INGEGNERIA DEL SOFTWARE

LINGUAGGI DI SPECIFICA

Avvertenza: gli appunti si basano sul corso di Ingegneria del Software tenuto dal prof. Picco della facoltà di Ingegneria del Politecnico di Milano (che ringrazio per aver acconsentito alla pubblicazione). Essendo stati integrati da me con appunti presi a lezione, il suddetto docente non ha alcuna responsabilità su eventuali errori, che vi sarei grato mi segnalaste in modo da poterli correggere.

e-mail: webmaster@morpheusweb.it
web: http://www.morpheusweb.it

SPECIFICA E NOTAZIONI

SPECIFICA ED IMPLEMENTAZIONE

I due termini non hanno una valenza assoluta, ma dipendono dalla fase in cui siamo. Nelle varie fasi del processo di sviluppo ogni implementazione che produciamo diventa la specifica per la fase successiva.

Specifica è un termine "overloaded", la nostra definizione è quella di un contratto tra un produttore ed un consumatore di un servizio.

Requisito: espresso in termini di fenomeni condivisi

Progetto: è l'interfaccia dei moduli

PROPRIETA' DESIDERABILI

Chiara, precisa, non ambigua, comprensibili, consistente, completa, incrementale

> Esempio:
> Vediamo alcune righe di una specifica per un editor di testo
>
> *Il processo di selezione è quello mediante il quale si possono definire delle aree in cui si vuole lavorare. La maggior parte delle funzioni di editino richiedono due fasi: selezione e poi azione*.
>
> E' incompleto e poco chiaro, voglio sapere quali operazioni richiedono due fasi e quali no, ed è ambiguo, non viene infatti definito il concetto di area (che è uno dei termini fondamentali)
>
> *Il testo deve essere mantenuto in linee di uguale lunghezza (specificata dall'utente), a meno che l'utente non dia un comando esplicito, si va daccapo alla fine di una parola*
>
> non dice nulla su cosa succede per parole più lunghe della lunghezza definita per la linea

NOTAZIONI

I linguaggi di specifica si distinguono in base al tipo di notazione usata in:

- **informali** (ad esempio il linguaggio naturale)
- **semiformali** (ad esempio UML, hanno una sintassi grafica precisa, ma non si possono fare controlli di tipo semantico)
- **formali** (sintassi e semantica definita in modo formale)

Dice cosa posso fare con i linguaggi. Ad esempio i linguaggi **testuali** sono più facili da testare con una macchina, mentre quelli **grafici** sono più intuitivi per l'utente.

Poi abbiamo notazioni:

- **operazionali** (descrivono il comportamento del sistema facendo riferimento a qualche macchina astratta, esempi sono gli automi a stati finiti o le reti di Petri)

- **descrittive** (i comportamenti sono dati da proprietà del sistema, ad esempio Zeta)

  Esempio:
  → *Operazionale: sia A un insieme di n elementi, il risultato dell'ordinamento è un array B di n elementi fatto in modo che il primo elemento sia il più piccolo degli elementi di A (se più elementi hanno lo stesso valore, uno qualsiasi va bene), il secondo è il più piccolo degli n-1 ottenuti da A togliendo il suo minimo e così via finché tutti gli elementi di A sono stati rimossi.*
  (Si descrive il risultato facendo riferimento ad una macchina astratta in cui ho più passi ed ogni passo si descrive cosa accade)
  → *Descrittivo: il risultato dell'ordinamento di un array A è un array B che è una permutazione di A ed è ordinato.*
  (da le proprietà. Possiamo anche completare la definizione dicendo cosa vogliono dire "permutazione ed "ordinato". Si descrivono le caratteristiche desiderate)

COME VERIFICARE UNA SPECIFICA

Voglio verificare le caratteristiche interne ed esterne. Lo si fa mediante **inspection**, **simulazione**, **test di specifica**.
Ci sono due macro approcci:

- uso di un qualche sistema (equivale al testing)

- uso di metodi formali (leggi matematiche)

CLASSI DI SISTEMI

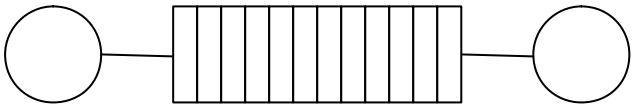
Linguaggi diversi hanno usi diversi a seconda del sistema.

Distinguiamo i sistemi in:

- **Sequenziali**: un unico flusso di esecuzione. La velocità impatta solo sulle prestazioni

- **Concorrenti**: ci sono diversi flussi di controllo logici (ad esempio i sistemi distribuiti, oppure quelli multiprocessore). La temporizzazione delle attività impatta spesso sulla correttezza del sistema.

  Esempio: produttore – consumatore



producer *buffer* *consumer*

Il buffer è la risorsa condivisa. Requisito è la sincronizzazione tra produttore e consumatore.

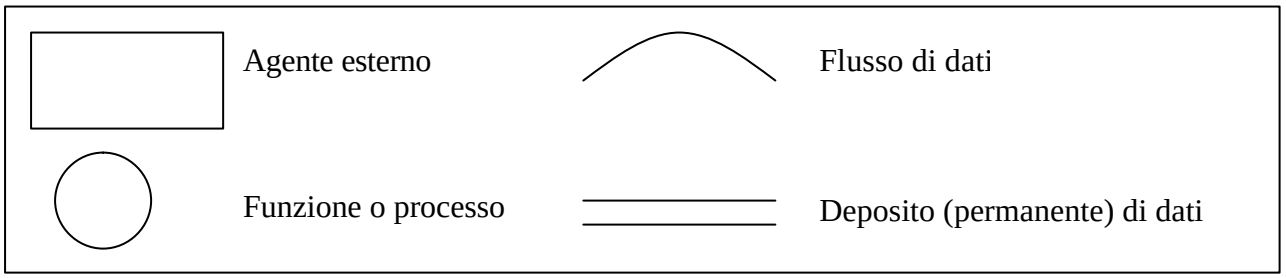
- **Real time**: la correttezza dipende anche dalla durata di esecuzione dei processi. Real time non vuol dire però sistemi veloci, i tempi di risposta non devono per forza essere bassi, però devono essere predicabili.
  I sistemi real time hanno rilevanza perché spesso sono safety critical (aerei, sistemi militari, medici, ecc…)

Regola empirica: c'è un fattore 3 in termini di costo quando si passa da un tipo di sistema all'altro.

