Progettazione del Software

Giuseppe De Giacomo & Massimo Mecella
Dipartimento di Informatica e Sistemistica
SAPIENZA Università di Roma

Diagramma degli stati e delle transizioni
Analisi

Il diagramma degli stati e delle transizioni

Il diagramma degli stati e delle transizioni viene definito per **una classe**, ed intende descrivere **l'evoluzione di un generico oggetto** di quella classe.

Il diagramma rappresenta le sequenze di stati, le risposte e le azioni, che un oggetto attraversa durante la sua vita in risposta agli stimoli ricevuti.

Uno **stato** rappresenta una situazione in cui un oggetto ha un insieme di proprietà considerate stabili

Una **transizione** modella un cambiamento di stato ed è denotata da:

Evento [Condizione] / Azione

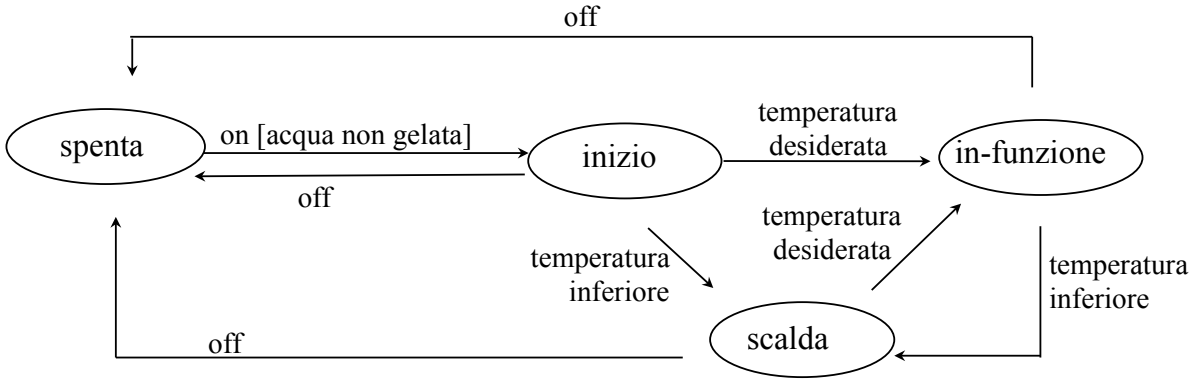
Il diagramma degli stati e delle transizioni

Il significato di una transizione del tipo di quella qui mostrata è:

- se l'oggetto
  - si trova nello **stato** S_1 e
  - riceve l'***evento*** *E* e
  - la ***condizione*** *C* è verificata
- allora
  - attiva l'esecuzione dell'***azione*** *A* e
  - passa nello **stato** S_2.

Esempio di diagramma degli stati e delle transizioni per la classe Caldaia

Descriviamo il diagramma degli stati e delle transizioni relativa ad una **classe** "Caldaia". In questo diagramma ogni transizione è caratterizzata solamente da eventi e condizioni (i cambiamenti di stato non hanno bisogno di azioni perché sono automatici)

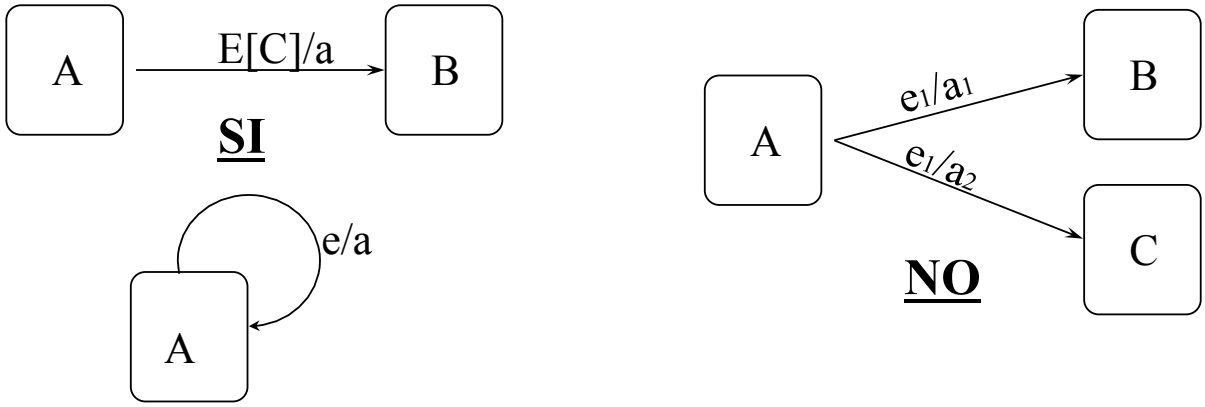


Stato

- Lo stato di un oggetto racchiude le proprietà (di solito statiche) dell'oggetto, più i valori correnti (di solito dinamici) di tali proprietà
- Una freccia non etichettata che parte dal "vuoto" ed entra in uno stato indica che lo stato è **iniziale**
- Una freccia non etichettata che esce da uno stato e finisce nel "vuoto" indica che lo stato è **finale**
- Stato iniziale e finale possono anche essere denotati da appositi simboli

Transizione

- Ogni transizione connette due stati
- Il diagramma corrisponde ad un automa **deterministico** (transizioni dall stesso stato hanno eventi diversi), in cui un evento è un input, mentre un'azione è un output
- La condizione è detta anche "guardia" (guard)
- L'evento è (quasi) sempre presente (condizione e azione sono opzionali)

