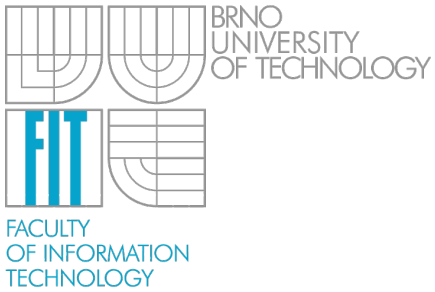
Strukturování Petriho sítí

Vladimír Janoušek

Vysoké učení technické v Brně, Fakulta informačních technologií
Božetěchova 2, 612 66 Brno
janousek@fit.vutbr.cz



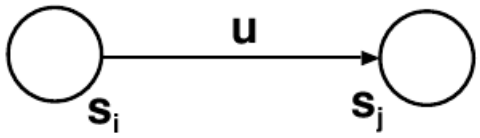
5/3/10

Obsah

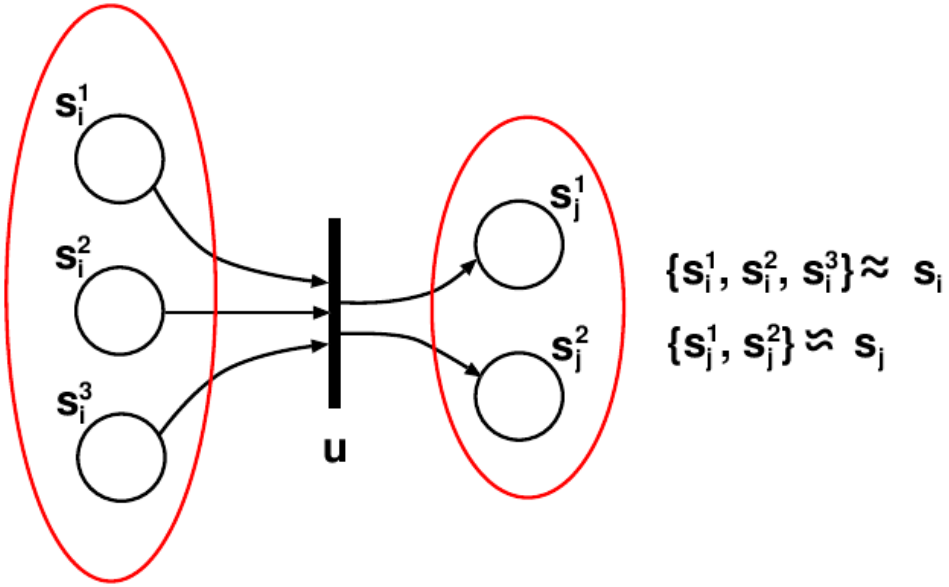
- Petriho sítě
- Strukturování Petriho sítí
- Víceúrovňové Petriho sítě
- Objektově orientované Petriho sítě

Petriho síť - základní idea

- C. A. Petri: Kommunikation mit automaten, 1962
- Přechod v konečném automatu:



- Přechod v Petriho síti:

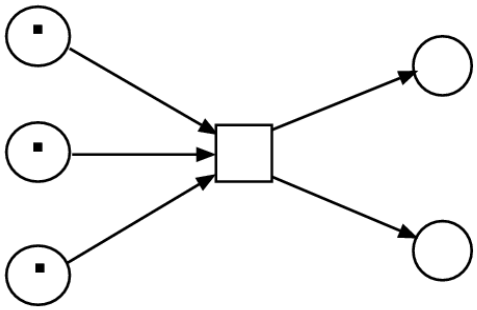


Parciální stavy -
vstupní a výstupní
podmínky přechodu

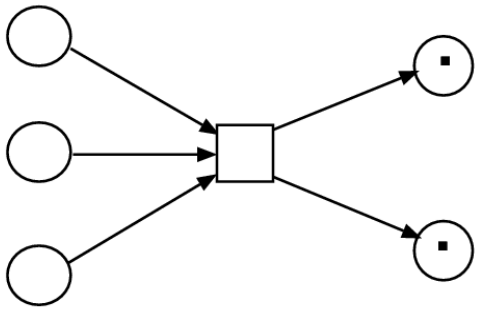
- Paralelismus, nedeterminismus, synchronizace
- Prvky Petriho sítě: místa, přechody, hrany, značky (tokens)

Dynamika PN - provádění přechodů

- Stav: před provedením (a) po provedení (b)

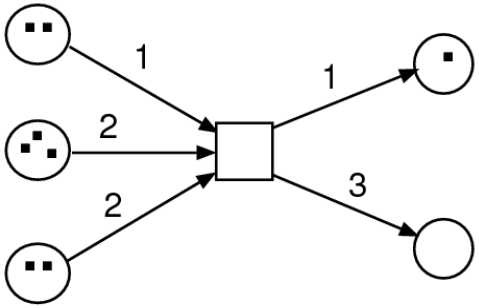


(a)

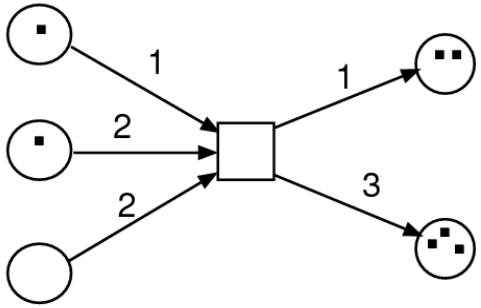


(b)

- Místo může obsahovat více značek, hranové výrazy specifikují počet značek



(a)



(b)

Petriho síť

Petriho síť je struktura

$$PN = (P, T, I, O, M_0),$$

splňující tyto podmínky:

- P, T jsou konečné množiny míst a přechodů, $P \cap T = \emptyset$,
- $I : P \times T \longrightarrow \mathbf{N}$ je funkce vstupních podmínek,
- $O : T \times P \longrightarrow \mathbf{N}$ je funkce výstupních podmínek,
- $M_0 : P \longrightarrow \mathbf{N}$ je počáteční značení.

Přechod t je **proveditelný** při značení M, když

$$\forall p \in P : I(p, t) \leq M(p)$$

Provedením přechodu t při značení M dojde ke změně značení na M' takové, že

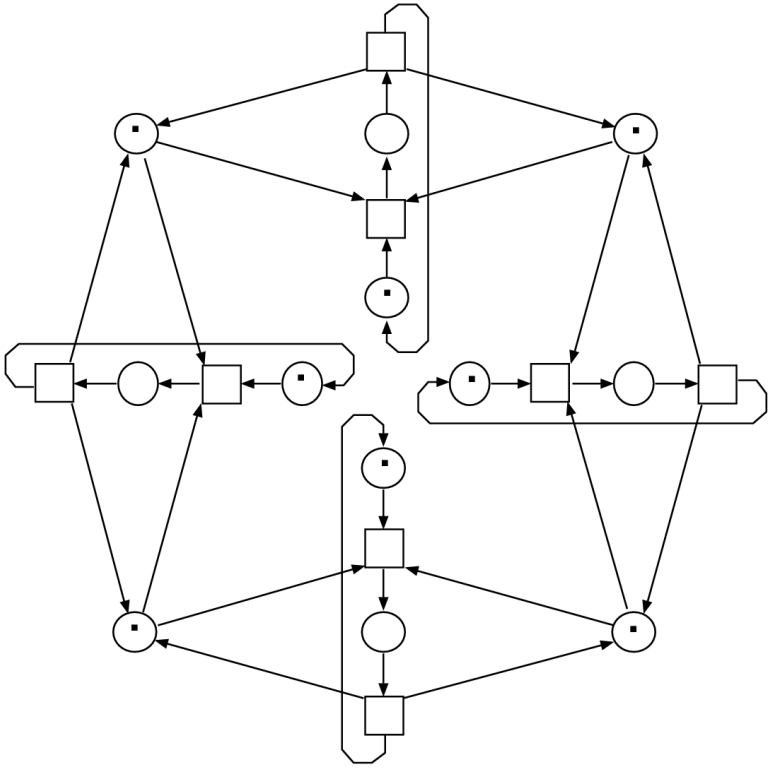
$$\forall p \in P : M'(p) = M(p) - I(p, t) + O(t, p)$$

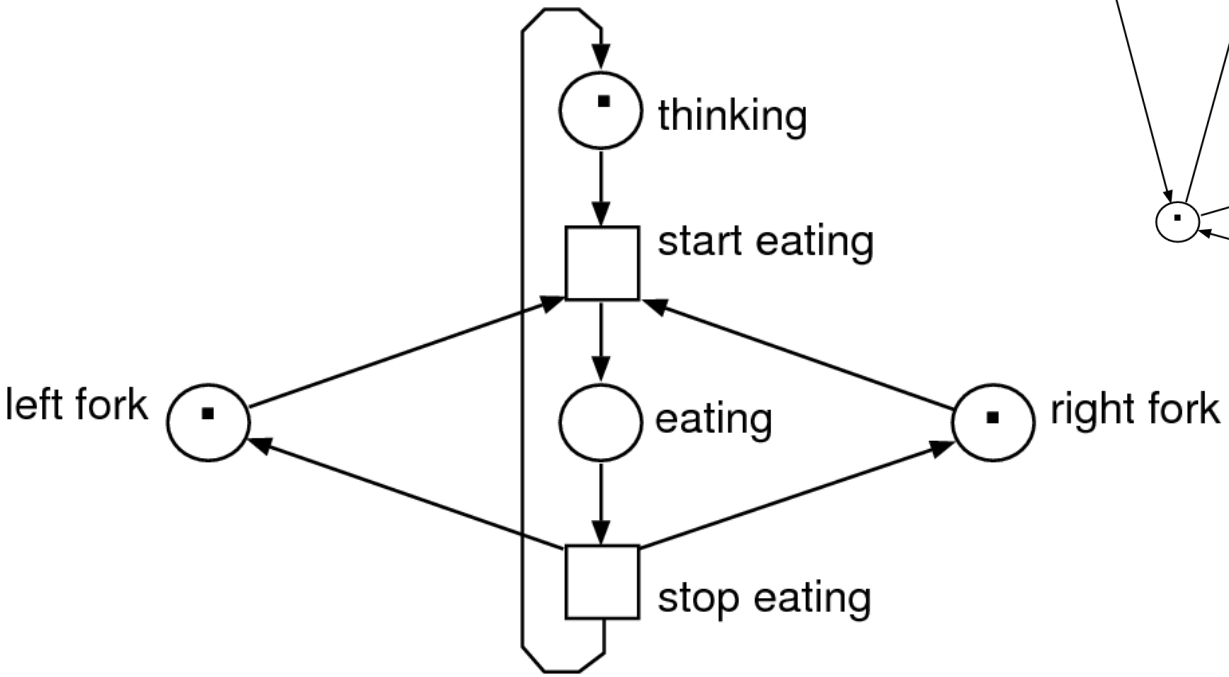
Notace pro provední přechodu: $M[t\rangle M'$.

Výpočetní posloupnost: $M_0[t_1\rangle M_1[t_2\rangle M_2[t_3\rangle M_3 \ldots$.

Příklad: Sdílení zdrojů

- Večeřící filosofové





Vysokoúrovňová Petriho síť

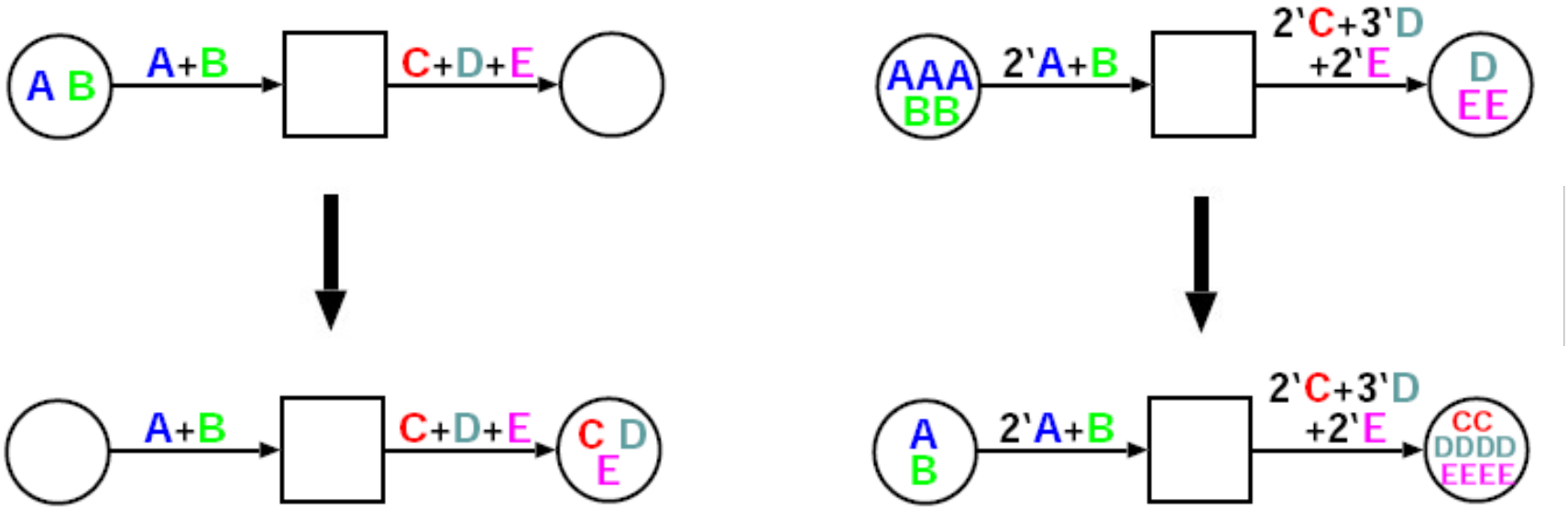
- Značky v PN lze individualizovat (obarvit) a PN lze rozšířit o možnost datových manipulací.

