

Diagrammi di Interazione

Ingegneria del Software

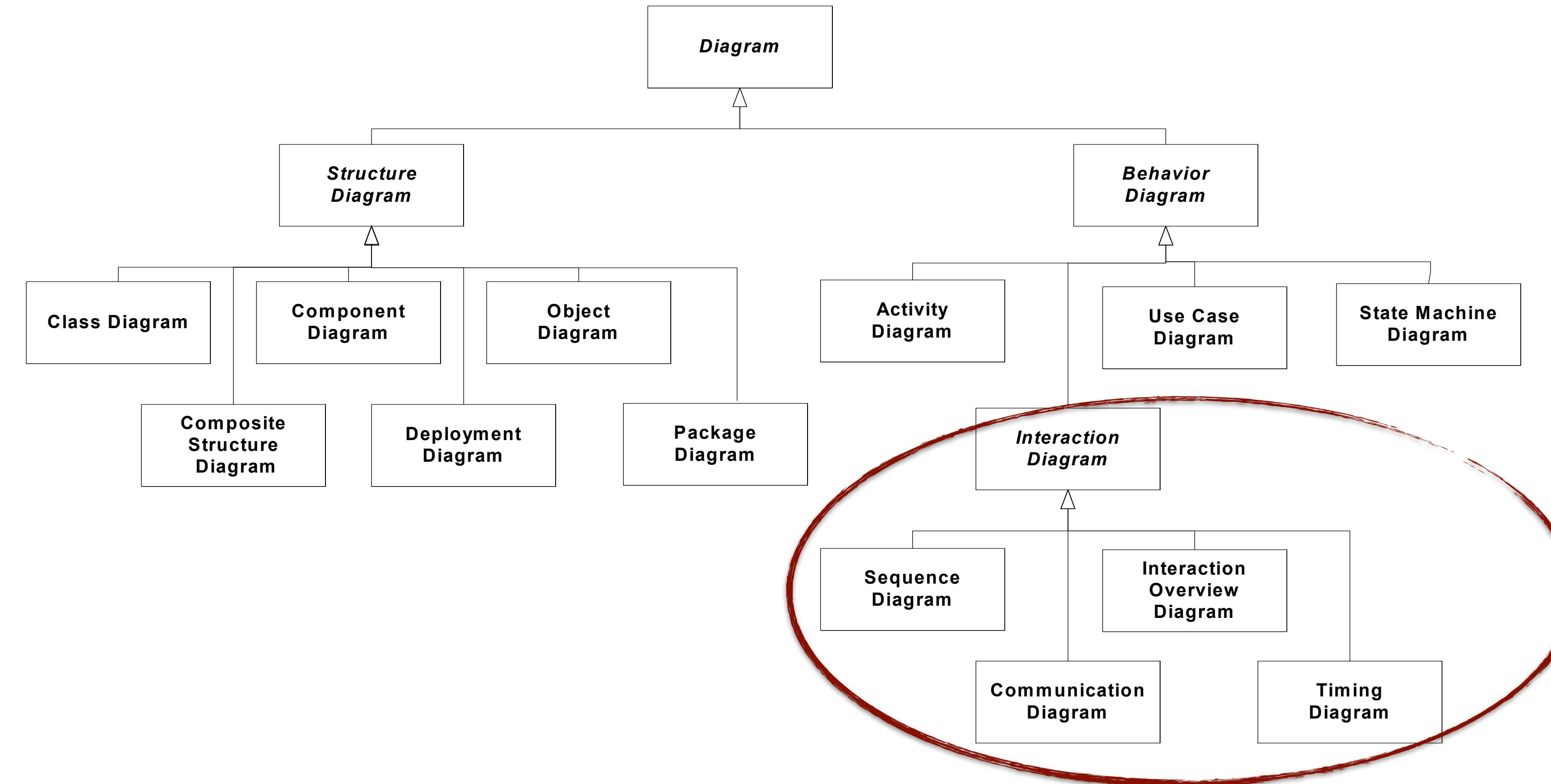
Tassonomia dei diagrammi UML

- UML 2 definisce 13 diagrammi divisi in due categorie
 - structure diagrams: come è fatto il sistema
 - behaviour diagrams: come funziona (o si comporta) il sistema

Structure	Behavior
Class diagram	Use Case diagram
Object diagram	Activity diagram
Package diagram*	State Machine diagram
Composite Structure diagram*	Sequence diagram
Component diagram	Communication diagram
Deployment diagram	Interaction Overview diagram*
	Timing diagram*

* : non esiste in UML 1.x

Tassonomia dei diagrammi UML 2



Interaction Diagrams

Interaction Diagrams

• Quattro tipi di Interaction Diagrams:

- Sequence diagrams: mostra la sequenza temporale degli avvenimenti per ogni entità nel diagramma
- Communication diagrams: mostra le relazioni strutturali e i collegamenti tra le entità e messaggi che vi transitano
- Interaction overview diagrams: mostra la costruzione di interazioni complesse a partire d'interazioni più semplici.
- Timing diagrams: mostra l'evoluzione dell'interazione in tempo reale.

Interaction Diagrams

• Quattro tipi di Interaction Diagrams:

- Sequence diagrams: mostra la sequenza temporale degli avvenimenti per ogni entità nel diagramma
- Communication diagrams: mostra le relazioni strutturali e i collegamenti tra le entità e messaggi che vi transitano
- Interaction overview diagrams: mostra la costruzione di interazioni complesse a partire d'interazioni più semplici.
- Timing diagrams: mostra l'evoluzione dell'interazione in tempo reale.

Interaction Diagrams

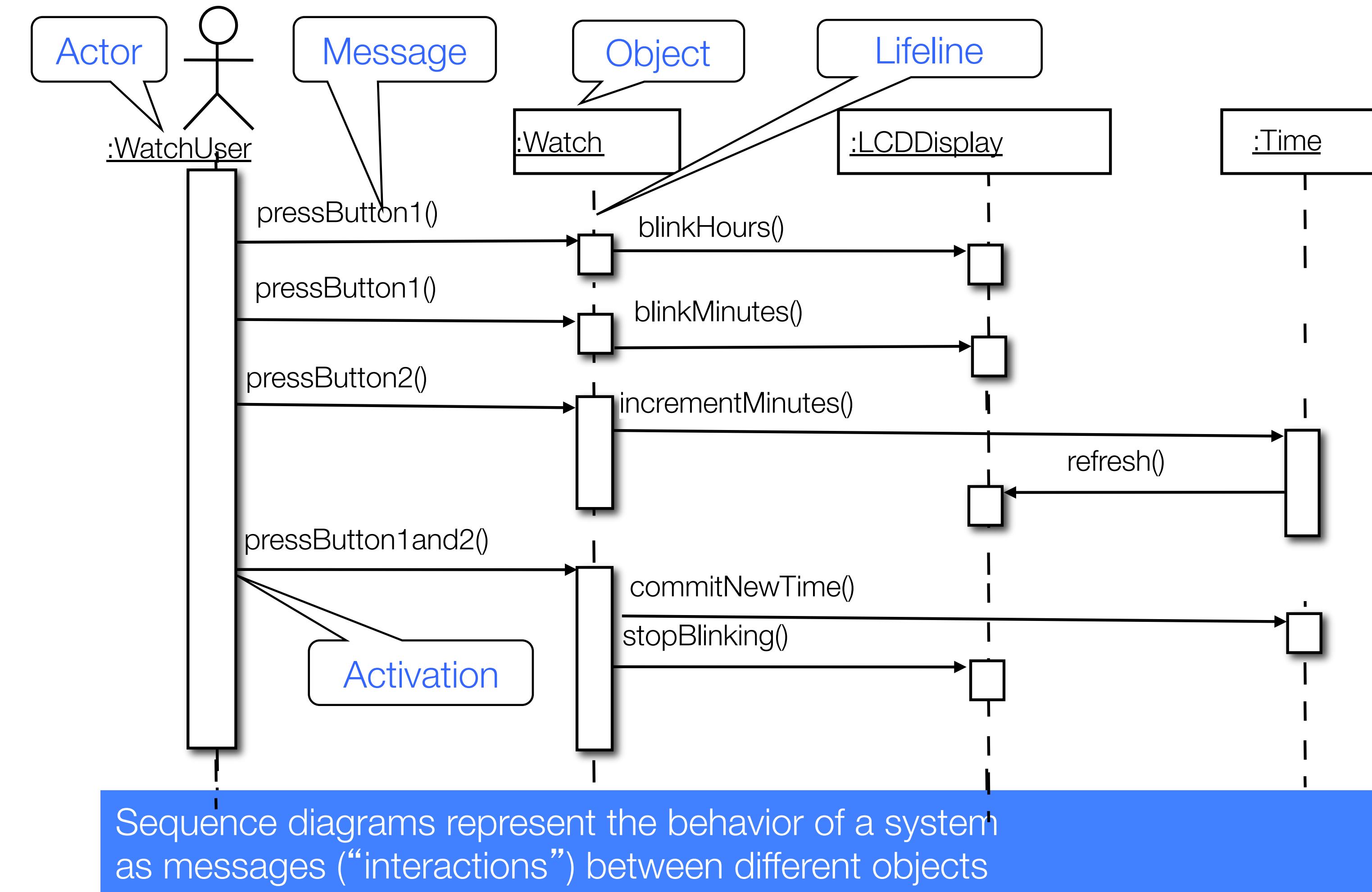
• Quattro tipi di Interaction Diagrams:

- Sequence diagrams: mostra la sequenza temporale degli avvenimenti per ogni entità nel diagramma
- Communication diagrams: mostra le relazioni strutturali e i collegamenti tra le entità e messaggi che vi transitano
- Interaction overview diagrams: mostra la costruzione di interazioni complesse a partire d'interazioni più semplici.
- Timing diagrams: mostra l'evoluzione dell'interazione in tempo reale.

• Il concetto di base degli Interaction Diagrams è la interazione

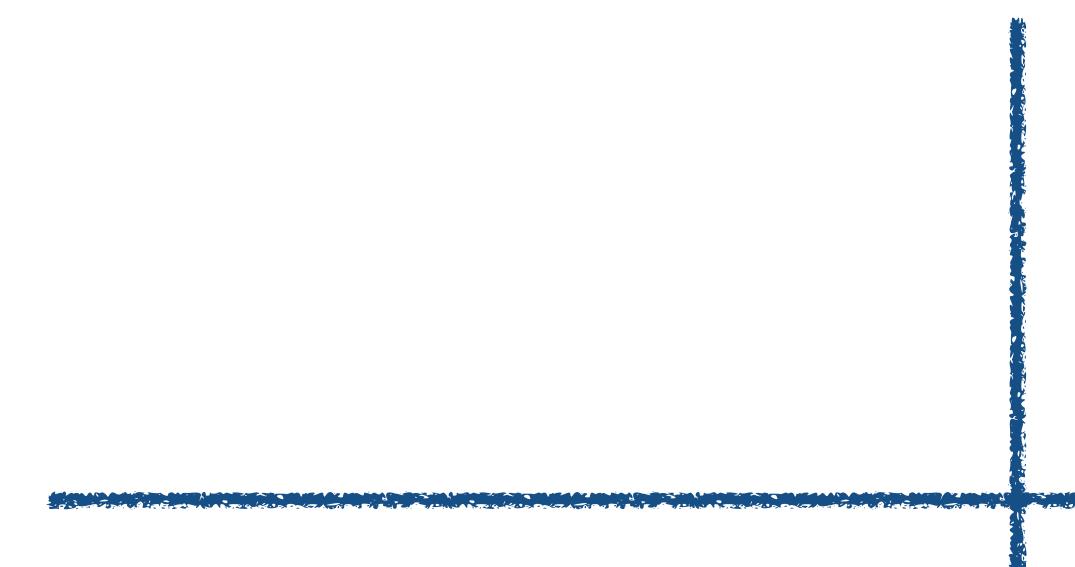
- è una "unità di comportamento" che si focalizza su scambi di informazione osservabili tra elementi

Sequence diagram: un primo esempio





Sequence diagram: notazione





Sequence diagram: notazione

- Usati durante l'analisi



Sequence diagram: notazione

- Usati durante l'analisi
- rifinire la descrizione dei casi d'uso



Sequence diagram: notazione

- Usati durante l'analisi
- rifinire la descrizione dei casi d'uso
- trovare oggetti addizionali ("participating objects")



Sequence diagram: notazione

- Usati durante l'analisi
 - rifinire la descrizione dei casi d'uso
 - trovare oggetti addizionali ("participating objects")
- Usati durante il system design