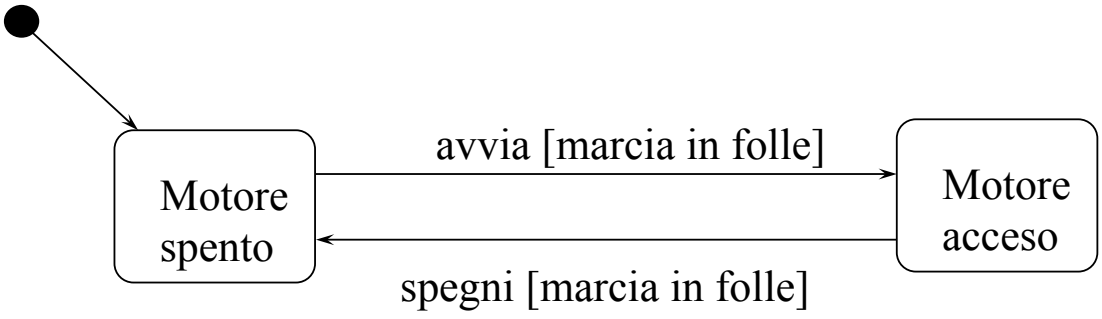


Esempio di diagramma degli stati e delle transizioni per la classe Motore

L'analisi dei requisiti ha evidenziato l'esistenza, nel diagramma delle classi, di una classe "Motore". Tracciare il diagramma degli stati e delle transizioni a partire da questi requisiti.

Un motore di automobile può essere spento o acceso, ma può essere avviato o spento solo se la marcia è in folle

Esempio di diagramma degli stati e delle transizioni per la classe Motore (soluzione)



Esercizio 1

Supponiamo che nel diagramma delle classi abbiamo rappresentato la classe “Menu a tendina”. Tracciare il diagramma degli stati e delle transizioni per tale classe, tenendo conto delle seguenti specifiche.

Un menu a tendina può essere visibile oppure no. Viene reso visibile a seguito della pressione del tasto destro del mouse, e viene reso invisibile quando tale tasto viene sollevato. Se si muove il cursore quando il menu è visibile, si evidenzia il corrispondente elemento del menu.

Esercizio 1: soluzione

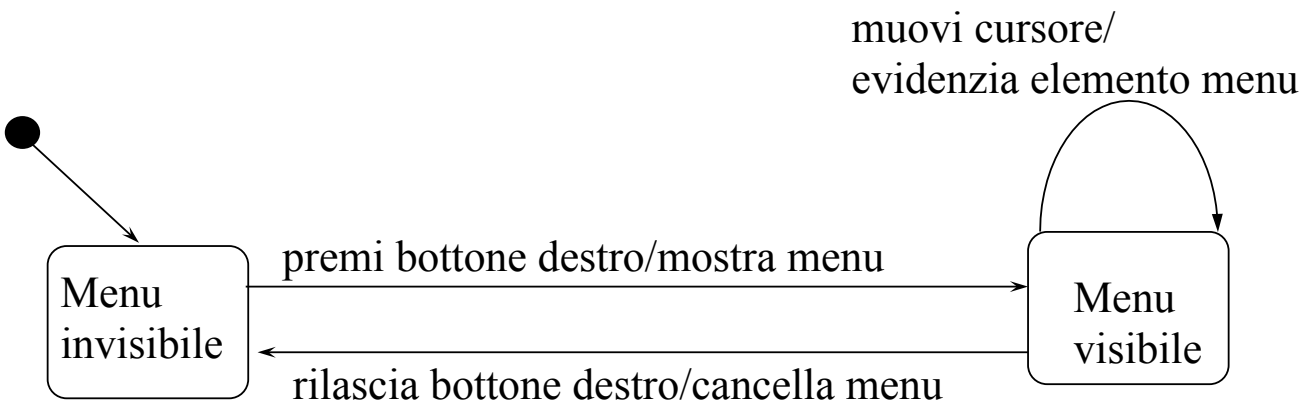
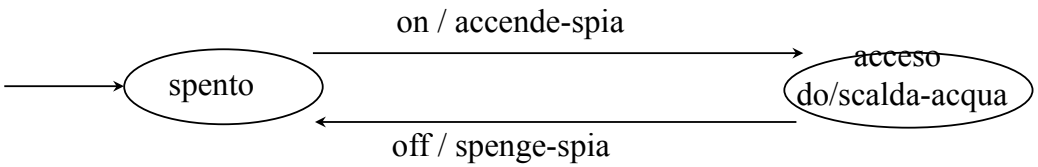


Diagramma degli stati e delle transizioni

Alcune volte vogliamo rappresentare dei processi che l'oggetto esegue senza cambiare stato. Questi processi si chiamano **attività**, e si mostrano negli stati con la notazione:

