Ing. Jaroslav KLEČKA
Závod výpočetní techniky OKD, Ostrava

OPERAČNÍ SYSTÉM Z HLEDISKA PROGRAMÁTORA

1. Úvod

Moderní rychlý elektronický počítač byl vytvořen proto, aby počítal. Aby tato jeho funkce byla splněna co nejefektivněji, musí se v co největší míře z jeho obsluhy vyloučit lidský činitel a člověka musí nahradit nějaká automatická činnost. Z tohoto důvodu byly také vytvořeny o p e r a č n í s y s t é m y .

Vzhledem k tomu, že bez výpočetní techniky se dnes neobejdeme prakticky v žádném odvětví našeho národního hospodářství a vzhledem k tomu, že výpočetní technika je dnes nejdynamičtějším oborem, chci vám sdělit naše praktické zkušenosti s operačními systémy, které byly a jsou využívány v našem výpočetním středisku.

V současné době jsou u nás instalovány dva střední samočinné počítače od předních světových výrobců této techniky. Jsou to počítače ICT 1905 a IBM 370/145.

První z nich je počítačem ještě druhé generace a jeho in-

stalace je z hlediska vnějších pamětí orientována výhradně na magnetické pásky. Firma ICL dodává pro řadu počítačů ICL 1900 několik operačních systémů různé úrovně, pod společným obchodním názvem GEORGE (Generalized Organization Equipment). Všechny tyto operační systémy mají vysokou úroveň. V našem závodě jsme zkoušeli pouze nejjednodušší z nich, pod jménem GEORGE 1. Tento operační systém vyžaduje pro svoji činnost 1 stojan MP, 1 snímač DŠ nebo DP a jednu rychlotiskárnu, při čemž pouze rozšiřuje řídící funkce exekutivního programu (supervizoru), který řídí vlastní multizpracování uvnitř celého výpočetního systému. Instalovaný počítač umožňuje simultánní zpracování 4 úloh a celá konfigurace vypadá následovně:

1 centrální jednotka s vnitřní pamětí 32 k slov (a 24 b.)
8 MP stojanů 7stopých s přenosovou rychlostí 60 kc
1 pomalá řádková tiskárna - 300 ř/min.
2 rychlé řádkové tiskárny - 1350 ř/min.
2 snímače DŠ - 900 DŠ/min.
2 snímače DP FS 1500 - 1500 zn./sec.
2 děrovače DP - 100 zn./sec.

Vzhledem ke struktuře zpracovávaných úloh v Závodě výpočetní techniky OKR (velké procento dlouhých tisků) a vzhledem k již zmíněnému nároku operačního systému na periferní zařízení, při současné možnosti řídit chod pouze jedné úlohy v daném okamžiku, se tento operační systém pro počítač ICT 1905 neprosadil.

Druhý počítač IBM 370/145 byl v ZVT OKD instalován v roce 1973 a má k dispozici jeden z nejvýkonnějších operačních systémů, které v současné době v ČSSR existují. Je jím IBM OS/VS1, operační systém pro řízení virtuální paměti, který je v ČSSR k dispozici od konce roku 1973. Před tímto datem jsme používali operační systém IBM OS/MFT, dodávaný ještě pro počítače IBM 360.

Ve svém dalším výkladu bych se chtěl podržet právě posledně jmenovaného operačního systému. Důvodem je to, že v ČSSR se začínají nasazovat počítače Jednotného systému elektronických počítačů, jejichž výrobci jsou země RVHP. Nejvyšší členy této jednotné řady - EC 1040 a EC 1050 - pak mají mít podle našich informací k dispozici operační systémy, které budou téměř shodné s již zmíněným IBM OS/MFT. V dohledné době se jistě najde velká řada zájemců o tyto dva typy počítačů a tedy i zájemců o operační systémy.

Pro informaci uvádím ještě složení konfigurace počítače IBM 370/145, která pracovala pod řízením operačního systému IBM OS/MFT.