Sequence diagram: notazione

- Usati durante l'analisi
 - rifinire la descrizione dei casi d'uso
 - trovare oggetti addizionali ("participating objects")
- Usati durante il system design
 - per rifinire le interfacce dei sottosistemi

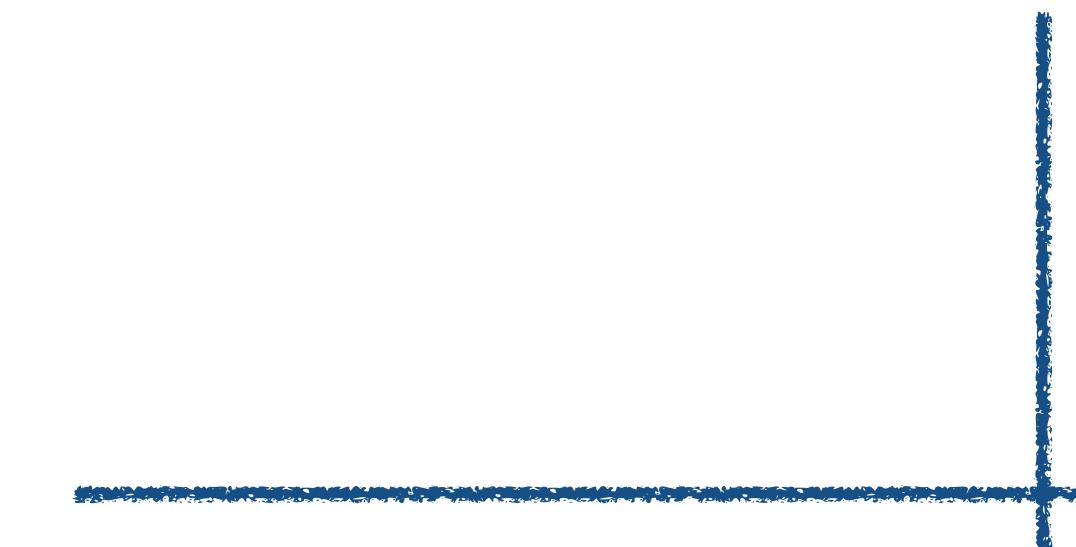
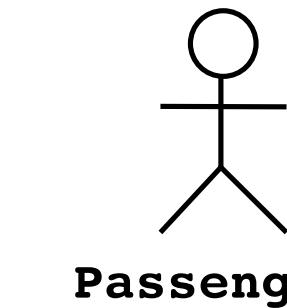


Sequence diagram: notazione

- Usati durante l'analisi
 - rifinire la descrizione dei casi d'uso
 - trovare oggetti addizionali ("participating objects")
- Usati durante il system design
 - per rifinire le interfacce dei sottosistemi
- Le *istanze* sono rappresentate da rettangoli. Gli attori da "omini"

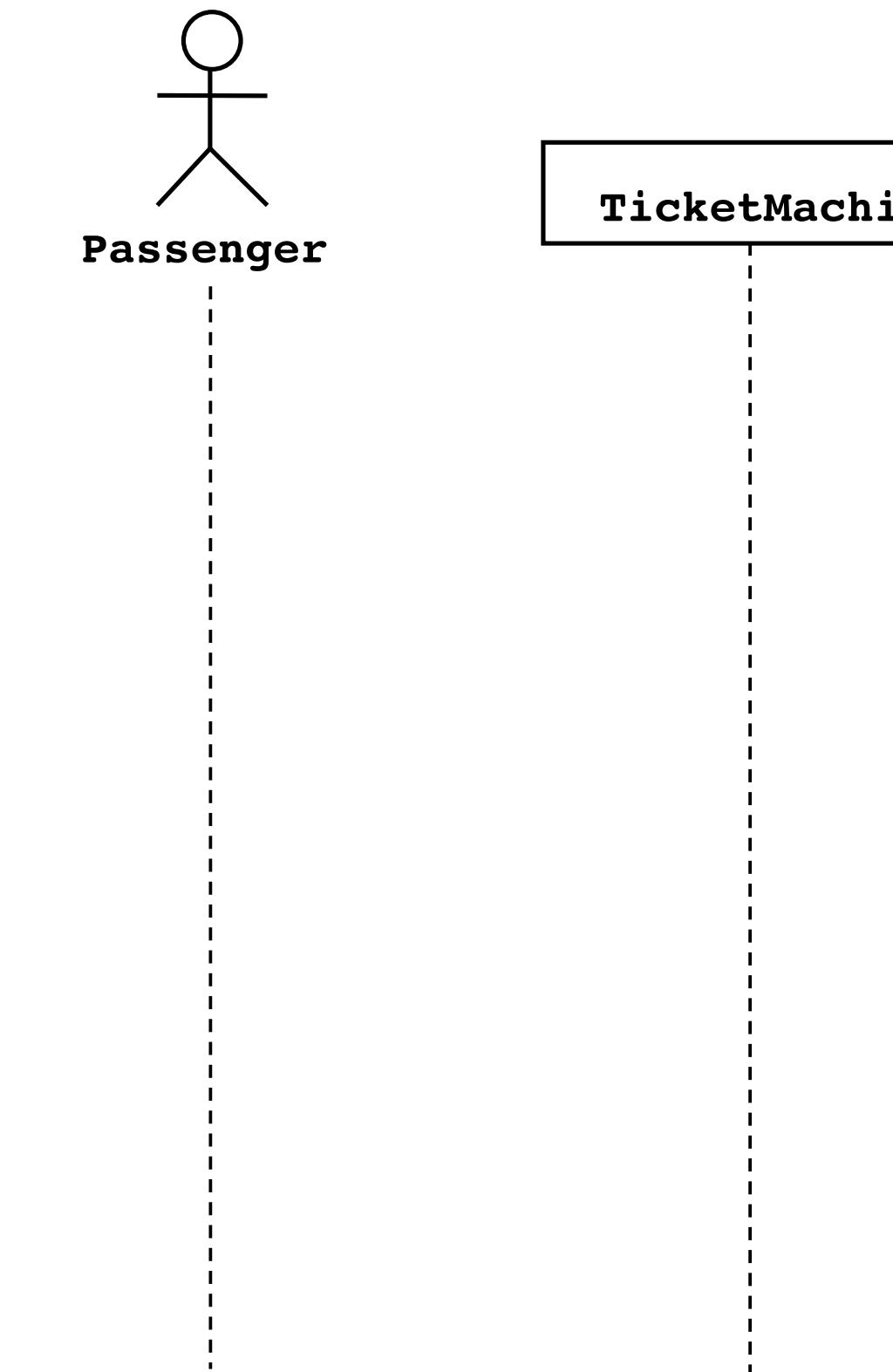
Sequence diagram: notazione

- Usati durante l'analisi
 - rifinire la descrizione dei casi d'uso
 - trovare oggetti addizionali ("participating objects")
- Usati durante il system design
 - per rifinire le interfacce dei sottosistemi
- Le *istanze* sono rappresentate da rettangoli. Gli attori da "omini"



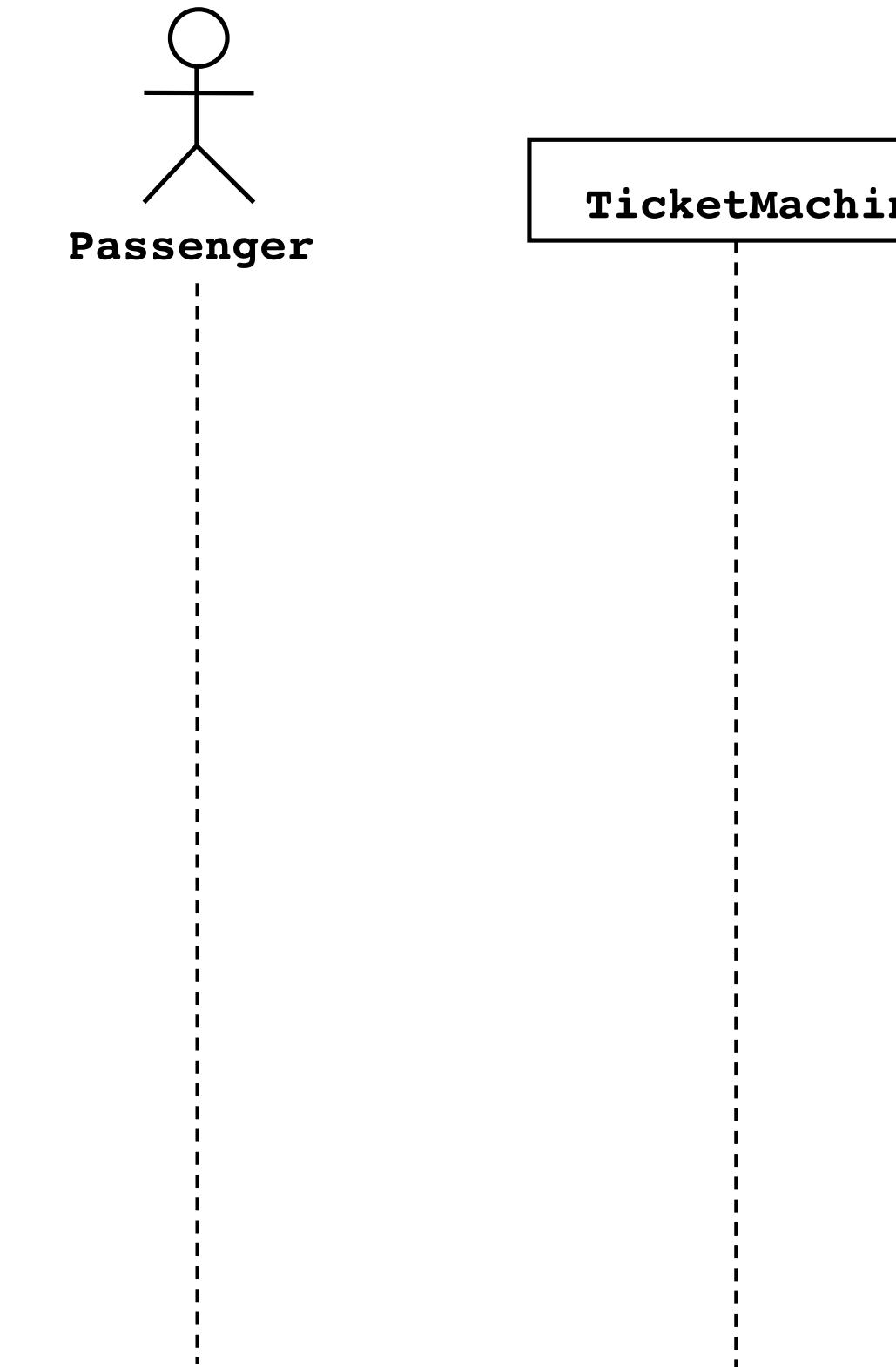
Sequence diagram: notazione

- Usati durante l'analisi
 - rifinire la descrizione dei casi d'uso
 - trovare oggetti addizionali ("participating objects")
- Usati durante il system design
 - per rifinire le interfacce dei sottosistemi
- Le *istanze* sono rappresentate da rettangoli. Gli attori da "omini"



