Základy AS/400 (IBM i)

Vladimír Župka

Obsah

Úvod

Charakteristika počítače AS/400 (systému IBM i)

Aplikační systém 400 (AS/400) byl navržen jako počítač obecně použitelný v obchodním prostředí a je pro toto prostředí optimalizován. Z toho vyplývají i požadavky sledované při návrhu tohoto systému:

snadné použití,
schopnost růstu bez narušení aplikací,
optimální výkon v komerčním prostředí.

Vlastnosti, odlišující AS/400 od ostatních počítačů:

vrstevná architektura,
objektová orientace,
souvislý paměťový prostor
multiprocesor,
operační systém zahrnující pokročilé komponenty (databázi, komunikace atd.)

Poznámka: V roce 2000 byl počítač AS/400 nahrazen počítačem POWER, který sjednocuje vybavení pro několik softwarových platforem:

IBM i – nástupnický systém po AS/400,
AIX – operační systém typu UNIX (verze IBM),
Linux – operační systém typu UNIX (verze IBM).

V rámci tzv. *logical partitions* může být provozováno v jediném počítači současně několik takových platforem.

Krátká historie

Počítač AS/400 měl bezprostředního předchůdce: *System/38*. Ten byl ohlášen v roce 1978 ale byl u nás embargován. System/38 nemá bezprostředního předchůdce. Byl vyvinut na podkladě zcela nových principů, z nichž vyniká především nezávislost na hardwaru a objektově orientovaná architektura.

Další originální vlastnosti jsou:

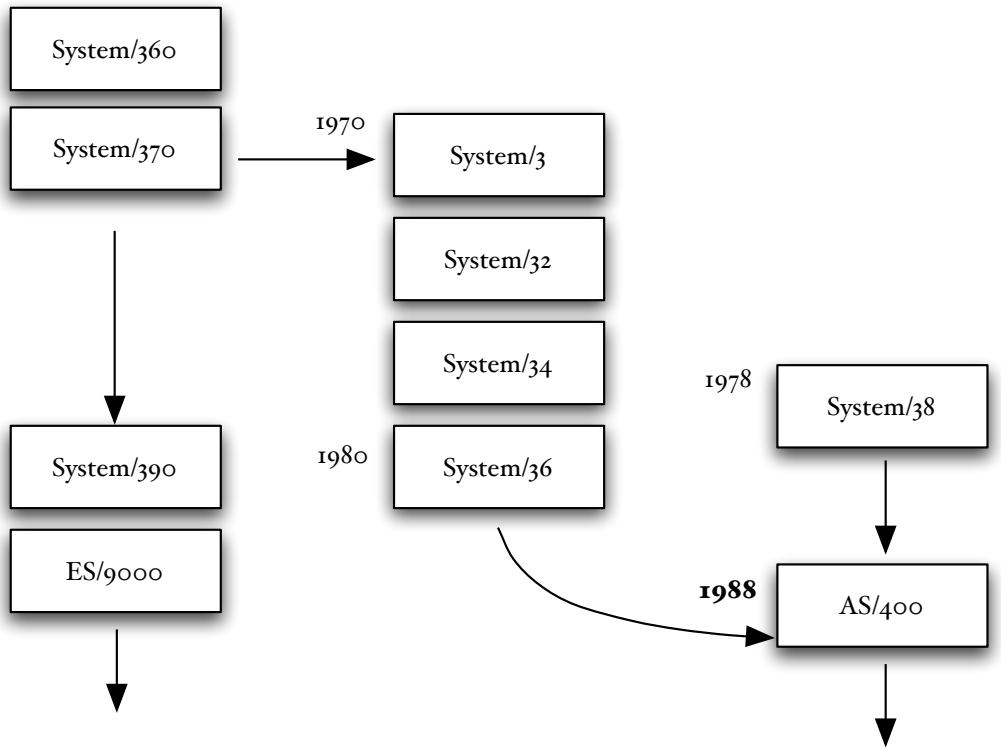
automatická kontrola oprávnění k objektům,
externí popisy dat,
integrovaný databázový systém,
integrovaná podpora komunikací,
kompilovaný řídicí jazyk a další.

Operační systém se jmenoval CPF - Control Program Facility. Tato zkratka se stále ještě vyskytuje u systémových zpráv.

Přestože System/38 byl vyvinut jako zcela původní, převzal některé prvky z předchozích počítačů:

kód EBCDIC z počítačů System/360 a 370,
jazyk RPG z počítače System/3 (podstatně přepracovaný z RPG II na RPG III).

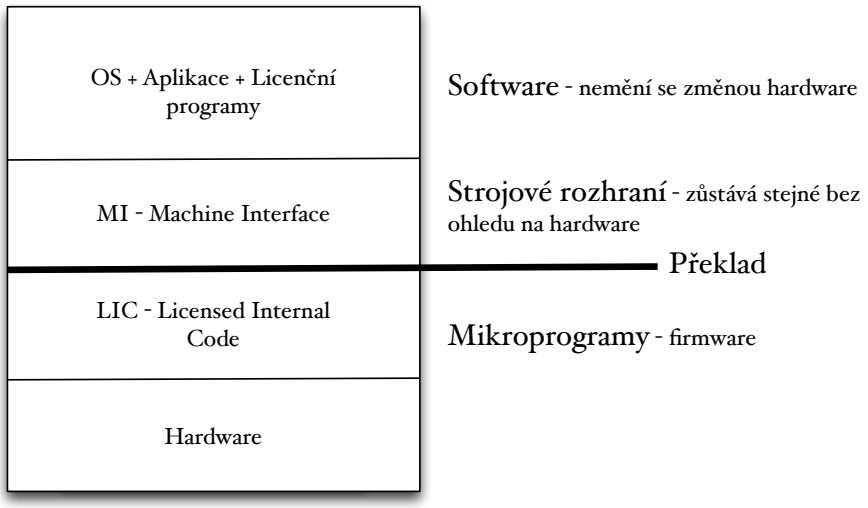
Přibližný časový vývoj je naznačen na následujícím obrázku.



Vrstevná architektura

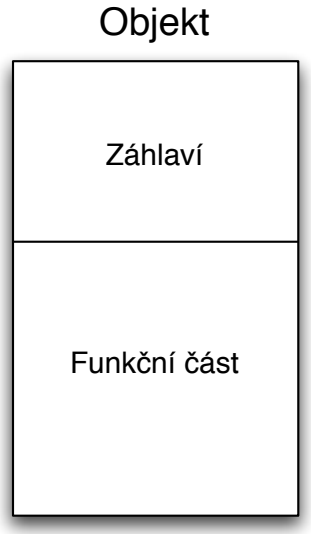
Počítač AS/400 (a dnes platforma IBM i) je navržen tak, aby aplikace na něm vytvořené a provozované byly použitelné i v budoucnosti, kdy technologický pokrok vynutí změnu technických prostředků počítače (hardware), tedy zejména procesorů, velkokapacitních pamětí, komunikačních linek apod.

Proto jsou jeho jednotlivé funkce rozděleny do oddělených vrstev, z nichž ty spodní lze modifikovat či zcela nahradit, aniž by to mělo vliv na horní vrstvu operačního systému a aplikací.



Objektová orientace

Vše, co se ukládá do počítače a vybírá z něj, je obsaženo v objektech. Objekt se skládá ze záhlaví (společného tvaru pro všechny typy objektů) a funkční části (pro každý typ objektu jiné). Objekt tak sdružuje údaje (data) a metody jejich použití do jednoho celku. Objekty zajišťují nezávislost uživatele na změnách v realizaci (implementaci) operačního systému a hardwaru. Prostor v diskové paměti je objektu přidělován automaticky při jeho vytváření a může být dále rozšiřován.



Objekty jsou:

- programy
- databázové soubory
- fronty zpráv
- příkazy řídicího jazyka
- popisy přídavných zařízení
- popisy komunikačních linek

atd.

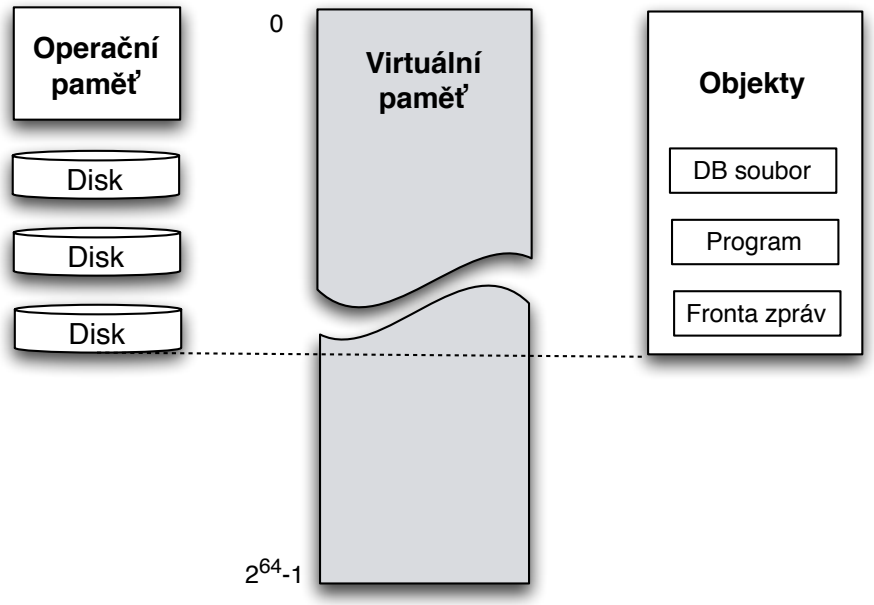
Souvislý paměťový prostor

Veškerá systémová paměť, tj. operační i disková, je adresována jednotně, jako by byla souvislá. Uživatel se nemusí zabývat tím, kde jednotlivé objekty sídlí, stačí, když se na ně odvolá jménem. Adresa obsahuje 64 bitů, takže adresový prostor je 0 až 2^{64} (tj. 18.446.744.073.709.551.616 bajtů, tj. 18,4 miliard GB). Takto koncipovaná paměť se nazývá *virtuální*. Nezávisí na velikosti vnitřní (operační) paměti, ani na typu, kapacitě a počtu diskových jednotek. O skutečné umístění objektů na médiích a jejich vyhledávání se stará operační systém a uživatel je nemůže ovlivnit.

To znamená, že aplikační programy zcela automaticky využívají výhody nových hardwarových technologií, zejména paměťových a procesorových, aniž by se musely jakkoliv modifikovat nebo rekompilovat.

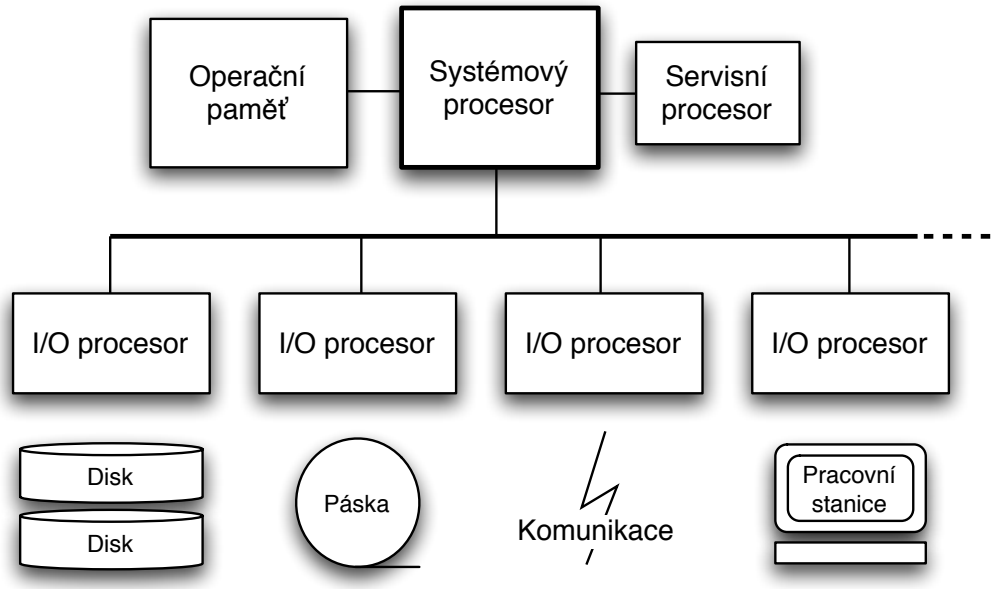
Adresovací schopnost lze kdykoliv rozšířit, a to beze změny "strojového" jazyka MI, což vysvětluje skutečnost, že 64bitové procesory Power PC, později POWER, velmi rychle nahradily dosavadní 48bitové. Jedinou (ale značnou) prací s touto náhradou spojenou bylo přepsání základních programů ve vrstvě SLIC (System Licensed Internal Code). Tu musela provést firma IBM a ta také nesla odpovědnost za hladký přechod od systémů CISC (48bitových) k systémům RISC (64bitovým).

Následující schema znázorňuje vztah fyzické paměti k virtuální paměti a k objektům.



Hierarchie procesorů

Počítač je tvořen hierarchickou soustavou procesorů. Hlavní systémový procesor (základní jednotka) je propojen jednak s operační pamětí, jednak s dalšími, podřízenými procesory. Kromě toho může být rovnocenných hlavních procesorů několik (multiprocesor).



Tato architektura dovoluje souběžné zpracování aplikačních programů: Systémový procesor zjistí požadavek aplikačního programu na čtení dat z disku, deleguje jej příslušnému vstupně/výstupnímu I/O (input/output) procesoru, který ten požadavek plní autonomně. Mezitím systémový procesor zpracovává instrukce jiného aplikačního programu. Podobně jsou zaměstnávány procesory pro komunikační linky, pracovní stanice apod.

Distribuce úkolů mezi specializované procesory zajišťuje vynikající výkonnost v transakčně orientovaném obchodním prostředí. Kromě toho mohou být použity nejnovější mikroprocesorové technologie, aniž by se nějak narušil zbytek systému.

Operační systém

Operační systém se nazýval původně Operating System/400 (OS/400), později byl přejmenován na i5/OS (podle počítače POWER 5) a naposledy na IBM i. Uživatel může využívat tři pracovní prostředí:

přirozené prostředí AS/400 (dnes IBM i),
prostředí počítače System/36,
prostředí počítače System/38.

Nadále se budeme zabývat jen přirozeným prostředím AS/400 (IBM i).

Funkce operačního systému:

Správa objektů	Object Management
Správa dat	Data Management
Řízení práce	Work Management
Komunikace	Communications
Řídicí jazyk	Control Language (CL)

V operačním systému je integrovaná správa objektů, správa dat a přídavných zařízení, řízení práce, i řízení komunikací.

Mnohé z těchto úkolů jsou realizovány ve vrstvě SLIC (System Licensed Internal Code), např. kontrola oprávnění uživatele k použití objektu, hledání v databázovém souboru, nebo třeba obsluha komunikační linky. Jak je patrné, jde většinou o značně složité programy.

Do operačního systému se zahrnují i tzv. licenční programy (kompilátory programovacích jazyků, podpůrné prostředky k vytváření aplikací, programy pro spolupráci s osobními počítači, prostředky k práci s komunikacemi apod.). Zde vyjmenujeme jen několik málo nejdůležitějších.

Programovací jazyky

od verze 2:	RPG/400
	COBOL/400
	C/400
	AS/400 Pascal
	AS/400 PL/I
	AS/400 Basic
od verze 3:	ILE RPG neboli RPG IV
	ILE COBOL
	ILE C, C++
od verze 4.2:	Java

Obslužné programy

ADTS	Application Development Tool Set
PDM	Programming Development Manager - prostředek pro vývoj programů
DFU	Data File Utility - prostředek pro pořizování a změnu databázových souborů
SEU	Source Entry Utility - editor zdrojových textů
SDA	Screen Design Aid - prostředek k návrhu obrazovkových formátů
RLU	Report Layout Utility - prostředek k návrhu tiskových sestav
FCMU	File Compare/Merge - porovnávání a slévání zdrojových textů
EDTF	Edit File - editování proudového nebo databázového souboru

Query/400

Efektivní prostředek k čerpání informací z databázových souborů (dotazovací program) pro neprogramátory.

SQL – Structured Query Language

Standardní prostředek k práci s databázovými soubory (normalizovaný jazyk) pro programátory.

Další prostředky pro práci se systémem

Existuje mnoho dalších prostředků k práci se systémem od třetích dodavatelů.

Následující produkty jsou bezplatné:

IBM i Access Client Solutions	Nástroje k řízení nejběžnějších úkolů pro ovládání systému
Visual Studio Code for IBM i	Programátorský nástroj
IBMiProgTool	Programátorský nástroj
IBMiSqlDisplay	Zobrazování, výběr a řazení záznamů, výběr sloupců
IBMISqlScripts	Vytváření a zpracování SQL skriptů; náhrada za Query/400
IBMiSqlUpdate	Výběr, řazení, přepisování a vkládání záznamů; náhrada za DFU

Placený produkt:

RDi (Rational Developer for i) – Nástroj určený systémovým inženýrům a programátorům. Přehledné výukové dokumenty ve tvaru PDF jsou na stránce https://www.ibm.com/support/pages/node/6120837#labs.

Objekty a jejich ovládání

Objekty a jejich typy

Objekt se skládá ze záhlaví a funkční části. Jisté informace obsažené v *záhlaví* objektu jsou společné všem objektům bez ohledu na typ:

Jméno objektu
Jméno knihovny, v níž je umístěn
Typ objektu
Podtyp (atribut) objektu
Jméno vlastníka objektu
Textový popis objektu
Datum a čas vytvoření
Datum a čas poslední změny
Datum a čas posledního uložení
Datum a čas posledního obnovení

Objekty se liší svým typem. Některé objekty mají kromě typu i podtypy (atributy).

Jméno objektu má maximálně 10 velkých písmen nebo znaků _ (podtržítko). Začíná písmenem. Například PROG_01, LIB_OBJ_2, apod.

Příklady typů a podtypů objektů

Objekt		**Typ**	**Podtyp**
Knihovna	library	*LIB	
Databázový soubor	physical file	*FILE	PF
	logical file	*FILE	LF
	source file	*FILE	SRC
Obrazovkový soubor	display file	*FILE	DSPF
Páskový soubor	tape file	*FILE	TAPF
Ukládací soubor	save file	*FILE	SAVF
Fronta zpráv	message queue	*MSGQ	
Příkaz řídicího jazyka	command	*CMD	
Program	program	*PGM	
Modul	module	*MOD	
Výstupní tisková fronta	output queue	*OUTQ	
Vstupní fronta úloh	job queue	*JOBQ	
Složka (pořadač)	folder	*FLR	
Dokument	document	*DOC	
Popis zařízení	device description	*DEVD	
Popis komunikační linky	line description	*LIND	
Uživatelský profil	user profile	*USRPRF	
Nabídka	menu	*MENU	
Definice dotazu	query definition	*QRYDFN	

Knihovny (libraries)

Knihovna je objekt typu ***LIB**, který slouží k seskupování ostatních objektů. Objekty nemusí být nutně umístěny vedle sebe, ale jsou roztroušeny ve virtuální paměti tak, jak postupně vznikají. Knihovna je vlastně jen seznam jmen a adres objektů, ale říkáme, že knihovna *obsajuje* objekty.

Některé knihovny jsou dodávány již s počítačem (obsahují objekty operačního systému), ostatní si uživatel vytváří z objektů, které si sám vytvořil nebo které získal od jiných dodavatelů. Na rozdíl od jiných operačních systémů nemají knihovny hierarchickou strukturu, ale jsou na sobě nezávislé. Samy jsou (jakožto objekty typu *LIB) sdruženy v systémové knihovně QSYS, která je základní knihovnou.

Každý objekt má jméno, které jej spolu s typem jednoznačně určuje v rámci knihovny. Jiný objekt stejného jména a stejného typu může být obsažen v jiné knihovně. Jeden objekt nemůže být obsažen ve dvou knihovnách. V jedné knihovně nemůže být více objektů stejného jména a typu. Úplná identifikace objektu je tvořena jménem knihovny a jménem objektu s oddělujícím lomítkem, tzv. kvalifikovaným jménem.

Příklady kvalifikovaného jména objektu:

```
KNIHOVNA/OBJEKT
UCETNICTVI/ZAKAZNICI
```

Co lze činit s objektem:

přejmenovat a ponechat v původní knihovně,
zkopírovat, přejmenovat a ponechat v původní knihovně,
zkopírovat do jiné knihovny,
zkopírovat do jiné knihovny a přejmenovat.

Pro práci s knihovnami slouží tyto příkazy:

CRTLIB Create Library - vytvoř knihovnu
CHGLIB Change Library - změň knihovnu
DLTLIB Delete Library - zruš knihovnu
RNMOBJ Rename Object - přejmenuj objekt (zde typu *LIB)

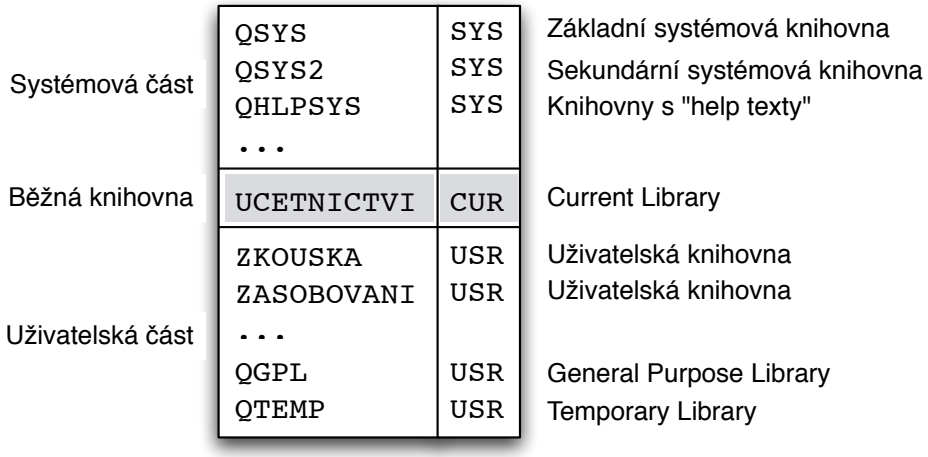
Knihovny jsou dvou podtypů:

produkční *PROD (nejobvyklejší),
testovací *TEST (jen pro zkoušení vyvíjených aplikací).

