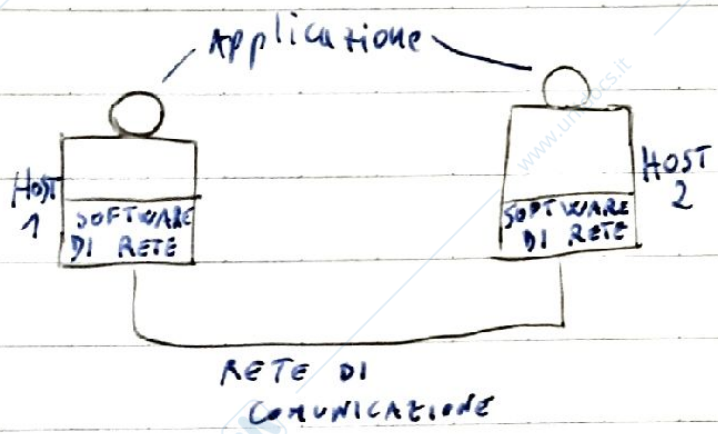
No. SISTEMI INF.

Mo Tu We Th Fr Sa Su

Date 13 · 11 · 19

CONTROLLO CON SISTEMI DISTRIBUITI

1. TECNOLOGIE RETI
2. REAL TIME SU RETI
3. SINCRONIZZAZIONI



Dobbiamo eseguire un processo entro le deadline tempordi che consentono la comunicazione

Strategie di commutazione:

- COMMUTAZIONE A CIRCUITO
- COMMUTAZIONE A MESSAGGIO
- COMMUTAZIONE A PACCHETTO

Vogliamo garantire l'accesso al mezzo trasmissivo agli host che hanno deadline

La commutazione a circuito non va bene

↓

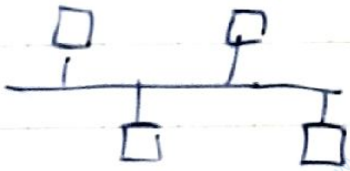
ossia lascio la linea trasmissiva esclusivamente a 2 host

La commutazione a messaggio non va bene

→ L'unica che possiamo utilizzare è una commutazione a pacchetto

Topologia della rete:

bus:



Stella:

Utilizzeremo la topologia a stella poiché nel caso in cui un host prende possesso del mezzo trasmissivo gli altri host non hanno più accesso.

Mo Tu We Th Fr Sa Su

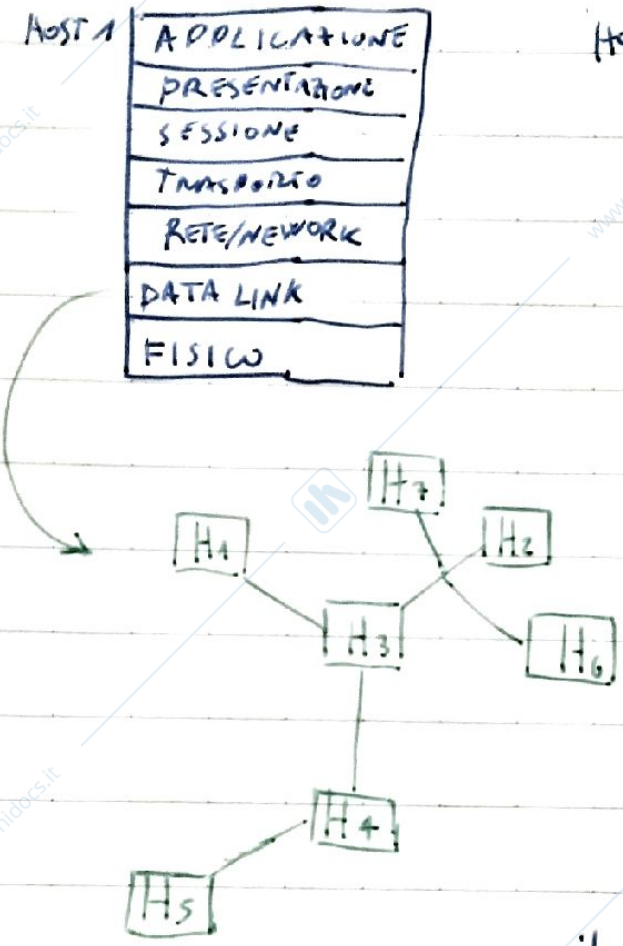
No. SISTEMI INF.

