

Analisi

Ingegneria del Software

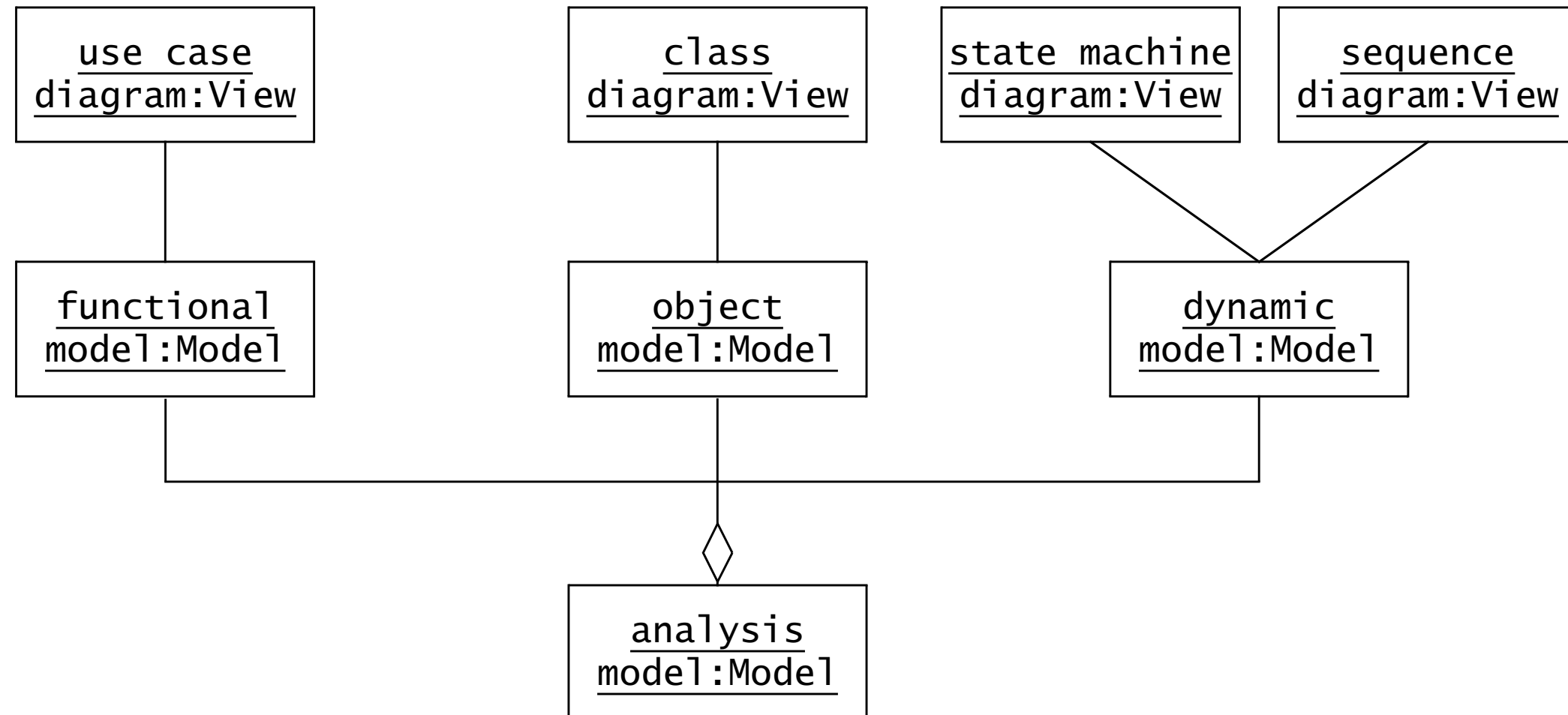


Figure 5-3 The analysis model is composed of the functional model, the object model, and the dynamic model. In UML, the functional model is represented with use case diagrams, the object model with class diagrams, and the dynamic model with state machine and sequence diagrams.

