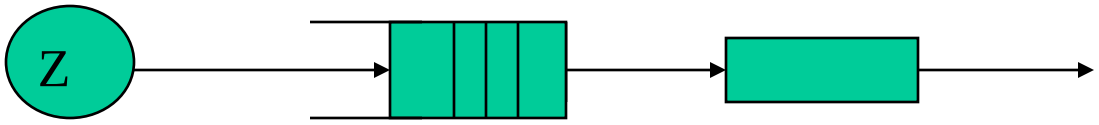
Úvod do systémů hromadné obsluhy

charakteristika SHO:

- systém, který poskytuje obsluhu určitého typu



zdroj požadavků | fronta | kanál obsluhy

Příklady reálných SHO:

požadavek:	**kanál obsluhy**	**obsluha**
auto	čerpací stanice	tankování
letadlo	přistávací dráha	přistání
osoba	lékař	ošetření
telefonní hovor	telefonní linka	spojení
stroj	seřizovač	seřízení
cestující	taxi	přeprava
auto	křižovatka	průjezd
program	procesor	provedení
paket	propoj. síť	spojení
terminál	počítač	přidělení stroj. času

Úvod do systémů hromadné obsluhy

Organizace SHO:

- z hlediska požadavků - co nejmenší doba strávená v SHO
- z hlediska provozovatele: jaké využití kanálů, jaká jejich organizace (specializované či universální), jaký počet kanálů, jak dlouhé fronty, atd.

Informace potřebné k modelování:

1) informace o příchodech požadavků do SHO:

- intervaly: pravidelné
  náhodné (znalost pravděpodobnost. rozložení)
- příchody požadavků: jednotlivé
  po skupinách (rozložení?)

2) organizace front

- SHO s neomezenou frontou
- SHO s omezenou frontou
  - frontová disciplina: FIFO, LIFO, výběr dle délky obsluhy, náhodný výběr, výběr dle priorit (slabé a silné priority)
  - chování požadavků:
    - trpělivé: čekají ve frontě potřebnou dobu
    - netrpělivé: rezignují ihned
      odpadnou po určité době

Úvod do systémů hromadné obsluhy

3) informace o procesu obsluhy:
doba obsluhy: stejná či náhodná (pravděpodobnostní rozložení)
kanály obsluhy : universální, specializované
obsluha: jednotlivá (u lékaře), po skupinkách (křižovatka, výtah)
dostupnost kanálů: nepřetržitá, s přestávkami (prav., či nikoliv)

Cíle modelů SHO**:**
(uvažujeme jednu frontu, jeden kanál obsluhy)

- **intensita provozu** (využití obsluhy)
  $\rho = \lambda / \mu = \lambda * Ts$
  λ …intensita příchodů (střední počet příchodů za jednotku času)
  μ…intensita obsluhy (střed. počet obsl. pož. za jednotku času)

- **statistika počtu požadavků v SHO**
  - Nw….počet požadavků ve frontě
  - Ns…..počet obsluhovaných požadavků
  - Nq….celkový počet požadavků v SHO: Nq = Nw + Ns

  E(Nw) =?, D(Nw)= ?, E(Ns) = ?, D(Ns) = ?, E(Nq) =?, D(Nq) =?

- **doby strávené požadavkem v SHO:**
  - Tw….doba strávená požadavkem ve frontě,
  - Ts…..doba obsluhy požadavku
  - Tq….celková doba strávená požadavkem v SHO

  E(Tw) =?, D(Tw) =?, E(Ts) = ?, D(Ts) = ?, E(Tq) = ?, D(Tq) = ?

Simulace SHO

pro popis SHO požadujeme:

- representace objektů a popis jejich chování
- objekty: trvalé (např. kanály obsluhy)
  přechodné (např. požadavky SHO)
- třídy: hromadný popis podobných objektů
- popis chování: diskrétní události či procesy

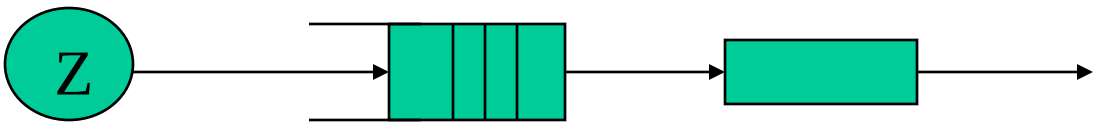
Přehled synchronizačních principů simulačních systémů

a) plánování událostí (event scheduling approach)

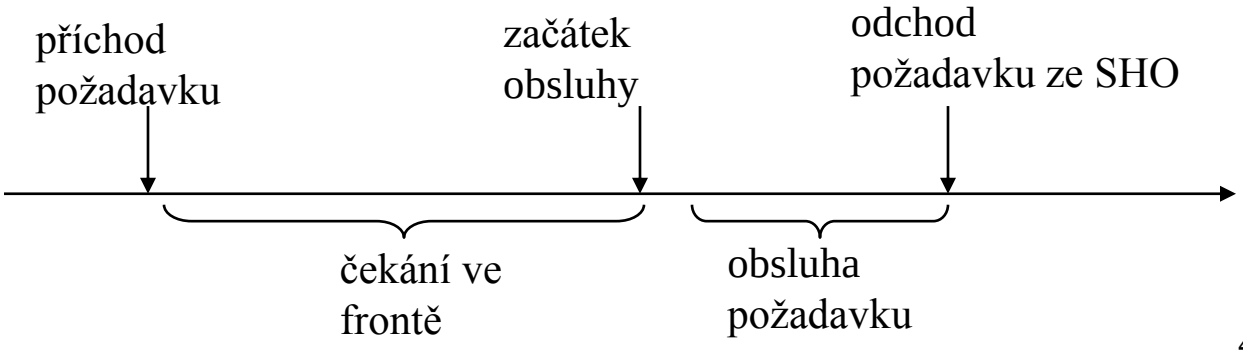
b) sledování aktivit (activity scanning approach)

c) interakce procesů (process interaction approach)

Příklad1: SHO s jedním kanálem obsluhy a s jednou frontou



obecný sled událostí (odvozený od každého požadavku):