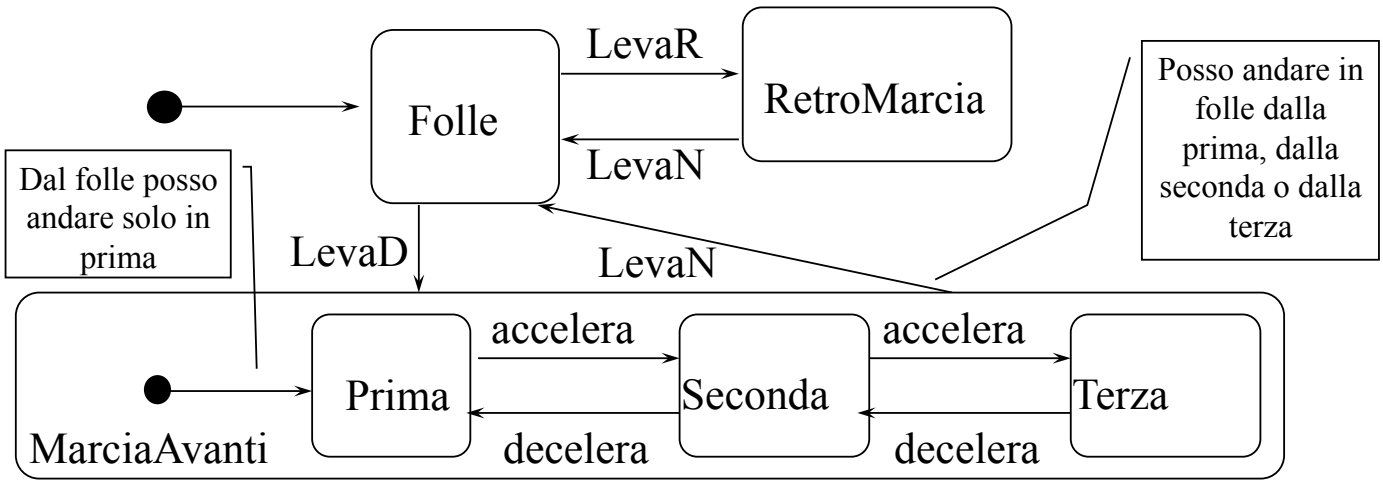
do / attività

Esempio (scaldabagno):



Stato composto

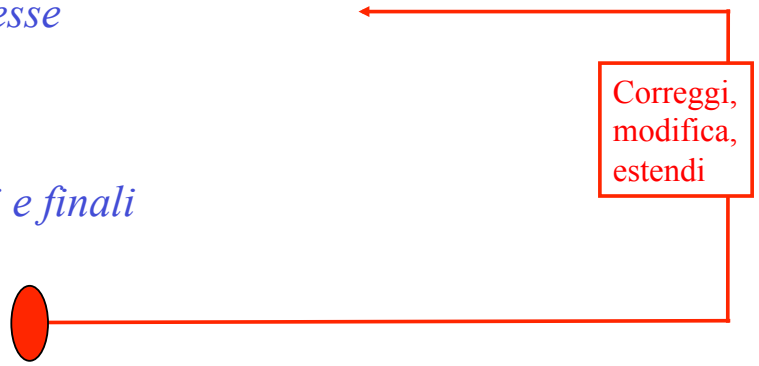
- Uno **stato composto** (o **macro-stato**) è uno stato che ha un nome, e che contiene a sua volta un diagramma
- Esiste uno stato iniziale del macro-stato
- I **sottostati** ereditano le transizioni in uscita del macro-stato



Aspetti metodologici nella costruzione del diagramma degli stati e delle transizioni

Un metodo comunemente usato per costruire il diagramma degli stati e delle transizioni prevede i seguenti passi

- *Individua gli stati di interesse*
- *Individua le transizioni*
- *Individua le attività*
- *Determina gli stati iniziali e finali*
- *Controllo di qualità*



Controllo di qualità del diagramma degli stati e delle transizioni

- Sono stati colti tutti gli aspetti insiti nei requisiti?
- Ci sono ridondanze nel diagramma?
- Ogni stato può essere caratterizzato da proprietà dell'oggetto?
- Ogni azione e ogni attività può corrispondere ad una operazione della classe?
- Ogni evento e ogni condizione può corrispondere ad un evento o condizione verificabile per l'oggetto?

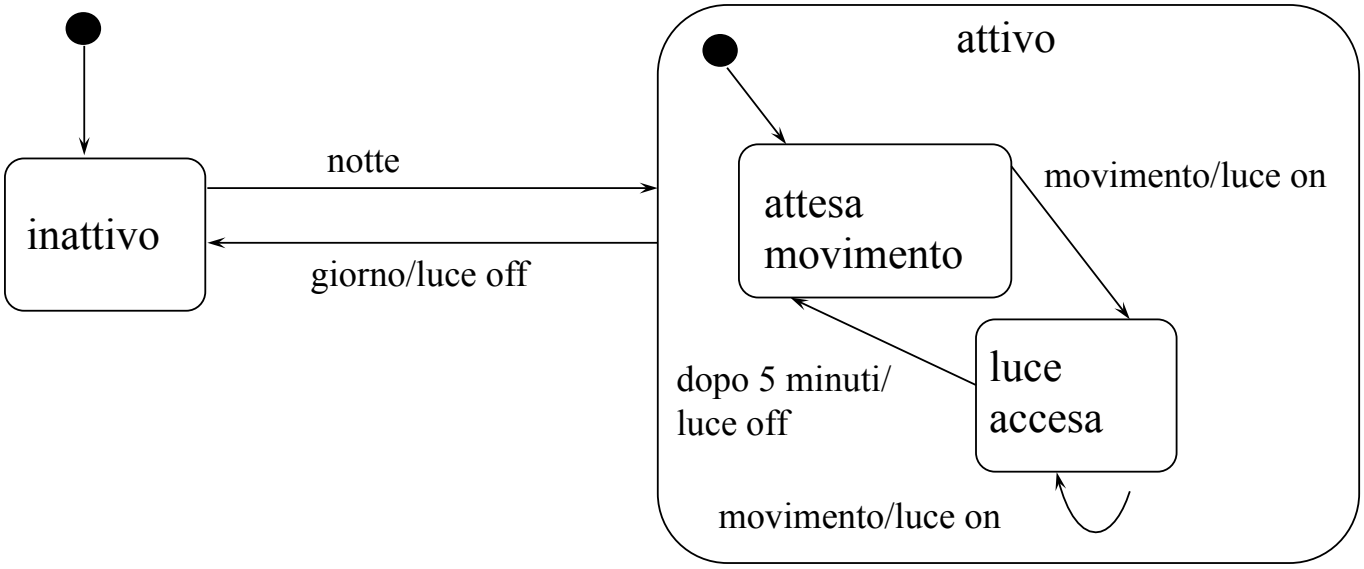
Esercizi su diagramma degli stati e delle transizioni

