ACTA UNIVERSITATIS AGRICULTURAE ET SILVICULTURAE MENDELIANAE BRUNENSIS
SBORNÍK MENDELOVY ZEMĚDĚLSKÉ A LESNICKÉ UNIVERZITY V BRNĚ

Ročník LIV 15 Číslo 6, 2006

INTERAKTIVNÍ STYL PRODUKCE TESTOVACÍ BÁZE DAT PRO PODNIKOVÝ IS

M. Mišovič, I. Rábová

Došlo: 15. června 2006

Abstract

MIŠOVIČ, M., RÁBOVÁ, I.: *An interactive style of the testing database production for EIS*. Acta univ. agric. et silvic. Mendel. Brun., 2006, LIV, No. 6, pp. 133–144

Using a progressive Information Technology for development of Software Modules for Enterprise Information Systems brings a lot of practical and theoretical problems. One of them is a verification of results achieved in Life Cycle Stages, especially in the analysis stage. Instead of a very deep theoretical approach we can use quite practical testing by means of a testing database. Such testing database has to be constructed gradually from the Data Flow Diagram by a special algorithm.
This article introduces a formal description of the entity population and entity states. There is suggested to deal with fragments of the DFD that are produced with respect to the event set. This DFD event fragment is refined to transactions and their elementary functions. There is defined a transaction path in every transaction. By means of a special state equation system is generally defined conception of a correct functional processing of entities going along a selected transaction path. Solutions of such state equation systems are platform for getting a testing database.

database for IS testing, event, transaction, state equations, control event graph, path in control event graph, function correctness

MATERIÁL A METODIKA

Ověřování výsledků funkční a datové analýzy zajímá informatiky již řadu let. Snaha odhalit závady na úrovni esenciálního modelu klade jistě verifikaci do velmi užitečné roviny. Kardinální otázkou takto situované verifikace je „co“ porovnat s „čím“. Je to situace poněkud složitější než obdobná problematika pro jednoduché programy.

Naformulovat však to, co bylo původně zamýšleno pro funkci stávajícího a požadovaného podnikového informačního systému (PIS) zpracování informace, je nejen obtížné, ale i těžko porovnatelné se získanými výsledky funkční a datové analýzy.

Jelikož podstatu takovéto verifikace tvoří sémantika, je obtížné najít vhodný formálně-sémantický systém, který by podstatě kardinální otázky vyhověl. Kardinální otázku verifikace nelze obejít, můžeme jen najít pro nás její vhodnou interpretaci. Jednou z nich může být zavedení role analytika-projektanta a zástupce(-ů) uživatele jako komparátorů mezi tím, co bylo zamýšleno, a tím, čeho bylo analýzou dosaženo. Takový komparátor ale není schopen odhalit všechny nesrovnalosti, tj. formální a logické nekonzistentnosti.

Velmi častá je orientace na testování funkcionality aplikačního software IS podniku na ***testovací bázi dat***, protože jde většinou o aplikace typu Data Driven a praktické testování zpracování instancí entit v řetězcích procesů. Proto je hlavním cílem příspěvku návrh algoritmu, který zaručí produkci samotné testovací báze dat a využije některých teoretických poznatků souvisejících se strukturovanou analýzou.

Rozhodující roli v tomto přístupu hraje pojetí stavů entit a specifické převedení – transformace procesních diagramů na procesní řetězce – cesty s fragmentací kompaktního DFD na fragmenty podle událostí.

Dalším závažným krokem je zjemnění procesů ve fragmentu DFD k dané události na transakce a jejich elementární funkce. Potom se procesní cesty převedou až na cesty elementárních funkcí v transakcích. Zpracování instancí entit danou cestou v transakci je potom charakterizováno systémem stavových rovnic, jehož neznámé veličiny jsou právě hledané stavy, tj. hodnoty atributů entit. Jestliže existuje řešení zmíněného systému, tak jsme vlastně našli potřebné hodnoty atributů instancí entit zpracovávaných na dané cestě. Začneme tedy pojetím stavu entity, potom přikročíme k vyšetření zpracování entit v cestách obecné transakce a sestrojení a nalezení řešení systému stavových rovnic.

1. Zpracovávané entity a jejich stavy

Každá typová entita má množinu atributů, populaci svých instancí, odlišnou hodnotou definičního atributu a informační vazby na jiné entity. Na základě těchto pojmů postavíme definici entity a jednotlivých stavů, kterých nabývají všechny její instance.

Definice 1:

Entita **e** je datovou strukturou, která je definována pomocí tzv. charakteristiky C = [A, b, D], kde A = $\{a_1,a_2,...a_n\}$ je množina atributů, **b** je definiční (klíčový, primární) atribut a **D** je množina všech domén D = $\{Da_i \mid a_i \in A\}$. Každá entita má reálnou populaci **P** tvořenou počtem výskytů – instancí, tj. tzv. prvků populace. Nad prvky populace lze provádět různé operace, jejichž množinu označíme Ω a mezi entitami mohou existovat jisté binární informační relace, jejichž množinu označíme ρ. Potom je zápis entity **e** jako datové struktury ve tvaru **e** = [C, P, Ω, σ], kde σ je stav entity, jenž je určen dvojicí [P, ρ], přičemž populace **P** je stanovena realitou podle vztahu $P \subset \mathbf{X}_{a \in A} D_a$

Z Definice 1 plyne, že stav entity je určován:

- konečným počtem prvků populace – výskytů,
- hodnotami atributů jednotlivých prvků populace (některá doména může být nekonečnou množinou),
- konečným počtem relací ρ mezi entitou **e** a jinými entitami v ERD.

Operace z Ω mají různý charakter, mohou pracovat s celou populací, resp. jen s některými výskyty (alespoň s jediným). Závažné jsou zejména tři operace:

- zřízení entity ... (operace CREATE nebo INSERT),
- naplnění entity – vytvoření prvotní populace ... (operace FULFIL),
- zrušení entity – zrušení celé populace, zrušení celé datové struktury ... (operace DELETE).

Množina Ω je obecně členěna na dvě třídy operací:

- **konstruktory** ... mění stavy entity (patří sem i CREATE, FULFIL a DELETE)
- **selektory** ... zpřístupňují celou populaci, výskyty, resp. atributy konkrétního výskytu a nemění stav entity, dávají výsledek (ano, ne) pro dotazy na vlastnosti entity (je prázdná populace?, je v populaci jistý výskyt?, je atribut jisté hodnoty?, ...).