Seznamy knihoven (library list)

Seznamy knihoven slouží k usnadnění práce při programování a řízení práce na počítači. Seznam knihoven je k dispozici od vzniku úlohy (jobu). S ukončením úlohy zaniká. V příkazech se označuje symbolem *LIBL. Program nebo příkaz se totiž může odvolávat na objekty, aniž by specifikoval jméno knihovny. Je-li knihovna s dotyčným objektem zapsána v seznamu, objekt je z této knihovny vybrán. Při vyhledávání objektu se postupuje od začátku seznamu až k první knihovně, v níž se objekt nalezne. Je-li dále v seznamu zapsána jiná knihovna obsahující stejný objekt, neuplatní se. Nenajde-li se objekt v žádné knihovně seznamu, je hledání neúspěšné a systém o tom vydá zprávu.

Struktura seznamu knihoven:



Nově vytvářený objekt se umístí do běžné knihovny (current library), není-li jméno knihovny výslovně zadáno. Chybí-li běžná knihovna v seznamu, nahrazuje její funkci knihovna **QGPL** (General Purpose Library).

Knihovna **QTEMP** (temporary library) vzniká a zaniká s výpočetní úlohou (jobem). Slouží hlavně k ukládání pracovních objektů (např. souborů), které po ukončení úlohy již nepotřebujeme.

Příkazy pro seznam knihoven:

DSPLIBL	Display Library List - zobraz seznam knihoven
EDTLIBL	Edit Library List - uprav seznam knihoven
CHGCURLIB	Change Current Library - změň běžnou knihovnu
ADDLIBLE	Add Library List Entry - přidej položku do seznamu knihoven
RMVLIBLE	Remove Library List Entry - odstraň položku ze seznamu knihoven
CHGLIBL	Change Library List - změň seznam knihoven

Změnou pořadí knihoven v seznamu lze např. nahrazovat produkční a testovací prostředí, aniž by bylo nutné měnit programy.

Operace s objekty

S objekty operuje jednak systém, jednak uživatel. Systémové funkce probíhají *automaticky*, kdykoliv uživatel použije objekt:

Kontrola oprávnění (authority)
Zjištění poškozených objektů (damaged objects)
Zajištění celistvosti transakcí (commitment control)
Zamykání objektů (locks)
Kontrola stavu objektu
Kontrola typu vzhledem k požadované funkci

Uživatel může s objekty provádět jednak obecné operace, jednak specifické operace. Obecné operace jsou ty, které lze aplikovat na většinu objektových typů, zejména:

Allocate (ALCOBJ) rezervuje objekt k výhradnímu nebo sdílenému použití
Deallocate (DLCOBJ) uvolňuje objekt z výhradního nebo sdíleného použití
Display (DSPOBJAUT) zobrazuje oprávnění k používání objektu
Grant authority (GRTOBJAUT) poskytuje oprávnění k použití objektu
Revoke authority (RVKOBJAUT) odebírá oprávnění k použití objektu
Change (CHG...) mění určité atributy objektu
Clear (CLR...) čistí obsah objektu, ale neruší objekt
Copy (CPY...) kopíruje objekt
Create (CRT...) vytváří objekt
Delete (DLT...) ruší objekt
Display description (DSP...D) zobrazí popis objektu
Dump (DMP...) vypíše obsah objektu
Move (MOV...) přesune objekt do jiné knihovny
Rename (RNMOBJ) přejmenuje objekt
Save (SAV...) uloží objekt na externí nosič
Restore (RST...) obnoví objekt z externího nosiče
Change ownership (CHGOBJOWN) přenáší vlastnictví objektu na jiného uživatele

Specifické operace prováděné uživatelem jsou ty, které se liší podle typu objektu. Např. operace prováděné s databázovým souborem nelze použít na programy (např. čtení záznamu), což systém hlídá.

Uživatelské rozhraní

Uživatelské rozhraní je pojem pro komunikaci uživatele s počítačem. Uživatel komunikuje s počítačem hlavně prostřednictvím tzv. pracovních stanic (work stations). Těmi mohou být obrazovkové alfanumerické terminály nebo osobní počítače. Osobní počítače přitom komunikují se systémem *IBM i* prostřednictvím potřebného programového vybavení (např. Access for Windows). Základní způsob zobrazení informací je textový (alfanumerický); na osobních počítačích existují programy, které ke styku se systémem používají grafické zobrazení (GUI - graphical user interface).

Pro začátečníka:	Soustava nabídek (menus) a nápověda (help)
Pro pokročilejšího uživatele:	Nabídky a náznaky (prompts), seznamy
Pro zkušeného uživatele:	Příkazy operačnímu systému (commands)

Použití těchto prostředků lze libovolně střídat. Tak se lze dostat k požadované funkci několika různými cestami.

Typy obrazovkových formátů

Nabídka (menu) - např. nabídka System Request - nabízí výběr z několika číselných voleb.

```
                              System Request
                                                          System:   IASSIST
Select one of the following:

     1. Display sign on for alternative job
     2. End previous request
     3. Display current job
     4. Display messages
     5. Send a message
     6. Display system operator messages
     7. Display work station user

    80. Disconnect job

    90. Sign off



                                                                       Bottom
Selection
    __

F3=Exit   F12=Cancel
(C) COPYRIGHT IBM CORP. 1980, 2003.
```

Nabídka (menu) se vyvolává příkazem **GO**, zde

GO SYSTEM

Přehled systémových nabídek lze získat vydáním příkazu

WRKMNU MENU(*ALL)

Vstupní formát - např. náznak (prompt) parametrů příkazu DSPMSG - vyzývá k zadání vstupních hodnot (parametrů).

```
                          Display Messages (DSPMSG)

Type choices, press Enter.

Message queue  . . . . . .   *WRKUSR___     Name, *WRKUSR, *SYSOPR...
  Library  . . . . . . . .     __________   Name, *LIBL, *CURLIB
Output . . . . . . . . . .   *_________     *, *PRINT, *PRTWRAP

                                                                      Bottom
F3=Exit   F4=Prompt   F5=Refresh   F10=Additional parameters   F12=Cancel
F13=How to use this display        F24=More keys
```

Seznam - např. výsledek příkazu WRKSPLF - nabízí seznam několika nebo mnoha položek, jimiž lze listovat a u nichž lze zadávat číselné (někdy i znakové) volby.

```
                          Work with All Spooled Files

Type options, press Enter.
  1=Send   2=Change   3=Hold   4=Delete   5=Display   6=Release   7=Messages
  8=Attributes        9=Work with printing status


                             Device or                      Total     Cur
Opt  File        User        Queue       User Data   Sts    Pages    Page  Copy
 _   QPRINT      QSYS        PRT01                   RDY        1             1
 _   QPRINT      QSYS        PRT01                   RDY        1             1
 _   QPRINT      QSYS        PRT01                   RDY        1             1
 _   QPRINT      QSYS        PRT01                   RDY        1             1
 _   QPRINT      QSYS        PRT01                   RDY        1             1
 _   QPRINT      QSYS        PRT01                   RDY        1             1
 _   QPRINT      QSYS        PRT01                   RDY        1             1
 _   QPRINT      QSYS        PRT01                   RDY        1             1
 _   QPRINT      QSYS        PRT01                   RDY        1             1
                                                                          More...
Parameters for options 1, 2, 3 or command
===> ____________________________________________________________________________
F3=Exit   F10=View 4   F11=View 2   F12=Cancel   F22=Printers   F24=More keys
```

Informační formát - např. výsledek příkazu DSPJOBLOG - Zobrazuje jednu nebo i mnoho položek, které lze jen prohlížet, popř. klávesou Help získat podrobnější informace.

```
                              Display All Messages
                                                                 System:   IASSIST
Job . . :   QPADEV0001    User . . :   VZUPKA        Number . . . :   042185



    Job 042185/VZUPKA/QPADEV0001 started on 09/13/05 at 16:01:53 in subsystem
      QINTER in QSYS. Job entered system on 09/13/05 at 16:01:53.
  > /*      */
3 > chgcurlib vzrpg_iv
    Current library changed to VZRPG_IV.
3 > dspmsg
3 > dspmsg
3 > DSPMSG
3 > WRKSPLF
3 > DSPJOBLOG
3 > wrksplf
3 > WRKSPLF SELECT(*ALL)
                                                                        More...
Press Enter to continue.

F3=Exit   F5=Refresh   F12=Cancel   F17=Top   F18=Bottom
```

Nápověda po stisku klávesy Help (popř. F1) funguje ve všech systémových formátech, většinou je proměnlivá v závislosti na poloze kurzoru (kontextová nápověda).

```
 MAIN                           OS/400 Main Menu
.....................................................................
:                         OS/400 Main Menu - Help                   :
:                                                                   :
:      The OS/400 Main (MAIN) menu allows you to select the general task     :
:      you want to do.                                              :
:                                                                   :
:  How to Use a Menu                                                :
:                                                                   :
:      To select a menu option, type the option number and press Enter.      :
:                                                                   :
:      To run a command, type the command and press Enter.  For assistance   :
:      in selecting a command, press F4 (Prompt) without typing anything.    :
:      For assistance in entering a command, type the command and press F4   :
:      (Prompt).  To see a previous command you entered, press F9            :
:      (Retrieve).                                                  :
:                                                                   :
:      To go to another menu, use the   Go to Menu (GO) command. Type GO     :
:      followed by the menu ID, then press the Enter key.  For example, to   :
:                                                              More...  :
:  F3=Exit help   F10=Move to top   F12=Cancel   F13=Information Assistant   :
:  F14=Print help                                                   :
:                                                                   :
:...................................................................:
```

Úrovně asistence

Úroveň asistence se určuje systémovou hodnotou QASTLVL příkazem CHGUSRPRF nebo parametrem ASTLVL v těch příkazech, které mohou mít různé úrovně asistence, např. v příkazu WRKSPLF nebo WRKCFGSTS.

základní úroveň (*BASIC)
střední úroveň (*INTERMED)
pokročilá úroveň (*ADVANCED)

Základní úroveň poskytuje nejrozsáhlejší pomoc a snaží se nepoužívat speciální počítačové pojmy. *Střední* úroveň podporuje všechny funkce a používá počítačové pojmy. *Pokročilá* úroveň podporuje všechny funkce jako střední úroveň, ale využívá prostor obrazovky co nejvíce pro informace na úkor návodů k použití funkčních kláves a číselných voleb.

Zprávy

Zprávy jsou krátká sdělení, která si posílají

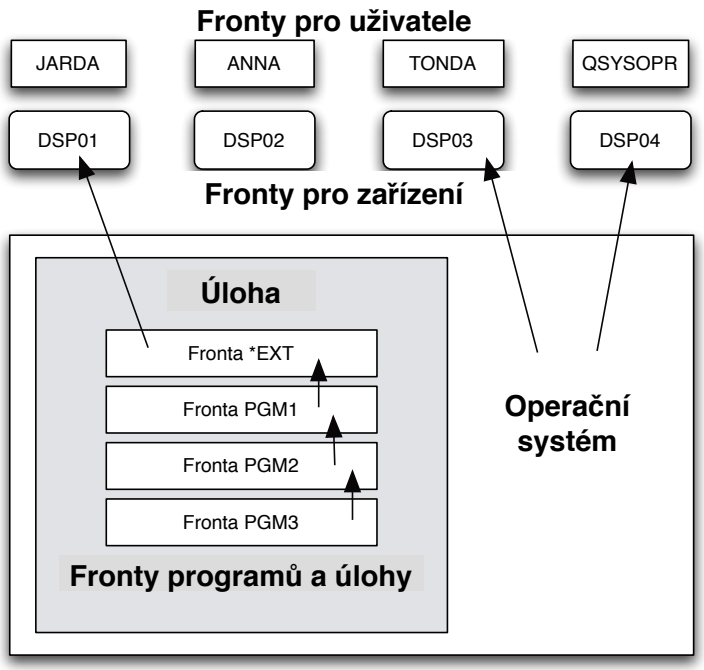
uživatelé navzájem,
programy navzájem,
program uživateli,
uživatel programu (jsou to většinou příkazy CL nebo odpovědi na dotazové zprávy).

Zprávy směřují do tzv. *front zpráv* (message queues) a odtud případně dále na obrazovku nebo do programu. Fronty zpráv jsou objekty typu ***MSGQ**.

Systémové fronty zpráv

pro každého uživatele,
pro každou pracovní stanici,
každému programu přísluší fronta zpráv (call message queue) stejného jména,
každé úloze přísluší fronta zpráv *EXT (externí fronta – k zobrazení na pracovní stanici),
pro systémového operátora QSYSOPR,
pro systémový protokol (history log QHST).

Vztah systémových front s uživateli a programy zachycuje následující obrázek:



Uživatelské fronty zpráv

Uživatel si může vytvořit svoje fronty zpráv příkazem CRTMSGQ a použít je v příkazech.

Způsob doručení zpráv do fronty

Způsob, jakým se na pracovní stanici projeví doručení zprávy do fronty, lze stanovit parametrem **DELIVERY** v příkazu **CHGMSGQ** a **CHGPRF**:

*HOLD	zpráva je zadržena, k zobrazení nutno použít příkaz DSPMSG
*NOTIFY	uživatel je upozorněn zvukovým nebo světelným znamením
*BREAK	přeruší se dosavadní zobrazení a provede se automaticky příkaz DSPMSG
*DEFAULT	zprávy jsou ignorovány, dotazové zprávy jsou automaticky zodpovězeny

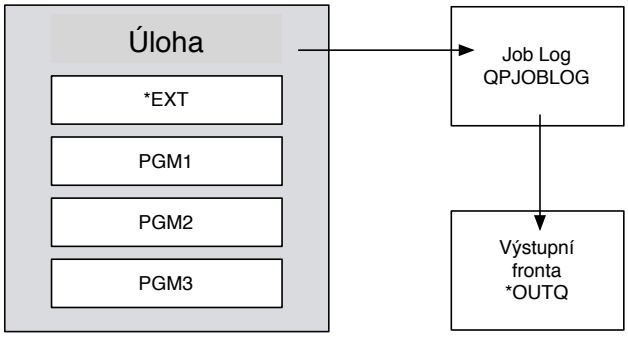
Soubory zpráv

Soubor zpráv je objekt typu ***MSGF** a obsahuje identifikace a texty zpráv, které se používají v programech zpravidla vícenásobně. V textech mohou být definovány proměnné, které se doplňují hodnotami při posílání zprávy. Systémový soubor zpráv se jmenuje QCPFMSGF.

CRTMSGF	Create message file
ADDMSGD	Add Message Description
WRKMSGD	Work with Message Descriptions

Job log

Po skončení úlohy se zprávy (popř. i příkazy CL) z programových front (a z fronty *EXT) zapisují do protokolu o průběhu úlohy (job log):



Obsah protokolu o průběhu úlohy lze řídit příkazem **CHGJOB** (parametr LOG).

Typy zpráv

*INFO	Information - informační zpráva nevyžadující odpověď
*INQ	Inquiry - dotazová zpráva vyžadující odpověď
*RPY	Reply - odpověď na dotazovou zprávu (z programu)
*DIAG	Diagnostic - diagnostická zpráva (zpravidla ze systému)
*NOTIFY	Notify - upozorňovací zpráva (z programu)
*ESCAPE	Escape - úniková zpráva (ukončuje program), lze ji však monitorovat
*COMP	Completion - oznamuje ukončení nějaké funkce (z programu)
*STATUS	Status - oznamuje stav výpočtu (z programu)
*RQS	Request - požadavek na operační systém (příkazy CL jsou *RQS zprávy)

Příkazy pro posílání zpráv

Z příkazového řádku i z programu:

SNDMSG	Send Message (uživateli, prac. stanici, do QSYSOPR, *ALLACT)
SNDBRKMSG	Send Break Message (prac. stanici, *ALLWS)

Jen z programu:

SNDPGMMSG	Send Program Message (uživateli, programu)
SNDUSRMSG	Send User Message (uživateli, programu)
SNDRPY	Send Reply (odpověď na *INQ zprávu)

Příkazy pro zobrazení zpráv:

DSPMSG	Display message - zobrazí zprávy v zadané frontě
WRKMSGQ	Work with message queues - zobrazí seznam front zpráv k práci

Příkazy pro přijímání zpráv:

RCVMSG	Receive message (jen v programu) - čte zprávy z fronty podle výběru
MONMSG	Monitor message (jen v programu) - zachycuje zprávy typu *ESCAPE

Příkazy řídicího jazyka CL

Příkazy operačnímu systému tvoří řídicí jazyk (CL - Control Language, někdy také Command Language). Každý příkaz se skládá ze jména a parametrů. Počet a náplň parametrů závisí na druhu příkazu. Některé parametry nemají žádné parametry. Příkazy jsou objekty typu ***CMD**.

Úplná pravidla pro zápis a použití příkazů jsou uvedena v dokumentaci Control Language.

Platí obecné pravidlo (z něhož se vymykají některé případy), že ve jméně příkazu první tři písmena jsou zkratkou anglického slovesa vyjadřujícího činnost, kterou příkaz od operačního systému vyžaduje. Další část zkratkovitě vyjadřuje předmět, s nímž systém má pracovat. Např.

DSPJOBLOG	DiSPlay JOB LOG - zobrazit "job log" tj. záznam o průběhu úlohy
WRKSPLF	WoRK with SPool Files - pracovat s tiskovými soubory
CRTLIB	CReaTe LIBrary - vytvořit knihovnu

Některé slovesné zkratky

ADD, RMV	Add - přidat, Remove – odstranit
CALL	Call - volat (program)
CHG	Change - změnit
CHK	Check - zkontrolovat
CLR	Clear - vyčistit
CPY	Copy - kopírovat
CRT, DLT	Create - vytvořit, Delete - zrušit
DCL	Declare - deklarovat (proměnnou, soubor) v CL-programu
DSP	Display - zobrazit
DUP	Duplicate - duplikovat
EDT	Edit - upravit, editovat
END, STR	End - ukončit, Start - zahájit
GO	Go - jít (na nabídku - menu)
GRT, RVK	Grant - zaručit, Revoke - odebrat
INZ	Initialize - inicializovat (pásku, disketu, fyzický soubor)
MON	Monitor - monitorovat (zprávy)
MOV	Move - přesunout (objekt)
OPN, CLO	Open - otevřít, Close - uzavřít
OVR	Override - předefinovat, přesměrovat
PRT	Print - tisknout
RCV, SND	Receive - obdržet (zprávu), Send - poslat
RGZ	Reorganize - reorganizovat (data souboru)
RNM	Rename - přejmenovat
RST, SAV	Restore - obnovit, Save - uložit (záložní objekty)
RTV	Retrieve - získat, přečíst (údaje do proměnných)
RUN	Run - spustit
QRY	Query - dotázat se
SBM	Submit - podřídit, předat k dalšímu zpracování
VRY	Vary - změnit vlastnost (zapnout, vypnout)
WRK	Work with - pracovat s (objekty, činnostmi, stavy)

Příklad CL příkazu

```
DSPOBJD OBJ(QRPGLE/QARPGLESRC) OBJTYPE(*FILE) DETAIL(*BASIC)
OUTPUT(*PRINT)
DSPOBJD QRPGLE/QARPGLESRC *FILE *BASIC OUTPUT(*PRINT)
DSPOBJD (QRPGLE/QARPGLESRC) *FILE (*BASIC) OUTPUT(*PRINT)
DSPOBJD OBJ(QRPGLE/QARPGLESRC) OBJTYPE(*FILE) DETAIL(*BASIC)
        OUTPUT(*OUTFILE) OUTFILE(*CURLIB/FILE01) OUTMBR(*FIRST *REPLACE)
```

Příkaz DSPOBJD (Display Object Description) obsahuje několik parametrů, které jsou pojmenovány klíčovými slovy OBJ, OBJTYPE, DETAIL, OUTPUT. Příkaz obsahuje ještě další parametry, které však lze zapsat jen tehdy, když hodnota parametru OUTPUT je *OUTFILE. Hodnoty parametrů jsou uvedeny v závorkách. Parametry jsou odděleny od jména příkazu a mezi sebou mezerami. První tři parametry mohou být zapsány bez klíčového slova a bez závorek (ale i se závorkami). Příkaz obsahuje ve skutečnosti ještě další parametry, které však zde nejsou zapsány, a proto zachovávají předvolené hodnoty. V textu příkazu mohou být i malá písmena.

Jak zjistíme informace o příkazech bez příručky

- Příkaz **GO MAJOR** vyvolá nabídku tematických okruhů a skupin příkazů, z ní lze dále vybírat podtémata, až se dopátráme konkrétního jména příkazu.
- Příkaz **GO CMDxxx** vyvolá seznam příkazů, které obsahují zkratku xxx, např. GO CMDWRK zobrazí nabídku všech příkazů obsahujících zkratku WRK.
- Zapíšeme jméno příkazu na příkazový řádek a stiskneme klávesu **F4**. Zobrazí se základní parametry a jejich názvy. Stiskem klávesy Help nebo **F1** dostaneme nápovědu týkající se toho parametru, na němž se právě nachází kurzor. Ten lze nastavit také na nadpis, a pak se nápověda týká celého příkazu.

Příkazy CL používané v programech

Některé příkazy jazyka CL slouží pouze k programování. Tak např. všechny příkazy dovolující zapsat ve svých parametrech jméno proměnné (začíná znakem &) lze použít jen v CL-programech. Jsou to např. příkazy:

PGM	Začátek programu s případnými parametry
ENDPGM	Konec programu
DCL	Deklarace proměnné
CHGVAR	Změna proměnné (change variable)
IF	Jestliže - podmínkový příkaz
ELSE	Jinak) - opačná podmínka k předchozímu IF
DO	Začátek skupiny příkazů
DOWHILE	Opakování skupiny příkazů se splněnou podmínkou
ENDDO	Konec skupiny příkazů
RCVMSG	Přijetí zprávy (receive message)
MONMSG	Monitor messages - zachycuje programové zprávy typu *ESCAPE
GOTO	Skok na jiné místo v programu (dané návěštím)
RTV...	Retrieve ... - získání informace z určitého objektu

Kromě nich se ovšem může v CL-programech používat většina ostatních příkazů.

Odkud lze vydávat příkazy CL

- z příkazového řádku (režim *INTERACT); klávesy F4 a Help/F1 slouží ke zjištění, jaké má příkaz parametry a co znamenají; klávesou Enter se příkaz spustí.
- z CL programu (režim *BATCH nebo *INTERACT); při pořizování programu lze též použít kláves F4 a Help/F1, klávesa Enter dokončí sestavení textu příkazu.
- z programu ve vyšším jazyku (včetně CL) vyvoláním systémového programu QCMDEXC (command execution) - režim *EXEC,
- z programu REXX (režim *BREXX nebo *IREXX),
- ze vstupního proudu úloh - job stream (zastaralé).

Pozn.: REXX je řídicí jazyk pocházející ze střediskových počítačů a je použitelný na všech počítačích IBM.

Ke zjištění nebo změně povolených režimů u jednotlivých příkazů lze použít těchto příkazů:

DSPCMD	Display Command
CHGCMD	Change Command

Zabezpečení (security)

Zabezpečením se zde rozumí ochrana objektů před zneužitím. Viz IBM i Security reference.