DATA FLOW DIAGRAMS

I diagrammi di flusso dei dati concentrano la descrizione del sistema informativo sulle operazioni effettuate sui dati e sulle dipendenze funzionali che si creano in virtù dei flussi di informazione esistenti tra i processi.

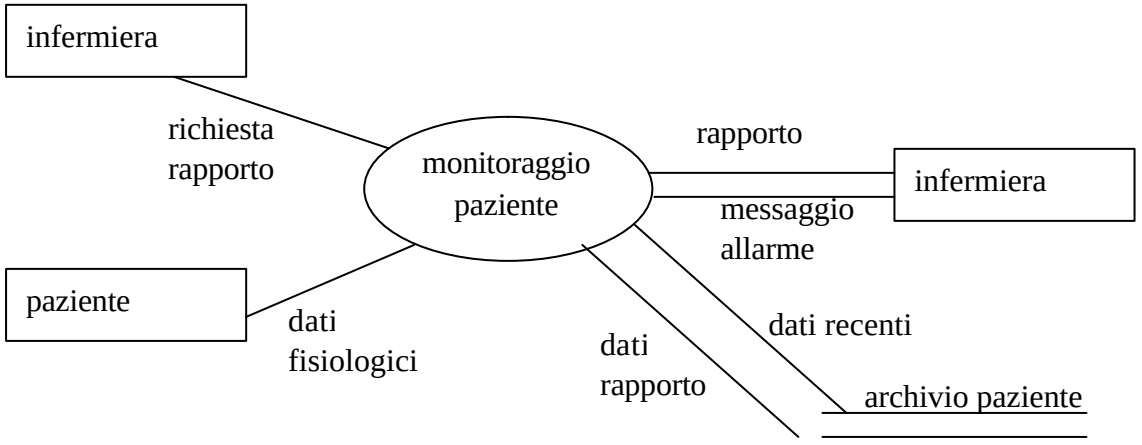
Oltre alle funzioni (rappresentate da cerchi), i data flow diagram possono includere agenti esterni al sistema, rappresentati mediante scatole rettangolari e non modellati ulteriormente, depositi di dati da usare come sorgente o destinazione di informazione permanente, rappresentati con coppie di linee parallele, e flussi di dati, rappresentati mediante archi orientati, scambiati tra funzioni oppure tra una funzione ed un componente di diverso tipo.



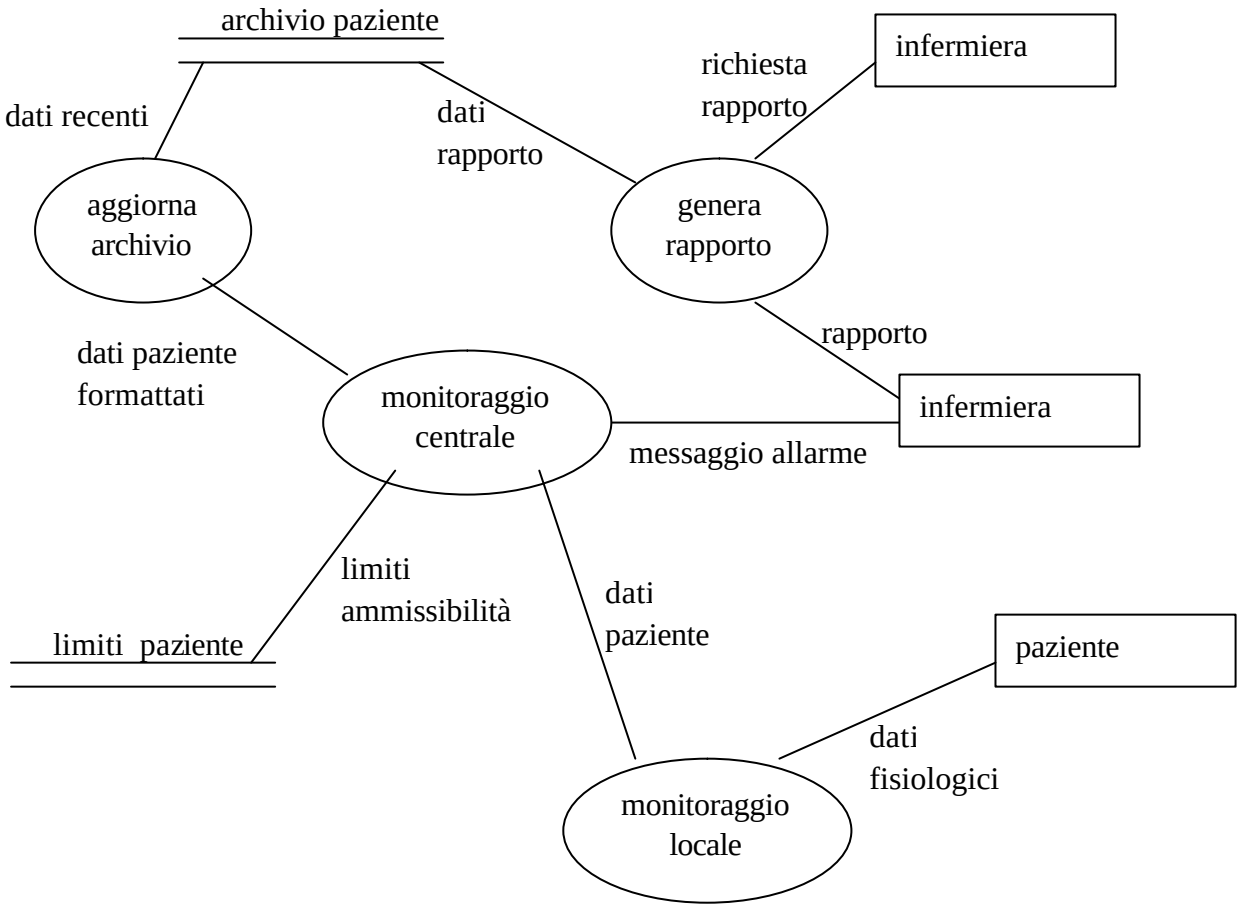
RAFFINAMENTI SUCCESSIVI

La specifica del sistema avviene tramite raffinamenti successivi e specializzazioni a partire dal diagramma iniziale.
Nel raffinamento di una funzione in un DFD, deve essere rispettato un vincolo detto della *continuità del flusso informativo,* che prescrive che nel diagramma siano presenti gli stessi flussi presenti in quella che si sta dettagliando.