Funkce, z nichž je reakce-transakce na událost **u** vytvořena, mohou převádět entitu **e** ze stavu do stavu. Začíná se stavem výchozím-vstupním pro tuto transakci a končí se stavem výstupním, kdy daná entita opouští událost **u** a její transakci. Některé stavy působí tedy v roli předchůdců, jiné následovníků. Entita, jak bylo již dříve naznačeno, může být zpracovávána funkcemi v transakcích ke dvěma odlišným událostem. Jedna událost přivede entitu do cílového stavu a ten může být vstupním pro transakci k jiné události. Funkce, patřící do Ω, jimiž se entity zpracovávají, mají jednu z následujících charakteristik.

Funkce **f** může být:

- aktualizační funkcí (mění hodnoty atributů) ... konstruktor měnící stav entity,
- výpočtovou funkcí měnící jeden z atributů ... konstruktor měnící stav entity,
- výběrovou funkcí ... je to selektor neměnící stav entity,
- rozhodovací funkcí ... je to predikát a způsobuje větvení v grafech řízení transakcí – viz v dalším textu, využívá stav entity.

Z charakteristiky funkcí se potvrzuje, že aktualizační a výpočtové funkce jsou především tvůrci historií stavů entity. Potom je ke každé historii stavů přiřazen řetězec funkcí, např. $f_0f_1f_2...f_n$, který vlastně historii generuje. V uvedeném řetězci se mohou vyskytovat funkce všech typů, určující jsou ale konstruktory. Ostatní vedou na jednoduché smyčky v daných uzlech.

Na základě složitého predikátového výrazu můžeme zaznamenat skutečnost, že po stavu σ, provedeli se nad entitou **e** funkce **f** typu konstruktor, následuje v historii stav σ′ anebo jeden ze stavů z množiny $\{\sigma_1',\sigma_2',...,\sigma_n'\}$ (Rábová, 2005).

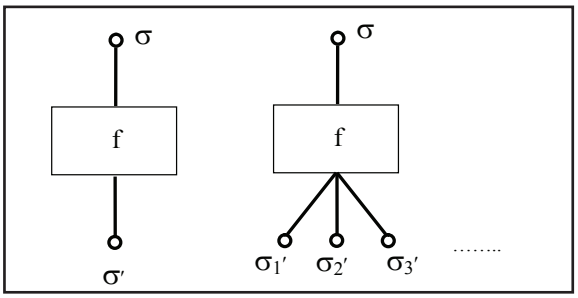
Predikátové – stavové výrazy mohou mít následující notaci:

$$\sigma_v=\sigma \Rightarrow^f \sigma^v \in \{\sigma_1',\sigma_2',...,\sigma_n'\}$$
$$\sigma_v=\sigma \Rightarrow^f \sigma^v=\sigma' \quad \text{... (1)}$$

Zápis můžeme číst takto:

„Jestliže je vstupní stav σ_v entity **e** roven stavu σ a provede se funkce **f**, potom to implikuje, že výstupní stav σ^v entity je buď roven stavu σ' anebo je jedním ze stavů z množiny $\{\sigma_1',\sigma_2',...,\sigma_n'\}$“.

To je situace, kdy vstupní stav je jediný, a kterou můžeme ilustrovat obrázkem č. 1.

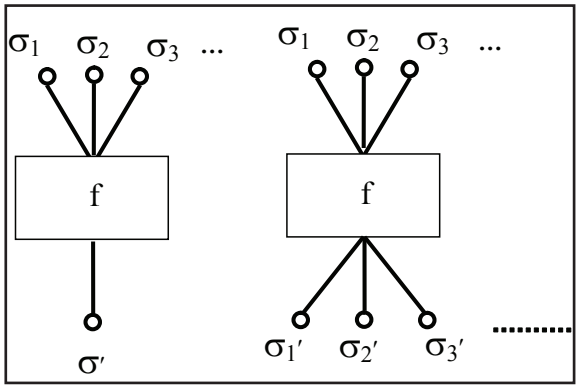


1: *Ilustrace deterministického výchozího stavu entity* ***e***

Vedle svázání stavů podle (1) mohou existovat i obecnější případy, kdy vstupní stav σ_v můžeme vybrat z množiny stavů $\{\sigma_1,\sigma_2,...,\sigma_n\}$. Tomu budou odpovídat stavové predikáty typu

$$\sigma_v \in \{\sigma_1,\sigma_2,...,\sigma_m\} \Rightarrow^f \sigma^v = \sigma'$$
$$\sigma_v \in \{\sigma_1,\sigma_2,...,\sigma_m\} \Rightarrow^f \sigma^v \in \{\sigma_1',\sigma_2',...,\sigma_n'\} \quad \text{... (2)}$$

kde m, n jsou přirozená čísla. Situaci vystihuje obr. č. 2.



2: *Ilustrace nedeterministického výchozího stavu entity* ***e***

Lze tedy říct, že stavové predikáty (1) + (2) vlastně vytváří formální popis grafu stavů pro danou entitu **e**.

Pochopitelně pod σ, σ', $\sigma_1',\sigma_2',...,\sigma_n'$, $\sigma_1',\sigma_2',...,\sigma_m'$ rozumíme predikátové proměnné, jejichž hodnotami jsou konkrétní stavy zadané alespoň výrokovými formami s individuálními proměnnými.

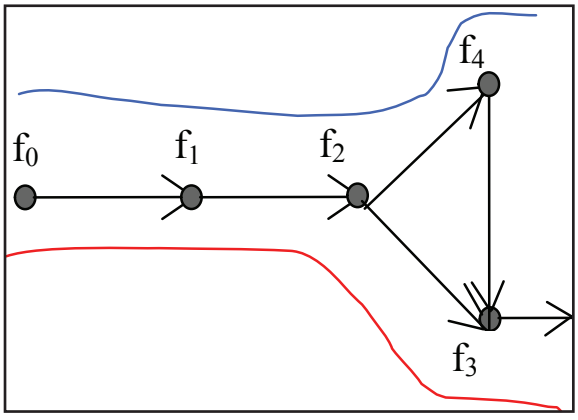
Ke každé historii stavů pak existuje posloupnost interpretovaných výrazů typu (1) + (2) představující pravdivé výrazy, když za σ, σ', $\sigma_1',\sigma_2',...,\sigma_n'$, $\sigma_1',\sigma_2',...,\sigma_m'$ jsou dosazeny konkrétní stavy entity **e**. Predikátové výrazy typu (1) + (2) jsou tedy prostředkem plně zachycujícím dvě skutečnosti:

1. složení celé historie stavů (předchůdce, následník jako uzly),
2. tvar řetězce $f_0f_1f_2...f_n$, který náleží k dané historii stavů a který je tzv. přípustným řetězcem.

