

OOP - Introduzione

Introduzione alla Programmazione Orientata agli Oggetti
Andrea Colleoni

Lezione 2 - UML

Introduzione a UML


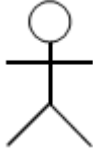


Tipi di diagrammi UML...

- ♦ Use case: usati per definire le caratteristiche funzionali di un programma nella prospettiva del committente; usati nella fase di analisi dei requisiti
- ♦ Activity diagrams: usati per descrivere i flussi di lavoro (workflow) dell'applicazione; sono utilizzati nella fase di analisi dei requisiti
- ♦ Interaction diagrams (cfr. Sequence diagrams): mostrano le interazioni che esistono tra gli oggetti dell'applicazione e si riferiscono a specifici casi più che alla situazione generale; sono utilizzati nella fase di progettazione

...Tipi di diagrammi UML

- ♦ Class diagram: consentono di rappresentare le relazioni che esistono tra le classi e definiscono quindi la struttura del sistema; rappresentano anche gli attributi, le operazioni e le associazioni che esistono tra le classi; sono utilizzati nella fase di progettazione
- ♦ State diagrams: descrivono per il singolo oggetto i possibili stati in cui può esistere e i passaggi di stato; sono utilizzati nella fase di progettazione

Casi d'uso e attori

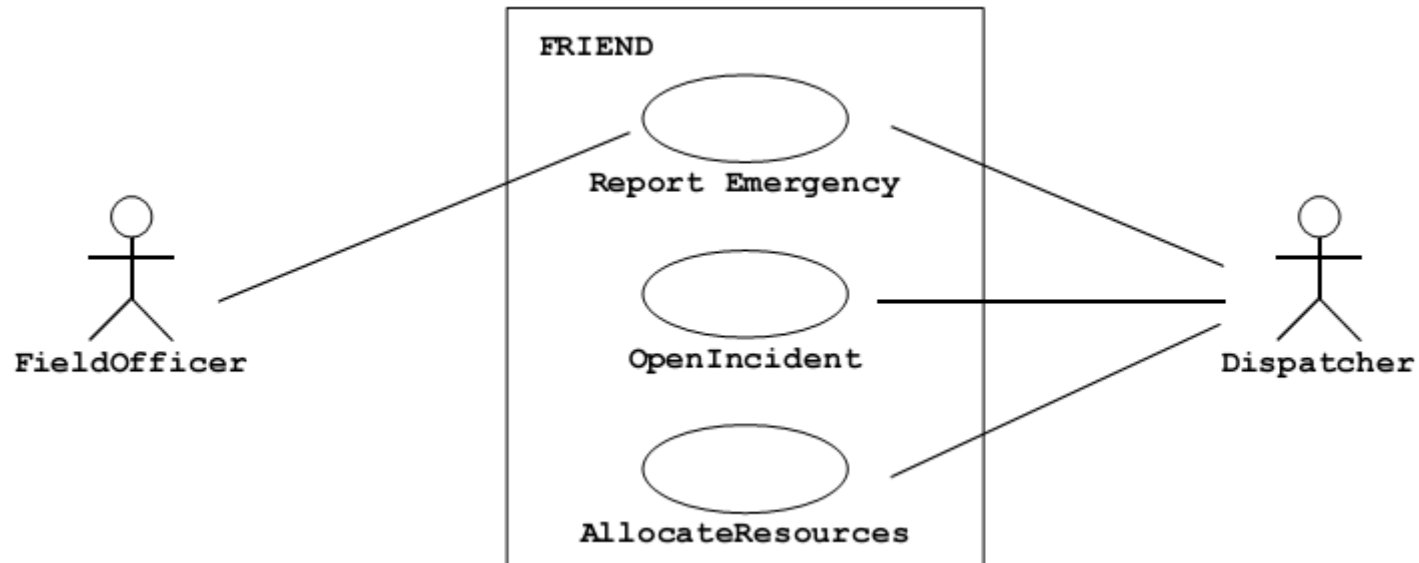
- ♦ Sistema: è il sistema in esame 
- ♦ Attore: è un'entità esterna che interagisce con il sistema;
 - ♦ Uno user role
 - ♦ Un altro sistema
- ♦ Caso d'uso: descrive il comportamento del sistema così come è percepito dal punto di vista di un attore che lo utilizza; è detto anche external bahavior. Rappresenta una funzionalità fornita dal sistema. 
- ♦ Interazione: stabilisce una relazione tra attore e caso d'uso 