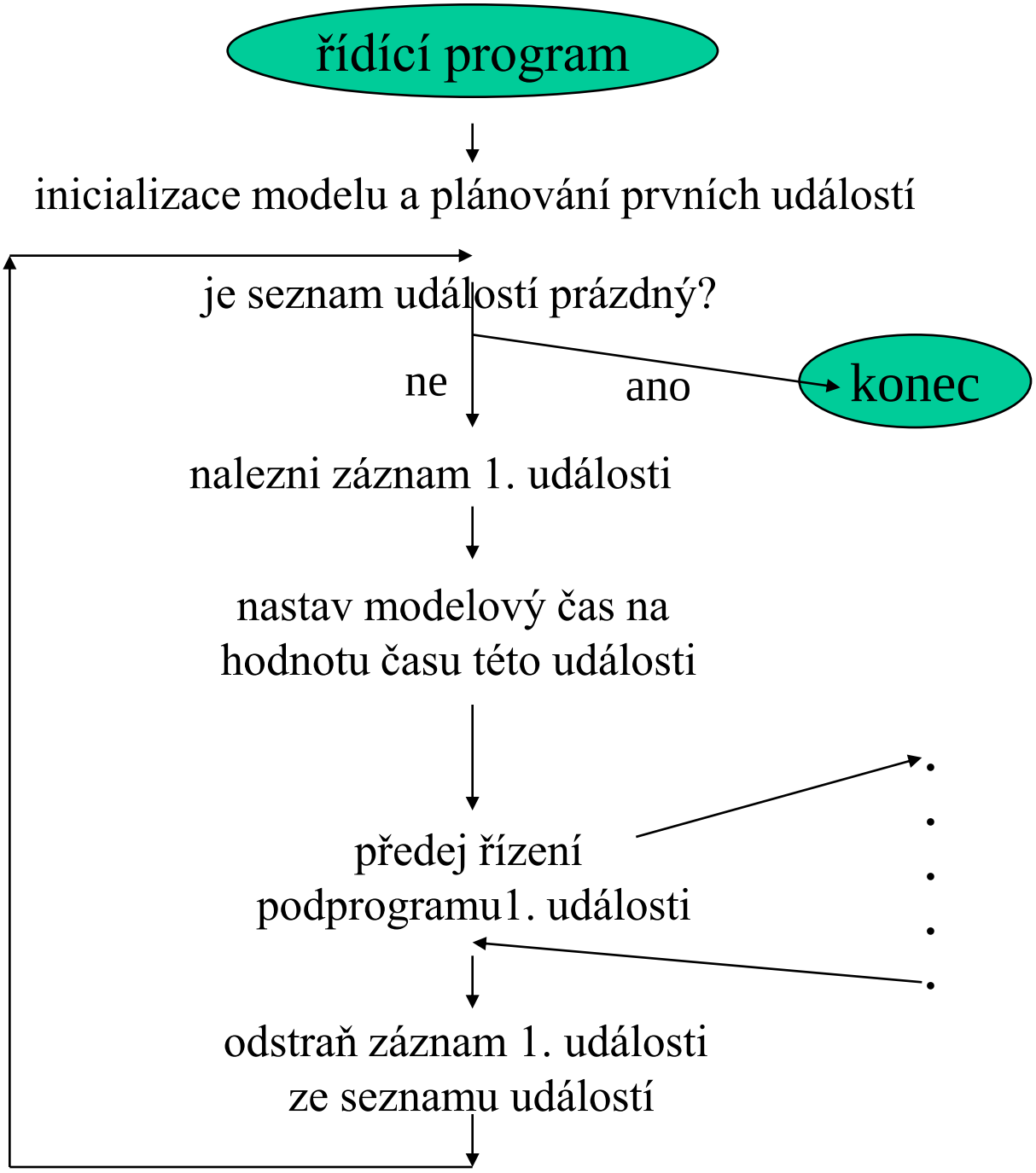
Metoda plánování událostí

Stručná charakteristika přístupu:

- plánovat události lze pouze na základě hodnot modelového času
- neexistuje možnost plánovat události v závislosti na splnění nějaké podmínky
- popis událostí - formou podprogramů
- změny atributů modelu - vázány výhradně na provedení zmíněných podprogramů
- podprogramy událostí - proběhnou bez přerušení
- uživatel má k disposici speciální příkazy (metody) pro dynamické plánování podprogramů
- podprogramy jsou plánovány do seznamu událostí dle hodnot modelového času
- návaznost provádění podprogramů událostí a údržbu modelového času - zajišťuje simulační systém pomocí řídícího programu (také kalendářní program)
- simulační jazyky orientované na události (také E -jazyky, např. SIMSCRIPT)