Tím se pomocí interpretovaných výrazů typu (1) + (2) získává celková představa o detailním životním cyklu entity.

2. Funkční korektnost DFD diagramů

V této kapitole zavedeme základní pojmy pro korektnost DFD postavených na principu řízení provádění procesů (tedy obsahují alespoň jeden řídící proces) a fragmentaci podle událostí. Každý takový fragment DFD se dá snadno převést do transakční podoby (procesy se rozloží na transakce, úložiště se napojí na transakce, transakce se rozloží na funkce) a potom do tzv. grafu $\mathbf{G_u}$ řízení pro danou událost **u,** viz obr. č. 3.



3: *Příklad možné transformace fragmentu události na graf řízení* $\boldsymbol{G_u}$

Dále zavedeme pojmy cesta, realizovatelnost cesty, funkční korektnost cesty a funkční korektnost DFD diagramu (Mišovič, Rábová; 2005). Než k tomu ale přikročíme, uveďme několik postřehů z praktické projekční činnosti a závěrů z její diskuse.

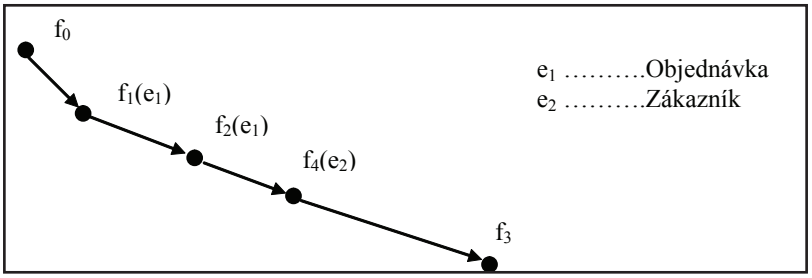
Již pro DFD běžných transakčních systémů je počet různých spojení - cest v grafu řízení značně velký. Nikterak nelze podceňovat schopnosti zkušeného analytika - projektanta. Je však jisté, že plnou pozornost ověření těchto cest nemůže věnovat. Jde nejen o množství, ale i značnou pracnost vlastního ověření vybrané cesty. Praxe potvrzuje, že bývá vcelku prověřeno asi 40 % cest. Závady ostatních se odhalí později a oprava může vyvolat velké náklady.

Přikročme teď k zavedení základních pojmů na úrovni transakcí a jejich členění na funkce, což je materiál, do kterého je původní událostní fragment DFD zjemněn.

Definice 2:

Cestou v transakci **t** k události **u** nazveme posloupnost $(f_0,f_1,f_2,...,f_k)$, kde $\mathbf{f_0}$ je zahajovací funkce asociovaná s událostí **u**, $\mathbf{f_k}$ je ukončovací funkce a pro i = 0,1,2,...,k–1 platí, že existují potenciální přechody $\mathbf{f_i \rightarrow f_{i+1}}$

Cesty se označují malými řeckými písmeny a budeme používat zápis ve tvaru $\alpha = (f_0,f_1,f_2,...,f_k)$.



4: *Cesta $\alpha_1 = (f_0,f_1,f_2,f_4,f_3)$ pro událost „Příjem objednávky“ z obr. č. 6*

Dále budeme předpokládat, že DFD je funkčně úplný vůči slovníku událostí, jestliže ke každé události existuje zahajovací funkce f_0 určující počátek reakce na událost.

V transakci z obrázku č. 6 jsou pouze dvě cesty:

$$\alpha_1 = (f_0,f_1,f_2,f_4,f_3),\ \alpha_2 = (f_0,f_1,f_2,f_3).$$

Dále se mohou sledovat obecné cesty, tj. s možnými cykly anebo jen lineární úseky obecných cest. Druhý přístup je poněkud „levnější“, ale s hrozbou, že dosažené výsledky jsou jen parciální a nelze z nich vyvozovat jejich platnost pro obecné cesty. Zavedení větví naplňuje následující definice.

Definice 3:

Větví transakce **t** nazveme cestu $(f_n,f_{n+1},f_{n+2},f_{n+3}, \ldots, f_m)$, kde
$\mathbf{f_n}$ je buď zahajovací funkce události **u** nebo funkce predikát,
$\mathbf{f_m}$ je buď ukončovací funkce události **u** nebo funkce predikát,
z funkcí $f_{n+1},f_{n+2},f_{n+3},\ldots,f_{m-1}$ není žádná funkcí predikát.

Z definice 3 je zřejmé, že větve jsou lineární části grafu řízení transakce. Pochopitelně, každý takový graf má konečný počet větví. Na rozdíl od větví nemusí být počet cest konečný. Vedle acyklických je zde hrozba existence i nekonečných cyklických cest.

Použití větví dané cesty nám dá možnost z důkazů jejich parciálních vlastností vyvozovat za jistých podmínek platnost stejných vlastností pro cestu, kterou tvoří. V dalších úvahách se budeme převážně orientovat na obecné cesty a budeme chtít dosáhnout vyššího stupně pro ověřování vlastností transakcí, než jejich členěním na větve.

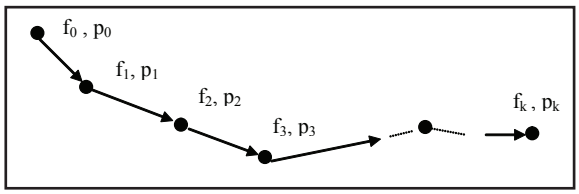
První a velmi zajímavou otázkou přístupu k transakcím je otázka **„kdy je zpracování entit ve funkci f korektní?“,** druhou zase otázka **„kdy je funkčně korektní zpracování entit v řetězci funkcí, tj. v transakci?“**. K zodpovězení využijeme přípustnost stavů, do nichž se zpracovávané entity dostanou.

Nechť jsou pro danou transakci vstupní entity z množiny $\mathbf{E_1}$ v přípustných stavech, výstupní entity z $\mathbf{E_2}$ rovněž. Nově zřízené entity v $\mathbf{E_2}$ jsou obvykle v počátečních stavech. Nabízí se podání korektnosti na základě stavů zpracovávaných entit.

