Ing.Alexander WINKLER
OKR - Automatizace řízení k.u.o

UŽIVATELSKÉ ASPEKTY ČÍSELNÍKOVÝCH MODULŮ V OBLASTI ANALYTICKO-PROJEKČNÍ A PROGRAMÁTORSKO-PROVOZNÍ

Než se budeme zabývat vlastní problematikou číselníkových modulů, je vhodné si povědět něco o funkci a postavení uživatelského software obecně. Obecný uživatelský software se vytváří obvykle ve dvou případech:
- firma danou softwarovou oblast nepokrývá firemními produkty,
- firemní software je příliš obecný a nebo drahý.

Uživatelský software můžeme rozdělit zhruba do tří typů:
1. typ: Software užívaný v oblasti programátorské (aplikační programátoři).
2. typ: Software, s jehož existencí se počítá již v oblasti analýzy a projekce.
3. typ: Software se speciálním určením (oblast systémového programování, obsluha databází atd.).

Dále budeme hovořit o číselníkových modulech, což je jedna z ukázek úspěšného uživatelského software. S ohledem na výše uvedené rozdělení software jsou číselníkové moduly typickým představitelem software 2. typu.

Co jsou číselníkové moduly?

"Číselníkový modul je ucelená funkční jednotka sloužící k přiřazení výstupní hodnoty (odpovědi) Y nějaké kombinaci vstupních hodnot X1 [,X2...] . Je to počítačová realizace ně-

jaké tabulky." Cituji přímo z rigorozní práce Petra Jiřička "Překladač číselníkových modulů", autora překladače číselníkových modulů v naší organizaci.

Pod označením číselníkový modul tedy neuvažujeme obrovské sekvenční soubory, ale takové funkční celky, které se vejdou při zpracování do vnitřní paměti.

Z programátorského pohledu jsou číselníkové moduly autonomně pracující podprogramy (procedury), které se uvádějí v činnost po naplnění parametrů a vyvolání v nadřazeném programu.

Číselníkové moduly mají velmi dobrou adaptabilitu ve vyvolávajícím programu, protože nejsou závislé na programovacím jazyku. V době překladu číselníkového modulu je totiž možnost vyřešit pomocí parametru pro překladač interface na jednotlivé programovací jazyky. Interface je v podstatě vyřešení způsobu předávání parametrů (existuje např. rozdíl mezi jazyky COBOL a PL/1).

Co přineslo zavedení číselníkových modulů do praxe?

Číselníkové moduly snižují potřebu zásahů do programů za účelem oprav. Například v subsystému účetnictví v naší organizaci došlo po zavedení číselníkových modulů ke snížení počtu oprav o 80%. (Porovnání bylo provedeno mezi roky 1975 (bez číselníkových modulů) a 1976 (po jejich zavedení)). Dále číselníkové moduly řeší dobře kontakt mezi analytiky, programátory a dokonce uživatelem, který je profesně mimo výpočetní techniku. Tato skutečnost je umožněna tím, že číselníkové moduly jsou ve své zdrojové formě velmi dobře pochopitelné a přehledné.

Číselníkové moduly po zavedení nepotřebovaly žádný administrativní zásah nebo nařízení pro své rozšíření. Z původního předpokládaného počtu asi 60 číselníkových modulů se jejich počet ke konci roku 1979 zvýšil na 398. S ohledem na tento bouřlivý růst počtu číselníků bylo zapotřebí řešit otázky

údržby a evidence cestou centralizace. Přitom číselníkové moduly nepotřebovaly pro své rozšíření žádné administrativní zásahy ve formě nařízení používání a následné kontroly. Číselníkové moduly totiž v žádném směru neomezují analytický přístup ani programátorskou individualitu, ale umožňují velmi širokou a tvůrčí aplikaci.

Jak pracuje číselníkový modul?

Práci číselníkového modulu si předvedeme na příkladě ochrany přístupu do datové základny přes displej.
Máme zajistit takovýto úkol:
Jednotlivým uživatelům databáze jsme přidělili třímístné číselné kódy, každý číselný kód umožňuje přístup jen do určitých oblastí (oblastí je 9). V datové základně navíc chceme rozlišit uživatele na uživatele s oprávněním pouze data prohlížet (pomocný kód 2) a na uživatele s oprávněním také data měnit (pomocný kód 1). Kromě opatření v programu zajišťujícím přístup, chceme na displej vyslat zprávu o povolení či nepovolení vstupu do datové základny. Posledním z úkolů je umožnit určitým uživatelům přístup do datové základny bez jakéhokoliv omezení. Jak by vypadala zdrojová forma číselníku, který zajistí všechny výše uvedené úkoly:

X1 - první volací úroveň
(tříciferný kód uživatele)
mnemotechnický název jsme zvolili KODUZ