Metoda plánování událostí

algoritmus řídícího programu: dle seznamu událostí určuje aktivní událost

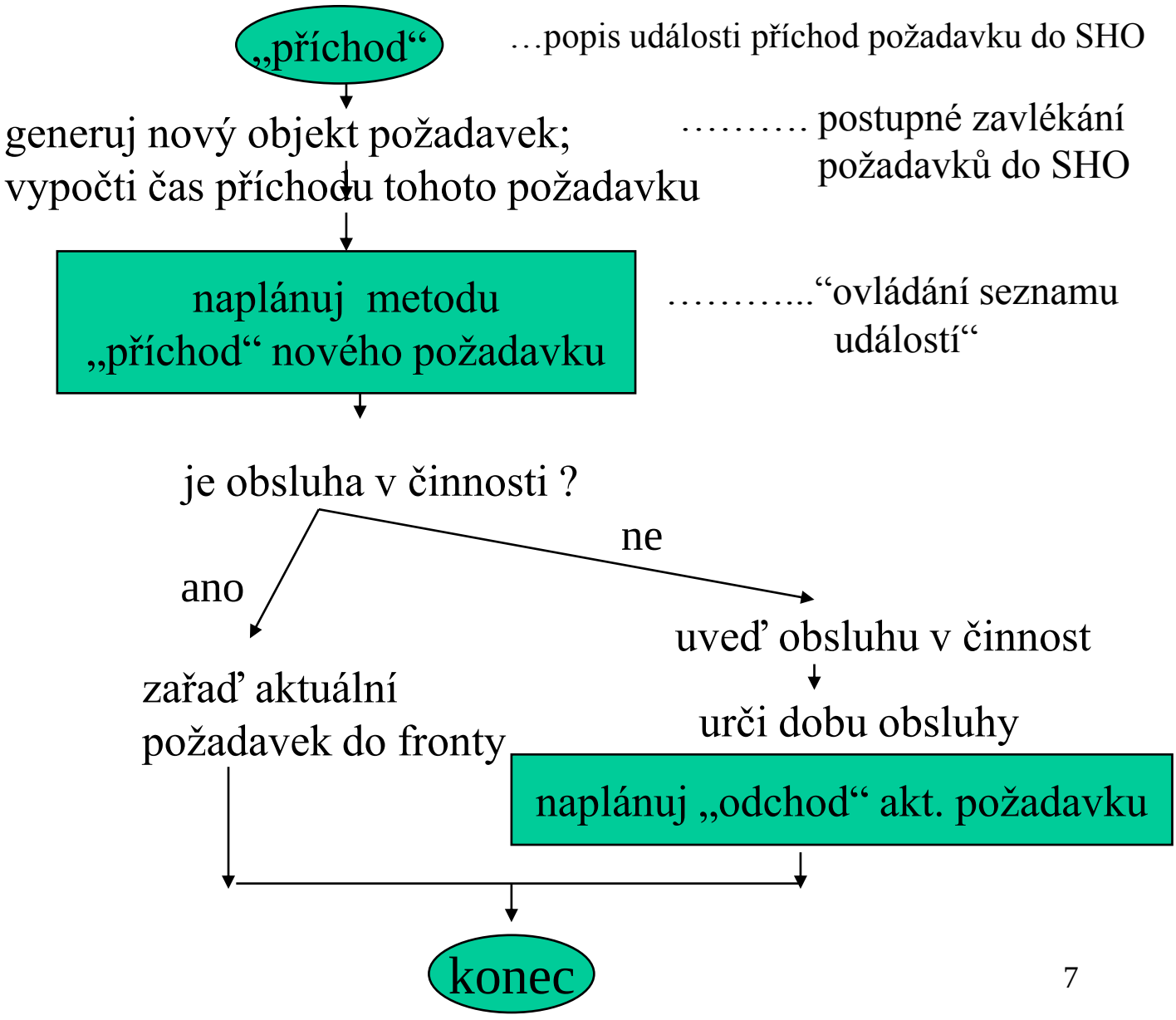


Metoda plánování událostí

demonstrační příklad: popis chování požadavku ze SHO

každý požadavek -3 události: „příchod“, „začátek obsluhy“, „odchod“

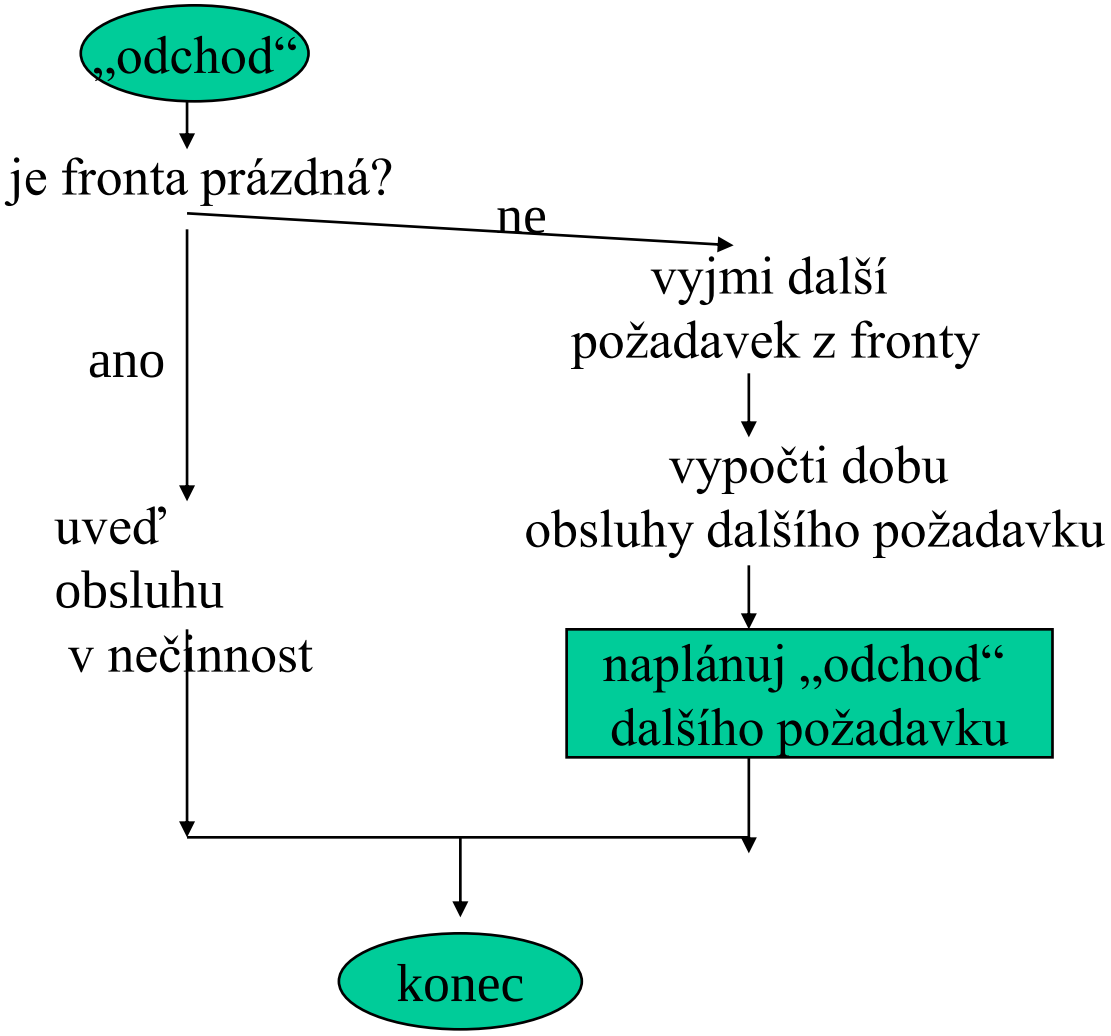
- obecně lze definovat 3 metody objektu požadavek;
- můžeme ale sloučit„začátek obsluhy“ s událostí „příchod“ (je-li kanál volný) nebo s událostí „odchod“ předchozího požadavku



Metoda plánování událostí

demonstrační příklad:

podprogram události „ odchod “ z příkladu 1:



poznámka: pro výpočet náhodných hodnot jako např. „časy příchodů jednotlivých požadavků“, „doby obsluhy“, atd. předpokládáme použití generátorů náhodných čísel

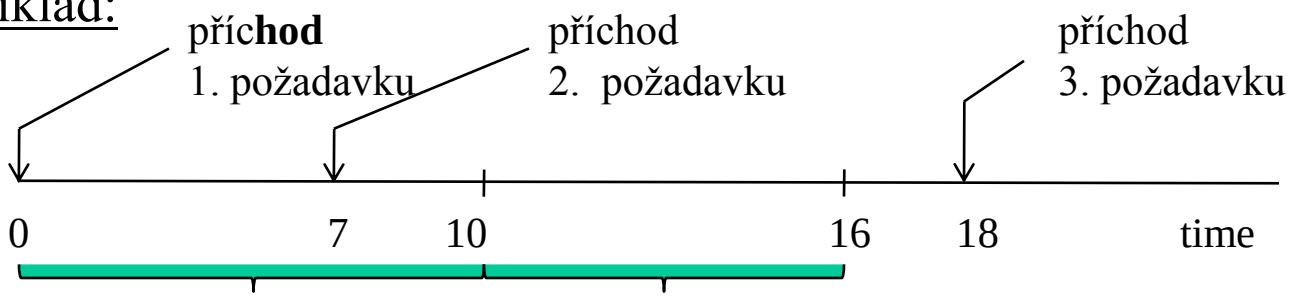
Metoda sledování událostí

Stručná charakteristika:

- netypický přístup: chybí plánování, neexistuje seznam událostí,
- metoda umožňuje aktivovat události na základě modelového času nebo pravdivosti nějaké podmínky,
- simulační program je rozdělen na části (v CSL nazývané „aktivity“), které reprezentují jednotlivé události v systému; tento program je periodicky prováděn: v závislosti na podmínkách provedení jednotlivých událostí se zmíněné části programu buď provedou nebo nikoliv;
- vazbu jednotlivých událostí na modelový čas lze zajistit pomocí tzv. časových proměnných, které svou hodnotou určují dobu, za kterou má být příslušná část programu (tj. událost) provedena,
- provedení časově závislých částí programu (*např. příchod požadavku, ukončení obsluhy*) lze podmínit nulovou hodnotou přidružené časové proměnné (PRICHOD.time, OBSLUHA.time); provedení ostatních částí (*zahájení obsluhy*) lze podmínit vyhodnocením přidružené booleovské podmínky (*prázdná fronta* & *obsluha v nečinnosti*),
- obsah všech časových proměnných je v průběhu simulace automaticky modifikován tak, že se po provedení všech událostí zmenší o minimum jejich hodnot; tímto způsobem se při simulaci realizuje metoda proměnného časového kroku,
- po každé z výše zmíněných modifikací nabývá alespoň jedna z časových proměnných nulovou hodnotu a uvolní se podmínka pro provedení přidružené časově závislé části programu,
- modifikaci časových proměnných lze před následujícím simulačním cyklem potlačit příkazem RECYCLE.

