K D E Z A Č Í N Á P R O G R A M Á T O R

Jiří Pechlát

1. Úvod

Diskuse programátorů připomínají často zmatení jazyků. Mluvíme o stejných věcech stejnými slovy, ale rozumíme každý něco jiného. Důvodů je víc, ale základní bude asi v tom, že odlišně chápeme úlohu programátora. Do značné míry je to dáno odlišnostmi postavení jednotlivých programátorů popřípadě skupin programátorů v různých podnicích. Od programátorů individualistů, kteří zcela sami v podstatě nekontrolovaně pracují na jediném mini či mikropočítači v podniku, až po velké programátorské kolektivy, vřazené do procesu tvorby ASŘ, ve kterých existují uvnitř programování specializace.

Různé postoje a různé názory souvisí s rozdíly v chápání konečného produktu vlastní práce. Hodnocení tohoto produktu okolím, zejména nadřízenými, ve značné míře ovlivňuje sebehodnocení programátora. V typových situacích ovšem dochází k typovým postojům. Typovou situací dneška je existence a budování rozsáhlých programových komplexů. Nebude tudíž od věci sledovat programátora v tomto kontextu.

2. Programování v praxi

Zdá se, že většina potenciálních či skutečných uživatelů výpočetní techniky má dojem /posilovaný i sdělovacími prostředky/, že na tom programování vlastně nic není: programátor sedne a naklepe do počítače pár povelů. Co je horší, podobnému dojmu podléhají často i vedoucí pracovníci odpovědní za tvorbu a využívání ASŘ. Práce programátora je pak hodnocena čistě utilitárně: co nejrychleji realizovaná sestava /popřípadě sestavy/, která co nejvíce odpovídá momentálnímu náhledu uživatele. Tento momentální náhled se vydává za jeho potřebu.

Vztah mezi uživatelem a programátorem je už roky v centru pozornosti. Praxe dokázala, že tradiční metoda opakovaných pokusů a omylů, jimiž se obě strany přibližovaly k přijatelnému kompromisu mezi požadavky a možnostmi, je neefektivní a pro budouc-

nost neudržitelná. Mnozí lidé, mezi nimi i řada programátorů a jejich nadřízených, si uvědomili, že mezi uživatelem a programátorem je vakuum, které brání kvalifikované dohodě. Snaha zaplnit tento prostor vedle ke vzniku nových funkcí ať už pod hlavičkou analytik, systémový analytik, organizátor či koordinátor. Bohužel zdaleka ne vždy plní to, co se od nich očekává, totiž úlohu spojky mezi uživatelem a programátorem. Proto jsou obcházeni a nadále je rozhodující kontakt programátora s uživatelem, ve kterém především programátor chtě nechtě musí jejich úlohu suplovat. Dělá to samozřejmě podle svých možností a schopností, což často nestačí, protože jeho schopnosti bývají zaměřeny na programování a jeho možnosti jsou omezovány rostoucí kapacitou vázanou na údržbu hotových programů a zvyšujícím se tlakem vedení na rychlé výsledky automatizace.

Nad tím, proč tyto funkce nefungují, by se měli zamyslet především vedoucí pracovníci odpovědní za budování ASŘ. Z vlastní zkušenosti mohu nabídnout několik dohadů:

- chybně formulovaná pracovní náplň,
- nedůslednost při praktickém uplatňování pracovní náplně,
- záměrné obcházení činností, které mají zmíněné funkce plnit.

Pracovní náplň bývá formulována obecně, nekonkrétně, jindy bývá opsána z náplně obdobné funkce v jiné organizaci a nerespektuje skutečnou organizaci práce ve vlastním útvaru. Ovšem ani dobře formulovaná pracovní náplň nemusí přinášet výsledky. Pracovníci v těchto funkcích jsou pověřováni úkoly, které jim nepřísluší, ale které se nadřízeným jeví jako důležitější, nebo jsou dokonce záměrně obcházeni, protože činnosti, které mají vykonávat, "zdržují" realizaci úlohy. Konec konců jsou podniky, které podobné funkce vůbec nemají a je otázka, zda na tom nejsou lépe.

Nemíním nikterak napadat pracovníky uvedených profesí, je ovšem nutné předpokládat, že uvedené skutečnosti zvyšují pravděpodobnost výskytu podprůměrných a neschopných. Kořeny jsou zjevně v organizační a metodické neujasněnosti. Vedoucí pracovníci sami nevědí, co by měli chtít. Požadují od těchto pracovníků organizační schopnosti, vzdělání v oboru, jehož problematiku příslušný ASŘ řeší, často i systémově inženýrské vlastnosti. Zapomínají však, že tito lidé musí být schopni navrhovat složité automatizované systé-

my, musí tedy mít širší znalosti v oblasti výpočetní techniky než tradiční programátoři.

