Ingegneria dei requisiti

Requisiti

Proprietà (funzionali e non) che l'applicazione dovrà avere, descrivono

CHE COSA il sistema dovrà fare piuttosto che COME lo dovrà fare,

sono focalizzati sul

PROBLEMA non sulla SOLUZIONE

(Vedi capitolo 5 Sommerville)

Requirements engineering

The process of establishing requirements, i.e. the SERVICES that the customer requires from a system and the CONSTRAINTS under which it operates and is developed

What is a requirement?

- Requirements may serve a dual function
  - May be the basis for a bid for a contract - therefore must be open to interpretation
  - May be the basis for the contract itself - therefore must be defined in detail
  - Both these statements may be called requirements

Types of requirement

- User requirements
  - Statements in natural language plus diagrams of the services the system provides and its operational constraints. Written for customers
- System requirements
  - A structured document setting out detailed descriptions of the system services. Written as a contract between client and contractor
- Software specification
  - A detailed software description which can serve as a basis for a design or implementation. Written for developers

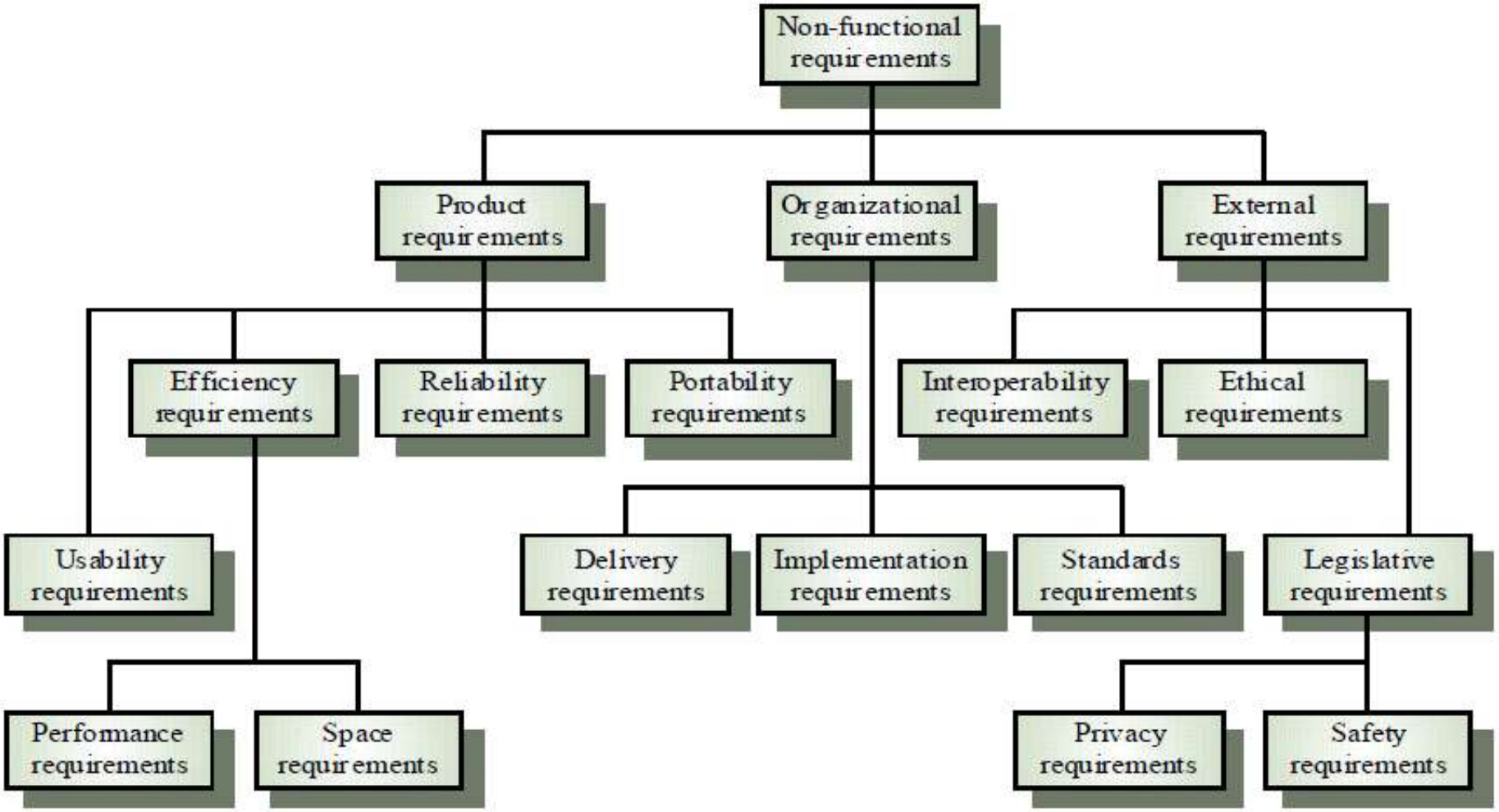
Functional and non-functional requirements

- Functional requirements
  - Statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.
- Non-functional requirements
  - Constraints on the services or functions offered by the system such as timing constraints, constraints on the development process, standards, etc.

Non-functional requirements

- Define system properties and constraints e.g. reliability, response time and storage requirements. Constraints are I/O device capability, system representations, etc.
- Process requirements may also be specified mandating a particular CASE system, programming language or development method
- Non-functional requirements may be more critical than functional requirements. If these are not met, the system is useless

Non-functional requirement types



Elicitazione e analisi dei requisiti

- L'elicitazione dei requisiti è una attività critica per la riuscita del progetto: errori introdotti in questa fase hanno un costo enorme se scoperti troppo tardi
- I requisiti sono uno strumento molto importante di comunicazione con il committente
- L'analisi ha luogo quando l'esperto del dominio è presente, altrimenti è pseudoanalisi
- Uno degli elementi più importanti quando si gestiscono i rischi legati ai requisiti è avere accesso alla conoscenza degli esperti del dominio
- La mancanza di contatto con gli esperti è una delle cause più comuni di fallimento di un progetto

(Vedi capitolo 6 Sommerville)

Problems of requirements analysis

- Stakeholders don't know what they really want
- Stakeholders express requirements in their own terms
- Different stakeholders may have conflicting requirements
- Organisational and political factors may influence the system requirements
- The requirements change during the analysis process. New stakeholders may emerge and the business environment change

Caratteristiche dei requisiti

1. Validità – ogni requisito esprime qualcosa di cui l'utente ha realmente bisogno
2. Correttezza – la descrizione di ciascun requisito non contiene errori
3. Consistenza – non ci sono conflitti tra i requisiti
4. Completezza (esterna e interna) – tutti gli stati, le funzionalità, gli input, gli output e i vincoli sono contemplati da qualche requisito
5. Realismo – effettiva realizzabilità
6. Comprensibilità e univocità di interpretazione
7. Tracciabilità – riconducibilità alle ragioni / origini
8. Modificabilità
9. Verificabilità – quando si formula un requisito, si deve stabilire come decidere a posteriori se esso è stato realizzato dal sistema oppure no

Completezza dei requisiti

- Interna

Ogni nuovo concetto o vocabolo utilizzato deve essere definito (ad es. mediante un glossario)

- Esterna

La specifica deve documentare tutti i requisiti necessari
Difficoltà: quando ci si ferma?

Requirements measures

Property	Measure
Speed	Processed transactions/second User/Event response time Screen refresh time
Size	K Bytes Number of RAM chips
Ease of use	Training time Number of help frames
Reliability	Mean time to failure Probability of unavailability Rate of failure occurrence Availability
Robustness	Time to restart after failure Percentage of events causing failure Probability of data corruption on failure
Portability	Percentage of target dependent statements Number of target systems

Requisiti disambigui, comprensibili, precisi

Esempio (da non imitare) di un frammento di specifica di un word-processor

La selezione è il processo attraverso cui si designano aree del documento su cui si vuole lavorare. La maggior parte delle azioni di editazione e formattazione richiedono due passi: prima si seleziona ciò su cui si vuole lavorare, come testo o grafici, quindi si inizia l'azione appropriata.

Un altro esempio (da non imitare), relativo un sistema in tempo reale safety-critical

Il messaggio deve essere triplicato. Le tre copie devono essere inoltrate attraverso tre diversi canali fisici. Il ricevente accetta il messaggio sulla base di una politica di voto due-su-tre.

Requisiti logicamente consistenti

Esempio (da non imitare) di un frammento di specifica di un word-processor

L'intero testo dovrebbe essere mantenuto in linee di uguale lunghezza. La lunghezza è specificata dall'utente.
A meno che l'utente non dia un comando esplicito di sillabazione, un ritorno a capo dovrebbe avvenire solo alla fine di una parola.

Documento dei requisiti in linguaggio naturale

I requisiti sono organizzati e numerati univocamente (dot notation) per categorie generali, suddivise ricorsivamente in sottocategorie, fino ad arrivare ai singoli requisiti atomici. La decomposizione riflette la prospettiva del cliente circa il sistema commissionato

Esempio:

1 Categoria
1.1 Sottocategoria
1.1.1 Requisito 1 della sottocategoria 1 della categoria 1
1.1.2 Requisito 2 della sottocategoria 1 della categoria 1
2 Categoria
……

Documento dei requisiti in linguaggio naturale (cont.)

Ogni azienda si dota del proprio formato per il documento dei requisiti, che tipicamente, per ogni requisito, include le seguenti informazioni:

1. Numero identificatore unico (es. 1. 2. 5)
2. Descrizione sintetica (una riga, es. stampa fattura)
3. Descrizione dettagliata
4. Ingressi: tipo e provenienza
5. Uscite: tipo e destinazione
6. Interazioni con l'utente
7. Altre entità coinvolte
8. Pre-condizioni
9. Post-condizioni
10. Effetti collaterali
11. Requisiti attinenti (see also…)

Documento dei requisiti in linguaggio naturale (cont.)

Esempio (1).

ID 3.5.1	
Function	Add node
Description	Adds a node to an existing design. The user selects the type of node, and its position. When added to the design, the node becomes the current selection. The user chooses the node position by moving the cursor to the area where the node is added.
Inputs	Node type, Node position, Design identifier.
Source	Node type and Node position are input by the user, Design identifier from the database.
Outputs	Design identifier.
Destination	The design database. The design is committed to the database on completion of the operation.

Documento dei requisiti in linguaggio naturale (cont.)

Esempio (2).

Requires	Design graph rooted at input design identifier.
Pre-condition	The design is open and displayed on the user's screen.
Post-condition	The design is unchanged apart from the addition of a node of the specified type at the given position.
Side-effects	None

Specifica

- termine ampio che significa “definizione”
- usato in fasi diverse del processo di sviluppo del sw con scopi diversi: un processo può essere visto come una catena di passi di specifica (cioè definizione) - implementazione - verifica

sw specification = definizione del comportamento agli effetti esterni; rappresenta un accordo fra analisti e progettisti; va verificato rispetto a user e system requirements (che rappresentano un accordo fra cliente e contraente)

design specification = definizione dell’architettura del sw; rappresenta un accordo fra progettisti e programmatori; va verificato rispetto a sw specification

Specifica dei requisiti

Tipologie di applicazioni (o di parti di esse)	Requisiti da specificare
Sequenziali (unico flusso di controllo)	Funzionalità offerte
Concorrenti (più flussi paralleli di controllo + meccanismi di sincronizzazione per l'accesso a risorse condivise)	Attività parallele, risorse condivise, meccanismi di sincronizzazione
Dipendenti dal tempo (dette "in tempo reale": la correttezza dei risulti dipende dal tempo di esecuzione delle attività)	Tempi di esecuzione

Linguaggi di specifica

NON esiste un linguaggio di specifica adatto per qualunque problema o classe di applicazioni

Ulteriore categorizzazione delle applicazioni (o di parti di esse)	Requisiti da specificare	Esempi di linguaggi di specifica
Orientate ai dati (es. sistemi informativi)	Struttura concettuale dei dati	Modello ER
Orientate alle funzioni (es. traduttori)	Funzioni e flusso dei dati	Diagrammi di flusso (DFD)
Orientate al controllo (es. applicazioni dipendenti dal tempo che interagiscono con l'ambiente esterno)	Attività parallele, risorse condivise, tempi di esecuzione	Automi a stati finiti, reti di Petri, diagrammi di stato UML (statechart), diagrammi di attività

Linguaggi di specifica: classificazione

Specifiche	Esempi di linguaggi di specifica	Note
Informali	linguaggio naturale (che è impreciso, ambiguo e ridondante)	
Formali: in formalismo matematico (che obbliga alla precisione, è passibile di manipolazione automatica, talvolta consente di provare automaticamente consistenza e completezza, può essere eseguito)	Z (sequenziale), Reti di Petri (concorrente), FSM	Una specifica eseguibile costituisce un prototipo → può essere convalidata da committenti/utenti
Semiformali: notazioni (più o meno) rigorose, spesso grafiche + annotazioni in linguaggio naturale	ER, DFD, casi d'uso, diagrammi UML	

Linguaggi di specifica

Notazione =
sintassi del linguaggio di modellazione (aspetto grafico nel caso di linguaggi grafici; la sintassi UML è descritta da un meta-modello in UML)

Notazioni semiformali:
il loro significato non è univoco, bensì viene lasciato all'interpretazione dell'utente o dello strumento CASE generatore di codice (es. Together)

Linguaggi di specifica: ulteriore classificazione

- Operazionali
  Specifica del comportamento desiderato attraverso una macchina astratta

- Descrittivi/dichiarativi
  Specifica del comportamento desiderato attraverso le proprietà dello stesso

Specifiche operazionali e descrittive: un esempio

Specifica di una figura geometrica E

- Operazionale

```
E può essere disegnata come segue:
1) Selezionare due punti su un foglio solidale a un piano
   rigido
2) Inserire due perni in tali punti
3) Prendere una cordicella e fissarne le estremità ai perni
4) Tendere la cordicella mediante una matita, appoggiando
   la punta della matita sul foglio, segnando così il punto P
5) Muovere la matita in senso orario, mantenendo la punta
   sul foglio e la cordicella tesa, fino a raggiungere il
   punto P
```

- Descrittiva

$a\ x^2 + b\ y^2 + c = 0$

dove a, b e c sono costanti appropriate

Specifiche operazionali e descrittive: un altro esempio

- Operazionale

Sia a una sequenza di n elementi. Il risultato del suo ordinamento è una sequenza b di n elementi tale che il primo elemento di b è il minimo di a (se più elementi hanno lo stesso valore, ognuno di essi è accettabile); il secondo elemento di b è il minimo degli (n – 1) elementi ottenuti togliendo da a il suo elemento minimo, e così via, finché tutti gli n elementi sono stati tolti da a.

- Descrittiva

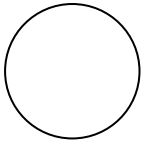
Il risultato dell'ordinamento di una sequenza a è una sequenza b che costituisce una permutazione di a tale per cui ogni elemento assume un valore minore o uguale di quello dell'elemento successivo.