Definice 4:

Zpracování vstupní entity **e** ve funkci **f** je funkčně korektní, je-li touto funkcí entita **e** převedena na výstupu z přípustného vstupního stavu opět do přípustného výstupního stavu. Cesta $\alpha = (f_0,f_1,f_2,...,f_k)$ je funkčně korektní, jestliže je funkčně korektní zpracování entit v každé její funkci. Funkčně korektní cesta je realizovatelná (tj. přechody mezi stavy entit jsou v souladu se stavovými diagramy entit).

Jestliže vezmeme v úvahu výsledky analýzy stavů v životním cyklu entit, můžeme hovořit o tom, že ke každému uzlu-funkci $\mathbf{f}_i$ transakční cesty α můžeme přidat vlastně predikátový výraz $\mathbf{p}_i$, který na základě stavů zachytí kvalitu zpracování entit. Pro jednoduchost předpokládejme, že funkce zpracovává jen jednu entitu. Predikáty $\mathbf{p}_i$ musíme vhodně interpretovat na základě povahy funkcí $\mathbf{f}_i$, tj. definici predikátu postavíme na tom, je-li funkce konstruktorem, selektorem nebo dokonce predikátem.



5: *Přiřazení predikátových výrazů $\mathbf{p}_i$ jednotlivým funkcím cesty*

Obrázek 5 ukazuje, že zahajovací funkce $\mathbf{f}_0$ je spojena s predikátem $\mathbf{p}_0$. Je-li $\mathbf{f}_0$ konstruktor, tak $\mathbf{p}_0$ může vyjadřovat zavedení entity.

Definice 5:

Predikát $\mathbf{p}_i$ je definován následujícími pravidly:
- **je-li** $\mathbf{f}_i$ ...selektor, je $\mathbf{p}_i$ „prázdný“, protože $\mathbf{f}_i$ pouze zpřístupňuje stavy entit,
- **je-li** $\mathbf{f}_i$ ...predikát, je $p_i \equiv f_i$, je to „hlídač“ výběru potenciálního přechodu k další funkci,
- **je-li** $\mathbf{f}_i$ ...konstruktor, vyjadřuje $\mathbf{p}_i$ kvalitu zpracování a je to konkrétní interpretace výrazů typu (1), (2).

Všimněme si teď nejdříve predikátů $p_i \equiv f_i$. Tyto predikáty představují řízení potenciálních přechodů v grafu řízení dané transakce. Má-li se projít po cestě α, musí v ní být takové predikáty které to zabezpečí, potom jistě musí existovat stavy entit, které tyto predikáty převádí na pravdivé výrazy, tedy cesta je realizovatelná – proveditelná.

Tvrzení 1:

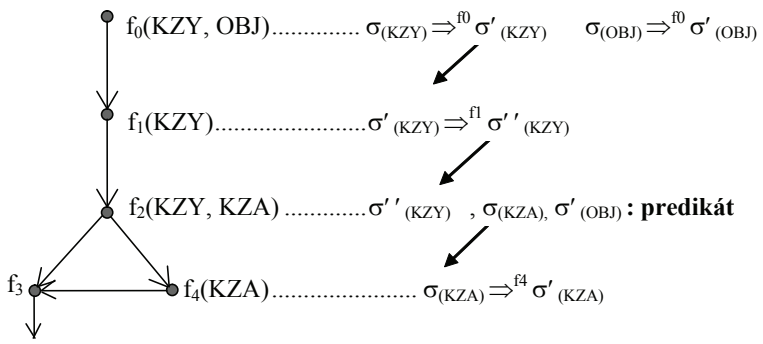
Nechť U je množinou všech událostí pro DFD diagram. Cesta $\alpha = (f_0,f_1,...,f_k)$ je funkčně korektní tehdy a jen tehdy, jestliže
1) nastala událost $\mathbf{u} \in \mathbf{U}$ asociovaná s funkcí $\mathbf{f}_0$ a
2) systém $\mathbf{N}_\alpha (p_i)$, i = 0,1,... složený z predikátů p_i má řešení.

Důkaz tohoto tvrzení spočívá v důkazu platnosti nutné a postačující podmínky tvrzení. Např. „cesta α je funkčně korektní“ je postačující k tomu, aby platilo 1) a 2). Důkaz se dá provést na základě Definic 4 a 5.

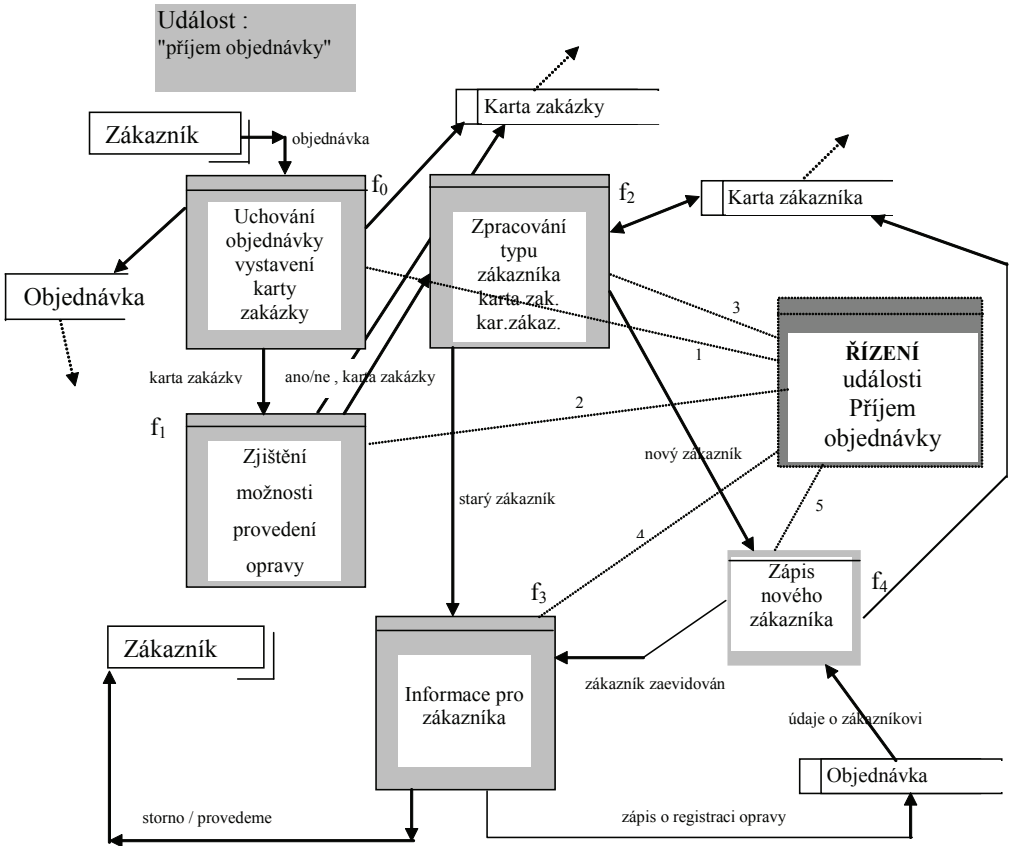
Slovo „řešení“ je zde chápáno ve smyslu, že existují takové stavy entit, aby predikáty $\mathbf{p}_i$ po jejich dosazení byly pravdivými výrazy.