- CPN - Coloured Petri nets (K. Jensen, 1981)
  - K. Jensen: *Coloured Petri Nets*. Monographs in Theoretical Computer Science, Springer-Verlag, 1992-1997.
    Základní koncepty, analýza a průmyslové případové studie.
  - Alternativní koncepty HLPN lze považovat za dialekty CPN

- Nástroje: Design/CPN, CPN Tools (Aarhus University)
  - Fyzický čas
  - Hierarchická kompozice
  - Analýza stavového prostoru

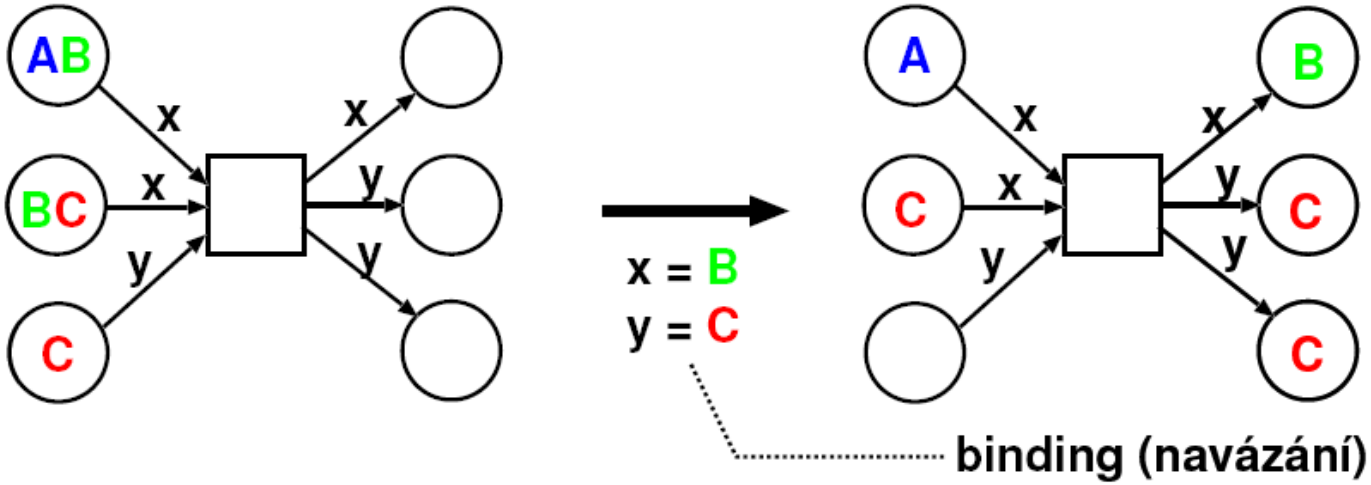
Vysokoúrovňová Petriho síť

- Místa obsahují multimnožiny značek (hodnot datových typů)
- Hranové výrazy specifikují multimnožiny značek



Vysokoúrovňová Petriho síť

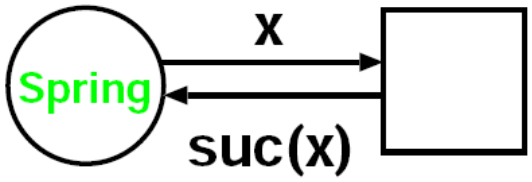
- Hranové výrazy mohou obsahovat proměnné
  - Přechod je proveditelný pro určitá navázání proměnných
  - (t, b) ... binding element



Vysokoúrovňová Petriho síť

- Hranové výrazy mohou obsahovat funkce pro datové manipulace

  Funkce jsou specifikovány inskripčním jazykem



suc(Spring)=Summer
suc(Summer)=Autumn
suc(Autumn)=Winter
suc(Winter)=Spring

- Kromě toho každý přechod může obsahovat strážní podmínku. Je to predikát, který musí být splněn pro navázání přechodu, aby mohl být přechod prohlášen za proveditelný.

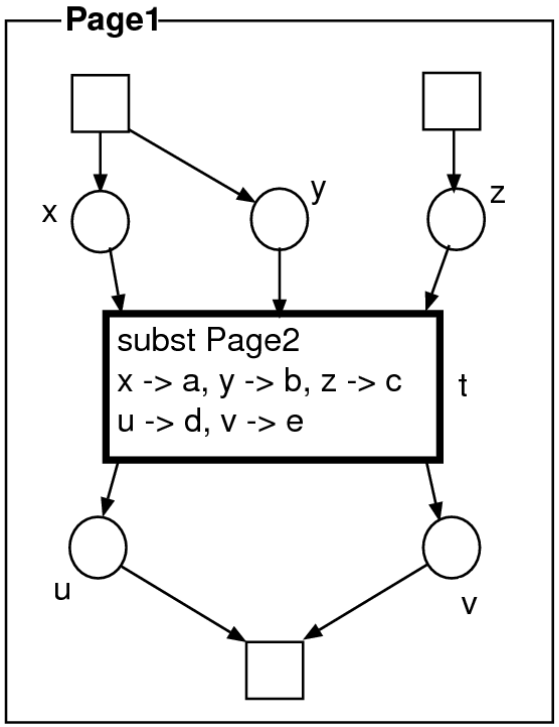
Strukturování Petriho sítí

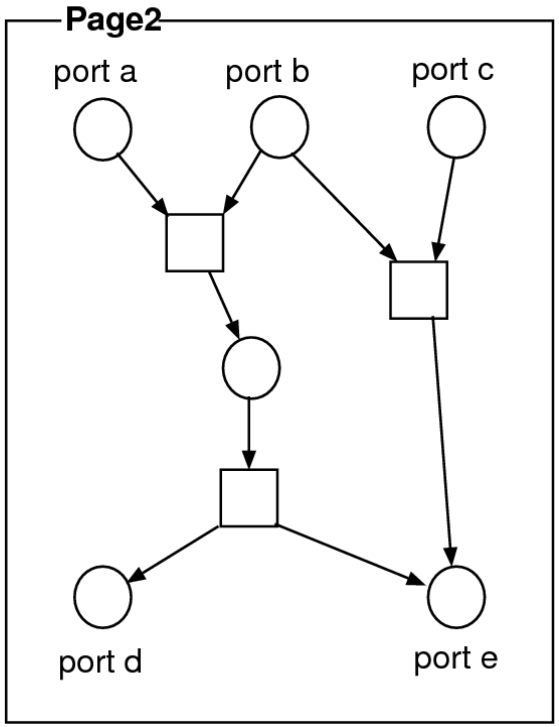
- Strukturovací a komunikační mechanismy:
  - Substituční přechod
  - Substituční místo
  - Invokační přechod
  - Fúzní množina míst
  - Synchronní kanál

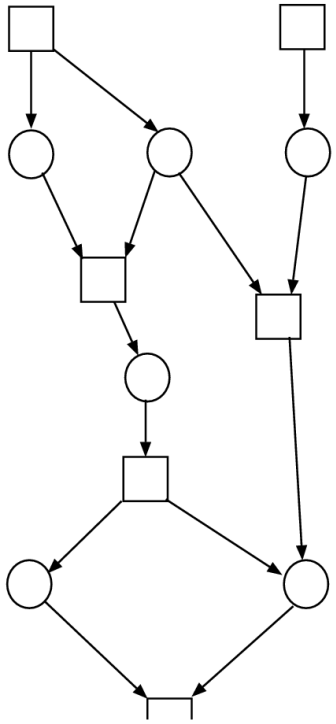
- P. Huber, K. Jensen, and R.M. Shapiro. **Hierarchies in Coloured Petri Nets.** In G. Rosenberg, editor, Advances in Petra nets 1990, volume 483 of Lecture Notes in Computer Science. Springer-Verlag, 1990.
- S. Christensen and N.D. Hansen. **Coloured Petri Nets Extended With Channels for Synchronnous Communication**. In G. Rosenberg, editor, Application and Theory of Petri nets, International Conference, volume 15 of Lecture Notes in Computer Science 815. Springer-Verlag, 1994.

Substituční přechod

- Stránky, porty, sokety
- Substuční přechod reprezentuje statickou instanci stránky