3. Budování rozsáhlých systémů

V současných podmínkách je typickým produktem automatizačních snah složitý a rozsáhlý systém, ať už jde o operační systém nebo ASŘ technologického či ekonomického charakteru. Budování těchto systémů má svá pravidla, která nelze beztrestně obcházet. Potvrzují to zkušenosti doposud častěji negativní než pozitivní.

Východiskem je analýza systému, podklad pro volbu oblastí, vhodných k automatizaci /popřípadě k inovaci automatizace/. Na ni navazuje projektování, kterým rozumíme návrh struktury automatizované části, postupně zpřesňovaný a rozváděný do stále detailnější úrovně. Na nejnižší úrovni jsou algoritmy realizovány pomocí programů a programových modulů. Následuje zavádění, ověřování a rutinní provozování systému. Závěrečné vyhodnocení provozu je už součástí nového kola analýzy systému.

Přijmeme-li výše uvedenou představu "výrobní linky", pak nás nutně dovede k závěru, že cílem není tvorba dokonalého jednotlivého programu, ale tvorba kvalitního, modifikovatelného, přenositelného projektu, tedy celého komplexu vzájemně propojených programů. Stejně nutně dojdeme k závěru o nezbytné návaznosti jednotlivých činností této linky, s čímž se samozřejmě neslučují časté averze mezi analytiky a programátory. Bohužel chápání programátorské činnosti v takovém kontextu je spíše výjimkou. Místo nutné systémového přístupu jsou programy /v lepším případě skupiny programů/ chápány jako izolovaná díla. Je možné, že takový pohled vychází od uživatele, který si nemusí uvědomovat nutné souvislosti, ale jak vedoucí ASŘ tak sami programátoři se mu z pohodlnosti či neznalosti podrobují, aby se ex post na něj vymlouvali.

V této souvislosti se vnucuje myšlenka na IPT /Improved Programming Technologies/ firmy IBM; zejména jeden aspekt je z hlediska našeho tématu zajímavý: zahrnutí všech činností, které předcházejí vlastnímu programování /tedy analýzy a návrhu systému/ pod programovací technologie. Otázku kvality programování opravdu nelze vyřešit odděleně od ostatních činností naší "výrobní linky".

4. Výrobní linka

Smyslem a cílem systémové analýzy z pohledu tvorby ASŘ je příprava podkladů k rozhodnutí o tom, zda vůbec systém automatizovat a které jeho části. Toto rozhodnutí přísluší samozřejmě vedení organizace, má-li však být kvalifikované, musí vycházet ze zmíněných podkladů. Nereálné úkoly, které vyjadřují nepodložená přání vedení, mají katastrofální vliv na přístup pracovníků /uživatelů i řešitelů/ k automatizaci.

Projektování definuje stavební prvky systému: datovou základnu, její strukturu, jednotlivé programy. Je to činnost zcela nezbytná a programátor ji potřebuje /dokonce i programátor individualista zcela nezávislý na jiných/. Vždy je nutno problém rozdělit na části, které lze zvládnout jedním programem nebo programovým modulem. Programátor nikdy nemůže zůstat zcela mimo projektování, naopak na kvalitě projektu je závislý konečný efekt programování.

Teoreticky by tedy mohlo být všechno jasné. Praxe ovšem ukazuje, že tyto činnosti jsou často oklešťovány nebo dokonce ignorovány. Počátky jsou asi už ve výchově. Zatímco k programování si většina vysokoškoláků během studia aspoň přičichne / i když ani tady není vše v pořádku/, s problematikou tvorby rozsáhlých programových systémů se nesetká. Je pravda, že se na školách vyučují předměty jako teorie systémů, systémová analýza, systémové inženýrství, jsou však vesměs příliš teoreticky zaměřeny a nemají vazbu na projektování. Ostatně stále ještě je mezi vysokoškoláky vysoký podíl těch, kteří si místo uceleného systému vědomostí odnášejí prales pro ně málo užitečných informací. Těžko pak od nich očekávat systémový přístup.

Patrně ještě neutěšenější je situace v projektování systémů řízení. Pokud je mi známo, jedině VŠ ekonomická se snaží naučit posluchače prakticky použitelný postup, jde však zatím o pokus, který musí teprve prokázat životaschopnost. To, co jsem se dověděl o výuce projektování ASŘ na některých jiných vysokých školách, je ještě horší. Předmětem výuky je pouhý přehled metod bez praktických příkladů a cvičení. Posluchači si odnášejí dojem, že jde o okrajový předmět, který nemá význam ve srovnání s hlavním zaměřením studia.

