným a kvalifikovaným výskytem - např. popis záznamů souboru. Každý výskyt je blíže určen kvalifikací /buď prefixem názvu parametru nebo hodnotou kvalifikujícího subparametru/

Příklady: INPUT RECORD FORMAT IS FIX
RECORD TYPE=1, FORMAT=FIX

3.3. Popis externí formy parametrové struktury a její realizace

Abychom byli zcela přesní, musíme rozlišovat popis a realizaci parametrové struktury. Popisem rozumíme vhodný zápis syntaxe posloupnosti parametrů, subparametrů a hodnot, realizací pak konkrétní vyjádření našeho požadavku, vyhovující popisu, dosazující konkrétní hodnoty, vybírající možnosti a zamlčující implicitní hodnoty.

K vyjádření syntaxe parametrových struktur lze s výhodou použít prostředků používaných pro zápis syntaxe programovacích jazyků, jako jsou BNF /Backus-Naurova forma/ a jazyky produkcí. Specifika parametrových struktur nás pak vede k tomu, abychom v BNF vyjadřovali i skutečnosti běžně neuváděné, jako jsou počty opakování znaků a dosazování implicitních hodnot exploatovaných v případě absence nepovinných subparametrů a jejich hodnot. V jazycích produkcí lze obdobně zjednodušeně zaznamenat „sémantiku" zpracování - jde zde zpravidla o standardní konverze do interní formy.

Jako příklad nám může sloužit zápis v rozšířené BNF:

$$\{9\}^{1,3}\,[\{X\}^{*3}]('vvv')$$

kde $\{9\}^{1,3}$ označuje opakování maximálně 3 numerických znaků oddělených čárkou

$[\{X\}^{*3}]('vvv')$ označuje nepovinný výskyt maximálně 3 alfanumerických znaků neoddělených oddělovačem /* má speciální význam/. V případě absence se dosazuje implicitní hodnota 3 mezer.

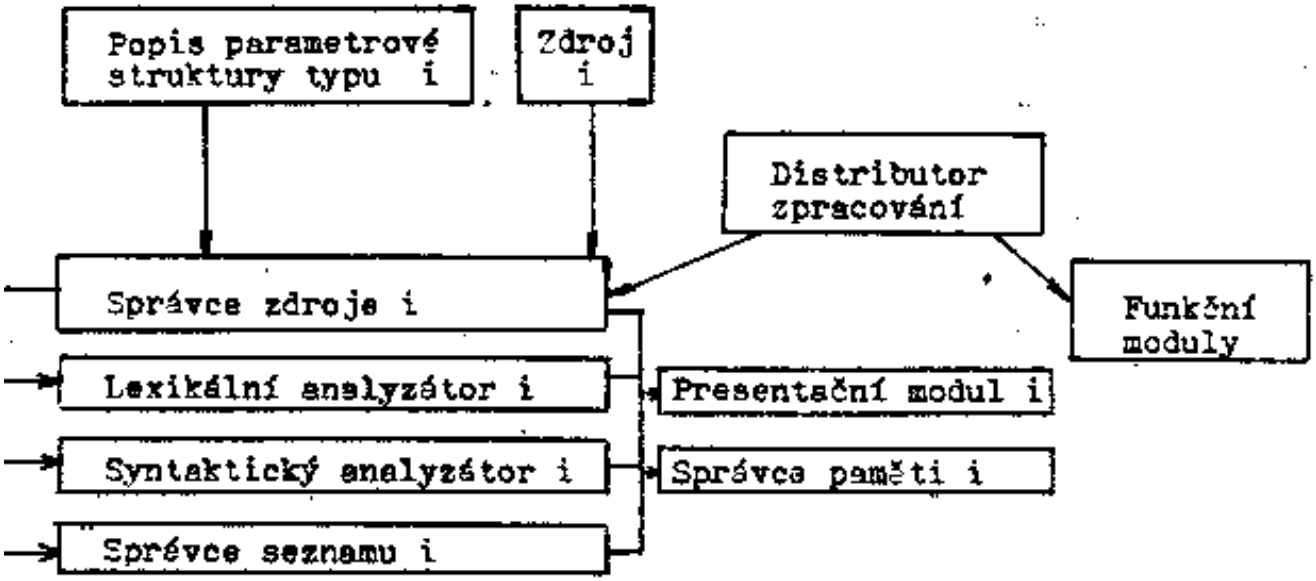
Problematice popisu parametrové struktury je nutno věnovat zvýšenou pozornost, neboť samotný popis může být významným vodítkem pro tvorbu lexikálních a syntaktických analyzátorů a v mezním případě může být přímo vstupem pro obecné analyzátory /viz např. jazyk produkcí systému TWS realizovaného ve VVS Bratislava/.