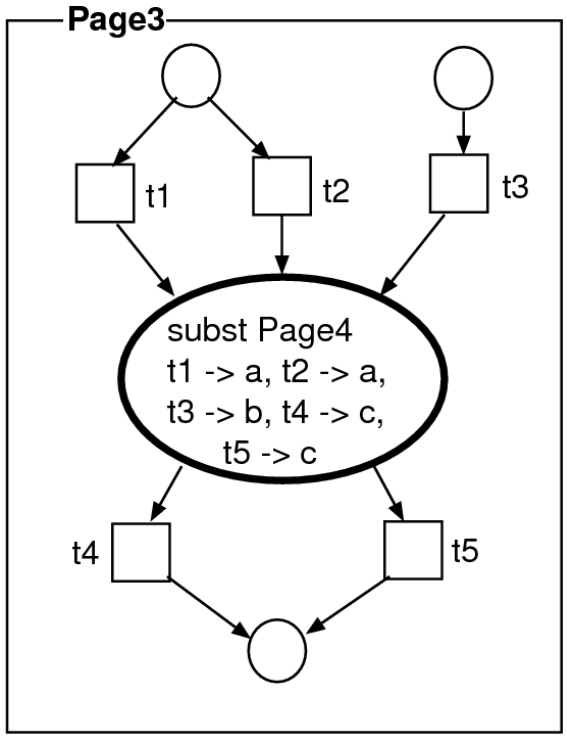


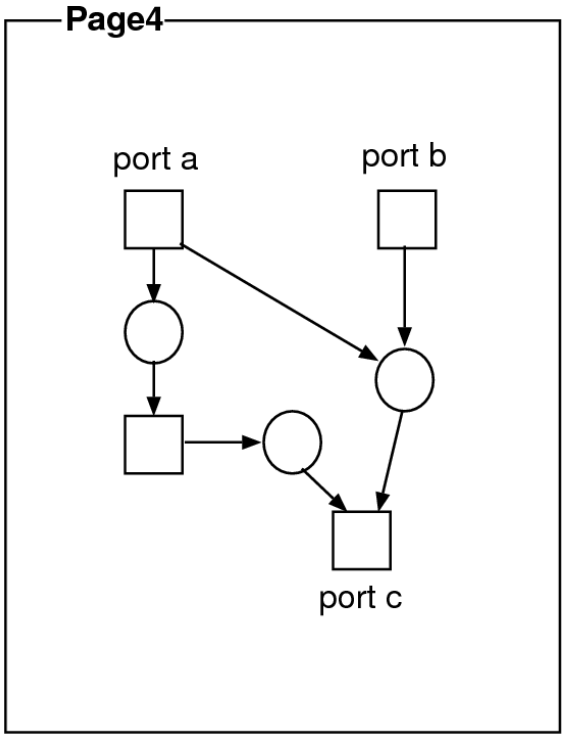


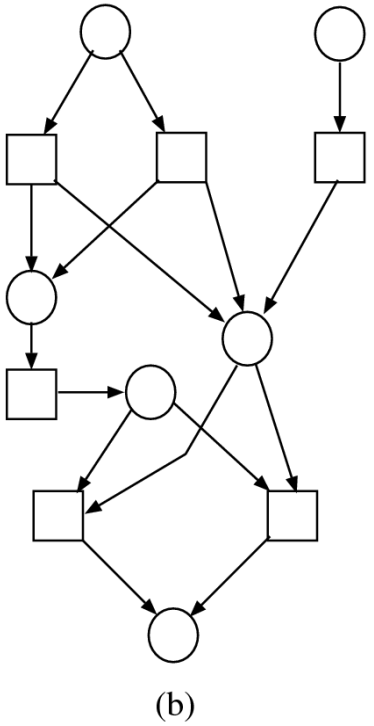


Substituční místo

- Duální koncept – porty a sokety jsou přechody

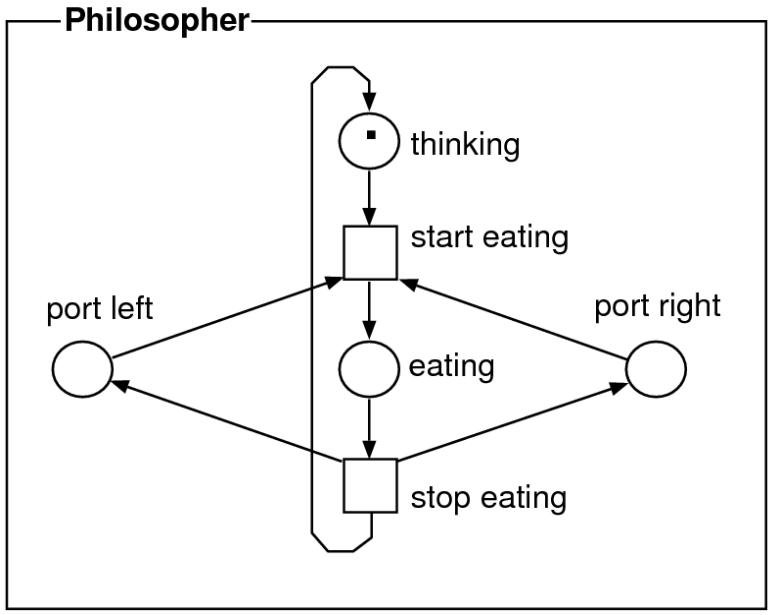


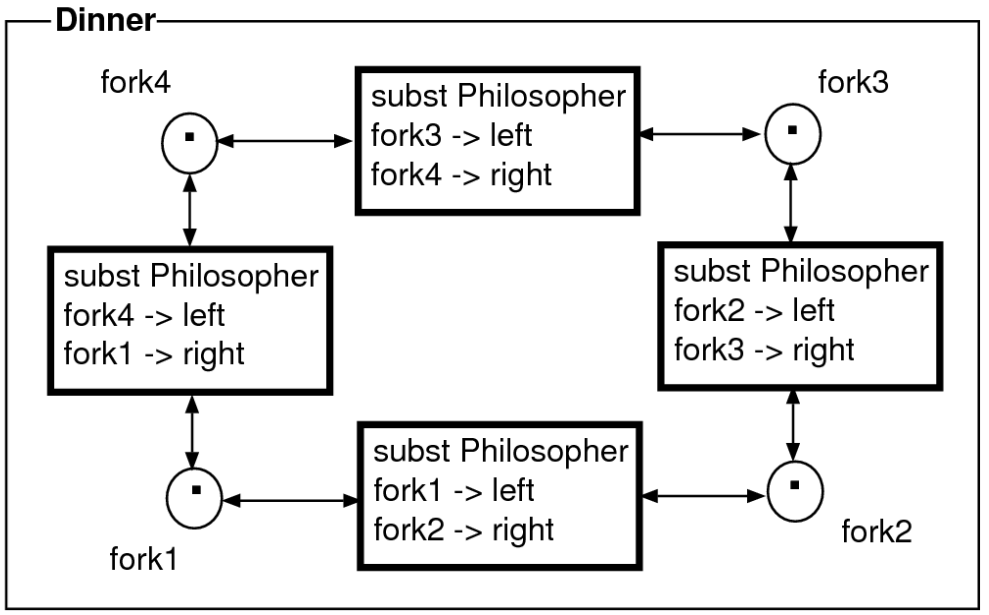




Příklad hierarchické Petriho sítě

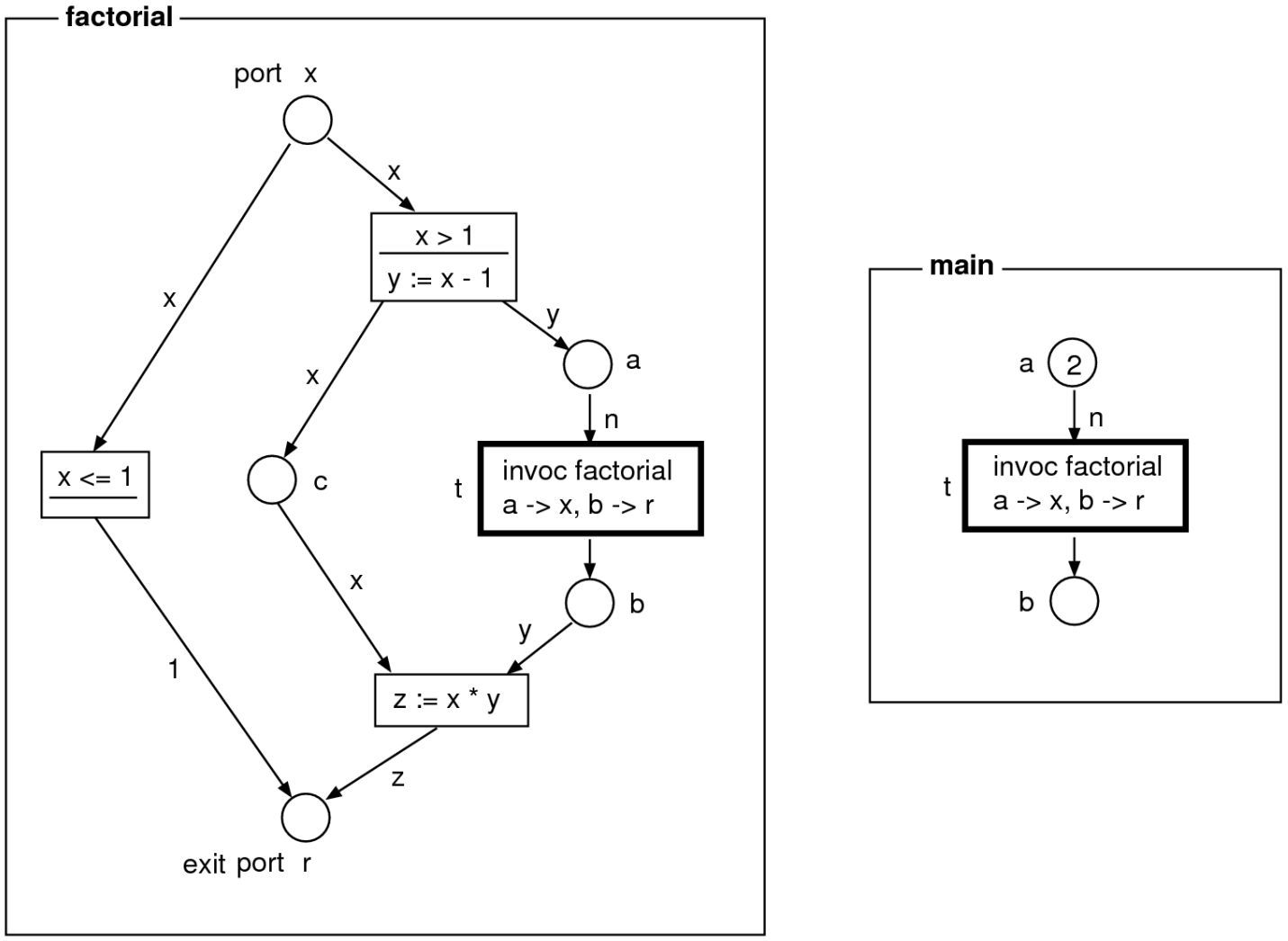
- Příklad: Večeřící filosofové





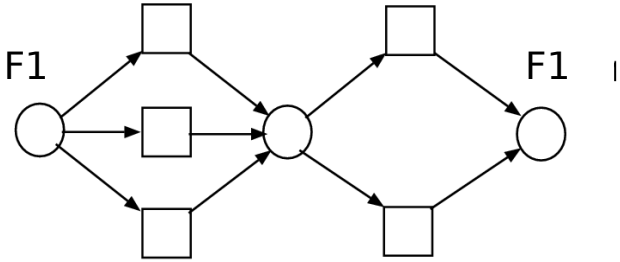
Invokační přechod

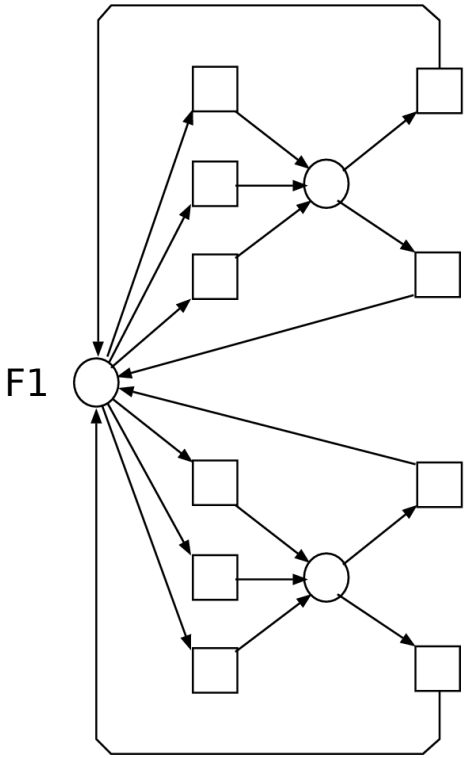
- Instance stránek vznikají dynamicky při provedení invokačního přechodu



Fúzní množina míst

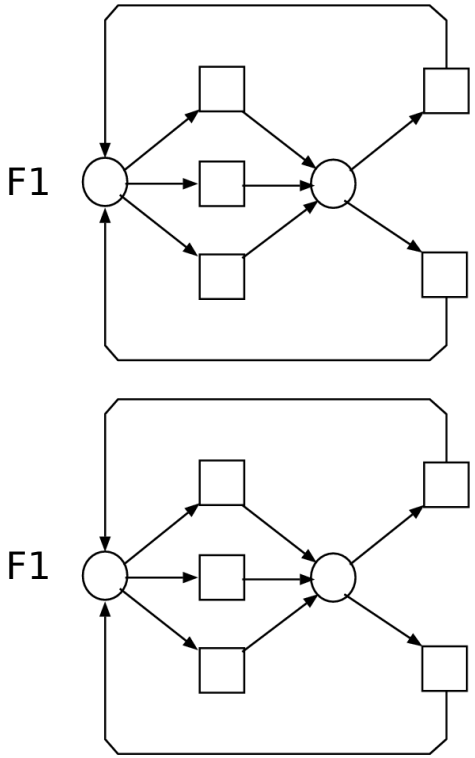
- Umožňuje asynchronní komunikaci mezi instancemi stránek
- Varianty: Instanční, stránková, globální





(a)

stránková
fúzní množina

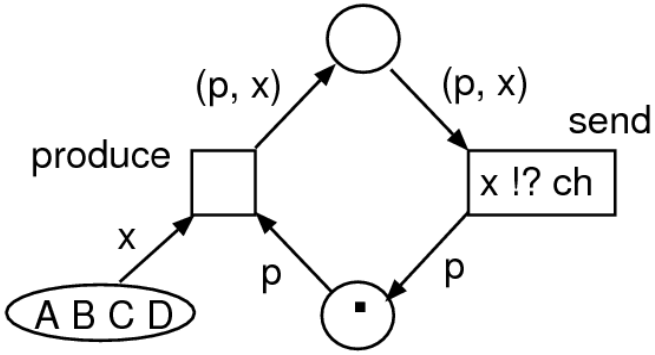


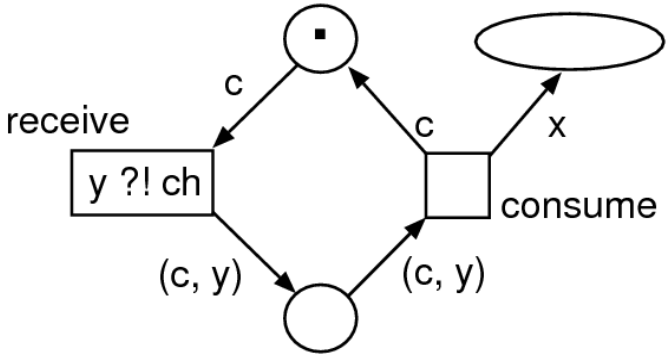
(b)

Instanční
fúzní množina

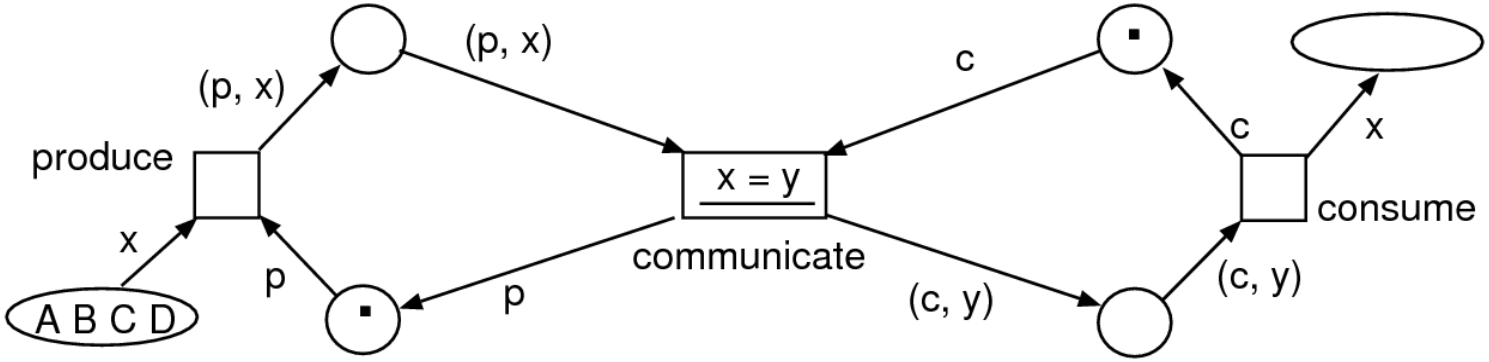
Synchronní kanály

- Globálně dostupné pojmenované sychronní kanály
- Umožňují synchronní komunikaci mezi různými instancemi stránek
- Přechody obsahují komunikační výrazy *expr!?ch*, resp. *expr?!ch.*



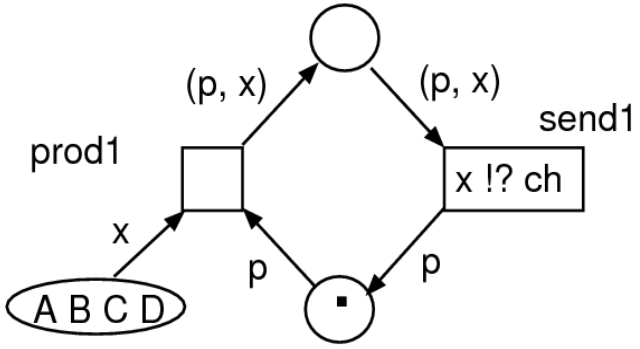


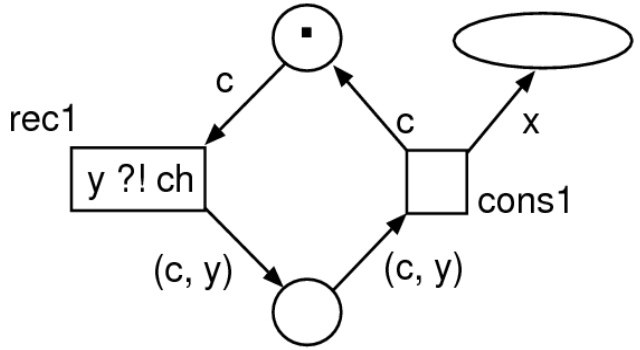
- Sémantika:

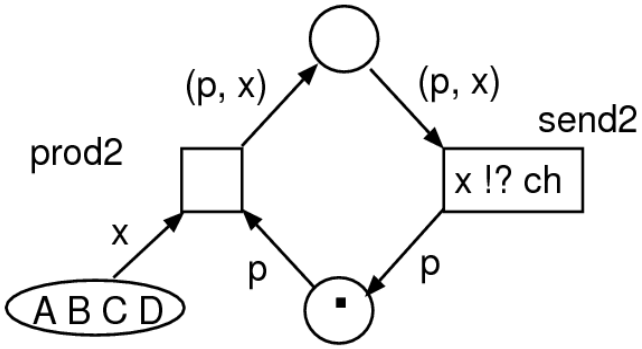


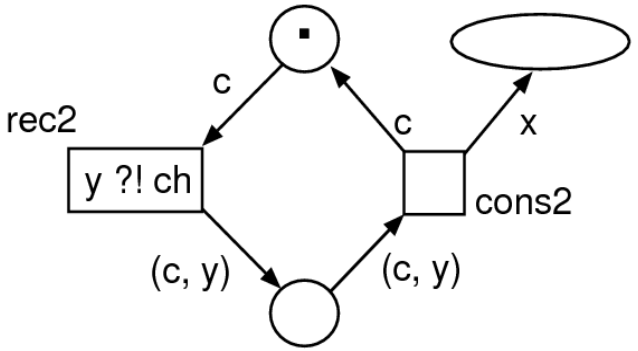
Synchronní kanály

- Stejný synchronní kanál může propojit více instancí stránek:



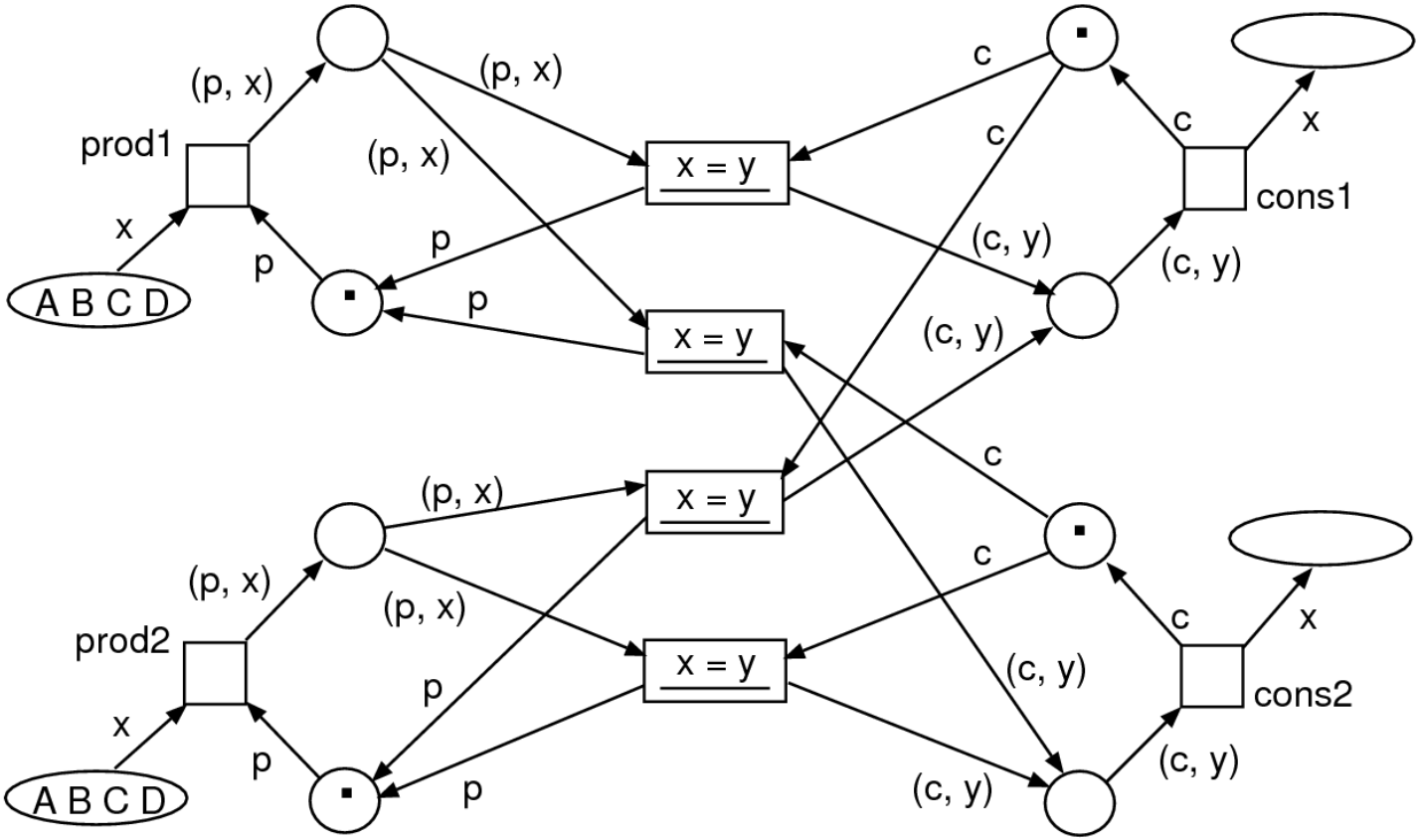






Synchronní kanály

- Sémantika synchronního kanálu



Víceúrovňové Petriho sítě

- Sítě v sítích (Nets in Nets)
  - R. Valk. **Petri Nets as Token Objects: An Introduction to Elementary Object Nets**. In *Jorg Desel, Manuel Silva (eds.): Application and Theory of Petri Nets*; Lecture Notes in Computer Science, volume 120. Springer-Verlag, 1998.

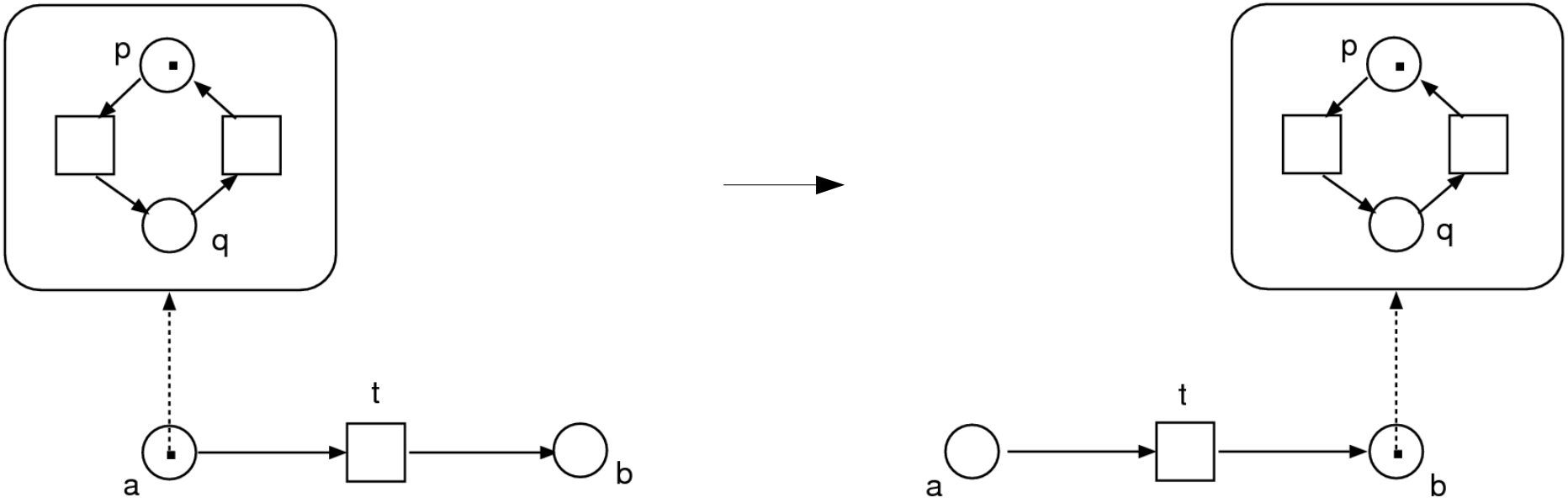
- Princip:
  - Značka v síti systému reprezentuje síť objektu
  - Přechody v různých sítích se mohou synchronizovat

- Varianty:
  - Síť systému a síť objektu (system net & object net)
  - Systém a množina objektů
  - Každý objekt může vystupovat jako systém pro jiné objekty

