Nástroje pro funkční dekompozici a minispecifikaci

Data Flow Diagram - DFD

Diagram datových toků - DFD - je modelovací nástroj, který umožňuje zobrazit systém jako síť procesů, které plní určené funkce a předávají si mezi sebou data. Poskytuje funkčně orientovaný pohled na systém.

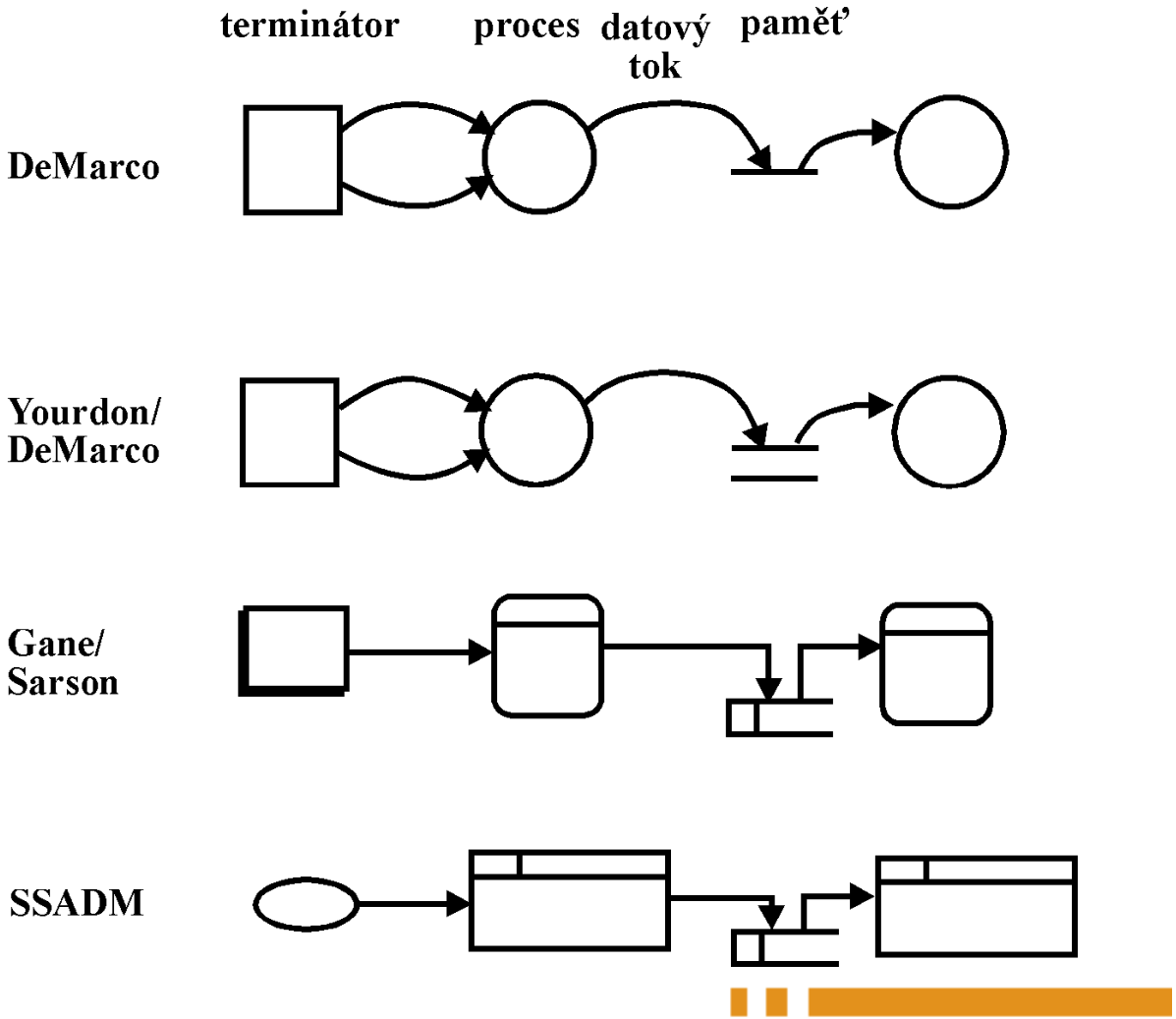
- Modely toků práce (work flow diagrams) v operačním výzkumu od začátku 20.století.
- Obrázek toho, co se tady děje.

Alternativní názvy:

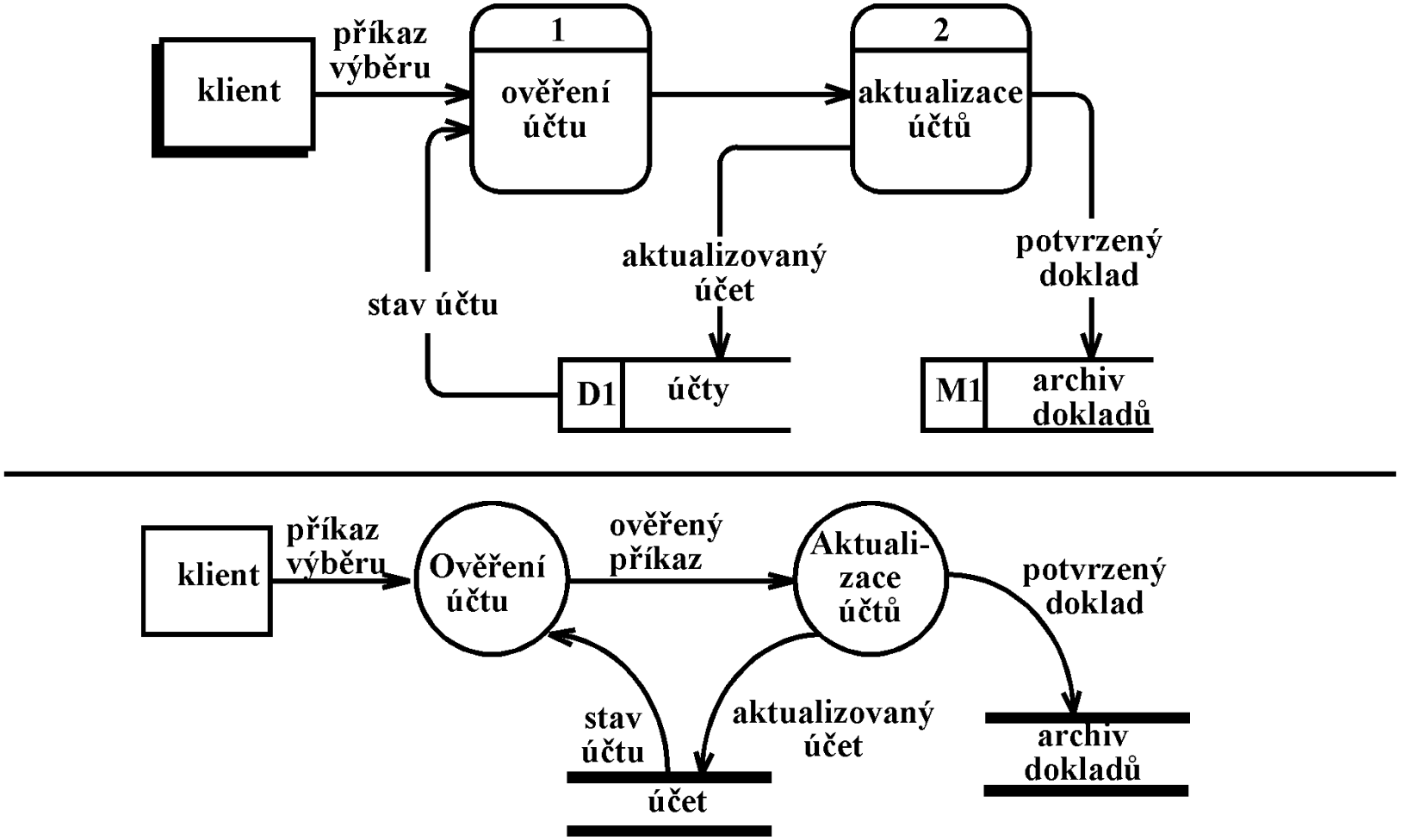
Function Model, Bubble Chart, (Yourdon, 1975),
Process Model (Gane, Sarson, 1977),
Work Flow Diagram (DeMarco, 1978)

Komponenty DFD a jejich notace

- Proces, funkce, transformace
- Paměť, datastór
- Datový tok, tok
- Terminátor, vnější entita



Příklad notací Gane/Sarson a Yourdon/DeMarco



Procesy

Proces ukazuje část systému, která transformuje určité vstupy na výstupy. Proces je pojmenován jediným slovem, frází nebo jednoduchou větou. Jméno vyjadřuje, co proces dělá.

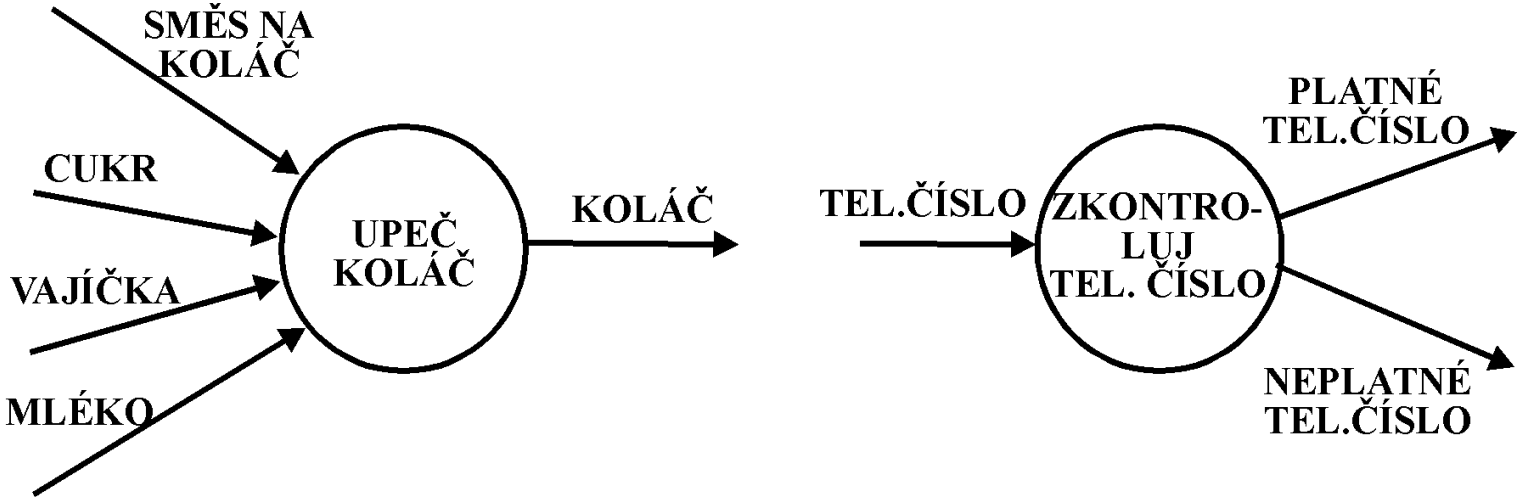
Někdy bude v názvu procesu obsaženo jméno osoby, skupiny osob, oddělení, počítače nebo mechanického zařízení. Pak jméno procesu vyjadřuje, kdo provádí nějakou činnost, transformaci dat, místo toho, aby vyjádřilo podstatu transformace.