Úrovně zabezpečení (QSECURITY)

10 vyžaduje jen identifikaci uživatele při přihlášení, pak má uživatel přístup ke všem objektům bez výjimky; není již umožněn.

20 vyžaduje při přihlášení identifikaci uživatele a heslo, pak má uživatel přístup ke všem objektům bez výjimky; od verze 7.5 není již umožněn.

30 vyžaduje při přihlášení identifikaci uživatele a heslo a k používání objektů potřebuje oprávnění (autorizaci),

40 stejné jako 30, aktivní kontrola integrity systému a přístup k nedokumentovaným funkcím.

50 stejné jako 40, audit, větší kontrola při předávání řízení programům (od verze 3.1).

Tyto úrovně se zadávají jako systémová hodnota QSECURITY. Při dodávce systému je zadána hodnota 30.

Přístupová práva k objektům (podrobé oprávnění)

Oprávnění k objektu jako celku

*OBJEXIST	Existence - zrušit, ukládat a obnovovat, přenášet vlastnictví
*OBJMGT	Řízení - určovat přístupová práva, přesunovat, přejmenovávat, přidávat členy souboru
*OBJOPR	Provozování - prohlížet popis objektu, používat objekt podle oprávnění k datům
*OBJALTER	Změna - měnit atributy databázových souborů, spouštěče (triggers), referenční omezení, atributy SQL (od verze 3.1)

Oprávnění k údajům v objektu

*READ	Číst údaje z objektu
*ADD	Přidávat údaje do objektu
*UPD	Měnit údaje v objektu
*DLT	Rušit údaje v objektu
*EXECUTE	Spouštět program, nalézt objekt v knihovně (od verze 3.1)

Zabezpečení prostředků (resource security)

Jde o oprávnění uživatele k používání určitých objektů. Takové oprávnění (authority) může přidělit nebo odebrat jen oprávněný uživatel (zpravidla security officer).

Speciální oprávnění

*ALLOBJ	právo ke všem objektům
*SECADM	právo spravovat objekty
*SAVSYS	právo k úklidu a obnově objektů
*JOBCTL	právo k provozu a řízení úloh
*SERVICE	právo servisního pracovníka
*SPLCTL	právo k řízení tiskových front
*AUTDIT	právo provádět revizi
*IOSYSCFG	právo ke změně konfiguračních objektů

Specifické oprávnění (definované systémem)

*ALL	plné právo k objektu
*CHANGE	právo měnit objekt
*USE	právo k použití objektu
*EXCLUDE	žádné oprávnění
*AUTL	oprávnění podle autorizačního seznamu (pouze pro *PUBLIC)

Vztah specifického a podrobného oprávnění

	*ALL	*CHANGE	*USE	*EXCLUDE
*OBJEXIST	X	–	–	–
*OBJMGT	X	–	–	–
*OBJOPR	X	X	X	–
*READ	X	X	X	–
*ADD	X	X	–	–
*UPD	X	X	–	–
*DLT	X	X	–	–
*EXECUTE	X	X	X	–
*OBJALTER	X	X	–	–

Uživatel, který vytvořil určitý objekt, je jeho vlastník (owner) a má k tomuto objektu plné oprávnění (*ALL). Protipólem vlastníka je veřejnost (public), tj. všichni ostatní uživatelé. Nově vytvořenému objektu jsou přiřazena práva pro vlastníka (*ALL) a pro veřejnost (zpravidla *USE).

Třídy uživatelů (user classes)

Třída uživatele je zapsána v uživatelském profilu. Systém definuje tyto třídy uživatelů:

*SECOFR	security officer (bezpečnostní úředník),
*SECADM	security administrator (bezpečnostní správce),
*PGMR	programmer (programátor),
*SYSOPR	system operator (systémový operátor),
*USER	user (uživatel).

Vztah tříd uživatelů a speciálního oprávnění

	***SECOFR**	***SECADM**	***PGMR**	***SYSOPR**	***USER**	
*ALLOBJ	X	<30	<30	<30	<30	
*SECADM	X	X	-	-	-	
*SAVSYS	X	X	X	X	<30	
*JOBCTL	X	X	X	X	-	
*SERVICE	X	-	-	-	-	
*SPLCTL	X	-	-	-	-	
*AUDIT	X	-	-	-	-	(od verze 3.1)
*IOSYSCFG	X	-	-	-	-	(od verze 3.1)

Poznámka: Od verze 7,5 úroveň QSECURITY <30 již není možné nastavit.

Příkazy pro práci s právy k objektům

GRTOBJAUT	Grant object authority
RVKOBJAUT	Revoke object authority
EDTOBJAUT	Edit object authority
ADDDLOAUT	Add document library object authority
EDTDLOAUT	Edit document library object authority
RMVDLOAUT	Remove document library object authority

Práva k datům v objektech IFS (viz dále)

*R	Číst údaje z objektu
*W	Přidávat údaje do objektu
*RW	Číst, měnit, přidávat a mazat údaje v objektu
*X	Spouštět program, nalézt objekt v knihovně nebo v adresáři
*RX	Kombinace *R a *X
*WX	Kombinace *W a *X
*RWX	Kombinace *RW a *X
*NONE	Žádná práva k údajům v objektu
*EXCLUDE	Žádné oprávnění k objektu

Uživatelské profily (user profiles)

Uživatelský profil je objekt typu *USRPRF. Jeho jméno je zároveň identifikací uživatele (user ID). Obsahuje také heslo pro správné přihlášení (sign on). Každý uživatel se musí přihlásit k systému svým jménem (user ID) na základní nabídce (signon menu). Profil plní nejen zabezpečovací funkce, ale také obsahuje mnohé parametry potřebné pro interakční úlohy provozované pod tímto profilem. Uživatelské profily by měly být vytvořeny také pro vzdálené uživatele.

Některé profily dodávané se systémem

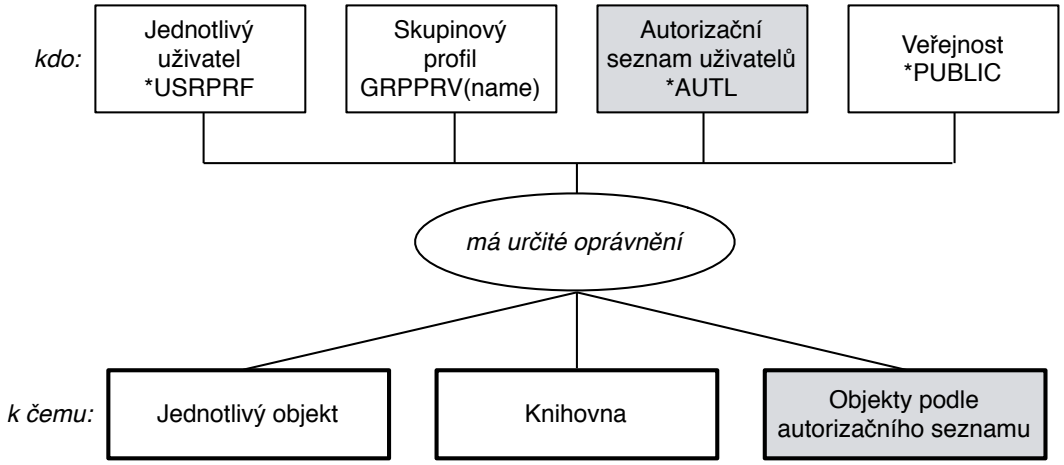
QSECOFR	Security Officer
QPGMR	Programmer
QSYSOPR	System Operator
QUSER	Work Station User

Příkazy pro práci s uživatelskými profily

CRTUSRPRF	Create User Profile
CHGUSRPRF	Change User Profile
CHGPRF	Change Profile
DSPUSRPRF	Display User Profile
DLTUSRPRF	Delete User Profile
WRKUSRPRF	Work with User Profiles

Pozn.: Uživatelské profily jsou také objekty, proto se i na ně vztahují pravidla zabezpečení. Zpravidla s nimi může pracovat jen systémový bezpečnostní úředník (security officer).

Metody autorizace



Jednotlivý uživatel

Uživatel je výslovně oprávněn k určitým objektům. Ty jsou pak vyjmenovány v jeho profilu. Soukromé oprávnění překonává ostatní typy oprávnění. Jestliže je uživatel vlastníkem objektu, má k němu vždy plné právo (*ALL).

Skupinový profil (group profile)

Skupinový profil se vytváří stejně jako profil pro jednotlivého uživatele. Do profilů těch uživatelů, kteří patří do skupiny, se zapíše parametr GRPPRF se jménem toho skupinového profilu. Každý člen ze skupiny pak má všechna práva skupinového profilu a navíc může mít ještě dodatečná práva (individuální). Uživatelský profil může být členem až 16 skupin. Jedna ze skupin je určena jako primární. Práva primární skupiny k objektu jsou uložena v objektu samotném, zatímco práva ostatních skupin jsou uložena v profilech jejich členů. Použití primární skupiny urychluje vyhodnocení přístupu k objektu, protože se nemusí zkoumat skupinový profil, ale stačí jen prohlédnout objekt.

Autorizační seznam (authorization list)

Autorizační seznamy umožňují seskupovat práva více uživatelů k objektu nebo skupině objektů. Autorizační seznam je objekt typu *AUTL, který obsahuje seznam uživatelských profilů s jejich právy ke skupině objektů.

Příkazy pro práci s autorizačním seznamem:

WRKAUTL	Work with authorization list
EDTAUTL	Edit authorization list
EDTOBJAUT	Edit object authority (pro přidávání chráněných objektů)

Veřejnost (public authority)

Veřejností nazýváme všechny uživatele, kteří se mohou přihlásit na pracovní stanici. Také se někdy míní všichni uživatelé kromě vlastníka objektu. Označuje se kódem *PUBLIC.

Převzaté oprávnění (adopted authority)

Programy přejímají práva svého vlastníka (obyčejně programátora), jestliže jsou provozovány jiným uživatelem. Při vytváření programu příkazem CRTxxxPGM určuje parametr USRPRF, pod kterým profilem program poběží. Normálně je zadána hodnota *USER, která poskytuje programu oprávnění uživatele, který program spustil. Lze také zadat hodnotu *OWNER, která dává programu kombinované oprávnění uživatele i vlastníka.

Převzaté oprávnění lze *odejmout* příkazem CHGPGM, kde vyplníme parametr USEADPAUT jako *NO. V tom případě program nepřevezme oprávnění volajícího programu, (který je v zásobníku volání před ním).

Oprávnění se kontroluje jen při prvním přístupu uživatele k objektu (v rámci úlohy).

1. **Jednotlivý uživatelský profil** (Individual user profile) - zkoumá se vždy:

   speciální oprávnění *ALLOBJ
   oprávnění k objektu (vlastnictví a soukromé oprávnění)
   autorizační seznam, je-li objekt uveden v autorizačním seznamu (soukromé oprávnění)

2. **Skupinový profil** (Group profile) - zkoumá se jen tehdy, když individuální uživatelský profil odkazuje na skupinový profil a nebylo zjištěno žádné oprávnění podle individuálního uživatelského profilu:

   speciální oprávnění *ALLOBJ
   oprávnění primární skupiny
   oprávnění k objektu (soukromé oprávnění)
   autorizační seznam, je-li objekt uveden v autorizačním seznamu (soukromé oprávnění)

3. **Oprávnění veřejnosti** (Public authority) - zkoumá se jen tehdy, když nebylo zjištěno žádné oprávnění podle individuálnho ani podle skupinového profilu:

   oprávnění k objektu (oprávnění veřejnosti)
   autorizační seznam, je-li objekt uveden v autorizačním seznamu (oprávnění veřejnosti)

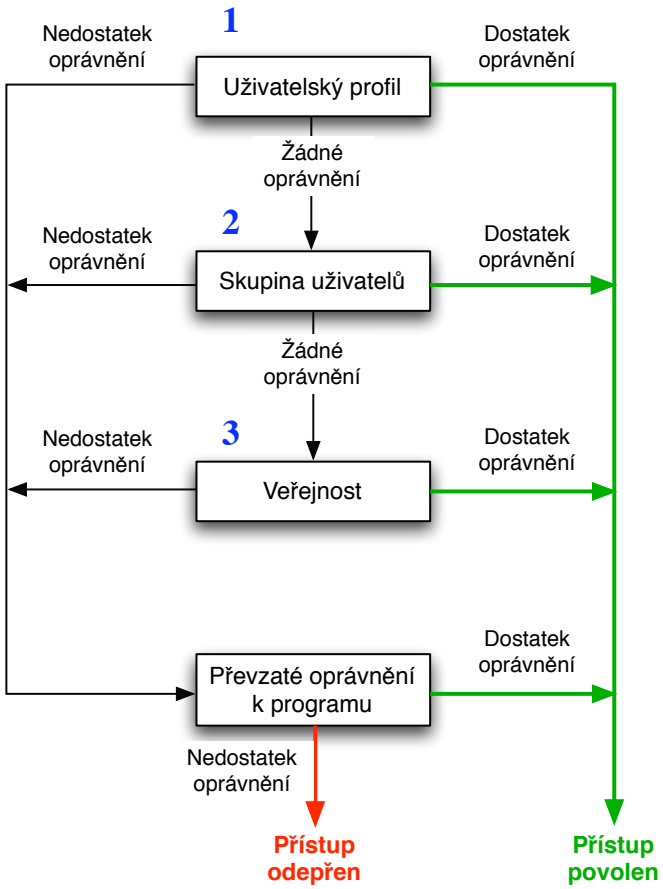
Poznámka 1:

Přímý přístup k objektu lze uživateli zakázat v jeho profilu oprávněním *EXCLUDE (např. příkazem GRTOBJAUT). Tento uživatel však přesto bude moci použít aplikační program, který s oním objektem pracuje.

Poznámka 2:

Tím, že se nejprve kontroluje jednotlivý uživatelský profil a tím, že se uplatní první vyhovující autorizace, lze jednotlivému uživateli udělit větší nebo i menší oprávnění než má skupinový profil, veřejnost a autorizační seznam.

Jak systém vyhodnocuje přístup k objektu



Na obrázku je schematicky znázorněn algoritmus vyhodnocování oprávnění uživatele při prvním pokusu o přístup k objektu. Systém zkoumá oprávnění podle následujícího seznamu podmínek a u prvního vyhovujícího bodu končí. Poslední zjištěné oprávnění pak uplatní na přístup k objektu.

Jde-li o program, přidá se ke zjištěnému oprávnění ještě převzaté oprávnění (adopted authority) programu.

Jestliže žádný bod nevyhoví, neumožní systém uživateli přístup k objektu vůbec.

Řízení práce

Viz Work management.

Úlohy (jobs)

Úloha (job) je určitý objem práce počítače vykonaný v rámci určitého subsystému. V průběhu úlohy může být provedena řada funkcí prostřednictvím příkazů jazyka CL a systémových či uživatelských programů.

Úlohy se dělí podle svého charakteru na dva základní druhy:

interakční (interactive) zahajované a provozované z pracovní stanice,
dávkové (batch) zahajované z jiných zdrojů a provozované nezávisle na pracovních stanicích.

Interakční úlohy

obyčejné - sign-on menu
skupinové (group jobs) - příkaz **TFRGRPJOB** s použitím klávesy Attention
sekundární (secondary job) - klávesa System Request a volba 1 z nabídky

Dávkové úlohy

úlohy zahajované příkazem **SBMJOB** (Submit Job)
úlohy automaticky zahajované při startu subsystému (autostart jobs)
komunikační úlohy zahajované ze vzdáleného počítače (communications jobs)
předem startované komunikační úlohy (prestart jobs)
zapisovací programy (spooling writers) - tisky údajů z výstupních front

Příkazy pro práci s úlohami

WRKUSRJOB	Work with user jobs - zobrazí seznam všech uživatelských úloh
WRKSBMJOB	Work with submitted jobs - zobrazí seznam dávkových úloh spuštěných příkazem SBMJOB
WRKSBSJOB	Work with subsystem job - zobrazí seznam úloh podle subsystémů
WRKACTJOB	Work with active jobs - zobrazí seznam aktivních úloh

Subsystémy

Celý operační *systém* je kontrolován systémovými hodnotami (viz příkazy **DSPSYSVAL**, **CHGSYSVAL**) a síťovými atributy (viz příkazy **DSPNETA**, **CHGNETA**).

Úlohy běží v rámci subsystémů. Subsystém je prostředí, které dovoluje provozovat úlohy určitého typu (např. interakční). Jiný subsystém může umožňovat jen provoz dávkových úloh, jiný zase provoz komunikačních úloh apod. Subsystémy pracují na sobě nezávisle, ale sdílejí společné prostředky počítače.

Dodávané subsystémy

S operačním systémem je dodáváno několik subsystémů, jejichž popisy jsou obsaženy v knihovně QSYS. Podle systémové hodnoty pro tzv. QCTLSBSD řídicí subsystém lze vytvořit dvě varianty standardních subsystémů:

1. varianta (v QCTLSBSD je QBASE):

    QBASE - řídicí subsystém: interakční, dávkové i komunikační úlohy, startuje QSPL
    QSPL - spooling subsystem: vstupní fronty úloh, výstupní tiskové fronty

2. varianta (v QCTLSBSD je QCTL):

    QCTL - řídicí subsystém startuje ostatní subsystémy
    QINTER - interakční subsystém
    QBATCH - dávkový subsystém
    QCMN - komunikační subsystém
    QSPL - spooling subsystem

Všechny subsystémy jsou řízeny tzv. systémovými úlohami, které lze vidět na přehledu aktivních úloh. Spouštějí se automaticky při startu počítače a slouží řízení operačního systému samotného. Jsou to např. tyto úlohy:

SCPF	Start Control Program Function (startuje další úlohy)
QSYSARB	System Arbiter (rozhoduje o prioritách a mj. startuje QLUS)
QLUS	Logical Unit Services (řídí komunikační úlohy)
... aj.	

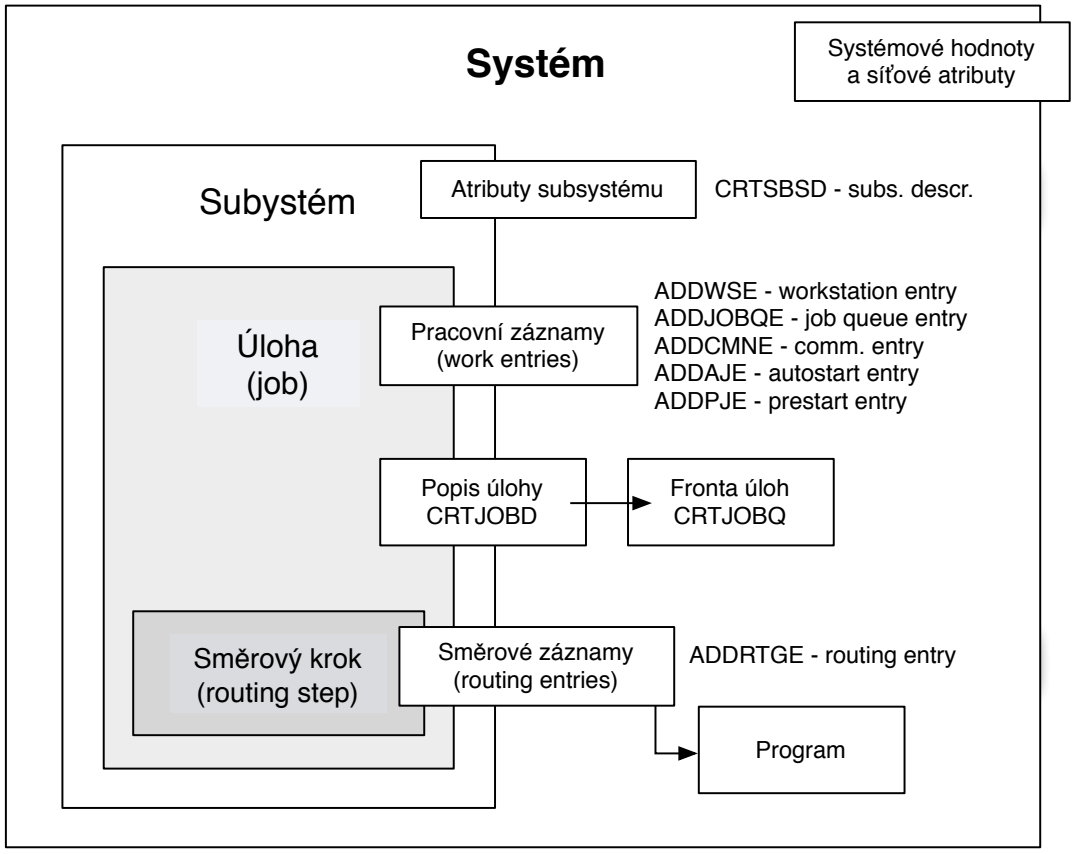
Popisy subsystémů

Vytvoření vlastního subsystému

CRTJOBQ	Create Job Queue (nepoužijeme-li standardní frontu úloh)
CRTJOBD	Create Job Description (nepoužijeme-li standardní popis úlohy)
CRTSBSD	Create Substem Description
ADDWSE	Add Work Station Entry - přidá záznam pro pracovní stanice
ADDJOBQE	Add Job Queue Entry - přidá záznam pro frontu úloh
ADDCMNE	Add Communication Entry - přidá záznam pro komunikační zařízení
ADDAJE	Add Autostart Job Entry - přidá záznam pro automaticky startovanou úlohu
ADDPJE	Add Prestart Job Entry - přidá záznam pro předstartovanou komunik. úlohu
CRTCLS	Create Class - vytvoří třídu úlohy (časové a rychlostní charakteristiky)
ADDRTGE	Add Routing Entry - přidá směrovací krok (pro určení prvního programu)

Z příkazů ADD... (pro pracovní záznamy) použijeme jen ty, které v subsystému chceme mít.

Vztah operačního systému, subsystému, úlohy a souvisejících popisů znázorňuje následující obrázek:



Příkazy pro práci s popisy subsystémů

WRKSBSD	Work with subsystem descriptions - zobrazí seznam popisů subsystémů
DSPSBSD	Display subsystem description - zobrazí popis subsystému
DLTSBSD	Delete subsystem description - ruší popis i odpovídající záznamy

Příkazy pro práci se subsystémy

STRSBS	Start Subsystem - zahajuje (spouští, startuje) subsystém
WRKSBS	Work with subsystems - zobrazí seznam subsystémů k práci
ENDSBS	End Subsystem - ukončuje subsystém
ENDSYS	End System - ukončuje všechny subsystémy

Standardní subsystémy se spouštějí při startu počítače podle předpisu zadaného v CL programu QSYS/QSTRUP. (Jeho zdrojový text lze získat příkazem RTVCLSRC.)