S problematikou syntaktické analýzy je úzce spjat problém sémantiky parametrových struktur. Lze jej řešit aplikací příslušných sémantických modulů /viz TWS/ nebo je řešen souběžně se syntaktickou analýzou na základě popisu struktury rozšířenou BNF. Výsledkem sémantické analýzy by měla být vhodná interní parametrová struktura, jejíž podoba je dána uživatelským softwarem /řeší se formou lineárních datových struktur, seznamů, stromů nebo plexů/, přičemž tvar da-

tových elementů je dán přímo rozšířenou BNF /doplňují se implicitní hodnoty nepřítomných subparametrů a hodnoty se normalizují do maximálních délek/. K interní parametrové struktuře by již měl parametrizovaný objekt pouze přistupovat, nikoliv ji analyzovat. Otevřenou ovšem zůstává otázka obsahových a kontextových kontrol, kterou musí řešit parametrizovaný objekt sám.

3.4. Konverze externí formy parametrové struktury do interní formy

Načrtněme si nyní vhodnou strukturu parametrizovaného objektu na jejímž základě můžeme diskutovat o problémech konverze a přístupu k hodnotám subparametrů



Externí forma parametrové struktury může vstupovat z různých zdrojů, na obrázku je z úsporných důvodů zaznamenán pouze jeden.

Distributor zpracování zajišťuje styk s parametrovou strukturou, tj. vyvolává a specifikuje její analýzu a vyžaduje její různé prvky. Ve fázi inicializace analýzy definuje seznam parametrů přípustných v daném bloku /na základě posloupnosti zpracovávaných bloků a znalosti globální funkce parametrizovaného objektu/. V exploatační fázi získává jednotlivé hodnoty na základě dotazu specifikovaného:

- způsobem přístupu: DIRECT /v případě mnohonásobných a kvalifikovaných výskytů téhož parametru, které jsou v interní formě reprezentovány tabulkou nebo stromem/
  FIRST /v případě požadavku na skalární hodnotu nebo první složku vektorové hodnoty/

NEXT /v případě přístupu k následným složkám vektorové hodnoty/

- určení parametru - uvede se jméno parametru
- určení kvalifikátoru - uvede se název kvalifikátoru /INPUT, OUTPUT viz př.v 3.2/ nebo hodnota kvalifikujícího subparametru /např.klíčová položka pro výběr hodnoty z číselníku/
- identifikace subparametru - uvede se název subparametru

Lze jistě odvodit situace, ve kterých mohou jednotlivé složky specifikace chybět. Obecně však je toto určení úplné.

Na základě požadavku získá distributor hodnotu a je indikován příznak vyjímečné situace /např.nalezena poslední hodnota vektorové položky, hodnota nenalezena a pod./. Z takto získaných hodnot se tvoří programová parametrová struktura pro vyvolání funkčních modulů. Distributor provádí rovněž globální obsahové a kontextové kontroly. Nutno poznamenat, že obě fáze práce distributoru mohou být zčásti přiřazeny funkčním modulům.