Toky

Tok znázorňuje cestu, po které se pohybují datové shluky (informační pakety) z jedné části systému do druhé.

V některých případech vyjadřují toky pohyb fyzických materiálů. U reálných systémů mohou být na DFD současně toky, které vyjadřují pohyb materiálu a dat.

Materiálové a datové toky

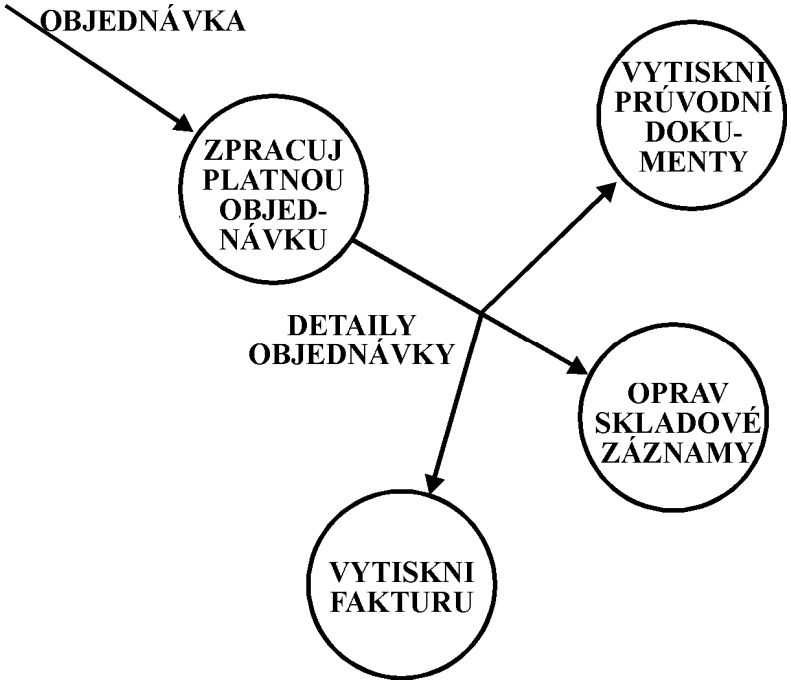


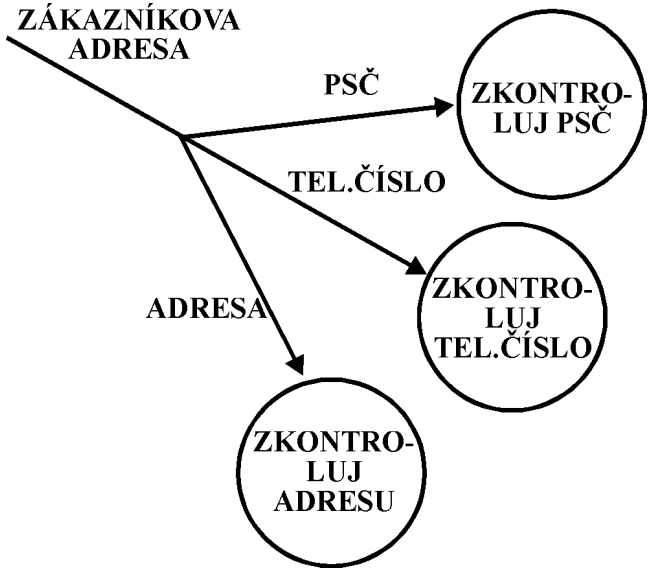
Toky jsou pojmenované. Jméno vyjadřuje význam paketu, který je po dané cestě přenášen. Tok přenáší pouze jeden typ paketu, který je určen jménem.

Paket má jiný význam, pokud putuje na různě pojmenovaných tocích. Stejná data na tocích TEL.ČÍSLO, PLATNÉ TEL.ČÍSLO a NEPLATNÉ TEL.ČÍSLO mají odlišný význam.

Divergující toky

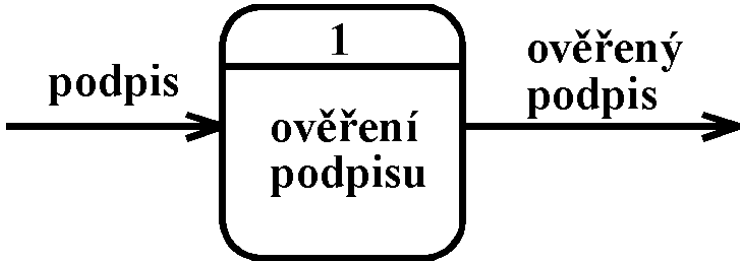
- Duplikáty stejného datového paketu jsou zaslány do různých částí.
- Datový paket je rozložen na jednodušší datové pakety, které jsou zaslány do různých částí systému.
- Na datovém toku se objevují položky s různými hodnotami, podle hodnot dochází k rozdělení.





Pojmenování datových toků

Jméno toku by mělo vyjadřovat podstatu transformace.

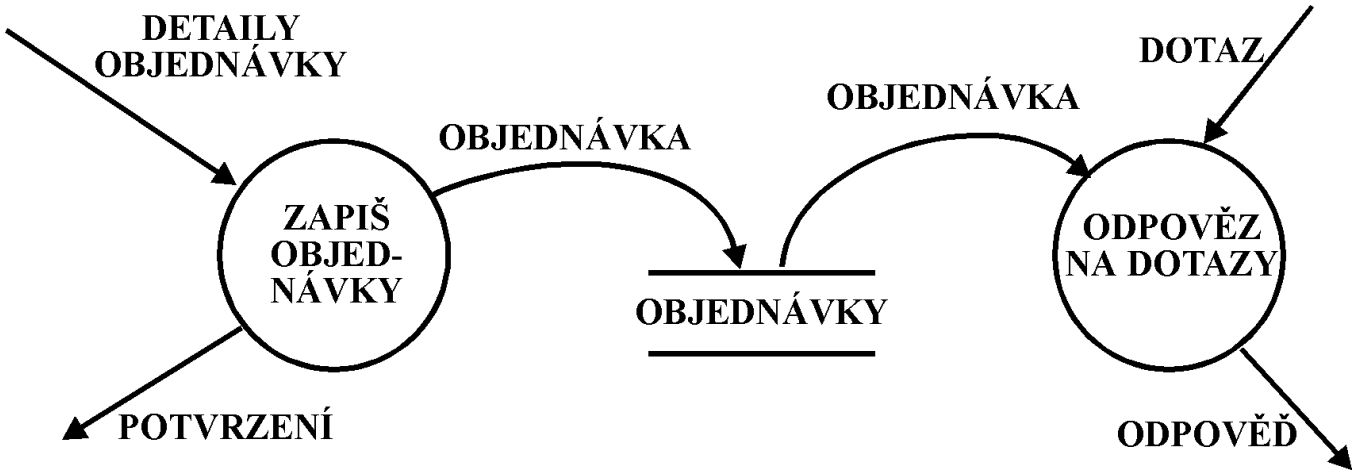


Př.: objednávka - prověřená objednávka
řetěz znaků - číslo v daném rozsahu
vyplněný formulář - dostatečně vyplněný formulář
rastrový obraz - ekvalizovaný rastrový obraz

Paměť, esenciální paměť

Paměť modeluje kolekci dat v klidu. Jméno je voleno obvykle jako množné číslo jména, kterým jsou označeny pakety na tocích vedoucích do a z paměti.