Diagrammi di flusso di dati (DFD)

- Sono focalizzati su operazioni e flussi di informazione
- Incarnano il concetto di raffinamento della specifica (o astrazione mediante incapsulamento) effettuando la scomposizione di un'operazione in operazioni più semplici → incrementalità nella produzione della specifica

Elementi grafici

Agente esterno,
interazione di I/O

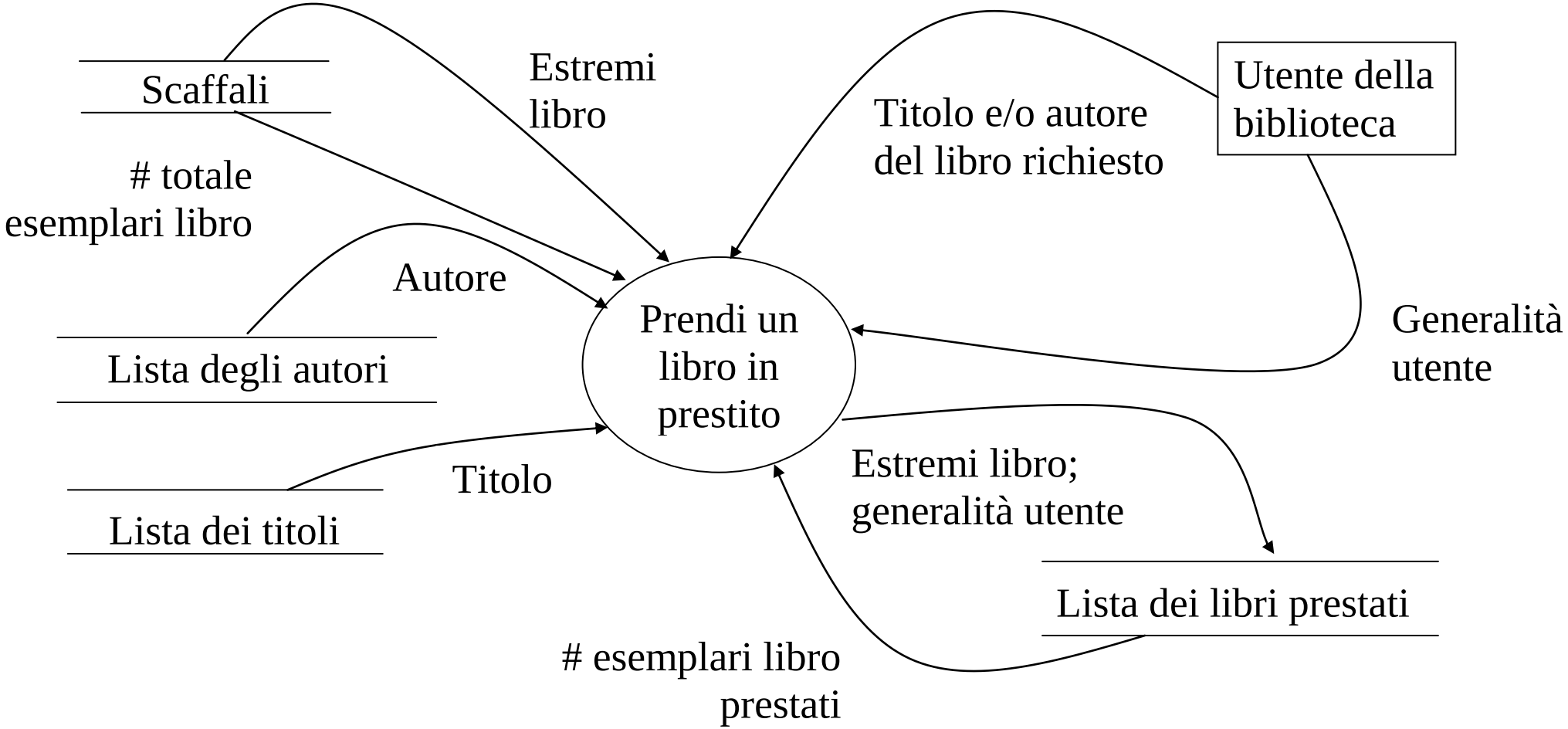


Funzione o processo

Flusso di dati

Deposito (permanente) di dati

DFD: l'esempio di una biblioteca

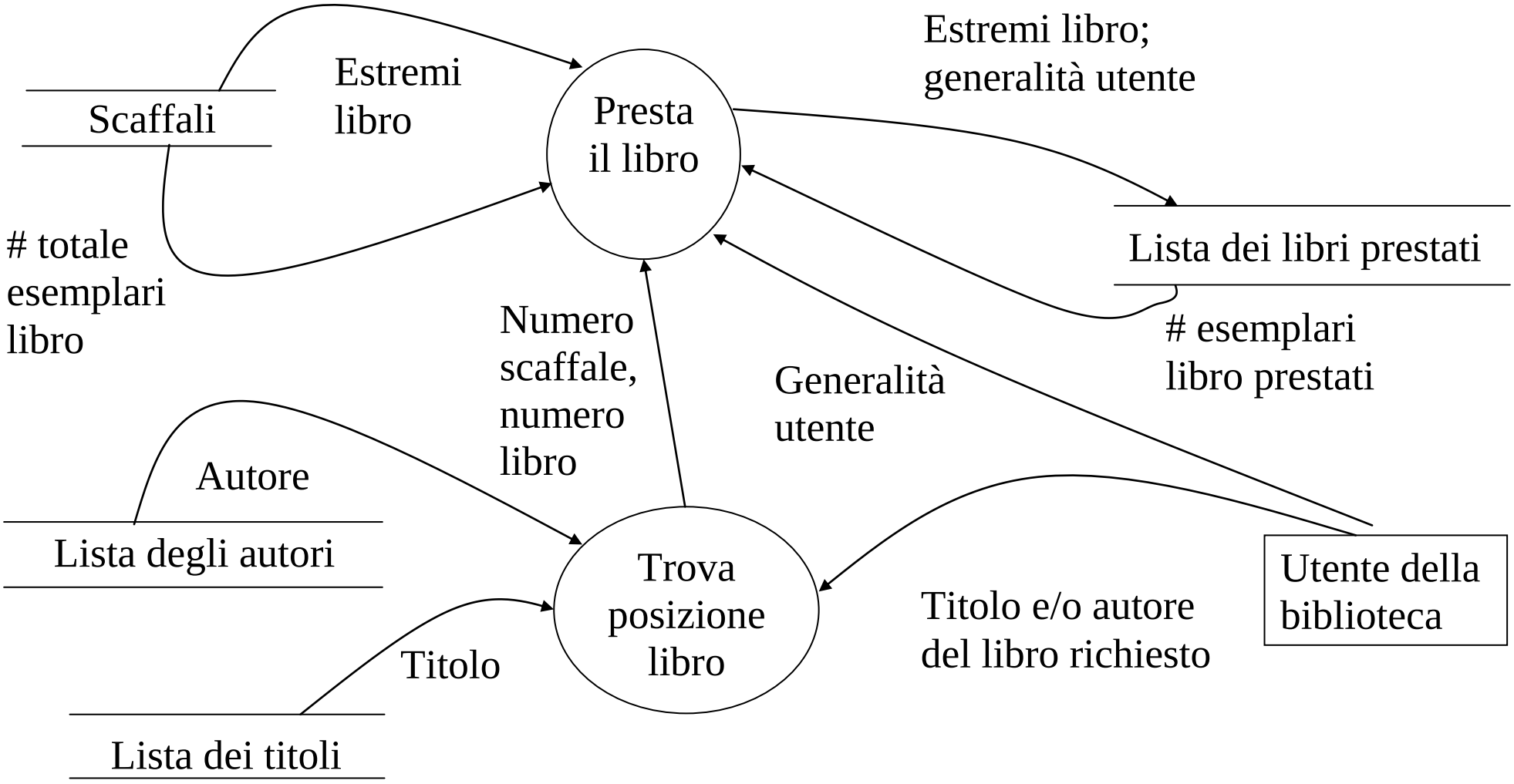


DFD: vincolo di continuità (o di bilanciamento) del flusso informativo

Nel diagramma corrispondente a una funzione che è stata esplosa, i flussi in entrata e uscita devono essere gli stessi flussi della funzione originale (mentre si possono introdurre nuovi agenti e depositi “locali”)

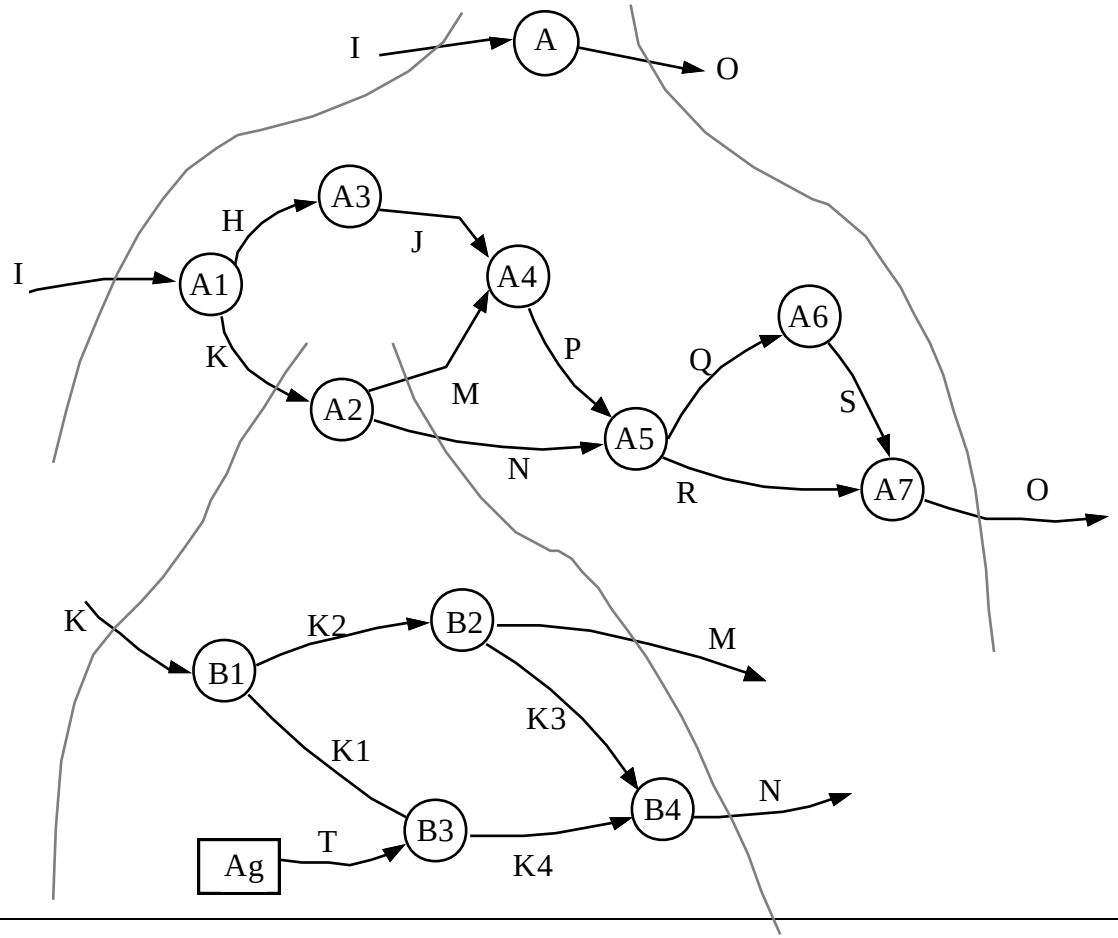
DFD: l'esempio di una biblioteca (cont.)

Raffinamento della funzione "Prendi un libro in prestito"



DFD: metodo di costruzione

1. Si inizia con un diagramma di contesto (contenente magari una sola bolla)
2. Si procede per raffinamenti successivi (preservando il bilanciamento del flusso informativo) finché non si raggiungono funzioni “elementari”



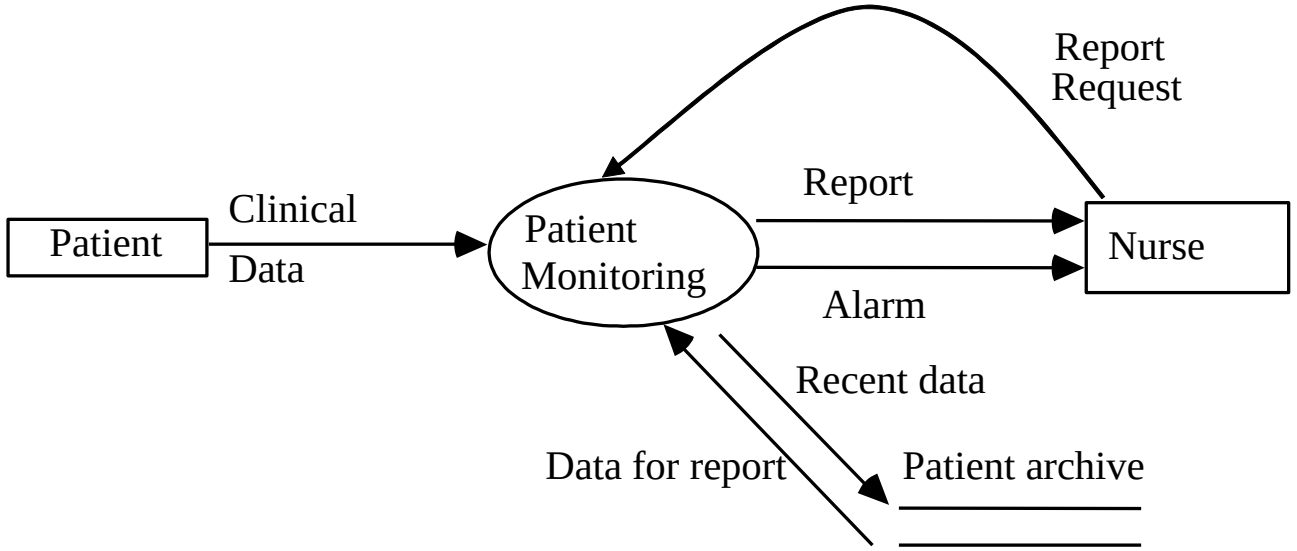
DFD: metodo di costruzione (cont.)

Linee guida per la costruzione di un DFD:

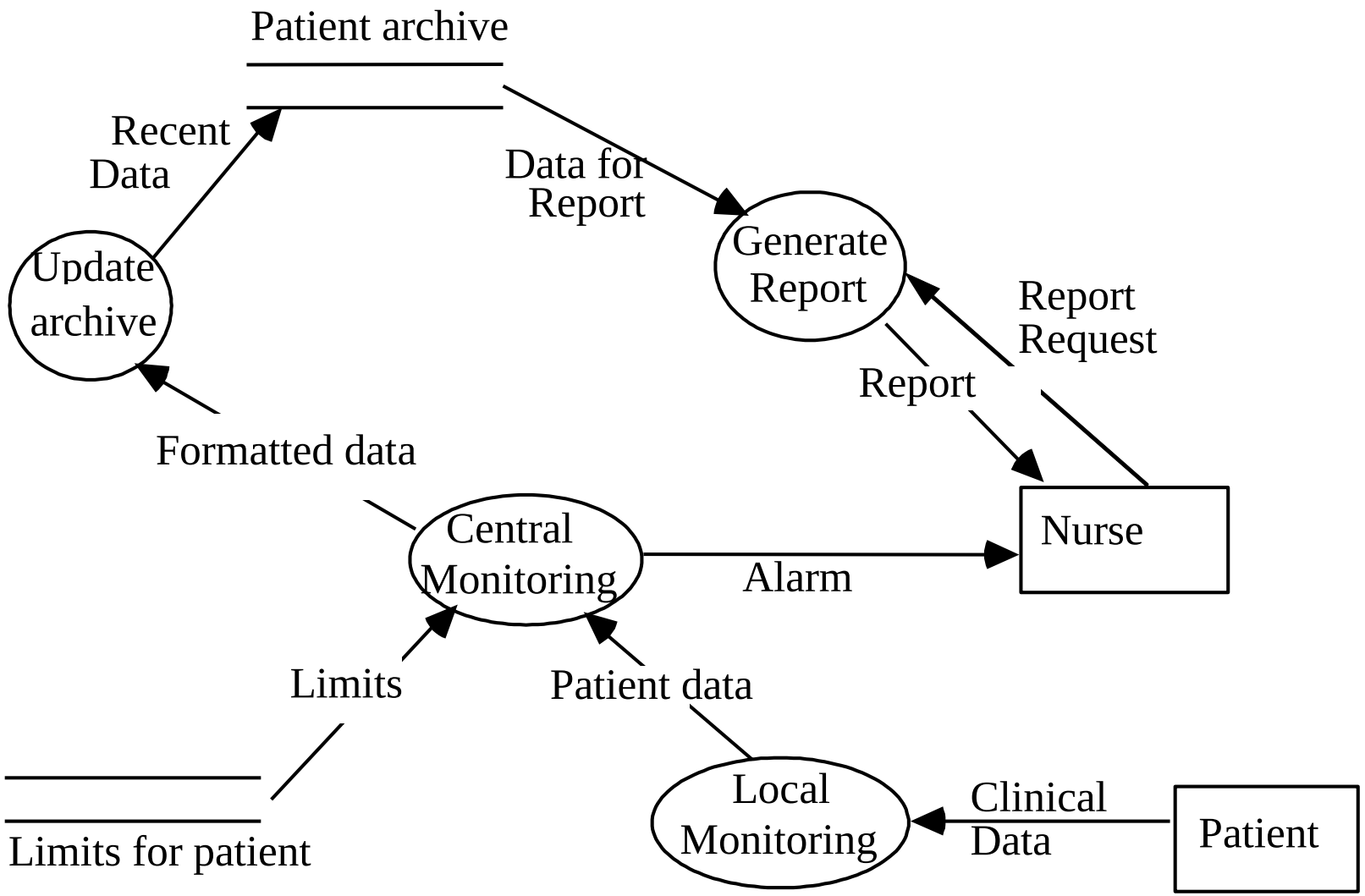
- ignorare inizializzazione, terminazione e casi di errore; concentrarsi sulle condizioni stabili
- ignorare flusso di controllo e sincronizzazione
- individuare innanzi tutto ingressi e uscite esterni al (sotto) sistema che si sta descrivendo
- usare nomi significativi
- percorrere i flussi informativi per valutare la correttezza e la consistenza del diagramma