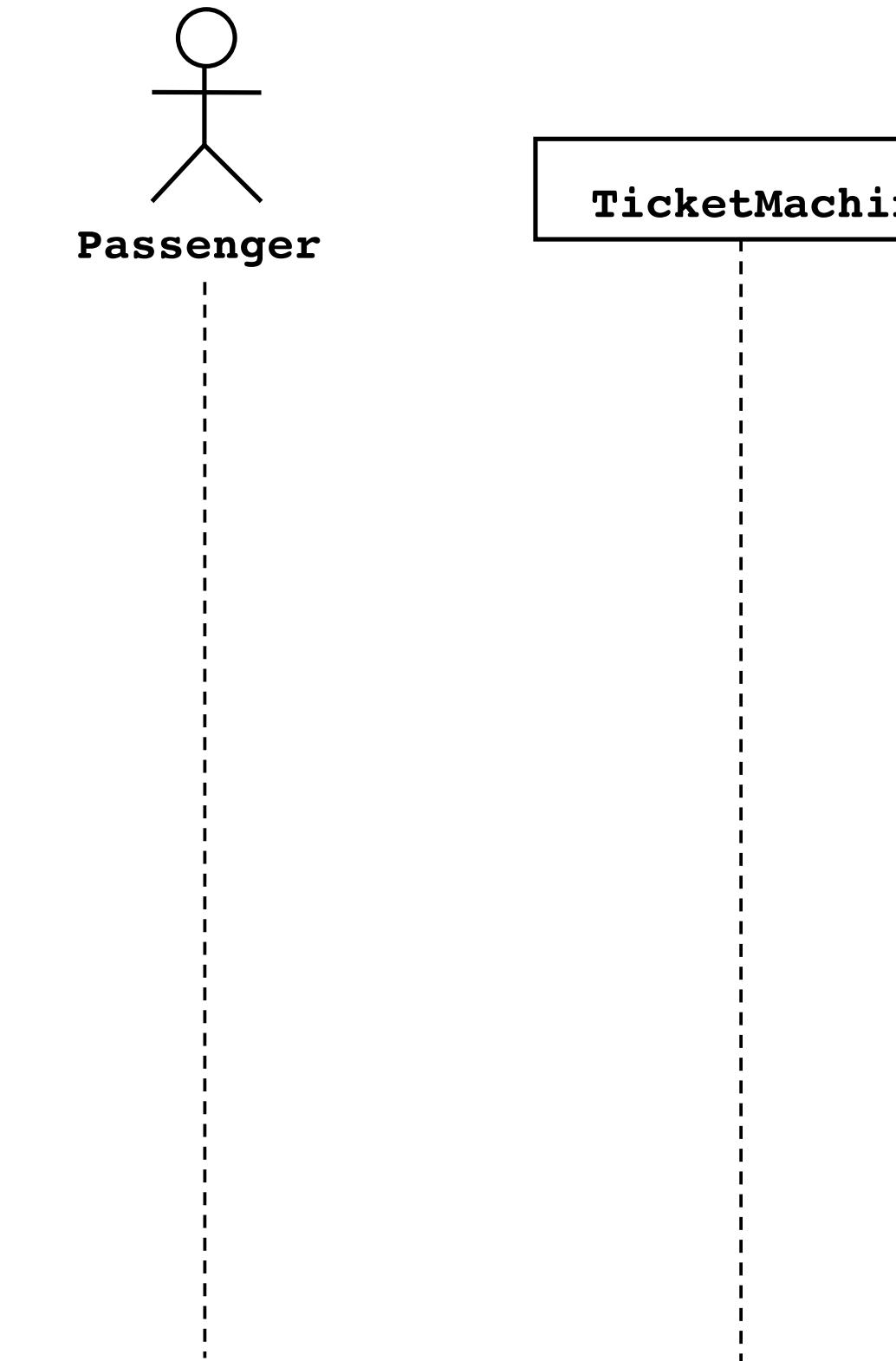
Sequence diagram: notazione

- Usati durante l'analisi
 - rifinire la descrizione dei casi d'uso
 - trovare oggetti addizionali ("participating objects")
- Usati durante il system design
 - per rifinire le interfacce dei sottosistemi
- Le *istanze* sono rappresentate da rettangoli. Gli attori da "omini"
- Le *lifelines* sono rappresentate da linee tratteggiate, indicano una sequenza temporale, gli eventi hanno luogo nel loro ordine



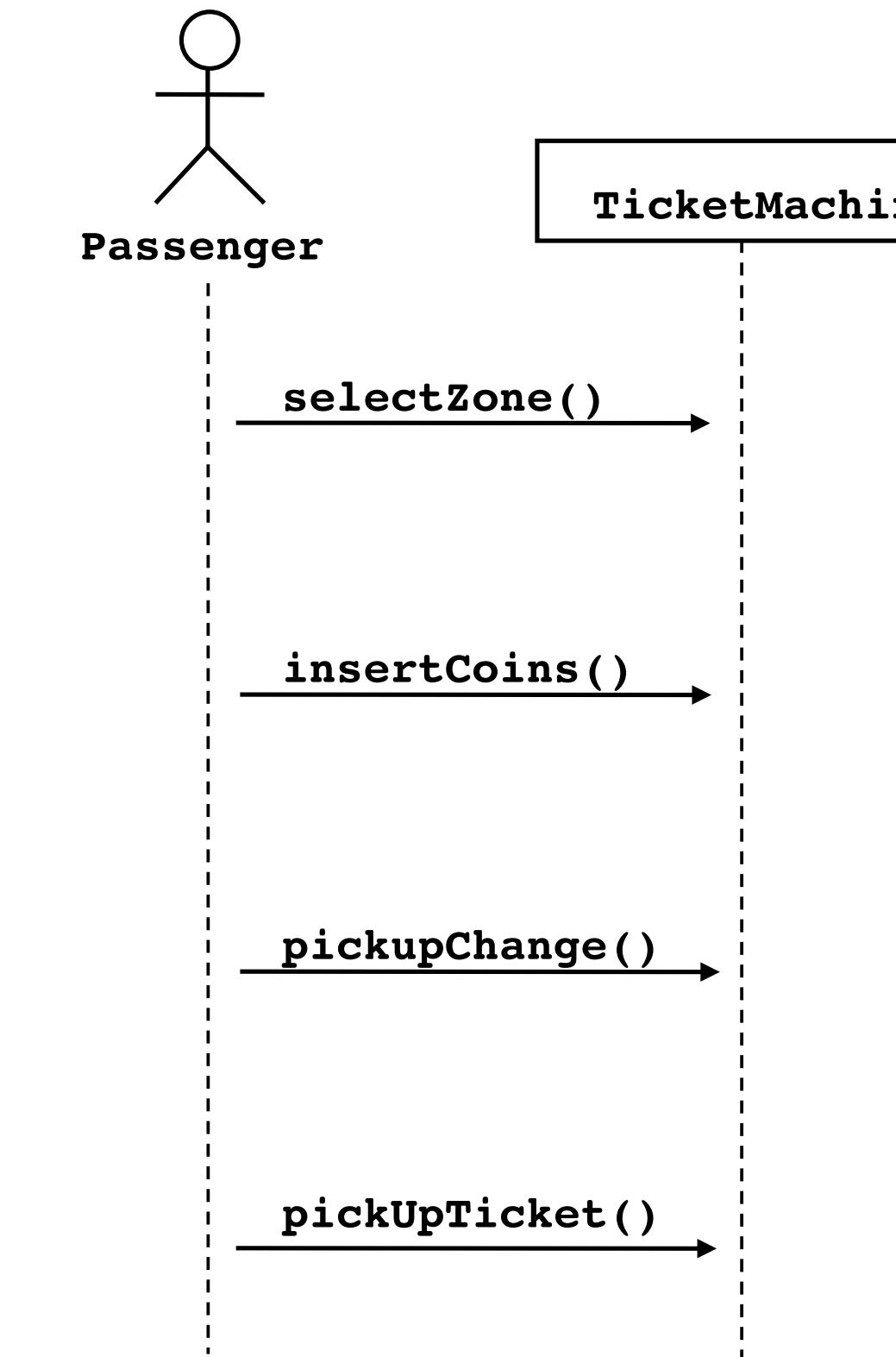
Sequence diagram: notazione

- Usati durante l'analisi
 - rifinire la descrizione dei casi d'uso
 - trovare oggetti addizionali ("participating objects")
- Usati durante il system design
 - per rifinire le interfacce dei sottosistemi
 - Le *istanze* sono rappresentate da rettangoli. Gli attori da "omini"
 - Le *lifelines* sono rappresentate da linee tratteggiate, indicano una sequenza temporale, gli eventi hanno luogo nel loro ordine
- I *messaggi* sono rappresentati da frecce



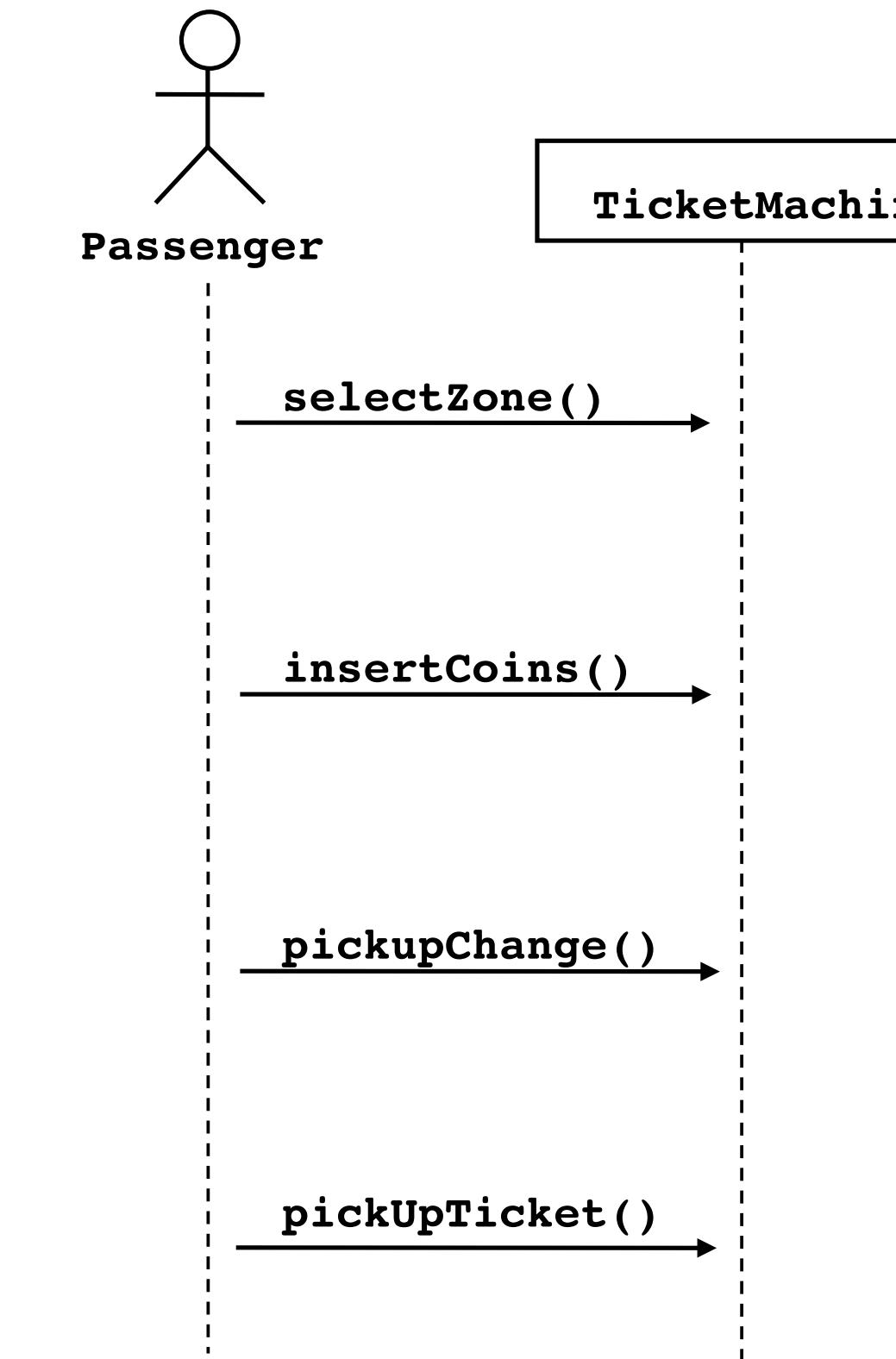
Sequence diagram: notazione

- Usati durante l'analisi
 - rifinire la descrizione dei casi d'uso
 - trovare oggetti addizionali ("participating objects")
- Usati durante il system design
 - per rifinire le interfacce dei sottosistemi
- Le *istanze* sono rappresentate da rettangoli. Gli attori da "omini"
- Le *lifelines* sono rappresentate da linee tratteggiate, indicano una sequenza temporale, gli eventi hanno luogo nel loro ordine
- I *messaggi* sono rappresentati da frecce