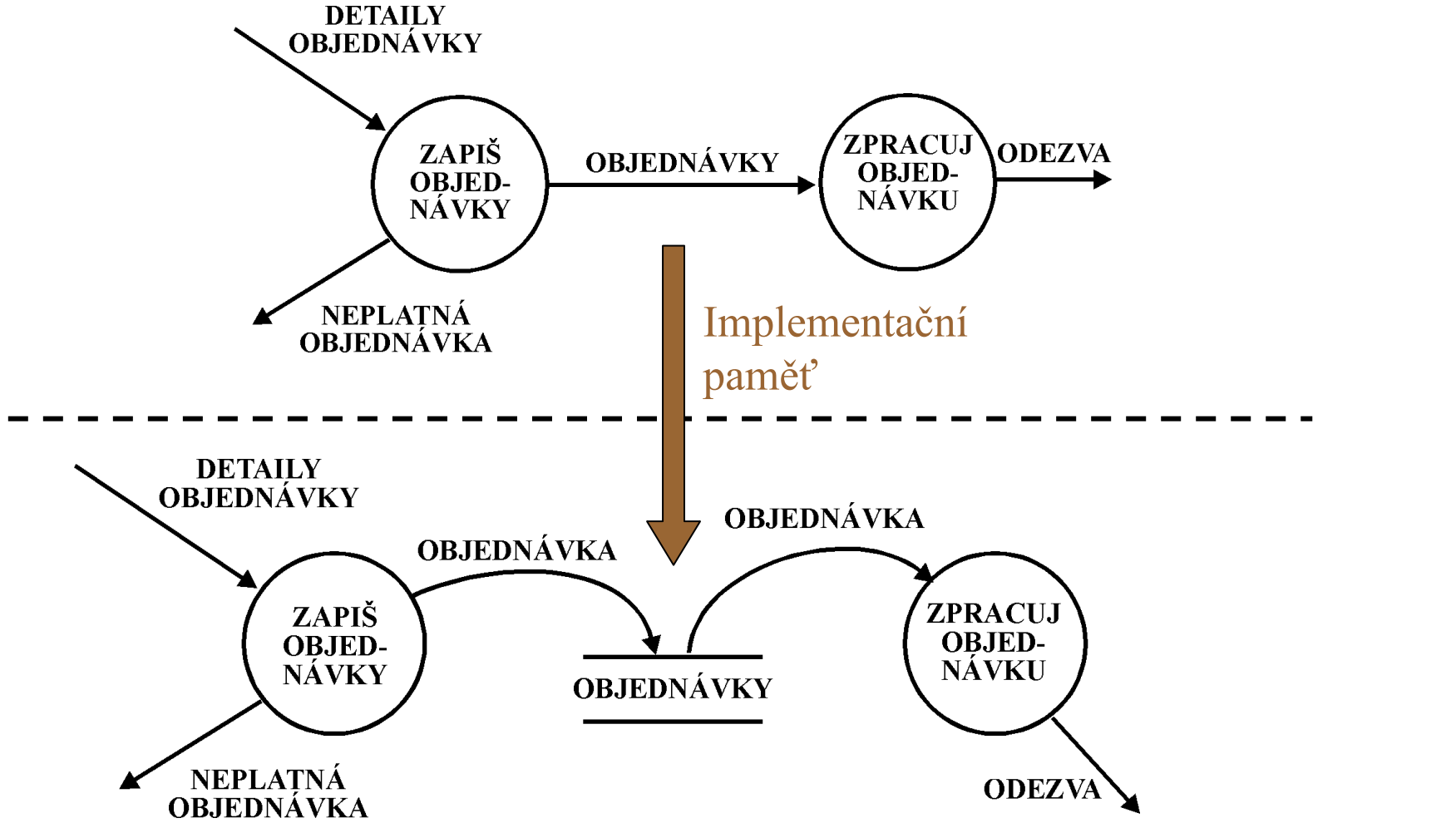
Esenciální paměť - data předávaná mezi dvěma a více procesy pracujícími v různém čase.



Vlastnosti paměti

- Paměť je pasivní částí systému, data nejsou přenášena do/z paměti, pokud o to proces explicitně nepožádá.
- Čtení je nedestruktivní, tj. paměť se nemění, když paket s informací putuje po výstupním toku z paměti.
- Tok vedoucí do paměti může mít význam zápisu, změny nebo zrušení. Může vyjadřovat následující situace:
  - Jeden nebo více nových paketů je přidáno do paměti; na konec nebo někam mezi existující pakety.
  - Jeden nebo více paketů je zrušeno, přemístěno z paměti.
  - Jeden nebo více paketů jsou přeměněny; to může znamenat změnu celého paketu nebo (častěji) části paketu, nebo obdobných částí více paketů.

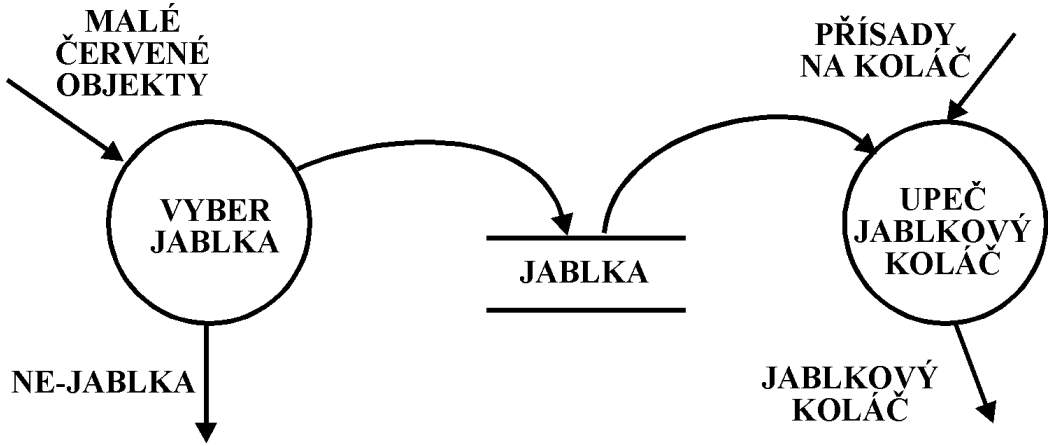
Implementační paměť



Důvody pro implementační paměť

- Oba procesy poběží na stejném počítači, ale není k dispozici dostatek paměti (nebo jiný HW prostředek), kterou by oba procesy použily ve stejném čase.
- Jeden nebo oba procesy budou řešeny na technickém vybavení, které není dostatečně spolehlivé. Paměť slouží jako prostředek pro uchování dosud zpracovaných částí dat a zvyšuje bezpečnost systému.
- Oba procesy budou implementovány různými programátory. Paměť pak slouží jako rozhraní mezi dvěma nezávisle řešenými subsystémy.
- Systémový analytik předvídá, že paměť bude v budoucnosti použita pro další, dosud nespecifikované funkce.