Příklad 1:

Uveďme fragment DFD – transakci zobrazující činnosti nutné ke zpracování zakázky pro opravu elektrického spotřebiče jako elementární funkce. Událostí je **„Příjem objednávky“** na opravu porouchaného přístroje. Zákazníkem vyplněný list OBJEDNÁVKA obsahuje základní údaje o zákazníkovi (např. jméno, příjmení, rod. číslo, adresu,,…) a zákazníkem charakterizovanou závadu přístroje. OBJEDNÁVKA je vyhotovena s kopií. Na originálu se zákazníkovi potvrdí příjem a kopie se ponechá v opravně. Přijímací technik po prohlédnutí přístroje stanoví číslo objednávky a vyhotoví KARTU ZAKÁZKY, která doprovází přístroj do dílny. KARTA ZÁKÁZKY obsahuje nejen číslo objednávky, ale také identifikaci přístroje, datum příjmu, datum dokončení opravy, datum převzetí, cena, opravář, popis práce,… Jelikož opravna chce odměnit věrné zákazníky přidělením bonusu, vede KARTU ZÁKAZNÍKA. Zde je vedle položek křestní, příjmení, rodné číslo, adresa vedena i celková platba a bonus.



7: *Graf řízení pro fragment transakce události „Příjem Objednávky“*



6: *Fragment transakce události „Příjem Objednávky“*

Charakteristika jednotlivých funkcí zmíněné transakce je následující:

f_0 .. **je konstruktor**, protože provádí aktualizaci entity KARTA ZAKÁZKY… (KZY), jelikož zapíše její nový výskyt, podobně ukládá nový výskyt entity OBJEDNÁVKA

f_1 .. **je konstruktor**, protože zpracovává nový výskyt entity KARTA ZAKÁZKY a aktualizuje ji tím, že do tohoto výskytu zapíše proveditelnost opravy, opraváře, pravděpodobnou cenu, …..

$\mathbf{f_2}$ .. **je predikát**, protože vezme identifikaci zákazníka (např. údaj „křestní+příjmení +rod.číslo“) z nového výskytu entity OBJEDNÁVKA přes KARTA ZAKÁZKY a podle této identifikace prohledá výskyty entity KARTA ZÁKAZNÍKA… (KZA). Najde-li zákazníka, tj. stejnou identifikaci, rozhodne se pro přechod na $\mathbf{f_3}$, jinak na $\mathbf{f_4}$,

f_3 .. nezpracovává entitu, je to ukončovací funkce transakce,

f_4 .. **je konstruktor**, aktualizuje entitu KARTA ZÁKAZNÍKA, protože připisuje nový výskyt. Potřebné hodnoty může převzít z aktuálního výskytu OBJEDNÁVKA.

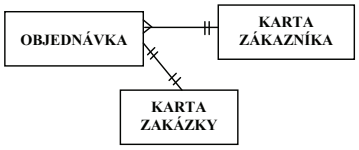
V uvedeném fragmentu jsou dvě cesty:

$\alpha_1 = (\mathbf{f_0,f_1,f_2,f_3})$
$\alpha_2 = (\mathbf{f_0,f_1,f_2,f_4,f_3})$

Zabývejme se cestou α_2. Systém $\mathbf{N_{\alpha 2}(p_i)}$ může být naformulován např. takto:

$\sigma_{(KZY)} \Rightarrow^{f0} \sigma'_{(KZY)}$
$\sigma_{(OBJ)} \Rightarrow^{f0} \sigma'_{(OBJ)}$
$\sigma'_{(KZY)} \Rightarrow^{f1} \sigma'_{(KZY)}$
{je stav $\sigma_{(KZA)}$ takový, že ∃výskyt kde (**ident_zákazníka**$_{KZA}$ = **ident_zákazníka**$_{OBJ}$ aktuálního výskytu) **?**}$^-$.
$\sigma_{(KZA)} \Rightarrow^{f4} \sigma'_{(KZA)}$

Řešením systému $\mathbf{N_{\alpha 2}(p_i)}$ by měly být hodnoty pro následující stavy (stav je vícerozměrná veličina viz Definice 1):



$\sigma_{(KZY)}, \sigma'_{(KZY)}, \sigma''_{(KZY)}$	pro KARTA ZAKÁZKY
$\sigma_{(OBJ)}, \sigma'_{(OBJ)}$	OBJEDNÁVKA
$\sigma_{(KZA)}, \sigma'_{(KZA)}$	KARTA ZÁKAZNÍKA

Systém $\mathbf{N_{\alpha 2}(p_i)}$ řešíme tak, že postupně hledáme stavy zpracovávaných entit pro $\mathbf{p_0}$, které ho převádí

na „pravdu“, potom pro p_1, p_2, … Zmíněný systém $\mathbf{N_{\alpha 2}(p_i)}$ bude mít obecně nekonečně mnoho řešení, protože má 7 neznámých stavů entit a pouze 5 predikátů, které je zpracovávají.

Abychom snadno získali výsledky, zřejmě položíme za konstantní tyto stavy:

$\sigma_{(KZY)}$, $\sigma_{(OBJ)}$, $\sigma_{(KZA)}$ a zbývající, tj. stavy $\sigma'_{(KZY)}$, $\sigma'_{(OBJ)}$, $\sigma''_{(KZY)}$, $\sigma'_{(KZA)}$ vypočteme řešením příslušných predikátů ze systému $\mathbf{N_{\alpha 2}(p_i)}$.