DFD: l'esempio di un sistema di monitoraggio di pazienti

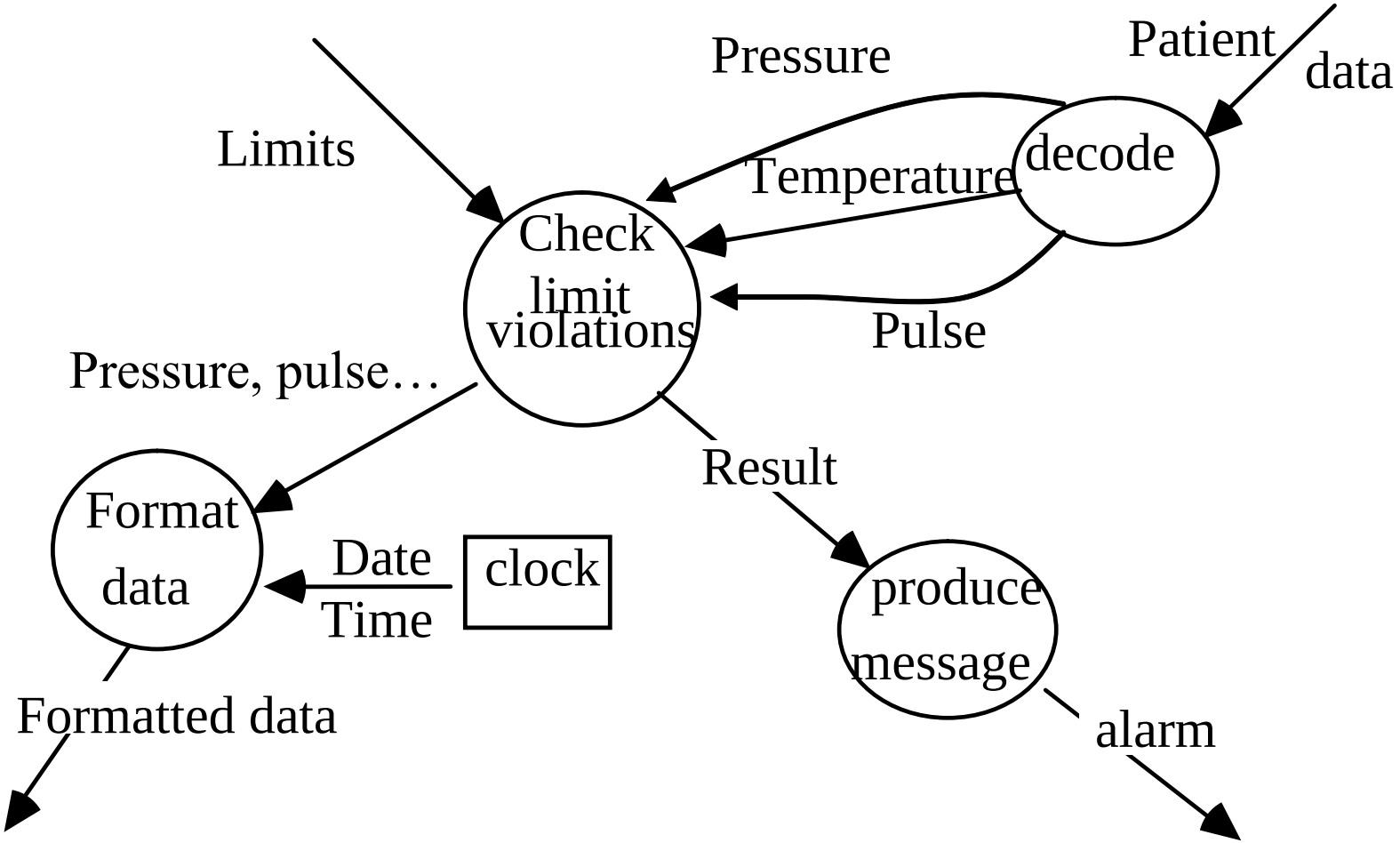
L'intento del sistema è monitorare i fattori vitali dei pazienti (pressione sanguigna, temperatura, …) leggendoli, secondo frequenze specificate, da dispositivi analogici e memorizzando le letture in una base di dati.
Se una lettura cade al di fuori dell'intervallo specificato per il paziente o un dispositivo è guasto, deve essere inviato un allarme all'infermiere/a.
Il sistema deve anche generare rapporti.



Un raffinamento di Patient Monitoring



Un raffinamento di Central Monitoring



DFD: pro e contro

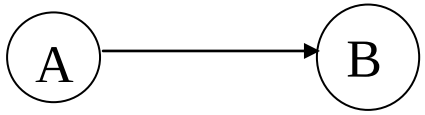
Pregio dei DFD: sono facili da leggere

Limiti dei DFD:

- non consentono di esprimere il controllo di flusso né le sincronizzazioni
- non consentono di esprimere requisiti non funzionali
- sono ambigui

Ci sono state proposte sia di formalizzazione sia di estensione (ad es. per sistemi in tempo reale)

DFD: ambiguità

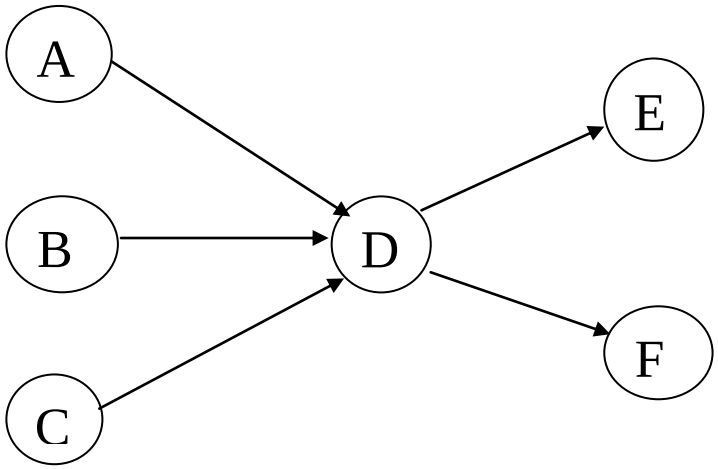
Es. 

Possibili interpretazioni:

(a) A produce un dato e aspetta finché B lo consuma

(b) B può leggere il dato più volte senza consumarlo

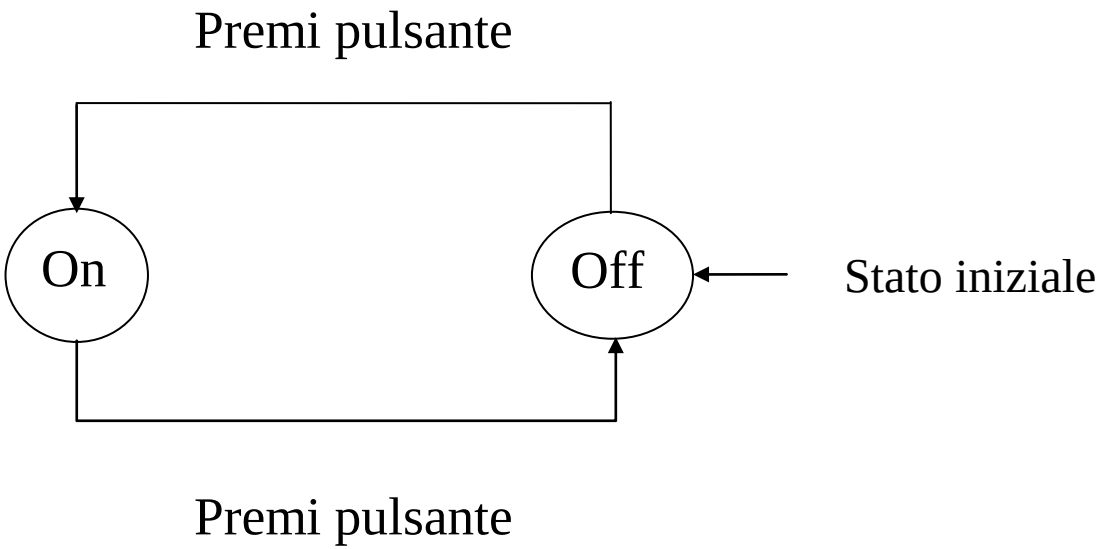
(c) A e B sono connessi mediante un meccanismo a “pipe”



- Le uscite di A, B, e C sono tutte sempre necessarie ?
- Le uscite inviate a E e F sono uguali, oppure sono prodotte contemporaneamente, oppure sono prodotte alternativamente ?

Macchine a stati finiti (FSM)

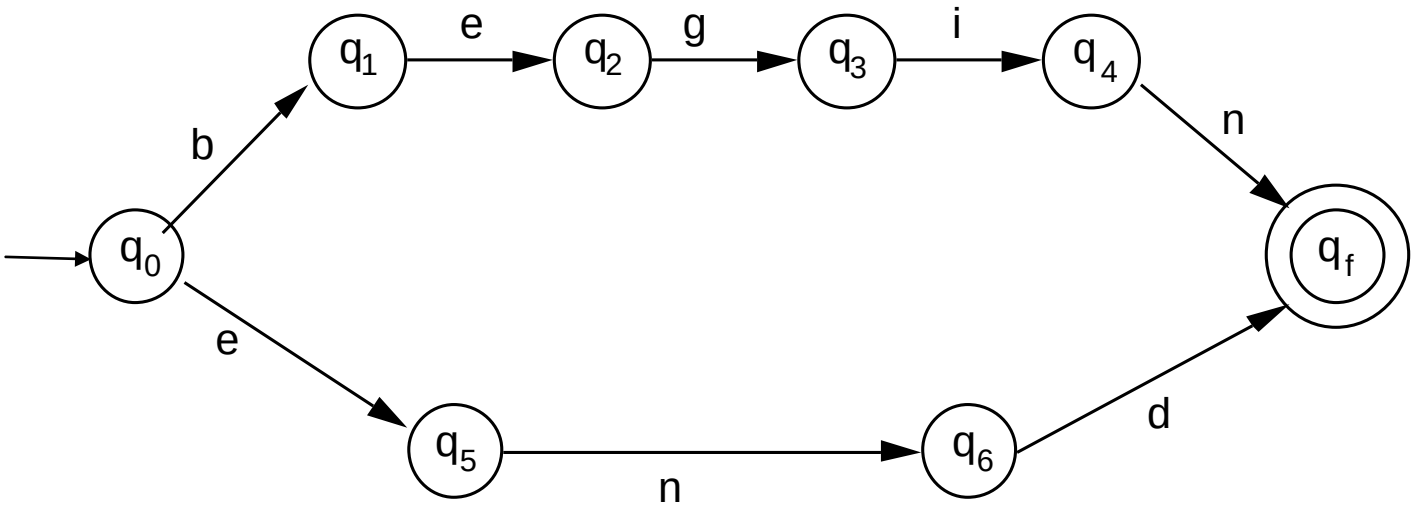
S: insieme finito di stati
I: insieme finito degli ingressi
δ: funzione di transizione
(anche parziale)



Classi di FSM

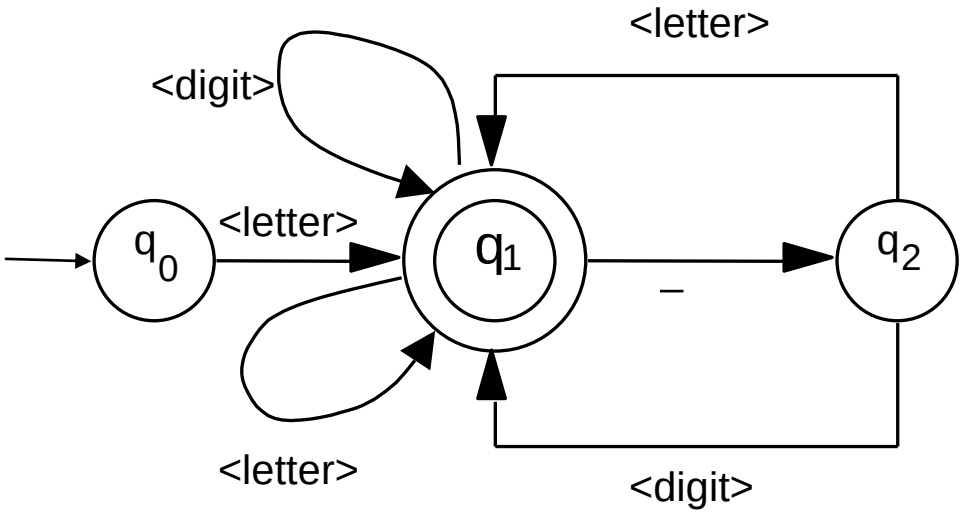
- Deterministiche/nondeterministiche
- Riconoscitori (dotati di stati finali)
- Traduttori (dotati di un insieme finito di uscite)

FSM come riconoscitori



Stato finale

FSM come riconoscitori (cont.)

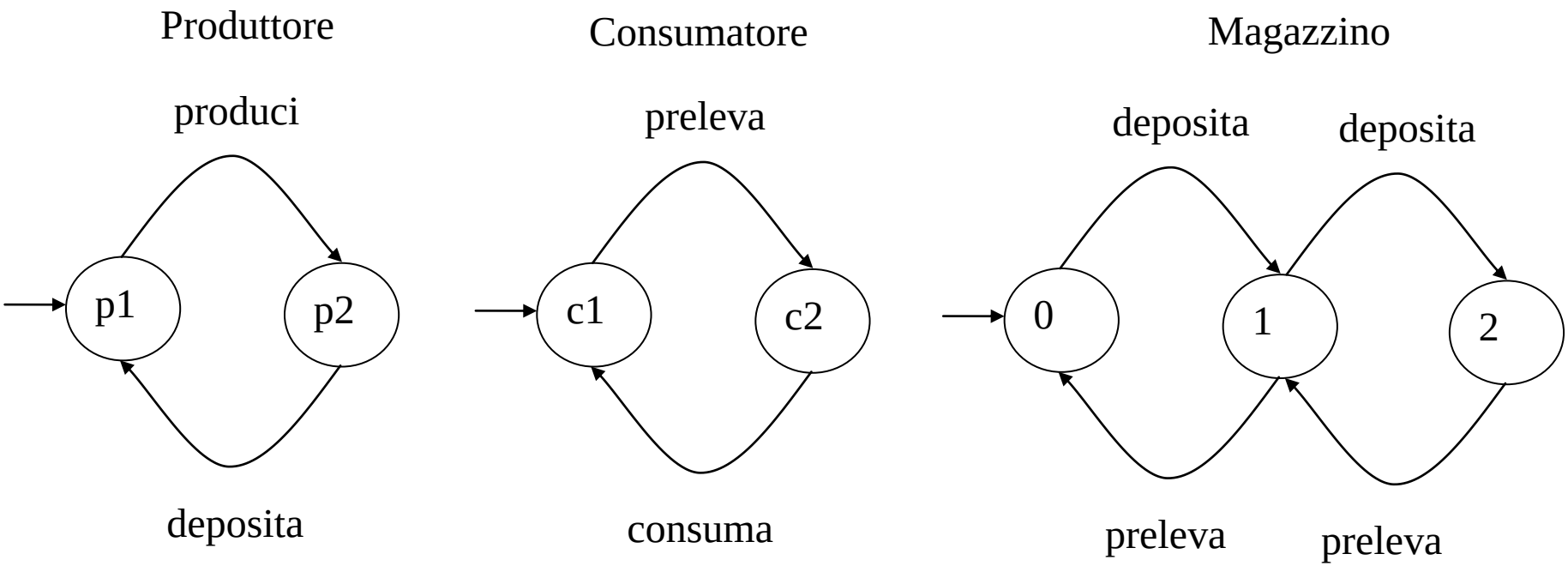


Legend:

<letter> → is an abbreviation for a set of arrows labeled a, b,..., z, A,..., Z,

<digit> → is an abbreviation for a set of arrows labeled 0, 1,..., 9, respectively

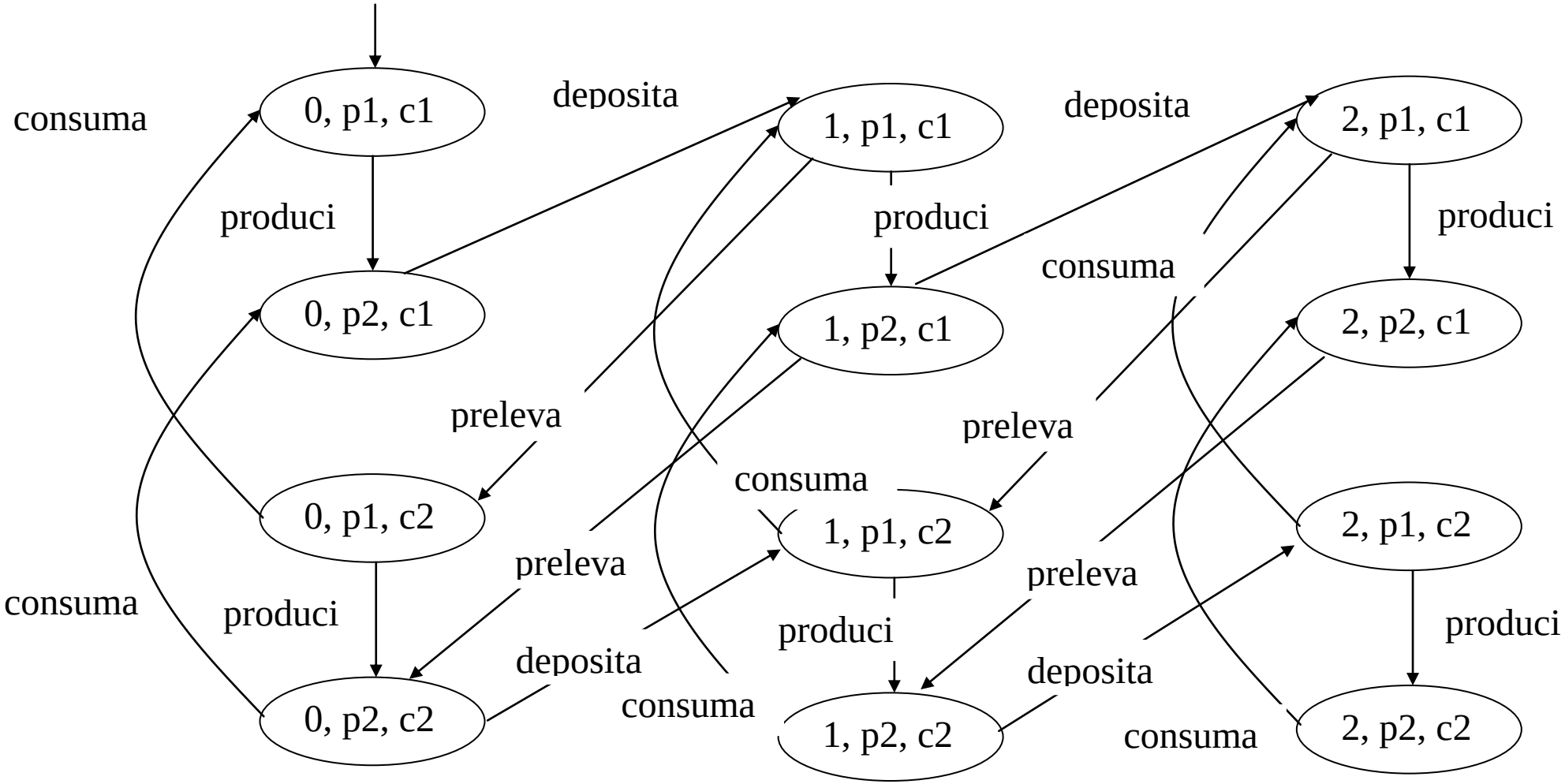
FSM: l'esempio dei processi produttore-consumatore



Limitazione

Esplosione degli stati quando si rappresentano sistemi concorrenti: date n FSM con $k_1, k_2, ..., k_n$ stati, la loro composizione è una FSM con $k_1 * k_2 * ... * k_n$ stati (mentre vorremmo che fossero $k_1 + k_2 + ... + k_n$)

FSM: l'esempio dei processi produttore-consumatore (cont.)

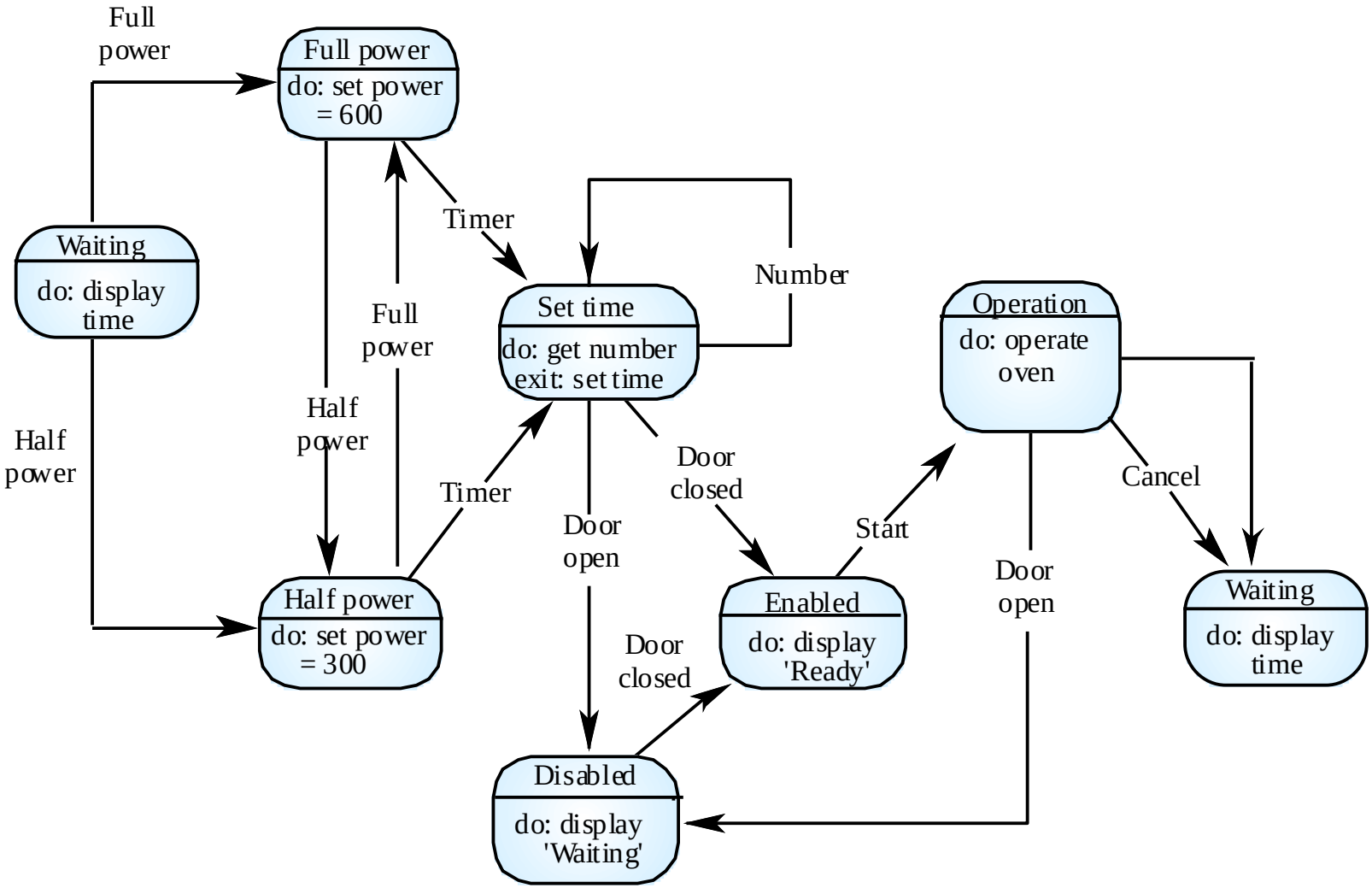


Macchine a stati finiti

Soluzione della limitazione delle FSM:

- Statechart (FSM cooperanti, usate in UML)
- Reti di Petri

Statechart


Full power
Full power
do: set power = 600
Waiting
do: display time
Timer
Number
Set time
do: get number
exit: set time
Full power
Half power
Half power
Operation
do: operate oven
Door closed
Timer
Door open
Start
Cancel
Door open
Half power
do: set power = 300
Enabled
do: display 'Ready'
Waiting
do: display time
Door closed
Disabled
do: display 'Waiting'

Reti di Petri (PN)

- I concetti di stato e transizione non sono più centralizzati ma distribuiti
- Descrizione naturale di sistemi asincroni
- L’evoluzione è non deterministica

Reti

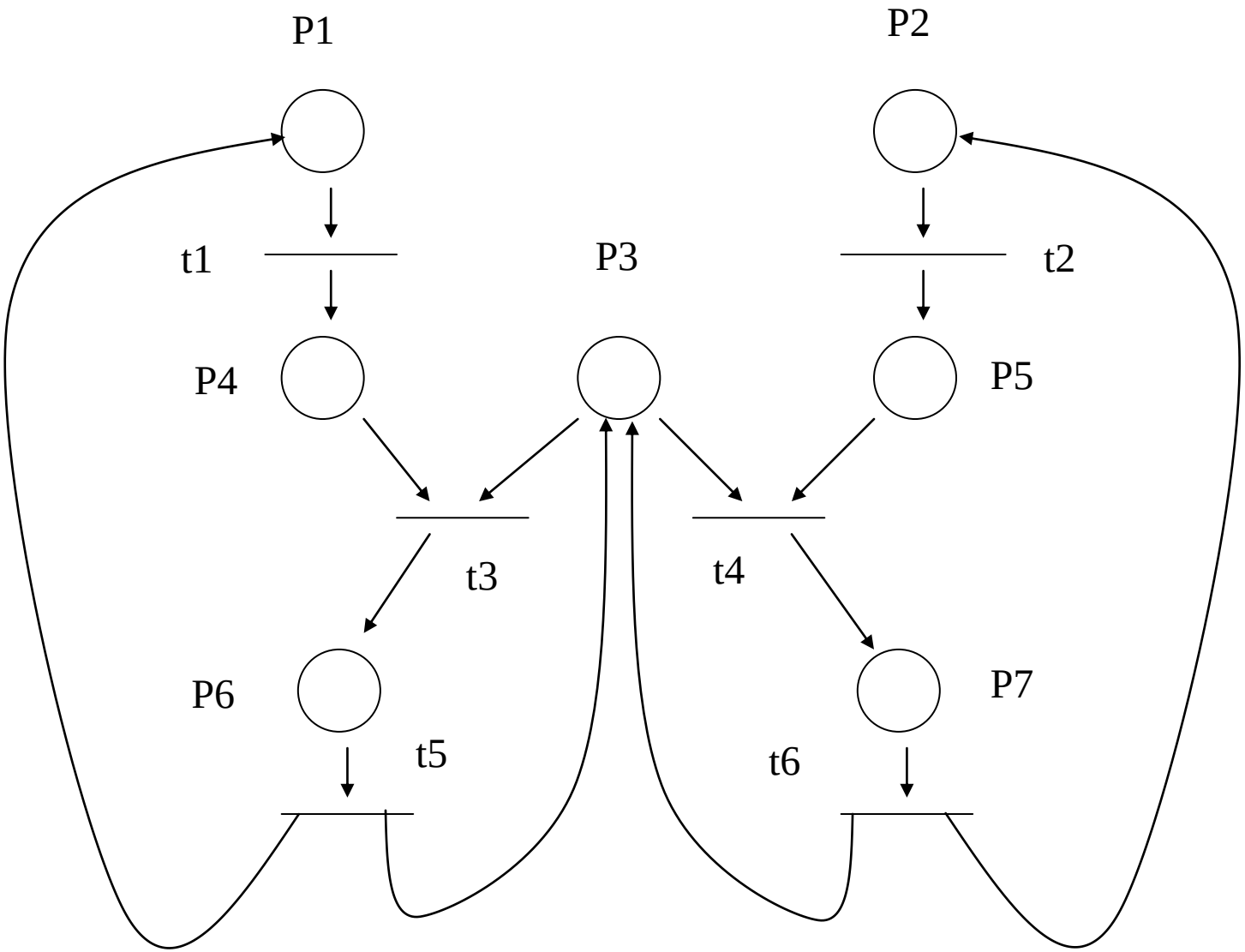
$N = (P, T, F)$

con
P: insieme finito di posti (places) 
T: insieme finito di transizioni 
F: relazione di flusso 

Vincoli:
(1) $P \cap T = \varnothing$
(2) $P \cup T \neq \varnothing$
(3) $F \subseteq (P \times T) \cup (T \times P)$

$\text{Pre}(y \in P \cup T) = \{ x \in P \cup T \mid \langle x, y \rangle \in F \}$
$\text{Post}(x \in P \cup T) = \{ y \in P \cup T \mid \langle x, y \rangle \in F \}$

Reti

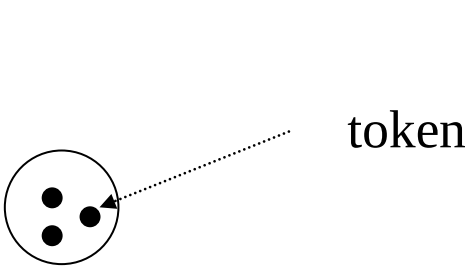


PN

Modello di base: PN = (P, T, F, W, M_0)

W: peso
M_0 : marcatura iniziale

Vincoli:
(4) W: F $\rightarrow$ N- {0}, valore di default di W è 1
(5) M_0: P $\rightarrow$ N (un funzione M: P $\rightarrow$ N è chiamata *marcatura*)

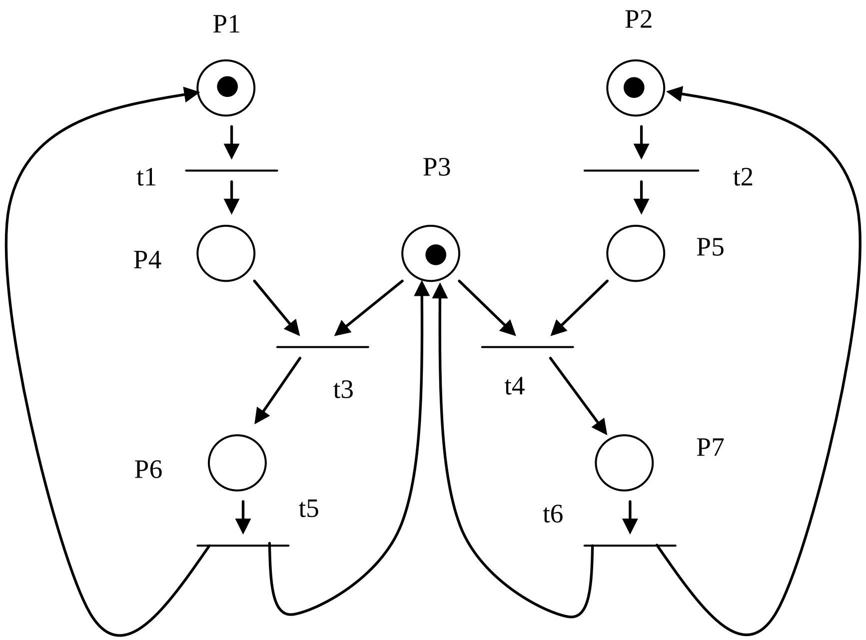


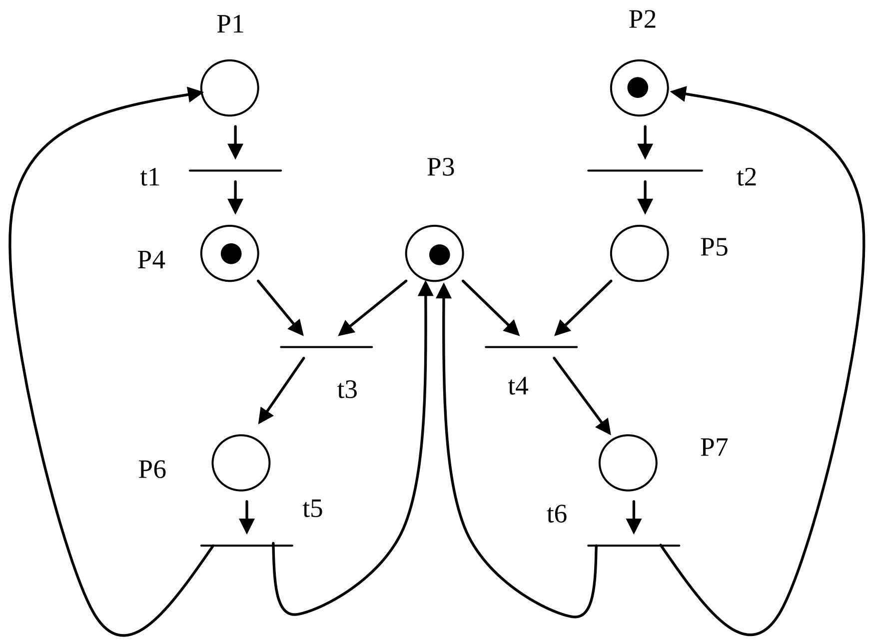
- Una marcatura è una rappresentazione di uno "stato" della rete
- La *semantica* definisce l'insieme delle possibili marcature prossime M', data la marcatura corrente M
- *Evoluzione dinamica*: $M_0 \Rightarrow M_1 \Rightarrow M_2 \Rightarrow$ ...