Date 13 · 11 · 12

Il protocollo usato da Ethernet (CSMA/CD) non garantisce il Real Time.

Pila protocollare:

ISO/OSI



Ogni pacchetto in circolo sulla rete viaggia un pacchetto e ogni strato della pila aggiunge un header

il pacchetto per essere trasmesso tra 2 host dovrà probabilmente passare per altri host

es. da H_2 a H_5

Prossimo destinatario: H_3 — poi H_4, infine H_5

Ultimo destinatario: H_5

A livello Data Link specifichiamo il Prossimo destinatario

A livello di Rete → Ultimo destinatario

4

No. SISTEMI INF.

Date 13 · 11 · 17

Standard:

- FISICO + STANDARD ⇒ ETHERNET
- RETE ⇒ IP
- TRASPORTO ⇒ TCP/UDP

TCP → SICURO

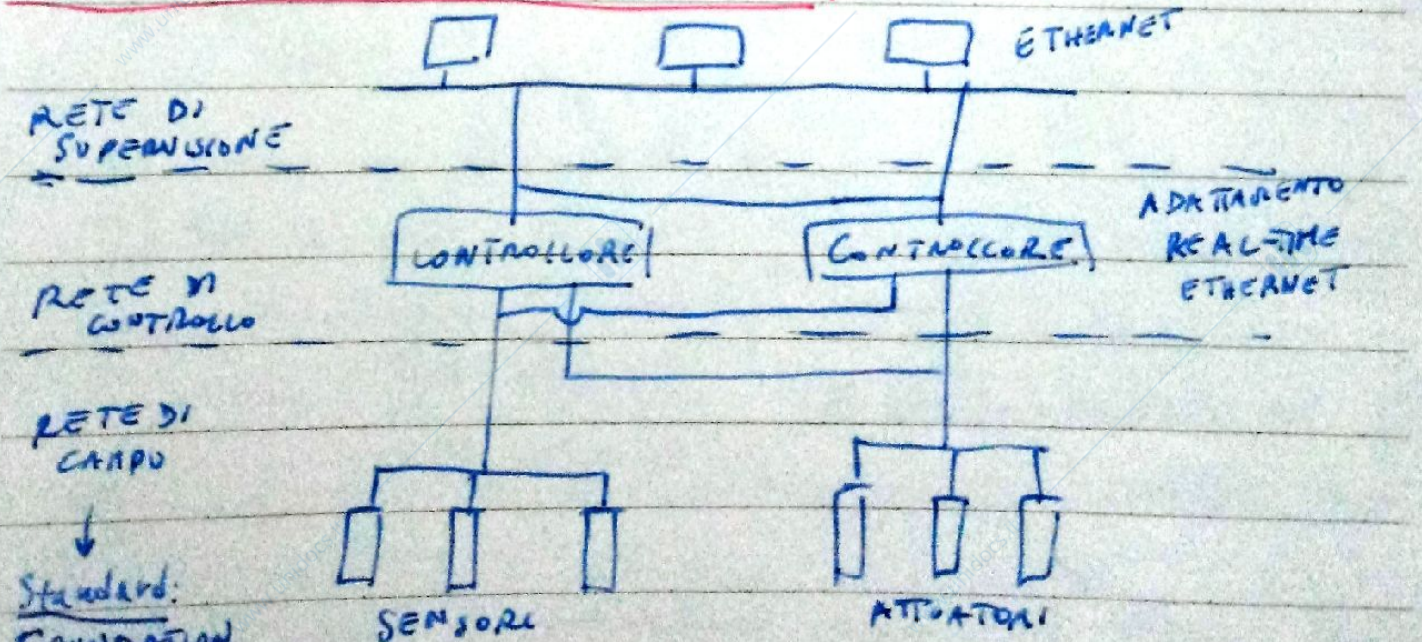
- TCP/IP: ci permette di avere una sorta di "certificato di consegna" del singolo pacchetto
- Ricostruzione del messaggio a partire dall'ordinamento corretto dei pacchetti

Vincolo: tempo

UDP → NON SICURO

Non abbiamo la conferma di consegna ↓ non possiamo ricostruire il messaggio

Sistema di controllo distribuito:



Standard: FOUNDATION FIELD BUS

Mo Tu We Th Fr Sa Su

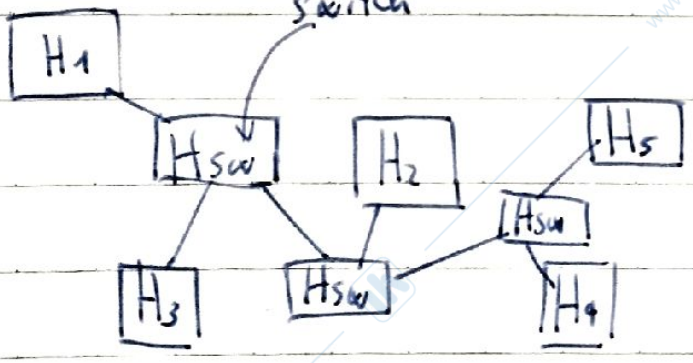
No. SISTEMI INF.

Date 13 · 11 · 19

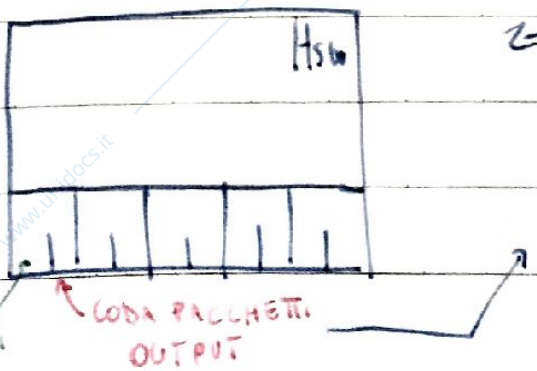
Foundation Fieldbus:

- FULL DUPLEX
- ~32 kbit/s
- NON RISPETTA COMPLETAMENTE LA PILA PROTOCOLLARE ISO/OSI
  - non sono presenti RETI, TRASPORTO, SESSIONE
  - ha un livello applicativo → LIVELLO UTENTE
- in livello DATA LINK è presente uno scheduler dei pacchetti per garantire real time

Adattamento ethernet per real time



lo switch non manda il pacchetto a tutti quelli a cui è collegato ma soltanto al prossimo destinatario del percorso per arrivare all'ultimo destinatario



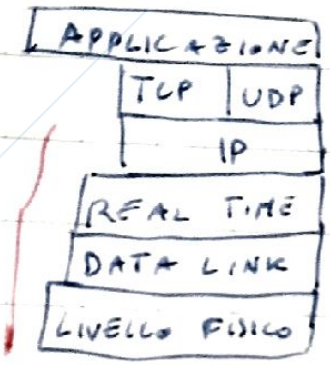
I pacchetti in output sono ordinati rispetto alla deadline

6

No. SISTEMI INF.

Date 13 · 11 · 19

Dal punto di vista protocollare possiamo usare TCP/UDP, IP + LIVELLO REALTIME



I pacchetti vengono ordinati per deadline crescente (EDP)

Fino ad adesso abbiamo ragionato in quanti di tempo → anche qui usiamo i quanti

quanti visti come istante tempo per comunicare il pacchetto sul link (velocità di trasmissione)

circa 125 μs

Prendiamo il disegno precedente

$H_1 \rightarrow H_4$ $d_1 \rightarrow$

↑ voglio assegnare una deadline

Dobbiamo considerare che il pacchetto può essere ritrasmesso un pò di volte, con un certo periodo

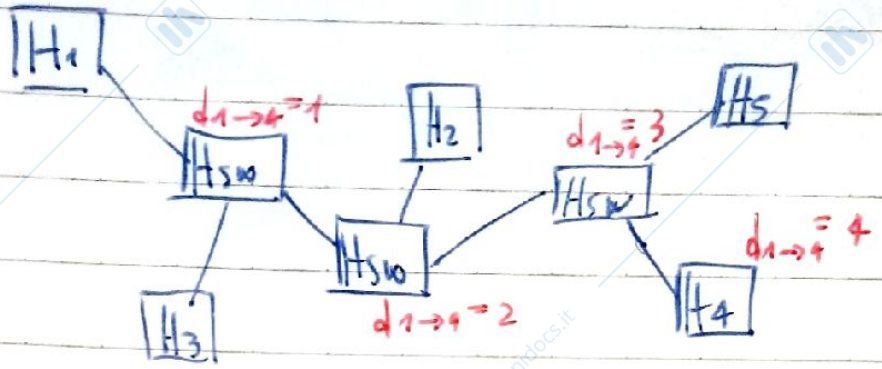
7

Mo Tu We Th Fr Sa Su

No. SISTEMI INF.