Nekonečnost řešení bude asi obecnou vlastností systémů podobných systému $\mathbf{N_{\alpha 2}(p_i)}$. Laicky řečeno:

po cestě α_2 se projde např. tehdy, jestliže sdělená **ident_zákazníka** přišlým zákazníkem odpovídá **ident_zákazníka** jedné uložené kartě zákazníka. Výsledky dále ukazují, že cestou se zpracovává konečný počet entit, pro které jsou rozhodující stavy, v nichž se nachází.

Teď je již snadné najít takové stavy, které vyhovují systému $\mathbf{N_{\alpha 2}(p_i)}$.

f_0 …….konstruktor

$\sigma_{(OBJ)}$ vstupní

číslo objednávky	zákazník			adresa		identif. přístroje	datum	
	křestní	příjmení	rod. číslo	město	ulice-č.		příjmu	opravy
123-1	jan	krk	42-06-22/125	brno	kounicová 45	vysavač-ls	1.12.1993	15.15.1993

$\sigma'_{(OBJ)}$ výstupní po zápisu výskytu č. 123-2

číslo objednávky	zákazník			adresa		identif. přístroje	datum	
	křestní	příjmení	rod. číslo	město	ulice-č.		příjmu	opravy
123-1	jan	krk	42-06-22/125	brno	kounicová 45	vysavač-ls	1.12.1993	15.15.1993
123-2	**ivan**	**hopjan**	**74-05-12/452**	**třebíč**	**horkého 5**	**vysoušeč**	**15.12.1993**	**18.12.1993**

$\sigma_{(KZY)}$ vstupní

číslo objednávky	identif. přístroje	opravář	oprava ano/ne	cena	popis práce	datum		
						příjem	oprava	převzetí
123-1	vysavač-ls	kovový	ano	450	…….	1.12.1993	15.15.1993	16.12.1993

$\sigma'_{(KZY)}$ výstupní po zápisu nového výskytu

číslo objednávky	identif. přístroje	opravář	oprava ano/ne	cena	popis práce	datum		
						příjem	oprava	převzetí
123-1	vysavač-ls	kovový	ano	450	…….	1.12.1993	15.15.1993	16.12.1993
123-2	**vysoušeč**					**15.12.1993**	**18.12.1993**	

f_1 …….konstruktor

$\sigma'_{(KZY)}$ vstupní

číslo objednávky	identif. přístroje	opravář	oprava ano/ne	cena	popis práce	datum		
						příjem	oprava	převzetí
123-1	vysavač-ls	kovový	ano	450	…….	1.12.1993	15.15.1993	16.12.1993
123-2	vysoušeč					15.12.1993	18.12.1993	

$\sigma''_{(KZY)}$ výstupní po zápisu dodatečných údajů opravář, proveditelnost, pravděp. cena

číslo objednávky	identif. přístroje	opravář	oprava ano/ne	cena	popis práce	datum		
						příjem	opravy	převzetí
123-1	vysavač-ls	kovový	ano	450	…….	1.12.1993	15.15.1993	16.12.1993
123-2	vysoušeč	**janoušek**	**ano**	**320**	…….	15.12.1993	18.12.1993	

f_2…….predikát

$\sigma''_{(KZY)}$ vstupní

číslo objednávky	identif. přístroje	opravář	oprava ano/ne	cena	popis práce	datum		
						příjem	opravy	převzetí
123-1	vysavač-ls	kovový	ano	450	…….	1.12.1993	15.15.1993	16.12.1993
123-2	vysoušeč	janoušek	ano	320	…….	15.12.1993	18.12.1993	

$\sigma_{(KZA)}$ vstupní

zákazník			adresa		platby	bonus
křestní	příjmení	rod. číslo	město	ulice-č.		
jan	krk	42-06-22/125	brno	kounicová 45	450	0 %

f_4 ……konstruktor

$\sigma_{(KZA)}$ vstupní

zákazník			adresa		platby	bonus
křestní	příjmení	rod. číslo	město	ulice-č.		
jan	krk	42-06-22/125	brno	kounicová 45	450	0 %

$\sigma'_{(KZA)}$ výstupní, po uložení nového výskytu

zákazník			adresa		platby	bonus
křestní	příjmení	rod. číslo	město	ulice-č.		
jan	krk	42-06-22/125	brno	kounicová 45	450	0 %
ivan	**hopjan**	**74-05-12/452**	**třebíč**	**horkého 5**	**320**	**0 %**

Tvrzení 2:

Buď **e** libovolná vstupní entita zpracovávaná cestou $\alpha = (f_0, f_1,...,f_k)$, nabývající cestou historie vstupních a výstupních stavů $\sigma^{f_0}/\sigma'^{f_0}$, $\sigma^{f_1}/\sigma'^{f_1}$, ..., $\sigma^{f_k}/\sigma'^{f_k}$. Cesta α je funkčně nekorektní, jestliže tato historie není součástí žádné z historií stavů entity **e** ze životního cyklu entity.

Stavy $\sigma^{f_0}/\sigma'^{f_0}$, $\sigma^{f_1}/\sigma'^{f_1}$, ..., $\sigma^{f_k}/\sigma'^{f_k}$ nemohou být řešením systému N_α. Kdyby tomu tak bylo, potom je α funkčně korektní a historie stavů je ze životního cyklu zpracovávané entity.

Definice 6:

Diagram toků a zpracování dat je funkčně korektní, když
a) je funkčně úplný vůči slovníku událostí,
b) transakce každé události mají všechny své cesty realizovatelné
c) a každá transakce provádí to, co analytik zamýšlel.

Tato definice představuje závěrečný pohled na funkční korektnost DFD diagramu, tedy i funkční analýzy. Provést ale takovou prověrku celého DFD není snadná záležitost, protože bod c) definice je vágní.

Kardinální potíže jsou rovněž v metodách řešení systémů predikátů. Obecné metody, např. jedna uveřejněná v (Manna, 1974), nedosahují úspěchu. Na druhé straně to, že necháme systém řešit analytikem – projektantem, bez výrazné podpory počítače, rovněž nevede k úspěchu. Chce-li analytik prověřit všechny cesty, musí mít nejdříve jejich seznam, poté musí získat systémy $N_\alpha(p_i)$, ty postupně řešit a to vše mu může usnadnit počítač.