Definizione di un caso d'uso

- ♦ **Nome:** deve essere univoco
- ♦ **Attori** partecipanti: l'elenco degli attori coinvolti
- ♦ **Prerequisiti:** descrizione delle condizioni che devono essere soddisfatte perché il caso possa iniziare
- ♦ **Flusso degli eventi:** descrizione di come si svolgono le attività del caso d'uso. Solitamente per descrivere i casi comuni e i casi eccezionali si fanno casi d'uso diversi
- ♦ **Condizioni di uscita:** descrizione delle condizioni che vengono soddisfatte al termine del caso d'uso
- ♦ **Requisiti speciali:** sono requisiti che non hanno relazione con la funzionalità del sistema (performance attese, struttura dell'HW del sistema, ecc)

Use case: esempio...

- ♦ Chiamata ad un servizio di emergenza



...Use case: esempio

<i>Use case name</i>	ReportEmergency
<i>Participating actor</i>	Invoked by FieldOfficer Communicates with Dispatcher
<i>Entry condition</i>	1. The FieldOfficer activates the “Report Emergency” function of her terminal. FRIEND responds by presenting a form to the officer.
<i>Flow of events</i>	2. The FieldOfficer fills the form by selecting the emergency level, type, location, and brief description of the situation. The FieldOfficer also describes possible responses to the emergency situation. Once the form is completed, the FieldOfficer submits the form, at which point the Dispatcher is notified. 3. The Dispatcher reviews the submitted information and creates an Incident in the database by invoking the OpenIncident use case. The Dispatcher selects a response and acknowledges the emergency report.
<i>Exit condition</i>	4. The FieldOfficer receives the acknowledgment and the selected response.
<i>Special requirements</i>	The FieldOfficer’s report is acknowledged within 30 seconds. The selected response arrives no later than 30 seconds after it is sent by the Dispatcher.

Scenari

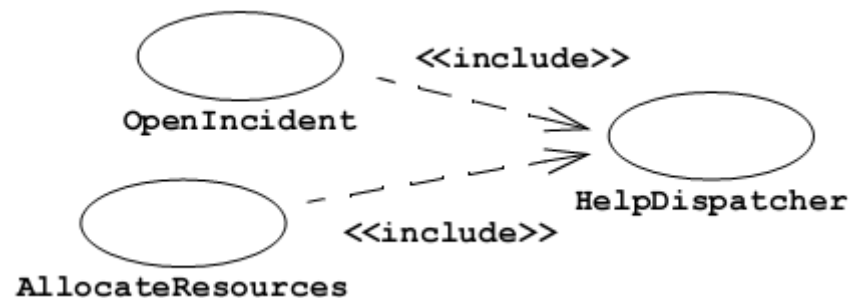
- ♦ Sono esempi concreti di use case; mentre l'use case descrive astrattamente la funzionalità, lo scenario ne fa un esempio.
- ♦ Nella notazione il nome dello scenario è sottolineato ad indicare che è l'istanza di un concetto astratto
- ♦ Anche i nomi degli attori sono sottolineati ad indicare che anch'essi sono istanze di attori astratti

Scenario: esempio

<i>Scenario name</i>	<u>warehouseOnFire</u>
<i>Participating actor instances</i>	<u>bob, alice: FieldOfficer</u> <u>john: Dispatcher</u>
<i>Flow of events</i>	<ol style="list-style-type: none">1. Bob, driving down main street in his patrol car, notices smoke coming out of a warehouse. His partner, Alice, activates the “Report Emergency” function from her FRIEND laptop.2. Alice enters the address of the building, a brief description of its location (i.e., northwest corner), and an emergency level. In addition to a fire unit, she requests several paramedic units on the scene given that area appears to be relatively busy. She confirms her input and waits for an acknowledgment.3. John, the Dispatcher, is alerted to the emergency by a beep of his workstation. He reviews the information submitted by Alice and acknowledges the report. He allocates a fire unit and two paramedic units to the Incident site and sends their estimated arrival time (ETA) to Alice.4. Alice receives the acknowledgment and the ETA.