Esercizio 2

Supponiamo che nel diagramma delle classi abbiamo rappresentato la classe "InterruttoreAutomatico". Tracciare il diagramma degli stati e delle transizioni per tale classe, tenendo conto delle seguenti specifiche.

Un interruttore automatico collegato ad una cellula fotoelettrica e ad un sensore di movimento comanda una luce di un sottoscala che deve essere accesa solo di notte ed in presenza di movimento. Un'assenza di movimenti per cinque minuti consecutivi causerà lo spengimento della luce.

Esercizio 2: soluzione

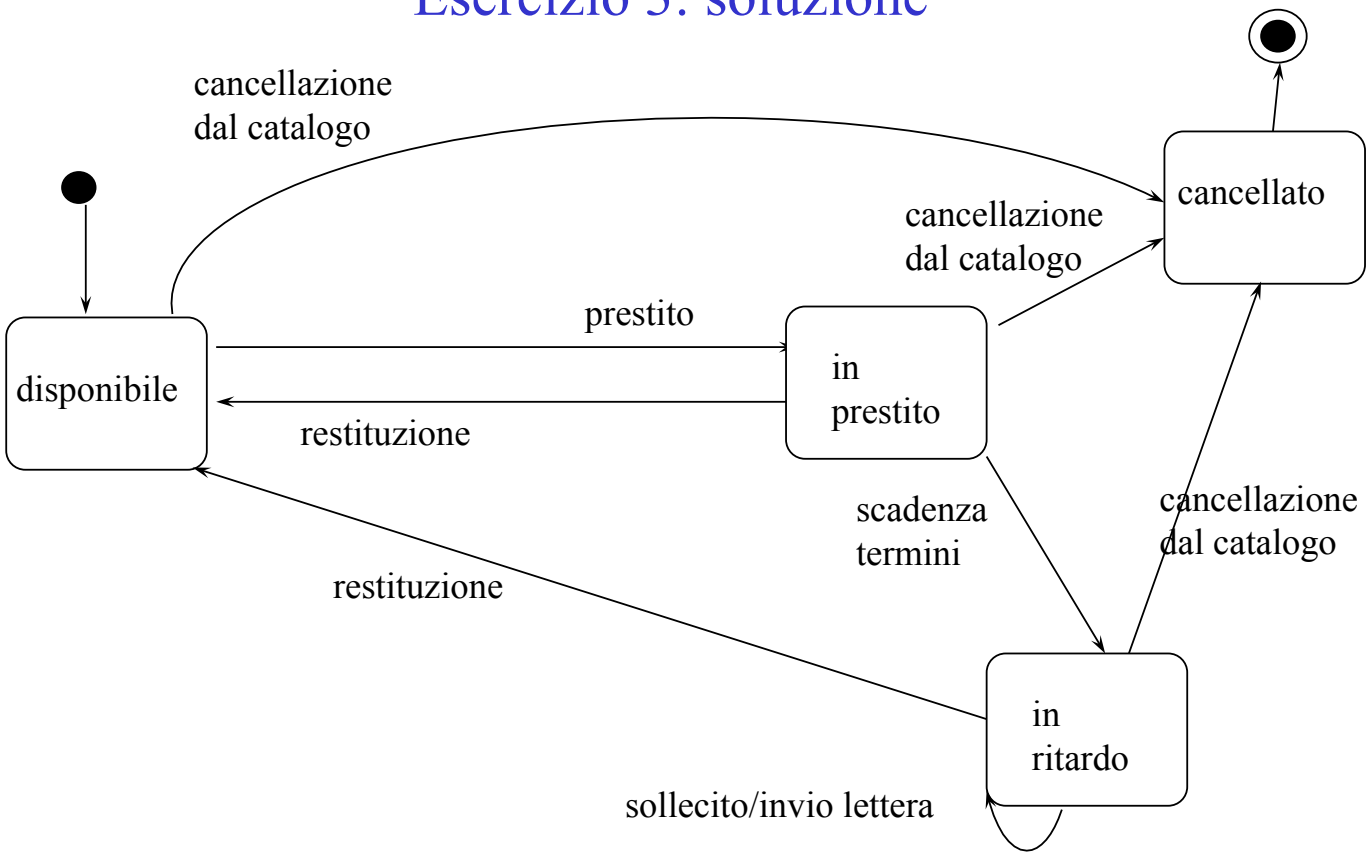


Esercizio 3

Supponiamo che nel diagramma delle classi abbiamo rappresentato la classe "Libro". Tracciare il diagramma degli stati e delle transizioni per tale classe, tenendo conto delle seguenti specifiche.