Metoda sledování aktivit

Příklad:



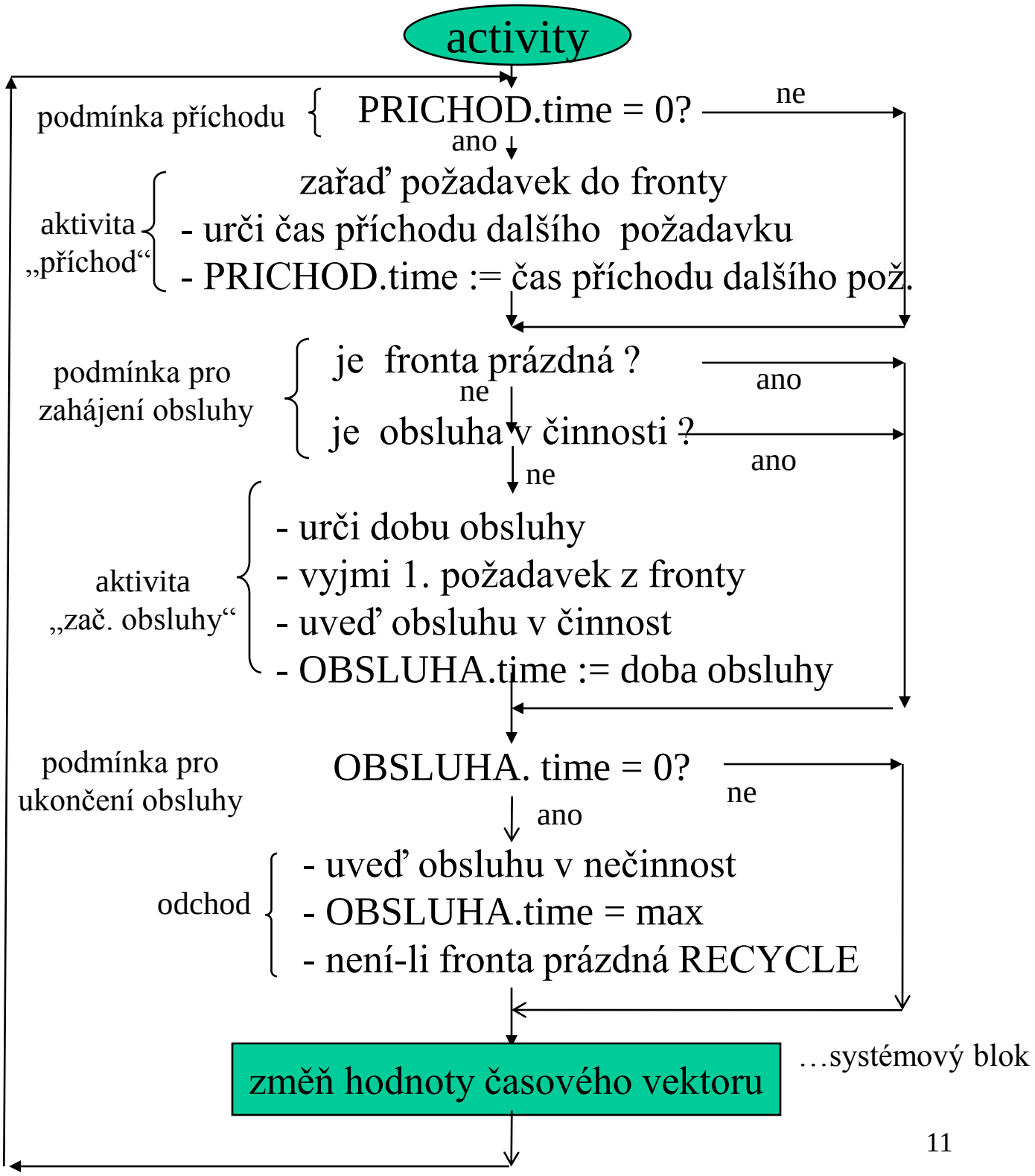
Obsah proměnných:

	PRICHOD.time	OBSLUHA.time	time
počáteční stav:	0	max	0
po příchodu 1. pož.:	**7**	10	0
po syst. bloku SB:	0	3	7
po příchodu 2. pož.:	11	**3**	7
po syst. bloku SB:	8	0	10
po ukončení obsluhy 1. požadavku	8	max	10
po syst. bloku SB **vlivem RECYCLE funkce potlačena**	8	max	10
po zahájení obsluhy 2. požadavku	8	**6**	10
po syst. bloku SB:	2	0	16

ukončení obsluhy, atd.