Úlohy mohou být spuštěny - zahájeny - teprve po zahájení příslušného subsystému. Zahajování úloh se liší podle jejich typu.

Systémové hodnoty (system values)

Systémové hodnoty jsou údaje uložené v počítači, které samy nejsou objekty, ale určují stav a chování počítače jako celku, tedy i všech objektů.

Některé systémové hodnoty:

QACTJOB	Počáteční nejvyšší počet aktivních úloh
QCCSID	Standardní soubor znaků (Coded Character Set ID)
QCONSOLE	Jméno pracovní stanice sloužící jako systémová konzola
QDATE	Systémové datum
QTIME	Systémový čas
QSRLNBR	Sériové číslo počítače
QDECFMT	Tvar dekadického čísla
QLANGID	Kód jazyka
QSECURITY	Stupeň zabezpečení
QSYSLIBL	Systémová část seznamu knihoven
QUSRLIBL	Uživatelská část seznamu knihoven

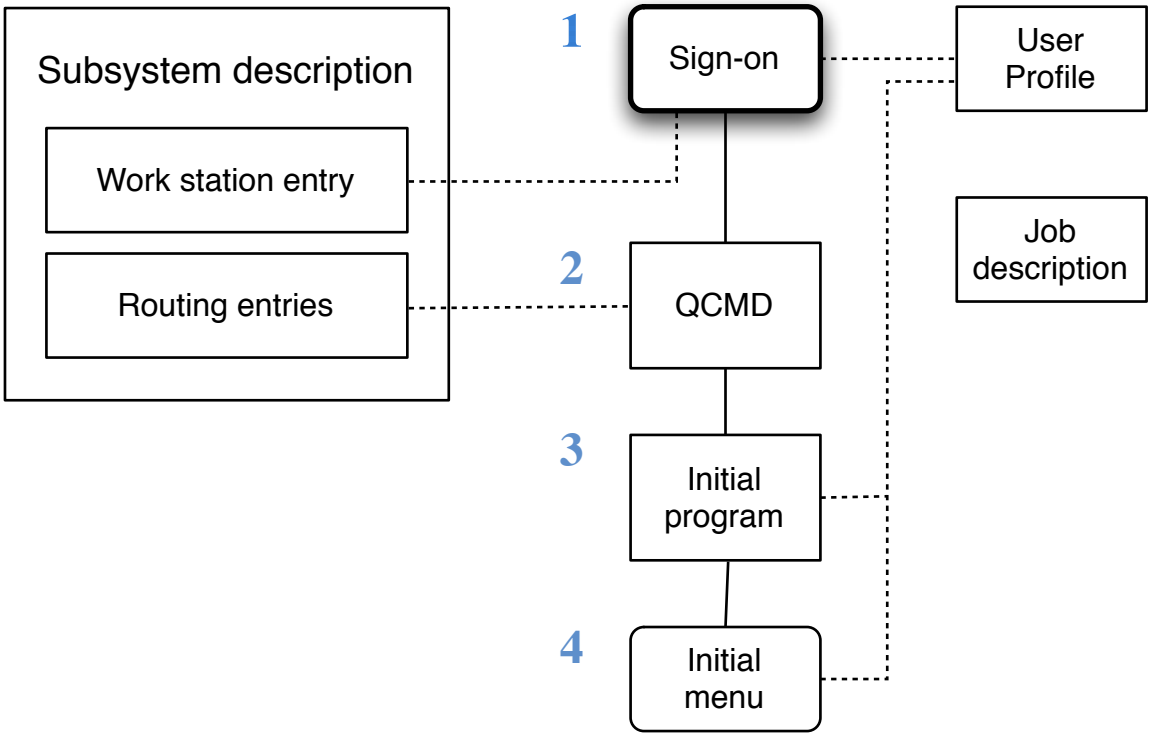
a mnoho dalších

Systémové hodnoty může prohlížet každý uživatel (veřejnost), měnit je může jen oprávněný uživatel, zpravidla bezpečnostní úředník (security officer).

Příkazy pro systémové hodnoty:

WRKSYSVAL	Work with System Values
CHGSYSVAL	Change System Value
DSPSYSVAL	Display System Value
RTVSYSVAL	Retrieve System Value

Zahájení interakční úlohy



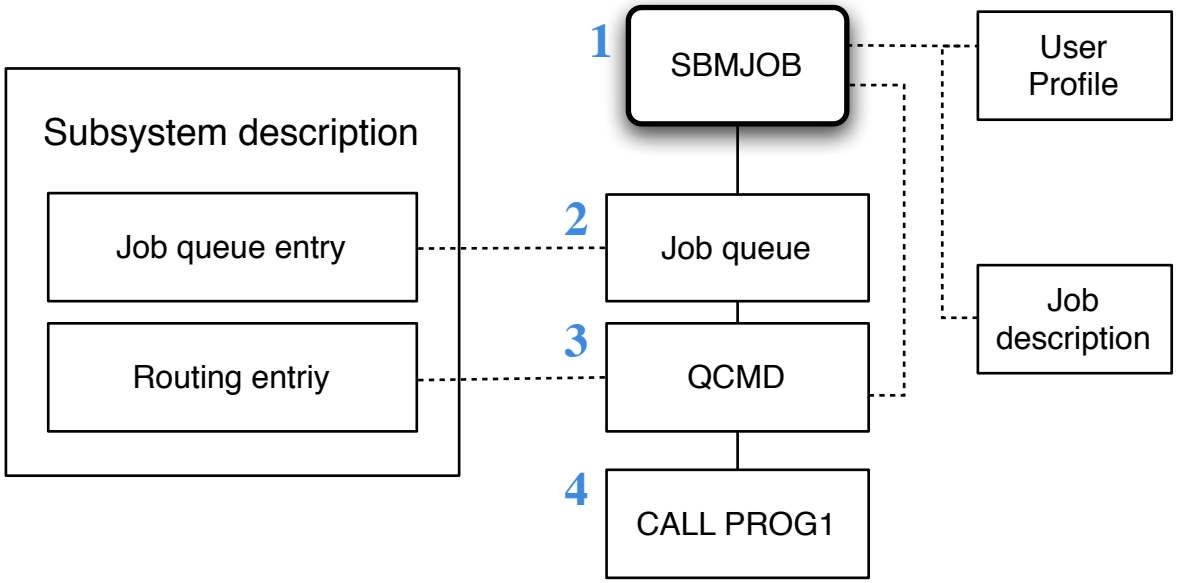
1. Po startu subsystému se na zapnuté pracovní stanici zobrazí přihlašovací obraz - Sign-on display. Uživatel se přihlásí zápisem svého uživatelského jména (user ID) a hesla. Systém si najde uživatelský profil, záznamy o pracovních stanicích (work station entries) v popisu subsystému a zjistí údaje z popisu úlohy (job description).

2. V popisu subsystému nalezne program, který má spustit: z popisu úlohy přečte směrovací data (routing data) a porovná je se směrovacími záznamy (routing entries) v popisu subsystému. V nalezeném směrovacím záznamu pak zjistí jméno progamu, který má spustit. Je to zpravidla program QCMD pro zpracování příkazů CL.

3. Program QCMD buď spustí úvodní program, je-li zadán v uživatelském profilu (parametr INLPGM), nebo

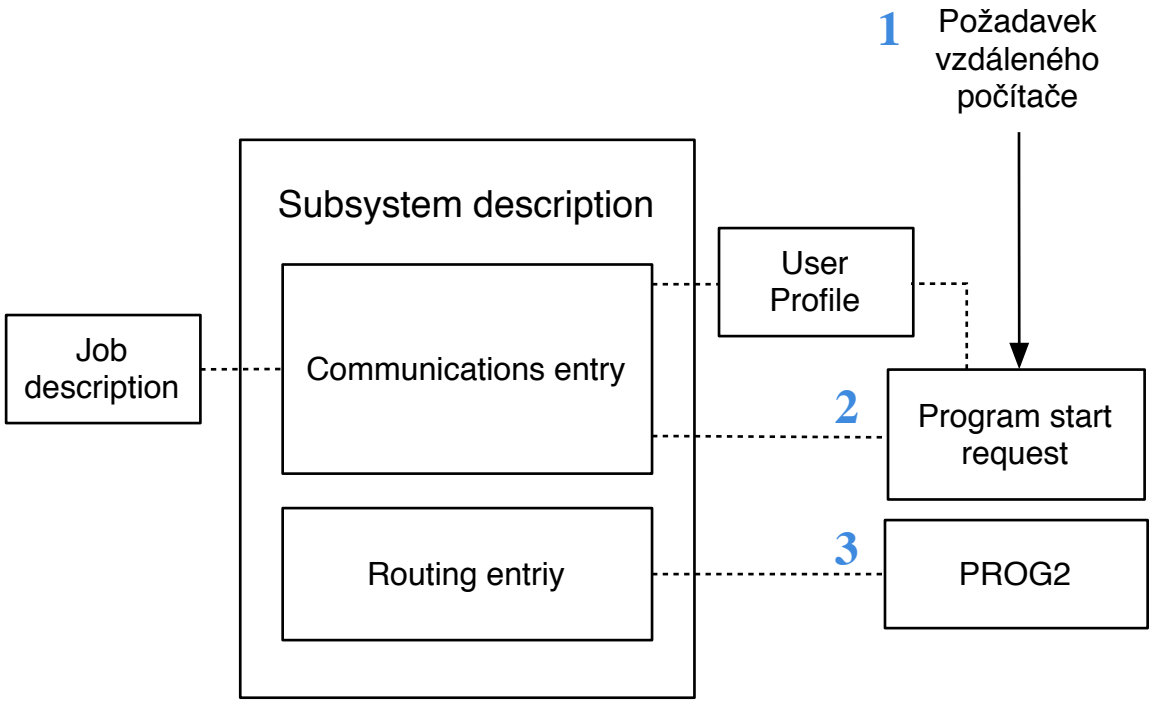
4. zobrazí úvodní nabídku (menu), jejíž jméno nalezne v uživatelském profilu (parametr INLMNU).

Zahájení dávkové úlohy



1. Uživatel nebo program vydá příkaz SBMJOB. Tento příkaz může být vydán z příkazového řádku, z programu běžícího v interakčním režimu nebo z programu běžícího v dávkovém režimu. Systém si z příkazu SBMJOB zjistí popis úlohy (job description), kde je mj. zadán uživatelský profil (user profile), pod nímž úloha poběží. Obyčejně se přebírá běžný profil (*CURRENT), tj. ten, pod kterým byl vydán příkaz SBMJOB. V příkazu SBMJOB je také zpravidla zadáno jméno příkazu, který se má provést jako první v rámci úlohy (parametr CMD).
2. Vytvořená úloha se umístí do fronty úloh zjištěné z popisu úlohy (standardně do fronty QBATCH).
3. V subsystému, k němuž je fronta přiřazena, se vyhledá potřebný směrovací záznam (routing entry) a v něm jméno programu, který se má spustit. Je to zpravidla program QCMD umožňující provést příkaz uvedený v SBMJOB.
4. Jakmile úloha přijde ve frontě na řadu, je uvolněna a zadaný příkaz (z parametru CMD) se provede. Tím je úloha spuštěna.

Zahájení komunikační úlohy

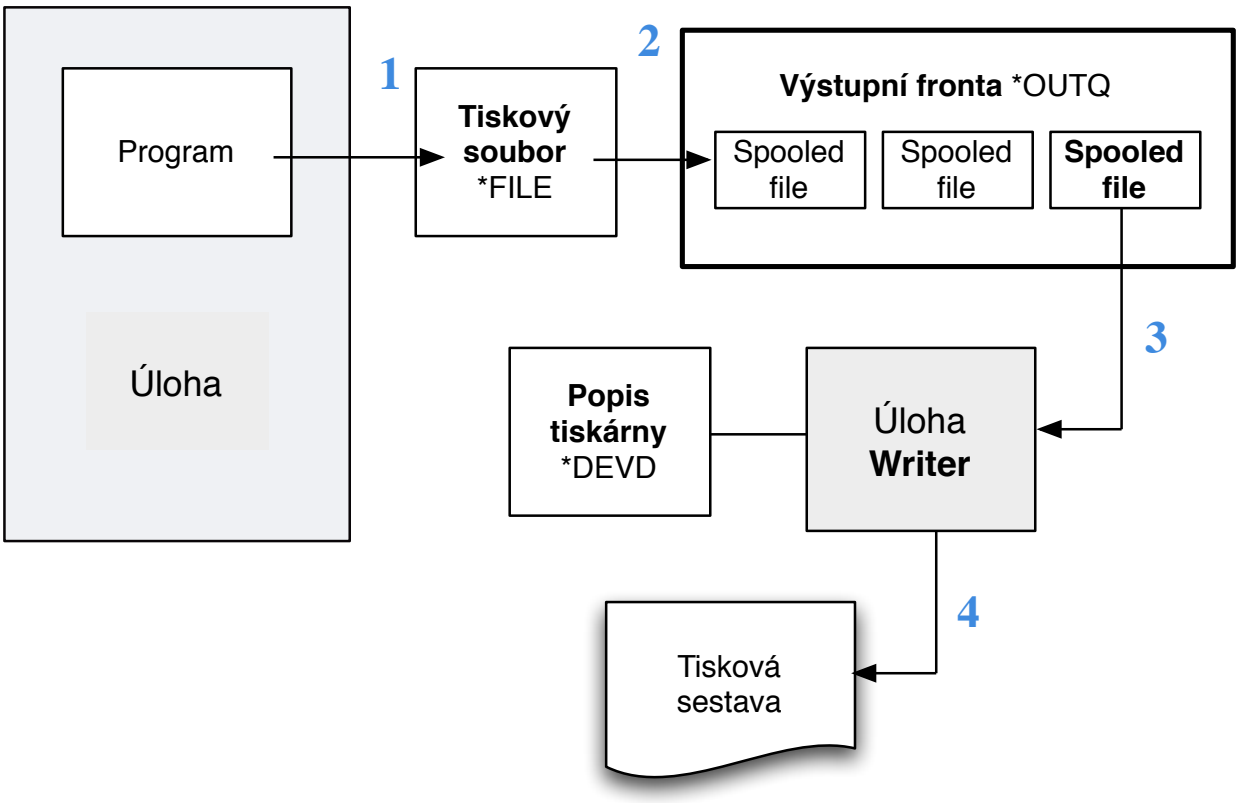


1. Požadavek na zahájení úlohy přichází ze vzdáleného počítače ve formě datové struktury (program start request). V něm je uvedena zkratka PGMEVOKE (v pozici 29).

2. Požadavek nejprve vyhledá komunikační subsystém, v němž má být program spuštěn (zpravidla QCMN, popř. QBASE). V něm najde komunikační záznam (communications entry) a směrovací záznam (routing entry) podle určitých pravidel. Jméno uživatelského profilu je převzato buď z požadavku na start programu nebo z komunikačního záznamu. Popis úlohy je určen v komunikačním záznamu.

3. Program, který je určen buď v požadavku na start programu nebo ve směrovacím kroku, se spustí a zahájí tak komunikační úlohu. Úloha však může být zahájena předem (prestart job) a čeká, dokud nepřijde požadavek ze vzdáleného počítače.

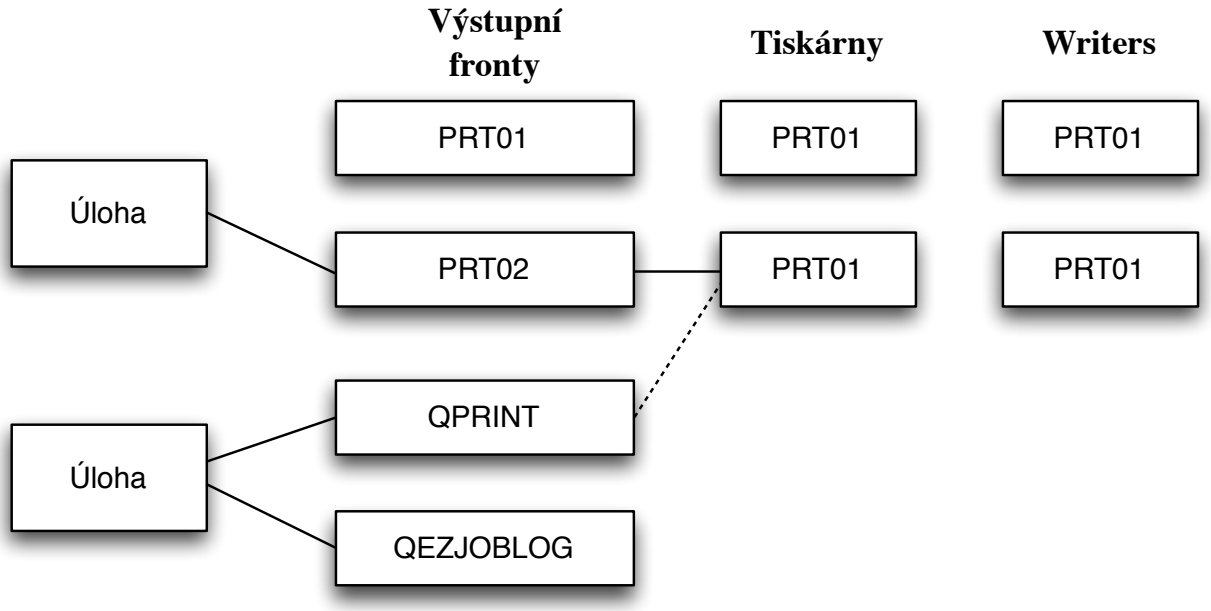
Úlohy tiskového výstupu (output spooled jobs)



1. Program (běžící v nějaké úloze) produkuje tiskový výstup s použitím tiskového souboru (objekt typu *FILE s atributem PRTF).
2. Tento výstup obsahující informace potřebné pro tisk (např. řídicí znaky pro posuv papíru) je zapsán do výstupní fronty (objekt typu *OUTQ) jakožto tzv. spooled file, tj.soubor připravený k tisku. (Spooled file není objekt!) Ve frontě bývá více souborů čekajících na vytištění. Jejich pořadí lze měnit.
3. Soubory jsou z fronty vybírány systémovým tiskovým programem (writer) a posílány na příslušnou tiskárnu (popsanou objektem typu *DEVD).
4. Tiskárna nakonec soubor vytiskne na papír.

Souvislost mezi úlohami, výstupními frontami a tiskárnami

Úlohy posílají výstupy do front podle určení v popisu úlohy. Výstupní fronty mohou a nemusí být přiřazeny tiskárnám. Tiskárnám jsou přiřazeny standardní fronty, které se jmenují stejně. Stejně se také jmenují tiskové systémové programy (writers), které příslušejí jednotlivým tiskárnám a běží jako samostatné dávkové úlohy.



Spojení mezi úlohou, frontou a tiskárnou

Dokud úloha ještě neběží, lze přiřazení tiskárny a fronty měnit na různých úrovních.

Následující seznam je uveden v pořadí priorit. Např. údaje v popisu souboru mají přednost před údaji v popisu úlohy, atd.

1. Popis souboru - **CHGPRTF** DEV(PRT01) OUTQ(PRT01)
2. Popis úlohy - **CHGJOBD** PRTDEV(PRT01) OUTQ(PRT01)
3. Popis uživatelského profilu - **CHGUSRPRF** PRTDEV(PRT01) OUTQ(PRT01)
4. Popis pracovní stanice - **CHGDEVDSP** PRTDEV(PRT01) OUTQ(PRT01)
5. Systémová hodnota - **CHGSYSVAL** QPRTDEV(PRT01) - Standardní systémová tiskárna

Za běhu úlohy lze měnit přiřazení tiskárny a fronty následujícími příkazy.

CHGJOB PRTDEV(PRT01) OUTQ(PRT01) - Změna tiskárny a fronty před spuštěním programu. Překonává údaj z popisu úlohy.
OVRPRTF PRTDEV(PRT01) OUTQ(PRT01) - Změna tiskárny a fronty před spuštěním programu. Překonává údaj v popisu úlohy.

Po skončení úlohy lze měnit přiřazení tiskárny a fronty následujícím příkazem.

CHGSPLFA DEV(PRT01) OUTQ(PRT01) - Změna tiskárny nebo přesun z jedné fronty do druhé (je-li již tiskový výstup v první frontě).

Příkazy pro práci s výstupními frontami:

WRKOUTQ	Work with output queue - zobrazí seznam výstupních front k další práci
CRTOUTQ	Create output queue - vytváří novou frontu
CLROUTQ	Clear outpur queue - vyčistí frontu (smaže všechny soubory)

Příkazy pro práci s tiskovými soubory (spooled files):

WRKSPLF	Work with spooled files
CPYSPLF	Copy spooled file - kopíruje tiskový soubor do databázového souboru

Příkazy pro práci s tiskovými programy (writers):

WRKWTR	Work with writers - umožní připojování a odpojování tiskáren.

Popisy úloh

Každá úloha je popsána popisem úlohy (objektem typu ***JOBD**). V něm je zapsána řada charakteristik (atributů), které jsou relativně stálé a mohou být použity jako standardní. Jsou to mj. tyto údaje:

jméno fronty úloh
priorita výběru z fronty úloh
priorita při výběru z výstupní fronty pro tisk
jméno tiskárny,
jméno výstupní fronty
způsob protokolování průběhu úlohy (job log)

Příkazy pro popisy úloh:

CRTJOBD	Create job description
CHGJOBD	Change job description
WRKJOBD	Work with job descriptions

Třídy úloh

Třída (class) je objekt typu ***CLS**; obsahuje údaje o dynamických vlastnostech úlohy (tj. vlastnosti směrového kroku, kterým se zahajuje úloha). Jméno třídy se zapisuje *do směrového záznamu (routing entry)* v popisu subsystému. Nejdůležitější údaje ve třídě jsou tyto:

- priorita výpočtu (run priority) vzhledem k ostatním úlohám; 1 je nejvyšší, 99 nejnižší; obvykle 50 pro dávkové úlohy, 20 pro interakční úlohy,
- časový interval (time slice), po nějž může úloha běžet bez přerušení; nejpozději po jeho uplynutí dostanou příležitost ostatní úlohy (např. 2000 milisekund),
- čekací doba (wait time) - např. 30 vteřin, po kterou má výpočet čekat na nějaký prostředek, např. při čtení databázového záznamu, který je právě zamčený,
- nejdelší povolený celkový čas pro výpočet (standardně *NOMAX),
- největší dočasná operační paměť (standardně *NOMAX).