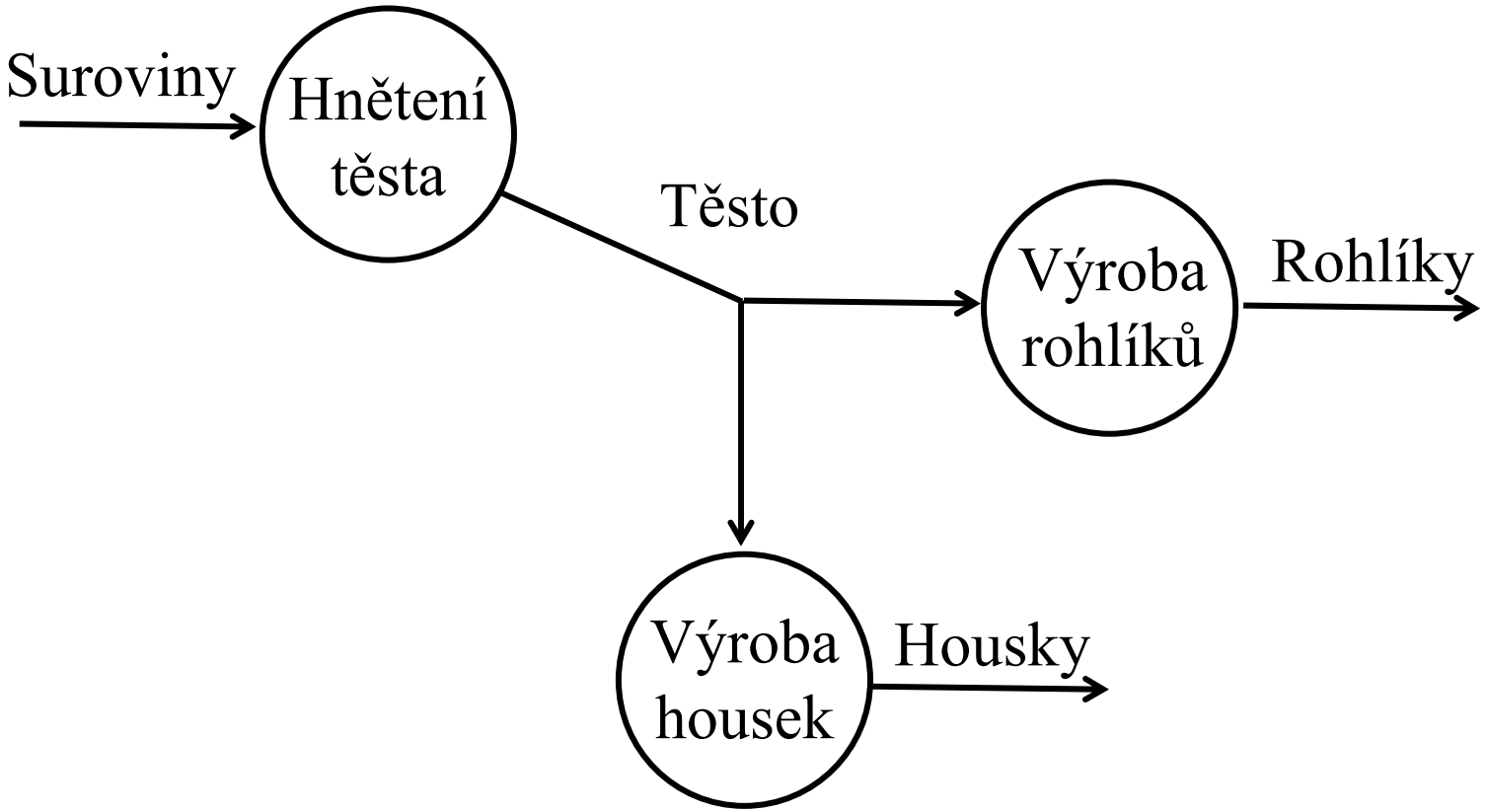
Paměti s nepojmenovanými toky



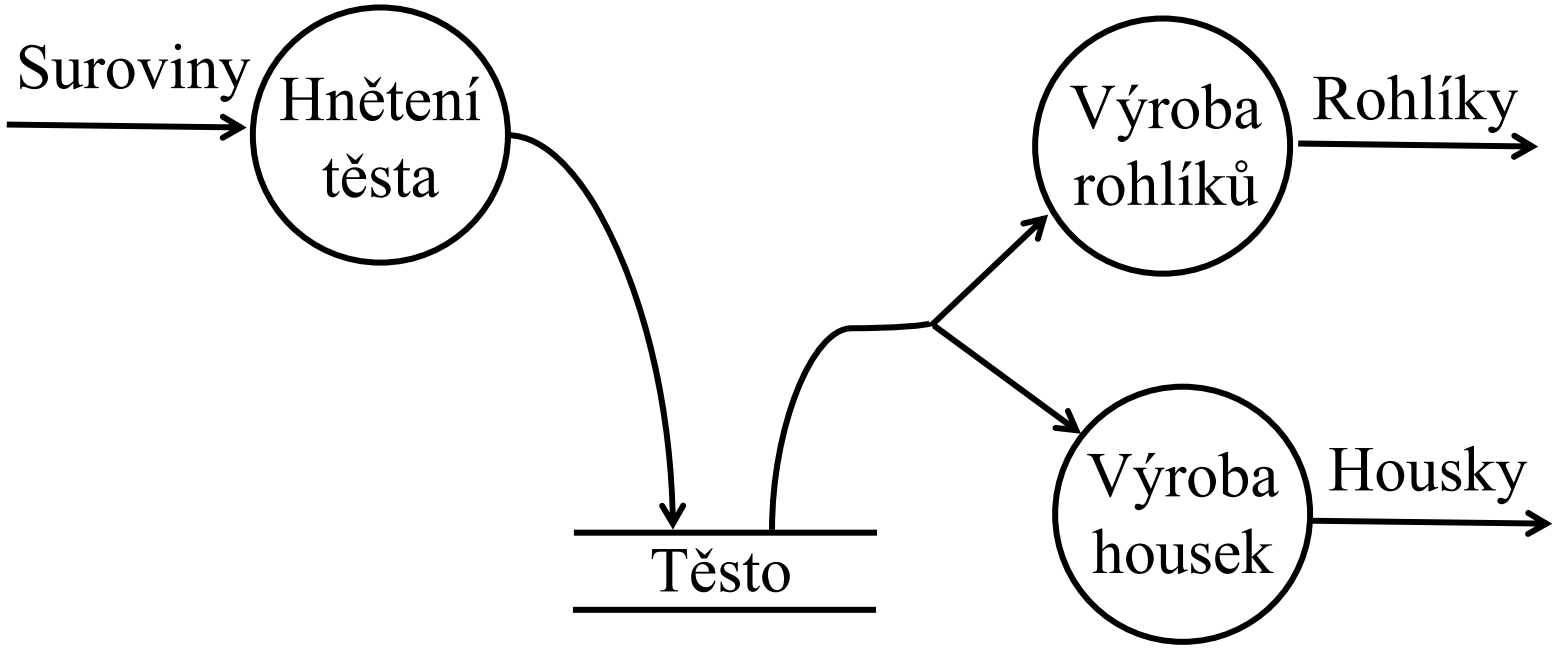
Z analytického hlediska je nepodstatné, zda vyjadřujeme situaci, kdy:

- jediný datový paket je čten z paměti
- více než jeden datový paket je získán z paměti
- je čtena pouze část paketu
- jsou čteny části více než jednoho paketu

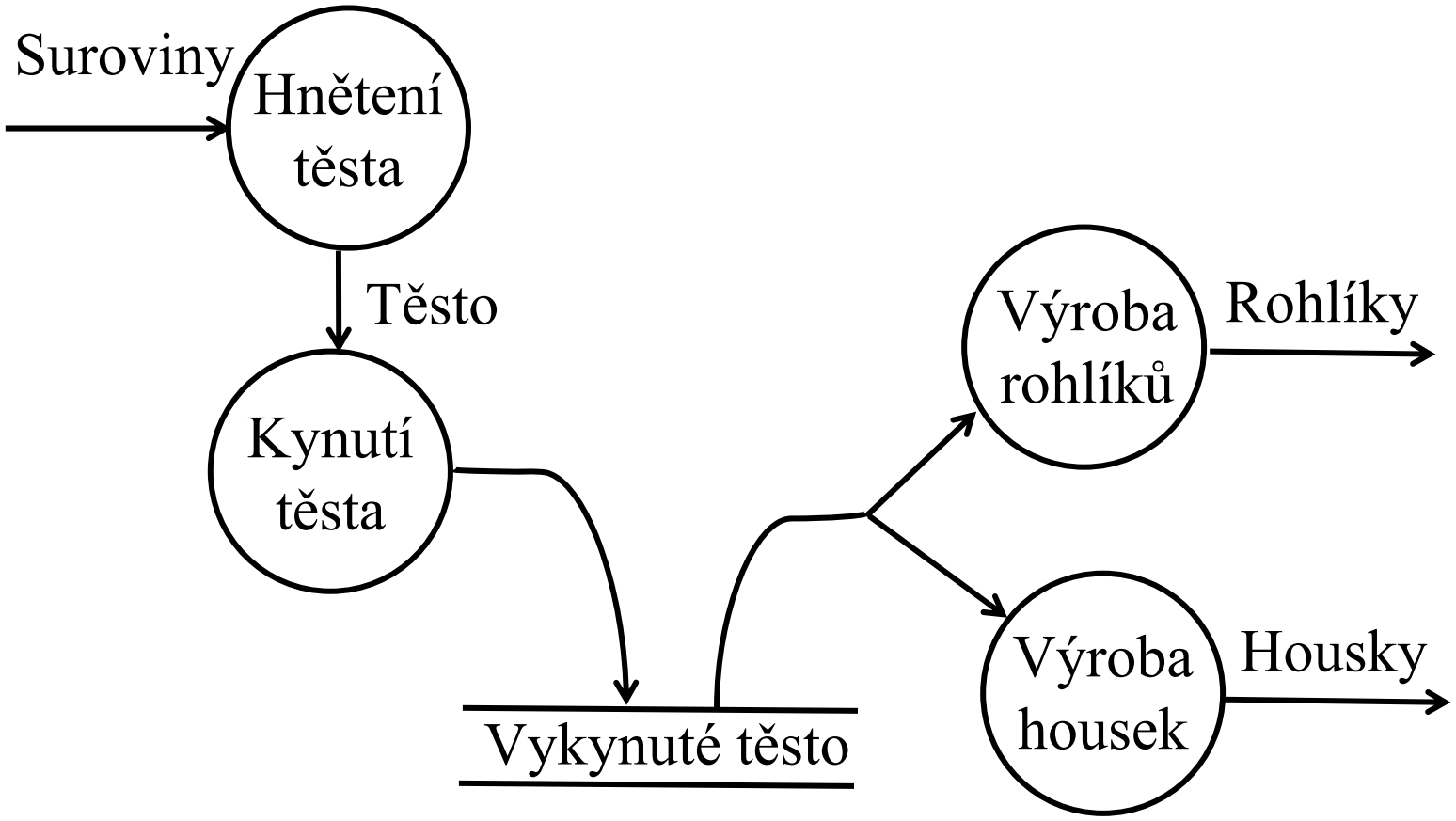
Příklad - Pekárna



Příklad - Pekárna



Příklad - Pekárna



Terminátor

Terminátor reprezentuje externí entity, se kterými systém komunikuje.

Terminátory jsou vně modelovaného systému, toky, které je propojují s procesy (nebo pamětmi) v systému, reprezentují rozhraní mezi systémem a vnějším světem.

Terminátor

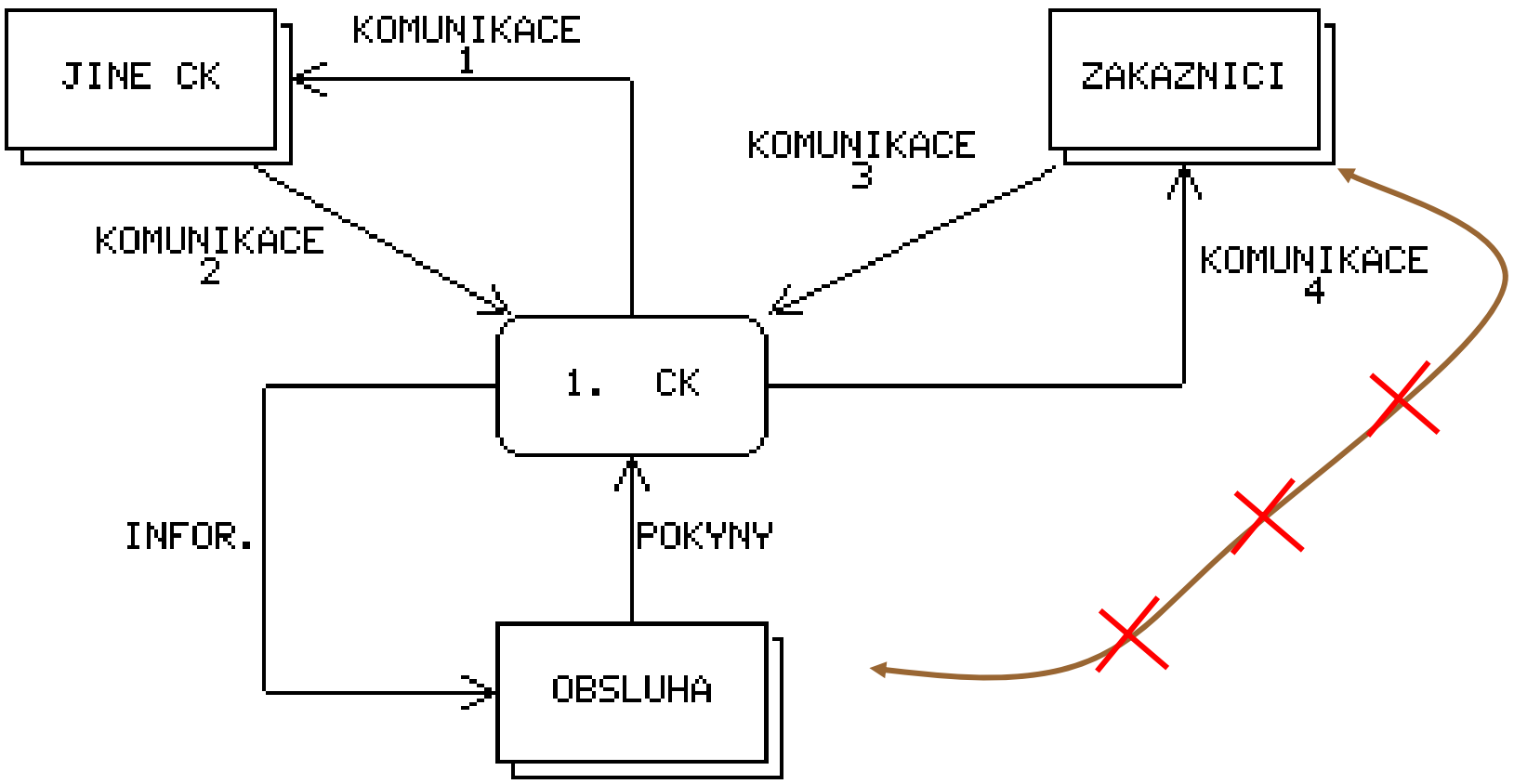
Systémový analytik ani návrhář nemohou změnit obsah terminátoru nebo způsob, jakým pracuje.