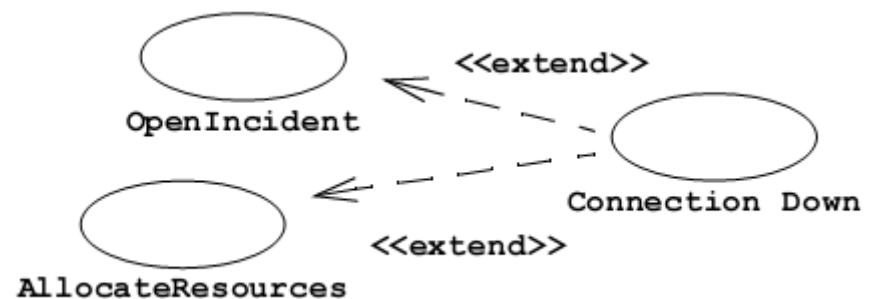
Inclusione

- ♦ È possibile indicare l'inclusione di un caso d'uso (che presumibilmente contiene una funzionalità comune a molti casi d'uso) in un altro:



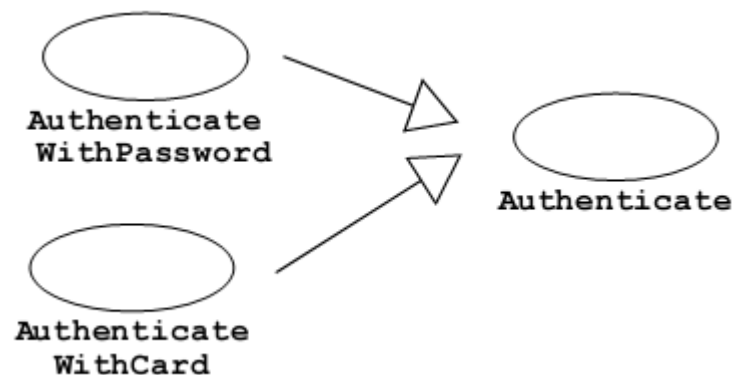
Estensione

- ♦ Per estendere la funzionalità di un caso è possibile aggiungere un nuovo caso che fa allo scopo; ad esempio in caso di interruzione della comunicazione possiamo gestire il caso d'uso eccezionale ConnectionDown; esso ha effetto su due casi d'uso estendendoli



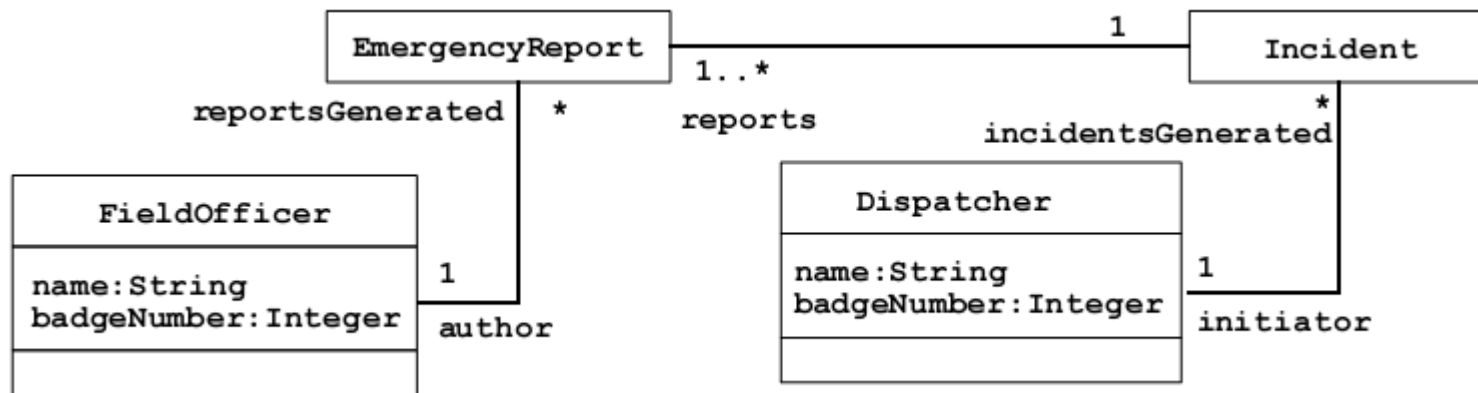
Generalizzazione

- ♦ Per ridurre la complessità del modello, è possibile iniziare da casi più generali e man mano dettagliarne le funzionalità. Supponiamo che durante la raccolta dei requisiti abbiamo descritto nei casi d'uso che l'operatore debba autenticarsi prima di poter operare e abbiamo chiamato il caso d'uso Authenticate; nell'analisi di dettaglio dobbiamo specificare che esistono vari modi per autenticarsi:



Class diagrams

- ♦ Rappresentano classi e oggetti
- ♦ Ne includono la struttura e le relazioni
- ♦ Possono essere disegnati in forma compatta
- ♦ Seguono le naming conventions (classi in maiuscolo, oggetti in minuscolo, ecc)

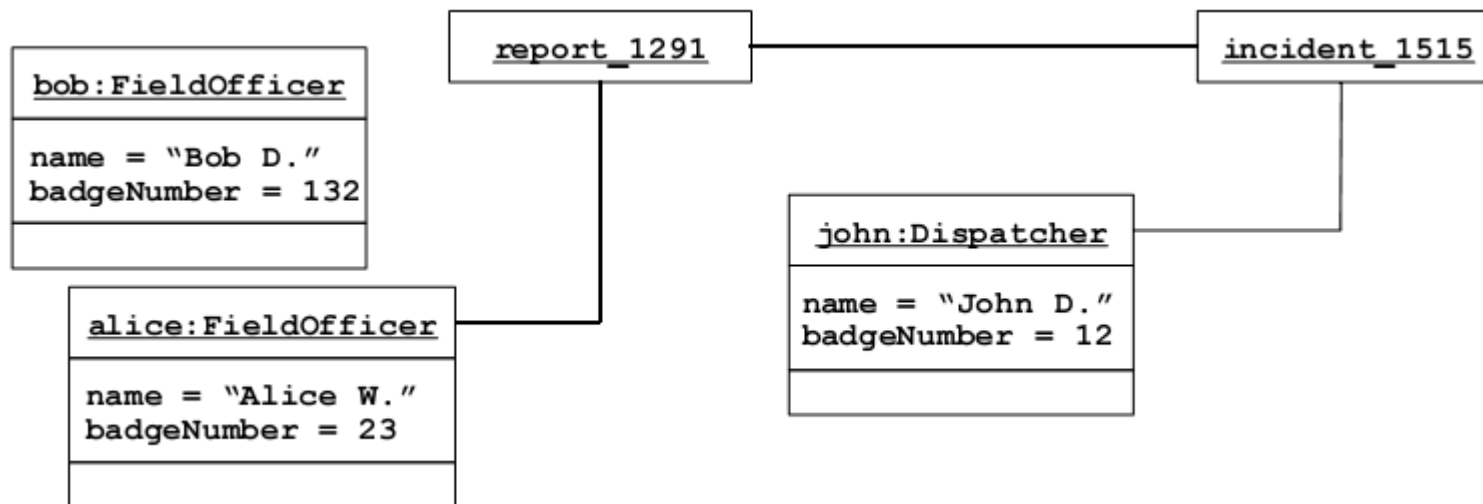


Attributi e operazioni

- ♦ La classe è rappresentata come un rettangolo diviso in tre parti:
 - ♦ Parte superiore: riporta il nome della classe
 - ♦ Parte centrale: riporta gli attributi
 - ♦ Parte inferiore: riporta le operazioni
- ♦ La visibilità di attributi ed operazioni è indicata dai tre simboli seguenti:
 - ♦ +: public
 - ♦ #: protected
 - ♦ - private

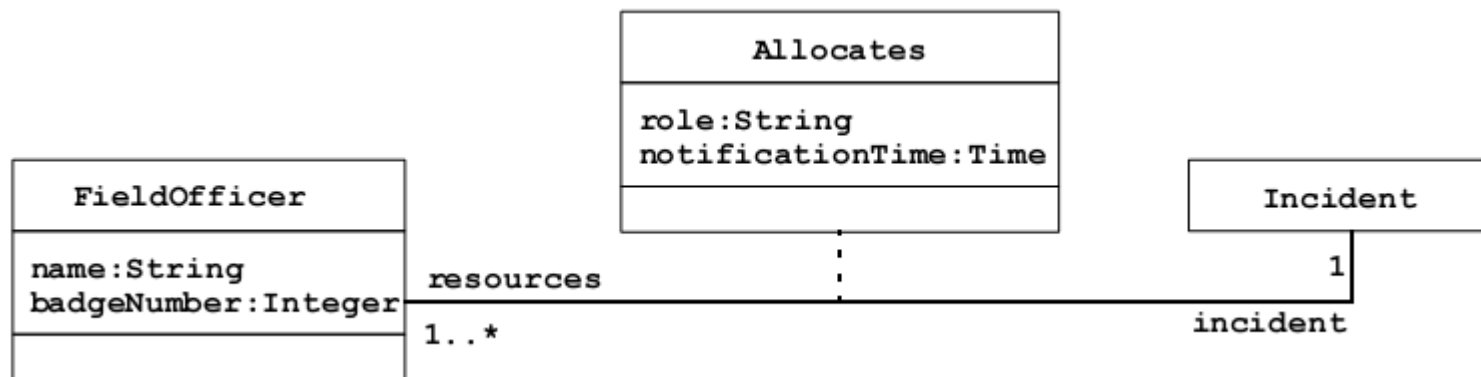
Collegamenti e associazioni...