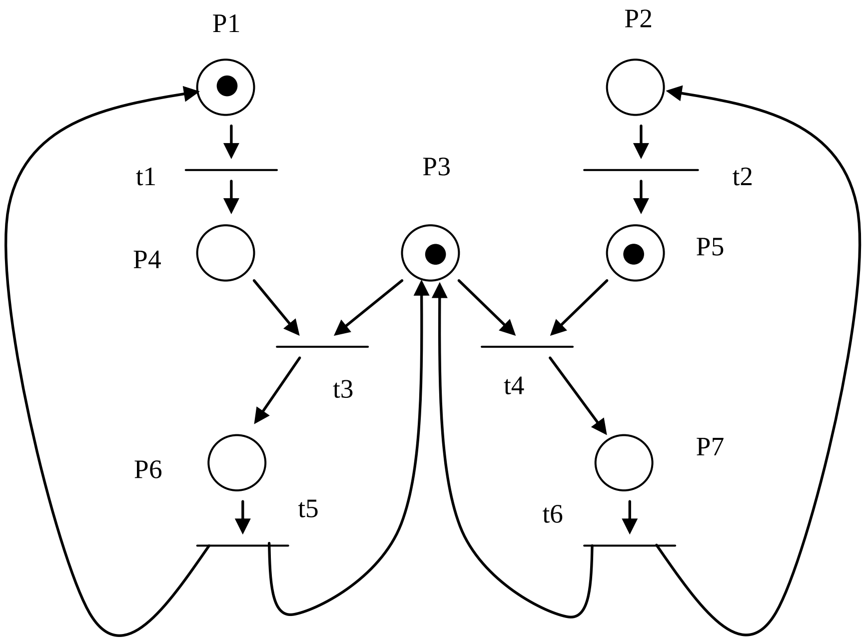
PN: semantica

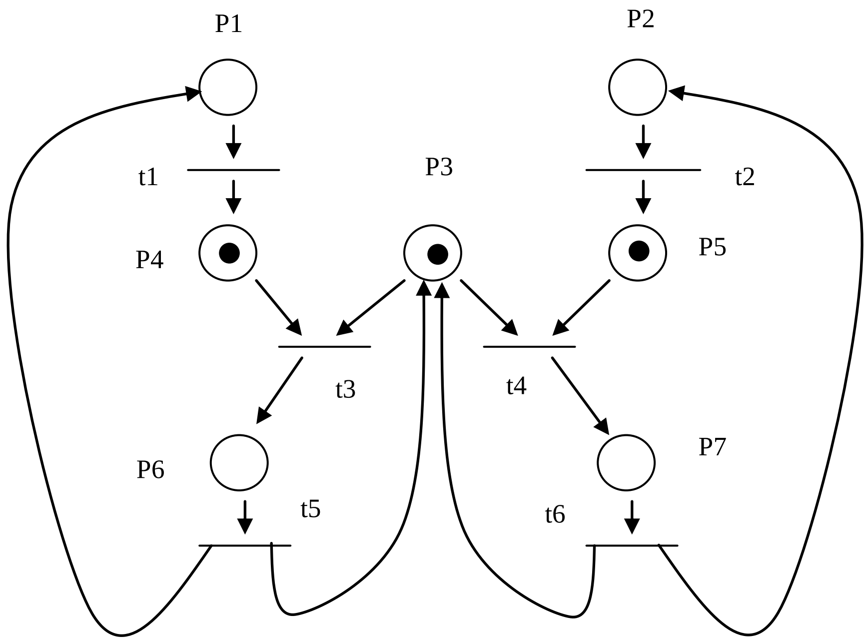
Descrizione informale (caso W=1)

- una transizione t è *abilitata* se c'è (almeno) un token in ogni $p \in Pre(t)$
- una transizione t abilitata può scattare, dando luogo a una nuova marcatura dove:
  - un token è cancellato da ogni $p \in Pre(t)$
  - un token è aggiunto a ogni $p \in Post(t)$
- se esistono più transizioni abilitate, la scelta della transizione da eseguire è nondeterministica


P1
P2
t1
P3
t2
P4
P5
t3
t4
P6
P7
t5
t6


P1
P2
t1
P3
t2
P4
P5
t3
t4
P6
P7
t5
t6


P1
P2
t1
P3
t2
P4
P5
t3
t4
P6
P7
t5
t6


P1
P2
t1
P3
t2
P4
P5
t3
t4
P6
P7
t5
t6

PN: semantica (cont.)

Descrizione formale (caso generale)

- Abilitazione di una transizione t data in corrispondenza della marcatura M (si scrive M[t>):

$$\forall p \in Pre(t) \bullet M(p) \geq W(\langle p, t \rangle)$$

- Scatto di una transizione t (M[t > M'):

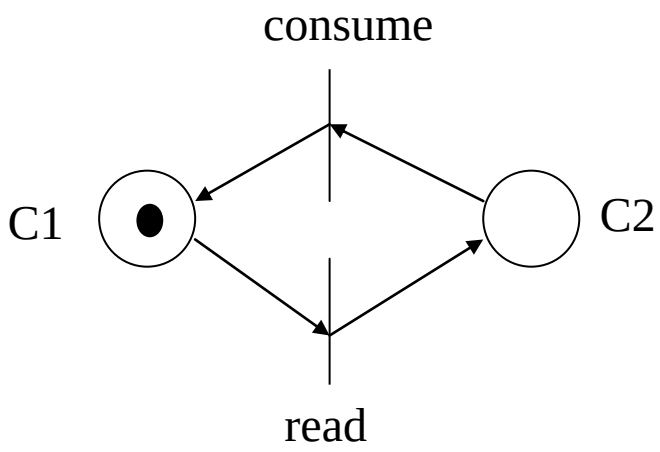
$$\forall p \in Pre(t) - Post(t) \bullet M'(p) = M(p) - W(\langle p, t \rangle)$$

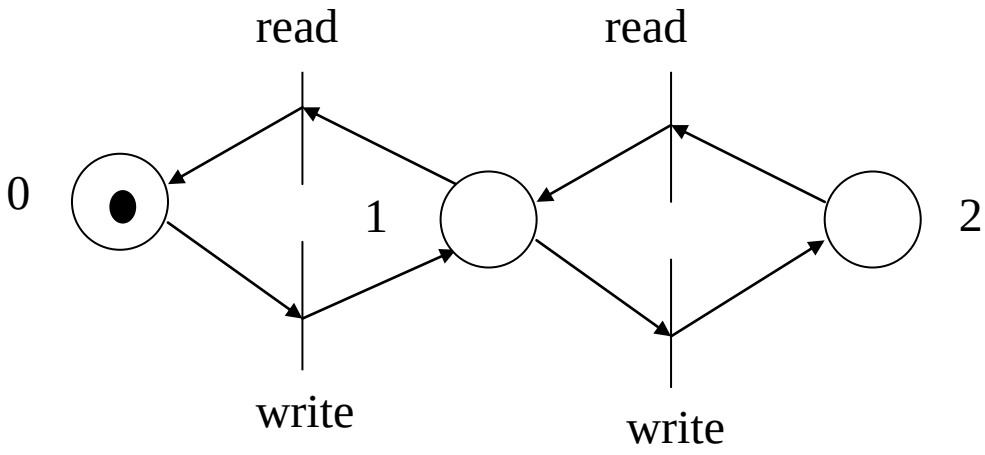
$$\forall p \in Post(t) - Pre(t) \bullet M'(p) = M(p) + W(\langle t, p \rangle)$$

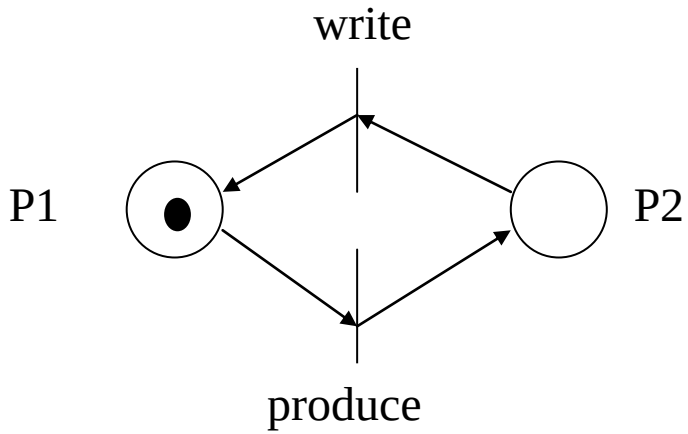
$$\forall p \in Pre(t) \cap Post(t) \bullet M'(p) = M(p) - W(\langle p, t \rangle) + W(\langle t, p \rangle)$$

$$\forall p \in P - (Pre(t) \cup Post(t)) \bullet M'(p) = M(p)$$

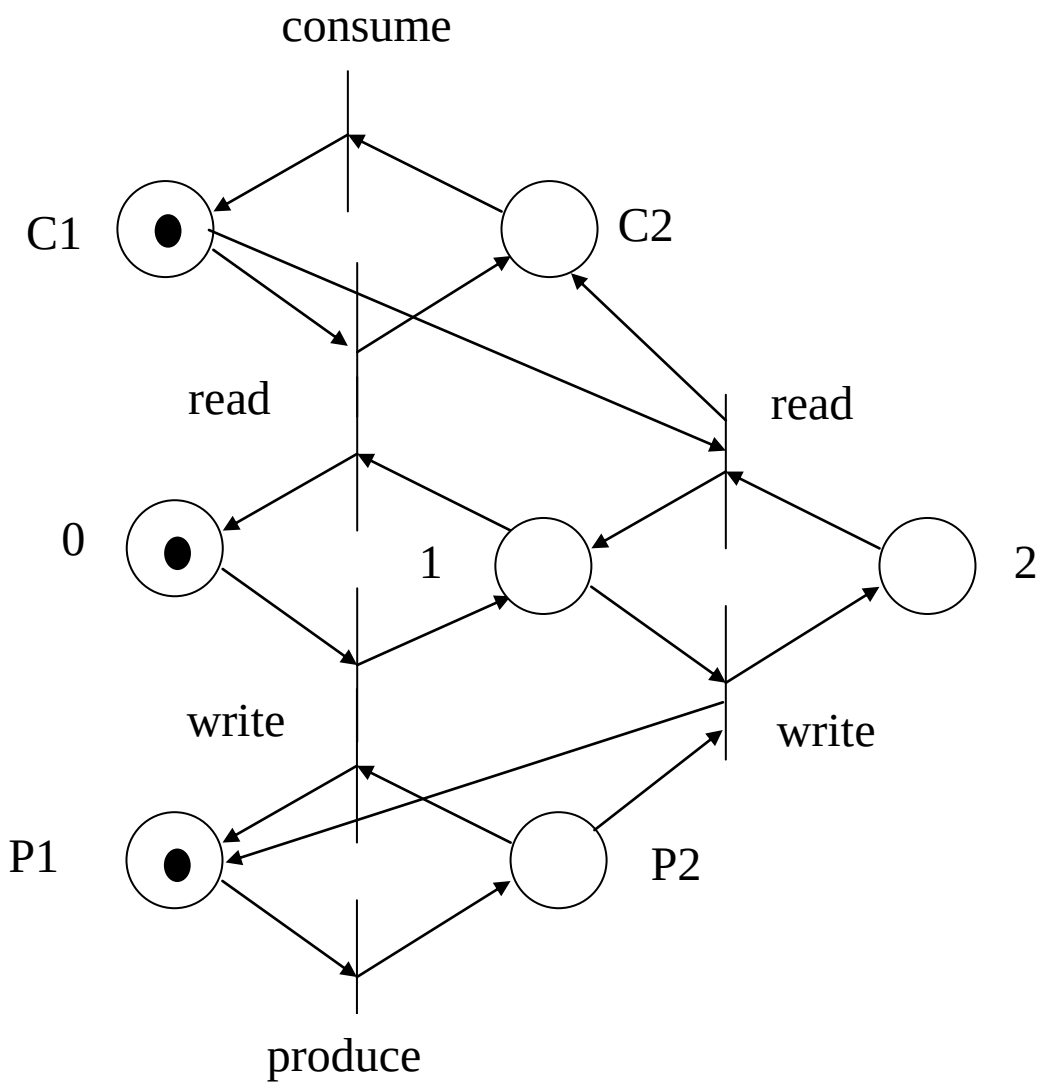
PN: l'esempio dei processi produttore-consumatore





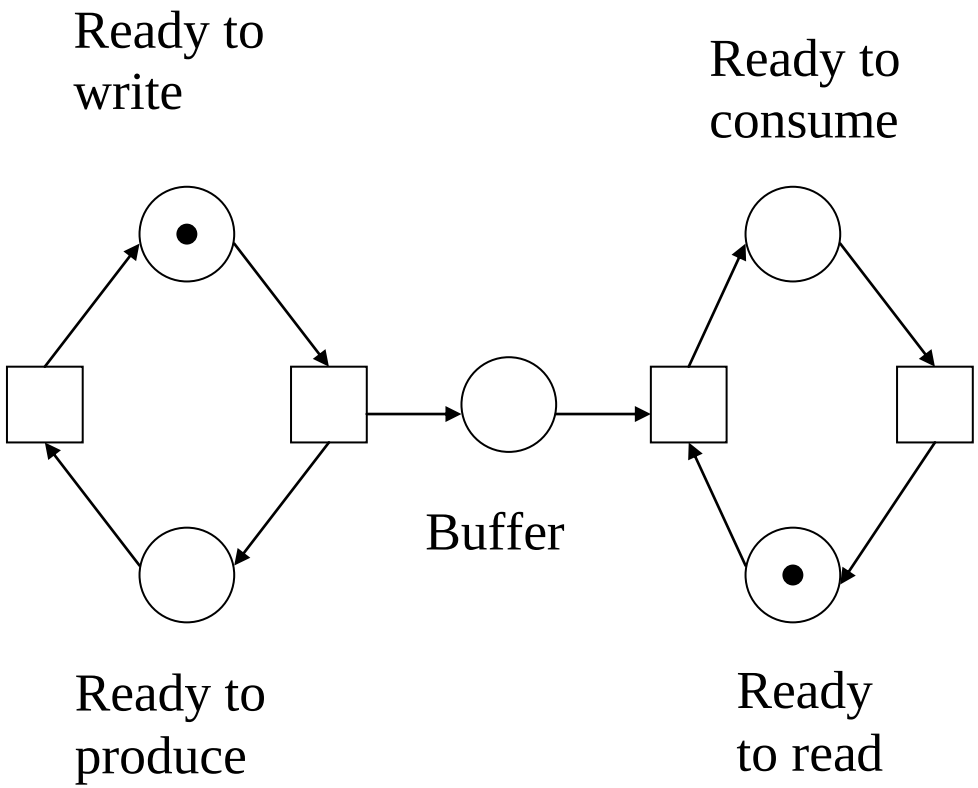


PN: l'esempio dei processi produttore-consumatore (cont.)