Správce zdroje zpracovává jako celek jeden blok parametrů. Je-li použit ve fázi analýzy, vyvolává příslušné lexikální a syntaktické analyzátory a zapojuje vytvořené interní datové struktury parametrů pomocí modulu správy paměti do globálního seznamu parametru v bloku. Provádí kontroly na úrovni identifikace parametrů a pomocí prezentačního modulu vytváří opis externí formy parametrové struktury s indikací chyb. Ve fázi exploatace zpřístupňuje pomocí správce seznamu jednotlivé hodnoty.

Lexikální analyzátor vyčleňuje jednotlivé lexikální jednotky parametrové struktury a signalizuje chyby zjištěné na této úrovni.

Syntaktický analyzátor provádí na základě popisu parametrové struktury /je dána buď logikou modulu, nebo vstupuje jako formální zápis/ syntaktickou analýzu parametru a vytváří zpravidla jeho interní reprezentaci /pokud se nepoužívají sémantické moduly/. Tato interní reprezentace je však pro vlastní parametrizovaný objekt nevýznamná. Je přitom použit buď obecný analyzátor, nebo pro každý parametr existuje zvláštní analytický modul. Využívají se známé metody syntaktické analýzy. Chyby jsou prezentovány pomocí prezentačního modulu a složitější datové struktury popisující parametr jsou ukládány do paměti pomocí modulu správy paměti.

Správce seznamu na základě požadavků vybírá z vytvořených interních struktur hodnoty subparametrů a předává je nadřízeným modulům.

Prezentační modul a modul správy paměti mají pomocný charakter a jejich funkce vyplývá z předchozího textu.

Souhrnu modulů podřízených distributoru zpracování a určených ke zpracování externí formy parametrové struktury z jednoho zdroje říkáme parametrový modul. Jeho vyčlenění z celého komplexu parametrizovaného objektu umožňuje

a/ jeho mnohonásobné využití v parametrizovaných objektech pracujících s jednotnou nebo podobnou parametrovou strukturou

b/ vyčlenění analytické fáze mimo zpracování parametrizovaného objektu. Parametrová struktura pak není interpretována nýbrž kompilována do uchovatelné formy. Tato forma je vstupem pro zjednodušený parametrový modul, který obsahuje pouze modul správy seznamu a zvláštní modul obnovy interní struktury

3.5. Shrnutí problematiky

Problematika návrhu parametrových struktur, jejich analýzy a exploatace je jedním z nejzávažnějších problému tvorby parametrizovaných objektů. S externí formou parametrové struktury nepracují specialisté v oblasti počítačového zpracování dat a proto by se měla blížit běžnému způsobu vyjádřování. Nevystačíme pak s primitivním způsobem zadávání parametrů jako posloupnosti hodnot s přesně danou pozicí, délkou a typem zobrazení. Musíme vytvářet struktury mnohem složitější, jejichž analýza, konverze a exploatace je nezřídka komplikovanější než realizace vlastních funkčních modulů. Indukovaným důsledkem je pak i růst kvalifikace programátorů, kteří si musí osvojovat metody zvládnutelné dříve pouze vysoce erudovanými odborníky.

4. Systémy s parametrizovanými prvky

V této části se budeme zabývat problematikou budování systémů, v nichž jsou některé /popř.všechny/ prvky systému parametrizovány. Přitom pod pojmem prvek budeme rozumět modul, program nebo pracovní chod /např.JCL-procedura pro OS IBM/.

Proč parametrizované systémy

V praxi se lze setkat se dvěma extrémy:
- zcela neparametrizované systémy, které mají všechna vnitřní data naprogramována „natvrdo" ve všech prvcích
- systémy v nichž jsou veškeré prvky parametrizovány

První extrém bývá často průvodním jevem agendových systémů. Např. při požadavku dvou sestav shodného tvaru, avšak s navzájem různým výběrem podniků jsou pro obě vypracovány zvláštní programy, i když by šlo výběr podniku zadat parametrem. Takové systémy pak nabývají na rozsahu a vznikají tradiční potíže s nízkou efektivností a nepružnou reakcí na změny nebo rozšíření /cyklus: požadavek uživatele - specifikace - programování - ladění - rutina = cca 2 měsíce/. Z toho je vidět, že si praxe sama vynucuje konstrukci parametrizovaných prvků v systému. Projektant se však setkává s problémem, do jaké míry parametrizovat, které části systému a jak navrhovat parametry, aby byly co nejjednodušší a přitom plnily všechny požadované funkce. Řešení je vždy kompromisem, pro který je třeba mít cit a který vede k tomu, že projekce se stává do značné míry uměním.