- Omezení
  - bez pozdní vazby, polymorfismu a dynamické instanciace

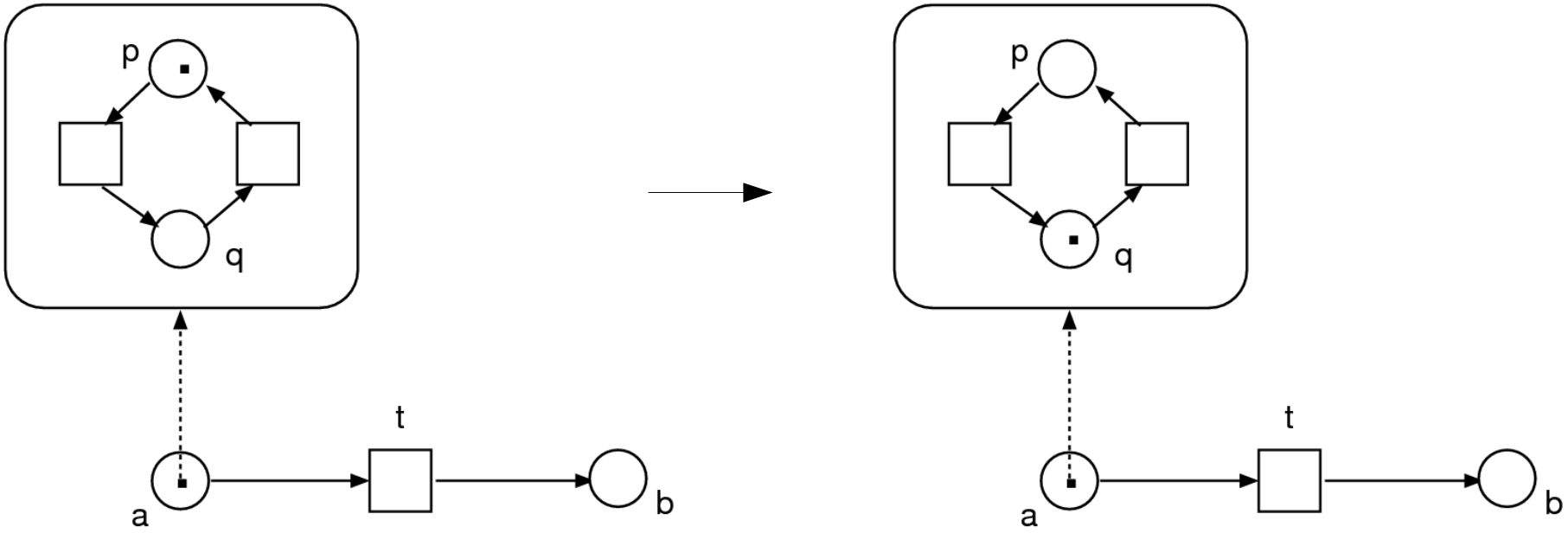
Princip dvouúrovňové sítě

- 1. Transport



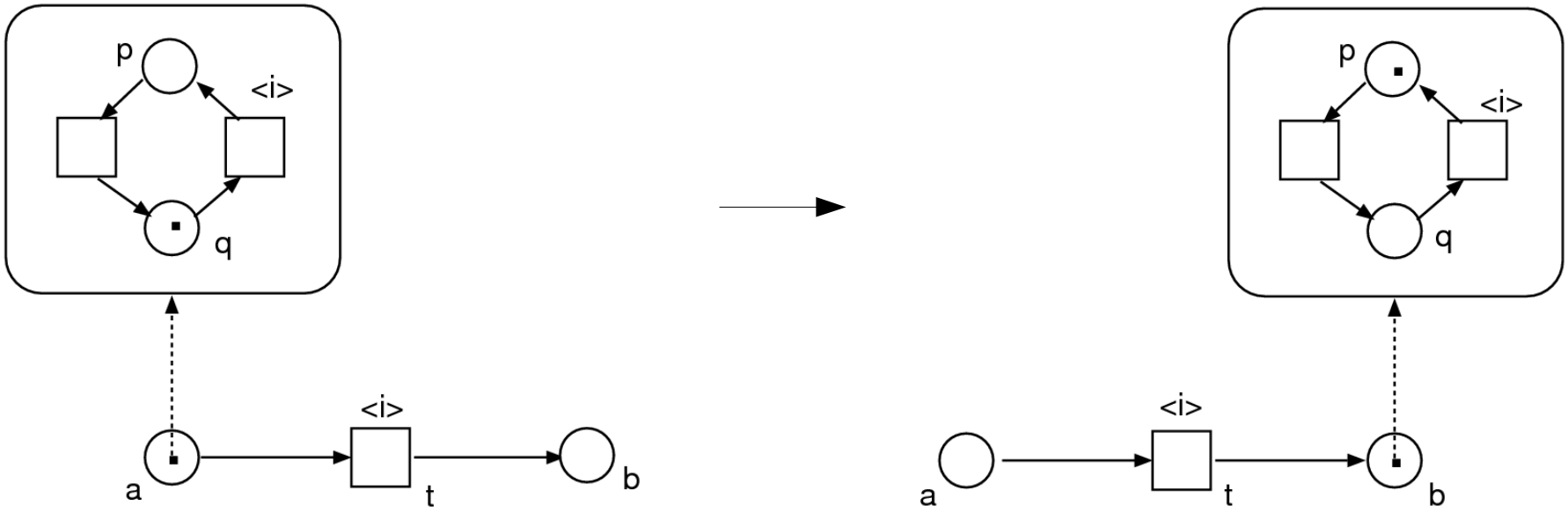
Princip dvouúrovňové sítě

- 2. Autonomní akce objektu



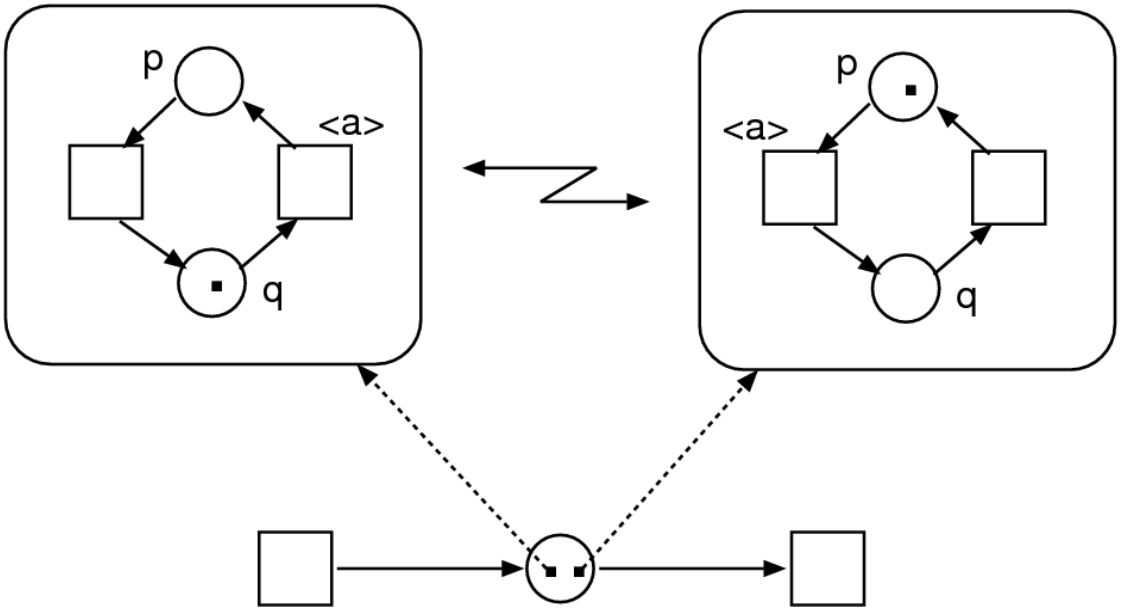
Princip dvouúrovňové sítě

- 3. Interakce sítě systému a sítě objektu
  - přechody se provedou synchronně



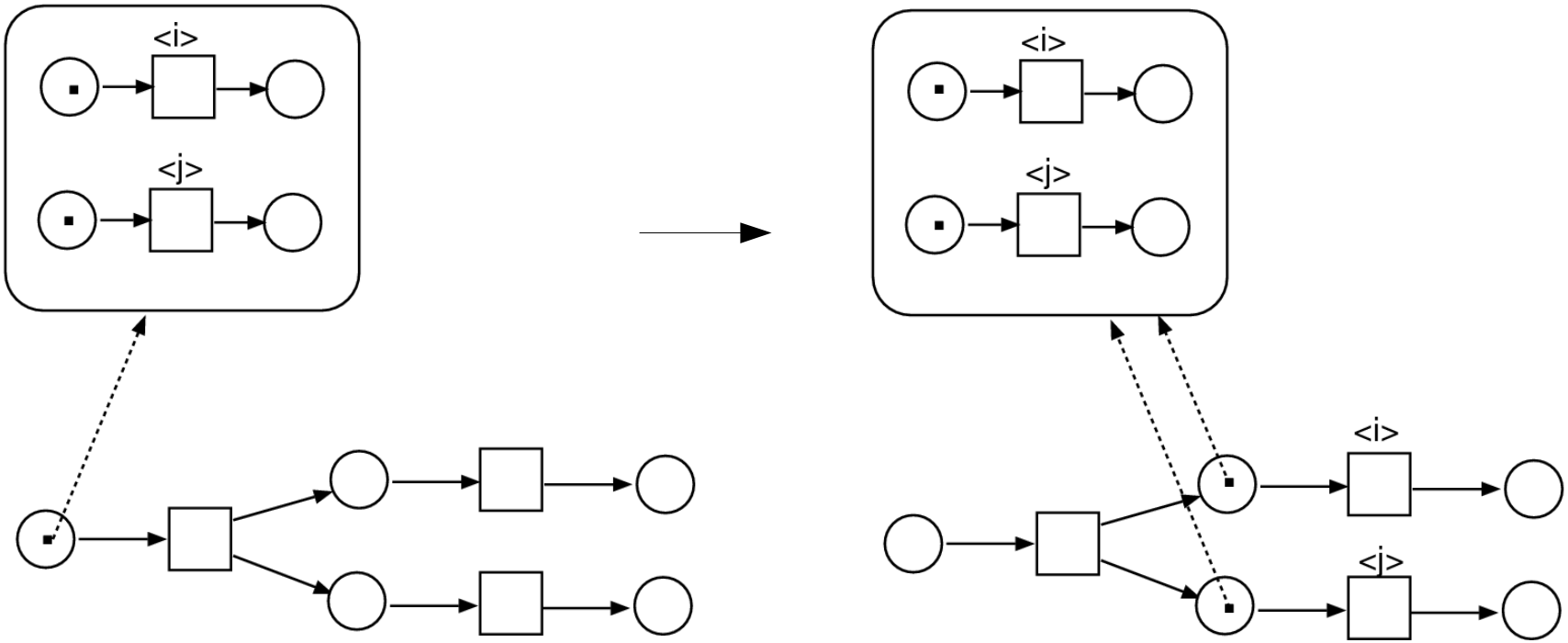
Princip dvouúrovňové sítě

- 4. Interakce objektů



Princip dvouúrovňové sítě

- 5. Distribuce referencí na objekty



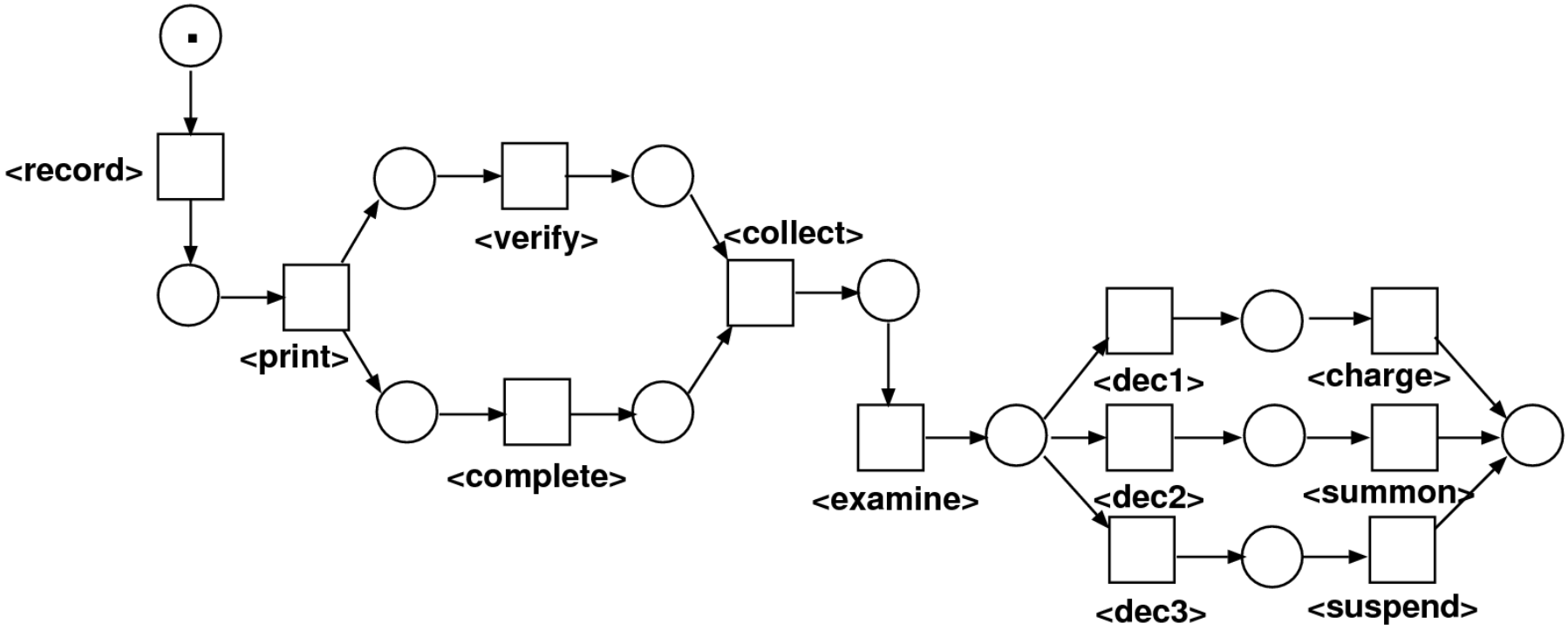
Aplikace

- Typické aplikace:
  - Agentní systémy – mobilní agenti
  - Komunikační protokoly
  - Řízení procesů a projektů
  - Výrobní systémy
  - Dopravní systémy

- Obvykle stačí 2 úrovně:
  - Síť systému
    - Modeluje organizační strukturu a toky dat mezi aktory
    - Značka reprezentuje zpracovávaný objekt (případ)
  - Síť objektu
    - Modeluje způsob zpracování