1 centrální jednotka - model 3145 s vnitřní pamětí 256 kB
4 stojany 9stopých MP - model 3420 - s rychlostí přenosu 120 kc
2 stojany 7stopých MP - model 3420 - s rychlostí přenosu 60 kc
(tyto dva stojany jsou určeny pro datovou kompatibilitu s počítačem ICL 1905)
6 jednotek velkokapacitních výměnných disků - model 2319 s kapacitou záznamu 30 MB.
1 snímač DŠ - model 3505 - čte 1200 DŠ/min.
1 snímač/děrovač DŠ - model 3525 - čte 300 DŠ/min. a děruje 100 DŠ/min.
2 rychlé řetězové řádkové tiskárny - model 1403 - 1100 ř/min.
2 snímače DP FS1500 - 1500 zn/sec.

Pro doplnění uvádím i současný stav konfigurace, která byla počátkem tohoto roku rozšířena o dalších 256 kB vnitřní paměti a o 7 znakových displejů modelu 3720 a dvě znakové mozaikové tiskárny modelu 3284 pro pořízení kopie obsahu obrazovky displeje. Všech 7 displejů je připojeno v lokálním módu přímo na centrální jednotku, tedy bez modemu.

2. Operační systém IBM 360 OS/MFT.

Zkratka MFT vznikla z počátečních písmen anglického náz-

vu operačního systému Multi Fixed Task a říká, že operační systém je určen pro současné zpracování více úloh, při čemž je každé z nich pro toto zpracování přidělena pevná, předem definovaná část paměti. Jednotlivé pevné části paměti lze sice během zpracování co do rozsahu měnit, ovšem pouze v případě, že v okamžiku redefinice paměti neprobíhá právě žádné zpracování.

2.1. Základní filozofie operačního systému.

Pro danou konfiguraci počítače je připraveno určité množství práce. Všechnu tuto práci není samozřejmě možno zpracovat vzhledem k omezeným zdrojům daného počítače najednou.

Operační systém se na celé vybavení počítače dívá jako na samostatné zdroje a to jak na vybavení hardware tak i na celý software. Všechny svoje zdroje pak operační systém přiděluje podle požadavku jednotlivých prací, tak jak přicházejí ke zpracování. Jestliže si tedy nějaká práce vyžádá vyvolání určitého programového modulu, musí si operační systém zjistit, zda je tento vyvolávaný modul k dispozici, jako součást zdroje programů (programové knihovny). Dále pak musí zjistit nároky vyvolaného programu na paměť i na délku pobytu v počítači a na všechny ostatní zdroje tímto programem dále nárokované (tzn. nároky na periferní zařízení) . Jakmile jsou všechny zdroje programovému modulu přiděleny, stává se daná část práce pro operační systém úlohou - taskem - a tato úloha může být pod řízením operačního systému zpracována.

Celý operační systém se skládá ze dvou základních částí:

a./ elementů řídícího programu;
b./ elementů zpracovávaných programů.

Výše zmíněné funkce jsou spojeny v konkrétním operačním systému a jsou vytvářeny podle požadavků uživatele a možností

dané konfigurace a jsou začleněny do systému při jeho generování. Generování je zahajovací operací na každé nové verzi operačního systému, která je výrobcem počítače vytvořena. Samotné generování probíhá tak, že podle parametrů, zadaných systémovými programátory jsou z distribučních knihoven vybrány funkce, specifikované pro konkrétní výpočetní systém. Celý operační systém se pak skládá ze systémových knihoven, uložených na diskovém nosiči. Z těchto knihoven jsou pak jednotlivé elementy operačního systému natahovány do vnitřní paměti a řídí celou činnost počítačového systému.

2.1.1. Elementy řídícího programu.

Řídící program má čtyři základní funkce:

a./ přijímá a plánuje práce ze vstupního proudu prací - j o b m a n a g e m e n t ;

b./ dohlíží buďto na sekvenční nebo prioritní bázi nad každou jednotkou kterékoliv práce, která je v počítači prováděna - t a s k m a n a g e m e n t ;

c./ provádí ukládání dat, jejich údržbu a znovuzískání s ohledem na zvolenou přístupovou metodu - d a t a m a n a g e m e n t ;