⇩

Žádný vztah mezi terminátory nebude ukázán na DFD.

Existující vztahy nejsou součástí studovaného systému. Pokud je nutné tyto vztahy zaznamenat při definici uživatelských požadavků, pak je terminátor ve skutečnosti částí systému a měl by být modelován jako proces.

Příklad - Cestovní kancelář



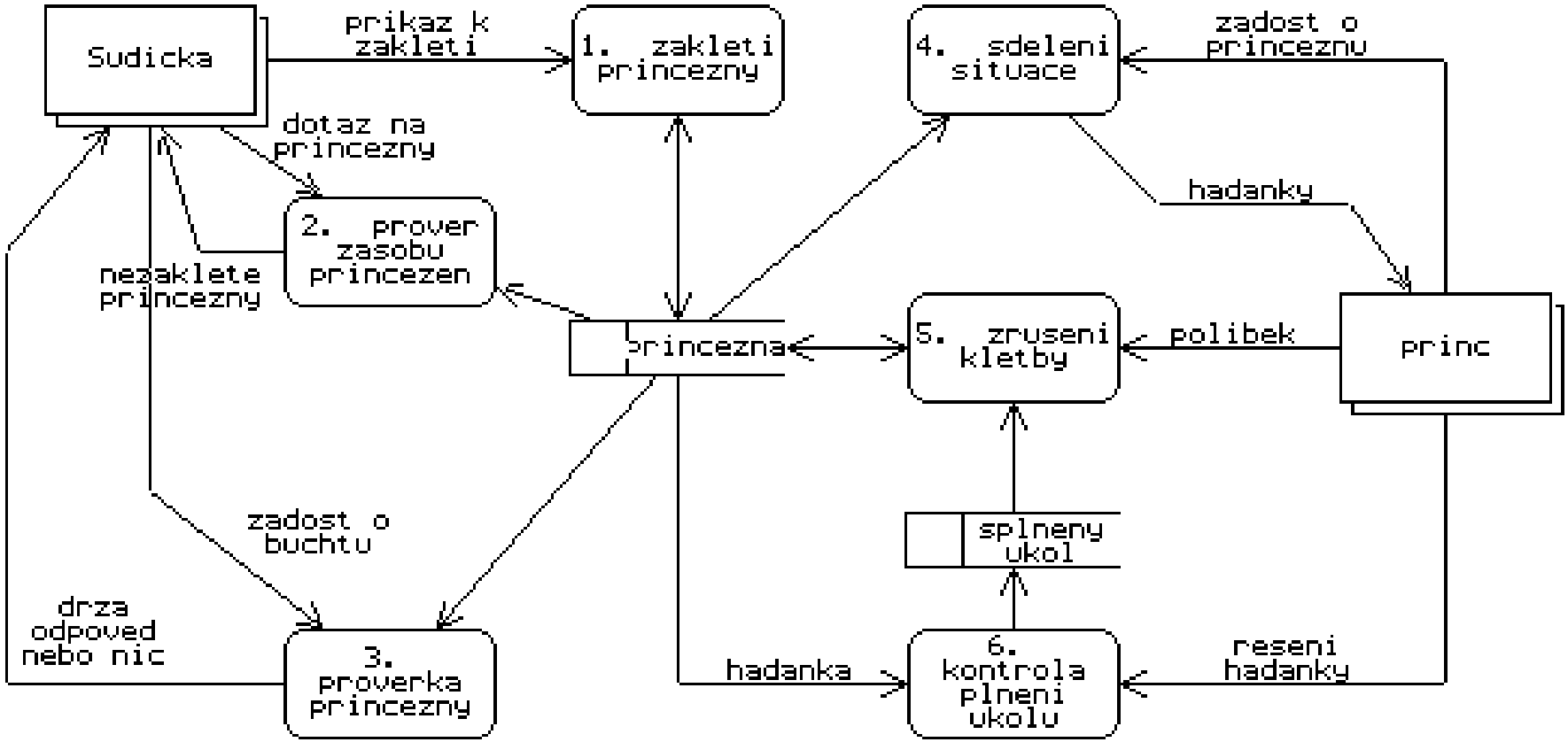
Příklad - Univerzální pohádka

Sudička si prověřuje, co je princezna zač. Obvykle požádá princeznu o něco k snědku. Po drzé odpovědi je princezna zakleta s podmínkou.

Princ, kterému rodiče přikázali, ať se bez nevěsty nevrací, se dotáže na princeznu. Je mu vysvětleno, že princezna je k mání, ovšem za určitých podmínek.

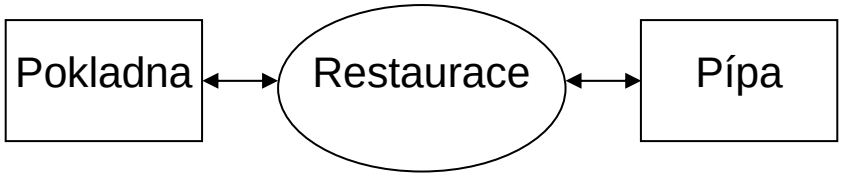
Princ začne řešit úkoly (např. odpovídá na hádanky). Správné odpovědi jsou evidovány. Pokud se pokusí osvobodit princeznu polibkem, je nejprve ověřeno splnění úkolů. Pak je princezna zbavena kletby a uvolněna pro sňatek. Princ odjíždí ...

Příklad - Univerzální pohádka

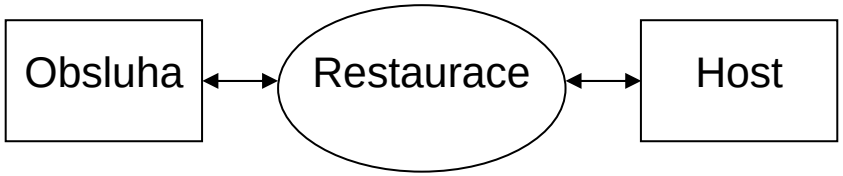


Příklad - Restaurace

Terminátory

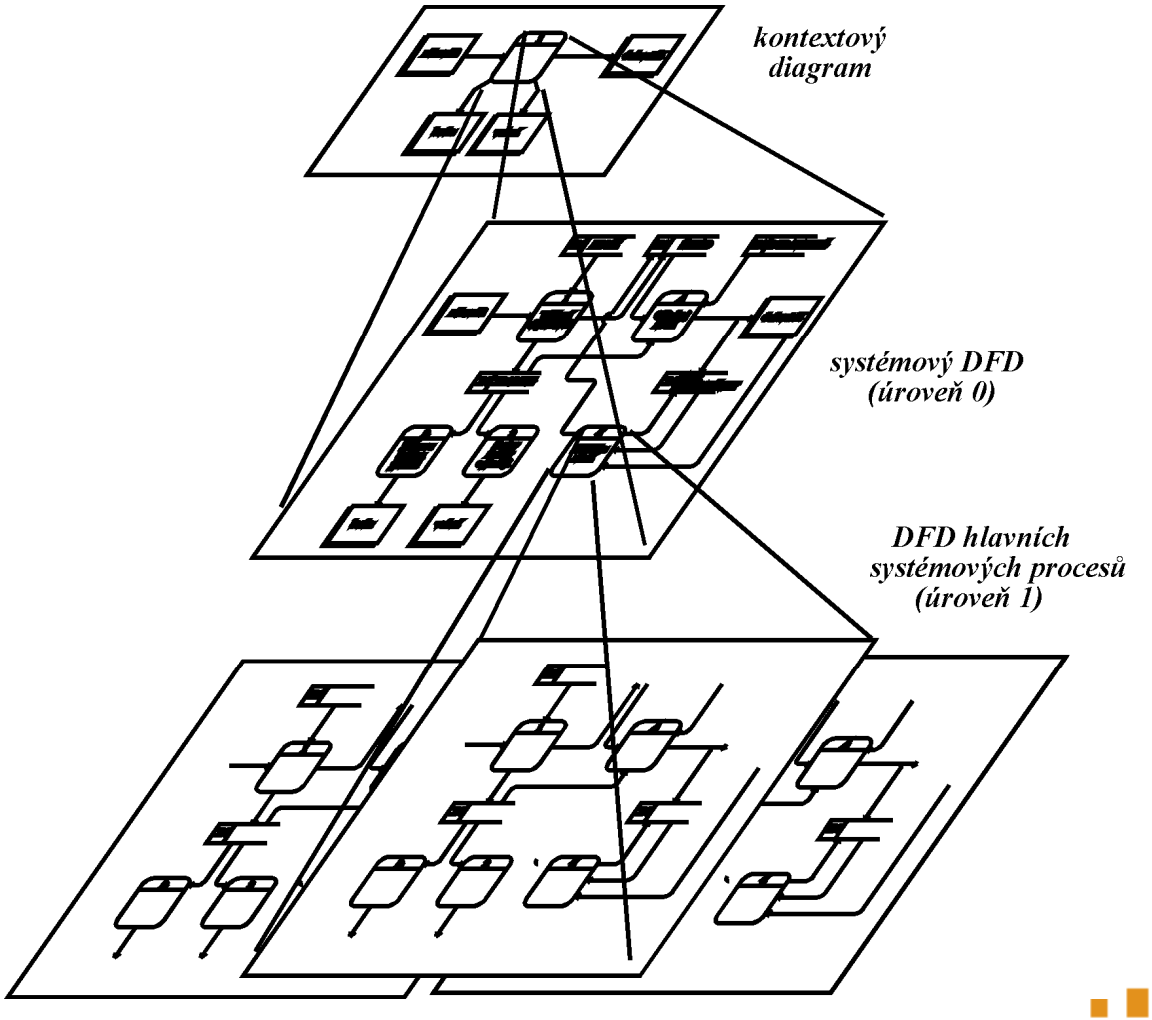


nebo?