- ♦ Un collegamento rappresenta una connessione tra due oggetti
- ♦ Un'associazione è una relazione tra classi e rappresenta un gruppo di collegamenti

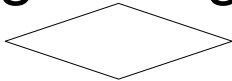

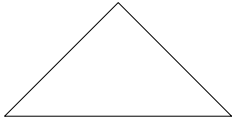


...Collegamenti e associazioni

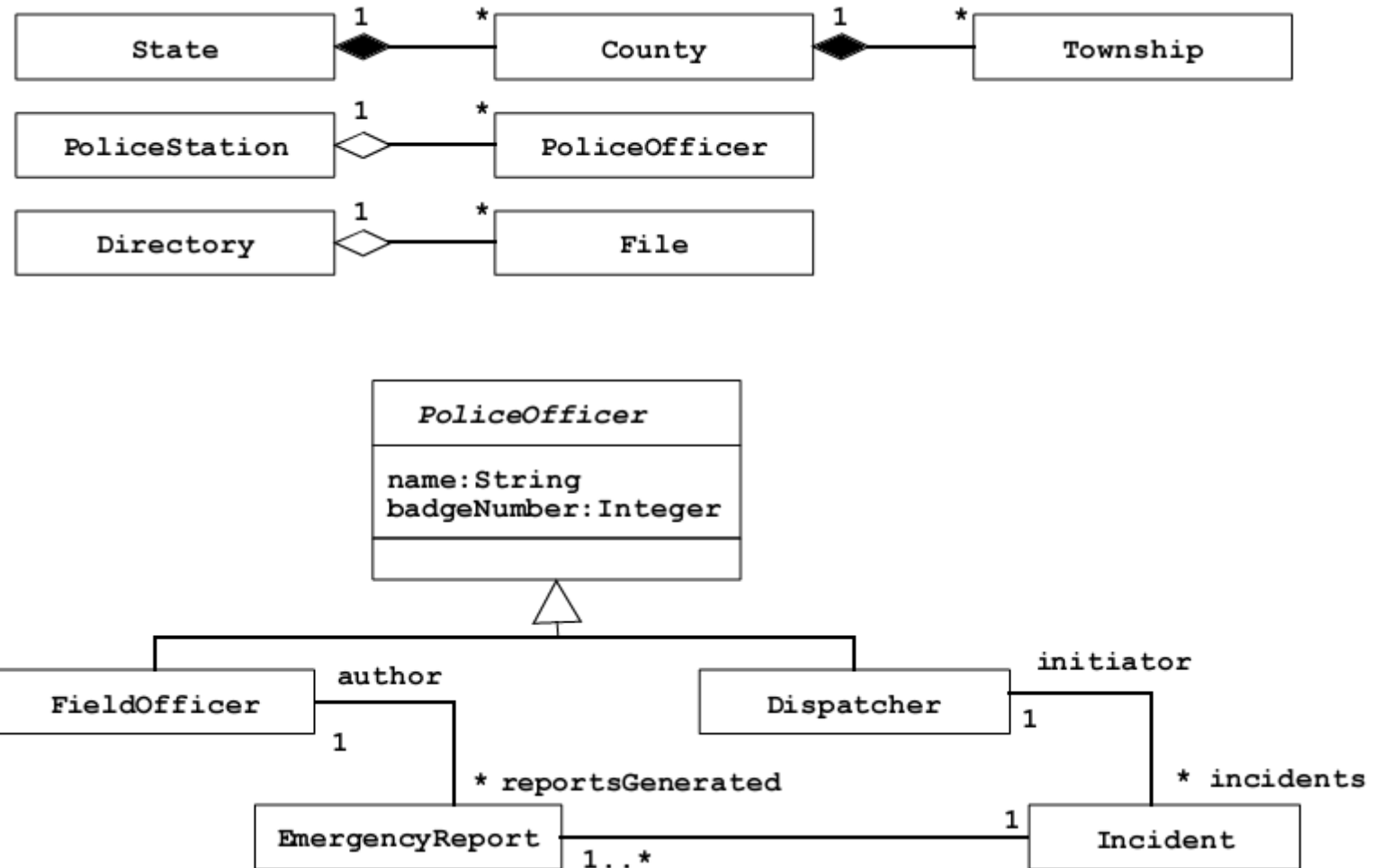
- ♦ Ruolo: ogni estremo di un'associazione può essere etichettato con un testo
- ♦ Molteplicità: ogni estremo di un'associazione può essere etichettato con un numero che ne rappresenta la numerosità nell'associazione
- ♦ Classe dell'associazione: l'associazione può essere rappresentata con una classe



Aggregazione e generalizzazione...