Date 13 · 11 · 19

Dal disegno si evince che per arrivare ad H_4 deve fare 4 ritrasmissione



Ogni trasmissione avrà una deadline

Dunque ho: — CANALI DI COMUNICAZIONE

es. $H_1 \rightarrow H_4 \quad d_{1\rightarrow 4} = 8$

$H_5 \rightarrow H_3 \quad d_{5\rightarrow 3} = 10$

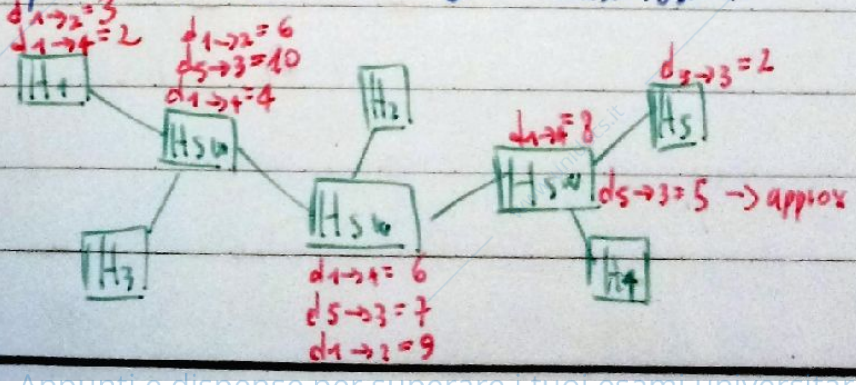
$H_1 \rightarrow H_2 \quad d_{1\rightarrow 2} = 9$

$H_2 \rightarrow H_5 \quad d_{2\rightarrow 5} = 12$

Vogliamo capire come distribuire le deadline sui nodi intermedi

Se distribuiamo in modo uniforme dovremo dividere la deadline per il numero di trasmissioni

↳ es.

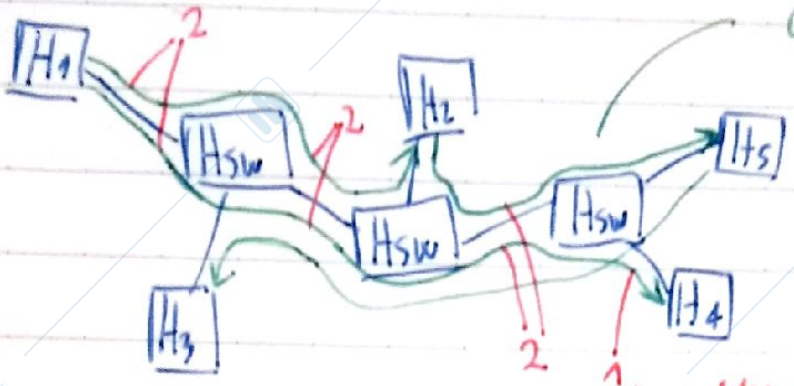


Mo Tu We Th Fr Sa Su

No. SISTEMI INF.

Date 13 · 11 · 19

Un modo più efficiente di distribuire le deadline è:



Per distribuire la deadline in modo che tenga in considerazione il peso (numero di canali sopra) dei link.

$$d_{1\to4}^{1^\circ link} = \frac{2\cdot 8}{2+2+2+1} = \frac{16}{7}$$

$$d_{1\to4}^{2^\circ link} = \frac{(2+2)\cdot 8}{7} = \frac{32}{7}$$

$$d_{1\to4}^{3^\circ link} = \frac{48}{7}$$

$$d_{1\to4}^{4^\circ link} = 8$$

→ si arrotonda in caso verso il basso a partire dal 1° link: poi vedi in base a come hai arrotondato prima