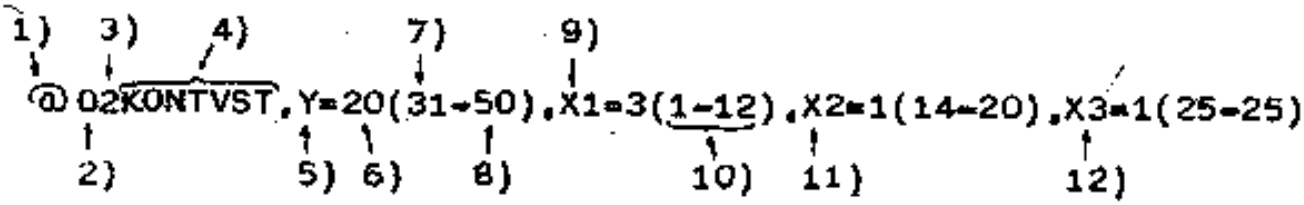
X2 - druhá volací úroveň
mnemotechnický název CISOBL

X3 - forma vstupu
mnemotechnický název ZPUSPR

Y - odpověď dvacetiznakový řetězec, kde na 1. pozici je kód, který použije programátor pro řízení práce přístupového programu a na dalších 19 pozicích je odpověď, kterou pošleme na obrazovku displeje
mnemotechnický název ODPOVED

Každý číselník má úvodní popisový řádek, který je uvozen tím-

to znakem '@'. V tomto popisování řádku jsou všechny podstatné informace pro překladeč.



1) - znak uvozující popisový štítek
2) - 0 znamená, že nebude potlačen tisk vět číselníku při vstupu do generace číselníkového modulu,
1 znamená potlačení tisku správných vět
3) - nabývá hodnot 0-3, udává počet následujících vět, z nichž bude vždy 72 bytů chápáno jako hlavička, která se bude tisknout vždy na každé nové stránce výpisu z generování číselníkového modulu
4) - název číselníkového modulu (max 8 znaků)
5) - odpověď
6) - rozsah
7) - počáteční pozice na DŠ
8) - konečná pozice na DŠ
9) - první volací úroveň
10) - pozn.: zde nemusí být rozmezí rovno rozměru, protože můžeme zadávat rozsahy a výčty
',' pokračování (výčet)
'-' rozmezí
11) - druhá volací úroveň
12) - třetí volací úroveň

KODUZ	CISOBL	ZPUSPR	ODPOVED
...	.	.	1VSTUP NEPOVOLEN
568-572,650	1-5,7,9	1	2VSTUP S PREPISEM
800,950,99.			3VSTUP BEZ OMEZENI
669,951-980	2-3	2	4VSTUP BEZ PREPISU
atd.	atd.	atd.	atd.

Nyní popíšeme slovně způsob práce tohoto číselníku: přes řádek vyplněný tečkami prochází číselník vždy, je to

nejobecnější příklad a pokud se v dalším nenajde žádná kombinace odpovídající hodnotám X1,X2,X3 při vyvolání, pak zůstává v odpovědi odpověď z prvního řádku.

Druhý řádek znamená, že uživatelé s kódem 568 až 572 a uživatel s kódem 650 mají povolen přístup do oblastí 1 až 5, 7 a 9 a mohou provádět přepis hodnot.

Třetí řádek pracuje tak, že uživatelé s kódem 800, 950 a uživatelé s kódem začínajícím 99 a na posledním místě nezáleží mají přístup bez jakéhokoliv omezení.

Analogicky lze odvodit funkci čtvrtého a dalších řádků.

Představíme-li si pro názornost vstup přes displej, který provede uživatel v tomto tvaru:

```
PANEL KONTROLY VSTUPU DO DATOVE ZAKLADNY
----------------------------------------

KOD UZIVATELE:      568
DATOVA OBLAST:      5
ZPUSOB PRISTUPU:    1


PROVED VOLBU VSECH NALEZITOSTI A STISKNI ENTER!
```

Konkrétní uživatel vyplnil svůj kód, oblast, se kterou chce pracovat a formu přístupu k datům. Odpověď z číselníku KONTVST je v tomto případě dvacetiznakový řetězec ve tvaru:
'2␣VSTUP␣S␣PREPISEM␣␣␣'

Jak to bude vypadat v programu?