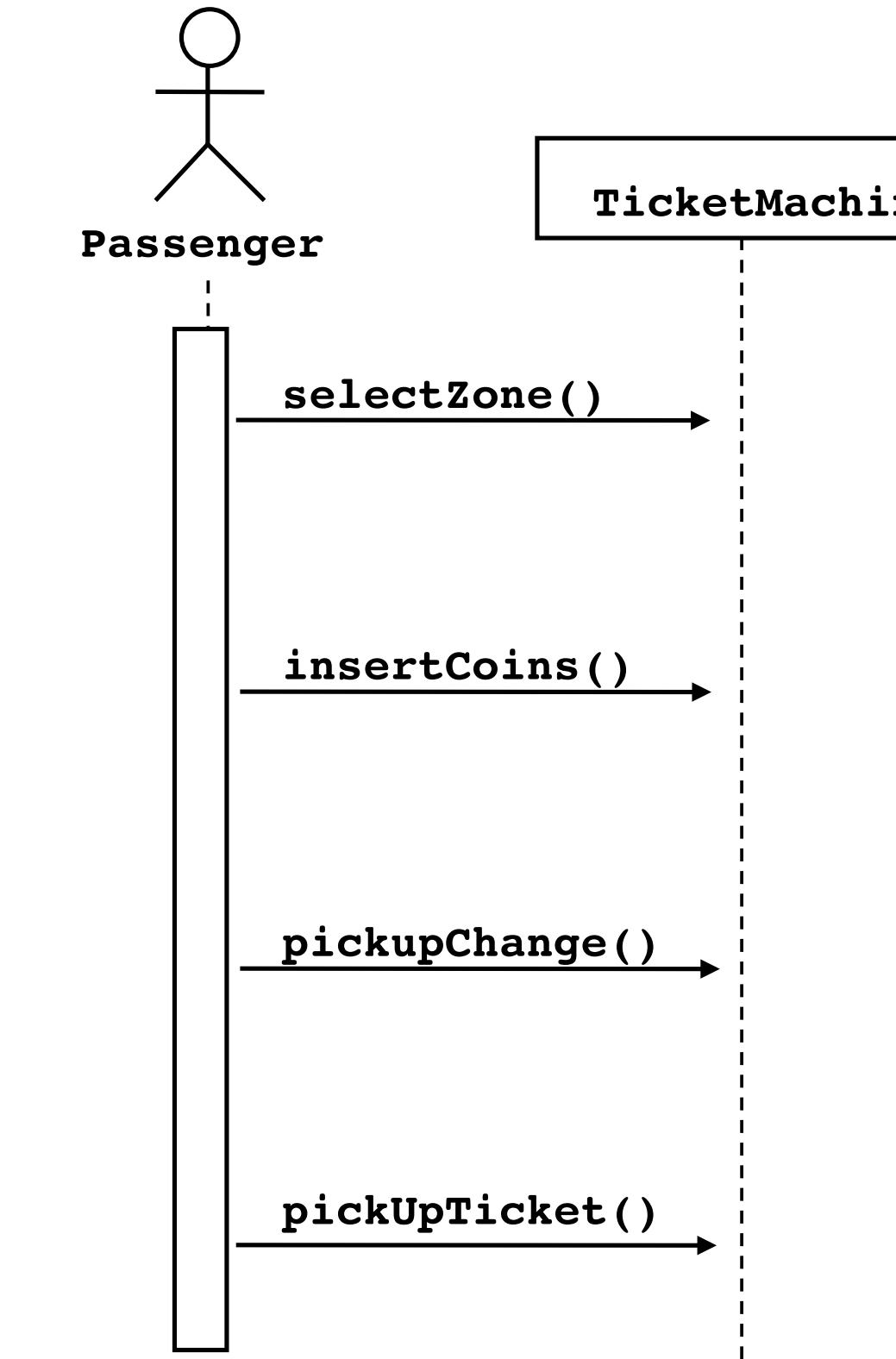
Sequence diagram: notazione

- Usati durante l'analisi
 - rifinire la descrizione dei casi d'uso
 - trovare oggetti addizionali ("participating objects")
- Usati durante il system design
 - per rifinire le interfacce dei sottosistemi
- Le *istanze* sono rappresentate da rettangoli. Gli attori da "omini"
- Le *lifelines* sono rappresentate da linee tratteggiate, indicano una sequenza temporale, gli eventi hanno luogo nel loro ordine
- I *messaggi* sono rappresentati da frecce
- Le *activations* sono rappresentate da rettangoli allungati.



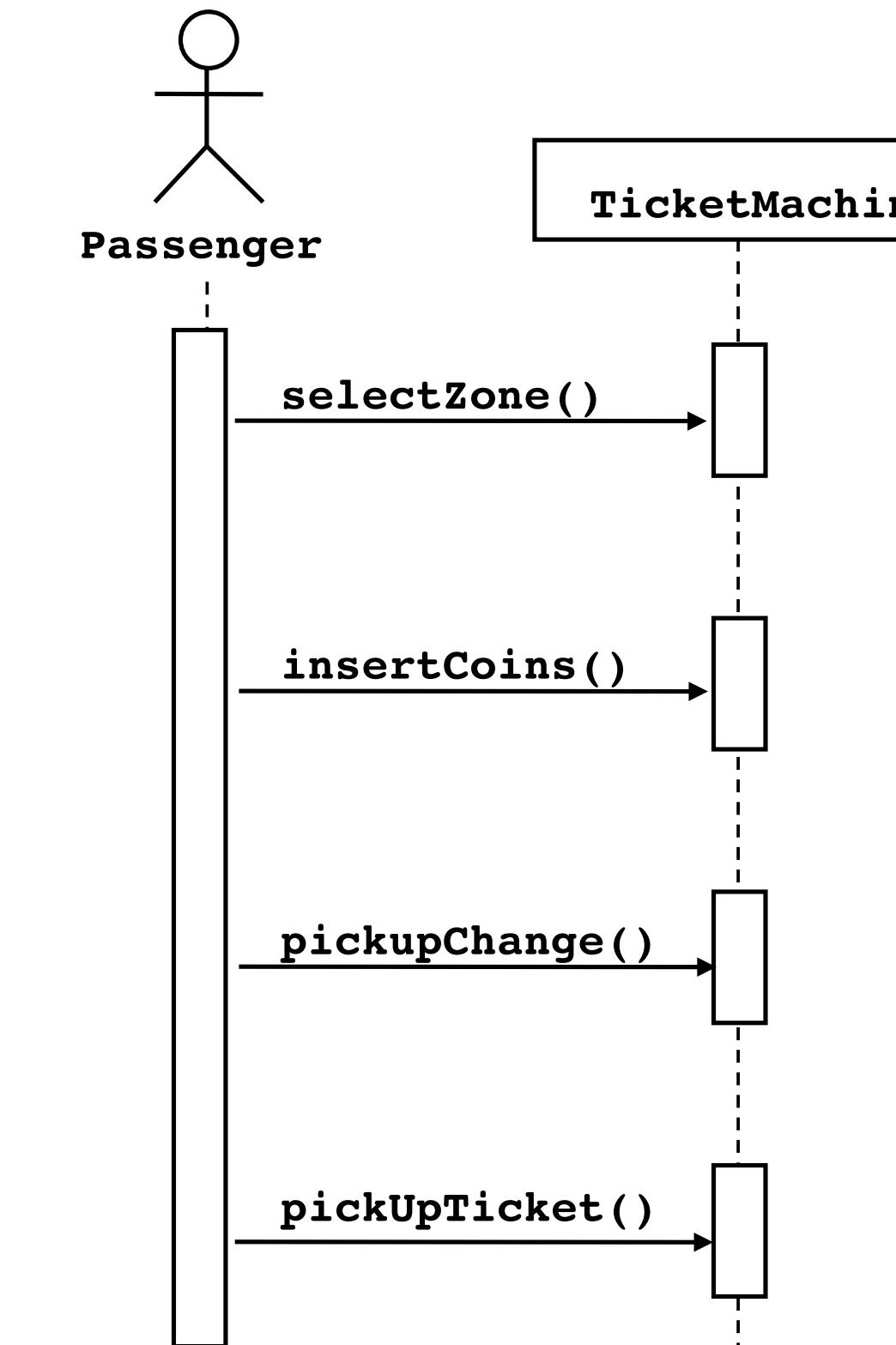
Sequence diagram: notazione

- Usati durante l'analisi
 - rifinire la descrizione dei casi d'uso
 - trovare oggetti addizionali ("participating objects")
- Usati durante il system design
 - per rifinire le interfacce dei sottosistemi
 - Le *istanze* sono rappresentate da rettangoli. Gli attori da "omini"
 - Le *lifelines* sono rappresentate da linee tratteggiate, indicano una sequenza temporale, gli eventi hanno luogo nel loro ordine
- I *messaggi* sono rappresentati da frecce
- Le *activations* sono rappresentate da rettangoli allungati.



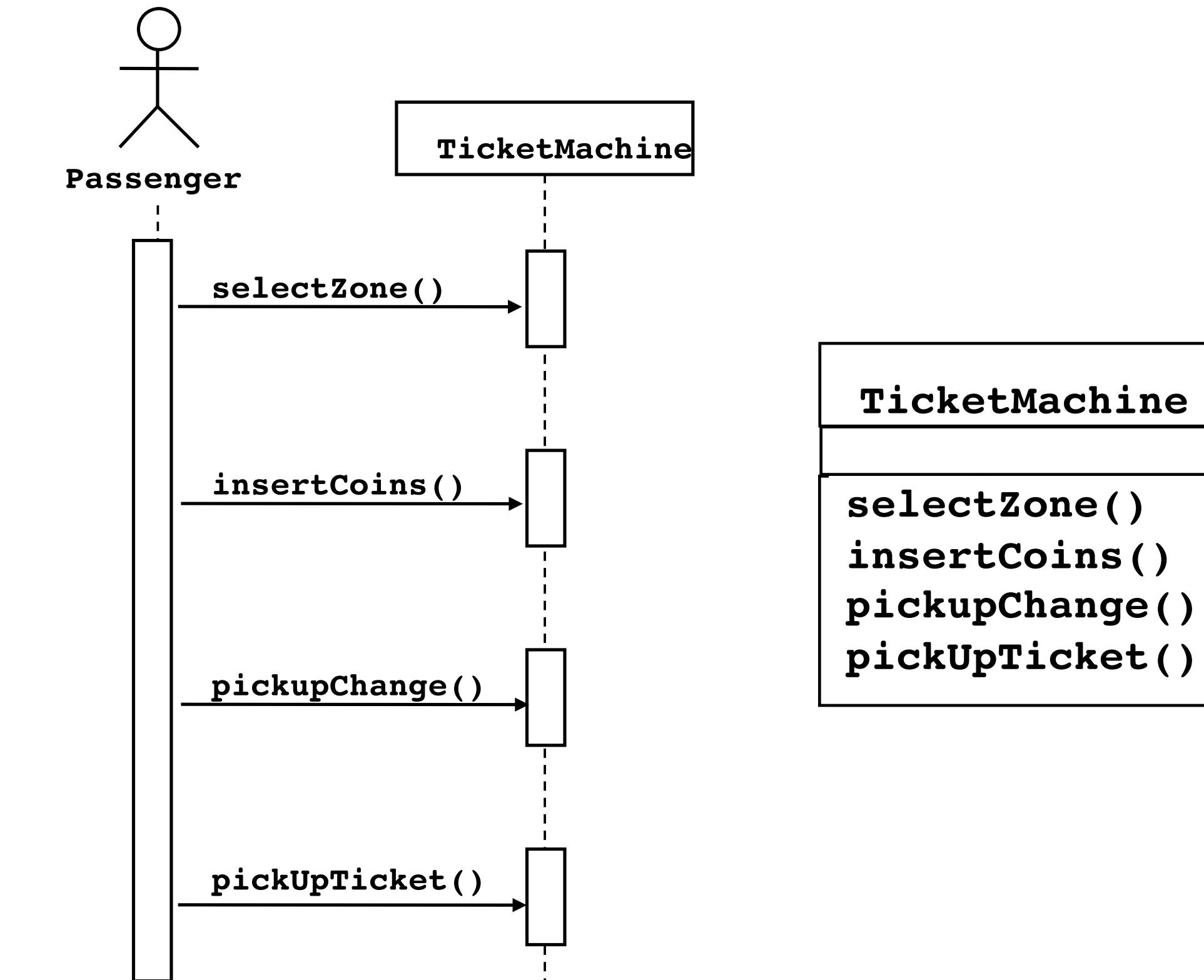
Sequence diagram: notazione

- Usati durante l'analisi
 - rifinire la descrizione dei casi d'uso
 - trovare oggetti addizionali ("participating objects")
- Usati durante il system design
 - per rifinire le interfacce dei sottosistemi
 - Le *istanze* sono rappresentate da rettangoli. Gli attori da "omini"
 - Le *lifelines* sono rappresentate da linee tratteggiate, indicano una sequenza temporale, gli eventi hanno luogo nel loro ordine
- I *messaggi* sono rappresentati da frecce
- Le *activations* sono rappresentate da rettangoli allungati.