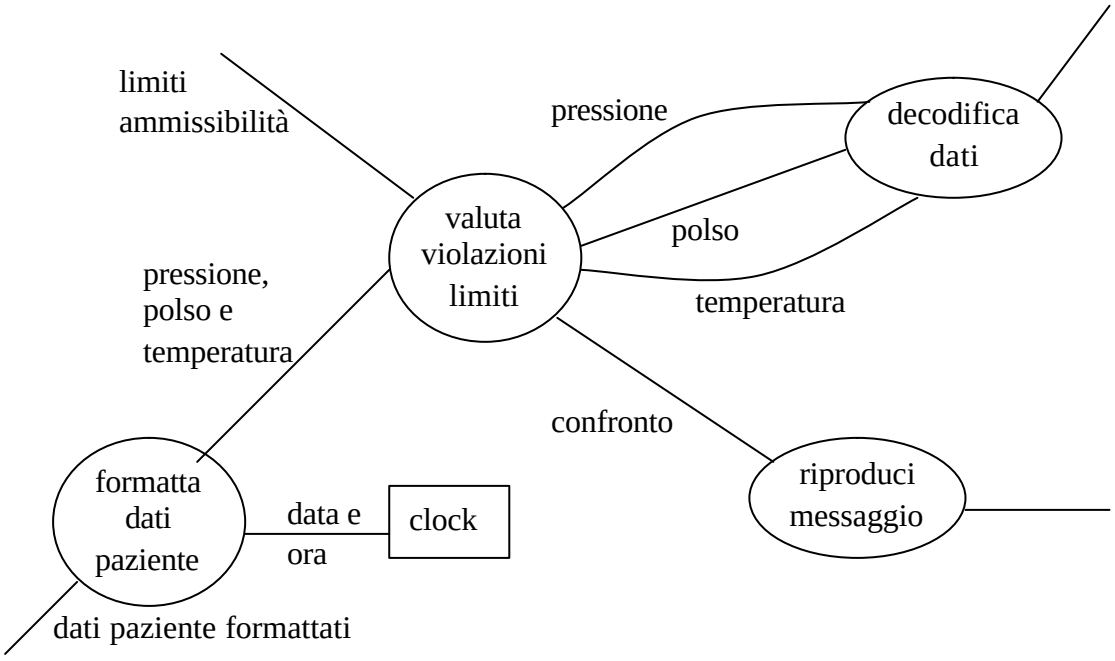
Esempio: monitoraggio di un paziente
I segnali di vita del paziente sono misurati periodicamente e convertiti in forma manipolabile da programma; essi vengono confrontati con i dati ammissibili per il paziente e memorizzati su un file. Se i dati del paziente sono al di fuori dei limiti prestabiliti, viene generato un messaggio di allarme che richiama il personale. Inoltre l'infermiere può occasionalmente chiedere un rapporto sulle condizioni del paziente, in questo caso il rapporto viene generato a partire dall'archivio storico.



Vediamo un raffinamento in cui enucleiamo le funzioni: monitoraggio centrale, aggiorna archivio e genera rapporto:



vediamo poi un dettaglio della funzione monitoraggio centrale.



CRITERI DI STESURA DEI DFD

- Ignorare le operazioni per l'inizializzazione del sistema, per la sua terminazione e per la gestione degli errori
- Ignorare il flusso di controllo e la sincronizzazione tra i processi
- Individuare le entrate e le uscite nette del sistema
- Assegnare ai flussi nomi significativi
- Assegnare alle funzioni nomi significativi

DIZIONARIO DEI DATI

Per la descrizione dei dati che compaiono in un DFD, viene spesso associato un dizionario dei dati in cui la struttura dei dati viene descritta usando le notazioni dei linguaggi formali.

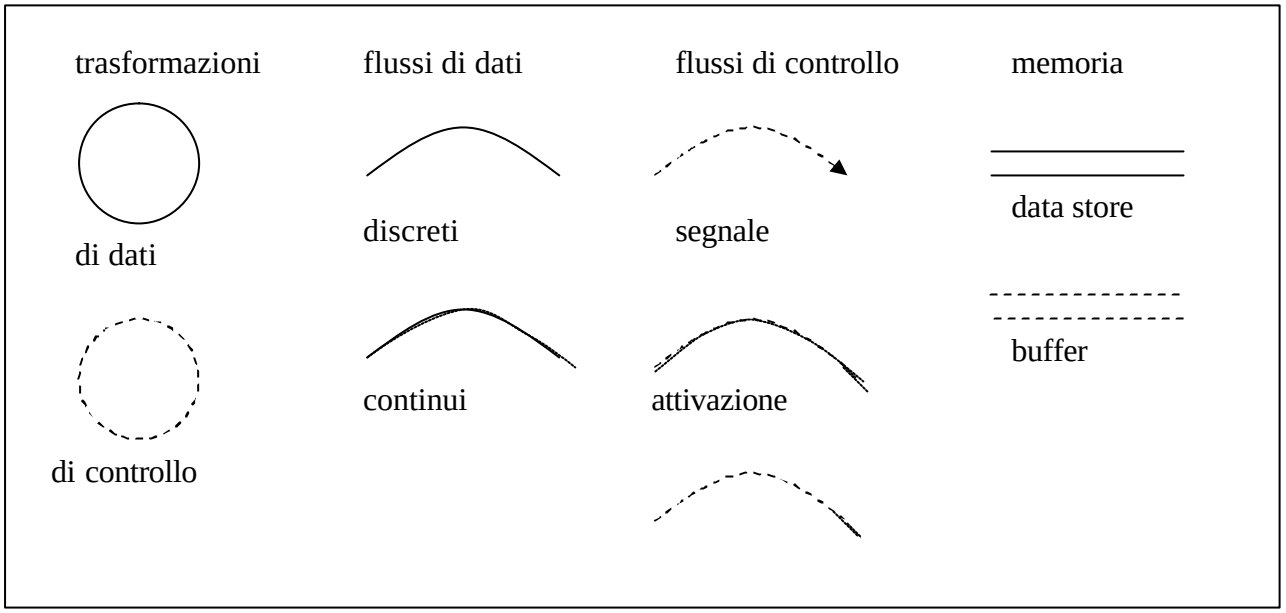
Costrutto	**Notazione**	**Significato**
Definizione	A≡B	A è definito come B
Sequenza	A+B	A concatenato con B
Selezione	[A|B]	A oppure B
Ripetizione	$\{A\}_i^j$	A ripetuto min i, max j volte
Opzionalità	[A]	A oppure nulla