$$d_{2\to5}^{1^\circ link} = \frac{1\cdot 12}{1+2+1} = 3$$

$$d_{2\to5}^{2^\circ link} = \frac{(1+2)\cdot 12}{4} = 9$$

$$d_{2\to5}^{3^\circ link} = 12$$

N.B direzioni canali si sommano solo se sono nella stessa direzione

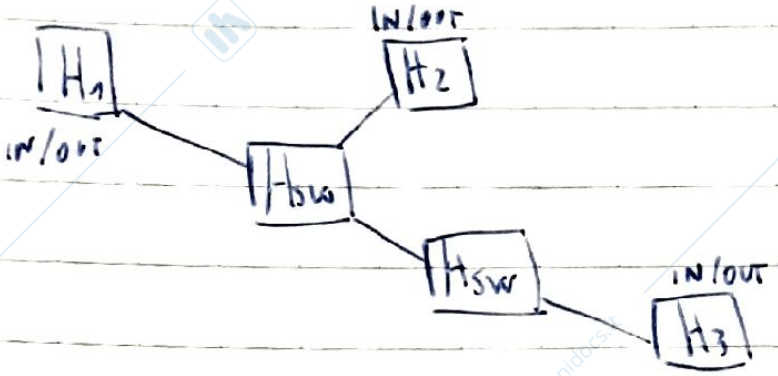
↓

perché è FULL DUPLEX

↓

2 direzioni cont. le può fare

Sincronizzazione



Vogliamo fare in modo che i timer di tutti gli host si sincronizzino

↓

Il modo più semplice è mandare un pacchetto di sincronizzazione su tutta la rete

per evitare eventuali ritardi che si sono creati

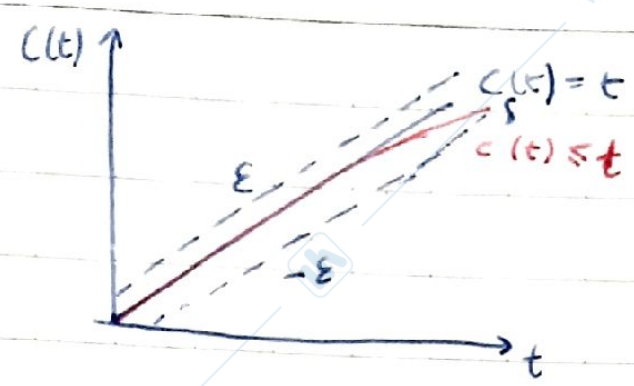
↓

La frequenza di sincronizzazione dipende dal problema

↓

Modello matematico del clock

- t = tempo del server
- C(t) = misura del tempo restituita da un clock



- $C_s(t)$ = clock standard

→ Correttezza del clock: $|C(t) - C_s(t)| \leq \varepsilon$

se vale ~~ft~~ è corretto

→ Deriva del clock: $\left|\frac{dc(t)}{dt} - 1\right| \leq \rho$

Se $c(t) = t$

$\frac{dc(t)}{dt} = 1$

es. tipi valore che un clock hardware deriva $\rho = 10\ \mu s/s$

SINCRONIZZAZIONE

ESTERNA

Tutti gli host sono sincronizzati rispetto a un punto di riferimento esterno

INTERNA

Tutti gli host sono sincronizzati tra di loro

3

No. SISTEMI INF.

Date 14 · 11 · 19

Sincronizzazione ottima

Supponiamo ρ $\left| \left| \frac{d\,c(t)}{dt} - 1 \right| \leq \rho \right.$

$\epsilon(t)$ ↑ ERRORE AL TEMPO t $\longrightarrow$ $t \in [c(t) - \epsilon(t), c(t) + \epsilon(t)]$

Quando $c(t) = r$ ⟹ sincronizzazione (aggiornamento)

$t \in [r - \varepsilon, r + \varepsilon]$

ε = errore di sincronizzazione

$\epsilon(t) = \varepsilon$

⇒ $\epsilon(t) = \varepsilon + \rho\,(c(t) - r) \leq$ BOUND DI CORRETTEZZA (BOUND)

Noi vogliamo capire quando aggiornare affinché valga la relazione precedente

$$c(t) - r \leq \frac{\text{BOUND} - \varepsilon}{\rho}$$

$c(t) - r$ → PERIODO DI SINCRONIZZAZIONE

es. $\varepsilon = 0{,}5$
$\rho = 10^{-5}$ ⟹ PERIODO DI SINCRONIZZAZIONE = 14 ORE
BOUND = 1 s

4

No. SISTEMI INF.