Uvedu část programu pracujícího s číselníkem KONTVST. Program je psán v jazyce PL/1.

```
DCL  KODUZ     CHAR(3),
     CISOBL    CHAR (1),      deklarace jednotlivých
     ZPUSPR    CHAR(1),       parametrů
     ODPOVED   CHAR(20);
```

```
DCL  A CHAR (1) DEF ODPOVED;
DCL  ZPRAVA CHAR (19) DEF ODPOVED POS (2);   +
                    .
                    .
                    .
CALL KONTVST (ODPOVED, KODUZ, CISOBL, ZPUSPR))    vyvolání
IF A='1' THEN GOTO W1;
IF A='2' THEN GOTO W2;
IF A='3' THEN GOTO W3;                          ++
IF A='4' THEN GOTO W4;
CALL DISPL (ZPRAVA);
            .
            .
            .
```

Pro využití v programu je podstatná deklarace a vyvolání, části označené +, ++ jsou jenom nezávazným nástinem řešení.
V jazyce COBOL popíšeme pouze vyvolání.

```
CALL 'KONTVST' USING ODPOVED KODUZ CISOBL ZPUSPR.
```

Z uvedeného příkladu je vidět, že vlastní použití číselníkového modulu v programu je velmi jednoduché a neodlišuje se od běžného vyvolání programového modulu.

Jak vypadá zpracování číselníku překladačem a co je třeba učinit pro připojení číselníku do programu?

Zdrojový text číselníkového modulu není samozřejmě použitelný pro vlastní práci počítače a prochází tedy určitými transformacemi. U starších typů překladačů se ze zdrojového textu vygeneroval program v jazyce ASSEMBLER a pak proběhla kompilace, jejímž výsledkem je OBJECT-modul. Po zpracování OBJECT-modulu programem LINKAGE-EDITOR vzniká LOAD-modul.
U nového překladače odpadá fáze generování programu v ASSEMBLERU, ale pomocí vlastního interpretátoru vzniká OBJECT-modul a po linkování LOAD-modul.
LOAD-modul lze již přímo zpracovávat počítačem.

Připojení číselníkového modulu k programu lze provést třemi způsoby:

- přiřetězením knihovny OBJECT modulů na vstup spojovacího programu
  (tento způsob má výhodu v tom, že se do programu připojí vždy poslední verze)
- přiřetězení knihovny LOAD-modulů a "přelinkování" programu
  (v tomto případě se připojí číselníkový modul pevně a existuje určité nebezpečí, že jestliže se zapomene na linkování po provedení změny číselníku, je v programu připojena stará verze)
- dynamické volání číselníkového modulu v čase GO
  (v tomto případě musí být stále nasazena knihovna, na níž jsou uloženy číselníkové moduly, v programu je vždy poslední verze číselníkového modulu).

Kde a jak byly číselníkové moduly aplikovány?

Rozšíření číselníkových modulů nemělo pouze kvantitativní povahu, ale se zvyšujícími se zkušenostmi došlo ke zvýšení kvality jednotlivých aplikací.

Popíšeme zde několik nejširších aplikačních oblastí včetně jednoduchých příkladů.

- Kontrola dat: velmi častý je případ, že je zapotřebí kontrolovat a to zejména u větších objemů dat určité položky na existenci v nějakém seznamu.
  Příklad: Určitá hospodářská jednotka má závody, hospodářská střediska, nákladová střediska a určité druhy činností, které jsou v nich provozovány. Při vstupu dat je zapotřebí tyto vesměs číselné údaje kontrolovat tak, aby bylo zajištěno, že všechny nesprávné kombinace budou z dávky vyřazeny. Pokud použijeme pro řešení tohoto problému číselníkového modulu, pak např. jakékoliv změny v rámci reorganizace výrobní jednotky znamenají pouze zásah do číselníkového modulu a ne do programu.
- Převodníky: zde je řešena problematika vzájemně jednoznačného přiřazení určitých číselných kódů.

Příklad: Při integrování různých subsystémů vzniká často potřeba přiřazovat určitým interním klíčům jednoho systému interní klíče systému druhého. Velmi efektivně lze tuto problematiku řešit zavedením jednoduchého číselníkového modulu. Ve volacím parametru má číselný klíč jednoho systému a v odpovědi jemu přislušející kód systému druhého.

- Přiřazování textů číselným kódům: číselný kód o mnoha položkách nemá pro uživatele, pokud nedisponuje fenomenální pamětí, potřebnou informační hodnotu. Zde v oblasti volání použijeme kód a v odpovědi název.
  Příklad: Číselník materiálu X1 = typové číslo materiálu, Y = název.
- Přiřazování algoritmů: určité konfiguraci vstupních podmínek je třeba přiřadit algoritmus zpracování.
  Příklad: Volací úroveň je číslo ukazatele, v odpovědi je uveden název členu na knihovně load modulů, v němž je odpovídající algoritmus výpočtu, který se pak dynamicky vyvolá.
- Komentování určitých situací: určité konfiguraci vstupních parametrů odpovídá nějaký verbální komentář, jestliže vhodně zpracujeme vstupní úroveň, pak do odpovědi vkládáme tento komentář.
  Příklad: Máme 4 výrobní ukazatele, každý může nabývat dvou stavů: 1 pozitivní tendence, 2 negativní tendence. Dále víme, že každou kombinaci můžeme ohodnotit určitým verbálním komentářem. Pak založíme číselník, kde ve volacích parametrech budou jednotlivé stavy a v odpovědi komentáře.