DESCRIZIONE DELLE FUNZIONI

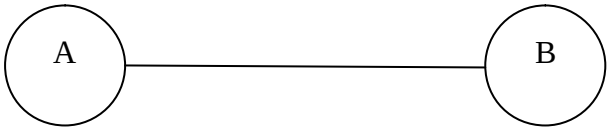
Le funzioni vanno invece descritte in pseudocodice.

SCHEMI DI TRASFORMAZIONE

Sono un'estensione dei DFD volta alla rappresentazione degli aspetti temporali e di sincronizzazione dei sistemi specificati.
Gli elementi relativi al controllo sono tutti rappresentati mediante simboli grafici della stessa forma dei corrispondenti dei DFD, ma tratteggiati.

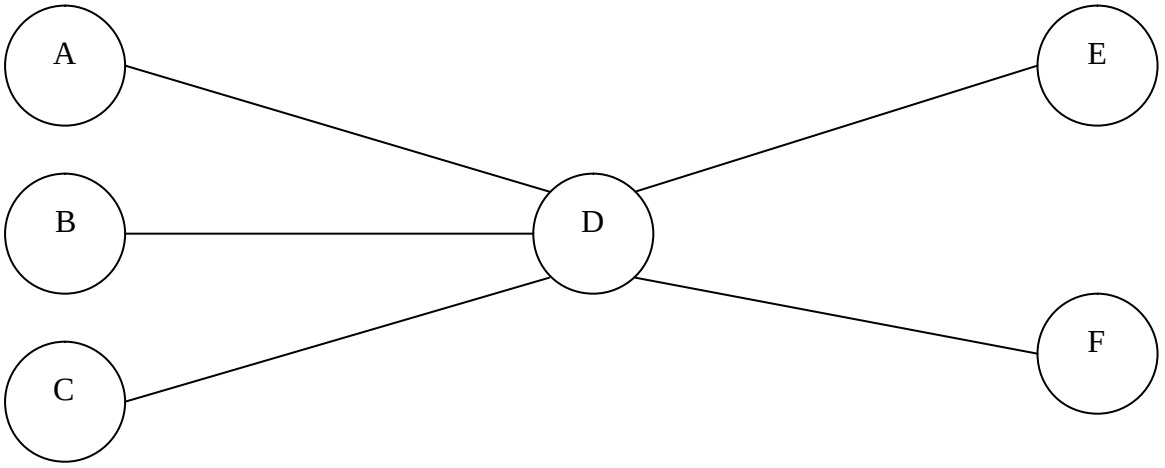

trasformazioni
flussi di dati
flussi di controllo
memoria
di dati
discreti
segnale
data store
buffer
continui
attivazione
di controllo

AMBIGUITA' DEI DFD



A e B rappresentano due attività distinte che si svolgono in modo indipendente ed asincrono. Di fatto esistono tre modi in cui A e B possono scambiarsi i dati:

- A potrebbe produrre un dato ed aspettare fino a che B non l'ha consumato.
- B potrebbe usare il dato in uscita da A più di una volta senza consumarlo
- A e B potrebbero averevelocità differenti nel produrre e consumare i dati.



Per gli ingressi:

- D potrebbe aver bisogno per essere attivata, di tutti i dati provenienti dalle funzioni A, B, C
- D potrebbe aver bisogno solo di uno tra questi dati
- potrebbero esserci regole più complesse

Per le uscite:

- D potrebbe inviare una dato distinto a e ed F
- D potrebbe inviare lo stesso dato a E ed F
- casi più complessi quali ad esempio uscite alternate a E ed F

AUTOMI A STATI FINITI

Un automa può essere rappresentato per mezzo di un grafo orientato.

In generale un automa può essere definito da una sestupla **<S, I, U, d, t, s_0>**

in cui:

- S è l'insieme (finito e non vuoto) degli stati
- I è l'insieme degli ingressi
- U è l'insieme delle uscite
- d è la funzione di transizione d: $S \times I$? S
- t è la funzione di uscita t: $S \times I$? U
- $s_0 \in S$ è lo stato iniziale

Nel passaggio dalla forma matematica a quella **grafica**, si associa:

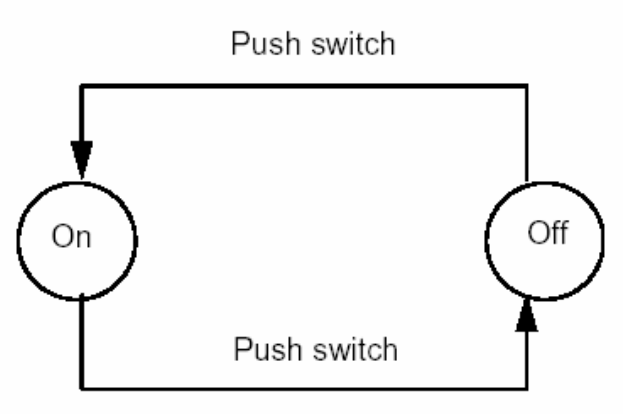
- a ogni stato un nodo del grafo
- a ogni coppia (stato, ingresso) per cui la funzione di transizione sia definita un arco del grafo
- a ogni arco si associano i corrispondenti valori ingresso e uscita
- lo stato iniziale viene caratterizzato dal nodo con un arco entrante che non proviene da alcun altro nodo

E' poi possibile rappresentare un automa con una **tabella**.