Date 14 · 11 · 19

INGEGNERIA DEL SOFTWARE

(E Object Oriented Programming)

CICLO DI VITA DEL SOFTWARE

① MODELLO A CASCATA:

Analisi e **Specifica** dei requisiti → qual è l'input e che proprietà deve avere l'output

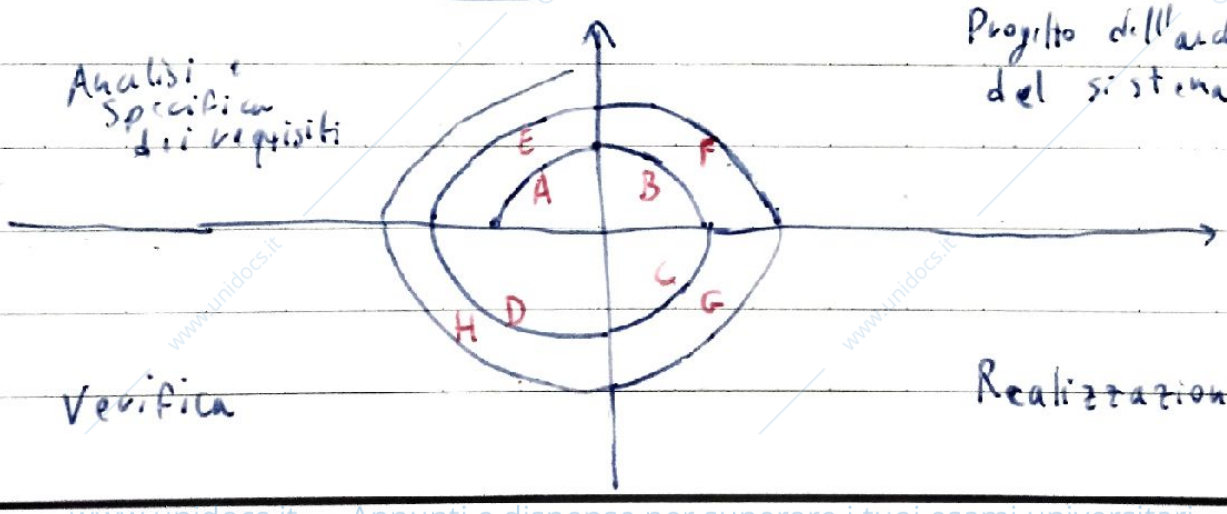
→ Progettazione del sistema e **specifica** → Specifico la struttura del sistema

→ Codifica e test dei moduli

→ Integrazione dei moduli e test

→ Consegna e manutenzione

② MODELLO A SPIRALE:



A = analisi iniziale

B = Progetto iniziale

C = Primo prototipo

D = Verifico primo prototipo

E = Nuove specifiche

F = progettazione intero sistema

G = Prima versione

H = Verifica prima versione (versione α)

Problema della specifica

Specifica funzionale

↓ esprime il funzionamento del sistema

↓

STATE CHARTS (linguaggio)

Specifica non funzionale

↓ esprime la struttura del sistema

↓

CLASS DIAGRAMM (permette di scrivere la struttura del codice)

STATE CHARTS

Include GRAFICA + LOGICA BOOLEANA

↓

LINGUAGGIO SEMIFORMALE DI SPECIFICA FUNZIONALE

↓

Usato per: - DESCRIVERE IL FUNZIONAMENTO DI UN SISTEMA
- VERIFICARE LA CORRETTEZZA DELLA SPECIFICA

→ Logica booleana

$x_i \in \{0,1\}$ → VARIABILI

$\neg, \wedge, \vee$ → OPERATORI LOGICI

$\neg$ (not), $\wedge$ (and), $\vee$ (or)

Possiamo costruire delle formule logiche

es. x_1, x_2, x_3 → $x_1 \wedge x_2$ $\quad x_1 \vee (\neg x_2 \vee x_3)$

$x_1 \vee x_2$

$\neg x_3$

Se scrivo che la formula logica $\phi = x_1$ è vera se x_1 è vera e poi in base agli operatori dipende la verità della formula.

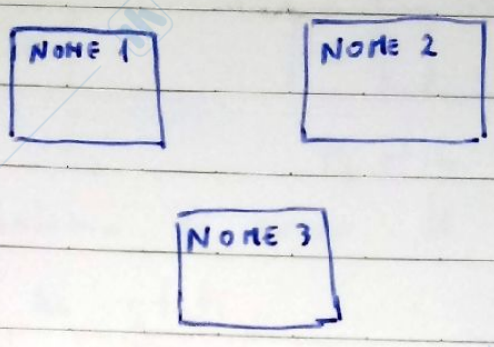
2

No. SISTEMI INF.