Další příčinou je špatná /nebo žádná/ definice "výrobní linky". Dokumenty, kterými jsou příslušné útvary zakládány a jimiž se řídí jejich činnost, jsou málokdy formulovány v tomto smyslu dobře. Poměrně nejlépe snad jsou na tom velké útvary /řádově stovky pracovníků/ či specializované organizace /například PVT/, kde se tvorby organizačních a provozních dokumentů zúčastňují odborníci se zkušenostmi z tvorby rozsáhlých programových komplexů. Na opačném pólu stojí jednotlivci nebo malé skupiny programátorů, kterých se podobné problémy téměř nedotýkají. Nejhorší je situace středně velkých podnikových útvarů /řádově desítky pracovníků/, neboť jsou pro podnik okrajovou záležitostí, která má sloužit hlavní činnosti podniku a jejich vnitřním fungováním se podnik málokdy zabývá. Pro tyto útvary je neefektivní vytvářet nějaké složité vlastní normy, a protože neexistuje obecná norma dostatečně konkrétní, nemají se tyto útvary čím řídit. Je žádoucí vytvořit pro ně poměrně jednoduchý typový projekční a programovací řád, který definuje "výrobní linku", náplně jednotlivých útvarů a funkcí, doplněný vzory důležitých dokumentů a formulářů.

5. Ještě jednou programování

Význam jednotlivých činností výrobní linky je relativní. S růstem složitosti systému roste i význam analýzy systému a projektování, a to zřejmě rychleji než obě zmíněné charakteristiky. Vývoj směřuje ke stále složitějším a rozsáhlejším systémům, jejichž tvorba snad bude jednou řízena centrálně, a proto dochází k trvalému relativnímu poklesu významu programování ve prospěch jiných činností. Čím složitější a rozsáhlejší je projekt, tím méně závisí jeho kvalita na vypiplání jednoho programu. Představa kvalitního programu se tak posouvá od řešení elegantního a chytrého k řešení pokud možno typovému, pružnému a průhlednému. Ve stejném směru působí i pokles relativní ceny zdrojů, zejména času a paměti. Úlohou programování se stává tvorba převážně typových prvků pro složité programové stavebnice, jejíž struktura je určována projektováním. Za těchto podmínek lze předpokládat, že v další etapě bude rukodělné programování nahrazeno automatizovanou velkovýrobou.

To však ještě pár let potrvá. Do té doby musí být programátor schopen vytvářet kvalitní programy ve výše uvedeném smyslu a daleko vyšší produktivitou než dosud. K dosažení tohoto cíle je nutné zdokonalovat jak vlastní programování tak celou "výrobní linku".

Existuje řada metod, které směřují ke zdokonalení programování. Je zbytečné právě na tomto semináři o nich znovu hovořit. Velkým nedostatkem je roztříštěnost jejich používání. Jen menší část vedoucích pracovníků má představu o těchto metodách a ještě méně je vyžaduje. Ani sami programátoři je příliš nemilují, protože znamenají změnu přístupu k vlastní práci. Zvyšující se tlak na automatizaci však nedává jinou možnost. Bylo by ovšem vhodné vybrat z těchto metod jednu nejvýše dvě, propracovat je do úrovně konkrétní implementace, vypracovat automatizovanou podporu a masově prosazovat. Je to jedna z cest, které bychom měli sledovat. Jisté kroky v tomto směru byly už podniknuty v oblasti Jacksonova strukturovaného programování.

6. Závěr

Ve svých poněkud neuspořádaných poznámkách jsem se snažil ukázat, že se nelze omezit jen na otázku vlastního programování a že je třeba o něm uvažovat v jistých souvislostech.

Problémy, kterých se tento příspěvek dotýká, jsou v zásadě společné všem programátorům, třebaže některých se dotýkají více, některých méně a některých snad zatím vůbec ne. Kdo jiný než programátoři, jejichž práce závisí na vyřešení těchto problémů, by měl mít větší zájem na vytvoření prostoru pro potřebné činnosti, ať už se rozhodneme zahrnout je do své vlastní pracovní náplně nebo si vynutit zřízení a důsledné využívání příslušných funkcí.

Zcela konkrétním námětem pro začátek může být návrh vytypovat, rozpracovat a prosazovat jednu metodu programování a vytvořit a uplatňovat typový projekční a programovací řád.