

RACIONALIZACE TVORBY ŘÍDICÍCH ALGORITMŮ ASŘTP PŘI PODOBNÝCH PROJEKTECH

Ing. R. Loydl, CSc., Ing. J. Rusek, Ing. M. Zdráhal, Ing. R. Klvaňa

1. Úvod

Výstavba automatických systémů řízení technologických procesů (ASŘTP) je práce mimořádně náročná na čas, kvalifikaci lidí a finanční náklady. Proto je snahou snížit pracnost využitím prvků opakovatelnosti. V přednášce budou popsány poznatky z výstavby dvou ASŘTP v cementářském průmyslu z hlediska využití opakovatelnosti projektů řídicích algoritmů pro řízení výpalu v rotačních pecích /1/, /2/.

Předpokládá se opakování základních prvků počítačové konfigurace. Jistě je možné najít prvky opakovatelnosti i bez tohoto předpokladu, ale i z jiných hledisek (např. náhradní díly, stejný software) je tento předpoklad reálný. V posuzovaném případě se jedná o řídicí počítače SIEMENS 330, nasazené v Kombinátě Záhorie v Rohožníku a v CVPR v Prachovicích. Dalším předpokladem je podobná sestava technologického zařízení, z čehož vyplývají požadavky na podobné úlohy řízení.

Řízení komplexu technologických procesů je vždy systém programů. V podstatě jsou dva extrémní případy organizace těchto programů. Jeden z nich je spojení po horizontálních liniích, tj. centralizovat programy sběru dat (analogových i digitálních), a druhý po vertikálních liniích, tj. každé technologické zařízení je řízeno systémem svých programů od sběru dat až po řídicí zásahy. Tato různá organizace programů je možná u systémů s jedním centrálním počítačem i u systémů víceprocesorových, i když organizace programů i použitý hardware se vzájemně ovlivňují.

V posuzovaných případech je programový systém vybudován takto (obr. 1) :

- a/ Sběr analogových, digitálních a binárních signálů z procesu je centralizovaný. Programy sběru dat ukládají výsledky měření do paměti, ke které mají přístup všechny ostatní programy systému.

b/ Programy kontrolních hlášení, alarmy jsou také centrální. Jsou to programy, provádějící jednoduché kontroly snímaných signálů.

c/ Jednotlivé programy ASŘTP mají přidělenou část paměti, do které nemohou ostatní programy zasílet informace. Čtení paměti, přidělené jinému programu, je možné. Úmyslně se zde nerozděluje na operační paměť a na vnější paměť. Toto rozdělení je již v současnosti velmi flexibilní.

2. Prvky opakovatelnosti na úrovni systému

Z obr. 1 vyplývá možnost opakovatelnosti všech vazeb, vyznačených na obrázku. Toto se plně využilo v uváděných konkrétních případech. Išlo o zachování hierarchie výstavby, změnil se reálný obsah programů s ohledem na jiné okolní prostředí.

V obou projektech se použila v zásadě stejná symbolika označení vstupních a výstupních signálů procesu.

2.1 Konfrontace "směna - opakovatelnost"

Při vyhodnocování prvního projektu, který i časově předcházel druhému, se objevilo několik chyb. Například analogové veličiny měly přidělený text tvaru

AAAA.Pmmn

kde mmmn je číslo veličiny

P (nebo jiný znak dle normy) označuje druh signálu, např. průtok, analýza, elektrický příkon, atd.

Čtyři písmena AAAA vyplývají z vlastnosti použitého vyššího programovacího jazyka /3/. Především celková délka textu je 10 znaků.

Použití čtyř písmen A na začátku bylo nepraktické a nevhodné pro běžný styk s počítačem. Proto se to pro druhý projekt skrátilo na dvě písmena A, čímž se délka textu skrátila na 8 znaků. Při velkému počtu veličin a dalších textů stejné délky (návy části programů, návy veličin v programech) se dosáhla úspora paměti.