- ◆ Aggregazione: rappresenta il concetto “ha” ma non “è composto da...”; ad esempio Stazione dei Vigili > Vigile del fuoco; si rappresenta con un rombo vuoto 
- ◆ Composizione: rappresenta il concetto “ha” nel senso di “è composto da...”; ad esempio Regione > Provincia > Comune; si rappresenta con un rombo pieno 
- ◆ Generalizzazione: rappresenta il concetto di ereditarietà. La classe più generale è chiamata **superclasse**, quella più specializzata **sottoclasse**; come nei casi d'uso si rappresenta graficamente con un triangolo. Le classi astratte e le interfacce sono rappresentate con il nome in corsivo e non producono oggetti 

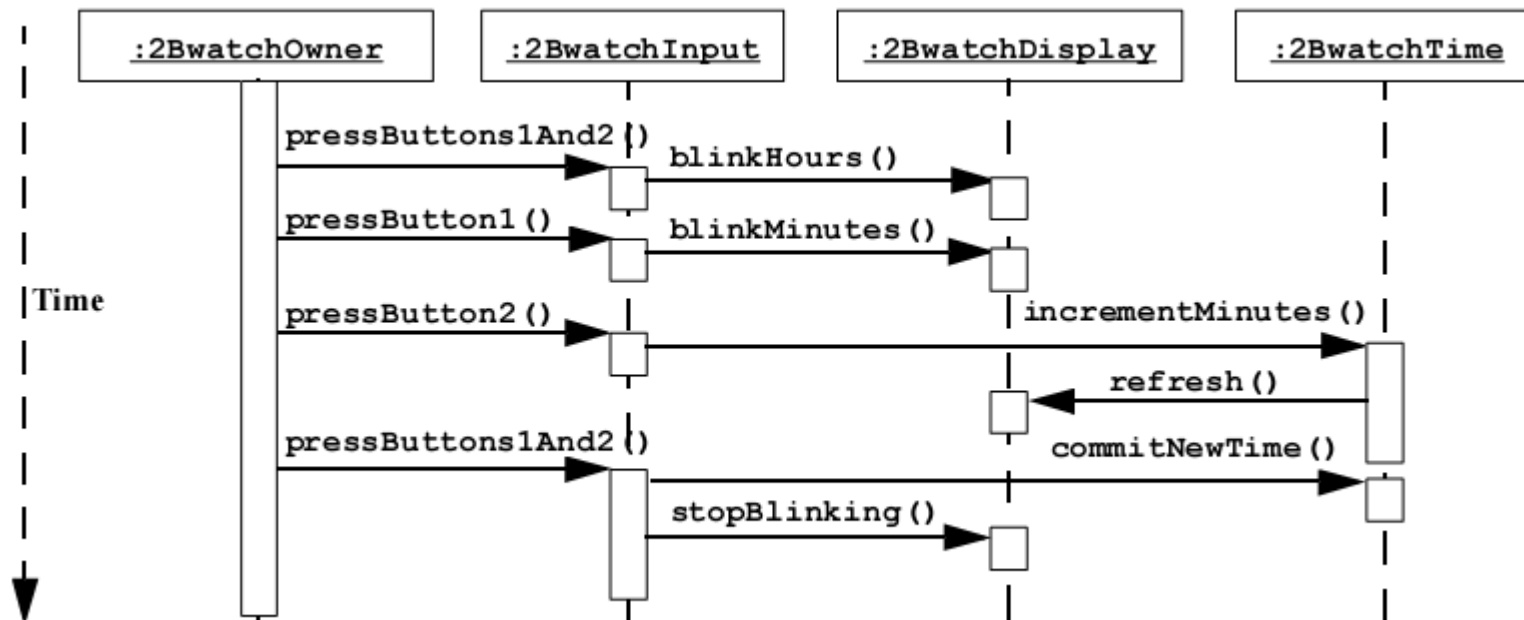
...Aggregazione e generalizzazione



Sequence diagrams

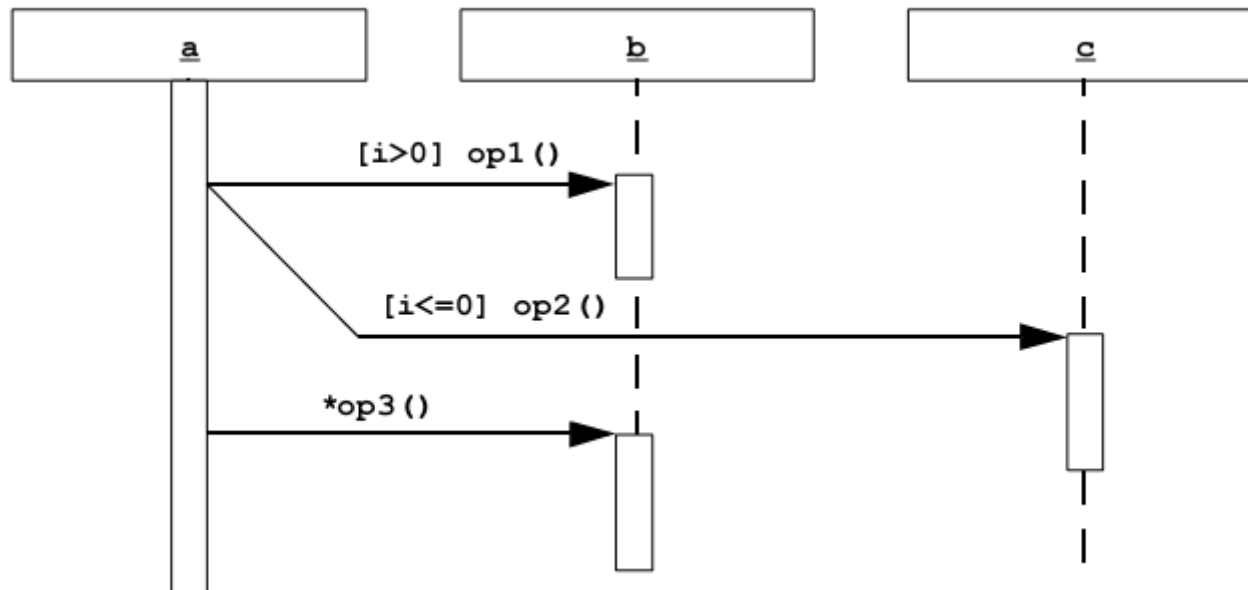
- ◆ Descrivono modelli di comunicazione tra oggetti
- ◆ Un oggetto interagisce con un altro scambiando **messaggi**
- ◆ I messaggi possono trasportare con se anche **argomenti**

Sequence diagram: esempio del cambio di orario su un orologio



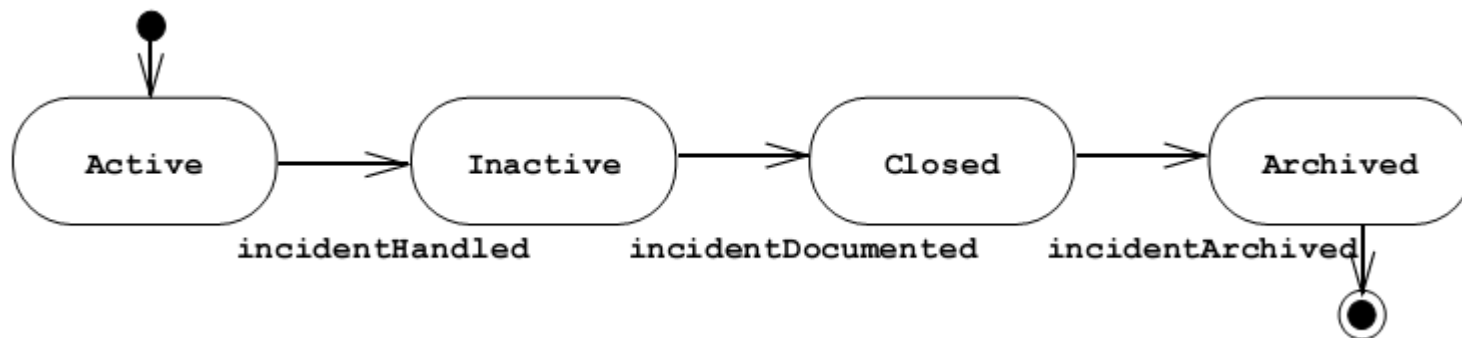
Sequence diagram generici

- ◆ Descrivono tutte le possibili interazioni tra oggetti
- ◆ Includono la notazione per le condizioni di invio di un messaggio [] e per la ripetitività *