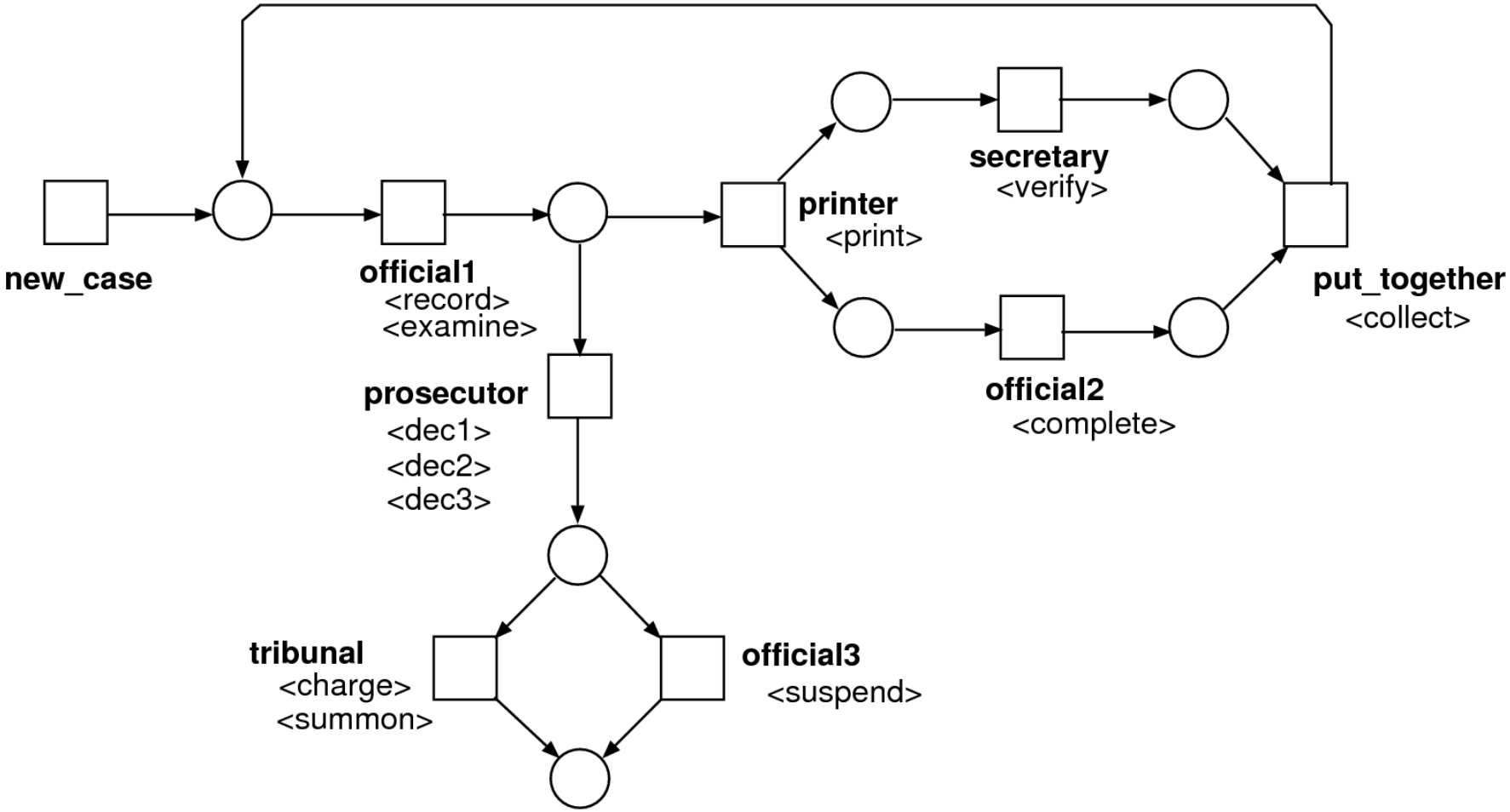
Příklad z oblasti soudnictví

- Kriminální případ



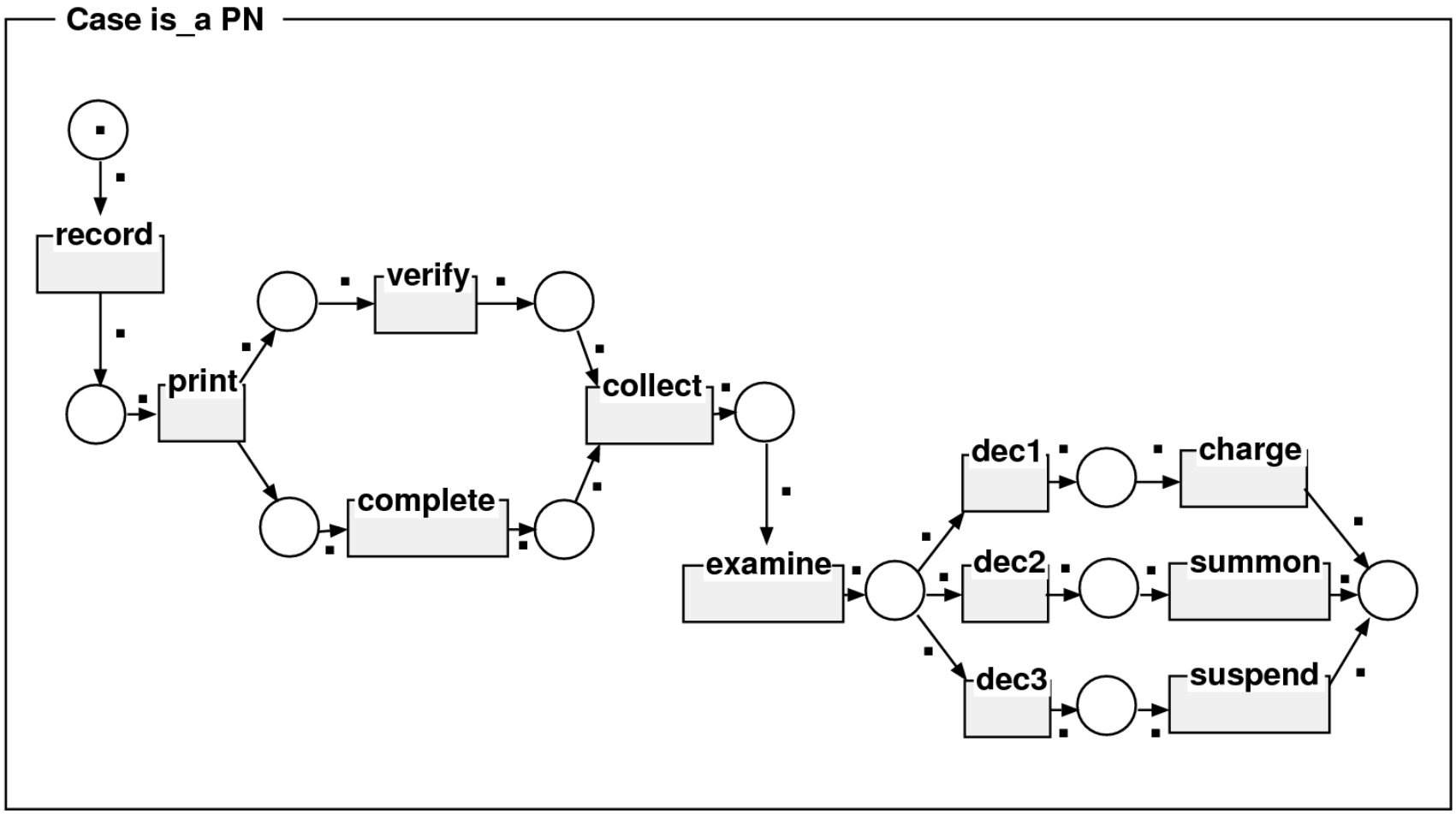
Příklad z oblasti soudnictví

- Organizační struktura pro řešení kriminálních případů

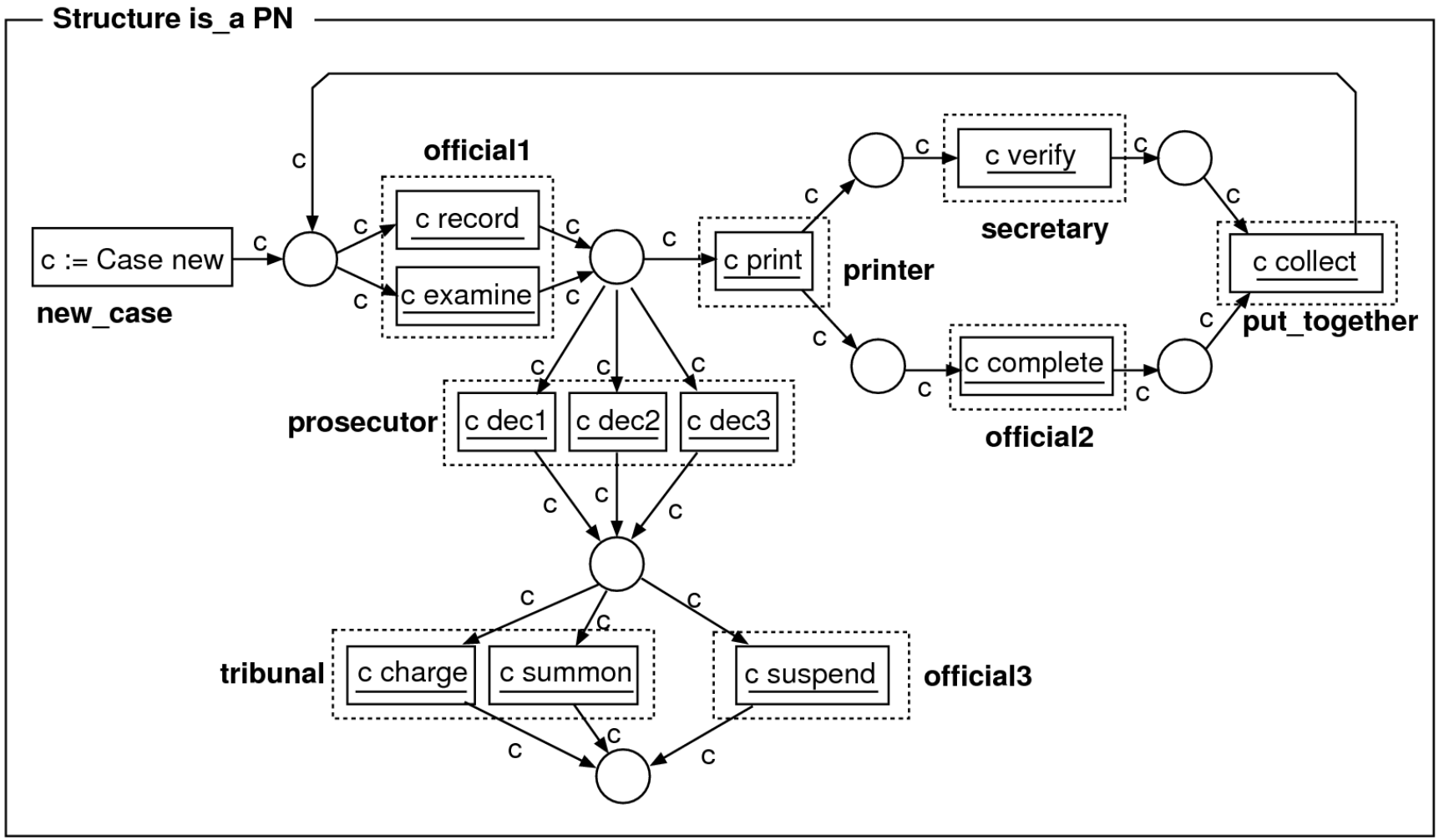


Implementace

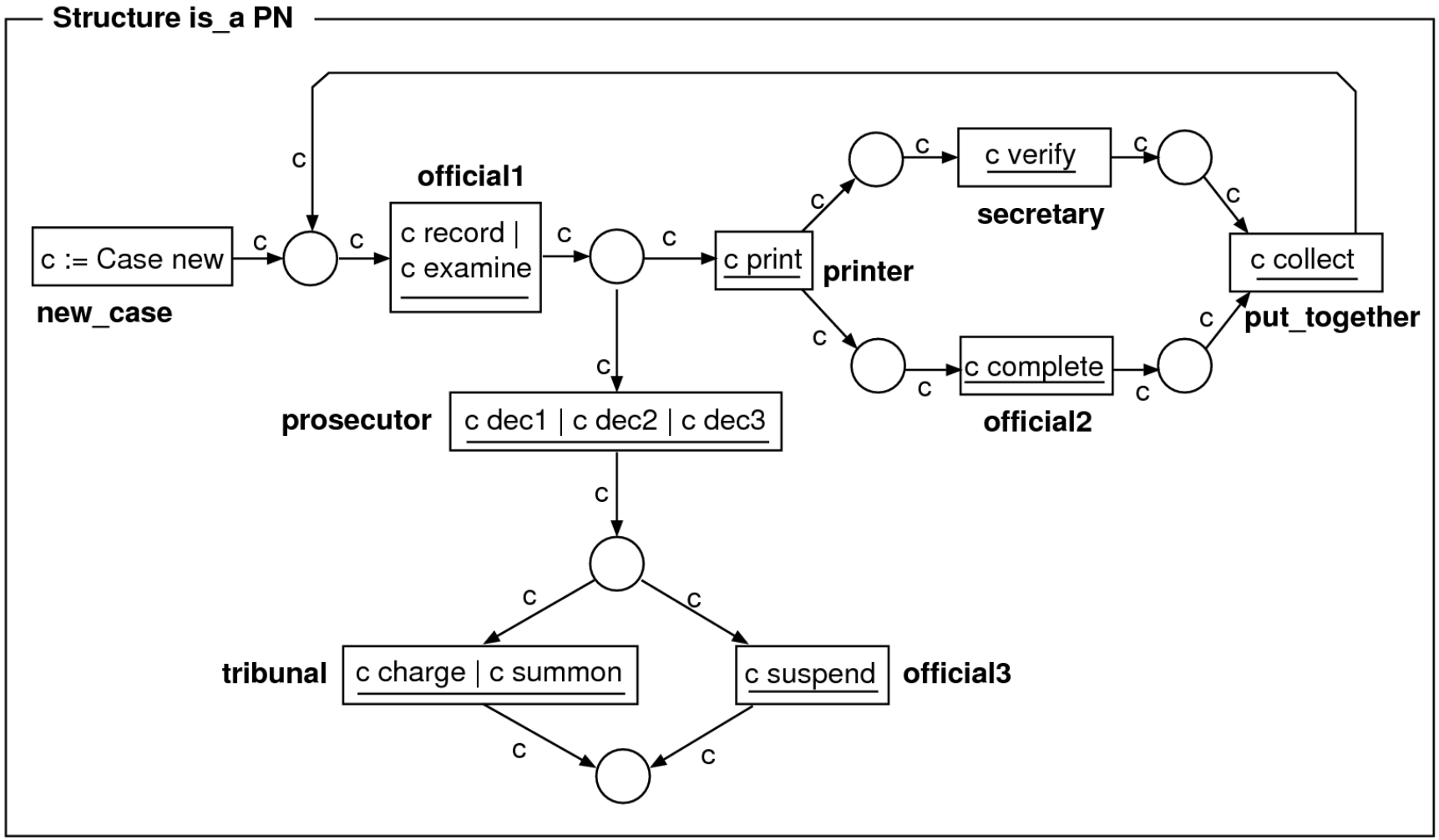
- Implementace v jazyce OOPN/PNtalk



Implementace



Implementace



Nástroje

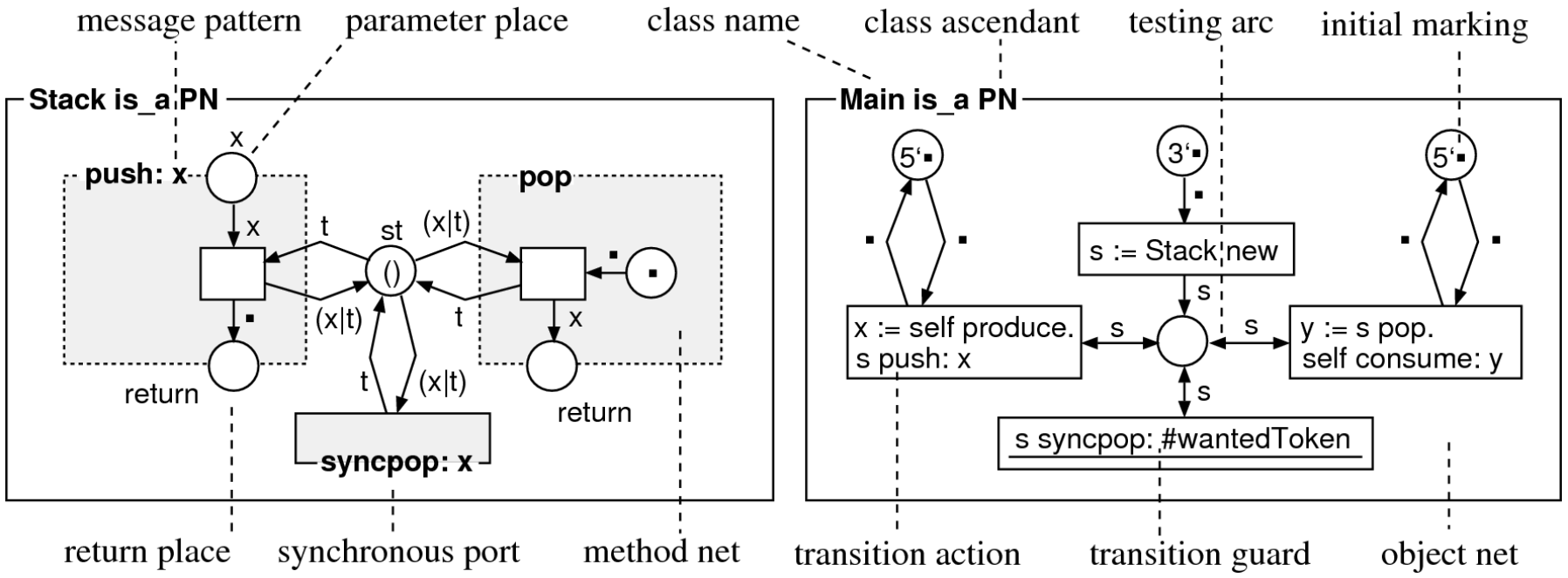
- Renew - Reference Nets Workshop (Hamburg University)
  - Objekty komunikující synchronními kanály
  - Interoperabilita s CPNTools, možnost analýzy
- OOPN/PNtalk (Brno University of Technology)
  - Objekty komunikují synchronními porty a invokacemi sítí metod
  - Interoperabilita se simulačními nástroji (DEVS) a Smalltalkem
- Vzájemná interoperabilita - PNML

Objektově orientované Petriho sítě

- OOPN ke známým strukturovacím a komunikačním mechanismům přidává polymorfismus, pozdní vazbu a bezešvou propojitelnost s OO jazykem, do kterého může být OOPN vnořena a který může být současně i jejím inskripčním jazykem.

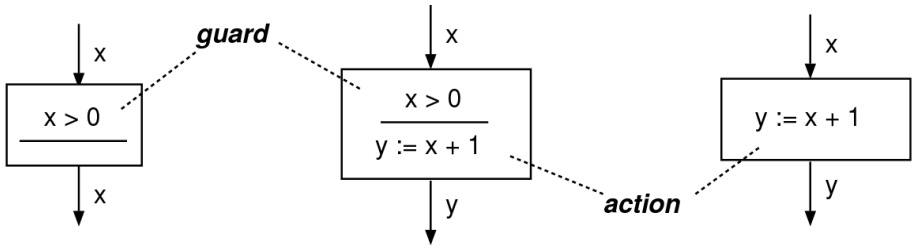
- Na rozdíl od Renew umožňuje invokaci sítí metod, které mohou realizovat neatomiceké operace.

- OOPN je množina tříd definovaných třídy pomocí HLPN. Třída obsahuje
  - Síť objektu
  - Množinu dynamicky instanciovatelných sítí metod
  - Množinu synchronních portů
- Místa sítě objektu jsou dostupná pro přechody sítí metod
- Značky v místech jsou reference na objekty