Příkazy pro práci s třídami:

CRTCLS	Create class
CHGCLS	Change class
DSPCLS	Display Class
WRKCLS	Work with classes

Systémových tříd je mnoho. Nejběžnější jsou:

QBATCH (Batch Subsystem Class)

```
                        Display Class Information
                                                              System:   XASSIST
 Class . . . . . :   QBATCH
   Library . . . . . :     QSYS
 Run priority  . . . . . :   50
 Time slice in milliseconds  . . . . . :   5000
 Eligible for purge  . . . . . :   *NO
 Default wait time in seconds  . . . . . :   120
 Maximum CPU time in milliseconds  . . . . . :   *NOMAX
 Maximum temporary storage in megabytes  . . . . . :   *NOMAX
 Maximum threads . . . . . :   *NOMAX
 Text  . . . . . :   BATCH SUBSYSTEM CLASS
```

QCTL (Controlling Subsystem Class)

```
                        Display Class Information
                                                              System:   XASSIST
 Class . . . . . :   QCTL
   Library . . . . . :     QSYS
 Run priority  . . . . . :   10
 Time slice in milliseconds  . . . . . :   2000
 Eligible for purge  . . . . . :   *YES
 Default wait time in seconds  . . . . . :   30
 Maximum CPU time in milliseconds  . . . . . :   *NOMAX
 Maximum temporary storage in megabytes  . . . . . :   *NOMAX
 Maximum threads . . . . . :   *NOMAX
 Text  . . . . . :   CONTROLLING SUBSYSTEM CLA
SS
```

Podpora práce s daty

Zařízení (devices)

Zařízením se původně rozumí přístroj (kromě disku) připojený k počítači místně (local device) nebo vzdáleně komunikační linkou (remote device). Později také osobní počítač s instalovaným programem Access for Windows, ještě později IBM i Access Client Solutions.

Lokální zařízení

- displejová pracovní stanice, tj. obrazovkový terminál IBM 5151 připojený twinaxiálním kabelem (zastaralé)
- tiskárna připojená twinaxiálním kabelem (zastaralé)
- pásková jednotka
- zařízení na vyjímatelná média (removable mass storage - RMS)

Vzdálené zařízení

- displejová pracovní stanice, tj. obrazovkový terminál připojený komunikační linkou (zastaralé)
- tiskárna připojená komunikační linkou
- jiný počítač připojený komunikační linkou (zařízení je vzdálený program)

Emulované nebo virtuální displejové zařízení

Emulované zařízení 5250, 3270 a VTxxx na PC je ovládáno přes protokol TCP/IP a server *Telnet*. Jména zařízení lze volit, ale zpravidla se ponechává automatická tvorba standardních názvů QPADEV0001, QPADEV0002, ... , QPADEV000A, ... až QPADEVZZZZ.

Originální emulátor zařízení 5250 je obsažen v aplikaci IBM i Access Client Solutions spolu s dalšími prostředky.

Další emulátor zařízení lze stáhnout bezplatně ze stránky tn5250j.

Popisy zařízení a konfigurace

Každé zařízení musí mít svůj popis, což je objekt typu ***DEVD**. Skupina zařízení podobného druhu je popsána tzv. řadičem (controller), což je objekt typu ***CTLD**. Popisy zařízení mohou být vytvořeny automaticky nebo příkazem CRTDEVxxx. Rovněž popisy řadičů lze buď ponechat operačnímu systému nebo je vytvořit příkazem CRTCTLxxx. Pro každý druh komunikační linky (typ ***LIND**) musí být ještě vytvořen popis příkazem CRTLINxxx, na nějž se odvolávají popisy řadičů.

Popisům zařízení, řadičů a linek se společně říká *konfigurace*. Přehled o konfiguracích lze získat pomocí příkazu

WRKCFGSTS Work with configuration status.

V konfiguračních popisech se vyskytuje parametr *resource name* (jméno prostředku, zdroje), což je pevné jméno hardwarové součásti - mj. též zařízení, řadiče či linky, tak jak je stanovil výrobce. Tato jména zjistíme pomocí příkazů

WRKHDWPRD Work with hardware products
WRKHDWRSC Work with hardware resources.

Automatická konfigurace *lokálních* zařízení a řadičů se provádí tehdy, když systémová hodnota QAUTOCFG je '1'. Způsob pojmenování všech automaticky vytvořených popisů pro lokální zařízení je řízen systémovou hodnotou QDEVNAMING podle následující tabulky:

Zařízení	*NORMAL	*DEVADR
Displejové stanice	`DSP01, DSP02, ...`	`DSP010000, ...`
Tiskárny	`PRT01, PRT02, ...`	`PRT010301, ...`
Řadiče displ. stanic	`CTL01, CTL02, ...`	`CTL01, CTL02, ...`
Páskové jednotky	`TAP01, TAP02, ...`	`TAP01, TAP02, ...`
Optické jednotky	`OPT01, OPT02, ...`	`OPT01, OPT02, ...`

Poznámka: Optické jednotky zahrnují obsluhu médií CD, DVD, RMS (removable storage, tj. vyjímatelná paměť), např. USB flash disk, aj.

Konfigurace komunikačních linek, řadičů a zařízení se vytvářejí podle jiných pravidel a jsou z povahy věci složitější. Podrobněji se o nich pojednává v kapitole o komunikacích.

Popisy dat (souborů) pro lokální zařízení

Zařízení se ovládají pomocí popisů souborů (file descriptions), což jsou objekty typu ***FILE**. Každý druh zařízení používá své vlastní příkazy pro vytvoření popisu souboru:

CRTDSPF	Create display file - displejový soubor *s externím popisem DDS*
CRTPRTF	Create printer file - tiskárnový soubor *s externím popisem DDS nebo bez něj*
CRTTAPF	Create tape file - páskový soubor *bez externího popisu*
CHG...F	mění charakteristiky souboru trvale
OVR...F	mění charakteristiky souboru jen do skončení úlohy

- *Displejový soubor* se popisuje externě pomocí **DDS** (Data description specifications). V externím popisu se určuje rozvrh údajů na obrazovce (formát). Určí se, které údaje jsou výstupní, vstupní, popř. obousměrné, rozmístí se doprovodné údaje (textové konstanty, nápověda, aj.). Jednotlivým údajům lze přidat zobrazovací atributy (jas, podtržení, blikání apod.), barvy a další vlastnosti, které tak není třeba určovat v programu. Jednotlivá datová pole a jejich vlastnosti mohou být podmíněny indikátory nastavenými v programu. Pomocí indikátorů se také definují a testují funkční klávesy. K návrhu obrazovkových formátů slouží program **SDA** (Screen Design Aid).
- *Tiskárnový soubor* se popisuje pomocí **DDS** (Data description specifications), když tištěné údaje jsou tvarovány složitěji než na prosté řádkové tiskárně, např. v různých velikostech a druzích písma, v přesně stanoveném rozvrhu stránky. Jednotlivá datová pole mohou být podmíněna indikátory. U jednodušších tisků se rozvrh údajů v řádcích a na stránce popisuje v programu bez DDS. K návrhu tiskových formátů DDS slouží program **RLU** (Report Layout Utility).
- *Páskové soubory* nemohou používat externích popisů dat. Zpracovávají se na úrovni celého záznamu.

Integrovaná databáze DB2

Popisy databázových souborů DDS

Data se organizují v databázových souborech (objekty typu *FILE). Fyzické soubory se rozlišují na datové (podtyp PF-DTA) a zdrojové (podtyp PF-SRC). Do zdrojových souborů se umisťují texty programů a popisů různých objektů. Fyzické soubory se skládají ze záznamů a mohou být buď nestrukturované (flat files) nebo se strukturou definovanou v externím popisu DDS (Data Description Specification).

Struktura záznamu může být popsána v programu nebo mimo něj (externě). Datový fyzický soubor se skládá ze záznamů a každý záznam se skládá z polí. Popis v programu (interní) je pevně spjat s programem a při každé změmě jeho struktury musíme změnit zdrojový program a přeložit jej.

V externím popisu DDS je definována struktura záznamu: charakteristiky polí, klíč souboru další vlastnosti. Externí popis je pevně spojen se souborem. Při použití externího popisu nemusíme vždy provádět změny do zdrojvého programu, ale mnohdy stačí znovu jej přeložit.

Výhody externích popisů:

- zvýšená produktivita programovací práce - stačí popsat strukturu záznamu jen jednou pro všechny programy, které jej používají,
- snadná údržba souborů a programů - změny v popisu souboru se provedou jen na jednom místě a není třeba měnit každý program, který popis používá,
- lepší zajištění celistvosti dat - programy používající tatáž jména polí, používají také tatáž data, protože se odkazují na stejný popis,
- automatická kontrola verzí - operační systém hlídá, zda verze externího popisu z doby, kdy byl program přeložen, odpovídá verzi externího popisu z doby spouštění progamu; tato kontrola se provádí při otevírání souboru,
- výběr a setřídění dat mohou probíhat mimo program - lze použít externího popisu klíčových polí,
- několik souborů lze spojit do jednoho ještě před použitím v programu.

Typy databázových souborů

Fyzické soubory - obsahují data.

Logické soubory - určují přístupovou cestu k fyzickým souborům, neobsahují data.

Fyzické soubory

datové (PF-DTA) obsahující jeden nebo někdy i více datových členů (members)
zdrojové (PF-SRC) obsahující zpravidla více datových členů
referenční neobsahující žádná data, jen popisy datových polí

Logické soubory

Dělí se na obyčejné a spojené

Obyčejné logické soubory mohou obsahovat

jeden nebo více formátů,
s vlastními formáty,
s formáty převzatými z fyzických nebo z jiných logických souborů.

Spojené logické soubory spojují záznamy z několika fyzických souborů do jednoho záznamu (podle párovacích polí), mohou však být použity jen jako *vstupní*.

Organizace databázových souborů

Datové záznamy (věty) jsou v souboru organizovány v souladu s tzv. *přístupovou cestou* (access path). Fyzické pořadí záznamů tak, jak postupně přibývají, se nazývá příchozí pořadí (arrival sequence) nebo také příchozí přístupová cesta (arrival access path). Je-li u souboru definován klíč (key), je tím zároveň definována tzv. přístupová cesta (access path). Někdy se přístupová cesta nazývá *index*.

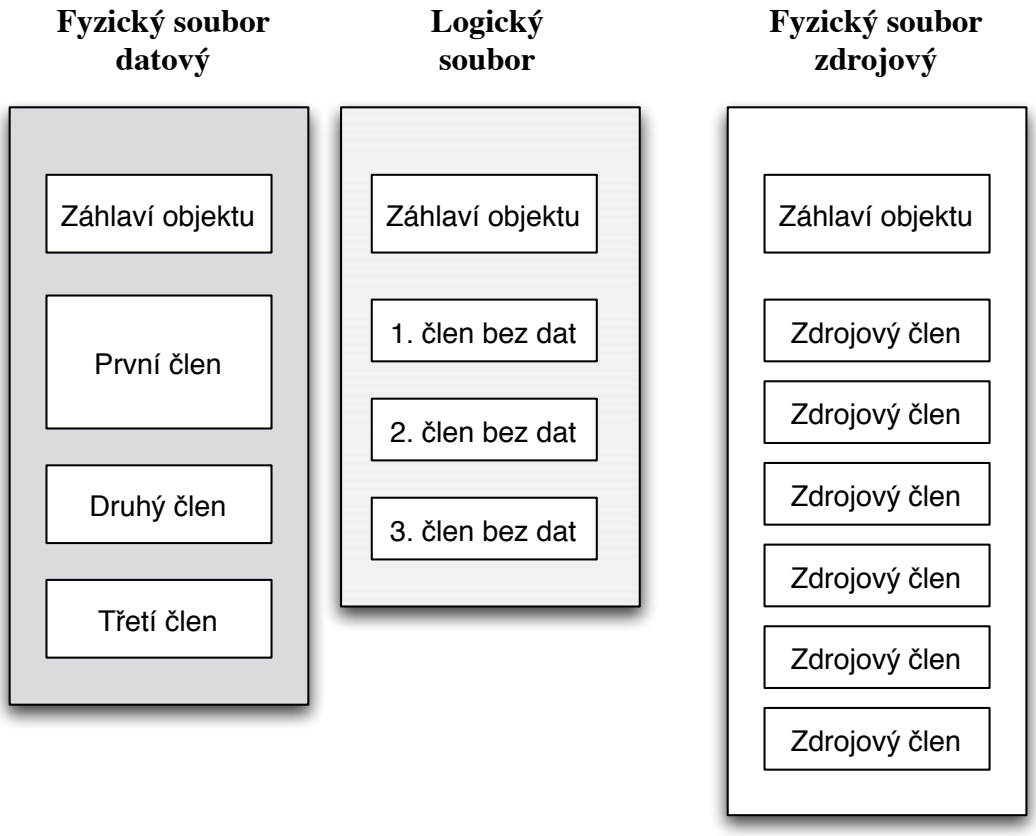
Fyzické soubory mohou být definovány jak s klíčem, tak bez klíče. *Klíč* může být tvořen jedním nebo více poli záznamu (třeba i všemi). Klíč však nemusí být použit v programu. V tom případě (a ovšem i u souborů bez klíče) se záznamy zpracují v pořadí jejich vzniku nebo přímo podle pořadového čísla.

Klíč fyzického souboru může být zadán jako jedinečný (unique). V tom případě systém nepovolí zapsat věty s duplicitním klíčem. Není-li klíč zadán jako unikátní, lze stanovit pořadí zpracování záznamů se stejným klíčem.

Datové členy (data members) jsou části souborů obsahující data - záznamy. Fyzický soubor může mít jeden nebo více členů (referenční soubor nemá mít žádný), zpravidla jde o verze či generace. Např. měsíční obraty zásob mohou být uloženy ve dvanácti samostatných členech pojmenovaných OBR01, OBR02, atd. Většina datových souborů mívá jen jeden člen. Další členy se vytvářejí příkazem ADD**P**FM.

Zdrojové soubory mají zpravidla větší množství členů - zdrojových programů či jiných textů. Jména členů jsou zpravidla stejná jako jména objektů, které se z nich vytvářejí (např. kompilací programů).

I logické soubory mohou mít členy. Logický člen reprezentuje přístupovou cestu k datům fyzického členu. První člen logického souboru se vytváří se souborem, další členy se vytvářejí příkazem ADD**L**FM.



Vytváření a změna souborů

Databázový souboru je objekt typu ***FILE** a má atribut PF (physical file) nebo LF (logical file).

Příkazy používané s fyzickými soubory:

CRTPF	Create Physical File
CHGPF	Change Physical File
CRTSRCPF	Create Physical File
CHGSRCPF	Change Physical File
ADDPFM	Add Physical File Member
CLRPFM	Clear Physical File Member
DSPPFM	Display Physical File Member
RGZPFM	Reorganize Physical File Member
INZPFM	Initialize Physical File Member

Příkazy používané s logickými soubory:

CRTLF	Create Logical File
CHGLF	Change Logical File
ADDLFM	Add Logical File Member
CHGLFM	Change Logical File Member

Společné příkazy používané se soubory:

DLTF	Delete File
DSPFD	Display File Description
DSPFFD	Display File Field Description
RMVM	Remove Member
RNMM	Rename Member

Vytvoření souborů s DDS (Data Description Specification)

Následujích příklady znázorňují externí popisy databázových souborů. Zápisy jsou uspořádány ve sloupcích formuláře DDS.

Popis a vytvoření referenčního souboru

```
 ************************************************************
 *   Referenční soubor REFMZP (materiálové zásoby).         *
 *   Neměl by obsahovat žádný datový člen - MBR(*NONE) v CRTPF.   *
 *   Slouží jako definice všech datových polí dané agendy.  *
 ************************************************************
A.........T.Name++++++RLen++TDpB......Functions++++++++++++++++++++++++++
A          R REFMZPF0
 *   CCCCCCCCCC
A            CENA          10P 2       COLHDG('Cena/j.')
 *   MMMMMMMMM
A            MATER          5          COLHDG('Číslo' 'mater.')
A            MNOZ          10P 2       COLHDG('Množství' 've skladu')
A            MNOBR         10P 2       COLHDG('Množství' 'obratu')
 *   NNNNNNNNN
A            NAZEV         50          COLHDG('Název materiálu')
 *   SSSSSSSSS
A            SKLAD          2          COLHDG('Skl')
 *   ZZZZZZZZZ
A            ZAVOD          2          COLHDG('Zav')
```

```
CRTPF FILE(*CURLIB/REFMZP) SRCFILE(*LIBL/QDDSSRC) SRCMBR(*FILE) MBR(*NONE)
```

Popisy a vytvoření fyzických souborů

```
 ************************************************************
 *   Soubor CENIKP - Ceník materiálu                        *
 ************************************************************
A.........T.Name++++++RLen++TDpB......Functions++++++++++++++++++++++++++
 *                                Jednoznačný klíč (zákaz duplicit)
A                                     UNIQUE
 *                                Referenční soubor definic polí
A                                     REF(REFMZP)
 *   Jméno věty (formátu, záznamuu)
A          R CENIKPF0
 *   Jména datových polí (s odkazem na referenční soubor)
A            MATER     R
A            CENA      R
A            NAZEV     R
 *   Definice klíče
A          K MATER
```

```
CRTPF FILE(*CURLIB/CENIKP) SRCFILE(*LIBL/QDDSSRC) SRCMBR(*FILE)
```

```
 ************************************************************
 *   Soubor STAVYP - Stavy materiálových zásob)               *
 ************************************************************
A.........T.Name++++++RLen++TDpB......Functions++++++++++++++++++++++++++
 *                                Jednoznačný klíč (zákaz duplicit)
A                                     UNIQUE
 *                                Referenční soubor definic polí
A                                     REF(REFMZP)
A          R STAVYPF0
A            ZAVOD     R
A            SKLAD     R
A            MATER     R
A            MNOZ      R
 *   Definice klíče (má tři složky)
A          K ZAVOD
A          K SKLAD
A          K MATER
```

```
CRTPF FILE(*CURLIB/STAVYP) SRCFILE(*LIBL/QDDSSRC) SRCMBR(*FILE)
```

```
 ************************************************************
 *   Soubor OBRATP - Obraty materiálu - klíč není jednoznačný *
 ************************************************************
A*.........T.Name++++++RLen++TDpB......Functions++++++++++++++++++++++++++
 *                                Referenční soubor definic polí
A                                     REF(REFMZP)
A          R OBRATPF0
A            ZAVOD     R
A            SKLAD     R
A            MATER     R
A            MNOBR     R
A          K ZAVOD
A          K SKLAD
A          K MATER
```

```
CRTPF FILE(*CURLIB/OBRATP) SRCFILE(*LIBL/QDDSSRC) SRCMBR(*FILE)
```

Popis a vytvoření logického souboru

```
 *****************************************************************
 *   Logický soubor STAVYL - Stavy materiálových zásob.          *
 *   Setříděno podle materiálu (klíč je číslo materiálu).        *
 *****************************************************************
 *   Jmeno věty (formátu, záznamu) převzaté z fyzického souboru
A*.........T.Name++++++RLen++TDpB......Functions++++++++++++++++++++++++++
A          R STAVYPF0
A                                     PFILE(STAVYP) FORMAT(STAVYP)
 *   Definice klíče (povinná) - Jiné setřídění
A          K MATER
A          K ZAVOD
A          K SKLAD
```

```
CRTLF FILE(*CURLIB/STAVYL) SRCFILE(*LIBL/QDDSSRC) SRCMBR(*FILE)
```

Popis a vytvoření spojeného logického souboru (joined logical file)

```
 *****************************************************************
 *   Spojený logický soubor STAVYJ - Stavy materiálových zásob.  *
 *   Záznam je tvořen ze dvou fyzických souborů - CENIKP a STAVYP.*
 *   Pojícím polem (join field) je číslo materiálu MATER.        *
 *****************************************************************
A*.........T.Name++++++RLen++TDpB......Functions++++++++++++++++++++++++++
 *   Jméno záznamu (format, record)
A          R STAVYJF0
A                                     JFILE(STAVYP CENIKP)
A          J                          JOIN(STAVYP CENIKP)
A                                     JFLD(MATER MATER)
A            ZAVOD
A            SKLAD
A            MATER                    JREF(STAVYP)
A            NAZEV
A            MNOZ
A            CENAJ
 *   Definice klíče (povinná) - Jiné setřídění
A          K MATER
A          K ZAVOD
A          K SKLAD
```

```
CRTLF FILE(*CURLIB/STAVYJ) SRCFILE(*LIBL/QDDSSRC) SRCMBR(*FILE)
```

Vytvoření SQL tabulek

Databázové tabulky SQL (Structured Query Language) jsou v systému realizovány jako fyzické databázové soubory PF-DTA. V SQL se používá jiné názvosloví:

OS/400	SQL
soubor – file	tabulka – table
záznam (věta) – record	řádek – row
pole – field	sloupec – column

Příklady vytvoření tabulek a phledů odpovídají shora popsaným souborům. V SQL neexistuje pojem referenčního souboru, sloupce stejného jména musí tedy v každé tabulce být znovu definovány. Předpokládá se, že tabulky a pohledy (views) budou umístěny v běžné knihovně *CURLIB (current library). Pohledy (views) nelze setřídit podle žádného klíče, nejsou tedy rovnocenné s logickými soubory.

```
CREATE OR REPLACE TABLE CENIKP
     ( MATER  CHAR(5)    UNIQUE,  -- jednoznačný klíč
       CENAJ  DEC(10, 2),
       NAZEV  CHAR(50)
     )
     RCDFMT CENIKPF0


CREATE OR REPLACE TABLE STAVYP
     ( ZAVOD  CHAR(2),
       SKLAD  CHAR(2),
       MATER  CHAR(5),
       MNOZ   DEC(10, 2)
       UNIQUE (ZAVOD, SKLAD, MATER)  -- jednoznačný klíč
     )
     RCDFMT STAVYPF0


CREATE OR REPLACE TABLE OBRATP  -- má cizí klíč pro tabulku STAVYP
     ( ZAVOD  CHAR(2),
       SKLAD  CHAR(2),
       MATER  CHAR(5),
       MNOBR  DEC(10, 2),
       FOREIGN KEY (ZAVOD, SKLAD, MATER) REFERENCES STAVYP (ZAVOD, SKLAD, MATER)
     )
     RCDFMT OBRATPF0


CREATE VIEW STAVYL AS
       SELECT * FROM  STAVYP  -- pohled není setříděn


CREATE VIEW STAVYJ AS         -- pohled není setříděn
       SELECT ZAVOD, SKLAD, S.MATER, NAZEV, MNOZ, CENAJ
       FROM STAVYP AS S
       JOIN CENIKP AS C ON S.MATER = C.MATER
```

Zpracování databázových souborů

Databázové soubory se zpracovávají zejména těmito způsoby

- Aplikačními programy v jazycích RPG, COBOL atd.
- CL programy (jen velmi omezeně),
- CL příkazem CPYF (Copy File),
- CL příkazem STRQRY (Start Query) - program Query/400,
- CL příkazem STRSQL (Start SQL) - program SQL/400,
- CL příkazem EDTF (Edit File).