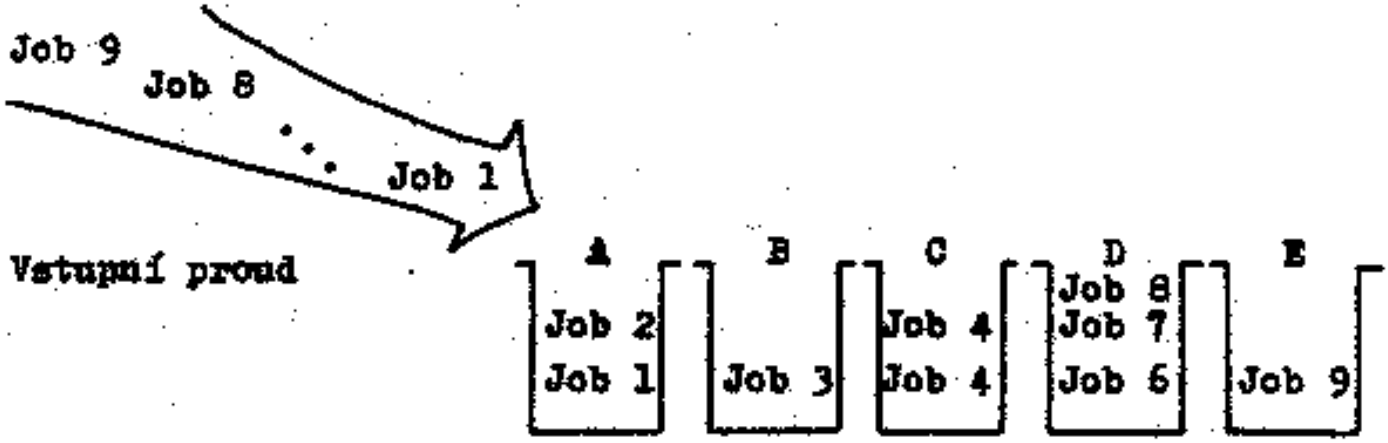
d./ provádí některé zásahy do zpracování v případě havárie celého výpočetního systému - r e c o v e r y m a n a g e m e n t ;

2.1.1.1. Job management.

Styk mezi uživatelem počítače (zadavatelem práce) a vlastním počítačem je zajištěn pomocí výroků speciálního řídícího jazyka - Job Control Language. Hlavní funkcí popisované složky operačního systému je přeložit výroky JCL, oddělit je od dat vstupujících ze vstupního proudu a uložit vytvořený řídící program a data do vstupních systémových front. Z těchto front jsou pak jednotlivé práce vybírány buďto na

sekvenčním nebo prioritním základě. V další fázi jsou pak jednotlivým krokům práce přidělena vstupní a výstupní periferní zařízení a na vyvolaném kroku je zahájena činnost operační jednotky počítače. Během zpracování zajišťuje job management předávání zpráv mezi programem a operátorem. Job management dále zpracovává ukončení kroku a řídí zapisování výstupních sestav do výstupních systémových front na discích. Z těchto výstupních front jsou pak po zkončení celé práce všechny výstupní sestavy pomocí zvláštního vypisovacího systémového programu vypsány na rychlotiskárně.

Příklad řízení vstupních front:



Vnitřní paměť:

		ACD	BDE
32 k	12 k	56 k	106 k
Supervisor	WTR	JOB1,JOB2 JOB4,JOB5	JOB3,JOB6,JOB7 JOB8,JOB9

2.1.1.2. Task management.

Jak jsem již říkal v úvodu této kapitoly, stává se část práce po přidělení jí náležejících zdrojů ú l o h o u - - t a s k e m/. Veškerou činnost úlohy (v běžném slova

smyslu programu), kromě práce s daty, pak řídí t a s k m a n a g e m e n t . Task management plní v podstatě funkci supervisoru. Mezi jeho řídící činnosti patří například dynamické přidělování či propouštění vnitřní paměti v rámci dané pevně přidělené paměti pro celou práci, ochrana paměti proti přepsání nebo zajištění správného multiprogramování všech současně zpracovávaných úloh.

2.1.1.3. Data management.

Veškerou práci s daty obstarává operační systém pomocí přístupových metod. OS/MFT má k dispozici velké množství přístupových metod a mezi běžně používané patří např. sekvenční (frontová - QSAM, základní - BSAM), indexně-sekvenční (frontová - QISAM, základní - BISAM), členěná (BPAM) a přímá (BDAM). Tyto přístupové metody jsou určeny pro práci se znakovými periferiemi, s magnetickými páskami a disky. Pro práci s dálkovými přenosy dat jsou pak určeny telekomunikační přístupové metody, jakými je např. v současné době používaná základní metoda BTAM, která je i součástí operačního systému OS/VS1.