Jak se zakládají číselníkové moduly?

Zakládání číselníkových modulů můžeme rozdělit na dvě základní oblasti:

1) Číselníkový modul je zakládán přes zdrojový text manuálně navrhovatelem.
2) Číselníkový modul je zakládán pomocí programu
   a) bez pomoci datového souboru

b) s pomocí datového souboru.

Ad 1) Běžný způsob, který je používán u převážné většiny číselníkových modulů. Číselník je specifikován, většinou analytikem, do programovacího formuláře, pak vyděrován a zpracován překladačem.

Ad 2a) Někdy je vhodné vygenerovat číselníkový modul nebo jeho část pomocí programu. Např. chceme mít ve volacích úrovních všechny možné kombinace stavů jednotlivých vstupních parametrů. Pak tuto část zdrojového textu můžeme generovat programem a doplnit pouze odpovědi.

Ad 2b) Pro založení číselníku použijeme programu a datového souboru, jehož transformací dostaneme přímo zdrojový text.

Příklad: Máme soubor vozidel, kde je mimo jiné SPZ a počty najetých kilometrů. Pro okamžitou informaci např. dispečera založíme z tohoto souboru číselníkový modul tak, že vygenerujeme popisový štítek a pak vytvoříme jemu odpovídající strukturu na základě vstupujících dat ve formě zdrojové formy číselníku. Tento číselník tedy obsahuje, zakládáme-li jej z aktuálního souboru, vždy poslední stav bez zásahu člověka do zdrojového textu.

Jednotlivé způsoby lze vhodně kombinovat.

Jak ovlivnily aplikace vývoj překladače a naopak?

Bouřlivý vývoj v aplikační oblasti zasáhl zpětně vývoj překladače a zvýšená kvalita překladače ovlivňuje další vývoj aplikací.

Vyjmenujeme pouze několik základních rozdílů, na nichž je zřetelně vidět veliký kvalitativní rozdíl v možnostech starého a nového překladače.

Z tohoto neúplného výčtu rozdílů bude zřejmé, že se novým překladačem otevírá další cesta ke zkvalitnění aplikací.

Starý překladač:	Nový překladač:
Počet volacích úrovní: 4	Počet volacích úrovní: 254
Odpověď konstantní	Odpověď proměnlivé délky
Forma parametrů X1,X2,X3,X4 a odpovědi pouze znaková	Forma parametrů znaková, pack, binární a hexadecimální
Možnost podmíněného překladu (preprocessor) ne	ano
Možnost pouze jednoho popisového řádku	Možnost více popisových řádků

Dochází také v mnoha případech ke změně vztahu mezi programem a číselníkovým modulem. Číselníkový modul se může díky své obecnosti a schopnosti akceptovat velmi rychle změny stát jakýmsi ústředním řídícím členem celého programu a odsoudit vyvolávající program pouze do pozice "manipulátoru".

Jak udržovat číselníkové moduly?

Údržba číselníkových modulů je relativně samostatná oblast. Z důvodů komplexního pohledu na problematiku číselníkových modulů je však zapotřebí podat alespoň stručný úvod do této problematiky. Číselníkové moduly jsou v naší organizaci centralizovány a pro jejich údržbu je vyvinut podpůrný softwarový aparát. Tento aparát se rozpadá do těchto základních oblastí:

- údržba knihovny zdrojových textů a knihovny LOAD-modulů
- komunikace knihoven počítačů IBM370/145 a EC 1040
- opravy a výpisy zdrojových textů a jejich centralizace
- zakládání nových modulů na základě požadavku uživatelů
- statistická část údržby a evidence
- logové funkce překladače
- možnost opravovat číselníkové moduly přes displej.

Kromě úkolů, které vyplývají z výše uvedeného softwaru,

je zapotřebí ještě řešit otázky kompetence oprav do tzv. centrálních číselníků, které jsou využívány více systémy a zásah jednoho může omezit zpracování systému jiného.

S ohledem na vysokou frekvenci používání číselníkových modulů jsou obě knihovny umístěny na systémových discích a tudíž stále nasazeny.

Závěr

Cílem tohoto příspěvku nebylo reklamní vystoupení za účelem propagace určitého překladače, ale snaha poskytnout informace o obecném řešení určité problémové oblasti se snahou odpovědět na všechny otázky obecného charakteru.