Na druhé straně se nedomyšlel vliv této změny. Všechna standardní i uživatelská hlášení měla přispůsobené formáty hlášení

na délku textů 10 znaků. Zkrácením textu bylo nutno přepracovat všechny formáty hlášení, včetně testovacích, poruchových i uživatelských. Tak se celý efekt zkrácení textu vytratil.

Z tohoto příkladu vyplývá, že při větších systémech je někdy vhodnější opakovat menší "chybu krásy", než ji opravovat a nedomyšlet všechny důsledky.

2.2 Opakovatelnost jednotlivých programů v systému

Jednotlivé programy systému jsou označeny číslem a názvem. Standardní programy (testovací, obslužné, redakční a pod.) mají pevně daná čísla a označení. Výhody této opakovatelnosti se ale nevyužily pro uživatelské programy.

3. Opakovatelnost programu

Využitím možnosti uspořádání paměti podle obr. 1 vzniká možnost opakovat jednotlivé řídicí programy pro podobné úlohy. Protože podobné programy jsou nasazeny v různých závodech, vzniká problém jak zabezpečit, aby rozdily co nejméně ovlivňovaly program. Bylo to vyřešeno tím, že úvodní část řídicích programů provádí čtení potřebných veličin centrálního paměťového pole a zanáší je do programového paměťového pole. Tam se jim přidělí označení, která jsou již nezávislá od místa realizace a tedy opakovatelná ve dvou různých projektech.

Celá další část programu pracuje potom už s těmito interními označeními. Program na závěr obsahuje výstupní část, která opět musí být přizpůsobená skutečnosti.

Popsaný postup je na obr. 2. Opakovatelnou pracovní část programu podle obr. 2 je při použití stejných čísel (označení) analogických programů možné přímo nahrát do podobného řídicího systému dalšího počítače.

3.1 Výhody zvoleného uspořádání řídicího programu

Uspořádání dle obr. 2 se ukázalo účinné. Pracovní část programu (při podobném řízeném zařízení) se dala z velké části

opakovat. Další výhody se ukazují při testování. Uspořádání dle obr. 2 umožní off-line testování řídicího programu bez nebezpečí zásahu do procesu, nebo bez nebezpečí neočekávaných výsledků při neznámých údajích z procesu. Stačí odepnout úvodní a závěrečnou část.

Programátorsky zvolené uspořádání přináší kromě výhod opakovatelnosti částí programu i odpaďnutí nutnosti přeorientování se na jiné reálné podmínky. To se jeví při téměř současném ožívování v obou závodech jako velké psychické ulehčení pro programátory a systémové inženýry.

Využití prvků opakovatelnosti přináší podstatné zrychlení ožívování a uvádění ASŘTP do provozu. Je vhodné upravit jednotlivé programy tak, aby opakovatelné části byly nezávislé od vnějších podmínek. Při změnách celkové výstavby systému je třeba zvážit mimo výhody i nevýhody z nutného přepracování navazujících částí systému. Důsledná modulová výstavba přináší při opakovaných podobných projektech citelné časové úspory.

4. Problémy programového zabezpečení ASŘTP a zásahů do programů za chodu ASŘTP

4.1 Struktura programů ASŘTP výpalu portlandského slínku

Při analýze požadavků na řízení výpalu portlandského slínku v rotační peci s šachtovým výměníkem Přerovských strojíren byly definovány 3 základní okruhy problémů :

- 1/ Vyhodnocování průběhu teploty pláště pece a příkonu pohonu a z těchto sekundárních veličin posouzení průběhu výpalu slínku. Obě úlohy jsou z hlediska řízení na úrovni čidel.
- 2/ Řízení výpalu slínku, řízení náběhu nebo změny výkonu a příp. řízení chlazení slínku. Tyto programy svými výstupy řídí technologii.
- 3/ Archivace určitého úseku historie, určování počátečních podmínek (žádaných hodnot) při startu řídicích programů a provádění opatření při poruše některého čidla, tj. nadřazené dohližecí úlohy.