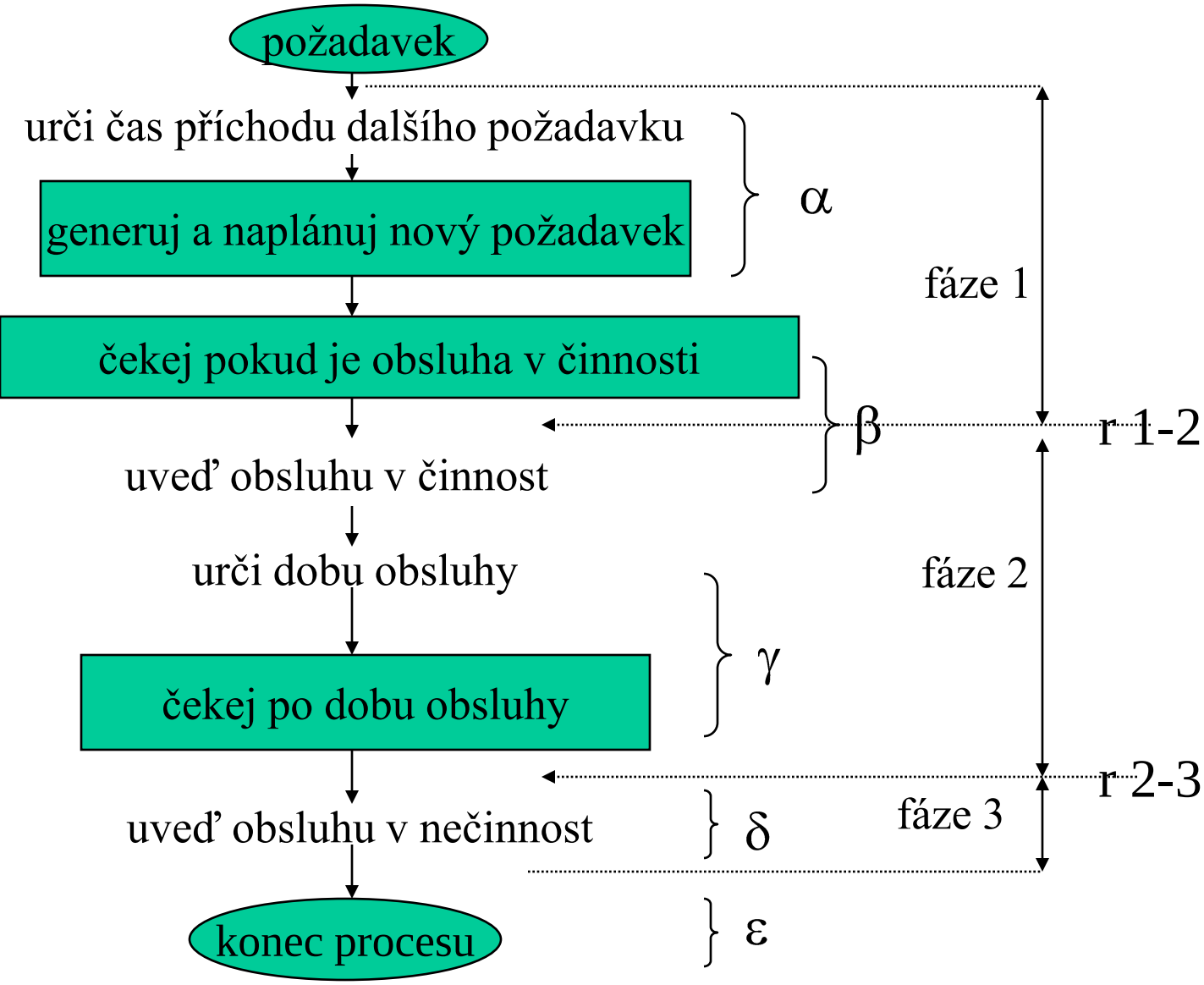
Metoda sledování aktivit

počáteční nastavení: PRICHOD.time = 0, OBSLUHA.time = max;



Metoda interakce procesů

demonstrační příklad: popis chování požadavku z příkladu 1



diskuse: tři fáze procesu požadavek => dva reaktivační body:

- r2-3: čekání „na dobu známou“ ....**nepodmíněný synchronizační příkaz** (pokračování procesu lze naplánovat)
- r1-2: čekání „na neznámou dobu“…**podmíněný synchronizační příkaz** (nelze naplánovat pokračování procesu) - obecná implementace je obtížná (viz následující slide)

Metoda interakce procesů

Stručná charakteristika:

- procesově orientovaný model má těsnější vazbu na souvislosti v simulovaném systému:
  - poskytuje přehlednější a čitelnější popis,
  - proces plně popisuje stavové změny určité komponenty,
- proces je přerušitelný ze dvou důvodů:
  - další události proběhnou v pozdějším čase,
  - další události jsou vázány na splnění nějaké podmínky,
- lze použít princip plánování jako u plánování událostí,
- je nutná podpora pro koprogramy,
- je komplikovanější synchronizace:i implementace:

1) „úplné systémové řešení“ : (seznam plánovaných událostí (SPLU) + seznam podmíněných událostí (SPOU)

Algoritmus řídícího programu:

1) Provedou se všechny fáze procesů naplánované pro aktuální hodnotu modelového času ze seznamu SPLU.
2) Prohlíží se (aniž by došlo k posunu modelového času) kompletní seznam SPOU a provedou se všechny fáze, které mají splněné podmínky; toto pokračuje, pokud není seznam SPOU prázdný a nebo tam taková fáze není.
3) Posune se modelový čas do času „nejbližší“ fáze v seznamu SPLU, pak se postup opakuje dle bodu 1.

Metoda interakce procesů

b) řešení založené na spolupráci s uživatelem -

b1) využití semaforů

demonstrace přístupu (použita syntax jazyka GPSS: General Purpose Simulation Systém)

GENERATE < parametry> (α)

SEIZE <identifikace kanálu obsluhy> (β)

ADVANCE <časový interval> (γ)

RELEASE <identifikace kanálu obsluhy> (δ)

TERMINATE <parametry> (ε)

Poznámky:

- seznam SPOU (viz. systémové řešení) je rozdělen na řadu dílčích seznamů typu FRONTA (každý z nich sdružuje procesy čekající na stejnou podmínku)
- uživatel explicitně upozorní na splnění podmínky, která může způsobit pozdržení některého procesu
- funkce SEIZE……..vyjmutí ze SU a zařazení do FRONTA
- funkce RELEASE...vyjmutí z FRONTA a naplánování do SU

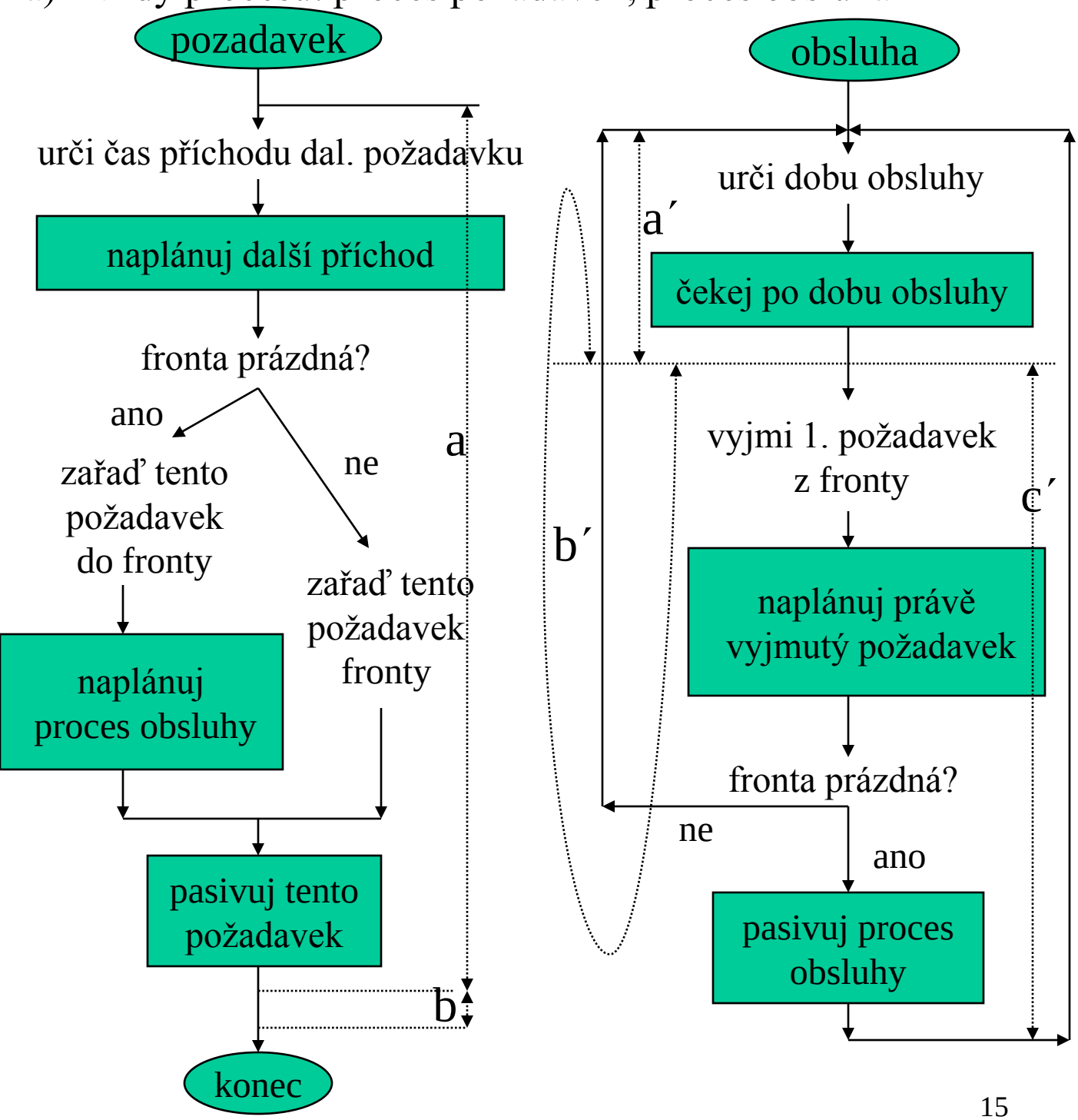
b2) využití explicitní pasivace procesů v případě nesplnění podmínek pro jejich pokračování (př. systém SIMULA 67)