4.2. Jak parametrizovat

Základní požadavek při parametrizaci systému je jednotnost všech parametrů v rámci celého systému. Přitom jednotnost musí být jak z hlediska jejich syntaxe, tak z hlediska obsahu. Při návrhu parametrů je nutno vycházet z datového modelu reality, která je systémem modelována. Je-li datový model dobře zpracován, budou také parametry jednotné. Provozovateli systému se tím značně ulehčí práce, neboť je-li dobře obeznámen s datovým modelem a zná-li vlastní realitu, je mu význam parametrů zřejmý takřka na první pohled a nemusí vyvíjet velké úsilí na to, aby se naučil pracovat se systémem.

Syntaktická jednotnost je pro provoz neméně důležitá. Vstupuje-li například do jednoho programu klíčové slovo POD a do druhého PODNIK, pak je to hrubá chyba, která přinese dříve nebo později potíže při rutinním zpracování. Podobně se stává, že do jednoho programu vstupuje datum rok-měsíc a do druhého měsíc-rok. Tyto chyby zpravidla vznikají, vstupují-li do více modulů v podstatě tytéž parametry a všude jsou separátně analyzovány. Chyby pak vznikají, opomeneme-li promítnout drobnou změnu do všech modulů. Požadavek jednotnosti vede

ve svých důsledcích k modulární stavbě všech prvků systému. Existuje-li pro tytéž parametry jediný modul, který je dynamicky volán všemi moduly, které pracují s týmiž parametry /nebo jejich částí/, nemůže k uvedené chybě dojít. Dynamičnost navíc zajistí trvalou aktuálnost. Musí se však zabezpečit, aby se interní forma parametrů nikdy neměnila. Toho lze docílit např.tím, že nové prvky v daném parametru jsou dodávány v jeho interní reprezentaci až na konec a moduly si přibírají jen tu část, která je zajímá.

Jak bylo řečeno ve druhé části, lze parametry rozdělit do několika typů. Jeví se nám výhodné využívat více zdrojů parameterizace v jednom programu resp.modulu, neboť pak lze rozdělit obsluhu systému tak, že každý pracovník zabezpečuje jen určitou část. Je však nutné hlídat, aby parametry s touž věcnou funkcí byly realizovány jako jeden typ a vstupovaly do modulu stejnou cestou.

K realizaci parametrizovaného systému lze použít také obecných parametrických programů pro řešení různých úloh /např. konverze souborů, obecný tiskový program, obslužné programy pro práci se soubory různých typů aj./. Každý z těchto programů však byl budován zcela obecně, neboť autor nemohl znát prostředí, ve kterém bude program používán. Dále každý autor navrhoval syntaxi parametrů na základě svých znalostí, zkušeností a osobního vztahu k určitému vyjadřovacímu prostředku. Úroveň parametrizace těchto programů bude patrně daleko přesahovat potřeby našeho systému. Očitud plyne, že zabezpečení jednotnosti všech parametrů v rámci systému bude nemožné. Provozní pracovníci budou muset znát širokou škálu všech možností práce se systémem, která se stane komplikovanou a nepružnou. Proto tento postup příliš nedoporučujeme.

Stojíme-li tedy před problémem vybudovat systém, který by měl být parametrický, je vhodné začít od datového modelu. Dále musíme prozkoumat, které parametry budou společné všem programům a modulům v systému. Pak teprve můžeme přistoupit k projekci systému, který musí bude modulární a v němž je vhodné již při projekci zvážit způsob spojování modulů.