- Objekty jsou definovány třídami (kromě primitivních objektů)



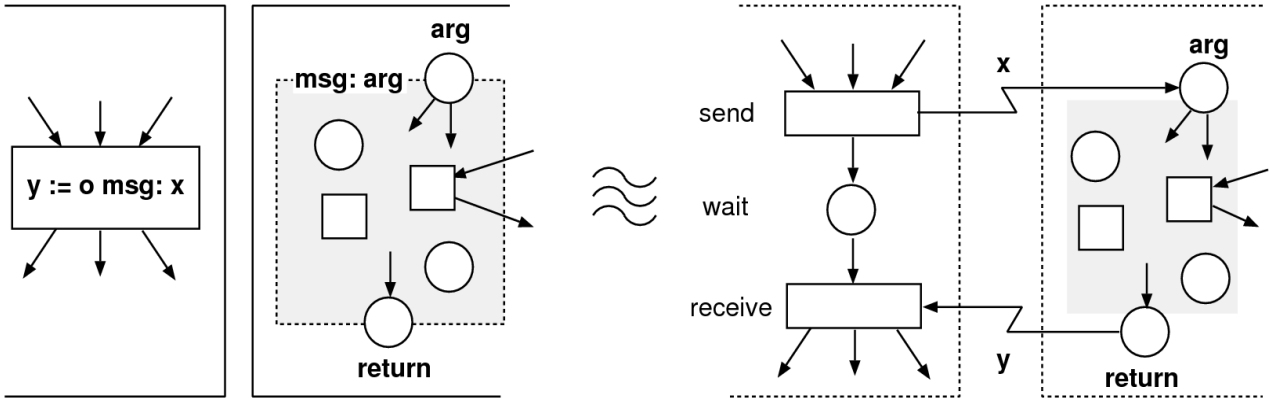
Interakce neprimitivních objektů

- Stráž a akce přechodu specifikuje zasílální zpráv objektům



- Invokace synchronního portu

- Invokace metody



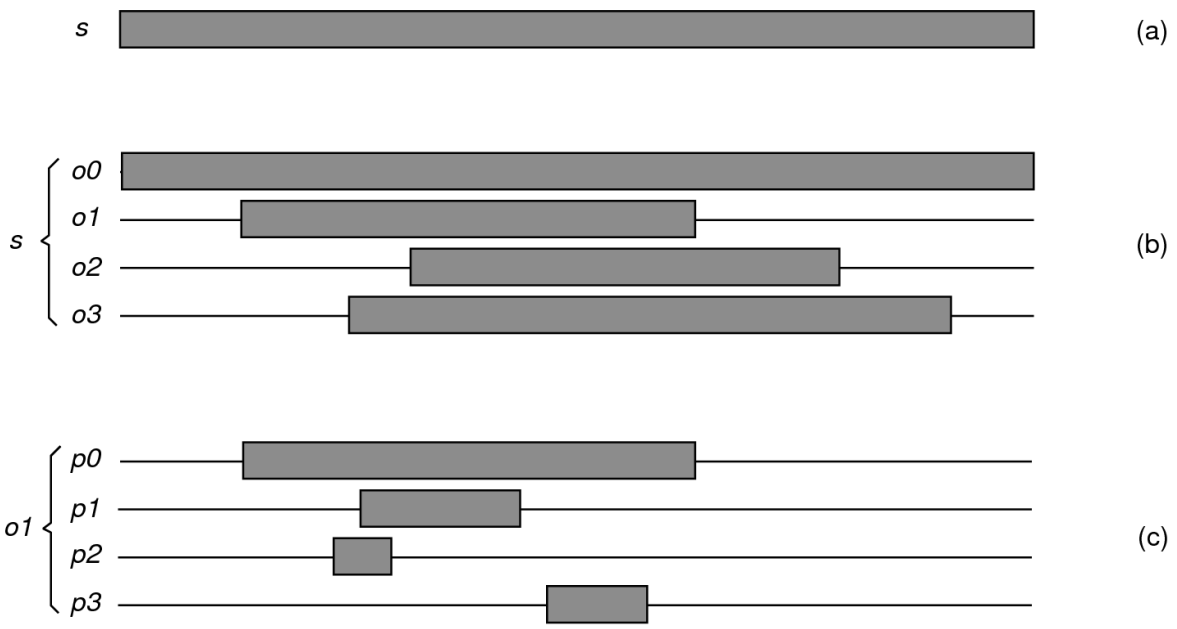
- Pozdní vazba

Dynamika OOPN

- Systém obsahuje objekty
  - Objekty mohou dynamicky vznikat a zanikat
- Objekt obsahuje instanci sítě objektu a množinu instancí sítí metod
  - Instance metod mohou dynamicky vznikat a zanikat

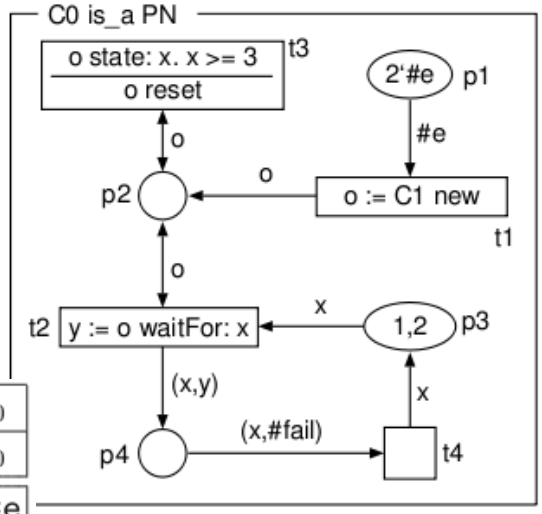
- Dynamika systému
  – změny stavu
  – 4 typy událostí:
    - **A** – interní událost
    - **N** – vytvoření objektu
    - **F** – invokace metody
    - **J** – ukončení metody
    - Garbage collector
- Krok simulace
  - $S[e, id, t, b\rangle S'$

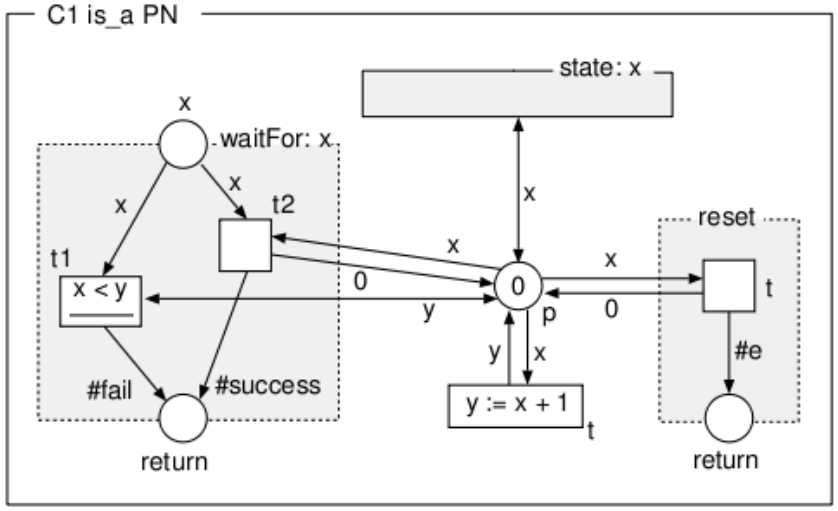
- Dekompozice systému a jeho vývoj v čase:



Simulace OOPN

- Počáteční stav





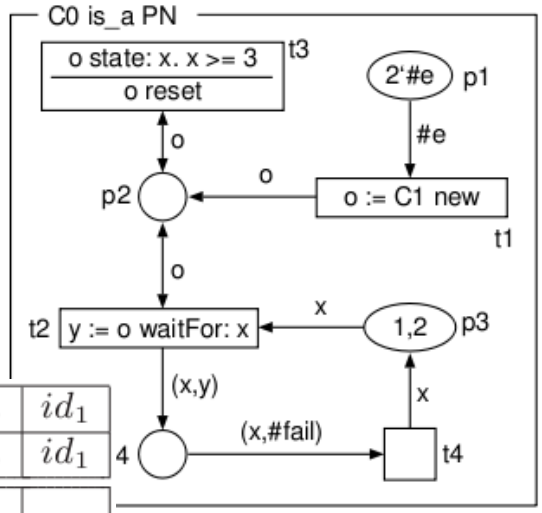
S_0			id_0
			id_0
C0	*object net*	p1	2'#e
		p2	empty
		p3	1, 2
		p4	empty
		t1	empty
		t2	empty
		t3	empty
C1	*object net*	p	
		t	
	waitFor:	x	
		return	
		t1	
		t2	
	reset	return	
		t	

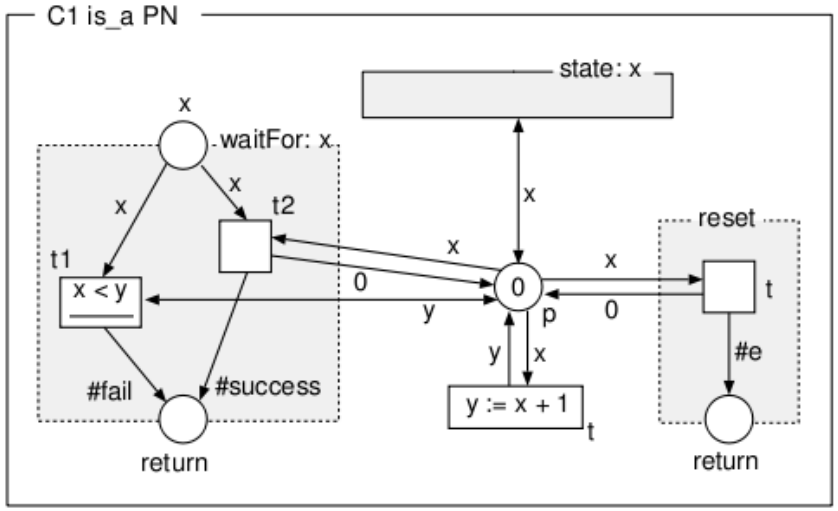
Proveďme nyní sekvenci kroků:

$$s_0\ [(N, id_0, C0.t1, ())\rangle$$
$$s_1\ [(F, id_0, C0.t2, (x = 1, o = id_1))\rangle$$
$$s_2\ [(F, id_0, C0.t2, (x = 2, o = id_1))\rangle$$
$$s_3\ [(A, id_1, C1.t, (x = 0))\rangle$$
$$s_4\ [(A, id_2, C1.waitFor : .t2, (x = 1))\rangle\ s_5$$