Jednotlivé přístupové metody řídí veškeré přenosy dat mezi vnějšími periferiemi a úlohou za pomocí různých technik vyrovnávacích pamětí.

2.1.1.4. Recovery management.

Operační systém má zabudovány rutiny, které se snaží automaticky zabránit některým chybám hardware a pokud jim již nemohou zabránit, zapisují o nich alespoň zprávy na diagnostický soubor, podle něhož má pak možnost technický personál poruchu odstranit.

Současně lze na základě činnosti těchto rutin provádět restart celého operačního systému.

2.1.2. Elementy zpracovávaných programů.

Zpracovávané programy se dělí na tři části a to na:

2.1.2.1. Překladače programovacích jazyků:

Operační systém OS/MFT má k dispozici následující překladače:

Algol	FORTRAN
Assembler	PL/I
COBOL	RPG

Všechny tyto překladače přeloží zdrojový modul do tzv. object modulu a uloží jej na pracovní medium (DŠ, MP nebo disk). Vlastní object modul není ještě schopen samostatného zpracování. Tuto možnost zajistí až L i n k a g e e d i t o r nebo L o a d e r , což jsou programy, které vytvoří tzv. load modul. Load modul již může být natažen do paměti a může být zpracován jako program. Linkage editor a Loader jsou součástí následující skupiny programů.

2.1.2.2. Obslužné programy.

Mezi obslužné programy patří dále,kromě již zmíněných Linkage editoru a Loaderu, pomocné programy pro práci s daty uživatele (kopírování souborů, jejich výpisy a j.), pomocné programy pro práci se systémovými daty (katalog, jmenovky souborů či nosičů), Sort - Merge program (třídící program) nebo nezávislé pomocné programy, pracující bez podpory operačního systému.

2.1.2.3. Uživatelem napsané programy.

Uživatelovy programy se stávají součástí OS, neboť jsou

zapsány na knihovny objekt modulů nebo load modulů a tyto knihovny pak operační systém obhospodařuje pomocí job management a speciálně pak pomocí JCL.

3. Programátor a operační systém.

Jak jsem již v úvodu celé své přednášky uvedl, byl v našem výpočetním středisku před nasazením počítače IBM 370/145 používán počítač ICT 1905. Tento počítač pracoval a stále ještě pracuje bez operačního systému. Pro tvorbu programů byly používány problémově orientované jazyky COBOL a FORTRAN a jazyk PLAN, což je jazyk na úrovni Assembleru, při čemž z historických důvodů bylo nejvíce programů napsáno právě pomocí jazyka PLAN. V roce 1967, kdy se začínalo s programováním úloh pro zpracování hromadných dat, které jsou hlavní náplní práce ZVT OKD, nebyl k dispozici kvalitní kompilátor pro COBOL. Z tohoto důvodu byl PLAN zvolen jako jediný jazyk pro zpracování hromadných dat. Vzhledem k tomu, že je tento jazyk poměrně bohatě vybaven rutinami pro obsluhu vstupních i výstupních zařízení, programuje se v něm poměrně dobře, i když jako v každém nižším programovacím jazyku, dosti zdlouhavě.

Teprve postupně, jak se kompilátory pro COBOL vylepšovaly a jak se vyjasňovala situace okolo nákupu další výpočetní techniky, začalo se COBOLu používat ve větší míře.

Při psaní programů pro počítač bez operačního systému musí programátor informovat operátora počítače o činnosti svého programu, zejména pak o tom, jak se má zachovat při správném či abnormálním konci programu. Operátor pak podle těchto informací a na základě zpráv z programu řídí další zpracování. Znamená to tedy, že vytvořením programu a popsáním všech výstupních zpráv a případně i popsáním pokynů pro nastavení některých potřebných ovládacích instrukcí při zahájení zpracování, končí veškerá činnost programátora ve vztahu ke zpracování úlohy na takovémto počítači.

Zcela jinak tomu je při řízení práce za použití operačního systému. Jak jsem uvedl ve druhé kapitole své přednášky, je styk zadavatele práce s operačním systémem prováděn pomocí výroků řídícího jazyka (JCL). Řídící výroky se dělí do několika základních typů, při čemž každý z těchto typů zadává parametry pro ovládání činnosti jednotlivých, již dříve popsaných částí operačního systému (Job, Task a Data management).