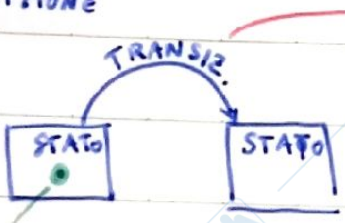
Date 15 · 11 · 19

Linguaggio

① STATI



② TRANSIZIONE



ogni transizione è caratterizzata da una condizione booleana

token (gettone)

si muove solo nelle transizioni

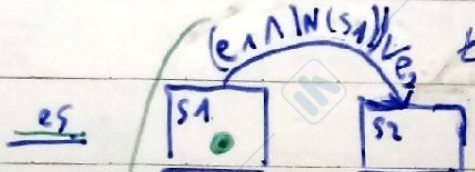
Condizione di transizione:

④ EVENTI → e_1, e_2, e_3 → l'evento diventa 1 quando accad. qualcosa

③ CONDIZIONE SUGLI STATI → IN (STATO)

Operatore

→ se il sistema si trova in quello stato allora è 1



es.

è già in S1

→ equivalente a: $e_1 \vee e_2$

Se fosse stato $(e_1 \wedge IN(S_2)) \vee e_2$ non si sarebbe mai verificata la transizione

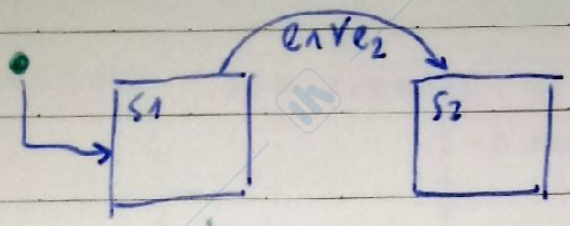
3

Mo Tu We Th Fr Sa Su

No. SISTEMI INF.

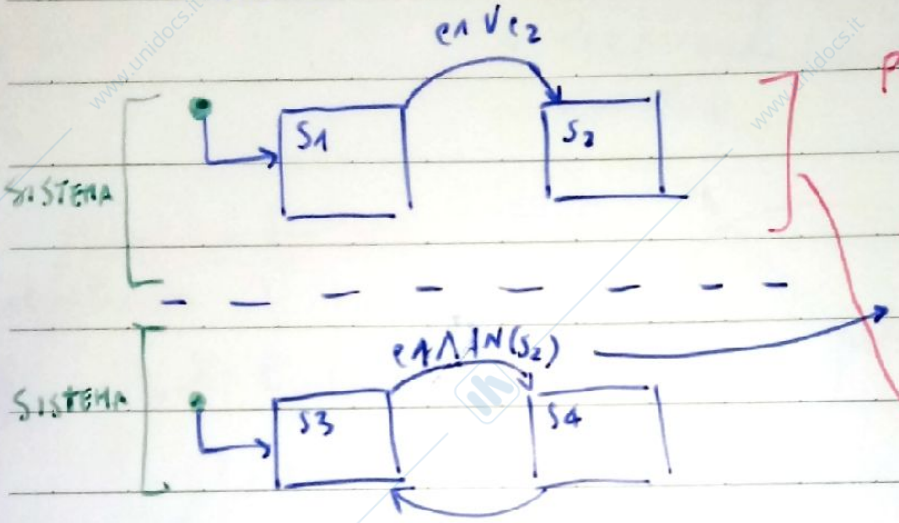
Date 15 · 11 · 19

(3) STATO DI DEFAULT



↳ Stato di default

(4) CONCORRENZA



Possiamo utilizzare l'operatore IN( ) per indicare la posizione del token nel sottosistema

→ La concorrenza è strettamente necessaria oppure il linguaggio è ugualmente espressivo senza?

↓

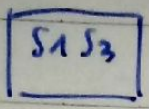
Si può fare senza

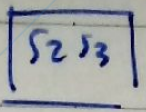
↓

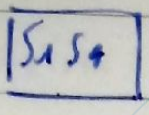
Tolgo la concorrenza dicendo che il mio sistema si deve trovare in S_1 e S_3 contemporaneamente