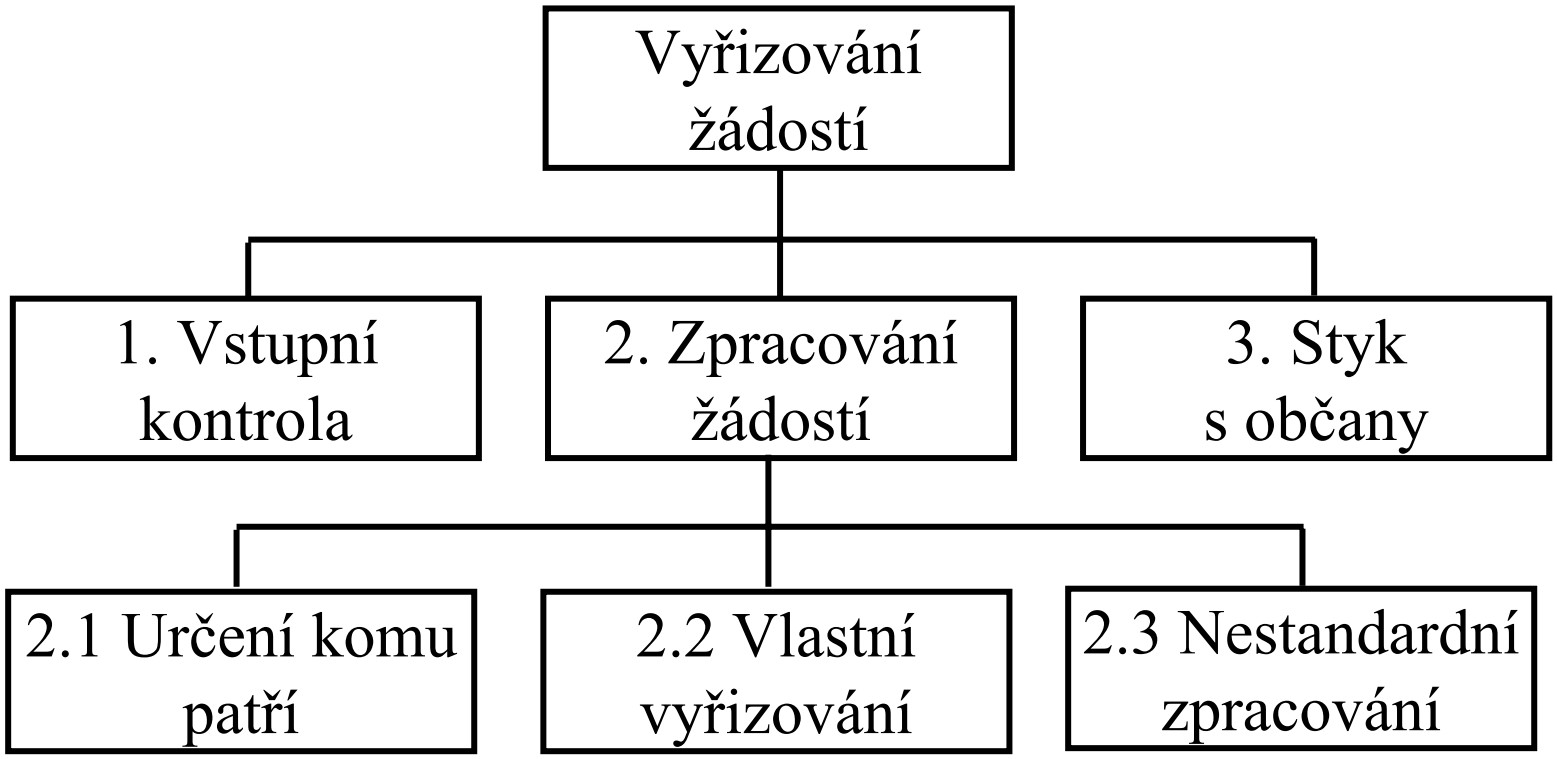
Jak modelovat hosty?

Hierarchie DFD



Příklad - Dekompozice zpracování žádosti

Diagram hierarchie procesů

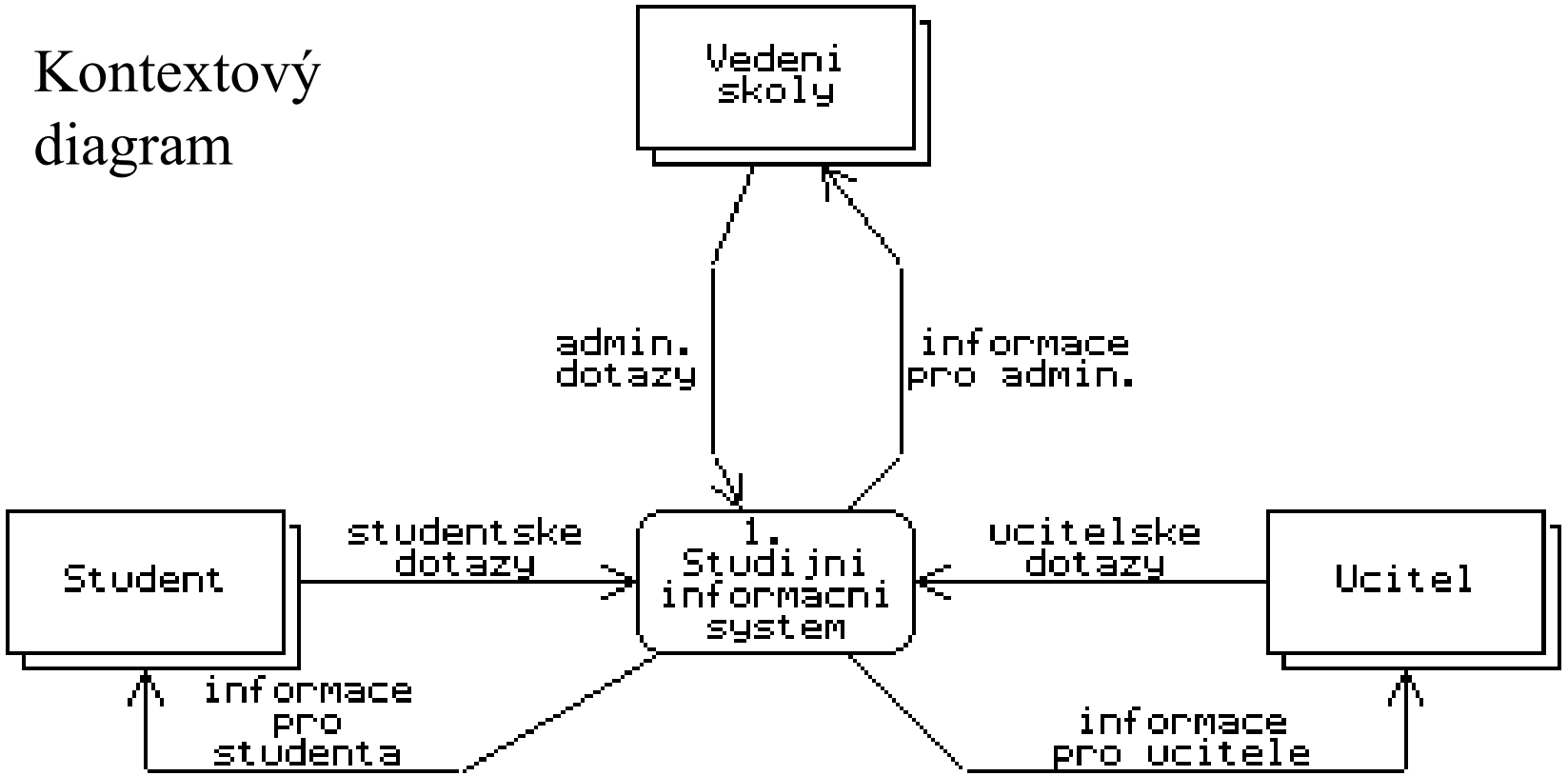


Doporučení při tvorbě DFD

- Volit výstižná jména pro procesy, datové toky, paměti a terminátory.
- Systematicky číslovat procesy.
- Překreslovat diagramy tak, aby byly co nejestetičtější.
- Vyhnout se příliš jednoduchým nebo příliš složitým DFD.
- Ověřit vnitřní i externí konzistenci DFD.

Příklad - Studijní IS

Kontextový diagram



Vnitřní konzistence DFD

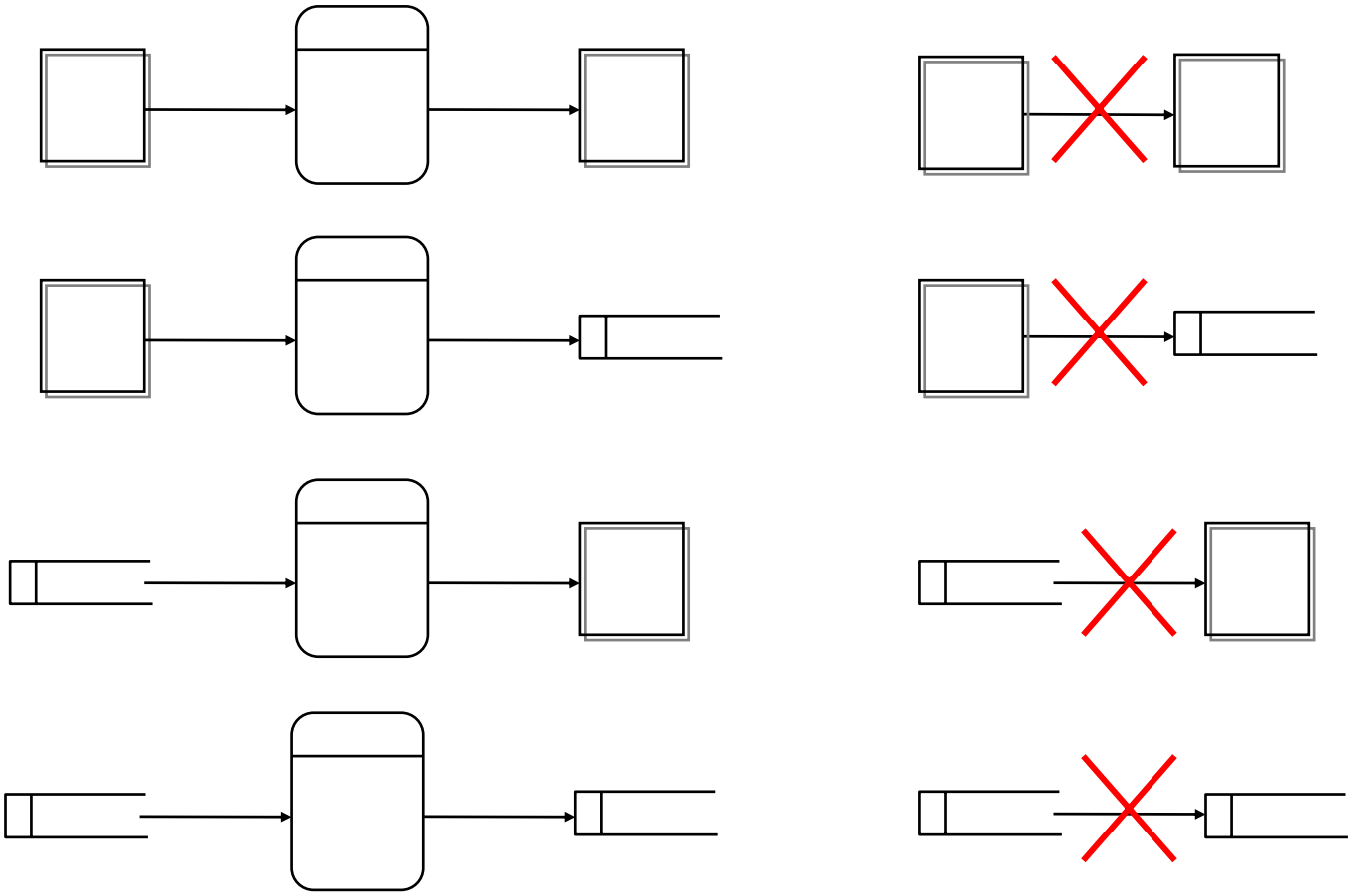
- Prověření a odstranění „černých děr“, tj. procesů, které mají pouze vstupy a neprodukují žádná data. (skryté rozhraní, ukrytý terminátor)