Programátor se tedy musí naučit nejen programovat v daném programovacím jazyce, ale musí zvládnout alespoň základy řídícího jazyka, aby vůbec mohl svůj program odladit a později i spracovávat. Je pochopitelné, že vzhledem k rozsáhlosti operačního systému je velmi obsáhlý i řídící jazyk - vždyť musí postihnout všechny znamce operačního systému.

Běžný aplikační programátor se s celým řídícím jazykem seznamovat nemusí, ale minimálně 50% jeho výroků musí znát proto, aby mohl ladit svoje jednodušší programy. Jestliže však začne sestavovat jednotlivé programy do prací tak, aby na sebe navazovaly, musí znát dalších 20 - 30% zbylých výroků řídícího jazyka. Zbytek výroků je pak určen pro systémové programátory. Pro srovnání je možno uvést, že k tomu, aby mohl programátor programovat, musí zvládnout literaturu v rozsahu nejméně 700 stránek formátu A4 a řídící jazyk je obsažen asi na 300 stránkách.

V současné době není ještě známo, ze kterého jazyka budou odvozeny výroky řídícího jazyka pro počítače EC 1040 a EC 1050, ale pro počítač IBM a jeho operační systémy se znalost angličtiny vyplatí, neboť všechny výroky jsou buďto plným nebo zkráceným vyjádřením činnosti, kterou ovládají.

Operační systém IBM má k dispozici celou škálu pomocných programů pro práci s daty uživatele, případně se systémovými daty. Znalost těchto pomocných programů, mających velmi široké použití (výpisy dat uložených v různých formách na různých

paměťových médiích, kopírování různých souborů na různých nosičích, generování souborů a j.) velmi usnadňuje a v některých případech přímo podmiňuje práci při tvorbě a ladění programu. Pro pomocné programy existuje opět manuál asi o 300 stránkách, z nichž nejméně polovina je běžně používána aplikačními programátory. Je pochopitelné, že ne všichni pracovníci jsou schopni si číst firemní literaturu v originále a pro ně bylo nutné připravit písemné, většinou zkrácené informace o funkci nejdůležitějších pomocných programů a usnadnit tak práci při tvorbě programů.

Největší změnu v myšlení programátorů však znamenal přechod na počítač, který má instalovány diskové paměti. Při použití pouze magnetických pásek existuje pouze sekvenční zpracování dat, kdežto u diskových pamětí a speciálně u IBM OS existují kromě této metody ještě metody indexně sekvenční, přímá a členěná. S těmito organizacemi se pochopitelně programátor seznámí v rámci studia konkrétního programovacího jazyka. Musí však použití konkrétní přístupové metody skloubit i s použitím výroků řídícího jazyka.

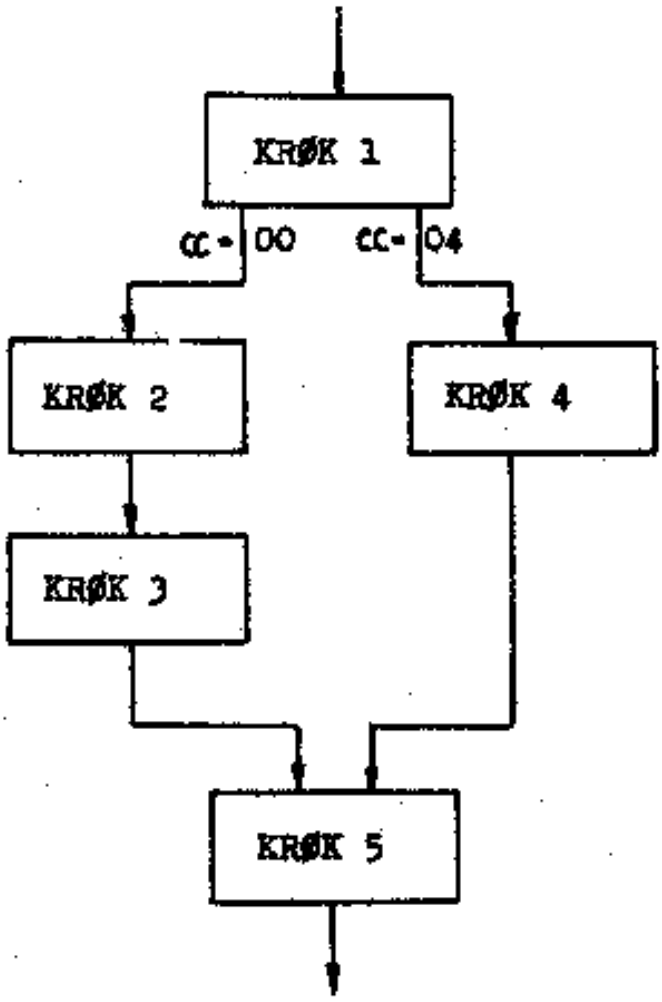
Další věcí, se kterou se v operačním systému dostane programátor do styku, zejména při ladění, je hledání ve výpisu paměti. Tento výpis je produkován operačním systémem ve dvou verzích (zkrácený nebo plný). Výpis paměti právě zpracovávané úlohy se vytváří při všech abnormálních koncích úlohy nebo na vyžádání programátora, například při dosažení určitého místa v programu nebo při překročení zadaného času.