Query/400

Query/400 je program určený uživatelům neprogramátorům k rychlému sestavení a provedení dotazu na obsah jednoho nebo více databázových souborů.

Query se vyvolává příkazem

STRQRY Start Query

Po volbě 1 volíme opět volbu 1 a pojmenujeme dotaz. Pak dostaneme nabídku k zadání podrobností dotazu.

```
Type options, press Enter.  Press F21 to select all.
  1=Select

Opt    Query Definition Option
 1     Specify file selections
 _     Define result fields
 _     Select and sequence fields
 _     Select records
 _     Select sort fields
 _     Select collating sequence
 _     Specify report column formatting
 _     Select report summary functions
 _     Define report breaks
 _     Select output type and output form
 _     Specify processing options
```

SQL/400

Structure Query Language (SQL) je všeobecně rozšířený jazyk určený k definování databázových objektů (tabulek, pohledů, indexů aj.), jejich modifikaci, manipulaci a dotazování. Používá pojmy Statement (příkaz) a clause (fráze - část příkazu).

SQL se vyvolává příkazem

STRSQL Start SQL

Hlavní příkazy jsou:

CREATE TABLE	vytvoření tabulky,
INSERT	vložení řádku do tabulky,
UPDATE	přepsání sloupců v tabulce,
DELETE	vymazání řádků z tabulky,
SELECT	výběr řádků z tabulek k sestavení dotazu.

Data File Utility (DFU)

Vyvolává se příkazem:

STRDFU Start DFU

Lze vybrat jednu z pěti voleb:

1. Spustit DFU-program
2. Vytvořit DFU-program
3. Změnit DFU-program
4. Smazat DFU-program
5. Změnit data dočasně vytvořeným programem (v PDM též volba 18 u databázového souboru)

Poslední volba 5 se používá nejčastěji k operativním změnám dat.

Může se použít i příkaz **UPDDTA**, který se chová jako volba 5 uvedená shora:

```
UPDDTA FILE(CENY)
```

```
WORK WITH DATA IN A FILE                    Mode . . . . :   ENTRY
Format . . . . :   CENYR_____               File . . . . :   CENY

Cislo mater.:______
Cena/j.:     ___________
Název zboži: ____________________________
```

Data File Utility (DFU) je prostředek vhodný k pořizování a modifikaci dat v *testovacích* databázových souborech. Nedoporučuje se k opravám provozních dat, protože provozní soubory jsou závislé jedna na druhé a izolované změny v jednotlivých souborech by mohly narušit vztahy mezi nimi. Proto je důležité změny v databázových souborech provádět výhradně aplikačními programy k tomu určenými.

Edit File (EDTF)

Edituje proudový soubor (stream file) nebo databázový soubor bez externího popisu.

```
EDTF STMF('/home/vzupka/python/zaklady/retezce.py')
```

```
Edit File: /home/vzupka/python/zaklady/retezce.py
Record :       1   of     225 by  10                    Column :    1     59 by 126
Control :

CMD ....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8....
     ************Beginning of data**************
    text = "abcd"   # uvozovky (dvojité)
    print(text)
    text = 'abcd'   # uvozovky jednoduché (aporetofy)
    ...
```

CL příkaz CPYF pro kopírování souborů

```
CPYF  FROMFILE(SOUBOR1)
      TOFILE(SOUBOR2)
      MBROPT(*ADD | *REPLACE)
      FMTOPT(*NOCHK | *MAP | *DROP)
      ...
```

Integrovaný systém souborů (IFS)

https://www.ibm.com/docs/en/i/7.3?topic=systems-integrated-file-system

Otevřenost vůči jiným operačním systémům

Integrovaný systém souborů (IFS - Integrated File System) představuje mechanismus pro spolupráci s jinými operačními systémy. Byl do systému AS/400 (IBM i) zaveden dodatečně jako nadstavba.

ISF usnadňuje použití systému *IBM i* jako serveru. Dovoluje totiž uchovávat a zpracovávat soubory pocházející z různých počítačů a operačních systémů, zejména UNIX, Windows, OS/2 a DOS. Struktura těchto souborů je známá vždy jen té aplikaci (aplikačnímu programu), která je vytvořila nebo je umí zpracovat, ne však operačnímu systému. Takové soubory mohou obsahovat nejrůznější údaje, jako jsou záznamy textů, obrazů, zvuků, filmů apod. Nazývají se *proudové soubory (stream files)* a jsou to objekty typu ***STMF**. Organizují se v *adresářích (directories)*, což jsou objekty typu ***DIR**.

Systém ISF také udržuje datové struktury pro otevřené proudové soubory a tzv. sokety (sockets), tedy struktury potřebné ke komunikaci v prostředí IP (internet protocol).

Stručně o IFS

Soubory v ISF jsou umístěny na koncích (listech) hierarchické (stromové) struktury. Nadřízené uzly takové struktury se nazývají *adresáře (directories)* nebo *složky (folders)*. Adresářová struktura je běžným mechanismem k organizaci souborů v různých operačních systémech. V IFS se tomu mechanismu přizpůsobuje i schéma knihoven a objektů, které je obvyklé v *IBM i*.

V systému IFS jsou zahrnuty tyto adresářové struktury:

/	"root" – hlavní (kořenový) adresář označený lomítkem
QOpenSys	adresář prostředí UNIX
QNTC	adresář prostředí Windows NT
QSYS.LIB	adresář prostředí *IBM i*
QFileSvr.400	adresář pro přístup k IFS na vzdáleném systému AS/400 (IBM i)
UDFS	adresář pro souborový systém definovaný uživatelem
NFS	adresář pro síťový souborový systém (Network File System)
a další.	

V hierarchické struktuře adresářů se zadává *jméno cesty (path name)*, což jsou jména adresářů vedoucích k souboru končící jménem souboru, oddělená lomítky a obklopená apostrofy. Pro pohodlí uživatelů PC lze v CL příkazech místo lomítka použít obrácené lomítko \. Jména cest se používají v příkazech pracujících se soubory.

```
'/Adr1/Adr2/Adr3/Soubor'
'\Adr1\Adr2\Adr3\Soubor'
```

V adresáři QSYS.LIB by jméno cesty vypadalo např. takto:

```
'/QSYS.LIB/UCETNI.LIB/POHLEDAVKY.FILE'
```

Zde je před tečkou název objektu a za tečkou označení jeho typu. `QSYS`, `UCETNI` a `POHLEDAVKY` jsou jména objektů (knihovny jsou objekty typu *LIB). `LIB` a `FILE` jsou označení typu. Jde vlastně o poněkud zobecněné schéma knihoven OS/400, kde by stačilo zadat `UCETNI/POHLEDAVKY`. Typy se ve jméně cesty musí zadávat, protože v knihovně může být několik objektů stejného jména (ovšem různéno typu).

IFS má velký význam při výměně dat mezi různými operačními systémy. Proto existují CL příkazy pro přesuny dat mezi různými adresáři a soubory.

Několik příkazů pro IFS:

WRKLNK	Work with object links - Umožní pracovat s objekty v adresářích. Příkaz WRKLNK OBJ(/) zobrazí kořenový (root) adresář s volbami.
DSPLNK	Display object links - Umožní zobrazovat objekty v adresářích.
CRTDIR	též **MD** nebo **MKDIR** - Create directory
RMVDIR	též **RD** nebo **RMDIR** - Remove directory
CHGCURDIR	též **CD** nebo **CHDIR** - Change current directory
CPY	též **COPY** - Kopíruje objekt nebo skupinu objektů
MOV	též **MOVE** - Přesune objekt do jiného adresáře
CPYFRMIMPF	Copy from import file - Kopíruje vybraná data z databázového nebo proudového souboru se strukturou do jiného databázového souboru
CPYTOIMPF	Copy to import file - Kopíruje databázový soubor do jiného databázového nebo proudového souboru se strukturou
CPYFRMSTMF	Copy from stream file - Kopíruje proudový textový soubor do databázového souboru
CPYTOSTMF	Copy to stream file - Kopíruje databázový textový soubor do proudového souboru
CHGOWN	Change owner - Změní vlastníka objektu
CHGATR	Change attribute - Změnní zvolený atribut objektu, např. *CCSID.
CHGAUT	Change authority - Změní oprávnění k objektu nebo skupině objektů pro uživatele a pro autorizační seznam. Práva k údajům objektu mají kódy *RWX *RX *RW *WX *R *W *X *NONE *EXCLUDE *AUTL
WRKAUT	Work with authority - Práce s oprávněním k objektu nebo skupině objektů pro uživatele a pro autorizační seznam
EDTF	Edit stream file - Editace IFS souboru (ale i databázového souboru bez externího popisu DDS, např. QCLSRC)

Alternativní jména příkazů (MD atd.) jsou zavedena pro pohodlí uživatelů jiných operačních systémů.

Jištění dat

Zálohování (backup)

Všechny objekty by měly být pravidelně ukládány na vnější paměťové médium - pásku, aby nebyl ohrožen provoz po případné havárii počítače. Jde zejména o databázové soubory, které představují nejcennější majetek firmy. K tomu slouží speciální příkazy operačního systému začínající zkratkou slovesa save (zachránit) - SAV.

Příkazy k ukládání objektů:

SAVLIB	Save libraries - uložit knihovny
SAVOBJ	Save objects - uložit objekty
SAVCHGOBJ	Save changed objects - uložit změněné objekty
SAVDLO	Save document library objects - uložit dokumenty nebo pořadače
SAVSYS	Save system - uložit operační systém
SAVSTG	Save storage - uložit celou diskovou paměť
SAVLICPGM	Save licensed programs - uložit licenční programy
SAVSECDTA	Save security data - uložit zabezpečovací data (profily a autorizační seznamy)
SAVCFG	Save configuration objects - uložit konfigurační objekty
SAV	Save - uložit objekty IFS (adresáře a proudové soubory)

Tyto příkazy ukládají příslušné objekty na pásku nebo do speciálního diskového souboru- tzv. zálohovacího souboru (save file), což je objekt typu *SAVF. Odtud lze objekty buď uložit na pásku příkazem

SAVSAVFDTA	Save save-file data - uložit data ze zálohovacího souboru.

Obnova (recovery)

K obnově uložených objektů dochází řidčeji než k jejich ukládání, zvláště po havárii počítače nebo programů. Někdy je také nutné přenést objekty na jiný počítač. K této práci slouží příkazy začínající zkratkou slovesa *restore* (obnovit) - RST.

Příkazy k obnově objektů:

RSTLIB	Restore libraries - obnovit knihovny
RSTOBJ	Restore objects - obnovit objekty
RSTDLO	Restore document library objects - obnovit dokumenty nebo pořadače
RSTCAL	Restore callendars - obnovit kalendáře
RSTLICPGM	Restore licensed programs - obnovit licenční programy
RSTAUT	Restore authority - obnovit autorizace
RSTUSRPRF	Restore user profiles - obnovit uživatelské profily
RSTCFG	Restore configuration objects - obnovit konfigurační objekty
RST	Restore - obnovit objekty IFS (adresáře a proudové soubory)

Obnovovací příkazy slučitelné s ukládacími příkazy:

SAVOBJ	RSTOBJ
SAVLIB	RSTOBJ
	RSTLIB
SAVCHGOBJ	RSTOBJ
SAVSYS	RSTOBJ
	RSTUSRPRF, RSTAUT
	RSTCFG
SAVSAVFDTA	RSTOBJ
	RSTLIB
	RSTUSRPRF, RSTAUT
	RSTCFG
SAVLICPGM	RSTLICPGM
SAVDLO	RSTDLO
SAVSECDTA	RSTUSRPRF, RSTAUT
SAVCFG	RSTCFG
SAV	RST

Prostředky usnadňující zálohování a obnovu

Použití nabídek SETUPBCKUP a BACKUP usnadňuje použití zálohovacích příkazů. Příkaz

GO SETUPBCKUP

umožňuje plánovat doby pravidelného zálohování (denní, týdenní, měsíční) a volit sestavu ukládaných objektů. Příslušné ukládací příkazy se pak vytvářejí automaticky. Rovněž spuštění příslušných úloh se naplánuje podle naplánované volby. Příkaz

GO BACKUP

umožňuje navíc specifikovat a okamžitě provést zálohovací úlohu, zjišťovat stav zálohování a inicializovat pásky nebo jejich skupiny. Velkou pomocí je také výtisk protokolu o výsledku zálohování.

K usnadnění obnovování objektů slouží příkaz

GO RESTORE

umožňující pohodlnější volbu obnovovacích příkazů na základě hierarchických nabídek.

BRMS

Ještě výkonnějším prostředkem k automatizaci zálohovacích a obnovovacích prací je program **BRMS** - Backup Restore and Media Services, který zcela automatizuje práce spojené se zálohováním, evidencí nosičů. Dovoluje navíc také použít automatizovaných páskových knihoven (automated tape libraries) - robotů, kdy se příslušný nosič (kazeta) vyhledá v knihovně a založí automaticky do páskové jednotky. Po provedení zálohy či obnovy se opět automaticky vyjme a vrátí zpět na své místo v páskové knihovně.

Žurnálování databázových souborů

Databázové soubory, jako nosiče životně důležitých údajů, lze zajišťovat ještě důkladněji než pořizováním záložních kopií. Jde o tzv. žurnálování (journaling). Použijeme-li žurnálování, můžeme za provozu zachycovat veškeré změny v těch databázových souborech, které si určíme. Dojde-li k havárii, můžeme data obnovit až k poslední zachycené změně.

K žurnálování jsou zapotřebí dva typy objektů:

*JRNRCV (journal receiver)	přijímač žurnálových dat
*JRN (journal)	žurnál

Journal receiver je přijímač žurnálových dat. Obsahuje informace o změnách ve sledovaných souborech. Journal je objekt, který určuje sledované soubory a přístupové cesty. Zaznamenávají se v něm také informace o přijímačích a sledovaných souborech. K několika žurnálovým přijímačům se vytváří jeden žurnál.

Příkazy pro žurnálování:

CRTJRNRCV	Create journal receiver
CRTJRN	Create journal
WRKJRN	Work with journal
WRKJRNA	Work with journal attributes (seznam přijímačů aj.)
DSPJRN	Display journal (zobrazí záznamy z přijímačů)
STRJRNPF	Start journaling of physical file (zahájí žurnálování fyzického souboru)
STRJRNAP	Start journaling of access path (fyzického nebo logického souboru)
ENDJRNPF	End journaling of physical file
ENDJRNAP	End journaling of access path
APYJRNCHG	Apply journaled changes (obnova databáze z žurnálu)

Přijímače je třeba vytvořit dříve než žurnál. Příkazy STRJRN... a ENDJRN... se vyvolávají zpravidla z CL programů, které obklopují aplikační programy zpracovávající příslušné databázové soubory.

Zabezpečení transakcí (commitment control)

Transakcí se rozumí jedna nebo několik změnových operací (přidání, zrušení, přepis záznamu) s jedním nebo několika databázovými soubory v aplikačních programech. Je-li transakce komplikovaná a trvá-li déle, zejména při interakčním zpracování, hrozí nebezpečí, že při havárii se transakce nedokončí. Všechny změny v zúčastněných databázových souborech se sice zaznamenají do žurnálových přijímačů, ale nebude zřejmé, zda se celá transakce dokončila nebo ne. Tomu se dá zabránit použitím speciálních příkazů *v aplikačních programech*. Tyto příkazy tvoří doplněk k obyčejnému žurnálování:

	RPG IV	**COBOL**	**CL**
Commit (svěřit, potvrdit)	COMMIT	COMMIT	COMMIT
Rollback (odvolat)	ROLBK	ROLLBACK	ROLLBACK

Příkaz Commit potvrzuje platnost transakce, příkaz Rollback odvolává celou transakci. Transakce je definována aplikačním programem, a to tím, jak provádí příkazy Commit. Příkazem OPEN je zahájena první transakce, příkazem Commit končí. Další transakce trvá až do dalšího příkazu Commit. Příkaz Rollback se používá ve výjimečných situacích, kdy aplikační program rozhodne, že je nutné transakci odvolat, protože nemůže být z nějakého důvodu dokončena. Při havárii programu operační systém sám odvolá poslední rozpracovanou transakci. Příkazy Commit a Rollback zaznamenávají, popř. ruší informace v žurnálovém přijímači.

Příkazy CL pro zahájení a ukončení transakčních příkazů

Použití příkazů Commit a Rollback je nutné zahájit a ukončit následujícími příkay:

STRCMTCTL	Start commitment control
ENDCMTCTL	End commitment control

Diskové oblasti (auxiliary storage pools)

Při žurnálování se doporučuje umístit přijímače a žurnály do jiné diskové oblasti než databázové soubory, které jsou jimi sledovány. Zlepší se tím výkonnost počítače. Diskové oblasti se označují ASP (Auxiliary Storage Pools). První ASP je určena systému (ASP 1), ostatní (ASP 2 až 16) si může definovat uživatel. Definice uživatelských ASP se provádí při *manuálním startu* počítače použitím volby "Use Dedicated Service Tools (DST)". Přehled o diskových oblastech lze získat také pomocí příkazu STRSST - Start system service tools.

Číslo uživatelské diskové oblasti se zadává v příkazu CRTLIB (vytvoření knihovny) parametrem ASP. Jméno takto vytvořené knihovny se zadává v příkazech CRTJRNRCV, CRTJRN a CRTSAVF.

Kromě systémového a uživatelských ASP lze vytvořit nezávislé diskové oblasti (IASP - independent auxiliary storage pool) s použitím programu *IBM Navigator for i*. IASP je skupina diskových jednotek, které lze vyřadit off-line a zařadit on-line nezávisle na zbylé systémové paměti včetně systémového ASP, uživatelských ASP a ostatních IASP. Nezávislé ASP jsou užitečné jak v samostatném systému, tak v prostředí více systémů. K identifikaci IASP systém generuje a používá jedinečné číslo od 33 do 99.

Žurnál pro audit

Žurnál pro audit (Security audit journal) je součást operačního systému a umožňuje provádět revizi (audit) všech procesů v systému. Lze definovat tři úrovně auditu:

přes celý systém pro všechny uživatele
pro určité objekty
pro určité uživatele

Systémové hodnoty QAUDCTL (audit control), QAUDLVL (audit level) a QAUDLVL2 (audit level extension) společně s parametrem AUDLVL (action auditing) v uživatelských profilech zajišťují kontrolování činností.

Příkaz CHGSECAUD umožňuje snadno nastavit tyto systémové hodnoty.
Příkaz DSPSECAUD zobrazí tyto systémové hodnoty.

Žurnál pro audit se nastavuje v několika krocích následujícími příkazy.

CRTJRNRCV	Vytvořit žurnálový přijímač v knihovně pravidelně zálohované
CRTJRN	Vytvořit žurnál
WRKSYSVAL SYSVAL(*SEC)	
	Nastavit úroveň auditu v systémových hodnotách QAUDLVL, QAUDLVL2
CHGOBJAUD	Změnit audit určitého knihovního objektu
CHGAUD	Změnit audit u objektu nebo skupiny objektů IFS
CHGUSRAUD	Změnit audit pro určité uživatele

Žurnál lze analyzovat příkazem DSPJRN (Display Journal).

Podpora programování

Programming Development Manager (PDM)

Programming Development Manager (PDM) je podpůrný prostředek pro vývoj programů. Ulehčuje práci s knihovnami, objekty a členy zdrojových souborů. Členy zdrojových souborů jsou vlastně texty programů, popisů dat aj., které programátor napíše a pak je překládá (kompiluje). Každý překlad (kompilace) je vlastně produktem příkazu (CRT...), který vytváří nový objekt. Vytvořený objekt bývá nejčastěji typu *PGM (program) nebo *FILE (soubor). PDM umožňuje snadno vyvolávat další podpůrné prostředky:

programátorský editor (SEU - Source Entry Utility),
program pro návrh obrazovky (SDA - Screen Design Aid),
program pro návrh tiskového výstupu (RLU - Report Layout Utility).

Příkazy pro práci s PDM:

STRPDM	Start PDM
WRKLIBPDM	Work with Libraries using PDM
WRKOBJPDM	Work with Objects using PDM
WRKMBRPDM	Work with Members using PDM

Source Entry Utility (SEU)

Source Entry Utility (SEU) je prostředek k pořizování zdrojových textů, ať již pro programy v jazycích CL, RPG, COBOL atd., nebo pro popisy dat (DDS - Data Descrption Specifications). Zdrojové texty jsou členy (members) zdrojových souborů. Zdrojový soubor je speciální typ databázového fyzického souboru s množstvím členů. Vyvolává se příkazem

STRSEU Start SEU

Zdrojový text je tvořen záznamy (řádky) o 92 znacích, z nichž prvních 12 je určeno pro datum a číslování, a dalších 80 znaků je vyhrazeno pro text příkazu, komentáře apod. Lze však zvolit i jinou délku záznamu. Editor SEU je specializován na maniulaci s řádky programových a specifikačních textů, a přitom je citlivý na typ zdrojového textu. Tak např. pro zdrojový text typu CLP dokáže kontrolovat formální správnost zápisu příkazů CL, pro zdrojový text typu RPG kontroluje správnost příkazů jazyka RPG apod. Vždy však kontroluje jen jeden příkaz izolovaně; nemůže tedy kontrolovat souvislosti mezi různými příkazy.

Zdrojový soubor se vytváří příkazem

CRTSRCPF Create Source Physical File

Různé typy zdrojových členů se zpravidla zapisují do různých zdrojových souborů, i když je lze uložit třeba jen do jednoho. Doporučená jména nejčastějších zdrojových souborů jsou tato:

QCLSRC	pro CL a ILE/CL programy
QRPGSRC	pro RPG III programy
QRPGLESRC	pro ILE/RPG (RPG IV) programy
QCBLSRC	pro COBOL programy
QCBLLESRC	pro ILE/COBOL programy
QDDSSRC	pro specifikace popisů dat DDS

Z těchto jmen editor SEU pozná, o jaký typ zdrojového textu jde, tj. zda text je zapsán v jazyku CL, RPG, COBOL, DDS apod. Podle toho také provádí kontrolu správného zápisu příkazů.