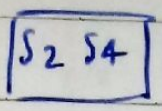
↓

facendo il <u>prodotto cartesiano</u> degli Stati









Oss. Supponiamo di avere k sottosistemi ognuno con m stati

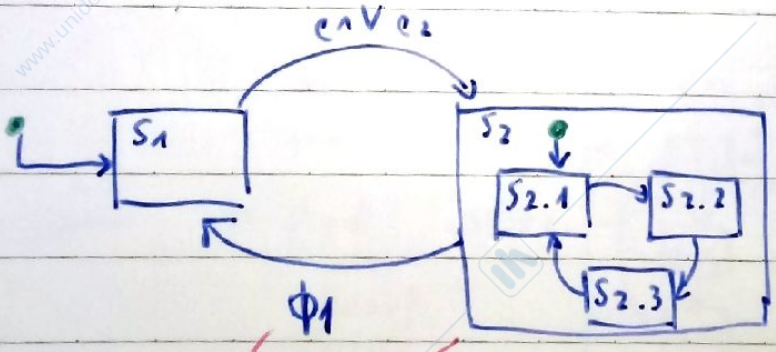
↳ • con concorrenza $m \cdot k$ stati

• senza concorrenza (prodotto cartesiano) m^k stati

↓

la concorrenza non lo rende più espressivo ma rende il linguaggio più **EFFICIENTE**

(5) GERARCHIA



può accadere che si verifichino contemporaneamente Φ_1 e una condizione interna a S_2

↳ per questioni di **gerarchia** domina Φ_1 (la condizione **esterna**)

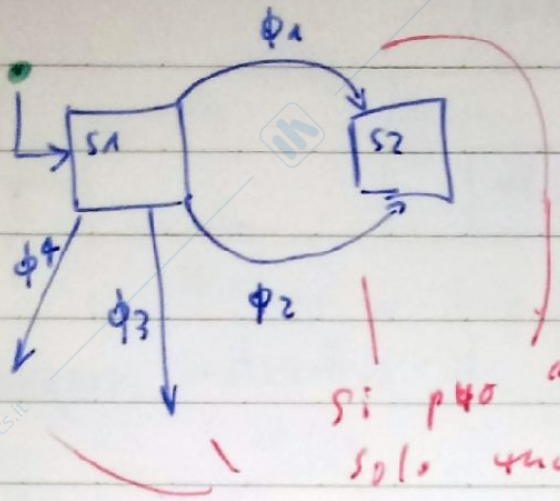
5

Mo Tu We Th Fr Sa Su

No. SISTEMI INF.

Date 15 · 11 · 19

Se abbiamo

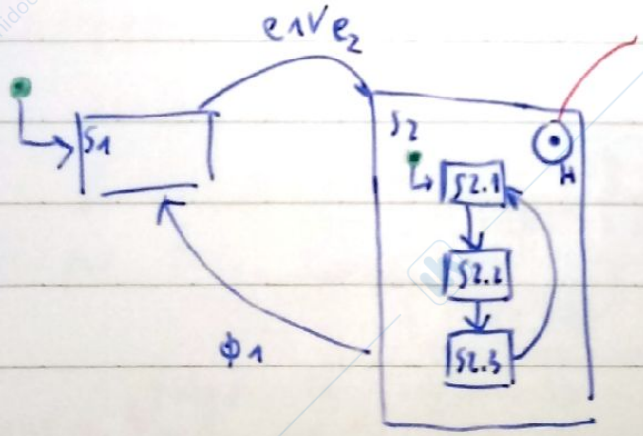


si può "accendere" solo una transizione alla volta

$\phi_1, \phi_2, \phi_3, \phi_4$ devono essere mutuamente esclusive

quando la verifica vediamo se c'è un errore

(6) STATO DI STORIA



significa che è uno stato di storia

↓

ossia quando entriamo di nuovo in quello stato entriamo nel sottostato di uscita precedente

es. il token si trovava in S2.2. quando si è verificato ϕ_1 (uscita)

Se entriamo in S2 non entriamo in S2.1 (stato Default) ma S2.2

Provare a costruire un sottosistema concorrente in modo da evitare di mettere lo stato di storia

(7) Transizioni entranti e/o uscenti su sottostati

es. Abbiamo: - 1 ascensore

- si muove su i piani 0,1,2,3, solo se FERMO
- caratterizzato da un pulsante SET → fa selezionare il piano incrementando ad ogni tocco di SET il piano
  e un pulsante START → dopo aver selezionato il piano fa partire l'ascensore
- dotato di porte (parte solo con porte chiuse e dopo aver schiacciato START)
- può essere → FERMO (posso cambiare piano solo se FERMO)
  → IN MOVIMENTO

EVENTI: - A PRIPORTE
- CHIUDI-PORTE
- START
- SET

→ condizionano l'azione dell'umano

7

Mo Tu We Th Fr Sa Su

No. SISTEMI INF.

Date 15 · 11 · 19