Una biblioteca può acquisire libri, che possono essere dati in prestito e successivamente restituiti. Quando scadono i termini del prestito, la restituzione è in ritardo, ed in tal caso la biblioteca può inviare (anche più volte) una lettera di sollecito. In ogni momento, un libro può essere cancellato dal catalogo.

Esercizio 3: soluzione

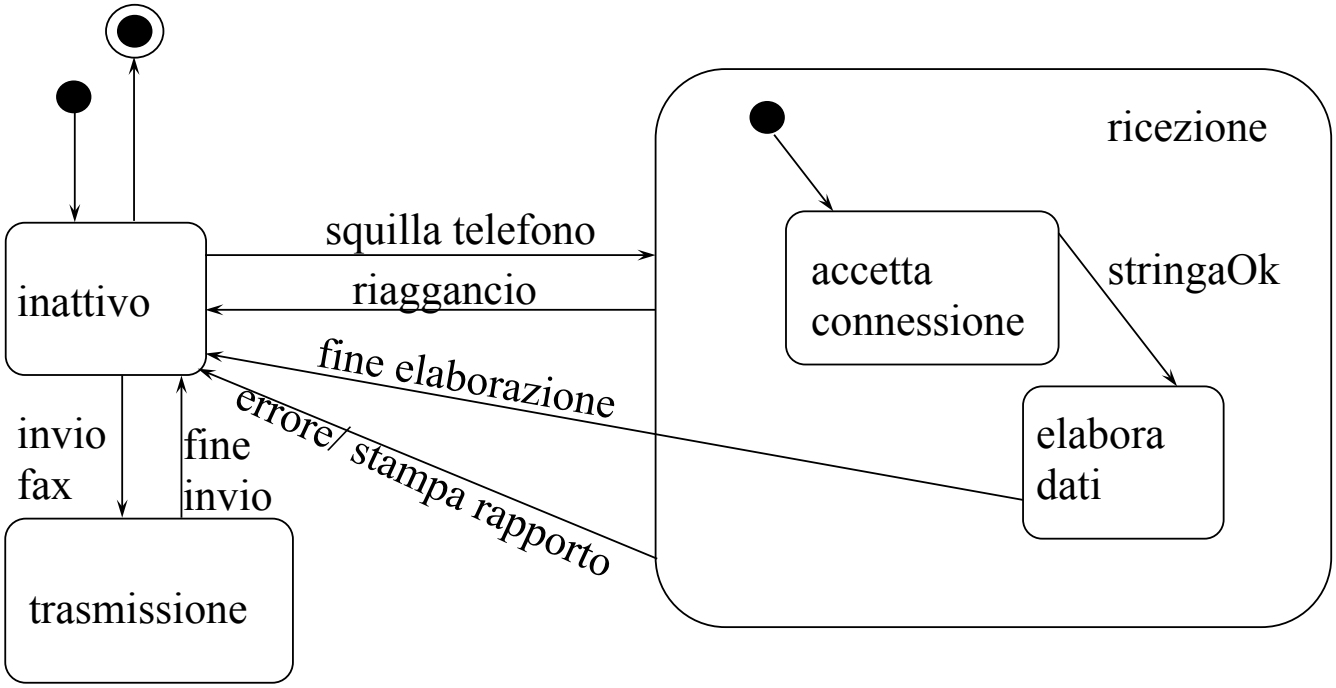


Esercizio 3

Supponiamo che nel diagramma delle classi abbiamo rappresentato la classe "Fax". Tracciare il diagramma degli stati e delle transizioni per tale classe, tenendo conto delle seguenti specifiche.

Un fax può essere inattivo, ricevente o trasmittente. Se il fax è inattivo, con il comando di invio fax si porta il dispositivo nello stato trasmittente, e con il comando di fine invio si riporta nello stato inattivo. Quando il fax è inattivo e si verifica una chiamata (segnalata da uno squillo del telefono), va in stato ricevente, e quindi accetta la connessione. Se la stringa iniziale è corretta, il fax elabora i dati, e infine ritorna inattivo. In ogni momento della trasmissione, il chiamante può riagganciare, facendo ritornare il fax nello stato inattivo. In ogni momento della trasmissione, se si verifica un errore in ricezione, il fax ritorna inattivo e stampa un rapporto di errore.

Esercizio 3: soluzione



Esercizio 4

Supponiamo che nel diagramma delle classi abbiamo rappresentato la classe "DispositivoPortatile". Tracciare il diagramma degli stati e delle transizioni per tale classe, tenendo conto delle seguenti specifiche.

Un dispositivo portatile per la comunicazione di emergenze può essere acceso o spento con lo stesso tasto "OnOff". Gli altri due tasti del dispositivo sono: "Emergenza" e "Invio". Per comunicare un'emergenza bisogna, nell'ordine, premere il tasto "Emergenza" e poi "Invio". Per disattivare la tastiera del dispositivo bisogna premere il tasto "Invio". Per riattivare la tastiera quando è stata precedentemente disattivata bisogna premere il tasto "Invio". In ogni momento si può spengere il dispositivo. In ogni circostanza, la pressione di un tasto non contemplato nella descrizione precedente non produce alcun effetto.