Příklady nejpoužívanějších příkazů pro manipulaci s textem:

I	Insert - vložit prázdný řádek (za)
I5	Insert - vložit 5 prázdných řádků
RP	Repeat - zopakovat řádek
RP5	Repeat - zopakovat řádek 5krát
D	Delete - zrušit řádek
D5	Delete - zrušit 5 řádků
C	Copy - kopírovat řádek (cíl se určí příkazem A nebo B)
C5	Copy - kopírovat 5 řádků
CC	Copy - kopírovat blok řádků (CC se napíše u prvního a posledního řádku)
M	Move - přesunout řádek (cíl se určí příkazem A nebo B)
M5	Move - přesunout 5 řádků
MM	Move - přesunout blok řádků (MM se napíše u prvního a posledního řádku)
A	After - cíl kopie nebo přesunu je za označeným řádkem
B	Before - cíl kopie nebo přesunu je před označeným řádkem
Fxx	Format - zobrazit formát (pravítko) pro zdrojový příkaz (zvl. DDS, RPG)
F?	Dotaz na existující formáty
aj.	

Tyto příkazy (zkratky) se píší do sloupce obsahujícího pořadová čísla (na začátku řádků).

Poznámka: Zdrojový text může nebo musí někdy být delší než 92 znaků. Například pro jazyk RPGIV nejméně 112 znaků, pro jazyk C bývá i delší.

Screen Design Aid (SDA)

Screen Design Aid (SDA) je prostředek k návrhu, vytváření a údržbě obrazovkových formátů včetně nabídek (menus). Programátor navrhuje nebo mění tvar obrazovkového formátu ve skutečné podobě přímo na obrazovce. Vyvolává se příkazem

STRSDA Start SDA

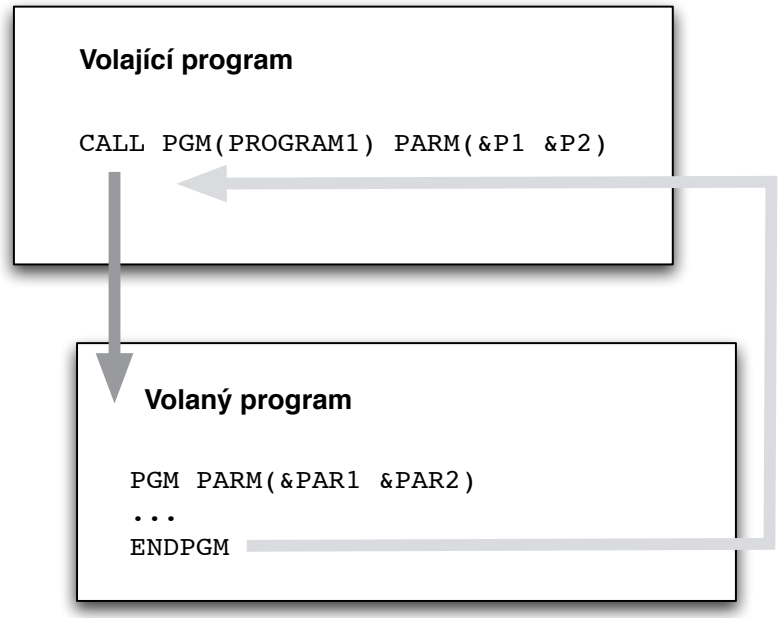
Po několika odpovědích na výzvy (zadání jména souboru, jmen formátů apod.) se objeví volná plocha, na niž lze zapisovat textové konstanty, vytvářet datová pole (vstupní, výstupní, obousměrná), odvozovat vlastnosti polí z definic databázových souborů, přemisťovat údaje, přidávat k údajům zobrazovací atributy, barvy, rámečky apod. Po dokončení návrhu se vytvoří, popř. opraví zdrojový text v jazyku DDS, a vytvoří se i příslušný objekt typu *FILE, podtypu DSPF. Tento objekt je potřebný při překladu programu používajícího tento obrazovkový soubor.

Komunikace mezi programy

Komunikace v rámci úlohy

Parametry volání

Programy v rámci jedné úlohy, ať už interakční nebo dávkové, se vzájemně vyvolávají pomocí příkazů CALL a RETURN. Tyto příkazy umožňují předávat data pomocí *seznamu parametrů*.. V jednotlivých programovacích jazycích mají příkazy CALL a RETURN různou podobu. V jazyku CL je schema následující:



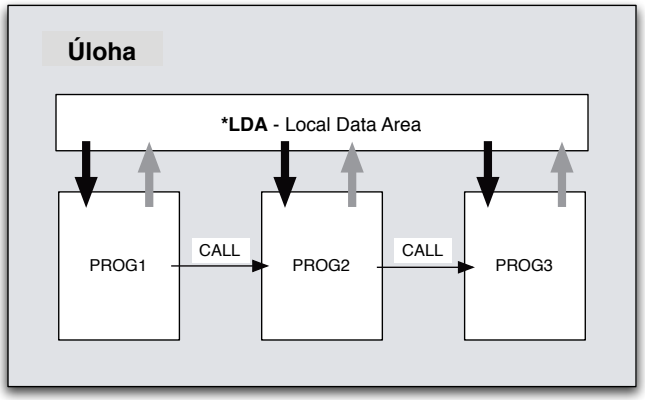
Oba parametry v seznamu parametrů musí mít stejný typ a stejnou délku. Změna hodnoty proměnné &PAR1 ve volaném programu způsobí změnu hodnoty proměnné &P1 ve volajícím programu (podobně &PAR2 a &P2).

Lokální datová oblast

Lokální datová oblast je zvláštní paměť vyhrazená úloze. S úlohou vzniká a zaniká. Programy si v ní mohou předávat údaje místo parametrů nebo zároveň s nimi. K zápisu a čtení dat slouží v různých jazycích různé příkazy. V jazyku CL jsou to příkazy

CHGDTAARA	DTAARA(*LDA)	Change data area
RTVDTAARA	DTAARA(*LDA)	Retrieve data area

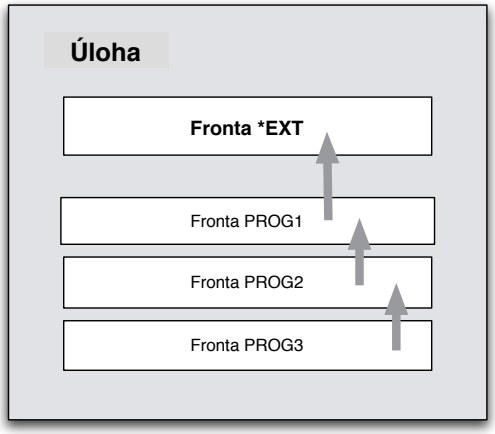
V následujícím schematu jsou znázorněny tři programy, PROG1, PROG2 a PROG3, které se postupně volají příkazem CALL a každý používá oblast *LDA (local data area).



Programové fronty zpráv

Programové fronty zpráv (objekty typu *MSGQ) se vytvářejí automaticky pro každý program. Programy i operační systém je používají zejména pro sdělování informací o chybách a mimořádných stavech. V následujícím obrázku jsou naznačeny programy, které jsou volány příkazem CALL a předávají si zprávy prostřednictvím svých programových front.

V následujícím obrázku jsou naznačeny programové fronty PROG1, PROG2 a PROG3 odpovídající programům stejného jména a externí fronta *EXT odpovídající úloze. Program PROG3 vyšle zprávu programu PROG2, ten zase programu PROG1, a ten vyšle zprávu do fronty *EXT. Všechny tyto zprávy jsou posílány CL příkazem SNDPGMMSG nebo systémovým voláním (API - Application Program Interface).



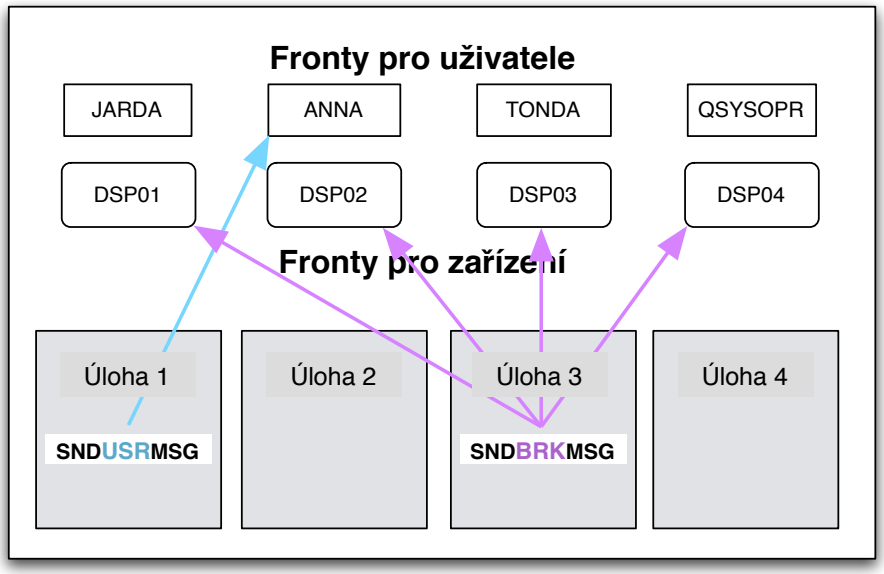
Uživatelské fronty zpráv

Fronty zpráv existují pro každého uživatele a pro každou pracovní stanici. Těchto front využívají programy spíše pro jednostrannou komunikaci z programu k uživateli (speciálně systémovému operátorovi - QSYSOPR). K tomu využívají příkazy SND...MSG.

Komunikace mezi úlohami

Fronty zpráv

Mezi několika různými úlohami *interakčního* typu lze ke komunikaci využít uživatelské fronty zpráv přiřazené každému uživateli, tj vlastně interakční úloze, nebo fronty přiřazené každé pracovní stanici (také spojené s určitou interakční úlohou, jestliže je na ní právě provozována). Úlohy si tak mohou předávat informace pomocí CL příkazů SND...MSG. Tímto způsobem je výhodné předávat jen krátká sdělení popř. odpovědi.



Databázové soubory

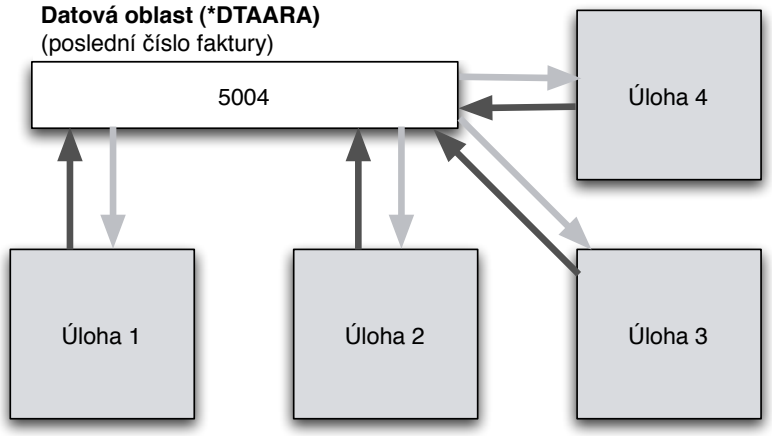
Databázové soubory jsou univerzálním prostředkem komunikace mezi programy, ať už běží v interakčních nebo dávkových úlohách. V nich se zpravidla předávají velké objemy dat. Nejsou vhodné ke sdělování krátkých zpráv, protože spotřebovávají značný objem prostředků (zdrojů) počítače.

Uživatelské datové oblasti (data areas)

Datová oblast je objekt typu ***DTAARA** a představuje menší objem dat v trvalé paměti. Můžeme si ji představit jako jeden izolovaný databázový záznam. Datovou oblast lze vytvořit, měnit a číst následujícími příkazy CL:

CRTDTAARA	Create data area
CHGDTAARA	Change data area
RTVDTAARA	Retrieve data area

Rozdíl proti lokálním datovým oblastem je ten, že uživatelské datové oblasti jsou objekty, a tedy existují, dokud nejsou zrušeny příkazem DLTDTAARA. Příkazy ke čtení a změně datových oblastí v ostatních programovacích jazycích jsou různé. Datové oblasti jsou výhodné při komunikaci mezi úlohami (dávkovými i interakčními), předává-li se jen malý objem dat, který se v čase mění (např. poslední přidělené číslo faktury).



Datové fronty

Datové fronty jsou objekty typu *DTAQ a slouží k rychlému předávání zpráv či menších kousků dat tak, aby předávající program nemusel čekat na odpověď a přijímající program nemusel zprávu ihned převzít (přečíst). Zprávy lze z fronty vybírat podle pořadí, jak byly doručeny (FIFO - First-In-First-Out), nebo v obráceném pořadí (LIFO - Last-In-First-Out), anebo podle klíče. Datové fronty se vytvářejí příkazem

CRTDTAQ	Create data queue

Zprávy se do datové fronty posílají pomocí systémového volání (API - Application Program Interface):

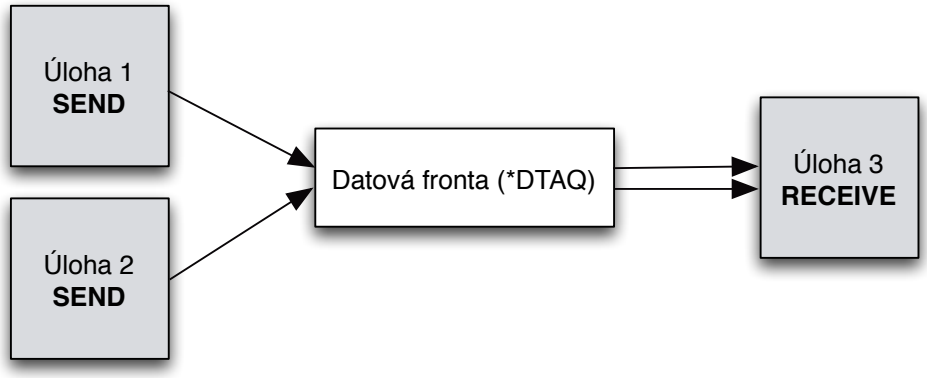
CALL QSNDDTAQ s příslušnými parametry	Send to data queue

Zprávy se přijímají z datové fronty voláním:

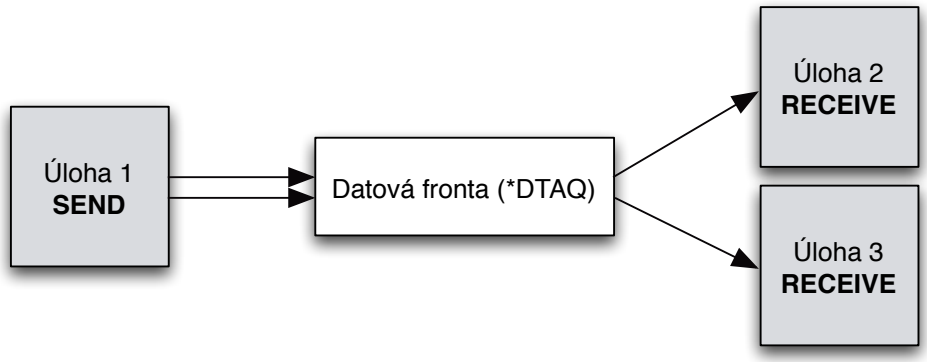
CALL QRCVDTAQ s příslušnými parametry	Receive from data queue

Datové fronty představují nejlepší prostředek ke komunikaci mezi dávkovými úlohami a k jejich synchronizaci.

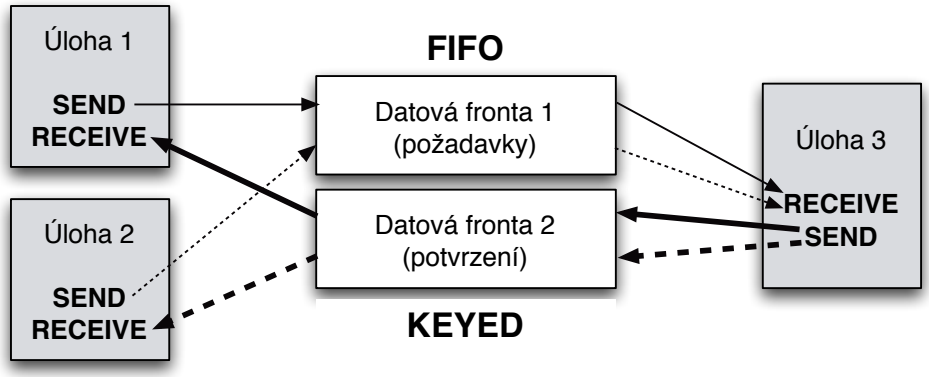
Následující schema ukazuje dvě úlohy 1 a 2 posílající požadavky do fronty (např. jména nebo čísla tiskových programů) a služební úlohu 3 plnící tyto požadavky. Služební úloha nepotvrzuje splnění úkolu.



Další schema ukazuje úlohu 1, která posílá požadavky do fronty, z níž si je v příhodných okamžicích odebírají dvě služební úlohy 2 a 3. Každý požadavek je zpracován tou úlohou, která si jej dříve přečte. Po přečtení již zpráva s požadavkem není ve frontě k dispozici.



Další schema ukazuje dvě úlohy, 1 a 2, které posílají požadavky do požadavkové fronty a přijímají potvrzení o splnění úkolu z potvrzovací fronty. Úloha 3 je služební a čte všechny požadavky postupně z požadavkové fronty, každý požadavek splní a pošle o tom hlášení (potvrzení) do potvrzovací fronty. Součástí potvrzovací zprávy musí být nějaké znamení, které určuje úlohu, které je určena. To je nejčastěji číslo úlohy, které je již součástí požadavkové zprávy. Zatímco požadavky se do první fronty řadí v časovém sledu (FIFO), jsou zprávy v druhé frontě uspořádány podle klíče, např. čísla úlohy.



Komunikace mezi počítači

Komunikační prostředky jsou integrální součástí systému IBM i. Aplikační programy pracují s komunikačními funkcemi podobně jako s lokálními zařízeními a soubory. Existují programy (dodávané s operačním systémem nebo jako licenční programy), které plní většinu komunikačních úloh. Každý z nich je specializován na určitý druh komunikace (např. DDM dovoluje přenášet data na úrovni databázových záznamů).

Hlavní komunikační metody

- **TCP/IP** - Transmission Control Protocol/Internet Protocol je světově rozšířená komunikační metoda spojující počítače nejrůznějších druhů, zejména těch, které provozují operační systémy typu UNIX. Zejména je známá z použití v síti Internet. U počítače AS/400 se začala rozvíjet později než APPC v rámci snahy o větší otevřenost vůči ostatním počítačovým systémům. Dnes již obsahuje veškeré funkce obvyklé u jiných operačních systémů a stala se dokonce hlavní komunikační metodou.
- **APPC** - Advanced Program-to-Program Communications (pokročilá komunikace program-program) je implementací metody SNA LU 6.2 a PU 2.1. Pracuje s pojmy LU - Logical Unit (logická jednotka) a PU - Physical Unit (fyzická jednotka). Zatímco fyzické jednotky jsou počítače, logické jednotky jsou komunikační programy umístěné ve fyzických jednotkách. APPC tedy představuje soustavu komunikačních programů v počítačích AS/400, ale i v jiných počítačích, zejména osobních (PC). Tyto programy jsou schopny vzájemné komunikace a lze je vyvolávat různými způsoby v aplikačních programech. APPC se používá jako hlavní prostředek komunikace mezi počítači AS/400 navzájem a mezi AS/400 a PC když je nutné vyhnout se nebezpečí virů nebo hackerů z Internetu. Existuje řada hotových softwarových produktů používajících APPC dodávaných firmou IBM jako licenční programy. APPC však lze také snadno použít v aplikačních programech psaných např. v jazyku RPG nebo COBOL, popř. C.

Hlavní síťové metody

- **TCP/IP** zahrnuje mezinárodní síť (internet) i podnikové sítě (intranet).
- **APPN** Advanced Peer-to-Peer Networking je síťová metoda založená na komunikačním protokolu APPC, tedy SNA. Je nejběžnější metodou propojení počítačů AS/400, System/36, System/38 a osobních počítačů IBM.
- **IPX** Internet Package Exchange je síťová metoda firmy Novell, která používá protokolů TCP/IP. V počítači AS/400 je podporována jednak samostatně, jednak prostřednictvím aplikace AnyNet/400.
- **AnyNet/400** je prostředek podporující protokoly APPC a TCP/IP. Umožňuje aplikačním programům použít komunikační rozhraní APPC/ICF nebo APPC/CPI-C pro metodu TCP/IP nebo obráceně, rozhraní AF-INET (Address Family - InterNET) pro metodu SNA (Systems NetworkArchitecture). Od verze V3R6 je k dispozici také podpora metody IPX.

Hlavní linkové protokoly

Asynchronní linka. Velké množství zařízení (včetně počítačů IBM i a PC) lze propojit asynchronní linkou. Použitý protokol není slučitelný s architekturou SNA ani TCP/IP. Asynchronní linka se často používá ke spojení nevyžadujícímu velkou přenosovou rychlost.

Ethernet je druh lokální sítě - LAN (Local Area Network). Počítá se k linkám, přestože jde o síť počítačů. Používá se prostřednictvím APPC nebo TCP/IP.

Token-ring je druh lokální sítě - LAN (Local Area Network). Používá se prostřednictvím APPC nebo TCP/IP.

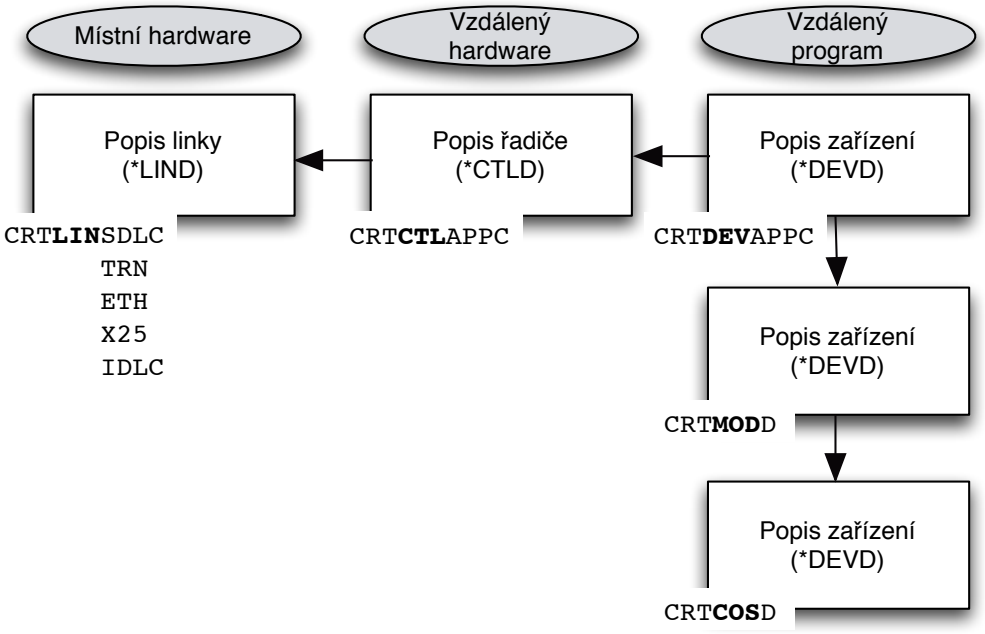
SDLC - Synchronous Data Link Control je komunikační protokol pro synchronní přenos binárních dat po telefonních spojích. Vyhovuje architektuře SNA, používá se prostřednictvím APPC/APPN.