Z důvodů zvětšení přehlednosti, usnadnění testování a modifikací v průběhu testování a provozu a modifikací při použití v dalších projektech byly požadavky na řízení realizovány několika hierarchicky uspořádanými programy v souladu s dělením ad 1/ až 3/, viz obr. 3.

Řídicí programy mají strukturu podle obr. 2. Většina řídicích programů v prvním projektu obsahovala rychlou část (cyklus 10"), která prováděla čtení a základní zpracování dat a kontrolovala, jestli je technologie a signály v takovém stavu, aby mohl být program připojen, pomalou část (cyklus 2' a více), která obsahovala řídicí algoritmy, středně rychlou část (cyklus asi 40"), která realizovala některé kontrolní algoritmy a výstupní smyčky, a část startovanou jednorázově v prvním průběhu po zapnutí řízení, která zajišťuje uložení aktuálních programových konstant na předepsané adresy programově residentní paměti. Přenos dat mezi programy se uskutečňuje přes centrální paměťové pole, do kterého jednotlivé programy ukládají sdílená data.

Skutečnost, že většina programů obsahovala mimo pomalejší i rychlé smyčky, vedla při dané konfiguraci počítače k tomu, že každý řídicí program musel být ve svém nejrychlejším cyklu přenášen z periferní paměti do operační, odstartován a po ukončení činnosti uložen zpět. Přenosy mezi paměťmi zabíraly neúnosně mnoho času.

V dalších projektech, kdy se stalo časové zatížení centrální jednotky limitujícím faktorem, byly rychlé části všech řídicích programů shrnuty do jednoho krátkého rychlého programu a tím se dosáhlo minimálně čtyřnásobné snížení četnosti startování řídicích programů. Z hlediska programování zůstaly algoritmy bez podstatných změn, došlo jen k přečíslování programových bloků a adres v programové paměti.

Startování řídicích programů organizuje rychlý koordinační program, který je startován v základním cyklu startování uživatelských programů (10") organizačním programem.

Řídicí programy mohou být ve třech stavech, a to : řízení je zapnuto, řízení je vypnuto a řídicí programy jsou odpojeny. Stav "řízení zapnuto/řízení vypnuto" je odvozen od polohy pře-

pínačů "ručně/počítač", které jsou u každého akčního orgánu. Pro zapnutí řízení musí být všechny přepínače akčních orgánů, ovládaných daným řídicím programem, v poloze "počítač". Pro stav "řízení vypnuto" stačí přepnout jediný z těchto přepínačů do polohy "ručně". V tomto případě je rychlý koordinační program nadále startován organizačním programem, ale startování dalších řídicích programů je blokováno. Ve stavu "řídicí program odpojen" je činnost programu sablokována prostřednictvím obslužného programu. V tomto případě není program uveden v činnost, i když je startován. Odpojení programů se běžně napoužívá a má se být prováděno jen pověřenou osobou.

4.2 Manipulace s programy

Před zapnutím řízení, které je popsáno v předcházejícím odstavci, je nutno zadat žádané hodnoty regulovaných veličin. Žádané hodnoty jsou uloženy do centrálního paměťového pole buď operátorem podle jeho volby, nebo automaticky jako výsledky výpočtu, probíhajícího v dohlížečím programu. Uložení žádaných hodnot ručně lze provést pomocí instrukcí standardního obslužného programu přímo na odpovídající adresy v centrálním paměťovém poli. Tato manipulace může být provedena jen systémovým pracovníkem. Pro pracovníky obsluhy technologie je určeno zadávání pomocí kódových slov nebo zadávacího formuláře.

Kódové slovo je prostředek, který sdružuje potřebný počet instrukcí standardního obslužného programu, ale z hlediska uživatele je reprezentováno mnemotechnickým výrazem a sledem žádaných hodnot v předepsaném pořadí, např. RIZRP:1300,900,1.5; ◊. Nevýhodou kódových slov je to, že standardní obslužný program slouží k manipulacím se všemi uživatelskými programy a nemůže současně zpracovávat více požadavků, tzn. že kódové slovo nemusí být provedeno je-li obslužný program zaměstnán nějakou delší souvislou činností, nebo když je prováděno tak blokuje provedení jiných požadavků.