Esercizio 4: commento (1)

Per comodità, numeriamo i requisiti.

1. Un dispositivo portatile per la comunicazione di emergenze può essere acceso o spento con lo stesso tasto "OnOff".
2. Gli altri due tasti del dispositivo sono: "Emergenza" e "Invio".
3. Per comunicare un'emergenza bisogna, nell'ordine, premere il tasto "Emergenza" e poi "Invio".
4. Per disattivare la tastiera del dispositivo bisogna premere il tasto "Invio".
5. Per riattivare la tastiera quando è stata precedentemente disattivata, bisogna premere il tasto "Invio".
6. In ogni momento si può spengere il dispositivo.
7. In ogni circostanza, la pressione di un tasto non contemplato nella descrizione precedente non produce alcun effetto.

Esercizio 4: commento (2)

- I requisiti 1 e 2 affermano che esistono tre simboli nell'alfabeto di input ("OnOff", "Emergenza" e "Invio").
- Il requisito 1 implica l'esistenza di (almeno) due stati: "acceso" e "spento".
- Il requisito 6 suggerisce che è conveniente modellare lo stato "acceso" come macro-stato.

Esercizio 4: commento (3)

- I requisiti 3 e 4 implicano l'esistenza di altri tre stati, tutti interni al macro-stato "acceso":
  - "tastiera attivata": lo stato iniziale del macro-stato
  - "tastiera disattivata": lo stato a cui si giunge con la pressione del tasto "Invio"
  - "pronto": lo stato a cui si giunge dopo la pressione del tasto "Emergenza"
- Le transizioni fra stati sono dettate dai requisiti 3, 4, 5 e 7.
- Le transizioni sono tutte prive di condizioni.
- Si ha l'azione di "comunica emergenza" in corrispondenza della transizione dallo stato "pronto" a quello "tastiera attivata"

Esercizio 4: soluzione

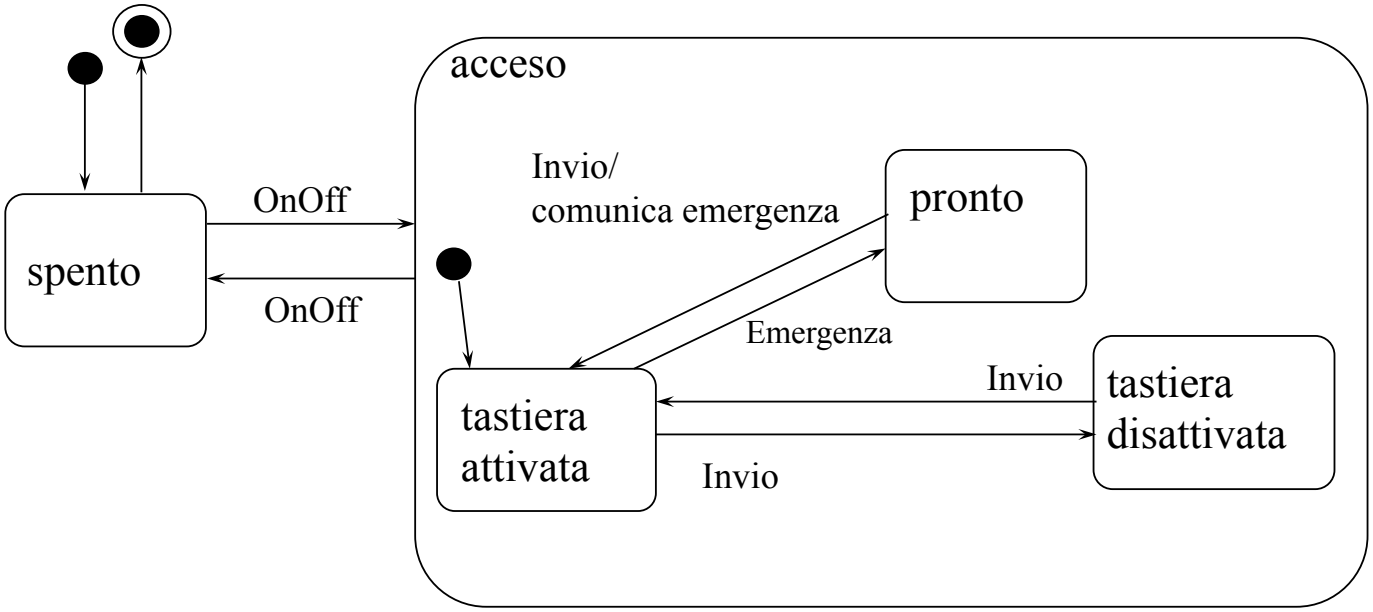


Diagramma degli stati e delle transizioni di oggetti reattivi

Principi generali

- Assumiamo di avere diversi **oggetti reattivi**, cioè con associato un diagramma stati-transizioni.
- Assumiamo che l'interazione sia basata su scambio esplicito di **eventi**
- Assumiamo che gli eventi abbiano un **mittente** ed un **destinatario**
  In particolare ammettiamo
  - **Messaggi punto-punto**: un oggetto manda un messaggio ad un altro oggetto
  - **Messaggi in broadcasting**: un oggetto manda un messaggio a tutti gli altri oggetti (vedremo esempi successivamente).
- Inoltre gli eventi possono avere **parametri** con specifico contenuto informativo (il cosiddetto *payload* del messaggio)
- Una **azione** può a sua volta lanciare un **evento** (uno solo per semplicità) per un'altro oggetto o in broadcasting.