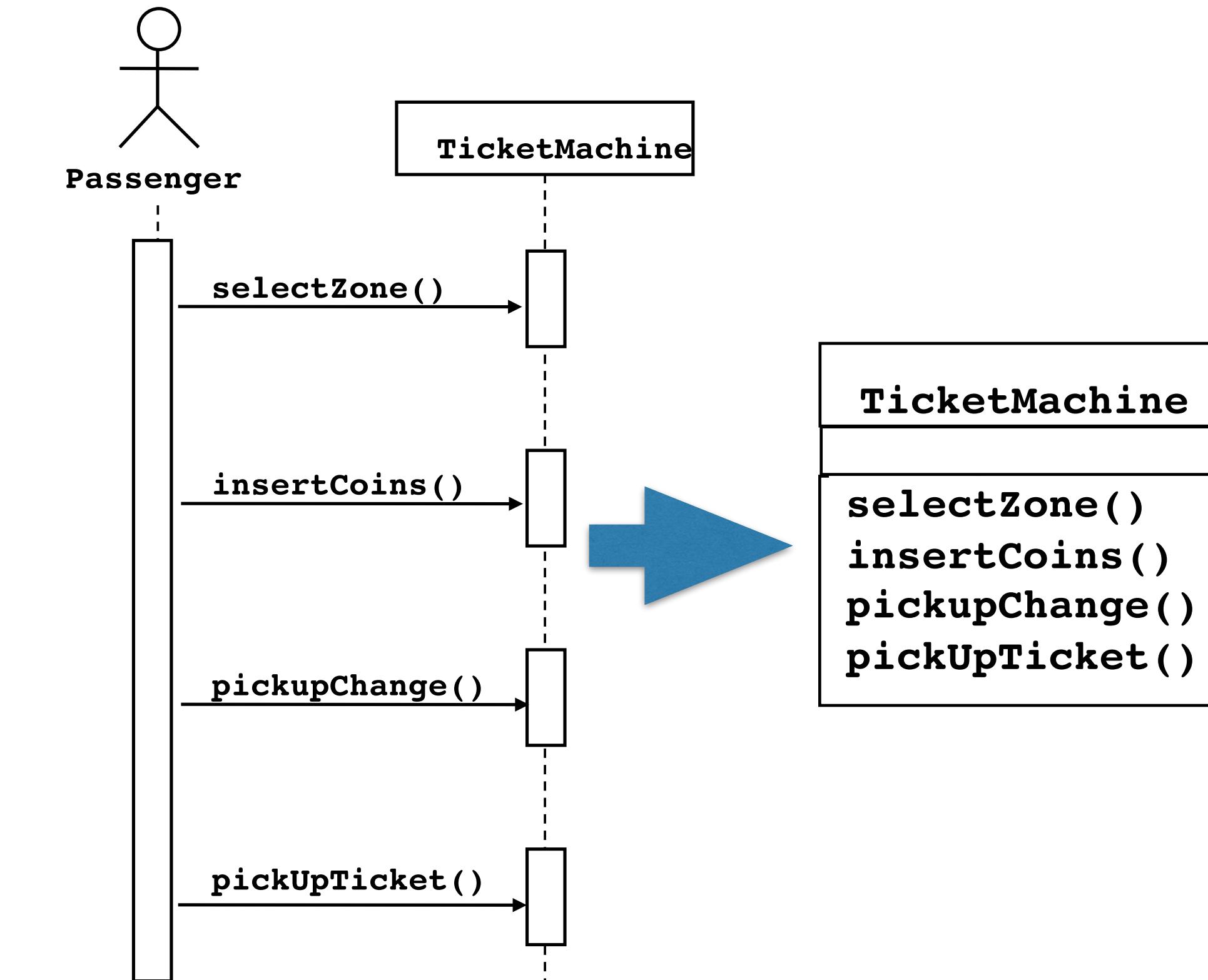
Sequence diagram: notazione

- Usati durante l'analisi
 - rifinire la descrizione dei casi d'uso
 - trovare oggetti addizionali ("participating objects")
- Usati durante il system design
 - per rifinire le interfacce dei sottosistemi
 - Le *istanze* sono rappresentate da rettangoli. Gli attori da "omini"
 - Le *lifelines* sono rappresentate da linee tratteggiate, indicano una sequenza temporale, gli eventi hanno luogo nel loro ordine
 - I *messaggi* sono rappresentati da frecce
 - Le *activations* sono rappresentate da rettangoli allungati.



Sequence diagram: notazione

- Usati durante l'analisi
 - rifinire la descrizione dei casi d'uso
 - trovare oggetti addizionali ("participating objects")
- Usati durante il system design
 - per rifinire le interfacce dei sottosistemi
 - Le *istanze* sono rappresentate da rettangoli. Gli attori da "omini"
 - Le *lifelines* sono rappresentate da linee tratteggiate, indicano una sequenza temporale, gli eventi hanno luogo nel loro ordine
 - I *messaggi* sono rappresentati da frecce
 - Le *activations* sono rappresentate da rettangoli allungati.



Sequence diagram: notazione

- Nei diagrammi di interazione generalmente non compaiono direttamente classificatori come le classi: al loro posto ci sono
 - Istanze di classificatori (oggetti, istanze di attori, etc.)
 - Linee di vita (lifeline) di classificatori (esprimono ruoli, non specifici oggetti)
- La differenza tra istanze e linee di vita è sottile, ma in generale è preferibile usare le linee di vita.

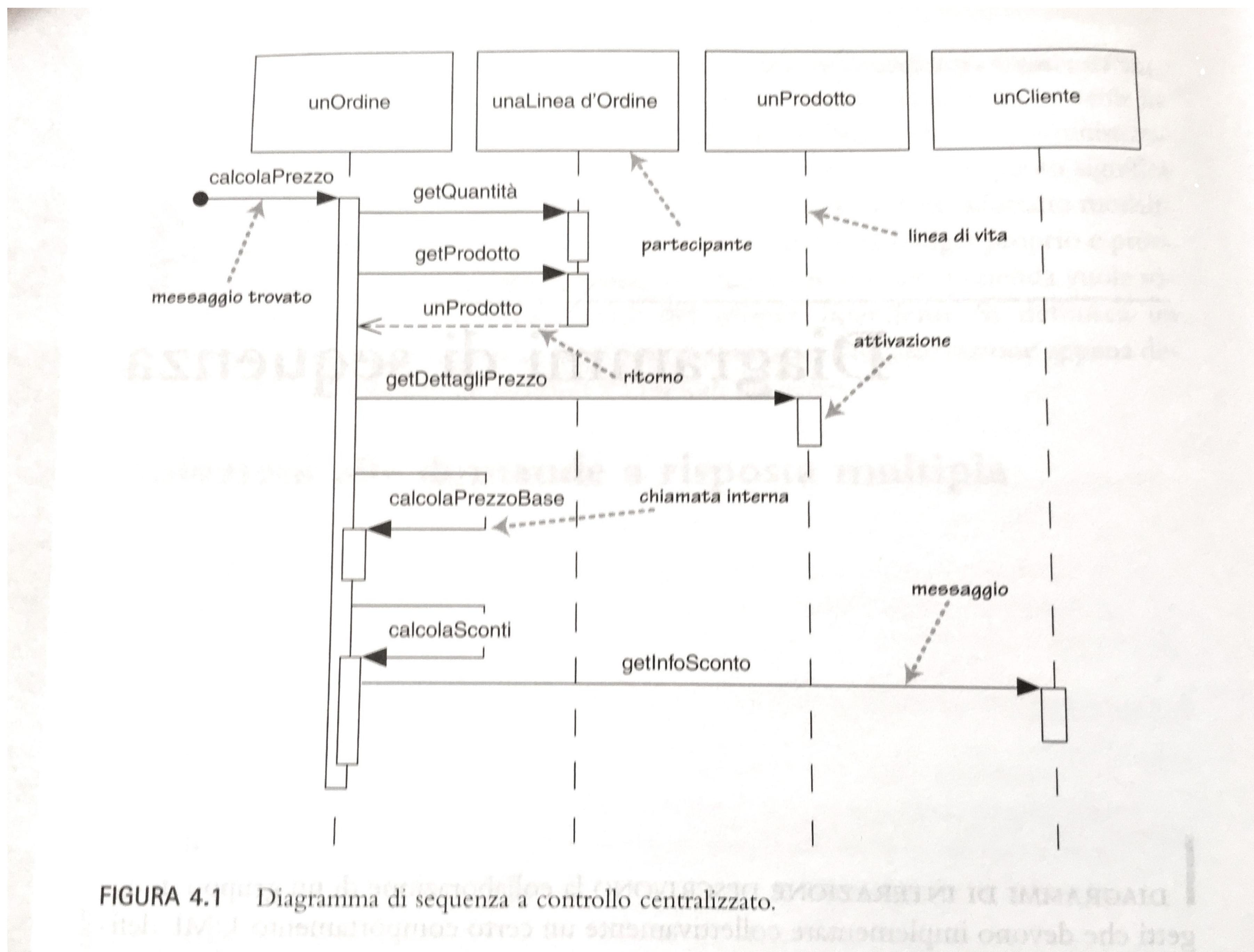


FIGURA 4.1 Diagramma di sequenza a controllo centralizzato.

Istanze e linee di vita

• ♀. Un'istanza:

- rappresenta uno specifico oggetto, e solo quello

• ♀. Una linea di vita:

- è più generale

- rappresenta un'istanza arbitraria di un classificatore

- fornisce modi per specificare come quest'istanza viene scelta

- esprime il ruolo giocato da un'istanza senza preoccuparsi della sua identità

• ♀. Un diagramma basato su istanze è chiamato in forma d'istanza, uno basato su linee di vita è in forma generica.

Istanze e linee di vita

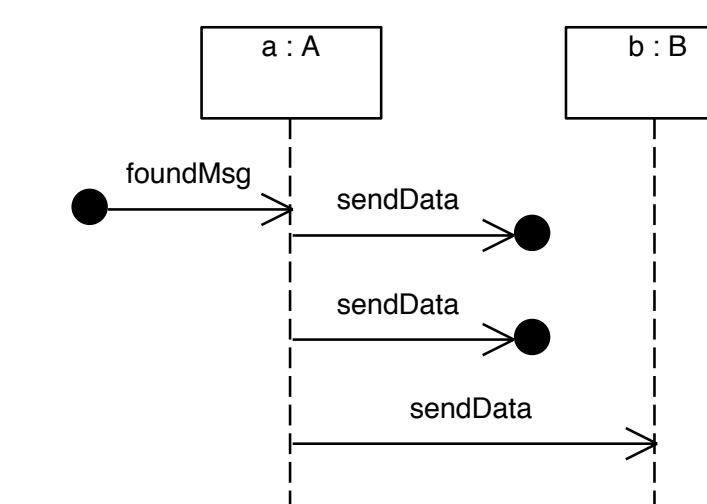
Jim : Person

buyer : Person

- Entrambe usano la notazione del loro classificatore, ma le linee di vita non sono sottolineate.
- La notazione completa per una linea di vita è: nome [selettore] : tipo
- Il selettore è un'espressione booleana opzionale che permette di scegliere un particolare rappresentante del classificatore, ad es. [id="1234"].
- Senza selettore la linea di vita rappresenta un'arbitraria istanza.