- Prověření a odstranění „bílých trpaslíků“, tj.procesů, které mají pouze výstupy (skryté DB rozhraní, ukrytý terminátor).

Vnitřní konzistence DFD

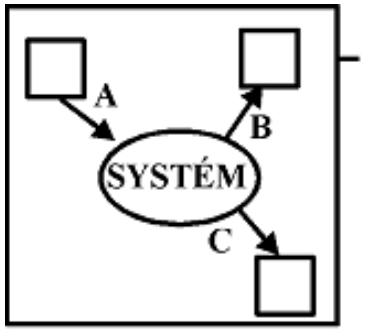
- Je nutné ověřit neoznačené toky a procesy.
  Podezření:
  - *U nepojmenovaného datového toku byly seskupeny náhodně různé skupiny dat a je obtížné je pojmenovat.*
  - *U nepojmenovaného procesu není zřejmá funkce, rozhodnutí bylo "odloženo" na pozdější dobu.*
- Prověřit paměti, které slouží výhradně pro čtení nebo výhradně pro zápis. Má smysl pouze na rozhraní mezi systémem a terminátorem (a většinou je považováno za chybu ...).

Vztahy podrobené analýze

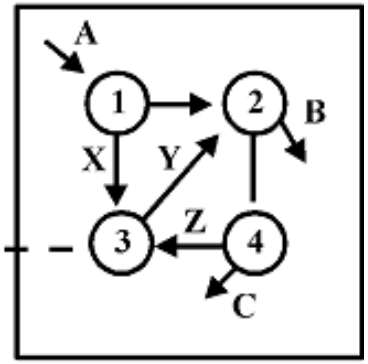


Hierarchie DFD

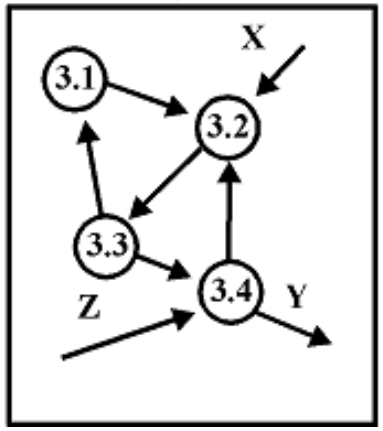
DFD 0.úrovně: pohled na hlavní funkce a na rozhraní mezi těmito funkcemi.



Kontextový diagram



DFD 0. úrovně



DFD 1. úrovně

Pravidla tvorby víceúrovňových DFD

Číslování procesu se přenáší na nižší úrovně. Má-li proces číslo *n*, jsou procesy o úroveň níže číslovány *n.1*, *n.2*, ...

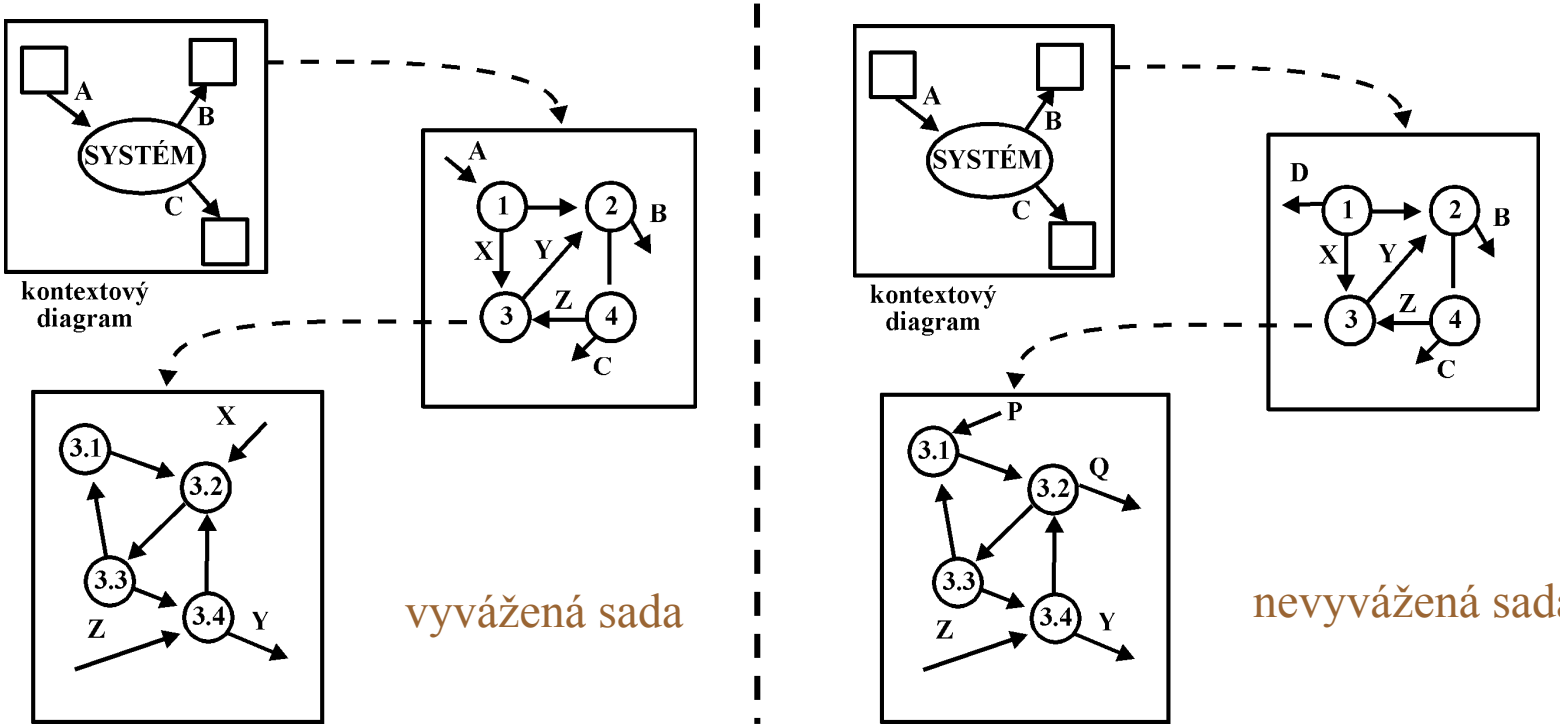
Jméno procesu se stává jménem DFD na nižší úrovni.

Počet úrovní v sadě DFD je volen tak, aby na DFD nebylo mnoho procesů a pamětí. Doporučuje se tvořit DFD na A4 s průměrně 5-7 procesy a pamětmi.

Systém střední velikosti bude mít cca 3-6 úrovní DFD. Počet musí být zvládnutelný a udržovatelný.

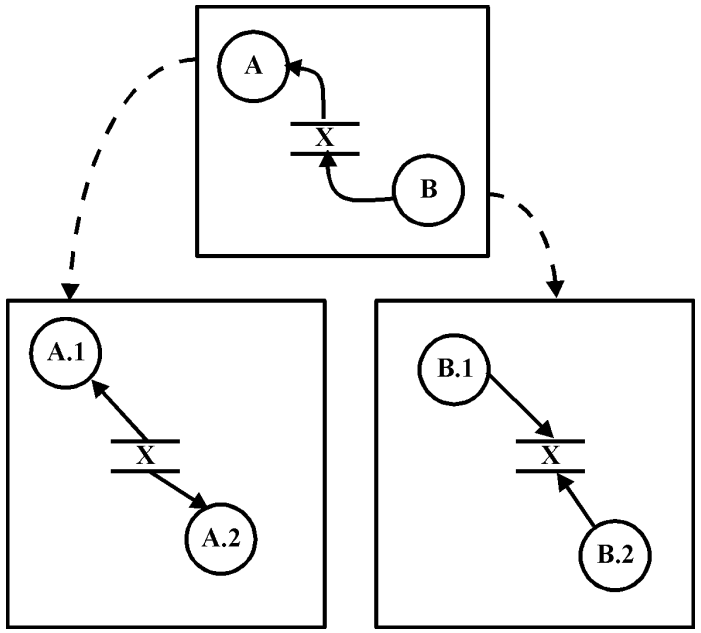
Pravidla tvorby víceúrovňových DFD

Pro zajištění konzistence na jednotlivých úrovních DFD musí souhlasit vstupní a výstupní datové toky u procesů a jim odpovídajících DFD na nižších úrovních.



Pravidla tvorby víceúrovňových DFD

Paměť zakreslíme poprvé na úrovni, kde je použita jako rozhraní mezi dvěma a více procesy. Na nižších úrovních ji zopakujeme u všech DFD, které obsahují dekompozici procesů spolupracujících s pamětí na vyšší úrovni.



Minispecifikace

Minispecifikace popisuje (definuje) logiku procesů. Je nástrojem analýzy a návrhu a je konzultována se zákazníkem.