Osservazioni

- Nel diagramma degli stati e transizioni per semplicità identifichiamo **l'azione** stessa con **l'evento lanciato**.

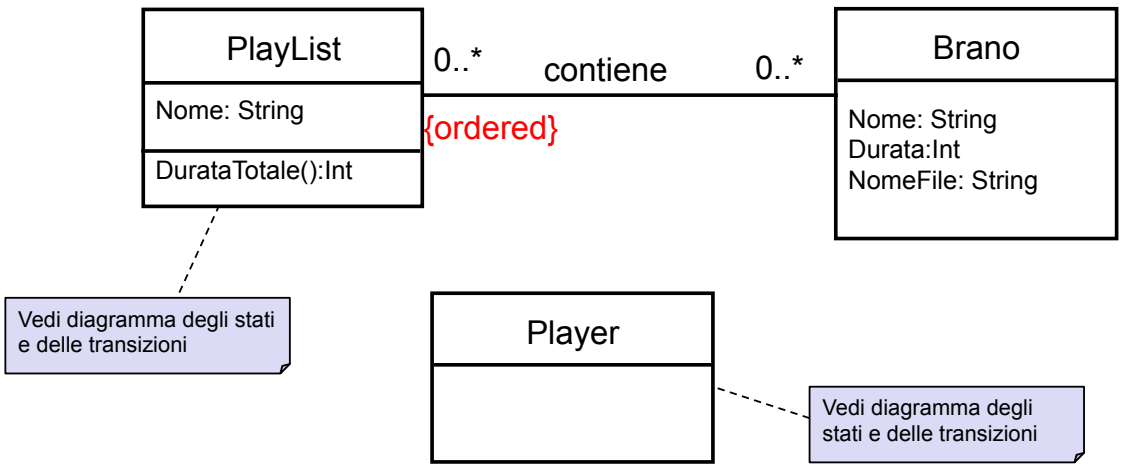
- Diamo una specifica dettagliata di ciò che avviene ad ogni transizione:
  - Quali **eventi** sono **recepiti** e **lanciati** (e a chi)
  - Come cambiano eventuali **variabili di stato ausiliarie** associate allo stato dell'oggetto (vedi sotto)
  - Come **cambia l'istanziazione del diagramma delle classi**

- Le **variabili di stato ausiliarie** non sono di interesse per il cliente servono solo alla corretta realizzazione delle azioni associate alle varie transizioni. Quindi non vanno confuse con con gli attributi dell'oggetto stesso.

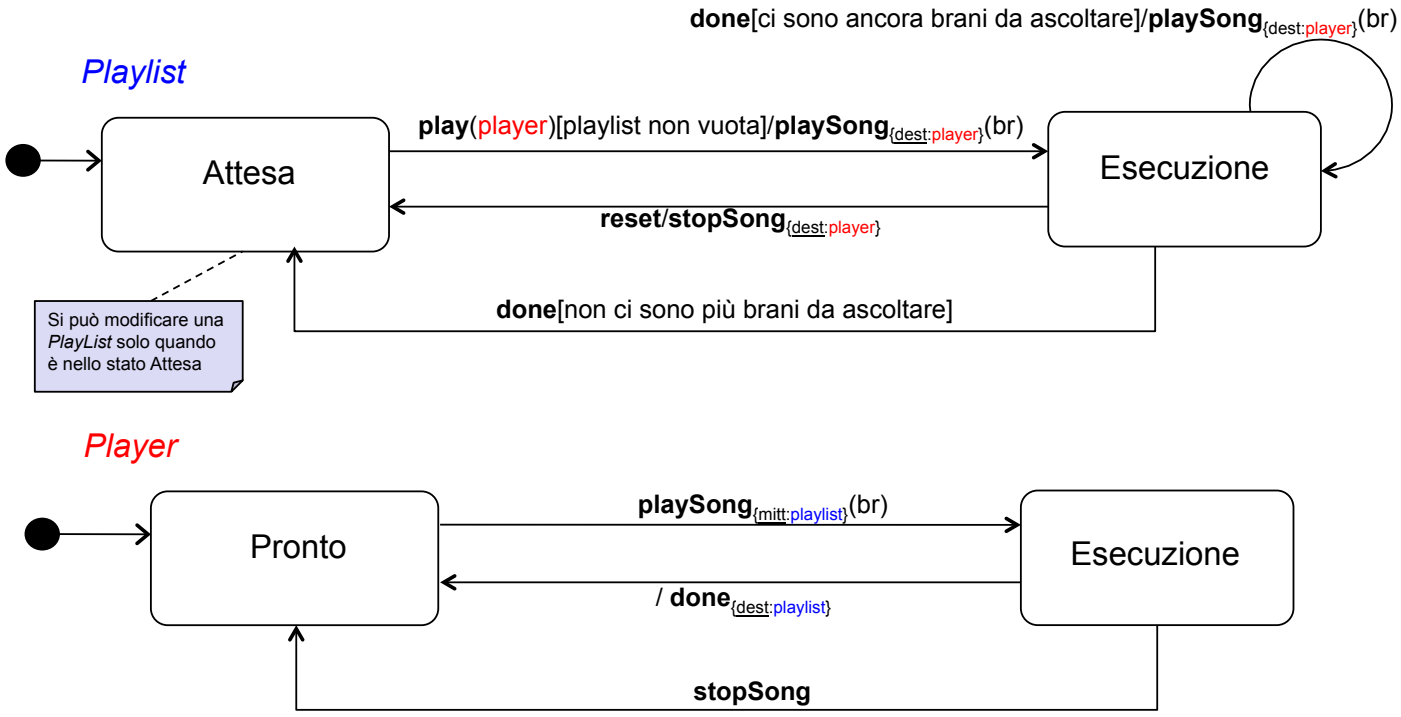
- Il **diagramma degli stati e transizioni** è sempre corredato da detta **specifica** che ne chiarisce in dettaglio la semantica.