Zadávací formulář je zobrazen na alfanumerické obrazovce - 24 řádků, 80 sloupců. Obsluha vyplní odpovídající políčka a zmáčkne příslušné tlačítko na tastatuře. Obdobně je možno

zadávat žádané hodnoty i z barevného semigrafického displeje. Měnit žádané hodnoty je možno kdykoliv v průběhu řízení. Zanesení žádaných hodnot do centrálního paměťového pole realizuje standardní program, který slouží jen k manipulaci se zadávacími formuláři. Jakékoliv zadávání ze zadávacího formuláře lze protokolovat na vybrané výstupní periférii.

Změna programových konstant je popsána v návodu k obsluze programů a může ji provádět systémový pracovník prostřednictvím standardního obslužného programu. Za chodu programu je možno změnu provést tak, že se nové konstanty dosadí do všech algoritmů, ve kterých se s nimi pracuje. Tato manipulace je však dosti riziková, zvláště u konstant, které jsou použity na více místech programu. Doporučený postup je : vypnout řízení přepínačem "ručně/počítač", zadat novou konstantu do programové části, která je startována jednorázově při prvním průběhu řízení (viz výše) a opět přepínačem "ručně/počítač" řízení zapnout.

Změny programu, jako jsou úpravy algoritmů, rozšíření nebo zkrácení programu o dílčí algoritmy a pod., může provádět jen systémový pracovník, který program podrobně zná. Standardní obslužný program umožňuje provádět změny i když je řízení zapnuto. Záleží na charakteru změny a rutinně systémového pracovníka, ve kterém ze tří stavů programu změnu provede.

Korespondence s počítačem pomocí standardního obslužného programu je možná přes dálkopis nebo obrazovku. U dálkopisu dochází automaticky k protokolování všech manipulací s programy. Při použití obrazovky je možnost protokolování manipulací na papírové médium volitelná. Ve většině případů protokolování probíhá.

Přeložené řídicí programy jsou uloženy na bubnové paměti počítače, ze které jsou cyklicky po segmentech nebo celé přenášeny do operační paměti. V případě poruchy počítače, při revizních prohlídkách a pod. může dojít k porušení obsahu bubnové paměti. Proto jsou řídicí programy ve strojovém kódu archivovány ještě na dalším médiu, většinou na děrné pásce nebo děrných štítcích. Po každé úpravě řídicích programů se doporučuje provést jejich novou archivaci.

4.3 Oživování a testování programů

Oživování a testování programů probíhá ve třech fázích. V 1. fázi jsou odstraněny syntaktické chyby a je proveden test v otevřené smyčce. Díky modulární stavbě programu jsou jednoduchým příkazem odpojeny vstupní a výstupní části programu a na odpovídající adresy v programové paměti jsou uloženy zvolené hodnoty vstupů. Program je manuálně startován a je kontrolována správnost činnosti programu, včetně výpočtu akčních zásahů. Ve druhé fázi jsou připojeny vstupní části programu a je testováno zpracování reálných vstupů. Z tohoto testu vyplývají úpravy konstant regulátorů a korekce vazeb uvnitř programu. Ve třetí fázi je prováděn test v uzavřené smyčce, kdy jsou připojeny jak vstupy, tak výstupy programu. Tento test postupně přechází ve skutečný provoz programu. Během testu v uzavřené smyčce jsou jen doladěny algoritmy. V případě nutnosti rozsáhlejších úprav algoritmu se doporučuje návrat ke druhé případně první fázi testování.