- Pro každý proces na nejnižší úrovni rozkladu DFD musíme vytvořit právě jednu minispecifikaci.
- Minispecifikace musí popsat pravidla transformace datových toků, které vstupují do procesu, na výstupní toky.
- Minispecifikace musí popsat postupy a pravidla, kterými se řídí transformace, nikoliv však implementaci těchto pravidel.
- Do specifikačního dokumentu nesmí minispecifikace vnášet redundanci žádného druhu.

Strukturovaná angličtina (čeština)

Strukturovaná angličtina je běžný jazyk, z něhož vynecháme:

- Rozvité přívlastky, zbytečná přídavná jména
- Složité větné konstrukce
- Všechny módy kromě imperativu
- Všechny větné konstrukce mimo omezené množiny podmíněných a logických příkazů
- Většinu interpunkčních znamének (otazníky, čárky, aj.)
- Vysvětlení vně průběžného textu (poznámky pod čarou)

Strukturovaná angličtina

Slovník je složen z

- imperativních sloves,
- pojmů definovaných v Datovém slovníku,
- rezervovaných slov pro formulaci logiky.

Syntaxe je omezena na

- jednoduché oznamovací věty,
- uzavřené rozhodovací konstrukce,
- uzavřené opakovací konstrukce.

Strukturovaná angličtina

IF <podmínka>,
THEN
<činnost pro platnou podmínku>.
OTHERWISE
<činnost, když podmínka neplatí>.

- -

SELECT :
CASE 1 (podmínka 1): <činnost pro podmínku 1>.
. . . . . .
CASE n (podmínka n): <činnost pro podmínku n>.

Strukturovaná angličtina

<úvodní fráze pro opakování>:
    <opakovaná činnost>.
<podmínka pro opakování>.

<úvodní fráze pro opakování a podmínka opakování>:
    <opakovaná činnost>.

REPEAT UNTIL, WHILE DO, FOR EVERY DO

Příklad - Stipendia

FOR EVERY student DO
IF student složil předepsané zkoušky, THEN
SELECT:
CASE 1 (Stud. průměr < 1.50):
Nejvyšší stipendium.
CASE 2 (Stud.průměr > 1.50):
Žádné stipendium.
OTHERWISE,
Žádné stipendium.

Kde je chyba?

Příklad - Stipendia

Systematické číslování podle úrovní vnořených konstrukcí:

1. FOR EVERY student DO
    1.1 IF student složil předepsané zkoušky, THEN
        1.1.1 SELECT:
            CASE 1 (Stud. průměr =< 1.50):
                1.1.1.1 Nejvyšší stipendium.
            CASE 2 (Stud.průměr > 1.50):
                1.1.1.2 Žádné stipendium.
    OTHERWISE
        1.1.2 Žádné stipendium.
2. Další činnost

Příklad - Koktejly v letadlech

„Jestliže je let obsazen více jak z poloviny a průměrná cena letenky přesahuje 350$, pak dáváme koktejly zdarma, pokud to není vnitrostátní let.
U vnitrostátních letů účtujeme všechny koktejly, když je podáváme.
Dáváme je, jen když je let více jak z poloviny obsazen.“

SELECT:
CASE 1 (Obsazeno z více než 50% a letenka stojí více než 350$
a není vnitrostátní let): Podávej koktejly zdarma.
CASE 2 (Vnitrostátní let):
Účtuj podané koktejly.
IF Obsazeno z více než 50%, THEN Podávej koktejly.
OTHERWISE Nic.

Příklad - Koktejly v letadlech

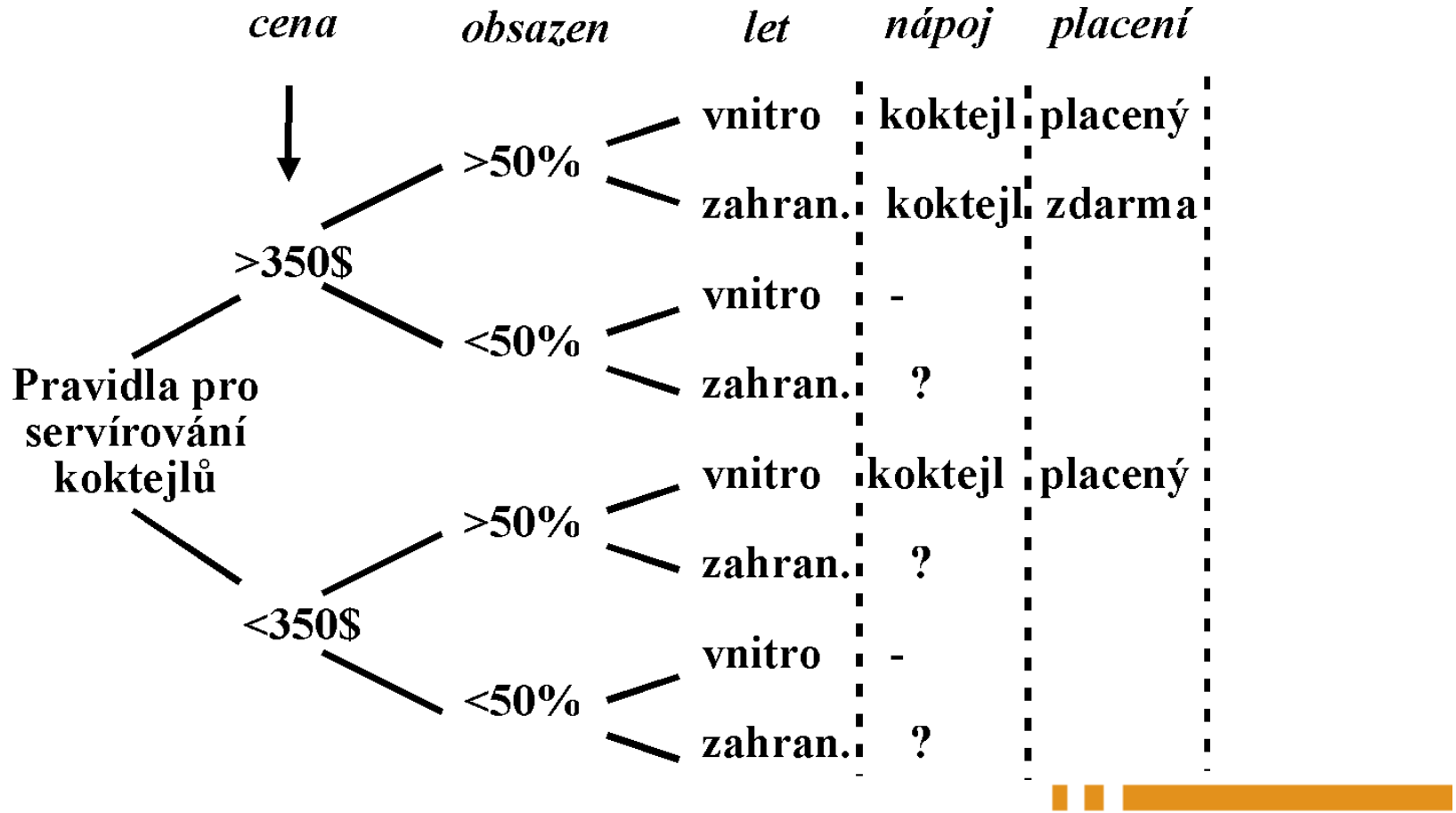
Specifikace pomocí *rozhodovací tabulky*

- jednodušší nalezení rozporů a nejasností

PODMÍNKY	*PRAVIDLA*							
	1	2	3	4	5	6	7	8
1. vnitrostátní let	A	N	A	N	A	N	A	N
2. obsazen nad 50%	A	A	N	N	A	A	N	N
3. cena > 350$	A	A	A	A	N	N	N	N
AKCE								
1. servírovány koktejly	A	A	N	?	A	?	N	?
2. bezplatně	N	A			N			

Příklad - Koktejly v letadlech

Specifikace pomocí *rozhodovacího stromu*



Úvodní a závěrečné podmínky

Použijí se když:

- Uživatel označuje činnost řešenou určitým procesem pomocí zvláštního, zcela konkrétního algoritmu, který byl používán dlouhou dobu.
- Analytik je přesvědčen, že mohou být použity různé algoritmy.
- Analytik chce, aby programátor vyzkoušel několik algoritmů, nechce se zabývat detaily, a zejména se nechce dohadovat s uživatelem o relativních výhodách těchto algoritmů.

Úvodní podmínky

- Které vstupy mají být k dispozici.

  Precondition

  Objeví se datový paket X.

- Jaké vztahy musí existovat mezi vstupy nebo uvnitř vstupů.

  Precondition

  Pacient je žena starší 18 let.

  Precondition

  Pacient je muž a oddělení je mužské.

- Jaké vztahy musí existovat mezi vstupy a datovými pamětmi.

  Precondition

  Pacient je již zaznamenán v paměti Pacient_na_lůžku.

Závěrečné podmínky

- Výstupy, které budou tvořeny procesem.
- Vztahy, které budou existovat mezi výstupními hodnotami a původními vstupními hodnotami.

  Postcondition

  Celková fakturovaná_cena bude vypočtena jako součet jednotková_cena + poštovné.
- Vztahy, které budou existovat mezi výstupními hodnotami a hodnotami v jedné a více pamětech.

  Postcondition

  Proměnná počet_faktur bude v paměti FAKTURY zvýšena o 1, hodnota bude na výstupu použita jako číslo_vystavené_faktury.
- Změny, které budou provedeny v pamětech

  Postcondition

  Vystavená_faktura zařazena jako poslední do paměti FAKTURY.

Úvodní a závěrečné podmínky

Doporučení: Nejprve popsat normální situace, poté připojit podmínky pro řešení chybových situací.

- Precondition 1
  Zákazník uvede číslo_účtu, které se shoduje s číslem účtu vedeným v paměti ÚČTY, jehož stavový_kód je nastaven na hodnotu „platný“.
- Postcondition 1
  Připravena faktura, na které jsou uvedeny číslo_účtu a prodejní_cena.
- Precondition 2
  Podmínka 1 není splněna (číslo_účtu nelze nalézt v paměti ÚČTY, nebo stavový_kód není „platný“).
- Postcondition 2
  Je sestavena chybová zpráva.

Nassi-Schneidermanovy diagramy - „Kopenogramy“