4.3. Které_věci_v_systému_parametrizovat

Automatizovaný systém lze rozdělit na tři základní subsystémy: vstupní, zpracovatelský a výstupní. Ve všech lze účelně využít para-

metrizace. Ve vstupní části lze předpokládat využití selektivních parametrů /např.datum pro výběr/ nebo generujících či modifikačních /např.pro generování modulu pro výběr dat ze vstupního souboru/. Ve zpracovatelském subsystému mají velký význam parametry, které mohou být předávány programu ve formě číselníkového modulu /viz 6/. Přitom mohou mít z hlediska typologie různorodý charakter. Lze např. do nich uložit generující parametry, na základě nichž budou vygenerovány některé části algoritmu, dále modifikační parametry, selektivní apod. Výstupní subsystém poskytuje největší prostor pro parametrizaci, zejména pokud jde o tiskové programy. Zde se nejvíce uplatní parametry selektivního a formátujícího typu, popřípadě inicializační /například tvar hlaviček, tiskových řádků, výběr klíčů apod/.

Oblastí, kterou je na základě zkušeností autorů užitečné parametrizovat, je samotná dynamická kompletace programů v průběhu jejich zpracování. Existují parametry, které jsou programům předávány ve formě modulů vygenerovaných off-line na jiném místě /viz např. zmíněné číselníkové moduly/. Může nastat situace, že dva nebo více subsystémů téhož systému používá parametrový modul téže struktury ale s různým obsahem /např.to může být číselník ukazatelů/. Tyto moduly je nutné na knihovně odlišit jmény. Byla-li tato jména součástí programů, pak muselo existovat více verzí téhož programu, což je velmi nežádoucí stav. Elegantní řešení je přiřadit každé skupině parametrových modulů téže struktury jediné interní jméno, kterým program parametrový modul volá. Prostřednictvím sestavujících parametrů se mu dodá informace o tom, jaké externí jméno na knihovně daný modul má a jak je k programu připojen.

Při projekci a tvorbě parametrizovaného systému je nutné hlídat rozsah parametrizace. Při nekoordinovanosti může nastat někdy situace, kdy parametrů je tolik, že orientace v nich je velmi obtížná. Proto je důležité snažit se jejich množství minimalizovat. Stejné oblasti musí být parametrizovány týmiž parametry. Používáme-li např. číselníkové moduly, musíme vždy zvážit, zda se nový problém nedá promítnout do již existujícího číselníku, popřípadě je zahrnout pod jiné existující parametry v systému.

Poslední částí systému, která bývá často parametrizována jsou technologické chody. Protože každý vyšší výpočetní systém má prostředky pro řízení zpracování úloh, jsou jednotlivé pracovní chody

popsány pomocí nich. Vhodný, zejména unifikovaný způsob jejich parametrizace může velmi zefektivnit rutiní zpracování systému. V tomto případě parametrizujeme okolí parametrických programů.

5. Závěr

V tomto příspěvku bylo obecně pojednáno o možných typech parametrizovaných objektů a parametrů, o struktuře analyzátoru parametru a byly diskutovány otázky parametrizace systému a obecné problémy parametrizace vůbec. Poznamenejme ještě, že zkušenosti autorů se odrazily ve výstavbě Automatizovaného informačního systému pro vedení generálního ředitelství koncernu OKD, který je možno chápat jako databankově orientovaný systém s výraznou parametrizací.

Literatura

1. A.V.Aho, J.D.Ullman: The theory of parsing, translation and compiling, ruský překlad, Mir 1978

2. J.M.Foster: Automatická syntaktická analýza ALFA 1973

3. D.Gries: Kompilátory číslicových počítačov SNTL 1981

4. V.Rajlich: Úvod do teorie počítačů MS SNTL 1979

5. J.Valas a kolektiv: Řešení informačního systému pro řídící pracovníky OKR Automatizace řízení, k.ú.o, Ostrava 1980

6. A.Winkler: Uživatelské aspekty číselníkových modulů v oblasti analyticko-projekční a programátorko-provozní. Sborník konference „Programování 80", Dům techniky ČSVTS, Ostrava 1980