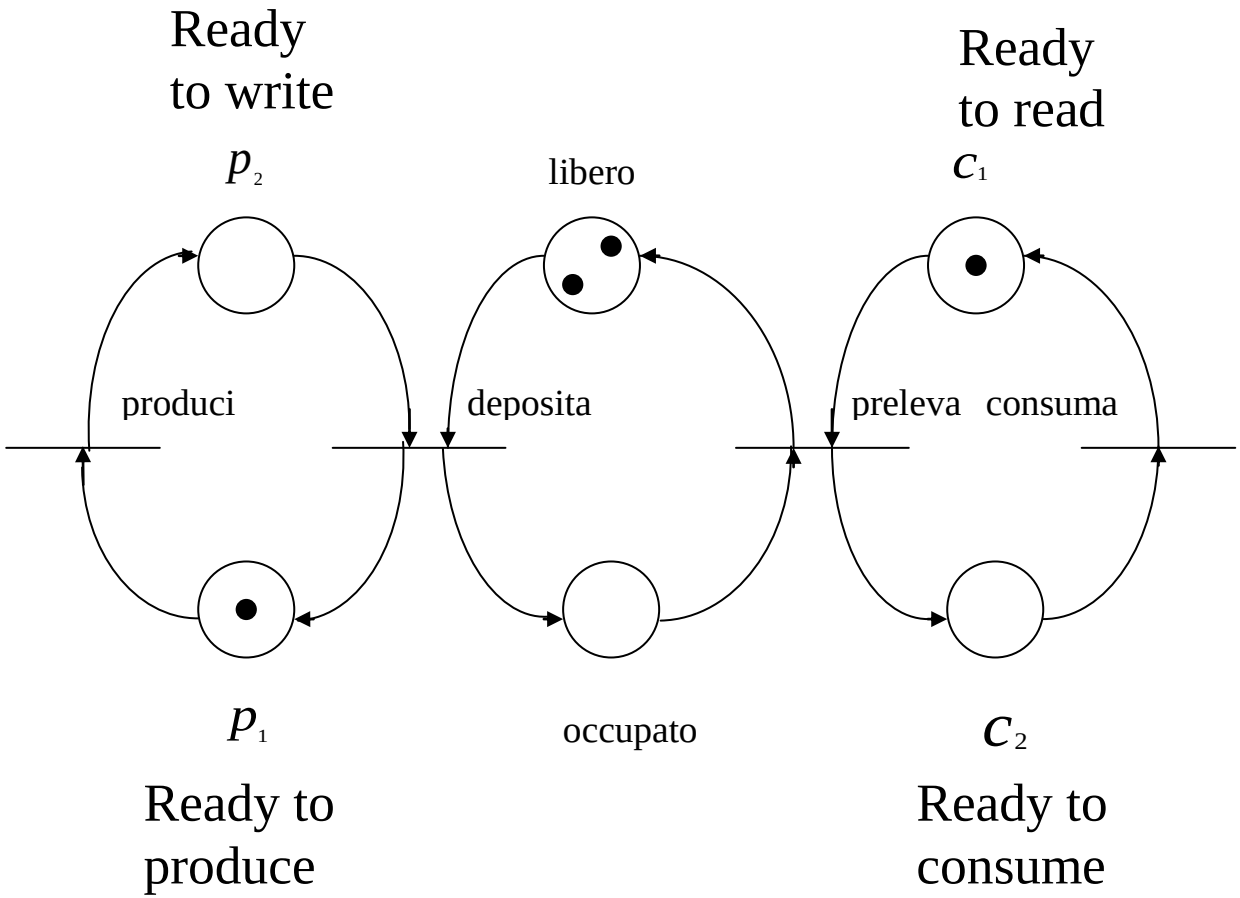


PN: processi produttore-consumatore con buffer illimitato

Una rete nei cui posti si possono verificare accumuli illimitati di token si dice *illimitata*



PN: processi produttore-consumatore con buffer finito

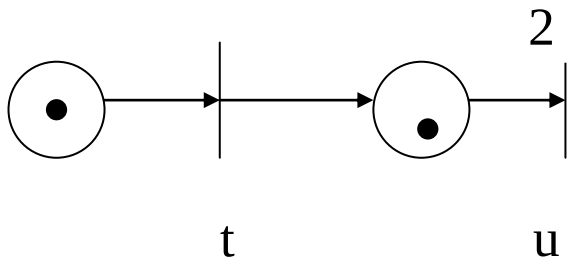


Sequenze e conflitti

Date due transizioni t e u, si definiscono

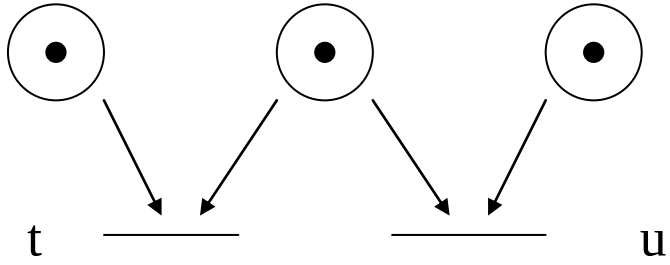
♦ Sequenza:

$M[t\rangle \wedge \neg M[u\rangle \wedge M[tu\rangle$



♦ Conflitto:

$M[t\rangle \wedge M[u\rangle \wedge \exists p \in Pre(t) \cap Pre(u) \bullet M(p) < W(\langle p, t \rangle) + W(\langle p, u \rangle)$

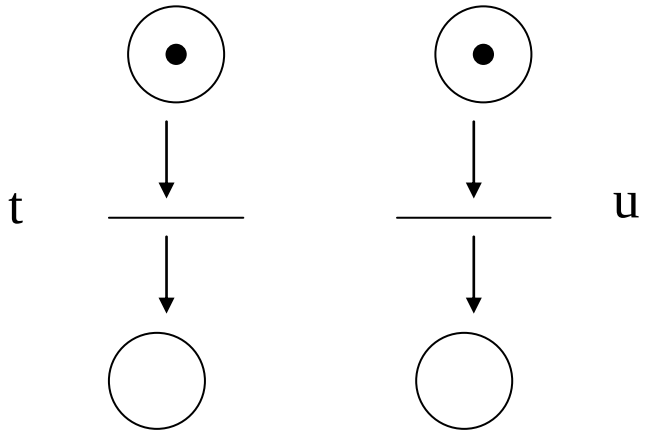


In presenza di conflitti, un processo può anche non accedere mai alle risorse necessarie alla sua evoluzione (*unfair scheduling*)

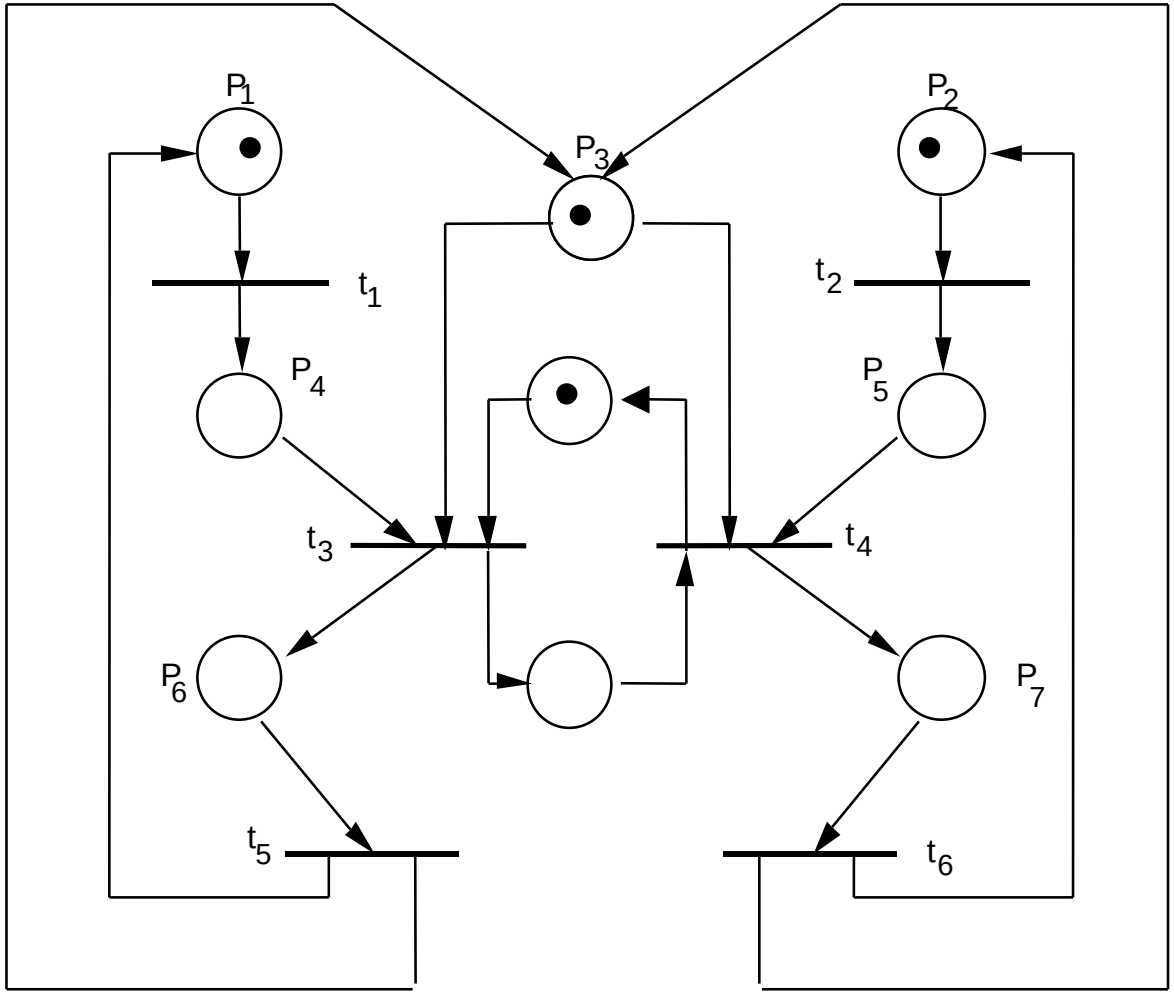
Concorrenza

♦ Concorrenza:

$M[t\rangle \wedge M[u\rangle \wedge \forall\ p \in Pre(t) \cap Pre(u) \bullet M(p) \geq W(\langle p, t \rangle) + W(\langle p, u \rangle)$