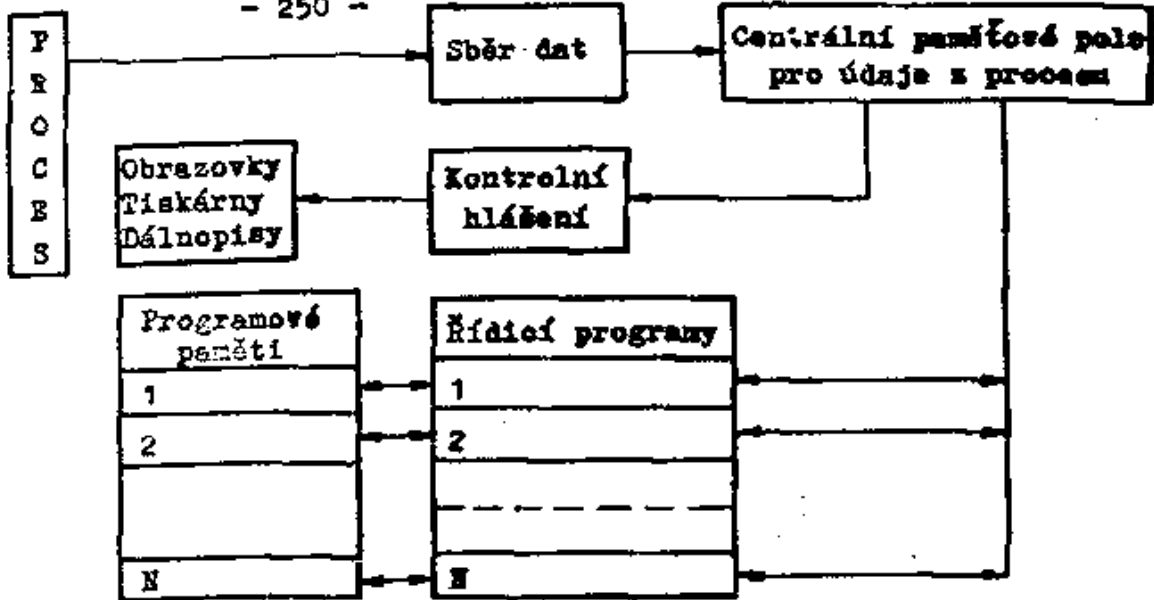
Při testování je vždy třeba volit takové postupy, aby nebylo ohroženo ani technologické zařízení, ani počítač.

5. Závěr

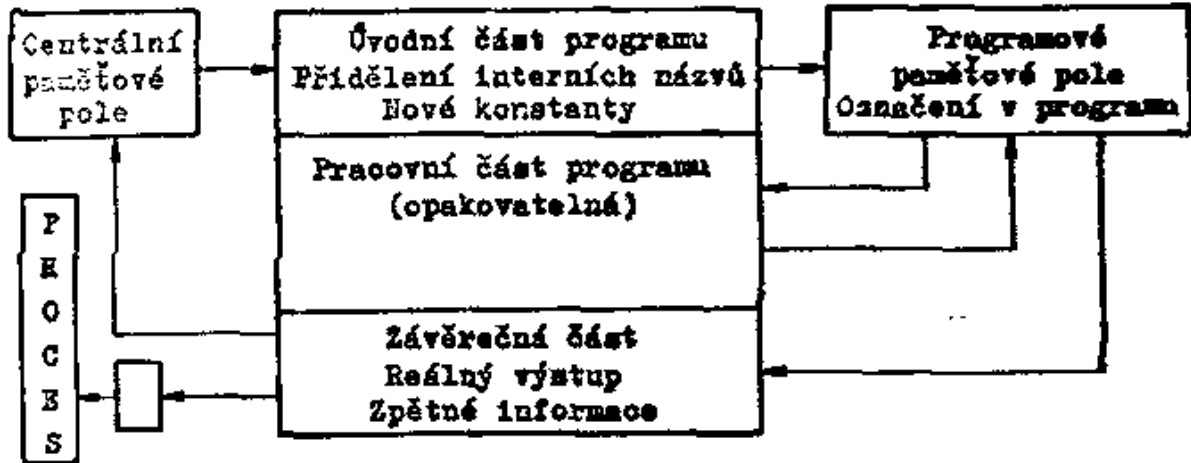
Real-time systémy ASŘTP jsou nedílnou součástí technologie. Při projektování je třeba přihlížet k tomu, aby celková úroveň technologického zařízení a vybavenost prvky měření a regulace byly adekvátní zásadám ASŘTP a naopak. Také při posuzování účinnosti ASŘTP je nutno vycházet z komplexního hodnocení technologie.