Messaggi

- Rappresentano comunicazioni tra linee di vita: segnali, chiamate di operazioni, creazione e distruzione di oggetti.
- Sette tipi definiti:
 - ◆ chiamata sincrona
 - ◆ chiamata asincrona
 - ◆ ritorno da chiamata
 - ◆ creazione
 - ◆ distruzione
 - ◆ messaggio trovato
 - ◆ messaggio perso

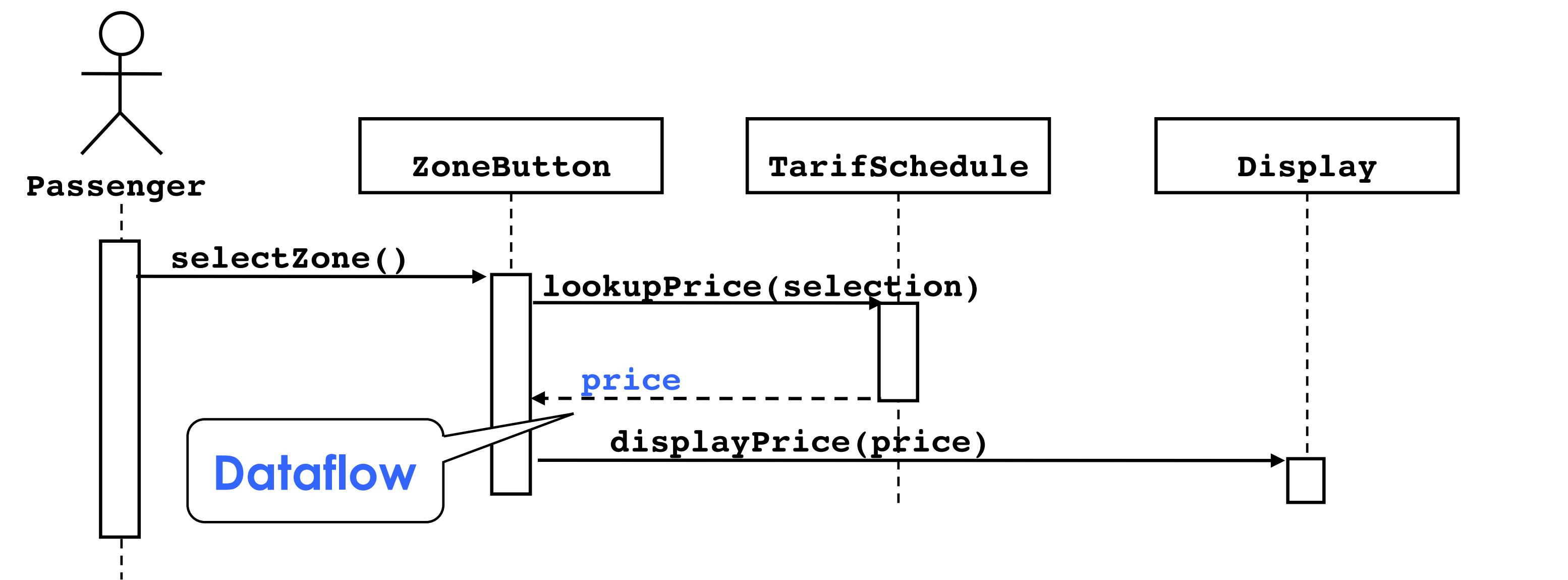


Messaggi

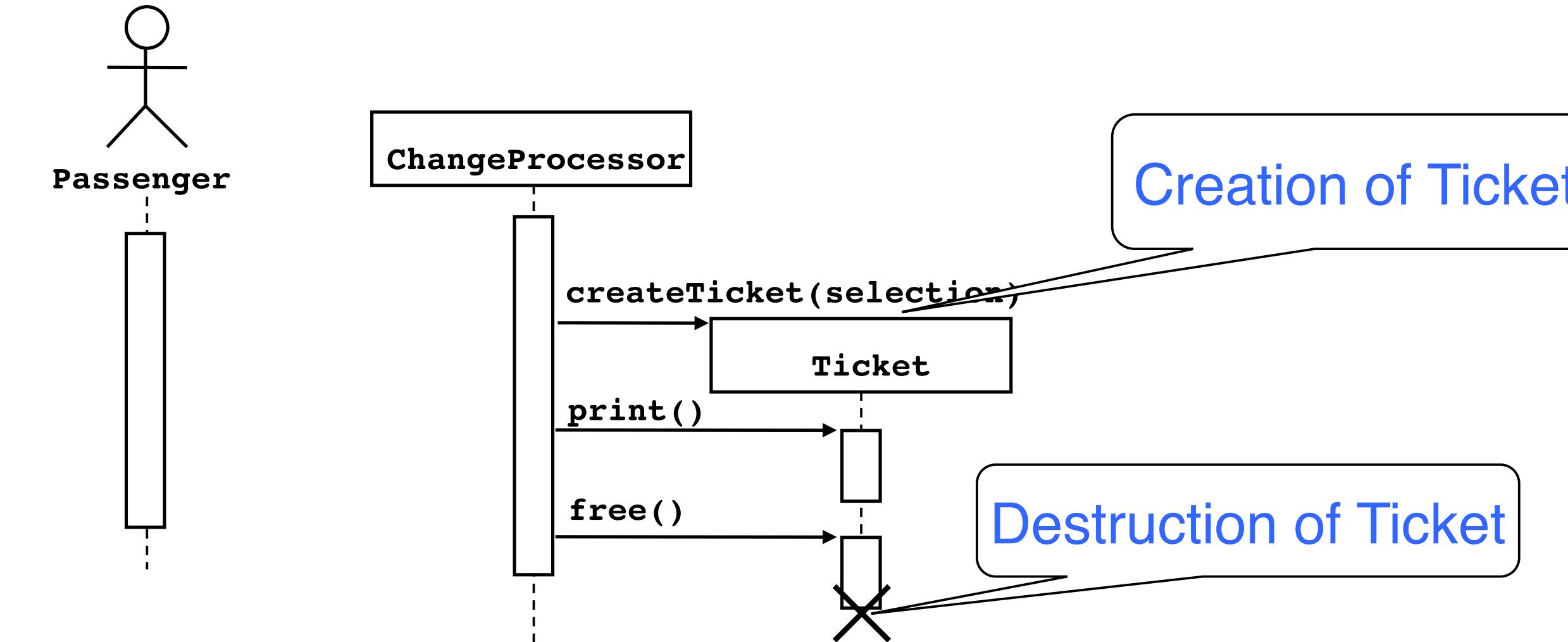
- I messaggi sincroni fanno bloccare chi li manda fino al messaggio di ritorno del destinatario.
- Nei messaggi asincroni il mittente non si aspetta un messaggio di ritorno e prosegue immediatamente.
- I messaggi di ritorno sono opzionali e in generale inclusi solo quando non ostacolano la leggibilità del diagramma oppure per segnalare valori di ritorno.
- La distinzione tra messaggi sincroni e asincroni in genere emerge in fase di progettazione.
- In fase di analisi, conviene indicare tutti i messaggi come sincroni (è il caso più vincolante).

Messaggi

- I sequence diagrams possono anche modellare il flusso dei dati:
- La ricezione di un messaggio innesca l'attivazione di una operazione (freccia piena)
- Un messaggio può avere un valore (dato) di ritorno (freccia tratteggiata)



Tipi di messaggi: creazione e distruzione



- La creazione di un oggetto è rappresentata da una freccia che punta all'oggetto
- La distruzione è rappresentata da una x alla fine della linea di attivazione
 - in ambienti dove è prevista la garbage collection, la distruzione può essere usata per denotare la fine della vita utile di un oggetto.

Capitolo 4 Diagrammi di sequenza

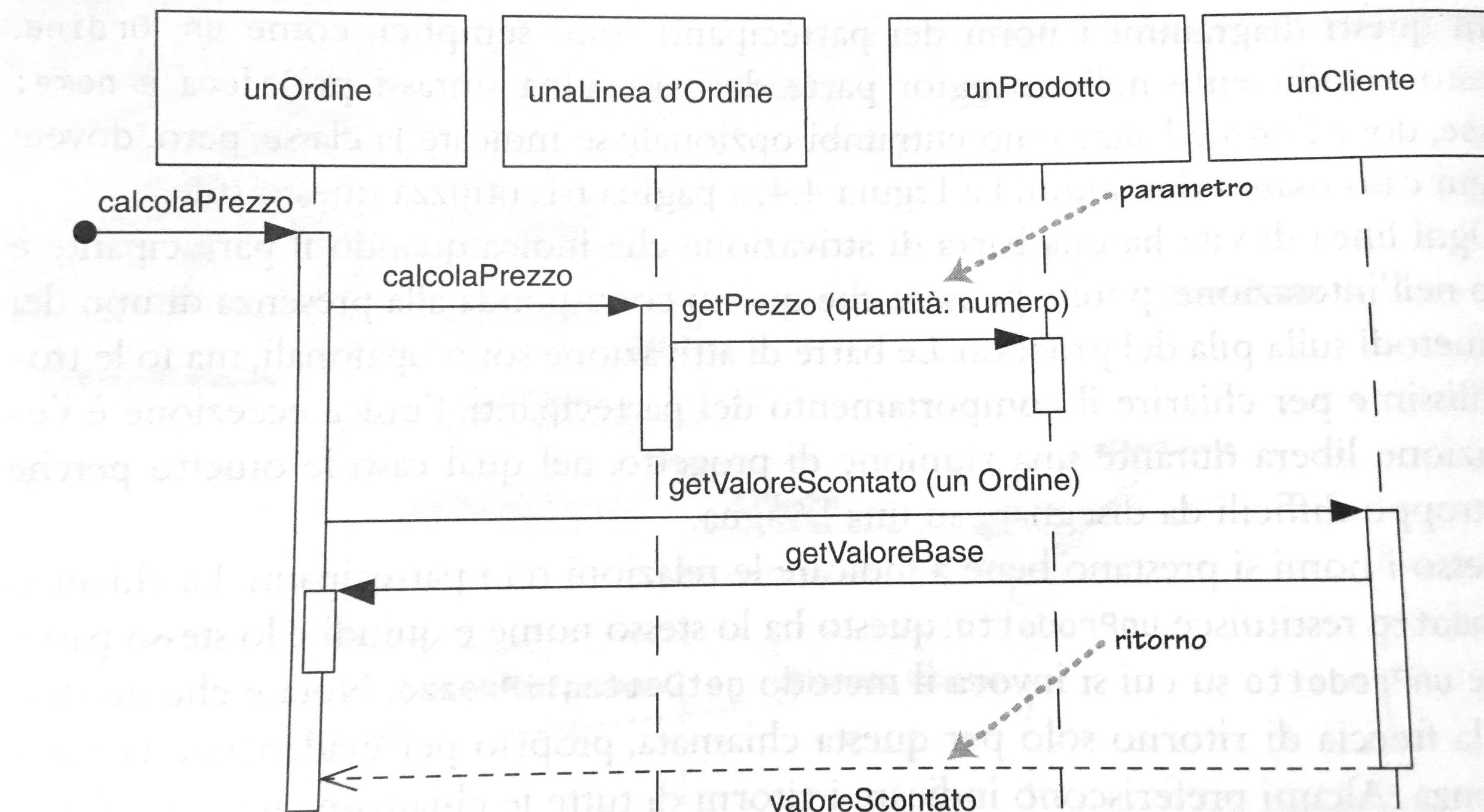


FIGURA 4.2 Diagramma di sequenza a controllo distribuito.

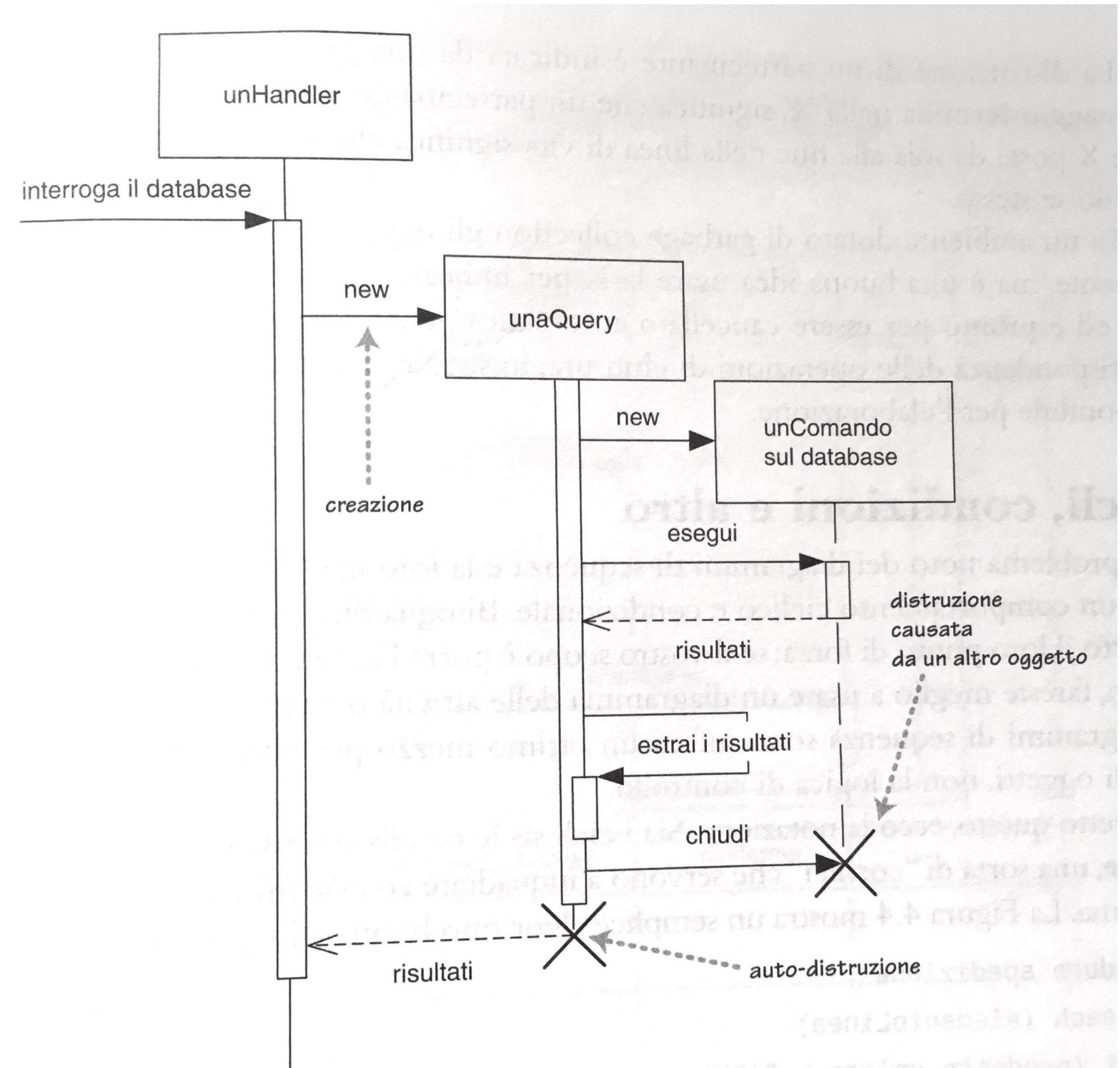


FIGURA 4.3 Creazione e distruzione dei partecipanti.