Funkce celého operačního systému je řízena pomocí různých řídících bloků, které jsou obsluhovány formou zpracování seznamů. I když většinu řídících bloků aplikační programátor nemusí znát, existuje nejméně jeden blok, s jehož obsahem musí být seznámení a musí vědět, jak se tento blok naplňuje řídícími daty. Jeho název je Data Control Block a existuje pro každý soubor a podle jeho obsahu může programátor z výpisu programu usoudit

na stav souboru v okamžiku přerušení práce. Obsah DCB je ovlivňován jednak samotným programem, dále pak řídícími výroky řídícího jazyka a nakonec obsahem jmenovky souboru. Podle jeho obsahu se dá poznat, zda byl soubor již otevřen nebo uzavřen, jaká je délka logického a fyzického záznamu nebo jaká je metoda organizace dat na tomto souboru. Spolu s obsahem všech pracovních registrů a případně i obsahem vyrovnávacích pamětí je programátorovi dána možnost se orientovat i v programu napsaném v problémově orientovaném jazyce.

Při sestavování jednotlivých programových kroků do prací vyvstávají další možnosti, se kterými musí programátor při tvorbě svého programu počítat.

První možností je větvení činností uvnitř práce - jobu. Každý program může mít několik různých stavů, při nichž je ukončen. V závislosti na těchto stavech se pak mají zpracovávat různé větve navazujících programů.



Ve všech programovacích jazycích, pracujících pod operačním systémem IBM lze nastavovat tzv. podmínkový kód a po jeho nastavení ukončit program, což znamená předat řízení rutinám operačního systému - v našem případě rutinám job managementu. Řídící jazyk má pak možnost testovat podmínkový kód a větvit zpracování.

Podmínkový kód je tedy součástí operačního systému, ale i každého programovacího jazyka a programátor jej musí umět používat. Kromě řešení podmínkových kódů je řídící jazyk schopen vyřešit i stavy abnormálních konců úloh a pokračovat v práci nebo zabájit požadovanou činnost, která se provádí pouze v případě abnormálního konce kteréhokoliv programu.

Druhou možností je využívání systémového souboru, nazývaného k a t a l o g . Do tohoto souboru lze pomocí rutin job managementu a zejména pak pomocí řídícího jazyka zapisovat některé údaje o používaných souborech dat. Mezi údaje, které se do katalogu zapisují, patří typ použitého nosiče dat, seriové číslo použitého nosiče, název souboru a jeho generační číslo. Údajů z katalogu se dále využívá pro předávání jednotlivých souborů mezi jednotlivými programovými kroky práce a zjednodušuje se také zápis výroků řídícího jazyka.