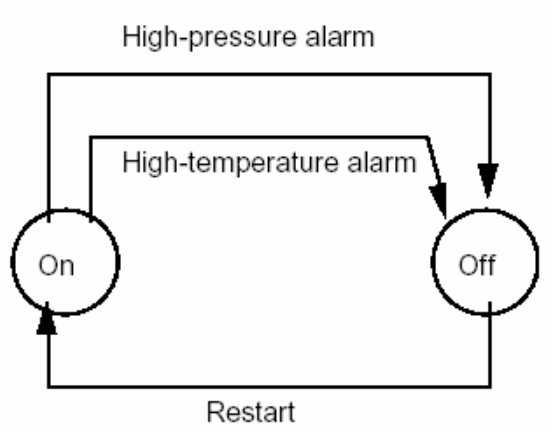
- a ogni stato dell'automa si associano una riga e una colonna di una matrice quadrata.
- se esiste una transizione che porta da uno stato s_i ad uno stato s_j, l'elemento della matrice in posizione (i,j) contiene tanto l'ingresso che provoca la transizione di stato, quanto l'uscita corrispondente.

ESEMPI

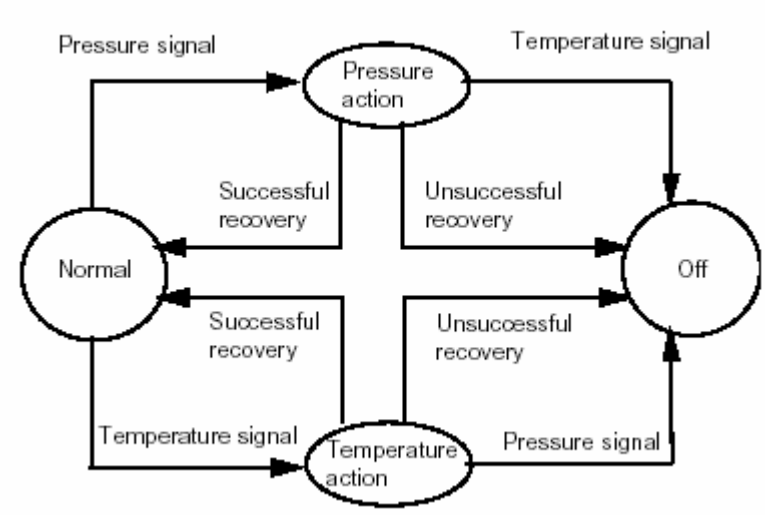
- Comportamento di una lampada:



- Sistema di controllo

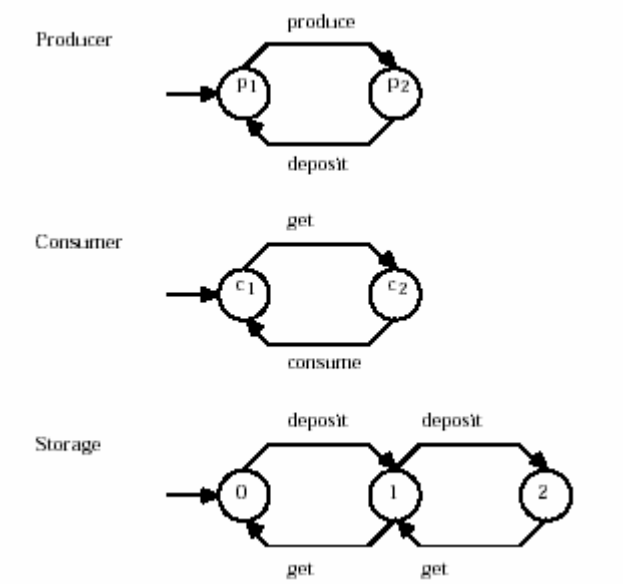


Raffinamento:



LIMITI

- Memoria finita
- Esplosione degli stati



Vengono preferiti gli STATECHARTS (UML) e le RETI DI PETRI

RETI DI PETRI

DEFINIZIONE DI RETE

Una rete N è una tripla **N = (P, T, F)** con

- **P** = insieme dei posti
- **T** = insieme delle transizioni
- **F** = relazione di flusso

P e T sono due insiemi finiti.

Devono inoltre valere le seguenti proprietà:

- **$P \cap T = \varnothing$**
  Gli insiemi dei posti e delle transizioni sono disgiunti
- **$P \cup T = \varnothing$**
  La rete non è vuota (esiste almeno un posto o una transizione)
- **$F \subseteq (P \times T) \cup (T \times P)$**
  Posti e transizioni sono tra loro in relazione tramite F che lega posti a transizioni e transizioni a posti, ma non posti a posti o transizioni a transizioni

RETE POSTI/TRANSIZIONI

Una rete posti/transizioni (rete P/T) è una quintupla

$P/T = (P, T F W M_0)$

dove P, T ed F definiscono una rete e W ed M_0 sono due funzioni:

- **$W: F \rightarrow N - \{0\}$**
  W associa a ogni elemento della relazione di flusso un numero intero positivo detto peso o molteplicità
- **$M_0: P \rightarrow N$**
  detta marcatura iniziale della rete P/T, associa ad ogni posto un numero intero non negativo

La marcatura iniziale indica l'insieme degli stati parziali, ossia lo stato globale, in cui la rete si trova all'inizio della sua evoluzione.

5La rete P/T evolve, portandosi in nuove marcature in cui effettivamente la rete può giungere dopo che una qualsiasi successione ammissibile di eventi si sia verificata.

Ossia individua il sottoinsieme di tutte le possibili funzioni P $\rightarrow$ N (le quali rappresentano tutte le marcature che possono essere attribuite a una rete P/T), detto delle marcature raggiungibili.