zione, una sorta di “cornici” che servono a inquadrare
gramma. La Figura 4.4 mostra un semplice algoritmo

```
procedura spedizione
    foreach (elementoLinea)
        if (prodotto.valore > $10K)
            raccomandata.spedizione
        else
            normale.spedizione
        end if
    end for
    if (necessitaConferma) messenger.conferma
end procedura
```

In generale, i frame selezionano uno o più framme

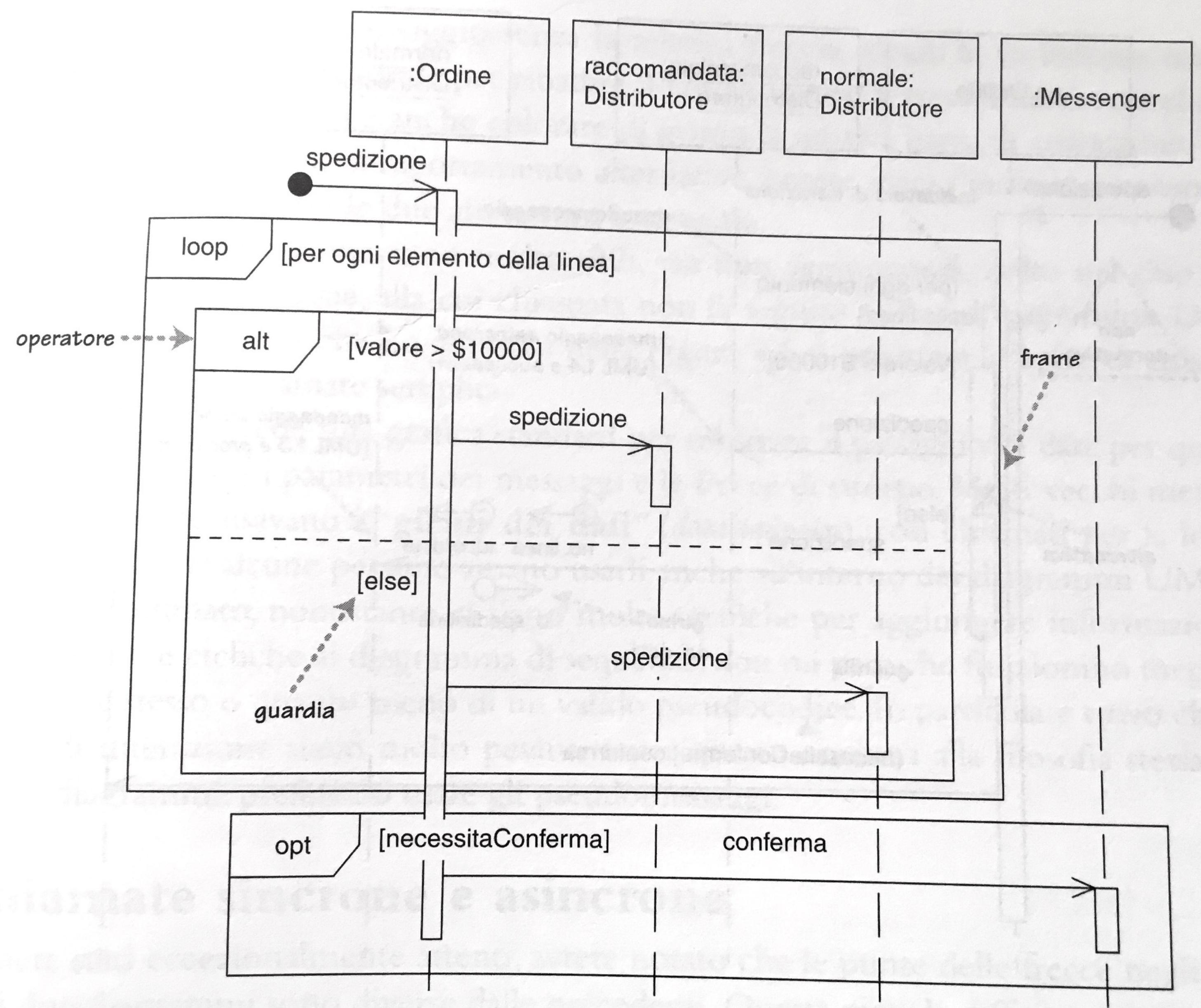


FIGURA 4.4 Frame di interazione.

FIGURA 4.4 Frame di interazione.

TABELLA 4.1 Alcuni operatori usati con i frame di interazione.

Operatore	Significato
alt	Frammenti multipli in alternativa; verrà eseguito solo quello per cui è verificata la condizione (Figura 4.4).
opt	Opzionale; il frammento viene eseguito solo se la condizione specificata è verificata. Equivalente a un alt con una sola traccia (Figura 4.4).
par	Parallelo; ogni frammento è eseguito in parallelo.
loop	Ciclo; il frammento può essere eseguito più volte, la base dell'iterazione è indicata dalla guardia (Figura 4.4).
region	Regione critica; il frammento può essere eseguito da un solo thread alla volta.
neg	Negativo; il frammento mostra un'interazione non valida.
ref	Riferimento; si riferisce a un'interazione definita in un altro diagramma. Il frame dev'essere disegnato in modo da racchiudere le linee di vita coinvolte nell'interazione. Potete indicare anche dei parametri e un tipo di ritorno.
sd	Sequence diagram; usato per racchiudere un intero diagramma di sequenza, se lo desiderate.

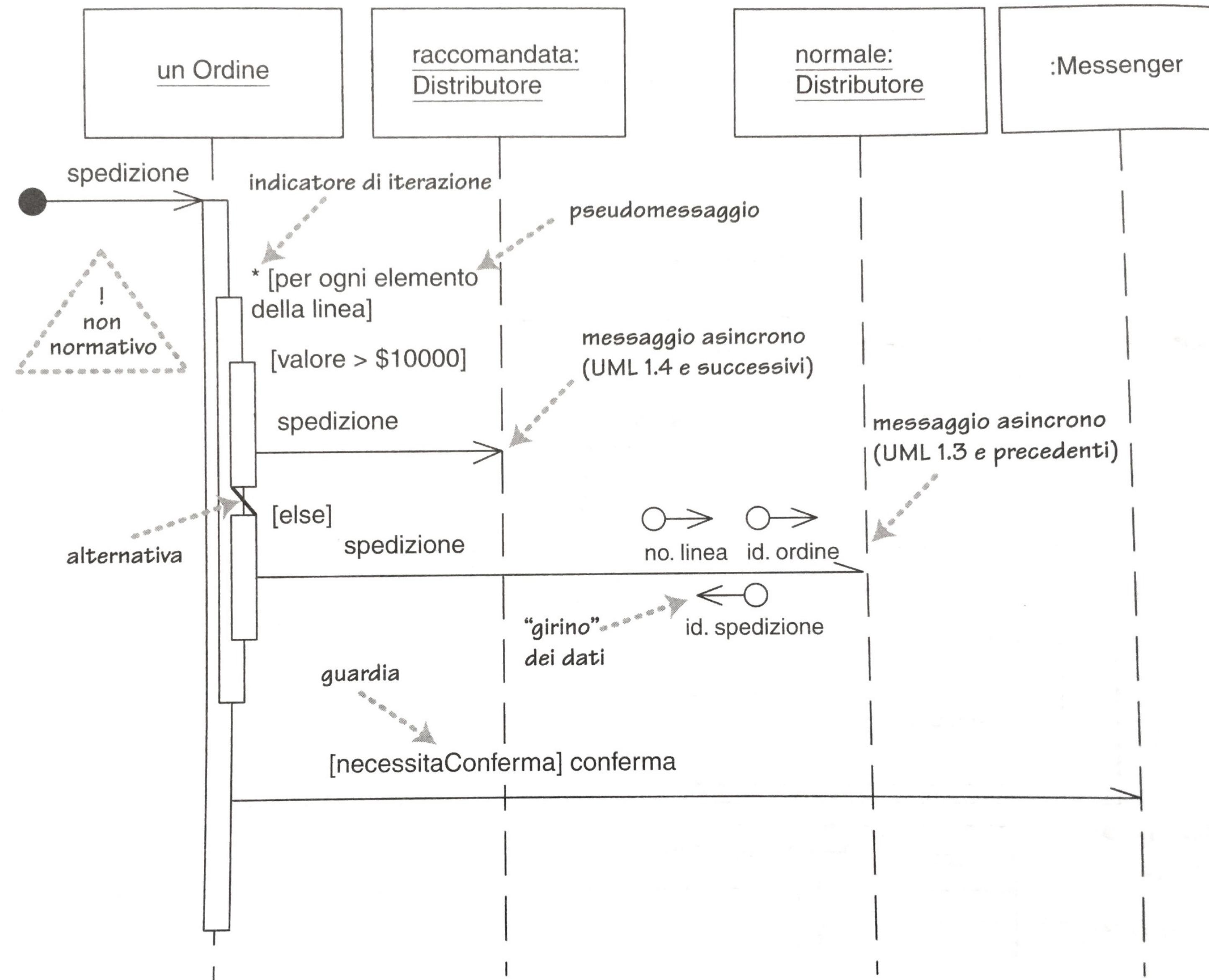
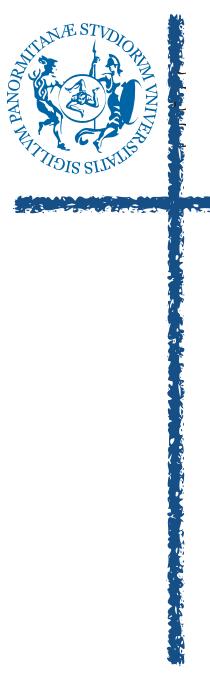


FIGURA 4.5 Vecchie convenzioni per la rappresentazione della logica di controllo.



Usiamo i Sequence Diagrams

Modellare le interazioni tra gli oggetti

Modellare le interazioni tra gli oggetti

• Finora abbiamo:

- ◆ identificato i requisiti non funzionali
- ◆ identificato i requisiti funzionali e li abbiamo rappresentati attraverso diagrammi dei casi d'uso
- ◆ abbiamo documentato il flusso degli eventi nei casi d'uso
- ◆ abbiamo identificato di oggetti:

• entity

• boundary

• control

• adesso leghiamo i casi d'uso agli oggetti tramite i sequence diagrams

Come....