Metoda interakce procesů

pasivace procesů

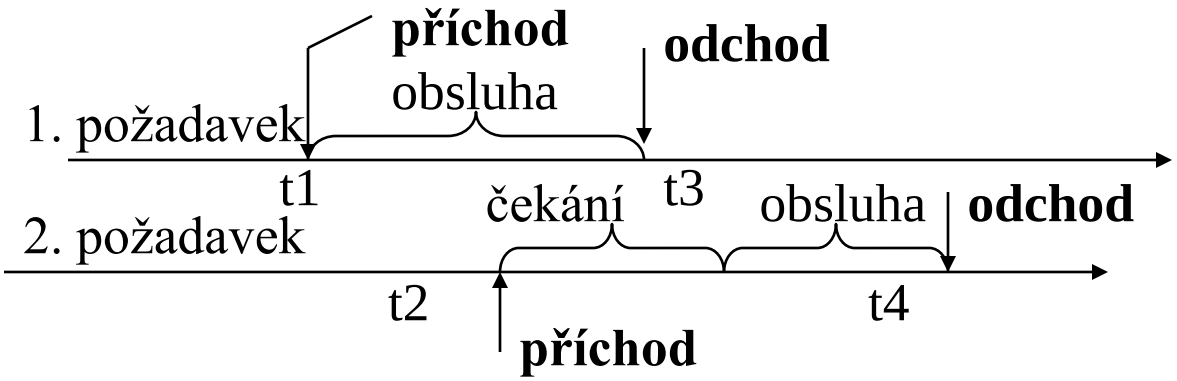
demonstrace přístupu: aplikace na příklad 1:

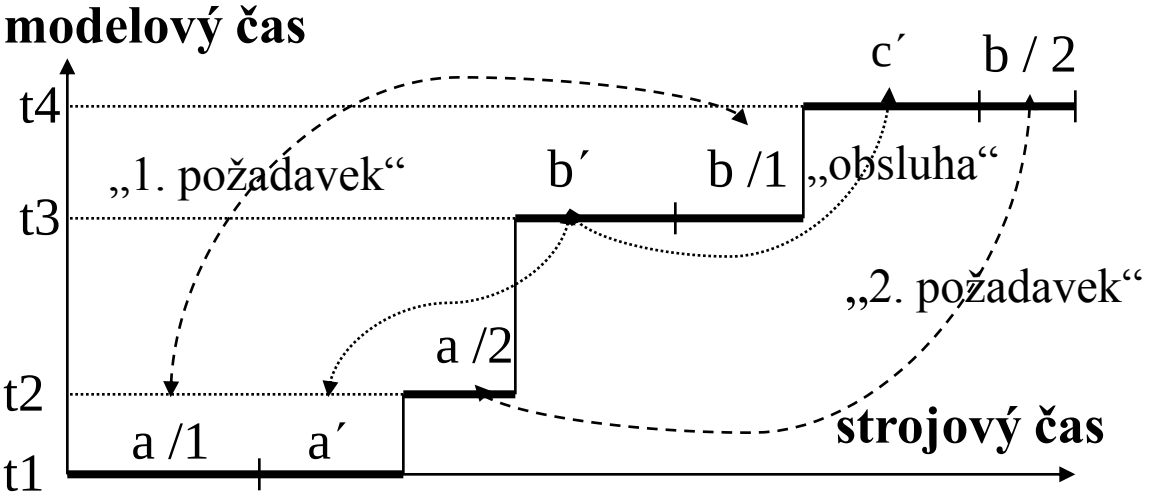
a) 2 třídy procesů: proces požadavek, proces obsluha



Metoda interakce procesů

demonstrace synchronizace s modelovým časem: viz procesy požadavek a obsluha





poznámka: **pro získání statistik** - doplnit proces požadavek:

a) kdekoliv ve fázi a:

cas_prichodu := time;

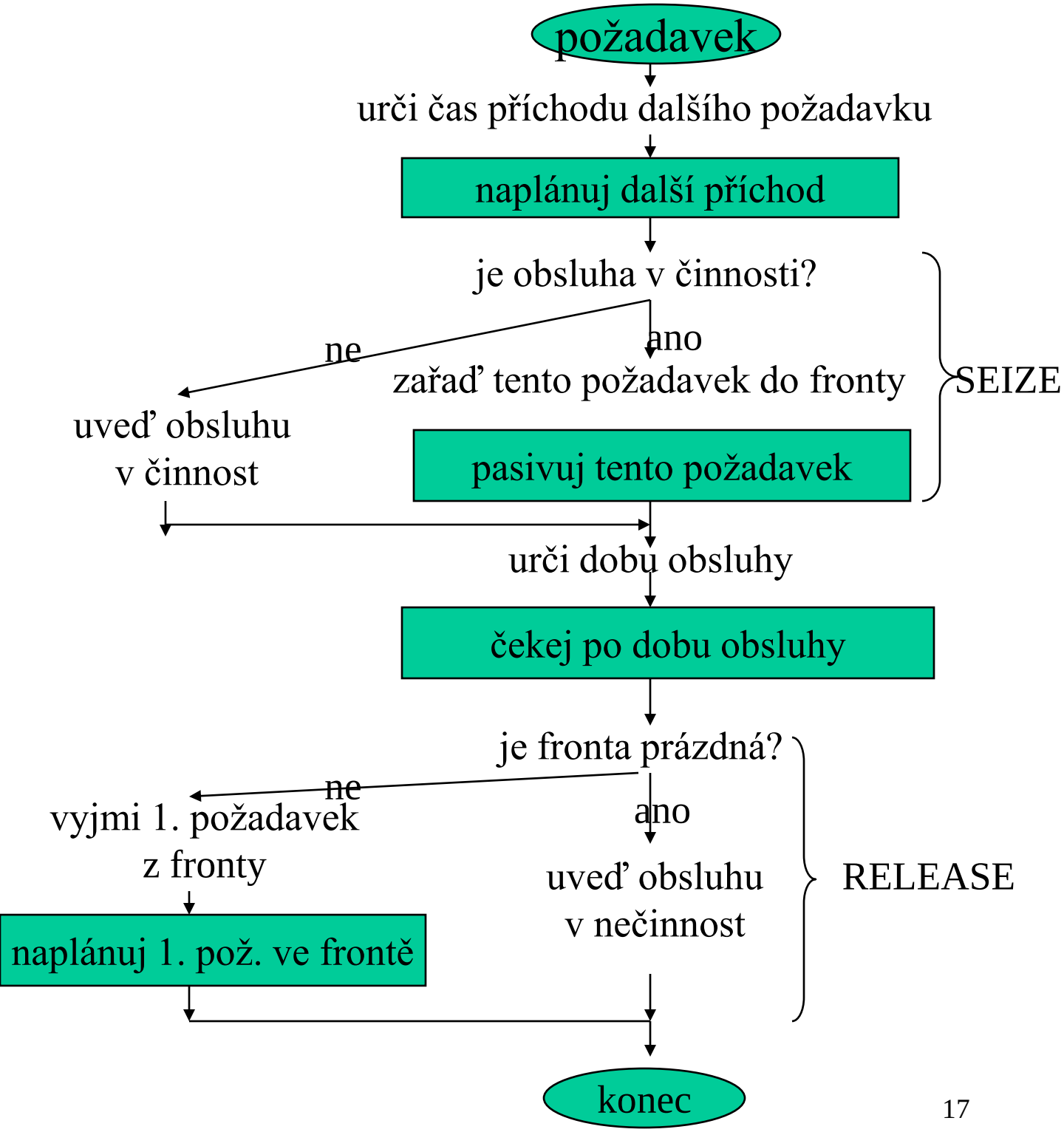
b) kdekoliv ve fázi b:

doba := time - cas_prichodu

„příkaz pro korekci histogramu hodnotou doba“;

Pasivace procesů

demonstrační příklad : použití jediné třídy procesů:



Histogram

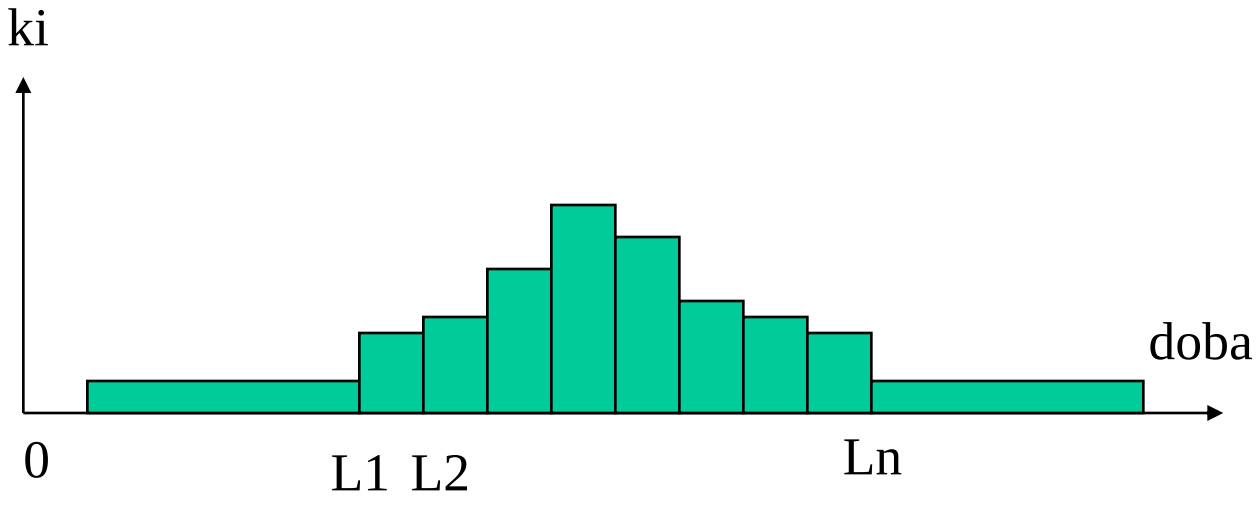
- tabulka četností hodnot (sledované veličiny), v určitých intervalech

L1	k1
L2	k2
L3	k3
L4	k4
Ln	kn
	kn+1

k1..počet hodnot z inter. (0:, L1 >
k2..počet hodnot z inter. (L1, L2 >

.
.
.
.
.
.
.
.
.

kn..počet hod. z inter. (Ln-1,Ln+1)
kn+1..počet hodnot z inter. (Ln,∞:)



Simulační systémy diskrétního typu:

obecná charakteristika:

- kromě běžných konstrukcí universálních programovacích jazyků nabízí další prostředky pro podporu následujících činností:

  - generování pseudonáhodných čísel pro často používaná rozložení,
  - generování, případně rušení přechodných objektů (případně garbage collector),
  - ovládání seznamu událostí (synchronizační příkazy),
  - případný popis struktury SHO,
  - provádění často používaných operací nad seznamy objektů,
  - statistická vyhodnocení sledovaných veličin (výpočet histogramů, středních hodnot, standardních odchylek apod.).

Simula 67

stručná charakteristika:

Simula I: 1965 - simulační jazyk diskrétního typu

Simula 67: (1967)

- objektově orientovaný universální programovací jazyk pro vytváření speciálních „nástrojů“pro různé účely
- dostupné konfigurace:
  - základní jazyk Simula 67 (ZJ) - universální program. jazyk
  - systémová třída SIMSET - podpora pro zpracování seznamů,
  - systémová třída SIMULATION - podpora pro simulaci

Základní jazyk Simula 67:

- blokově orient. jazyk, mohutná nadstavba Algolu 60
- hlavní vlastnosti navíc: **třídy** - prototypy dat a činností
  objekty - dynamic. representace tříd

stručná charakteristika:

- možné hierarchie tříd, dědičnost atributů z nadřazených tříd
- virtuální procedury
- chráněné atributy
- vyjma metod třídy disponují i t.zv. operační částí třídy:
  - rozštěpená operační část - možnost vkládání operačních částí odvozených pod tříd do operační části nadřazené třídy
  - charakter koprogramu (možnost osamostatnění objektu s vlastním zásobníkem)

Simula 67

možné stavy objektů v Simule:

samostatný (vlastní zásobník):