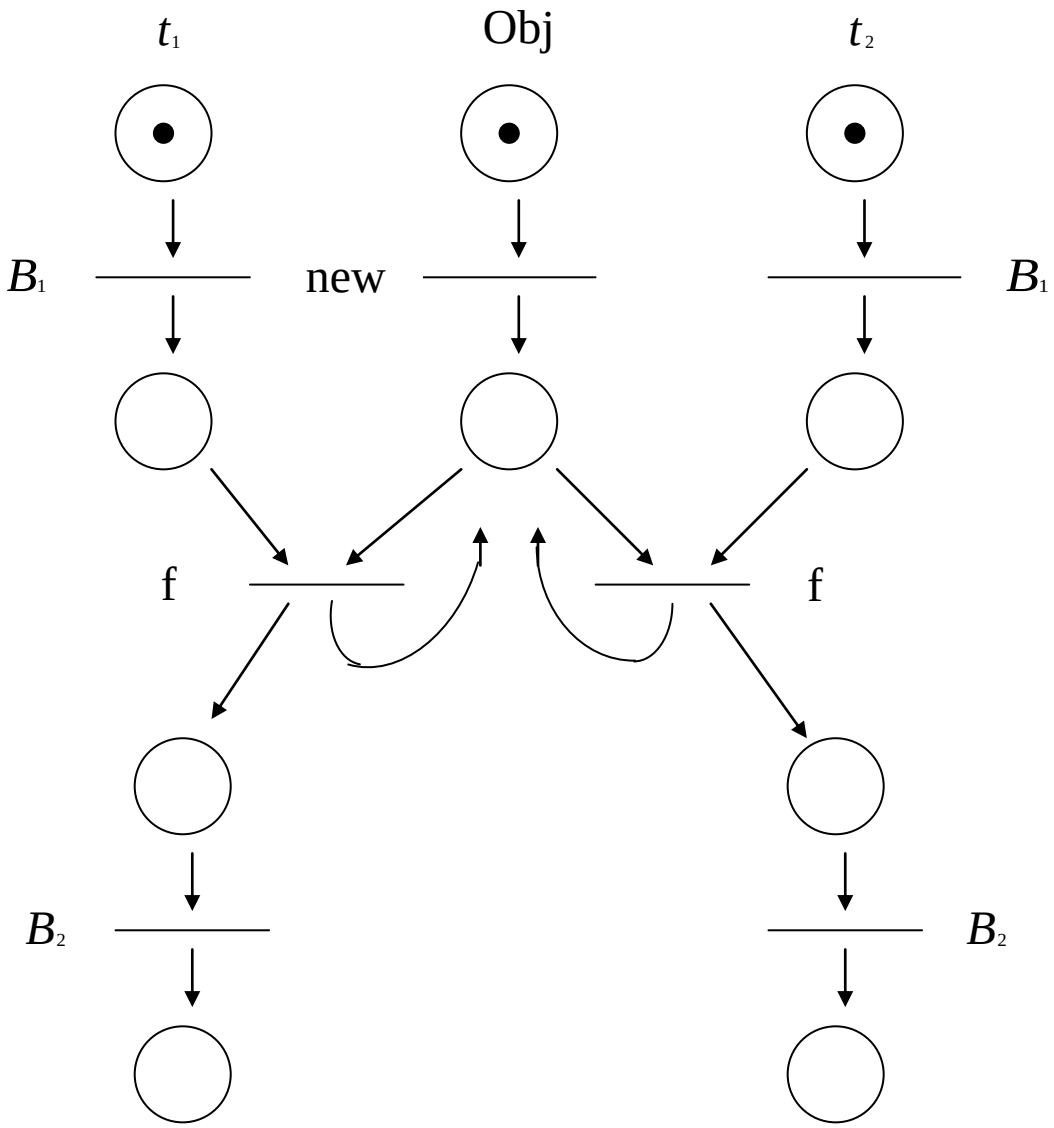


Fair scheduling

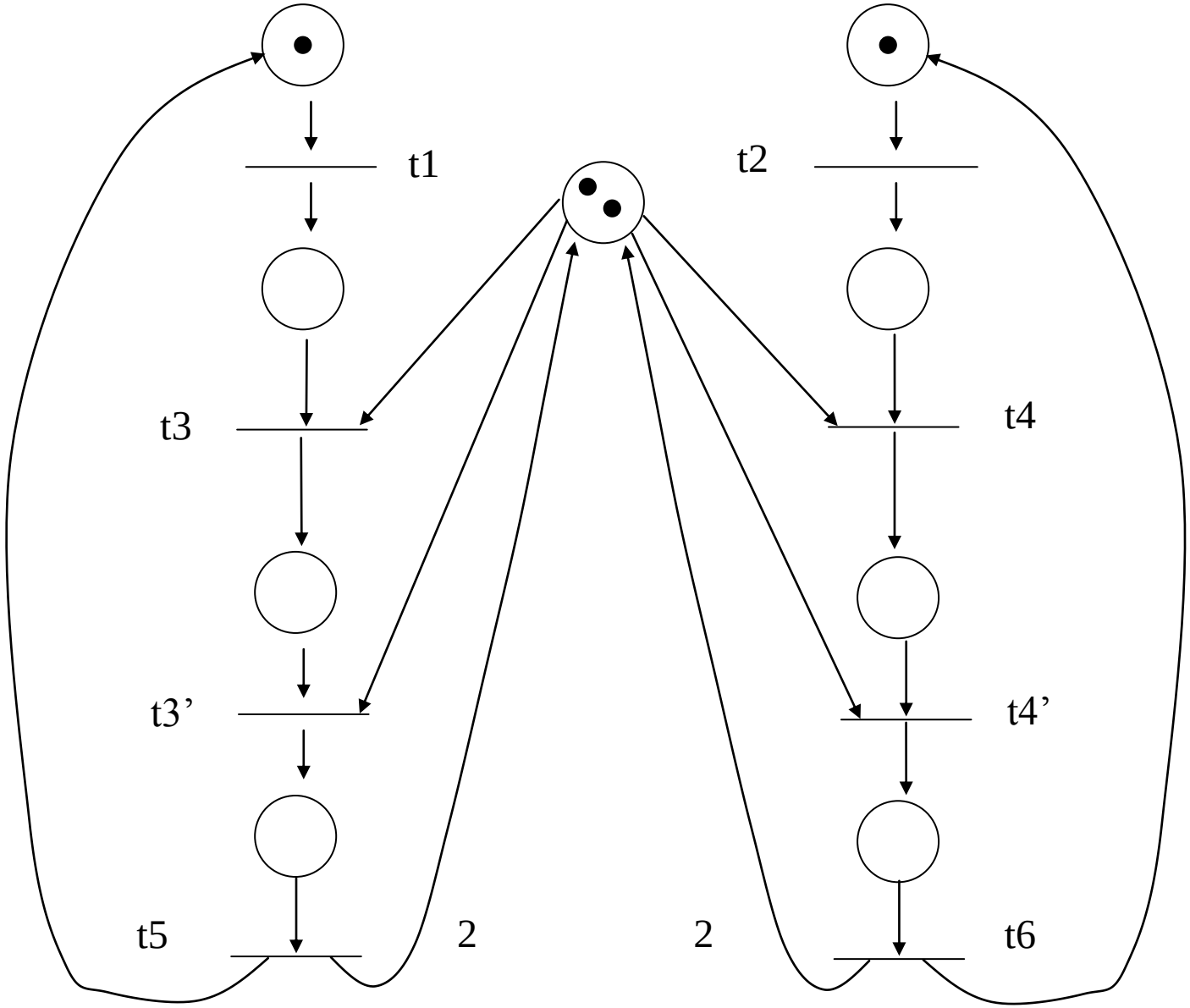


Modellizzazione di metodi sincronizzati

- ad es. in Java

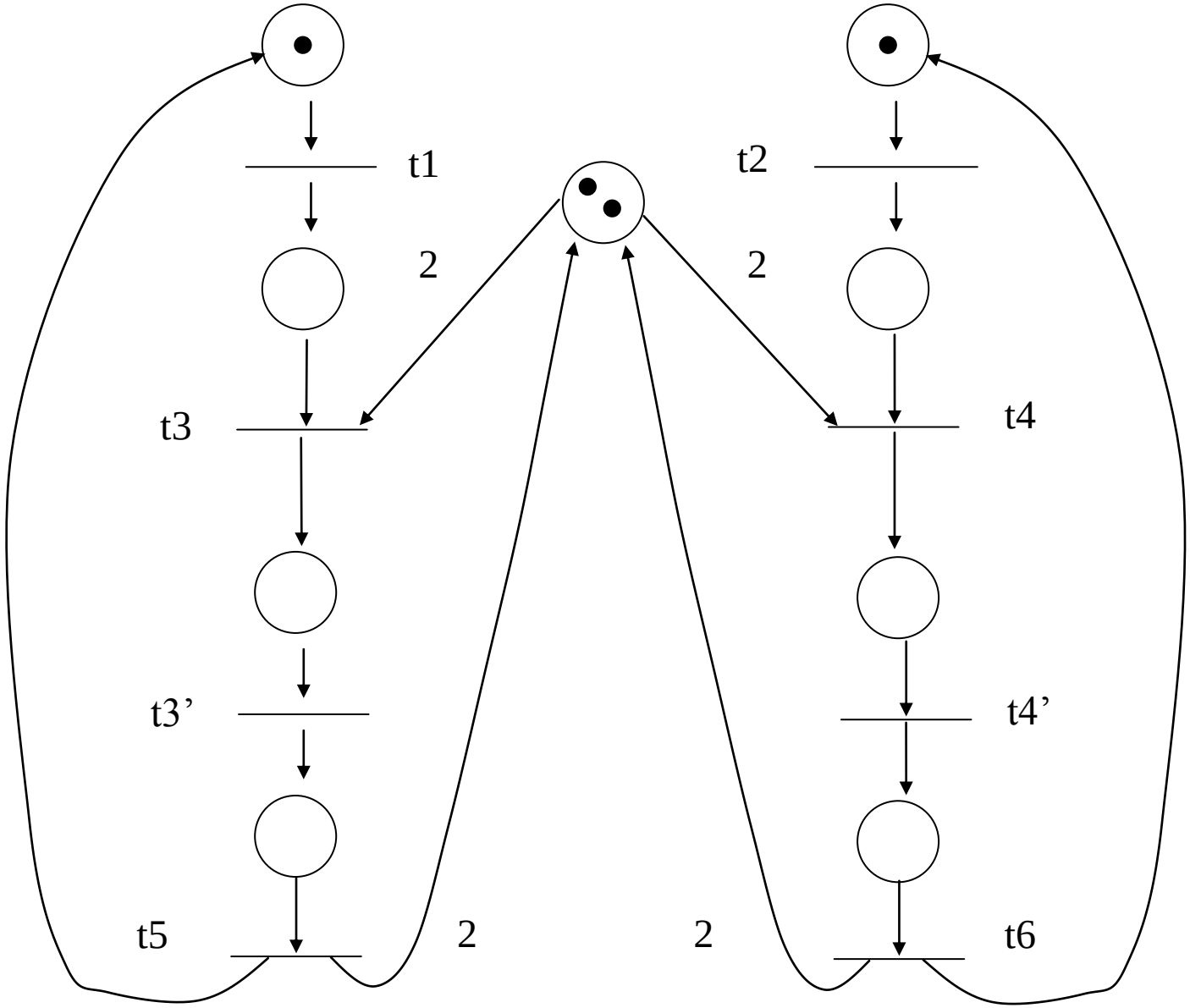


Deadlock



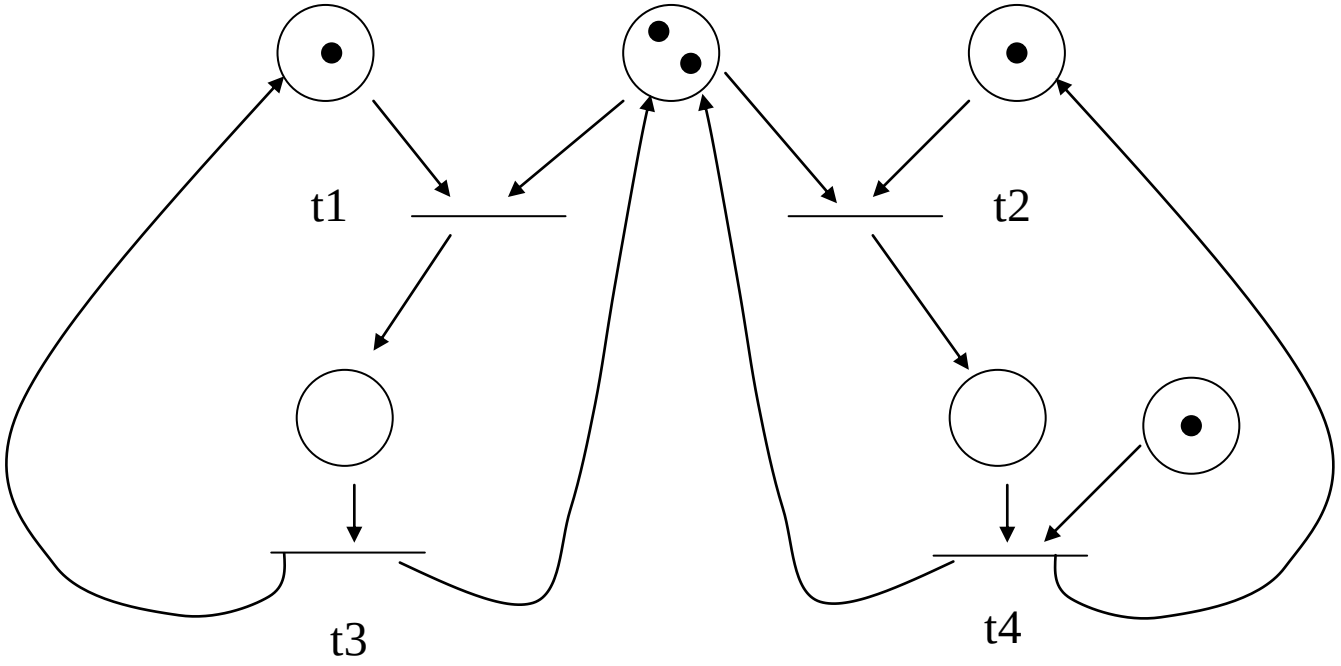
Nessun deadlock

Una rete priva di deadlock si dice *viva*



Starvation

Rete viva



Analisi di raggiungibilità

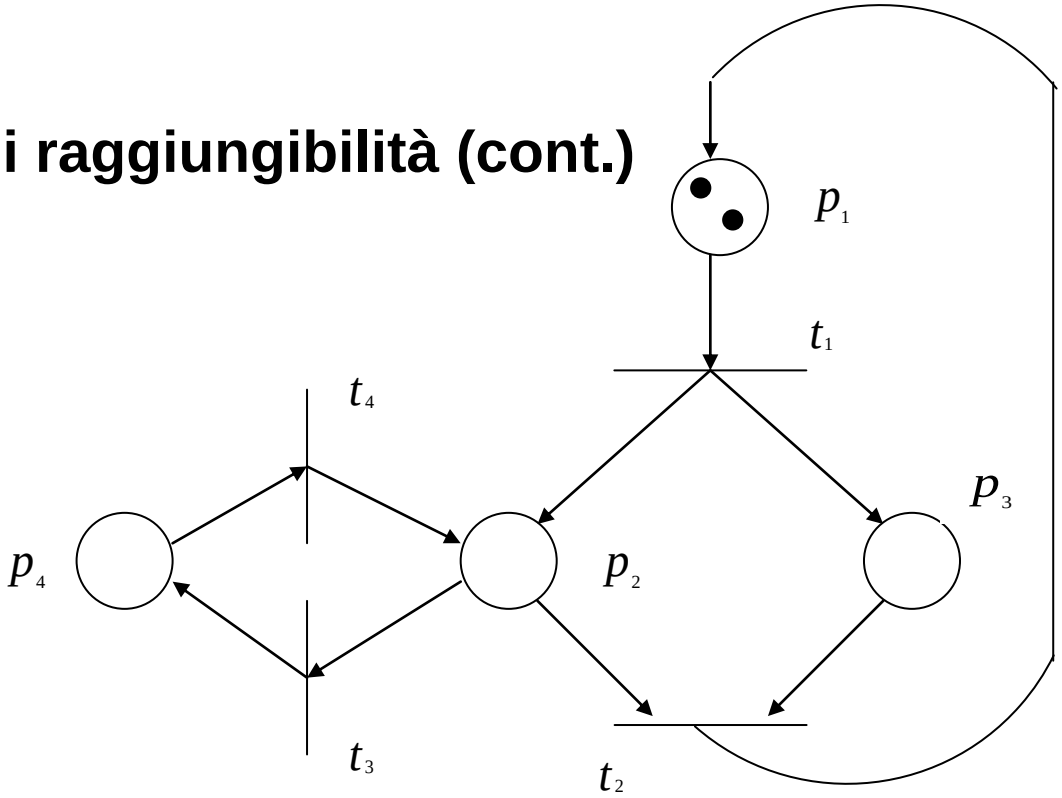
È tesa alla verifica delle proprietà

Consiste nel trovare le marcature raggiungibili a partire da una marcatura iniziale: problema decidibile ma di complessità esponenziale

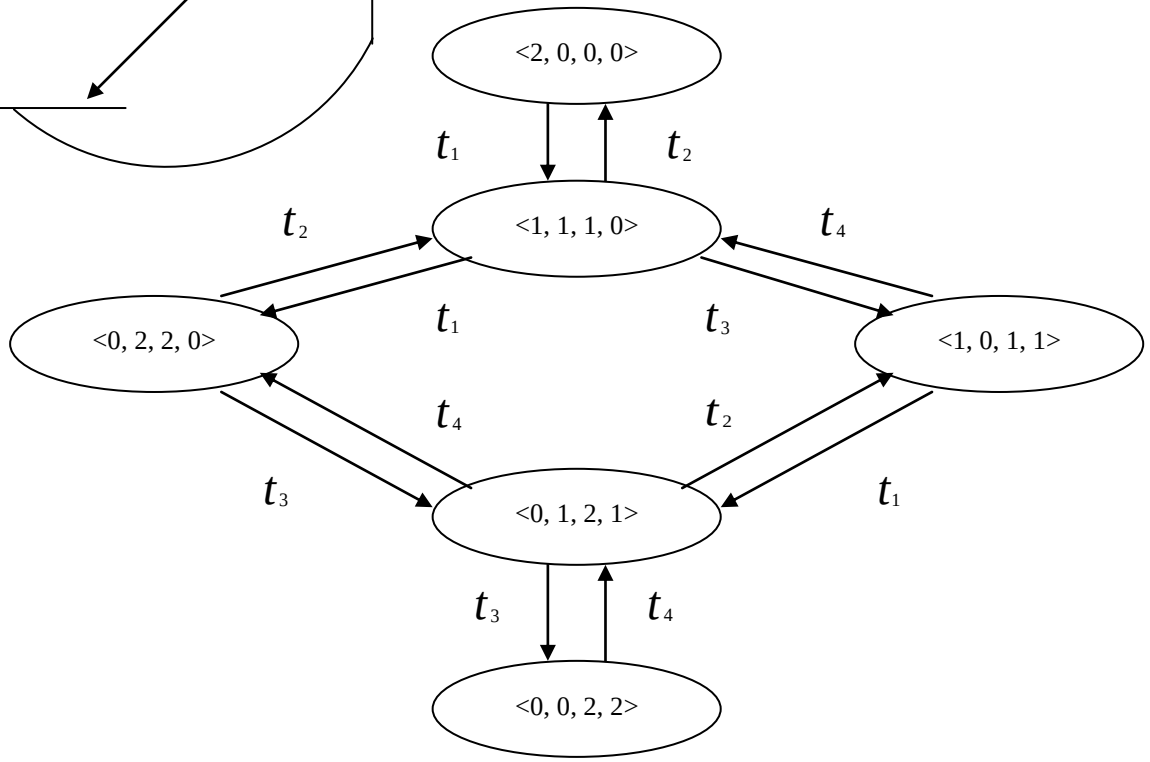
Raggiungibilità:
M' è raggiungibile in un passo se $\exists\ t \bullet M[t > M'$.
M' è raggiungibile se $\exists\ t_1 \ldots t_k \bullet M[t_1 \ldots t_k > M'$.

Rete illimitata = rete dotata di infinite marcature raggiungibili

Analisi di raggiungibilità (cont.)