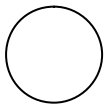
Ciò equivale a dire che lo stato iniziale del sistema individua tutti gli stati che da un tale stato possono essere raggiunti.

RAPPRESENTAZIONE GRAFICA

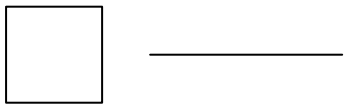
Una rete P/T viene rappresentata come un *grafo bipartito*, ossia i cui nodi sono di due tipi distinti (posti e transizioni) e sono collegati tramite archi orientati (gli elementi della relazione di flusso).

Tramite cerchi si rappresentano i posti e tramite quadrati o barre, si rappresentano le transizioni e tramite archi orientati si rappresentano i flussi.

La funzione di peso W è rappresentata con un'annotazione sull'arco corrispondente, mentre la marcatura M viene rappresentata con dei *token* (marche) rappresentati da tondini neri all'interno di un posto, di un numero uguale al valore che la funzione M assume nel posto.

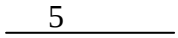


posto



transizione

flusso



peso

marcatura

EVOLUZIONE DELLA RETE

Abbiamo visto gli aspetti statici della rete.

Per modellare il verificarsi di un evento occorre considerare due aspetti:

- la possibilità che l'evento si verifichi (*abilitazione della transizione*)
- l'effetto che l'evento ha sullo stato (*scatto della transizione*)

ABILITAZIONE DI UNA TRANSIZIONE

Una transizione *t* è abilitata nella marcatura M se e solo se

$$\forall p \in \mathrm{Pr}e(t), (M(p) \geq W(< p,t >))$$

Il fatto che una transizione sia abilitata in una determinata marcatura M viene abbreviato con la notazione

$$M[t >$$

L'insieme dei token che abilita una transizione viene detto *tupla abilitante di t*.

REGOLA DI SCATTO DI UNA TRANSIZIONE

Lo scatto di una transizione t abilitata, produce a partire dalla marcatura M, una nuova marcatura M' nel modo seguente:

- ad ogni posto *p* che sia in ingresso a *t* viene rimosso un numero di token uguale al peso dell'arco che collega *p* a *t* ed in ogni posto *q* in uscita a *t* viene depositato un numero di token uguale al peso dell'arco che collega *t* a *q*.
- la marcatura dei posti che non siano ne di ingresso ne di uscita, rimane inalterata.

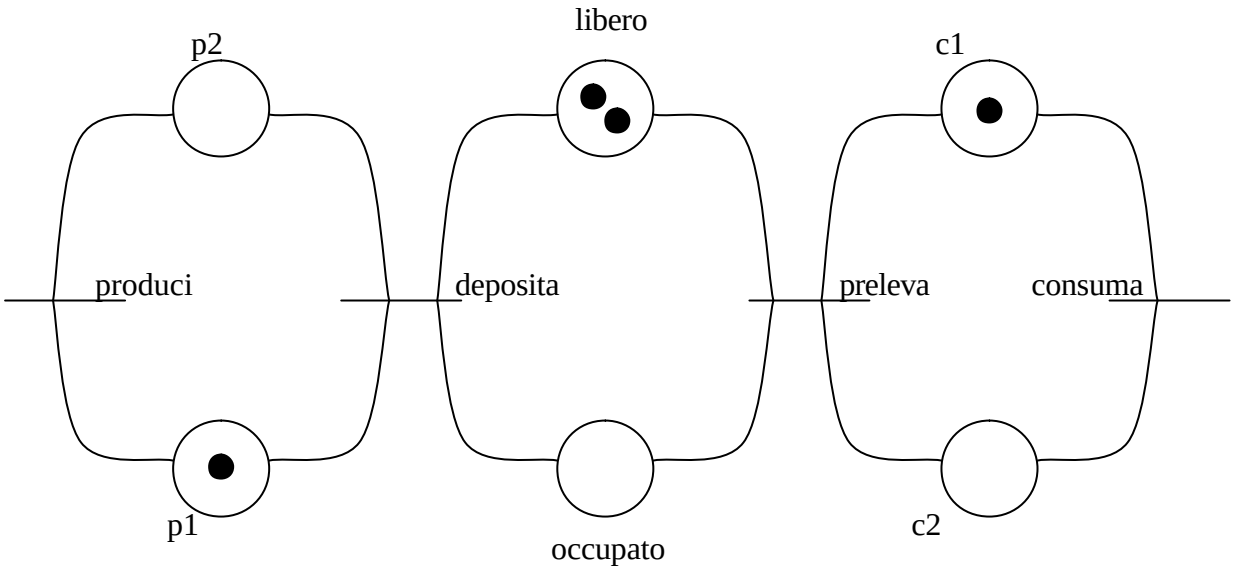
In modo formale:

Data la marcatura M, lo scatto di una transizione *t* abilitata, produce una M' tale che:

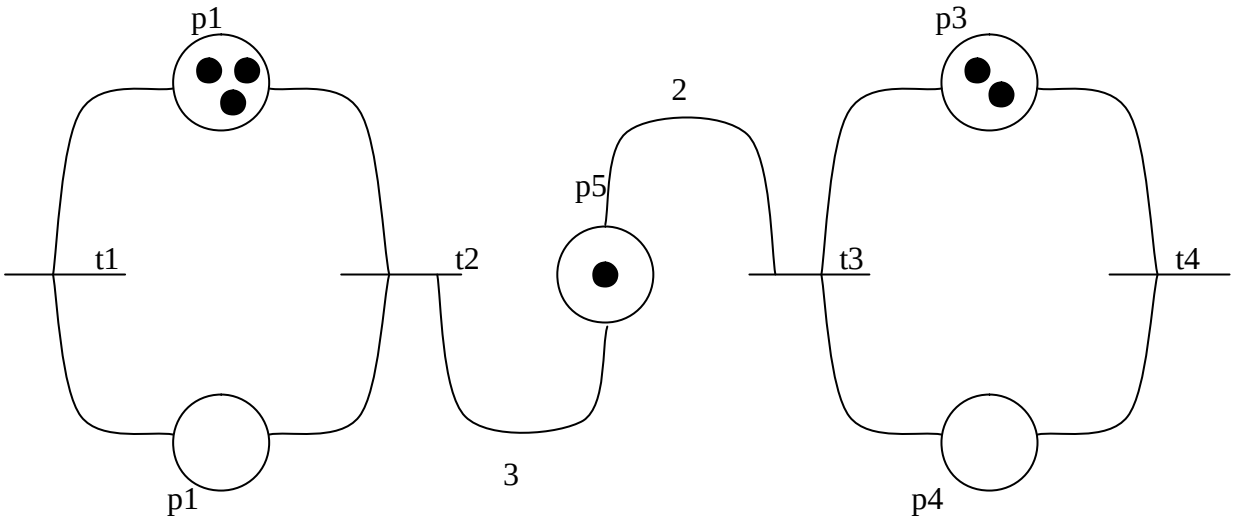
$$\begin{array}{ll}
\forall p \in \mathrm{Pr}e(t) - Post(t) & M'(p) = M(p) - W(< p,t >) \\
\forall p \in Post(t) - \mathrm{Pr}e(t) & M'(p) = M(p) + W(< p,t >) \\
\forall p \in Post(t) \cap \mathrm{Pr}e(t) & M'(p) = M(p) - W(< p,t >) + W(< t,p >) \\
\forall p \in P - (\mathrm{Pr}e(t) \cup Post(t)) & M'(p) = M(p)
\end{array}$$