Při používání generačních čísel souborů však vzniká jisté nebezpečí. Toto nebezpečí spočívá v tom, že jakmile je vytvořena další vyšší generace (zápis do katalogu se provádí při přidělování zdrojů, tedy ještě před započetím práce vlastního programu), a vzniknou-li jakékoliv obtíže při zpracování programu, musí se většinou program opakovat a je pak nezbytné se odvolávat na zmíněný soubor konkrétním číslem generace nebo opravit záznam o tomto čísle generace přímo v katalogu. Pokud by tento zásah do zpracování neproběhl, může dojít k automatickému zrušení záznamu v katalogu a tím i ke ztrátě přehledu o jednotlivých souborech, používajících generační čísla. Zmíněné nebezpečí je největší u souborů uložených na diskových

nosičích dat. U prací, které tedy používají generační čísla, je proto nezbytně nutné poznamenat v požadavku na její zpracování, aby celá práce byla při přerušení pozastavena až do dalšího kvalifikovaného zásahu programátora či zadavatele zpracování.

Třetí možností, kterou operační systém poskytuje, je tzv. nezávislost na použití periferního zařízení. Tato možnost říká, že data mohou být při sekvenčním ukládání zapsána na jakémkoliv typu zařízení, tento způsob uložení umožňujícím. Jedná se tedy o ukládání dat na MP, disky, DŠ nebo na rychlotiskárnu. O tom, kam budou data skutečně zapsána, lze rozhodnout až těsně před zpracováním podle toho, jaký je momentální stav obsazení periferních jednotek jinými pracemi. S touto možností operačního systému by měl každý programátor při psaní svého programu počítat a zadat popis souboru tak, aby si například neodporovaly možnosti zápisu fyzických záznamů na různých nosičích. Pro soubory ukládané alternativně na MP nebo na discích platí, že délka fyzického záznamu - bloku nesmí přesáhnout délku stopy použité diskové paměti. Při psaní programu se tedy doporučuje neudávat pevně typ výstupního ani vstupního zařízení, na němž se soubor ukládá a dále pak se doporučuje neudávat délku fyzického záznamu. Tyto údaje se pak mohou doplnit pomocí výroků řídícího jazyka.

Poslední věc, o které bych se chtěl zmínit, je omezení velikosti paměti programu. Toto omezení je dáno omezenými velikostmi jednotlivých, předem definovaných částí vnitřní paměti. Velikosti částí paměti, určených pro zpracování příslušných tříd prací (viz 2.1.1.1) jsou zadány při generaci operačního systému a lze je co do rozsahu měnit pouze při zahájení práce celého systému nebo při pozastavení všech vstupních front. Velikost programu je dána a do velké míry ovlivněna použitým programovacím jazykem. Přibližně platí následující závislost:

Assembler	COBOL	PL/I
1	1,6 - 2	3 - 5

Současně platí zhruba stejné závislosti na použitém programovacím jazyce ve vztahu ke spotřebě času centrální jednotky počítače na daný program.

Pokud má být práce určité třídy úspěšně provedena, musí se největší program vejít do příslušné části paměti, která je této třídě přidělena. Pokud se však stane, že některý program má větší nároky na paměť, jsou tyto tři možnosti:

1./ práce se nedá zpracovat - operační systém ji ukončí s abnormálním koncem na daném programovém kroku;

2./ před provedením práce se redefinují všechny části paměti tak, aby část pro požadovanou třídu měla odpovídající velikost;

3./ je-li programový krok tak náročný na paměť, že se nevejde ani do části paměti, vzniklé spojením všech ostatních částí, je třeba jej přepracovat. Program se musí rozdělit do několika samostatných částí, které se potom pomocí linkage editoru spojí pomocí overlay techniky do jednoho celku. Při použití overlay techniky se nepotřebné části programu překrývají těmi částmi, které se v daném okamžiku požadují.

4. Závěr

Z celého mého stručného nástinu činnosti operačního systému i z požadavků, které jsou kladeny na programátora při programování na počítači pod řízením operačního systému, je vidět složitost celého systému práce.

Nejnutnější literatura o operačním systému jako celku je obsažena asi na 5 000 stránkách.

Při přechodu na počítač, pracující s podobným operačním

systémem, jako je IBM OS/MFT se doporučuje seznámit se v dostatečném předstihu s jeho filosofií a upozornit všechny tvůrce programových systémů uživatele na možnosti celého výpočetního systému.

5. Použitá literatura.

1./ IBM System/360 Operating System - Concepts and Facilities GC 28-6535-8

2./ IBM System/360 Operating System - Job Control Language Reference GC 28-6704-2

3./ OS Utilities GC 28-6586-14