X.21 je linkový protokol pro telefonní spoje vyhovující metodám SNA, OSI a TCP/IP.

X.25 je linkový protokol pro telefonní spoje vyhovující metodám SNA, OSI a TCP/IP a asynchronní metodě.

Komunikační konfigurace

Příklad komunikační konfigurace pro metodu APPC



Příkazy pro vytvoření komunikační konfigurace:

CRTLIN...	Create line description (podle druhu linky)
CRTCTL...	Create controller description (podle sousedního počítače v síti)
CRTDEV...	Create device description (podle druhu zařízení)
CRTMODD	Create mode description (jen pro APPC)
CRTCOSD	Create class of service (jen pro APPC)

Vytvořená konfigurace musí být zapnuta (aktivována), aby mohla být použita ke komunikaci. Po skončení komunikace může být vypnuta. Zapnutí se nazývá "**vary on**" a vypnutí "**vary off**".

Příkaz k zapínání a vypínání konfigurace je jen jeden, ale jeho činnost se liší podle parametru STATUS:

VRYCFG STATUS(*ON)	Vary configuration on (vypnout konfiguraci)
VRYCFG STATUS(*OFF)	Vary configuration off (zapnout konfiguraci)

Příkaz VRYCFG lze použít jak na celou konfiguraci (linku, podřízené řadiče a zařízení) nebo jen na její část, popř. jen jeden konfigurační objekt (např. zařízení).

Výsledek příkazu VRYCFG lze pozorovat s pomocí příkazu

WRKCFGSTS	Work with configuration status

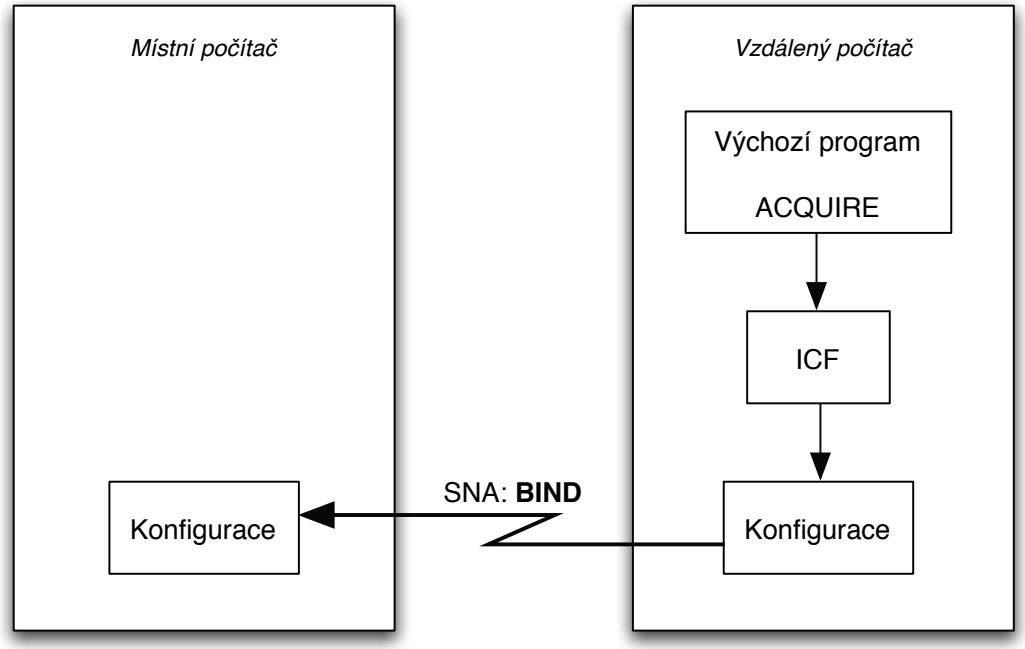
Schema vytváření spoje a konverzace v APPC

Programy v APPC (ale i v některých jiných metodách) spolu komunikují prostřednictvím konfigurací a komunikačních souborů (ICF souborů). Komunikační soubory jsou objekty typu ***FILE** s atributem **ICFF** (Intersystem Communications Function File).

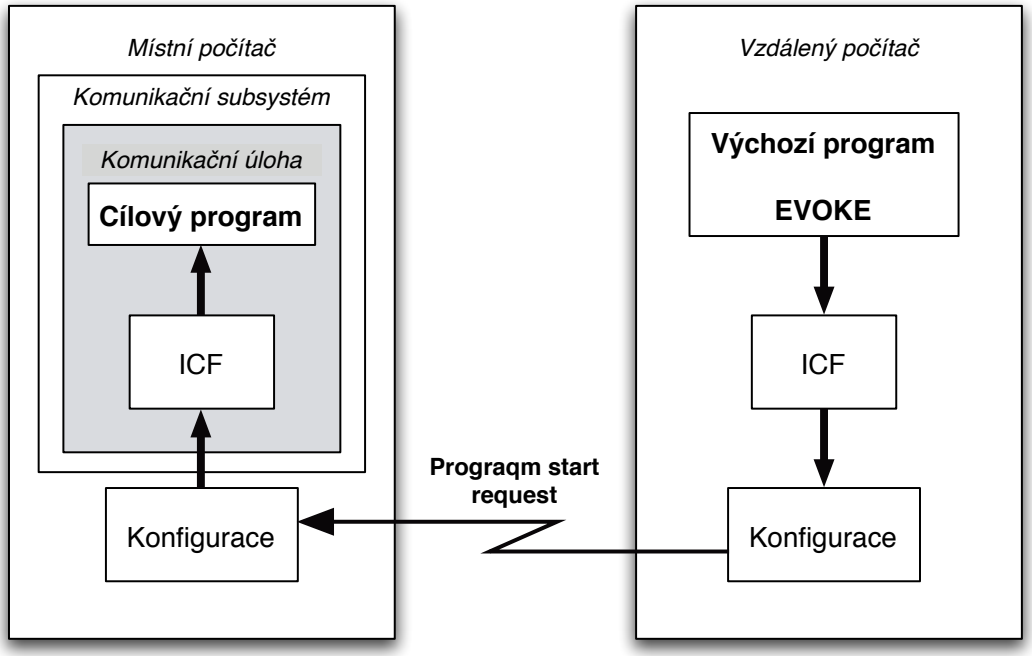
K zahájení dialogu je nutné po zapnutí konfigurace nejprve vytvořit tzv. logický spoj (session), což rovněž provádí výchozí program. Výchozí program může vytvořit více spojů a dialogů, a to se stejným cílovým počítačem nebo s jinými cílovými počítači. Jeden počítač může být zároveň cílový i výchozí (ovšem s různými výchozími a cílovými programy).

Následující obrázky naznačují (velmi zjednodušeně), jak se vytváří logický spoj (session) a jak se zahajuje dialog (conversation).

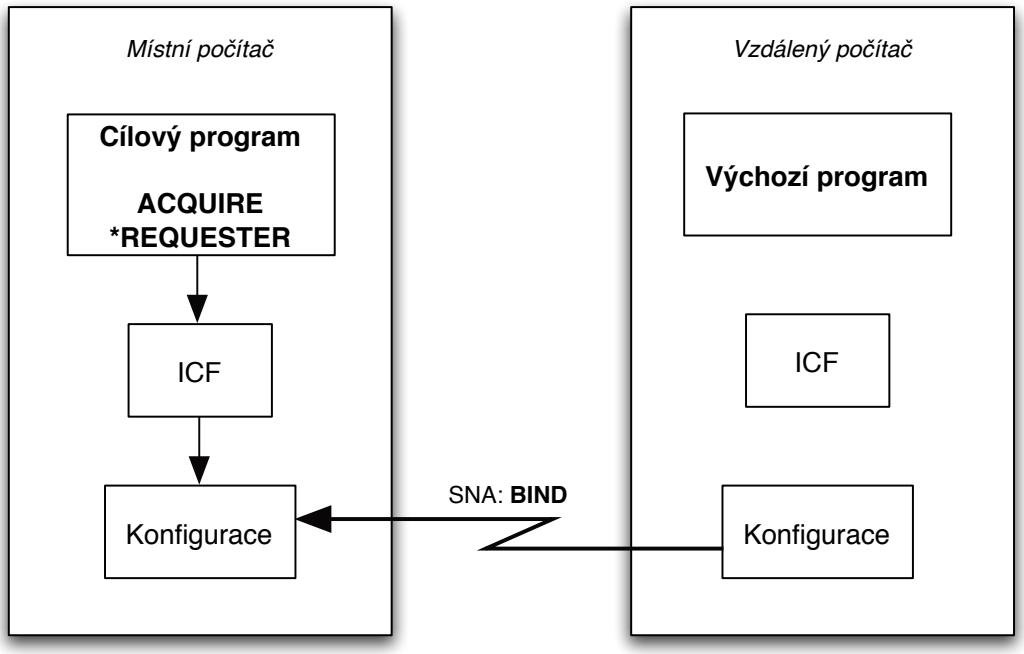
Výchozí program nejprve vydá příkaz ACQUIRE, čímž vytváří logický spoj (session).



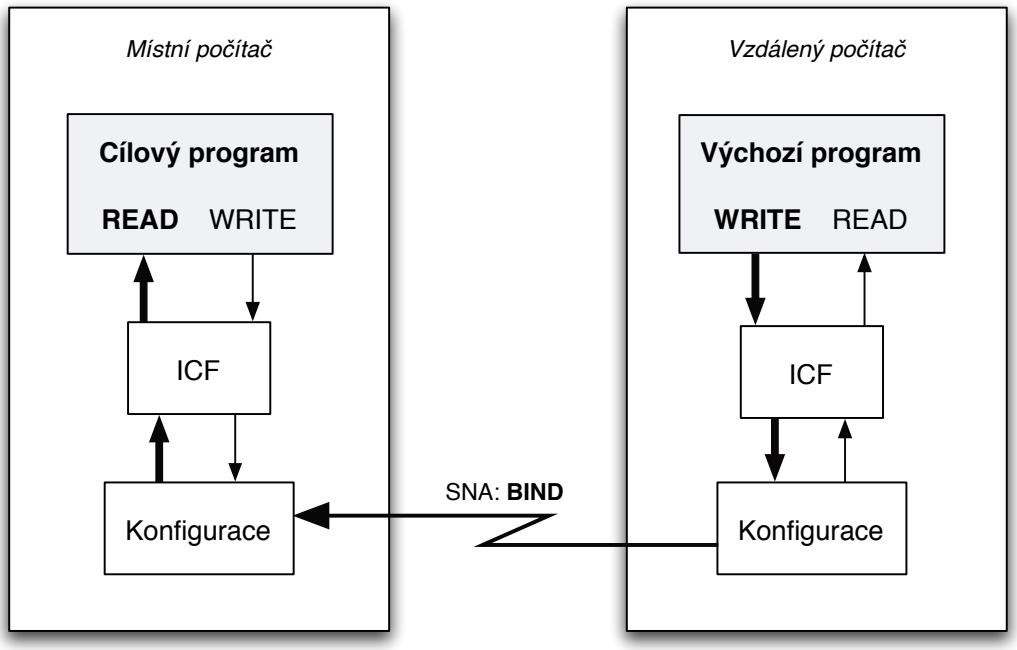
Dále výchozí program vydá příkaz EVOKE, v němž určuje program, který se má nastartovat v cílovém počítači. Tento cílový program pak běží v rámci komunikační dávkové úlohy, která sama běží v rámci komunikačního subsystému. Který subsystém to bude, je dáno poněkud složitějšími pravidly. Zpravidla to bývá subsystém QCMN.



Cílový program pošle příkaz ACQUIRE zpět výchozímu počítači, který vyslal příkaz EVOKE (ten se označuje jako *REQUESTER). Touto odpovědí cílového programu je zahájen dialog (konverzace).



Dialog pak pokračuje výměnou zpráv (WRITE, READ).



Oba komunikující počítače nemusí spolu sousedit, je-li použita síťová metoda APPN. Který počítač nazveme místním (local) a který vzdáleným (remote), závisí jen na stanovisku pozorovatele. Výchozí (source) program a cílový (target) program je však důležité rozlišovat. Výchozí progam zahajuje komunikaci, cílový je teprve vyvolán na základě příkazu z výchozího programu. Rovněž způsob výměny zpráv je zpravidla řízen výchozím programem.

Tyto principy se týkají jak aplikačních programů psaných v progamovacích jazycích, tak hotových programů, jako je např. Display Pass-Through nebo DDM (viz dále).

Display Pass-Through

Display Pass-Through znamená přibližně "průchod obrazovky". Tato funkce dovoluje uživateli přihlásit se ze své pracovní stanice "na dálku" k cílovému počítači, a to prostřednictvím tzv. *virtuálního zařízení*. (virtual device).

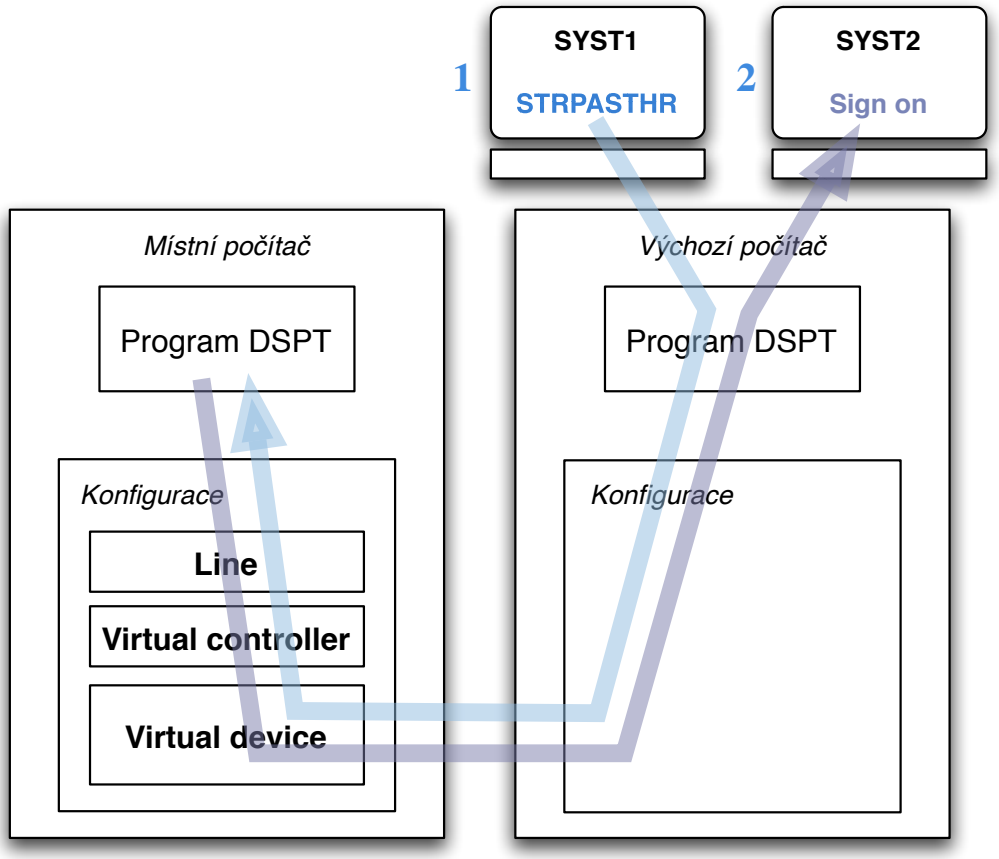
Ve výchozím počítači musí existovat popis řadiče (controller description) odpovídající sousednímu vzdálenému počítači a popis zařízení (device description). Popis zařízení se vytvoří automaticky, je-li použita síťová metoda APPN.

V cílovém počítači se vytvoří virtuální zařízení (virtual device) automaticky, dovoluje-li to systémová hodnota QAUTOVRT (určující maximální počet virtuálnních zařízení). Virtuální zařízení se připojuje k virtuálnímu řadiči (virtual controller) QPACTL*nn*. K propojení se vzdáleným počítačem slouží příkaz

STRPASTHR Start passthrough

V něm se uvede *jméno vzdáleného místa* (remote location name), které často bývá shodné se jménem počítače (zjistíme příkazem DSPNETA na vzdáleném počítači). Poté se na lokální pracovní stanici objeví obrazovka Sign-on, jako kdybychom se hlásili přímo ke vzdálenému počítači. Po přihlášení můžeme pracovat se vzdáleným počítačem jako kdybychom k němu byli připojeni lokálně. Propojení se zruší příkazem

ENDPASTHR End passthrough



1. Uživatel u počítače SYST1 vydá příkaz STRPASTHR se jménem cílového počítače (zde SYST2). Systémový program DSPT (Display Pass Through) zajistí vytvoření virtuálního řadiče (virtual controller), jestliže ještě neexistuje, a virtuálního zařízení (virtual device). Pak vytvoří logický spoj a dialog mezi oběma programy DSPT. Virtuální zařízení zprostředkovává komunikaci mezi oběma počítači. Program DSPT simuluje funkci obrazovkového terminálu na počítači SYST2.
2. Programy DSPT ve spolupráci zobrazí na obrazovce terminálu obrazovkový formát Sign-on pro počítač SYST2 a uživatel se k němu může přihlásit.

Distributed Data Management (DDM)

DDM - Distributed Data Management (řízení distribuovaných dat) je velmi silný nástroj umožňující pracovat s databázovými soubory umístěnými ve vzdáleném počítači. Umožňuje navíc provádět na vzdáleném počítači i CL-příkazy zadané z místního počítače. DDM nevyžaduje dodatečné programování. Používá mechanismu APPC a APPN.

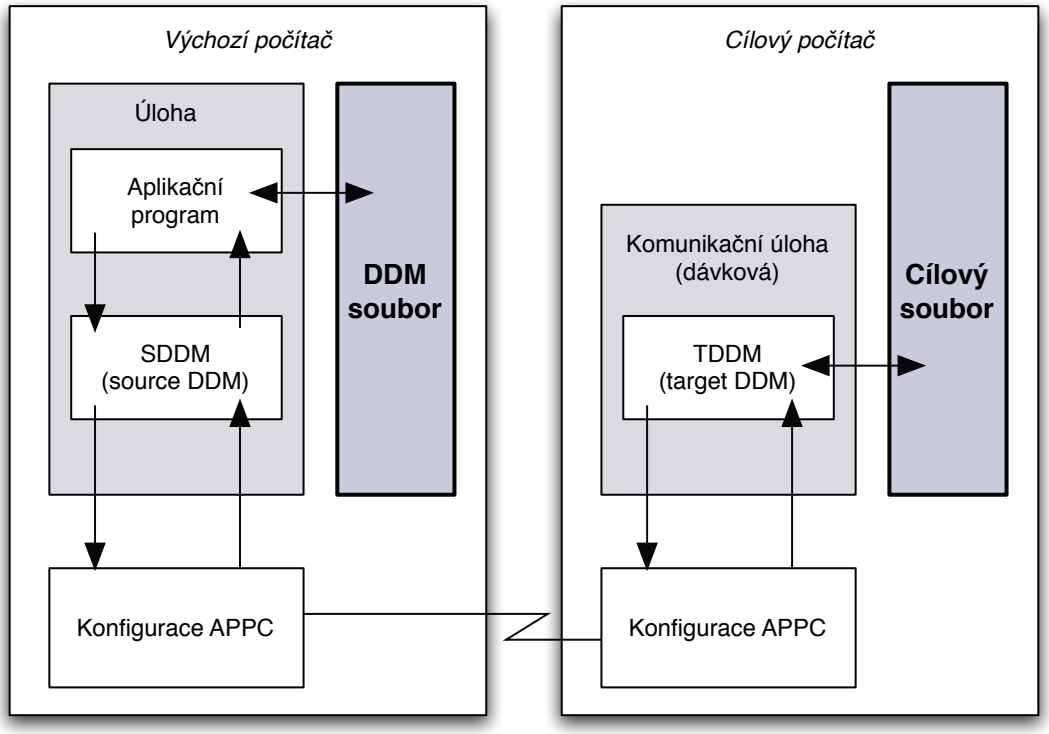
Místní počítač, v němž pracuje aplikační progam, nazýváme *výchozí - source*, vzdálený počítač, v němž sídlí databázový soubor (případně jiné objekty, s nimiž chceme pracovat, nazýváme *cílový - target*. V obou počítačích musí být pochopitelně vytvořeny příslušné komunikační konfigurace.

Ve výchozím počítači vytvoříme objekt zvaný *DDM soubor*, který slouží jako prostředník k práci se skutečným databázovým souborem umístěným v cílovém počítači. DDM soubor je objekt typu *FILE s atributem DDMF a neobsahuje žádná data. Musí být pojmenován (jako každý objekt) a musí odkazovat na skutečný databázový soubor umístěný v cílovém počítači (cílový soubor).

Příkazy pro práci s DDM souborem:

CRTDDMF	Create DDM file
WRKDDMF	Work with DDM files

Následující schema znázorňuje dva počítače komunikující prostřednictvím programů a souborů DDM.



Když se aplikační program po prvé pokusí otevřít DDM soubor, operační systém nastartuje program SDDM (source DDM). Program SDDM je součástí stejné úlohy, v níž běží aplikační program. Jestliže ještě není zahájena konverzace se vzdáleným cílovým počítačem, vytvoří se logický spoj a zahájí se konverzace. To se uskuteční tím, že SDDM vyšle požadavek na start programu TDDM včetně všech potřebných parametrů získaných z DDM souboru. Program TDDM je pak spuštěn jako dávková komunikační úloha připravená obsluhovat požadavky z výchozího počítače (aplikačního programu).

Požadavky aplikačního programu jsou transformovány programem SDDM a zasílány po lince programu TDDM. Program TDDM zpracuje požadavek s použitím cílového databázového souboru odpovídajícího DDM souboru a pošle výsledky zpět programu SDDM a aplikačnímu programu.

S DDM soubory pracují kromě aplikačních programů psaných ve vyšších jazycích i programy

Query/400
SQL/400
IBM i Access Client Solutions

a také příkazy CL určené pro práci s databázovými soubory

CPYF	Copy file
DSPPFM	Display physical file member
OVRDBF	Override database file
OPNQRYF	Open query file (výběr a třídění)

DDM pracuje i s jinými objekty než s databázovými soubory; např. z výchozího programu lze zadat příkaz

SBMRMTCMD	Submit Remote Command

který spouští zvolený příkaz na cílovém počítači.