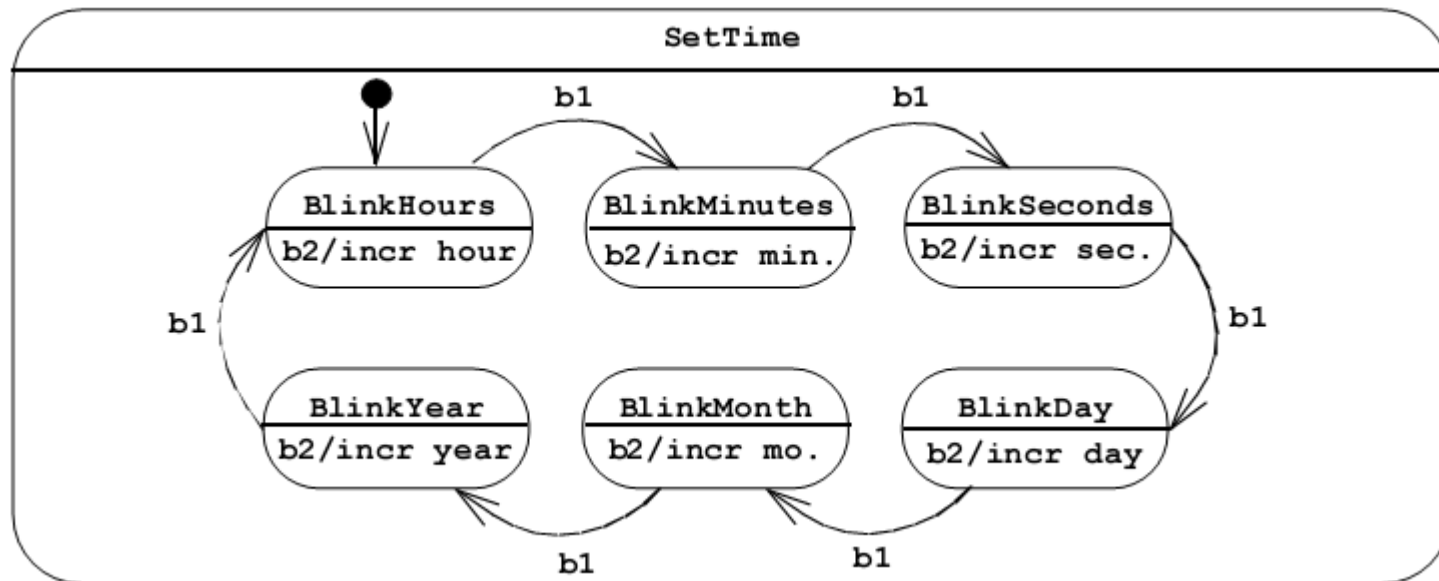
Statechart diagrams...

- ♦ Servono per descrivere i possibili stati in cui un oggetto si può trovare in risposta ad eventi esterni (come le macchine a stati)
- ♦ Uno stato è una condizione che un oggetto può soddisfare
- ♦ Un esempio per l'oggetto Incident

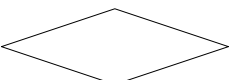



...Statechart diagrams

- ♦ Una transizione rappresenta il passaggio di stato —————>
- ♦ Lo stato iniziale e lo stato finale definiscono i limiti del diagramma
- ♦ La visione di dettaglio di uno stato può essere realizzata con un nuovo diagramma o riportando una descrizione delle transizioni interne, con le relative azioni
- ♦ Uno stato che ha transizioni interne effettua la transizione esterna al completamento di una di quelle interne
- ♦ Uno stato di questo tipo è detto **action state**



Activity diagrams

- ♦ Rappresentano un concetto simile allo statechart, ma nel diagramma tutti gli stati sono action states e le transizioni rappresentano un vincolo di precedenza
- ♦ Possono includere nodi decisionali 
- ♦ Possono includere transizioni complesse, le quali denotano la sincronizzazione di attività multiple 
- ♦ Le azioni possono essere raggruppate in
- ♦ **swimlanes** per indicare l'oggetto o il sottosistema che le implementa

Activity diagram: esempi