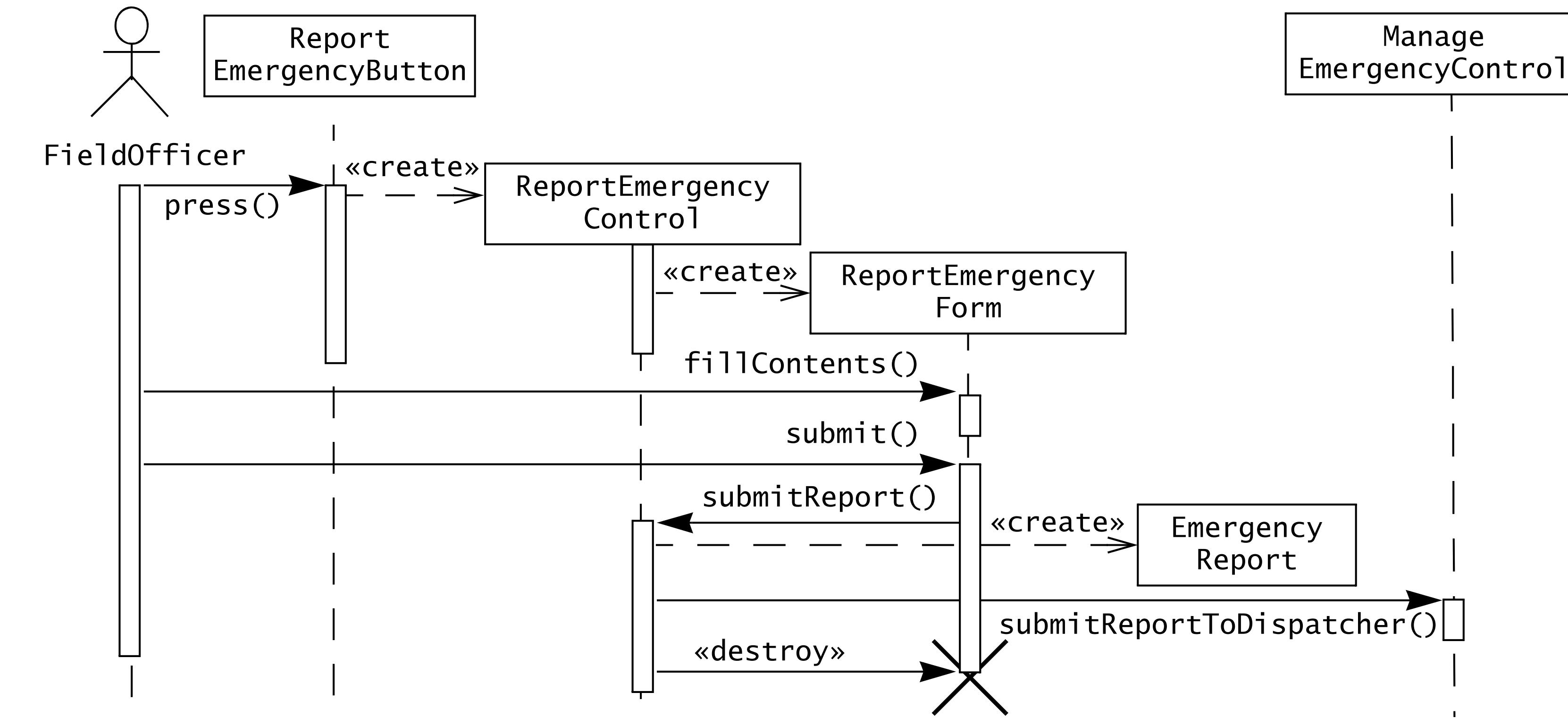
Heuristics for drawing sequence diagrams

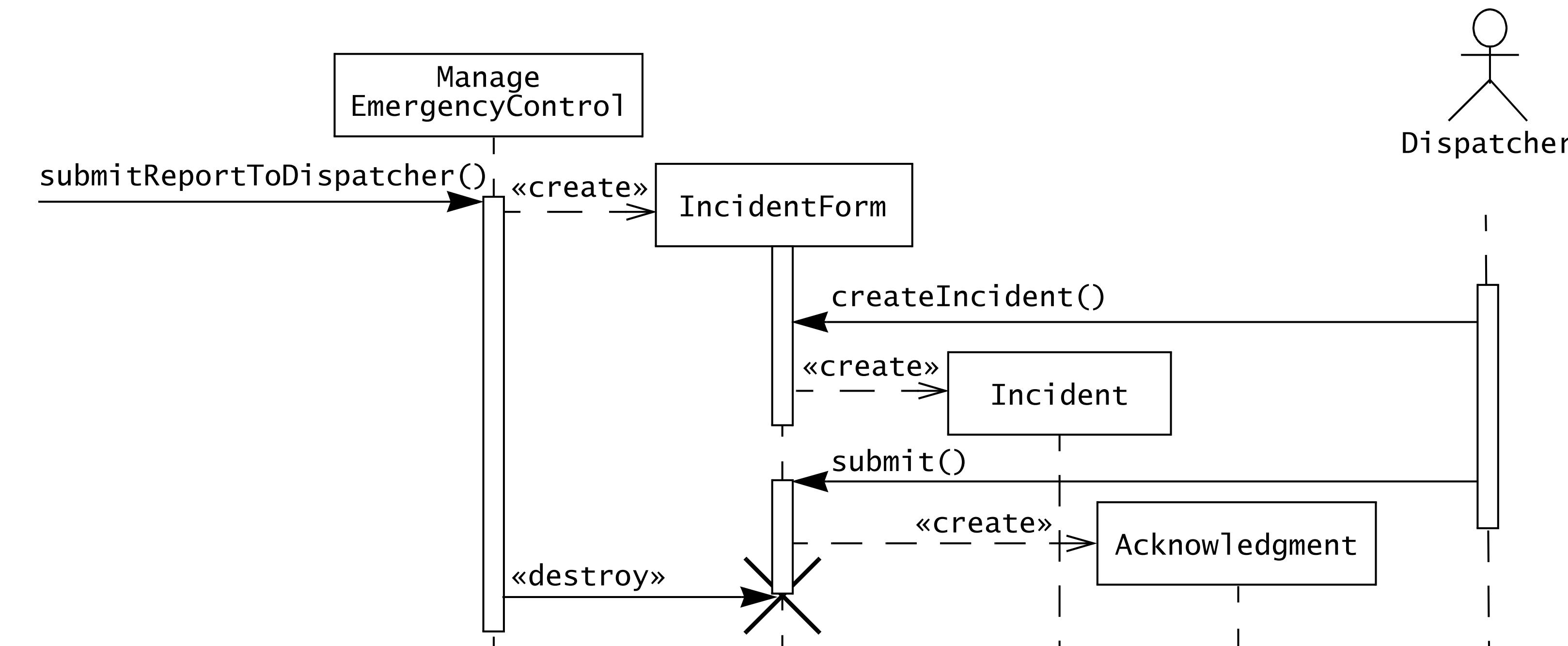
- The first column should correspond to the actor who initiated the use case.
- The second column should be a boundary object (that the actor used to initiate the use case).
- The third column should be the control object that manages the rest of the use case.
- Control objects are created by boundary objects initiating use cases.
- Boundary objects are created by control objects.
- Entity objects are accessed by control and boundary objects.
- Entity objects *never* access boundary or control objects; this makes it easier to share entity objects across use cases.

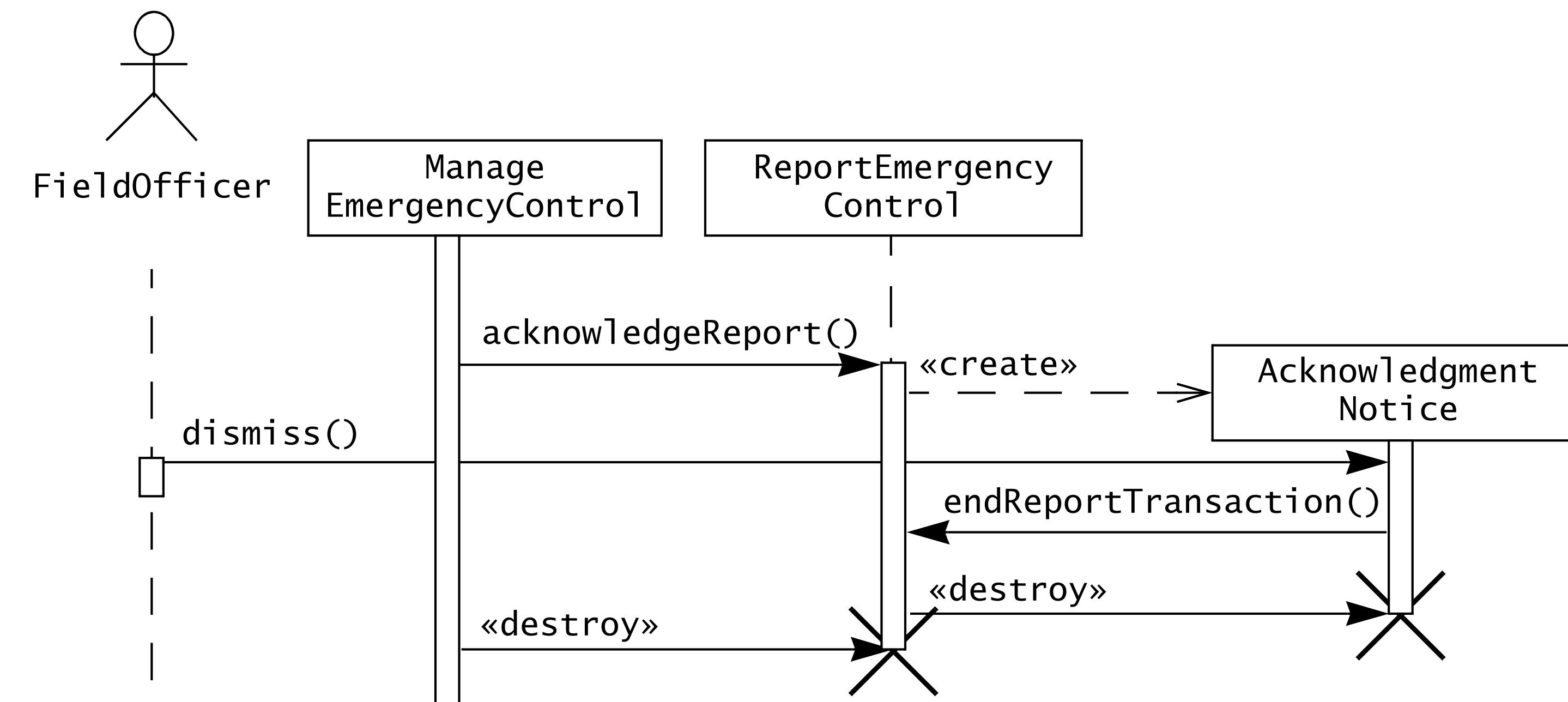
Esempio

<i>Use case name</i>	ReportEmergency
<i>Entry condition</i>	1. The FieldOfficer activates the “Report Emergency” function of her terminal.
<i>Flow of events</i>	2. FRIEND responds by presenting a form to the officer. The form includes an emergency type menu (general emergency, fire, transportation), a location, incident description, resource request, and hazardous material fields. 3. The FieldOfficer completes the form by specifying minimally the emergency type and description fields. The FieldOfficer may also describe possible responses to the emergency situation and request specific resources. Once the form is completed, the FieldOfficer submits the form by pressing the “Send Report” button, at which point, the Dispatcher is notified. 4. The Dispatcher reviews the information submitted by the FieldOfficer and creates an Incident in the database by invoking the OpenIncident use case. All the information contained in the FieldOfficer’s form is automatically included in the incident. The Dispatcher selects a response by allocating resources to the incident (with the AllocateResources use case) and acknowledges the emergency report by sending a FRIENDgram to the FieldOfficer.
<i>Exit condition</i>	5. The FieldOfficer receives the acknowledgment and the selected response.

Esempio







Considerazioni

- Costruendo il diagramma di sequenza abbiamo:
 - modellato l'ordine delle interazioni tra gli oggetti
 - distribuito il comportamento dei casi d'uso
- si assegna ad ogni oggetto la responsabilità in termini di operazioni
- Queste operazioni possono essere condivise da ogni caso d'uso al quale un certo oggetto partecipa

Considerazioni

- La definizione di oggetto che è condiviso tra due o più casi d'uso dovrebbe essere identica
- Se un'operazione appare in più di un diagramma di sequenza, il suo comportamento dovrebbe essere lo stesso
- Condividere le operazioni tra casi d'uso permette agli sviluppatori di rimuovere le ridondanze nelle specifiche dei requisiti e di migliorare la loro consistenza
- Frammentare il comportamento tra molte operazioni complica immancabilmente le specifiche dei requisiti
- Nella fase di analisi un sequence diagram è usato per aiutare a identificare nuovi oggetti partecipanti comportamenti mancanti.