Diagramma delle classi

- Consideriamo il seguente diagramma delle classi:
  - PlayList e Brano li abbiamo già incontrati in precedenza
  - Player è una classe che non contiene alcun dato (ma a cui è associato un diagramma stati e transizioni



Diagrammi degli stati e delle transizioni



Specifica degli stati di PlayList

InizioSpecificaStatiClasse PlayList

Stato: {Attesa, Esecuzione}
Variabili di stato ausiliarie:
player: Player
prossimobrano: intero

Stato iniziale:
statoCorrente = Attesa
player = --
prossimobrano = --

"***statoCorrente***" denota lo stato attuale dell'oggetto.

Viene aggiornato automaticamente facendo transizioni

"--" sta per non definito

FineSpecifica

Specifica delle transizioni di PlayList

InizioSpecificaTransizioniClasse PlayList

Transizione: Attesa→ Esecuzione
play(player)[playlist non vuota]/playSong$_{\{dest:player\}}$(br)

Evento: play(player:Player)

Condizione: *this*.contiene non vuoto

Azione:
pre: nessuna
post: *nuovoevento* = playSong{*mitt* = *this*, *dest* = player}(br: Brano)
and
this.player = player and
this.prossimobrano = 0 and
<this,br> in contiene and
posizione(contiene(*this*,br)) = *this*.prossimobrano

"***evento***" denota l'evento ricevuto, "***mitt***" e "***dest***", denotano il mittente e il destinatario dell'evento.

Ovviamente affinchè l'evento sia considerato deve essere ***evento.dest=this***

"***nuovoevento***" denota l'evento da mandare con l'azione

…
FineSpecifica

Specifica delle transizioni di PlayList

InizioSpecificaTransizioniClasse PlayList

…

Transizione: Esecuzione→ Esecuzione
done[ci sono ancora brani da ascoltare]/playSong$_{\{dest:player\}}$(br)

Evento: done

Condizione: *this*.prossimobrano < |{b | <*this*,b> in contiene}|

Azione:
pre: nessuna (implicitamente abbiamo sempre *evento.mitt=this*.player)
post: *nuovoevento* =
playSong{*mitt* = *this*, *dest* = *this*.player}(br: Brano) and
this.player = *Pre*(*this*.player) and
this.prossimobrano = *Pre*(*this*.prossimobrano)+1 and
<*this*,br> in contiene and
posizione(contiene(*this*,br)) = *this*.prossimobrano

…
FineSpecifica

Specifica delle transizioni di PlayList

InizioSpecificaTransizioniClasse PlayList

…

Transizione: Esecuzione→ Attesa
done[non ci sono brani da ascoltare]

Evento: done

Condizione: *this*.prossimobrano >= |{b | <*this*,b> in contiene}|

Azione:
pre: nessuna
post: *this*.prossimobrano = -- and
this.player = --

…
FineSpecifica

Specifica delle transizioni di PlayList

InizioSpecificaTransizioniClasse PlayList

…

Transizione: Esecuzione→ Attesa
reset/stopSong{dest:player}

Evento: reset

Condizione: nessuna

Azione:
pre: nessuna
post: *nuovoevento* = stopSong{*mitt* = *this*, *dest* = *Pre*(*this*.player)}
this.prossimobrano = -- and
this.player = --

FineSpecifica

Specifica degli stati di Player

InizioSpecificaStatiClasse Player

Stato: {Pronto, Esecuzione}
Variabili di stato ausiliarie:
playlist: PlayList
brano: Brano

Stato iniziale:
statoCorrente = Pronto
playlist = --
brano = --

FineSpecifica

Specifica delle transizioni di Player

InizioSpecificaTransizioniClasse Player

Transizione: Pronto→ Esecuzione
playSong$_{\{mitt:playlist\}}$(br)

Evento: playSong(br:Brano)

Condizione: nessuna

Azione:
pre: nessuna
post: *this*.playlist = *evento.mitt* and
this.brano = br

…
FineSpecifica

Specifica delle transizioni di Player

InizioSpecificaTransizioniClasse Player

…

Transizione: Esecuzione→ Pronto
/ done{dest:playlist}

Evento: evento interno generato da *this* stesso

Condizione: nessuna

Azione:
pre: nessuna
post: *nuovoevento* = done{*mitt* = *this*, *dest* = *Pre*(*this*.playlist)}
this.playlist= -- and
this.brano = --

…
FineSpecifica

Specifica delle transizioni di Player

InizioSpecificaTransizioniClasse Player

…

Transizione: Esecuzione→ Pronto
stopSong

Evento: stopSong

Condizione: nessuna

Azione:
pre: *evento.mitt* = *Pre*(*this*.playlist)}
post: *this*.playlist= -- and
this.brano = --
FineSpecifica