Per indicare lo scatto di una transizione si usa la notazione: $M[t > M'$

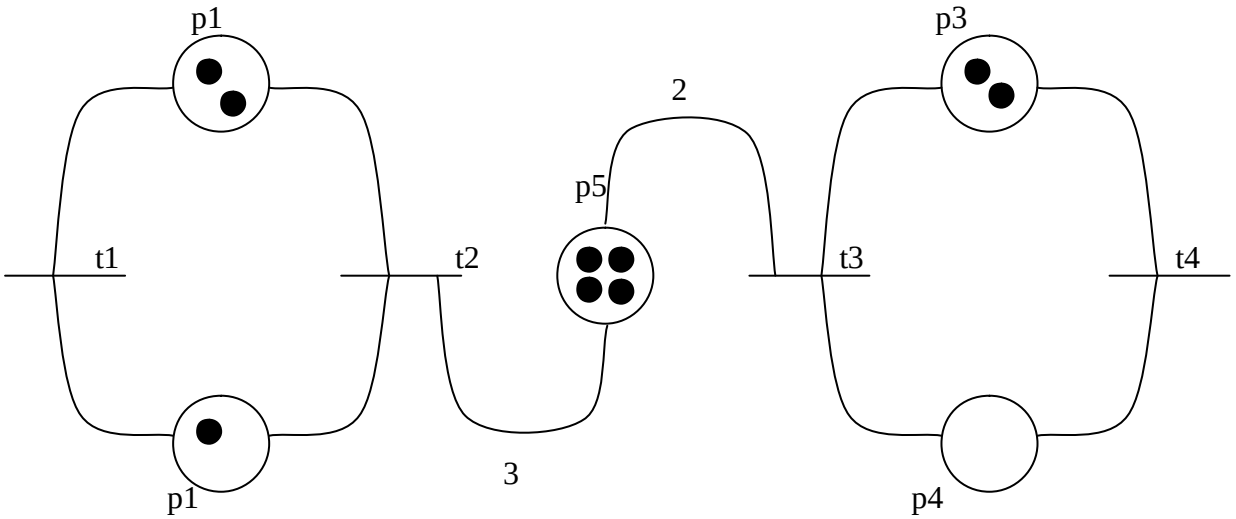
ESEMPIO: PRODUTTORE CONSUMATORE



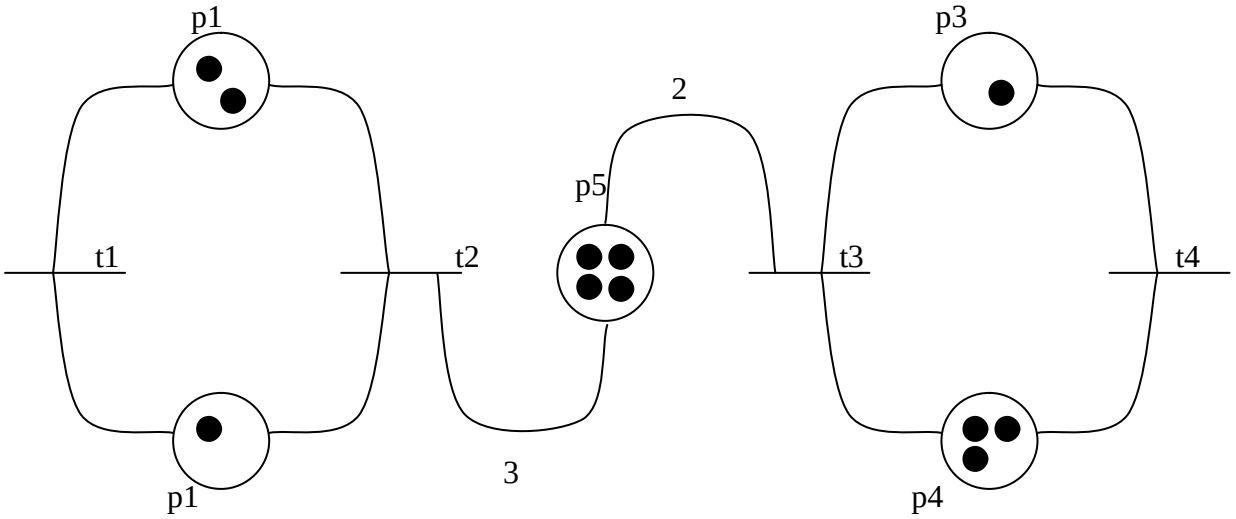
ESEMPIO: BUFFER



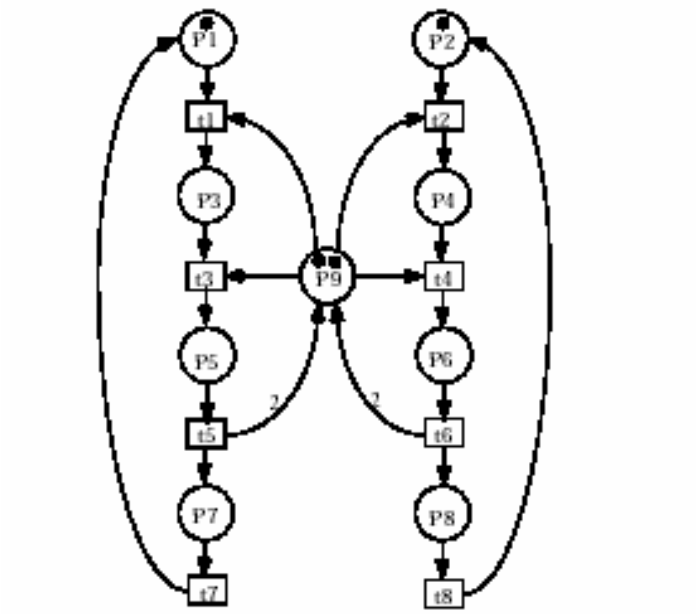
Scatta t2 (t2 produce 3 token; t3 ha bisogno di 2 token in p5 per scattare)



Scatta t3

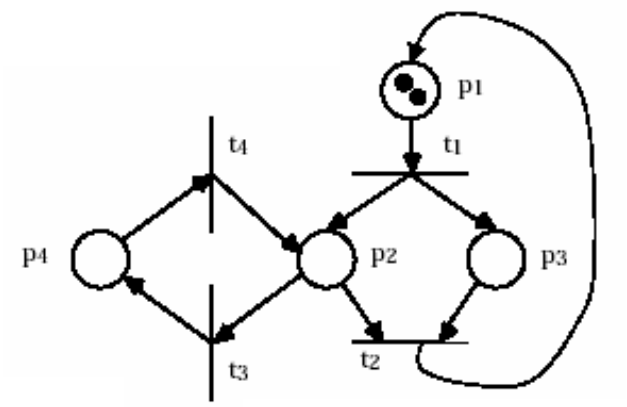


DEADLOCK

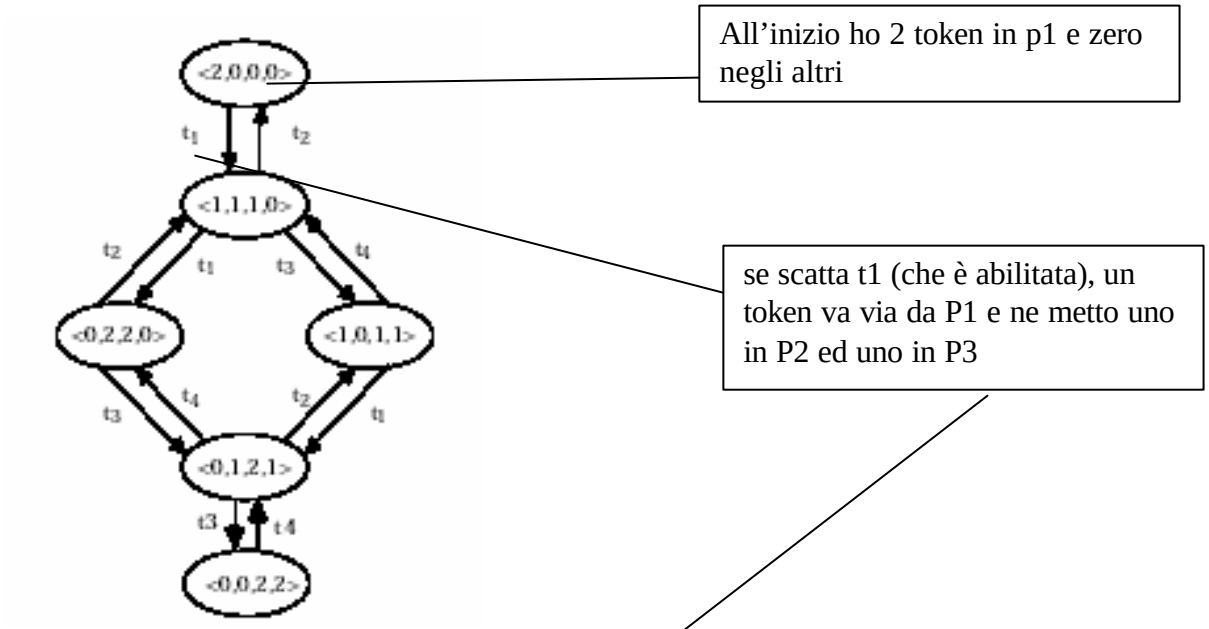


Scattano t1 e t2, in P9 non ho più token e non può scattare t3.
Entrambi i lati aspettano le risorse l'uno dell'altro.

ANALISI DI RAGGIUNGIBILITA’

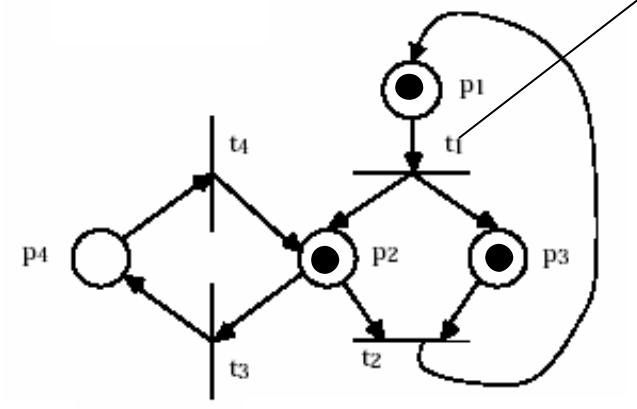


Dalla rete creo un grafo



All’inizio ho 2 token in p1 e zero negli altri

se scatta t1 (che è abilitata), un token va via da P1 e ne metto uno in P2 ed uno in P3



E così via si costruisce il grafo tramite cui vedo le transizioni che possono scattare in sequenza, gli stati raggiungibili…