Simulace OOPN

- Stav S1





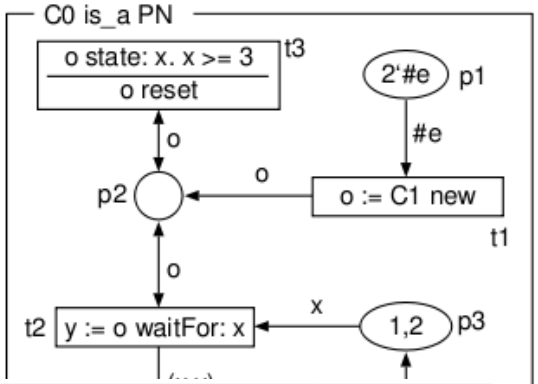
S_1			id_0	id_1
			id_0	id_1
C0	*object net*	p1	#e	
		p2	id_1	
		p3	1, 2	
		p4	empty	
		t1	empty	
		t2	empty	
		t3	empty	
C1	*object net*	p		0
		t		empty
	waitFor:	x		
		return		
		t1		
		t2		
	reset	return		
		t		

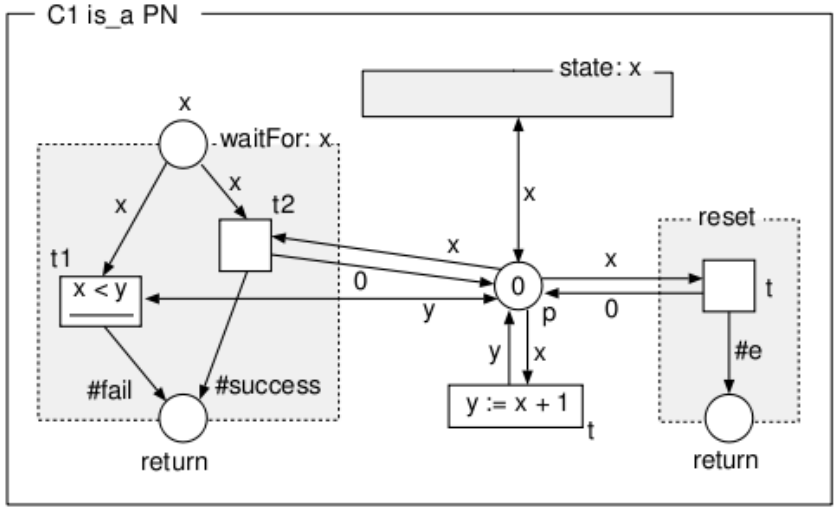
Stav po provedení

$$s_0\ [(N, id_0, C0.t1, ())\rangle$$

Simulace OOPN

- Stav S2





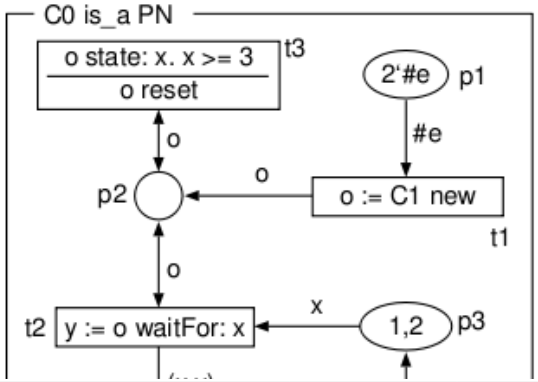
S_2			id_0	id_1	
			id_0	id_1	id_2
C0	*object net*	p1	#e		
		p2	id_1		
		p3	2		
		p4	empty		
		t1	empty		
		t2	$(id_2, \{(x, 1), (o, id_1)\})$		
		t3	empty		
C1	*object net*	p		0	
		t		empty	
	waitFor:	x			1
		return			empty
		t1			empty
		t2			empty
	reset	return			
		t			

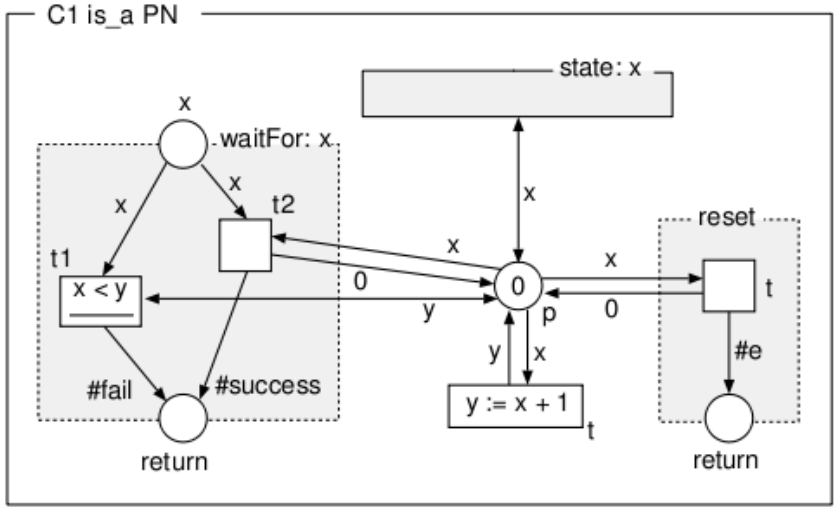
Stav po provedení

$$s_0\ [(N, id_0, C0.t1, ())\rangle$$
$$s_1\ [(F, id_0, C0.t2, (x = 1, o = id_1))\rangle$$

Simulace OOPN

- Stav S3





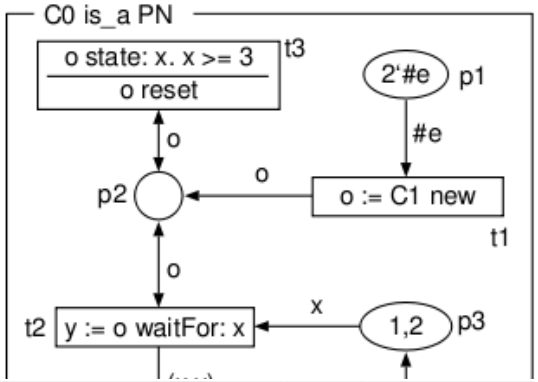
S_3			id_0	id_1		
			id_0	id_1	id_2	id_3
C0	*object net*	p1	#e			
		p2	id_1			
		p3	empty			
		p4	empty			
		t1	empty			
		t2	$(id_2, \{(x, 1), (o, id_1)\})$, $(id_3, \{(x, 2), (o, id_1)\})$			
		t3	empty			
C1	*object net*	p		0		
		t		empty		
	waitFor:	x			1	2
		return			empty	empty
		t1			empty	empty
		t2			empty	empty
	reset	return				
		t				

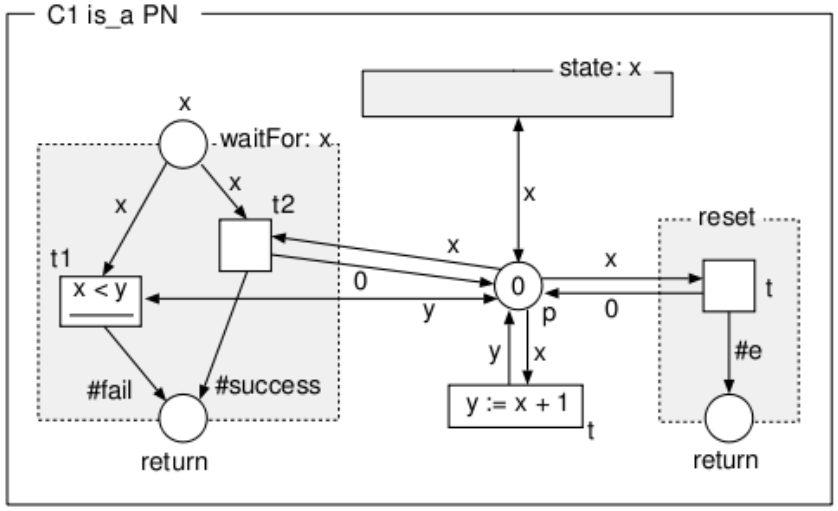
Stav po provedení

$$s_0\ [(N, id_0, C0.t1, ())\rangle$$
$$s_1\ [(F, id_0, C0.t2, (x = 1, o = id_1))\rangle$$
$$s_2\ [(F, id_0, C0.t2, (x = 2, o = id_1))\rangle$$

Simulace OOPN

- Stav S4





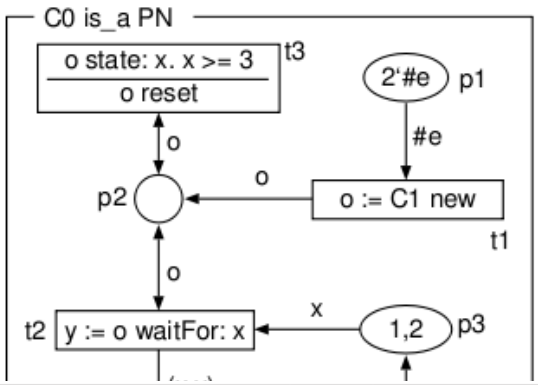
S_4			id_0	id_1		
			id_0	id_1	id_2	id_3
C0	*object net*	p1	#e			
		p2	id_1			
		p3	empty			
		p4	empty			
		t1	empty			
		t2	$(id_2, \{(x, 1), (o, id_1)\})$, $(id_3, \{(x, 2), (o, id_1)\})$			
		t3	empty			
C1	*object net*	p		1		
		t		empty		
	waitFor:	x			1	2
		return			empty	empty
		t1			empty	empty
		t2			empty	empty
	reset	return				
		t				

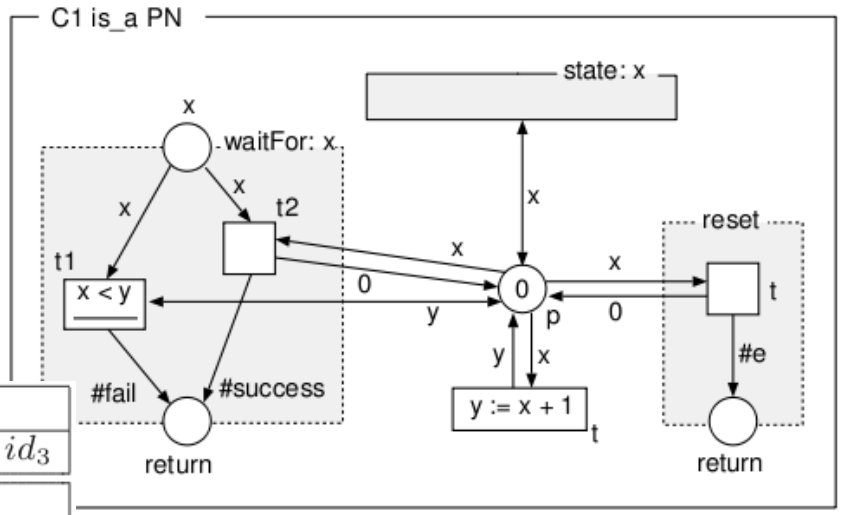
Stav po provedení

$$s_0\ [(N, id_0, C0.t1, ())\rangle$$
$$s_1\ [(F, id_0, C0.t2, (x = 1, o = id_1))\rangle$$
$$s_2\ [(F, id_0, C0.t2, (x = 2, o = id_1))\rangle$$
$$s_3\ [(A, id_1, C1.t, (x = 0))\rangle$$

Simulace OOPN

- Stav S5





S_5			id_0	id_1		
			id_0	id_1	id_2	id_3
C0	*object net*	p1	#e			
		p2	id_1			
		p3	empty			
		p4	empty			
		t1	empty			
		t2	$(id_2, \{(x, 1), (o, id_1)\})$, $(id_3, \{(x, 2), (o, id_1)\})$			
		t3	empty			
C1	*object net*	p		0		
		t		empty		
	waitFor:	x			empty	2
		return			#success	empty
		t1			empty	empty
		t2			empty	empty
	reset	return				
		t				

Stav po provedení

$s_0\ [(N, id_0, C0.t1, ())\rangle$
$s_1\ [(F, id_0, C0.t2, (x = 1, o = id_1))\rangle$
$s_2\ [(F, id_0, C0.t2, (x = 2, o = id_1))\rangle$
$s_3\ [(A, id_1, C1.t, (x = 0))\rangle$
$s_4\ [(A, id_2, C1.waitFor : .t2, (x = 1))\rangle\ s_5$

Fyzický čas

- Čas lze do Petriho sítí zavést různě, nečastěji takto:
  - Zpožďující přechody – výstupní část přechodu se provede až po specifikované době.
  - Provedení je možné až poté, co byl přechod nepřetržitě proveditelný po specifikovanou dobu.
  - Značky s časovými razítky – značka při umístění do místa dostane časové razítko, specifikující, jak dlouho musí v místě “zrát”, než ji může libovolný přechod odebrat.

Příklad: Procesy a alokace zdrojů

- Aktivita procesu (přechod)
  - Pokusí se alokovat zdroj
  - Uspěje-li, použije ho