blok s prefixem (hlavní program),

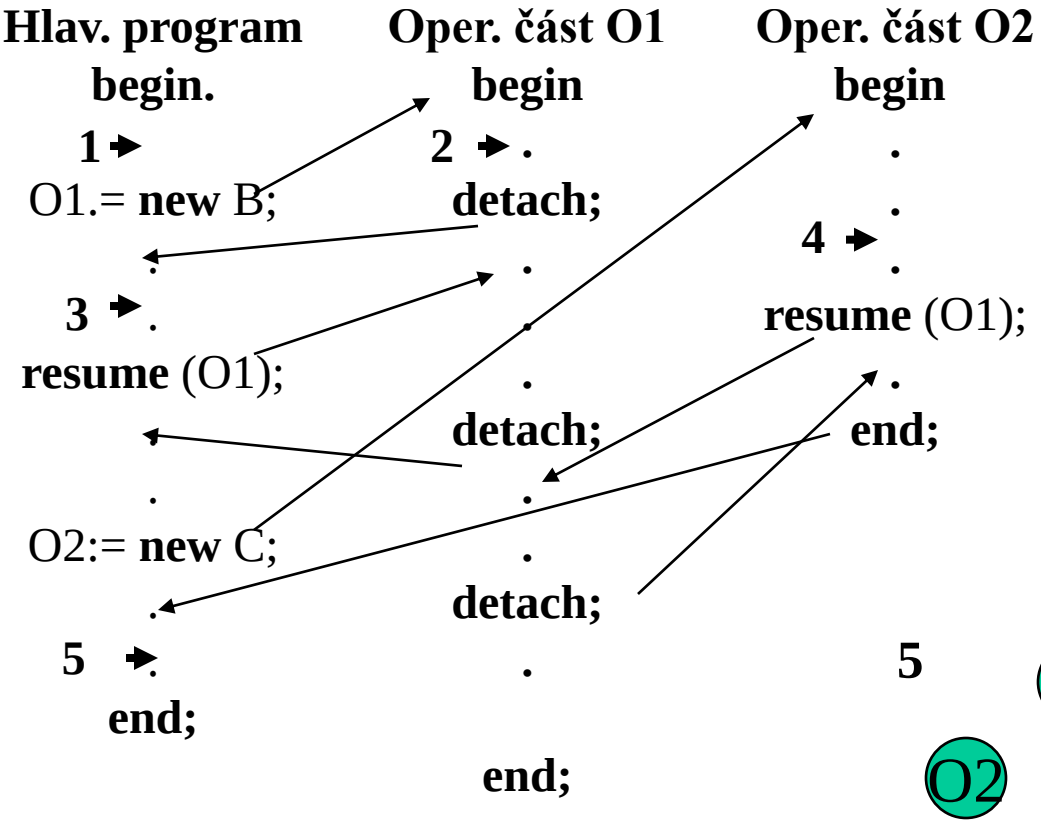
operační část třídy (po osamostatnění příkazem detach)

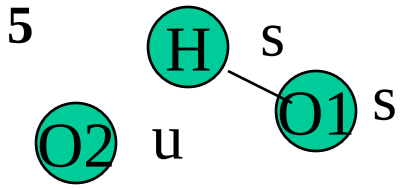
vnořený (zásobník vnořený do zásobníku rodičovského objektu):

podblok, procedura, funkce, operační část třídy (po vygenerování)

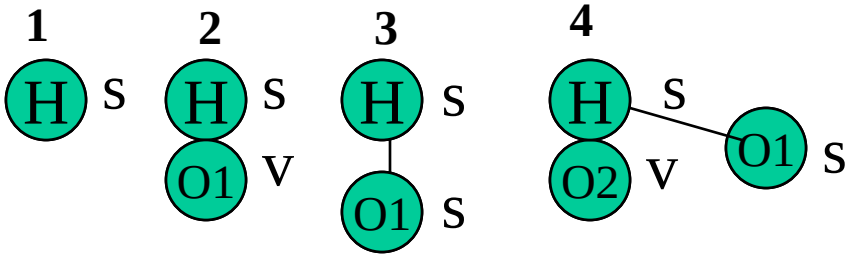
ukončený: operační část třídy po provedení všech příkazů

příklad: gener. a osamost. objektů O1, resp. O2….(obj třídy B, resp. C)





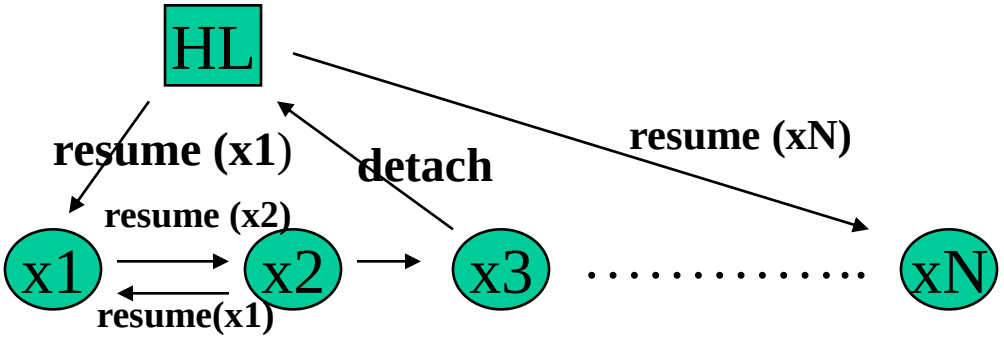
znázornění stavů:



Simula 67

kvaziparalelní systém v Simule 67: pro vytváření procesů

- obsahuje pouze objekty ve stavu samostatný (blok s prefixem + samostatné objekty tříd)
- jednoúrovňový nebo viceúrovňový (komponentou jednoúrovňového kvaziparalelního systému je opět blok s prefixem, který se stal hlavní komponentou nižší úrovně)
- jediná z komponent může být v daném okamžiku aktivní, ostatní mají vytvořen „reaktivační bod“
- situace v jednoúrovňovém kvaziparalelním systému „v ustáleném stavu“ (tj. po osamostatnění všech komponent)



přínos systémových tříd:

a) třída SIMSET - definuje základní operace nad seznamy objektů (pro operace nad frontami a pod.)

b) třída SIMULATION - definuje prostředky pro kvaziparalelní prostředí procesů:

- modelový čas + seznam událostí
- operace synchronizačního jádra (uvažují modelový čas a seznam událostí; předávání řízení)

Simula 67: demonstrační příklad

Příklad: SHO: jeden kanál obsluhy + jedna fronta

```
Simulation begin;                    (hlavní program: blok s prefixem)
; deklarace globálních. proměnných: pro generátory pseudonáhodných
    čísel, atd.
ref ( head ) fronta;  ptr kvalifikovaný pro objekty třídy head
                    ;  (representace hlavy seznamu)
ref (obsluha) obs ;   ptr kvalifikovaný pro objekty třídy obsluha
                    ;   (representace kanály obsluhy)
; následuje deklarace třídy požadavek (jde o podtříddu třídy process)
process class pozadavek;
        begin
        real cas_prichodu;  paměť příchodu požadavku do SHO
        activate new pozadavek delay negexp (4.0, u1); // λ=4
        if fronta . empty then begin
                                current  .into (fronta);
                                activate obs at time;
                                end
                        else current.into (fronta);
    passivate;
    histo (……); korekce histogramu před odchodem ze SHO
    end pozadavek;       konec procesu pozadavek
```

Simula 67 : demonstrační příklad

pokračování minulého příkladu-následuje popis procesu obsluha

```
process class obsluha;
        begin
    zac: hold ( negexp (5.0, u2)); //exp. rozložení doby obsluhy  λ=5
        activate fronta. first at time;
        fronta.first.out;
        if fronta.empty then passivate;
        go to zac
        end obsluha;

;následují výkonné operace hlavního programu
fronta:= new head;              generuje prázdný seznam FRONTA
; „určení semen pro  generátory pseudonáhodných čísel“
; „nastavení mezí pro intervaly histogramu“
obs:= new obsluha;              generuje jeden objekt třídy obsluha

activate new pozadavek at time; generuje 1. objekt třídy požadavek
     ;a plánuje jeho oper. část do SU pro okamžitou hodnotu model. času
hold (100 000);         pozdrží hlav. program na dobu trvání simulace
; „ tisk histogramu“

end;                    konec hlavního programu
```