Literatura

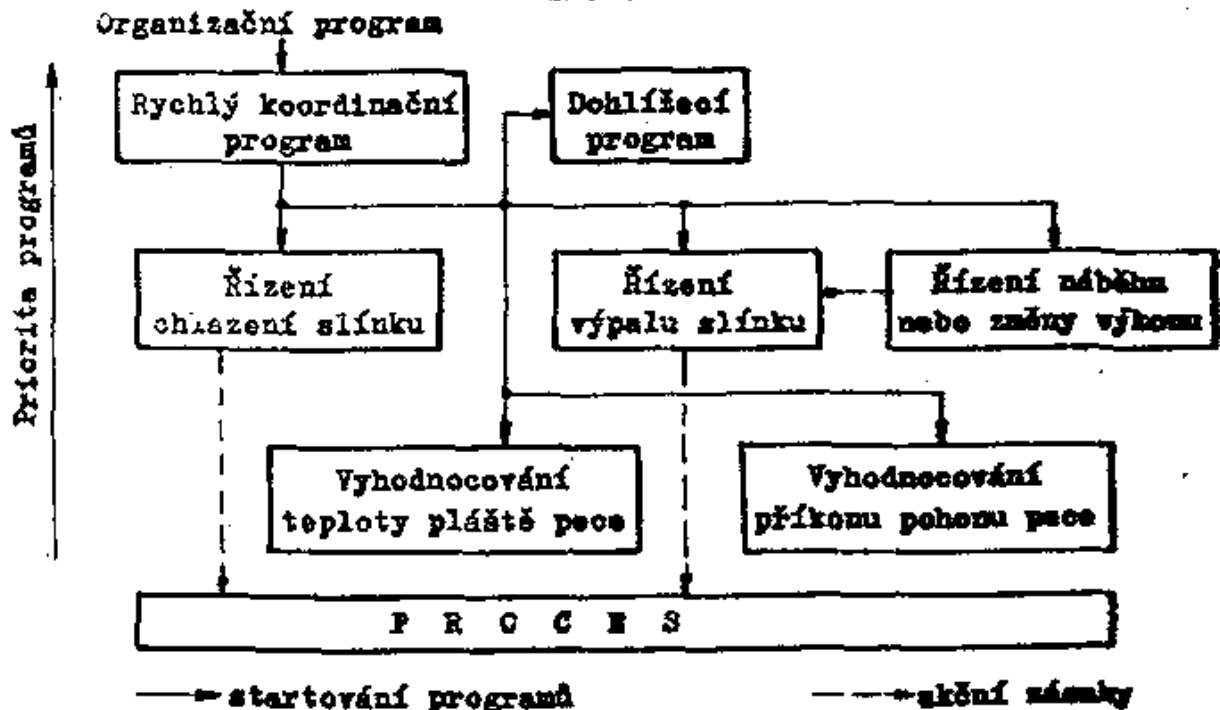
- /1/ Svobila, Klvaňa, Loydl, Schwobe, Hermann :
Automatisierung der Drehofen anlage des Zementwerkes Turna nad Bodvou (CSSR) mittels Prozessrechner.
Zement-Kalk-Gips 2/1978, Wiesbaden
- /2/ Svobila, Klvaňa, Loydl, Ruzak, Wackerle :
Adaptive Regelung von Zementofenlinien in der CSSR mit
Prozessrechner. Zement-Kalk-Gips 3/1983, Wiesbaden
- /3/ SIMAT-PROJ, Beschreibung
SIEMENS, Produkt Nr. P71100-80031-X-A2-35 - 1976, Erlangen



Obr. 1



Obr. 2



Obr. 3