3. Konstrukce testovací báze dat pro PIS

Chce-li analytik testovat výsledky vyjádřené v DFD diagramu, musí mít přijatelnou testovací množinu M instancí entit – testovací bázi dat. Za přijatelnou můžeme považovat takovou množinu M, která je úplná, tj. že jsou jejími prvky realizovatelné všechny cesty cílového DFD diagramu. Taková množina je dána definicí č. 7.

Definice 7:

Buď M_i souhrn testovacích instancí entit a jejich stavů pro cesty fragmentu $F_i \in$ DFD diagramu strukturovaného na fragmenty podle událostí, m přirozené číslo – počet událostí – počet fragmentů.

Množina $M = \bigcup_{i=1}^{m} M_i$ je úplnou testovací bází dat pro DFD, jestliže platí:
1. každý souhrn $\mathbf{M_i}$ realizuje všechny funkční cesty α^i_j z fragmentu $\mathbf{F_i}$,
2. jestliže souhrn $h \not\subset M$ realizuje cestu $\alpha \in$ DFD, potom musí existovat souhrn $h' \subset M$, který rovněž realizuje cestu α.

Na základě definice 7 můžeme vyslovit pro nalezení úplné testovací báze dat algoritmus $\mathcal{A}$, který pracuje pro každý fragment $\mathbf{F_i}$ tak, že pro jeho událost $\mathbf{u_i}$ a její graf řízení vypočte souhrn $\mathbf{M_i}$ testovacích entit.

Algoritmus $\mathcal{A}$:

1. Najde všechny cesty v grafu řízení události u_i, každou cyklickou cestu převede na konečný počet reprezentantů (odstranění cyklických cest).
2. Ke každé cestě α grafu řízení $\mathbf{G_i}$, náležícího k fragmentu $\mathbf{F_i}$ události $\mathbf{u_i}$ algoritmus najde systém $\mathbf{N}_\alpha$ a jeho řešení $\mathbf{M}_\alpha$. Potom $M_i = \bigcup_{\alpha \ni G_i} M_\alpha$ je dílčí testovací báze dat pro fragment $\mathbf{F_i}$.

Příklad 2:

Sestavme množinu $M_1 = M_{\alpha 1} \cup M_{\alpha 2}$ pro cesty z příkladu 1 pro událost „Příjem objednávky“, která je prvním fragmentem kompaktního DFD. Fragment má dvě cesty: $\alpha_1 = (f_0,f_1,f_2,f_3)$, $\alpha_2 = (f_0,f_1,f_2,f_4,f_3)$. Pro druhou z nich již máme

$M_{\alpha 2}$ = {OBJEDNÁVKA, KARTA ZAKÁZKY, KARTA ZÁKAZNÍKA se stavy:}
$\sigma_{(KZY)}, \sigma'_{(KZY)}, \sigma''_{(KZY)}, \sigma_{(OBJ)}, \sigma'_{(OBJ)}, \sigma_{(KZA)}, \sigma'_{(KZA)}$

Pro cestu α_1 teď vypočteme $M_{\alpha 1}$**.** Vyjdeme ze systému $\mathbf{N}_{\alpha 1}$, který má následující tvar:

$\sigma_{(KZY)} \Rightarrow^{f0} \sigma'_{(KZY)}$
$\sigma_{(OBJ)} \Rightarrow^{f0} \sigma'_{(OBJ)}$
$\sigma'_{(KZY)} \Rightarrow^{f1} \sigma''_{(KZY)}$
{je stav $\sigma_{(KZA)}$ takový, že ∃ výskyt, kde (**ident_zákazníka**$_{KZA}$ = **ident_zákazníka**$_{KZY}$ aktuálního výskytu) **?**}$^+$.

Můžeme tedy za řešení považovat tyto stavy zpracovávaných entit:

f_0 …….konstruktor

$\sigma_{(OBJ)}$ vstupní

číslo objednávky	zákazník			adresa		identif. přístroje	datum	
	křestní	příjmení	rod. číslo	město	ulice-č.		příjmu	opravy
123-1	jan	Krk	42-06-22/125	brno	kounicová 45	vysavač-ls	1.12.1993	15.15.1993
123-0	Tomáš	Velký	74-04-15/188	hodonín	smutného 13	hol.strojek	11.11.1993	25.11.1993

$\sigma'_{(OBJ)}$ výstupní po zápisu výskytu č. 123-2

číslo objednávky	zákazník			adresa		identif. přístroje	datum	
	křestní	příjmení	rod. číslo	město	ulice-č.		příjmu	opravy
123-0	Tomáš	Velký	74-04-15/188	hodonín	smutného 13	hol.strojek	11.11.1993	25.11.1993
123-1	jan	krk	42-06-22/125	brno	kounicová 45	vysavač-ls	1.12.1993	15.15.1993
123-2	Tomáš	Velký	74-04-15/188	hodonín	smutného 13	**vysoušeč**	**15.12.1993**	**18.12.1993**

$\sigma_{(KZY)}$ vstupní

číslo objednávky	identif. přístroje	opravář	oprava ano/ne	cena	popis práce	datum		
						příjem	oprava	převzetí
123-1	vysavač-ls	kovový	ano	450	…….	1.12.1993	15.15.1993	16.12.1993
123-0	hol.strojek	ondryhal	ano	200	…….	11.11.1993	24.11.1993	27.11.1993

$\sigma'_{(KZY)}$ výstupní po zápisu nového výskytu

číslo objednávky	identif. přístroje	opravář	oprava ano/ne	cena	popis práce	datum		
						příjem	oprava	převzetí
123-0	hol.strojek	ondryhal	ano	200	…….	11.11.1993	24.11.1993	27.11.1993
123-1	vysavač-ls	kovový	ano	450	…….	1.12.1993	15.15.1993	16.12.1993
123-2	**vysoušeč**					**15.12.1993**	**18.12.1993**	

f_1 …….konstruktor

$\sigma'_{(KZY)}$ vstupní

číslo objednávky	identif. přístroje	opravář	oprava ano/ne	cena	popis práce	datum		
						příjem	oprava	převzetí
123-0	hol.strojek	ondryhal	ano	200	…….	11.11.1993	24.11.1993	27.11.1993
123-1	vysavač-ls	kovový	ano	450	…….	1.12.1993	15.15.1993	16.12.1993
123-2	vysoušeč					15.12.1993	18.12.1993	