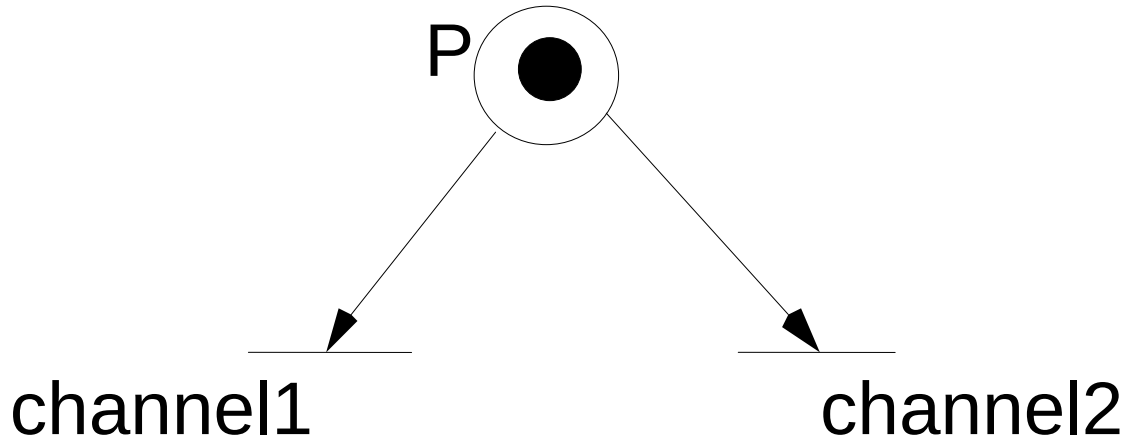


Grafo di raggiungibilità: nessun deadlock



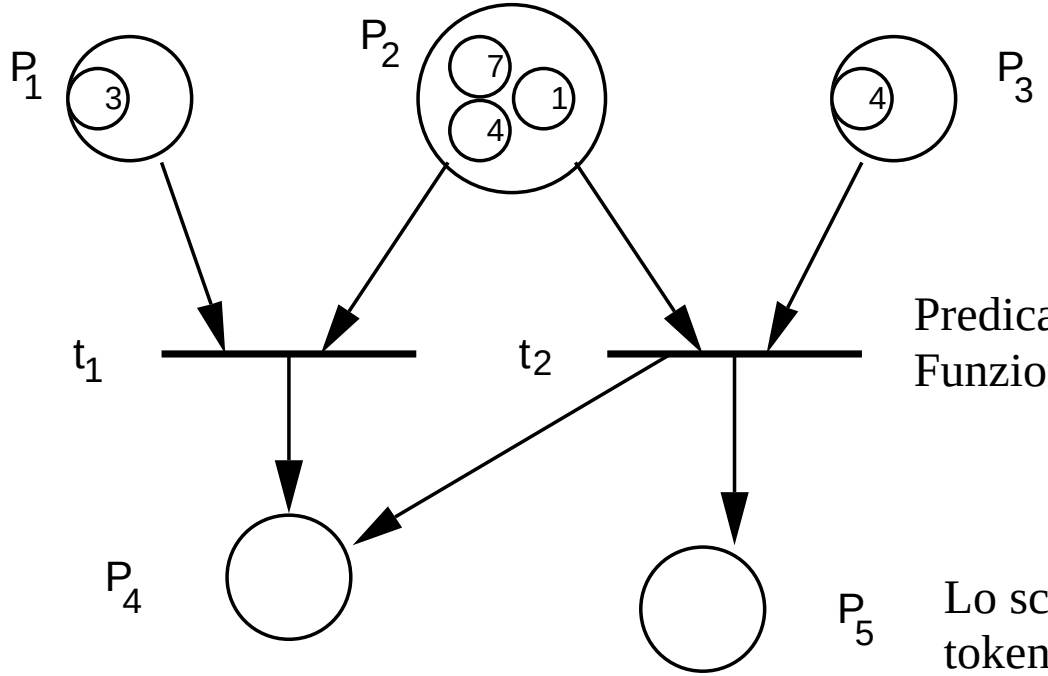
PN: limitazioni

Specifica non rappresentabile: un messaggio, a seconda del suo contenuto, viene inoltrato sul canale 1 oppure sul canale 2



Estensione delle PN dotando di valore i token

- A ogni transizione si associano un predicato e tante funzioni quanti sono gli elementi della relazione di flusso in uscita
- Il predicato rappresenta una condizione di abilitazione della transizione dipendente dai valori dei token nei posti predecessori della stessa
- Le funzioni definiscono i valori dei token da introdurre nei posti successori della transizione allo scatto della stessa



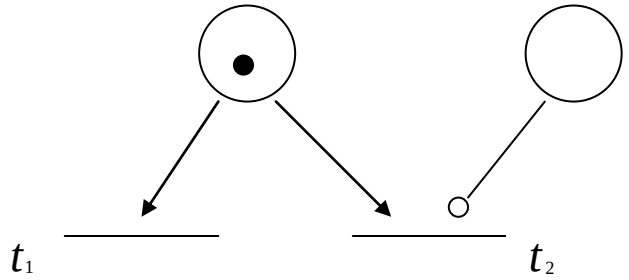
Predicato: P2 > P1
Funzione: P4 := P2 + P1

Predicato: P3 = P2
Funzioni: P4 := P2 + P3
P5 := P2 + P3

Lo scatto di t1 con <3,7> produce un token di valore 10 in P4, dopodiché t2 può scattare con <4,4>

Estensione delle PN con l'aggiunta di archi inibitori

Archi inibitori: abilitano la transizione solo in assenza di token nel posto da cui provengono



Estensione delle PN con l'aggiunta di priorità

- La priorità è una funzione $T \rightarrow N$
- Quando esistono più transizioni abilitate, lo scatto è consentito solo a quelle di priorità massima
- La scelta di quale transizione scatti fra quelle abilitate di priorità massima è nondeterministica

PN temporizzate

Reti di Merlin&Farber

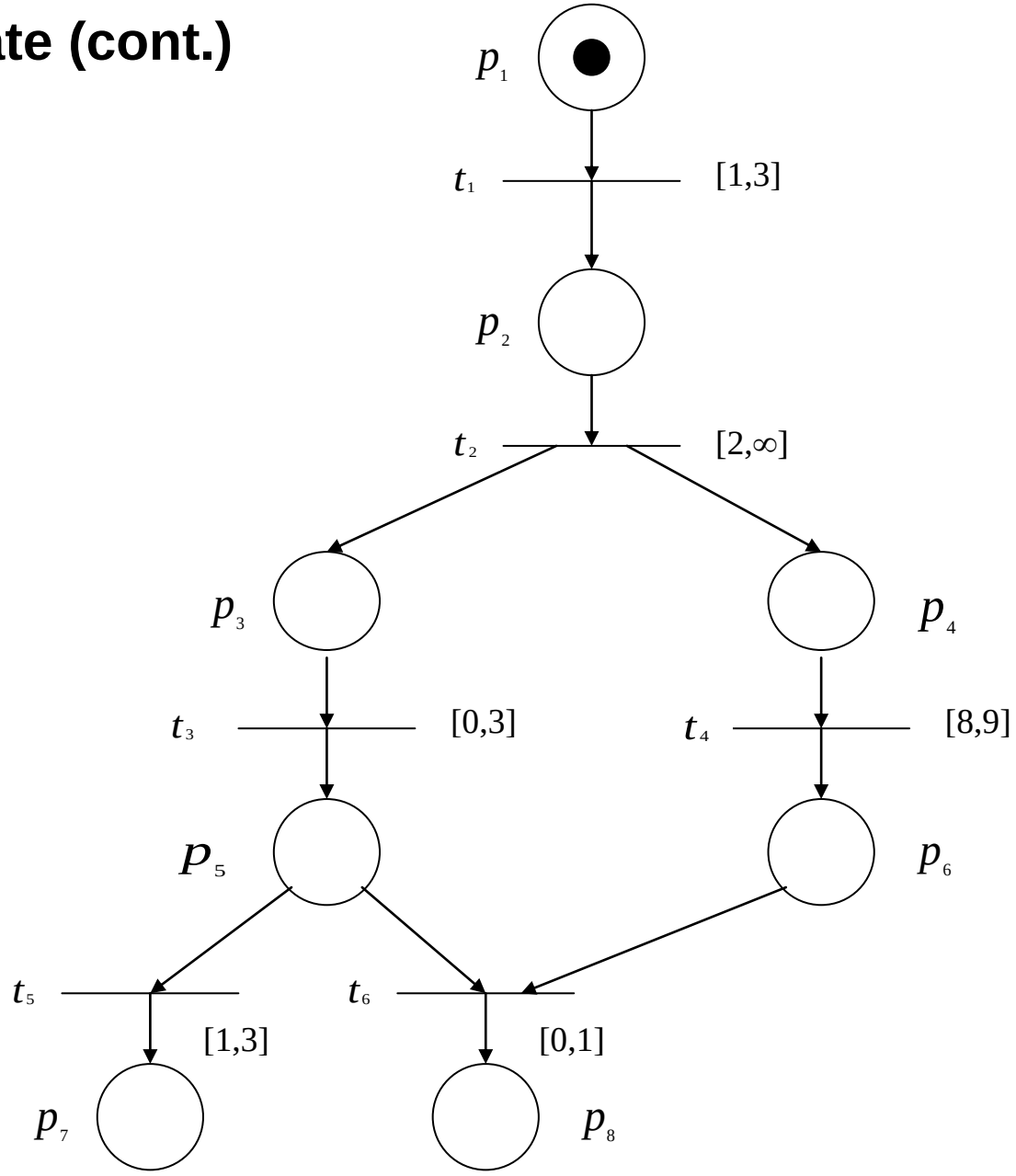
- a ogni transizione è associato un intervallo $[t_{min}, t_{max}]$ che rappresenta il tempo minimo/massimo per lo scatto → cambia la condizione di abilitazione di una transizione

⇓

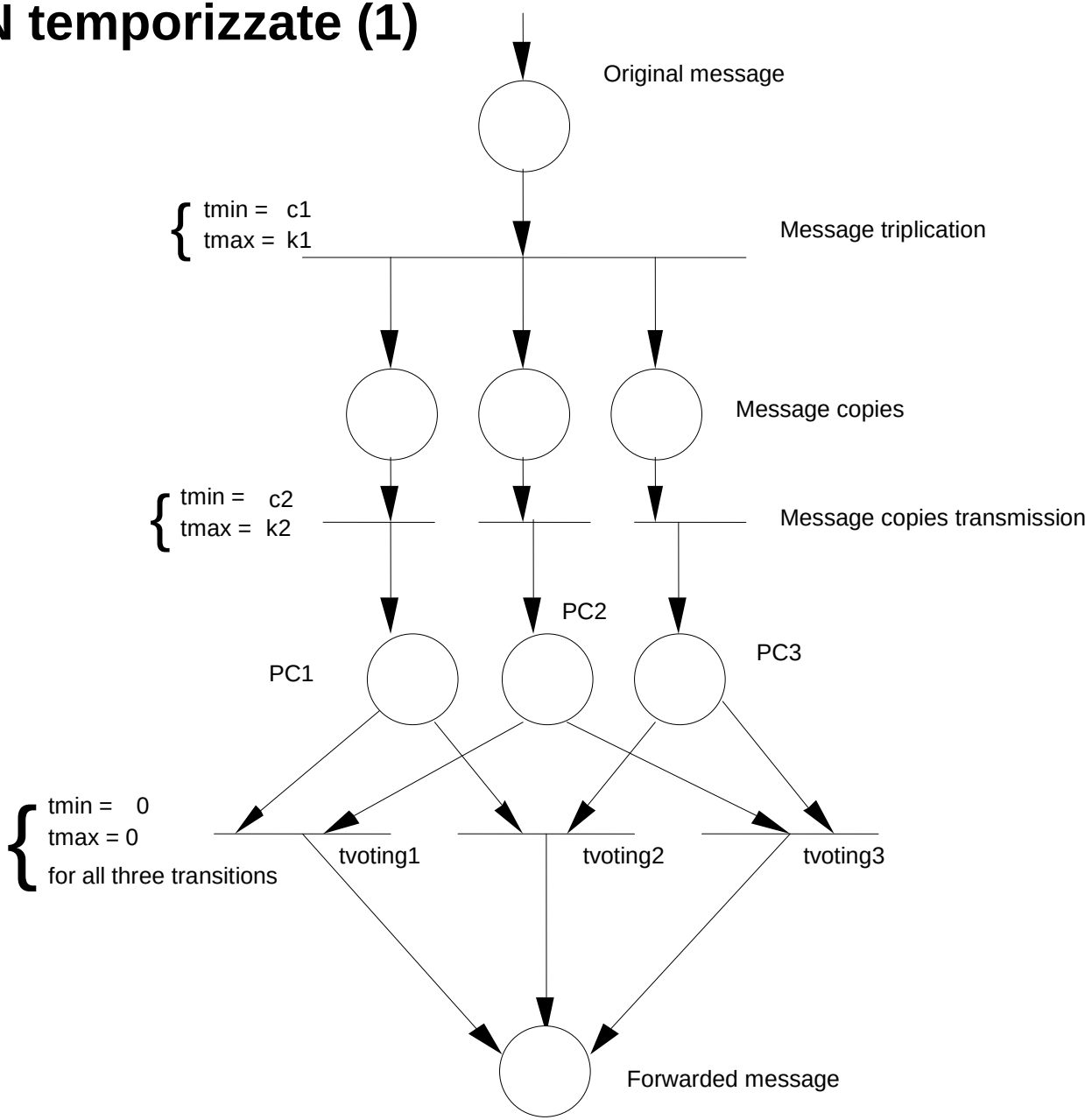
se una transizione è abilitata nel senso originale all'istante *t*, essa DEVE scattare fra l'istante $t + t_{min}$ e $t + t_{max}$, a meno che non venga disabilitata dallo scatto di un'altra transizione

- si presuppone l'esistenza di un orologio globale
- la rete è di tipo stocastico se per i tempi di scatto delle transizioni si forniscono le distribuzioni di probabilità

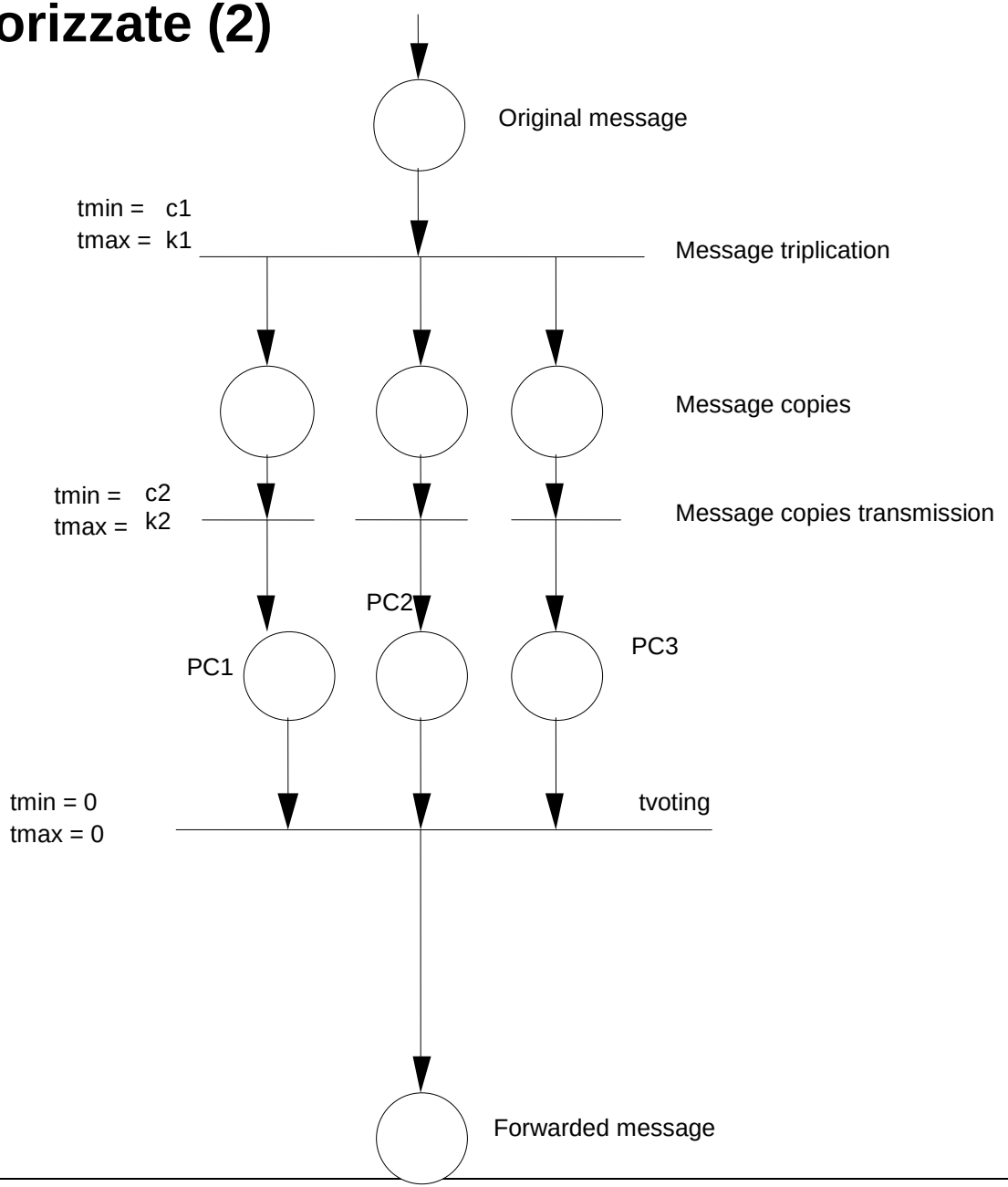
PN temporizzate (cont.)



Specifica mediante PN temporizzate (1)



Specifica mediante PN temporizzate (2)



Combinazione di due estensioni (priorità e temporizzazione)