$\sigma''_{(KZY)}$ výstupní po zápisu dodatečných údajů opravář, proveditelnost, pravděp. cena

číslo objednávky	identif. přístroje	opravář	oprava ano/ne	cena	popis práce	datum		
						příjem	opravy	převzetí
123-0	hol.strojek	ondryhal	ano	200	…….	11.11.1993	24.11.1993	27.11.1993
123-1	vysavač-ls	kovový	ano	450	…….	1.12.1993	15.15.1993	16.12.1993
123-2	vysoušeč	**janoušek**	**ano**	**320**	…….	15.12.1993	18.12.1993	

f_2…….predikát

$\sigma''_{(KZY)}$ vstupní

číslo objednávky	identif. přístroje	opravář	oprava ano/ne	cena	popis práce	datum		
						příjem	opravy	převzetí
123-0	hol.strojek	ondryhal	ano	200	…….	11.11.1993	24.11.1993	27.11.1993
123-1	vysavač-ls	kovový	ano	450	…….	1.12.1993	15.15.1993	16.12.1993
123-2	vysoušeč	janoušek	ano	320	…….	15.12.1993	18.12.1993	

$\sigma_{(KZA)}$ vstupní

zákazník			adresa		platby	bonus
křestní	příjmení	rod. číslo	město	ulice-č.		
Tomáš	Velký	74-04-15/188	Hodonín	Smutného 13	200	0%
jan	krk	42-06-22/125	brno	kounicová 45	450	0 %

Je tedy

$\mathbf{M}_{\alpha 1}$ = {OBJEDNÁVKA, KARTA ZAKÁZKY, KARTA ZÁKAZNÍKA se stavy:}
$\sigma_{(KZY)}, \sigma'_{(KZY)}, \sigma''_{(KZY)}, \sigma_{(OBJ)}, \sigma'_{(OBJ)}, \sigma_{(KZA)}$

Množiny $\mathbf{M}_{\alpha 1}$ a $\mathbf{M}_{\alpha 2}$ umožní analytikovi otestovat „logiku“ události, tj. zda vše pro událost **„Příjem objednávky“** probíhá tak, jak bylo původně analytikem zamýšleno.

Algoritmus pro nalezení úplné testovací báze dat je náročný na své provádění. Dá se říci, že již pro jednoduché DFD rozložené do fragmentů podle více než deset událostí, je žádoucí počítačová podpora. Ta by mohla zabezpečit nejen tvorbu grafů řízení a práci s cestami a transakcemi, ale též interaktivní způsob řešení systémů predikátů pro realizovatelnost cest.

VÝSLEDKY

Rozsáhlost diagramů ERD, DFD a ELH evokuje možnost zanechat v nich mnohé nekonzistentnosti. Užitečnou metodou pro jejich redukci jsou rozklady, které zužují záběr analýzy a umožní získávat jen fragmenty úplných diagramů. Počáteční orientace příspěvku na popis nekonzistentností mezi jednotlivými diagramy je vystřídána orientací na detailní rozbor fragmentace DFD na základě událostí, transakcí a stavů zpracovávaných entit. Zájem je soustředěn na realizovatelnost – proveditelnost transakcí, na základě realizovatelnosti jejich cest. V příkladech 1, 2 je ukázáno, jak interaktivně řešit systém $N_\alpha(p_i)$ stavových rovnic a přesvědčit se o funkční korektnosti cesty a celé transakce. Od začátku příspěvku je diskutována možnost sestrojení testovací báze dat podnikového IS a možnost počítačové podpory testování transakcí.

Zvláštností příspěvku je formalizace jednoho ze stěžejních problémů strukturované analýzy, tj. problému funkční korektnosti DFD fragmentovaného podle událostí. Praktické rozhodnutí o korektnosti je ale nakonec svěřeno analytikovi.

SOUHRN

V příspěvku je uvedena původní formalizace pojetí interaktivního stylu tvorby testovací báze dat pro podnikový IS. Definici zmíněného stylu předchází zavedení stavu datové struktury entita a rozbor elementárních funkcí, kterými je entita zpracována. Transakce, na které jsou procesy rozloženy, jsou potom

sestaveny z elementárních funkcí, které ovlivňují stav zpracovávané entity. Na základě charakteristiky vlivu funkcí na stav entit, zpracovávaných vybranou cestou v transakci, jsou formulovány stavové rovnice, které zachycují stav před a po zpracování entit elementárními funkcemi na vybrané cestě. Neznámými jsou stavy (vícerozměrné veličiny) zpracovávaných entit, které se musí najít. Systém stavových rovnic má obecně nekonečně mnoho řešení. Proto je zaveden algoritmus pro interaktivní styl nalezení řešení za přispění analytika.

testovací báze dat, událost, transakce, stavové rovnice, graf řízení události, cesta v grafu řízení události, funkční korektnost

LITERATURA

MANNA, Z.: *Mathematical Theory of Computation*. London, Mc-Graw Hill, 1974.

MARTIN, J.: *Information Engineering*. New Jersey, Prentice Hall International, 1982. ISBN 0-13-464462-X.

YOURDON, E.: *Modern Structured Analysis*. New Jersey, Prentice Hall International, Inc., 1989.

MOLNÁR, Z.: *Moderní metody řízení informačních systémů.* Praha, GRADA a. s., 1992. ISBN 80-85623-07-2.

MIŠOVIČ, M.: *New construction of Data Testing Set*. Praha, Mezin. konference RUFIS‘97, 1997. s. 107–112. ISBN 80-01-01710-9.

RÁBOVÁ, I.: *Formalizace transakcí v podnikových procesech*. Brno, Mezinárodní konference Firma a konkurenční prostředí, MZLU v Brně, Provozně ekonomická fakulta, 2005. s. 50-55. ISBN 80-7302-097-1.

MIŠOVIČ, M. – RÁBOVÁ, I.: *Konzistence podnikových procesů vyjádřená formálním aparátem transakcí*. Brno, Acta Universitatis Agriculturae et Silviculturae Mendelianae Brunensis, 2005. sv. 13, č. 3, s. 117–125. ISSN 1211-8516.

Adresa

Prof. RNDr. Milan Mišovič, CSc., Doc. Ing. Ivana Rábová, Ph.D., Ústav informatiky, Mendelova zemědělská a lesnická univerzita v Brně, Zemědělská 1, 613 00 Brno, Česká republika, e-mail: misovic@pef.mendelu.cz, rabova@pef.mendelu.cz