Analisi: concetti di base

- Analysis Object Models and Dynamic Models
- Entity, Boundary, and Control Objects
- Generalization and Specialization

Analysis Object Model

- Modello degli oggetti di analisi:
 - Parte del modello di analisi
 - si focalizza sui concetti individuali che sono manipolati dal sistema
 - proprietà
 - relazioni
- In UML è disegnato tramite il diagramma delle classi
 - Include classi, attributi, operazioni

Dynamic Model

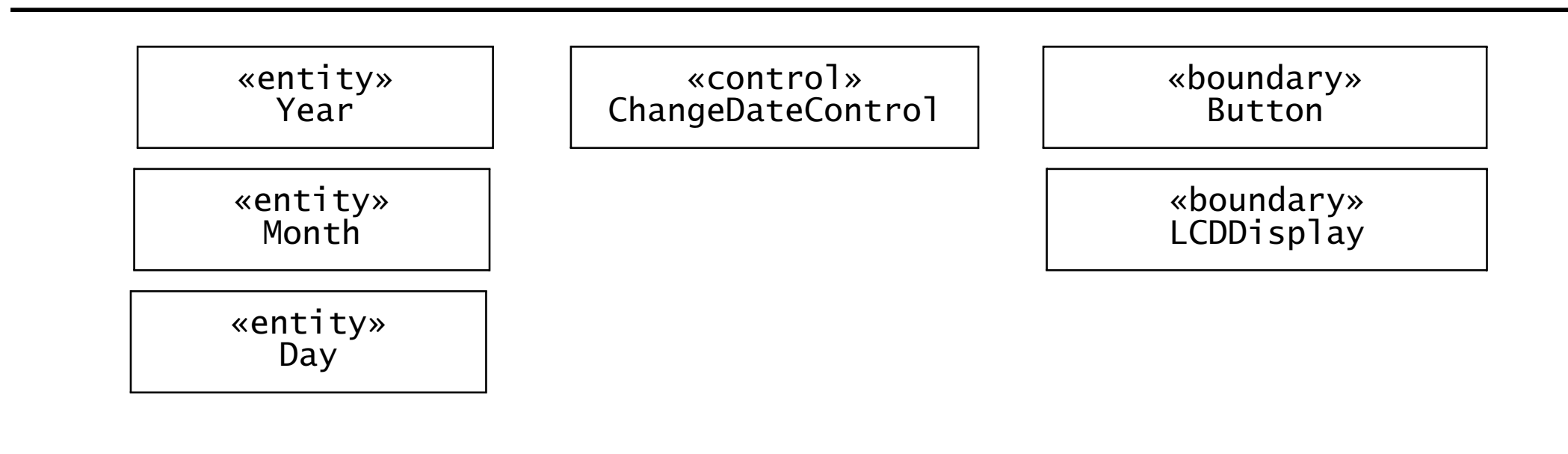
- Il modello dinamico si focalizza sul comportamento del sistema
- È disegnato tramite *sequence diagram* e *state machine diagram*
- Il *sequence diagram* rappresenta le interazioni tra insiemi di oggetti durante un singolo caso d'uso
- Lo State machine diagram rappresenta il comportamento di un singolo oggetto (o di un gruppo di oggetti molto correlati tra di loro)
- Il modello dinamico serve per assegnare le responsabilità alle singole classi
- Nel processo di progettazione serve per identificare nuove classi, associazioni e attributi da aggiungere al modello degli oggetti

Entity, Boundary e Control Object

- Il modello degli oggetti è composto da oggetti:
 - entity - rappresentano le informazioni persistenti tracciate dal sistema
 - boundary - rappresentano le interazioni tra attori e il sistema
 - control - sono responsabili di realizzare attivamente i casi d'uso
- La modellazione del sistema con entity, boundary e control fornisce agli sviluppatori le euristiche per distinguere concetti differenti ma correlati

Entity, Boundary e Control Object

- Per distinguere tra differenti tipi di oggetti:
- UML fornisce il meccanismo degli stereotipi
- Permette di abbinare delle meta-informazioni agli elementi di modellazione



Generalization e Specialization

- Generalizzazione: è l'attività di modellazione che identifica concetti astratti da un livello più basso
- Specializzazione: all'attività di modellazione che identifica concetti più specifici da un livello più alto

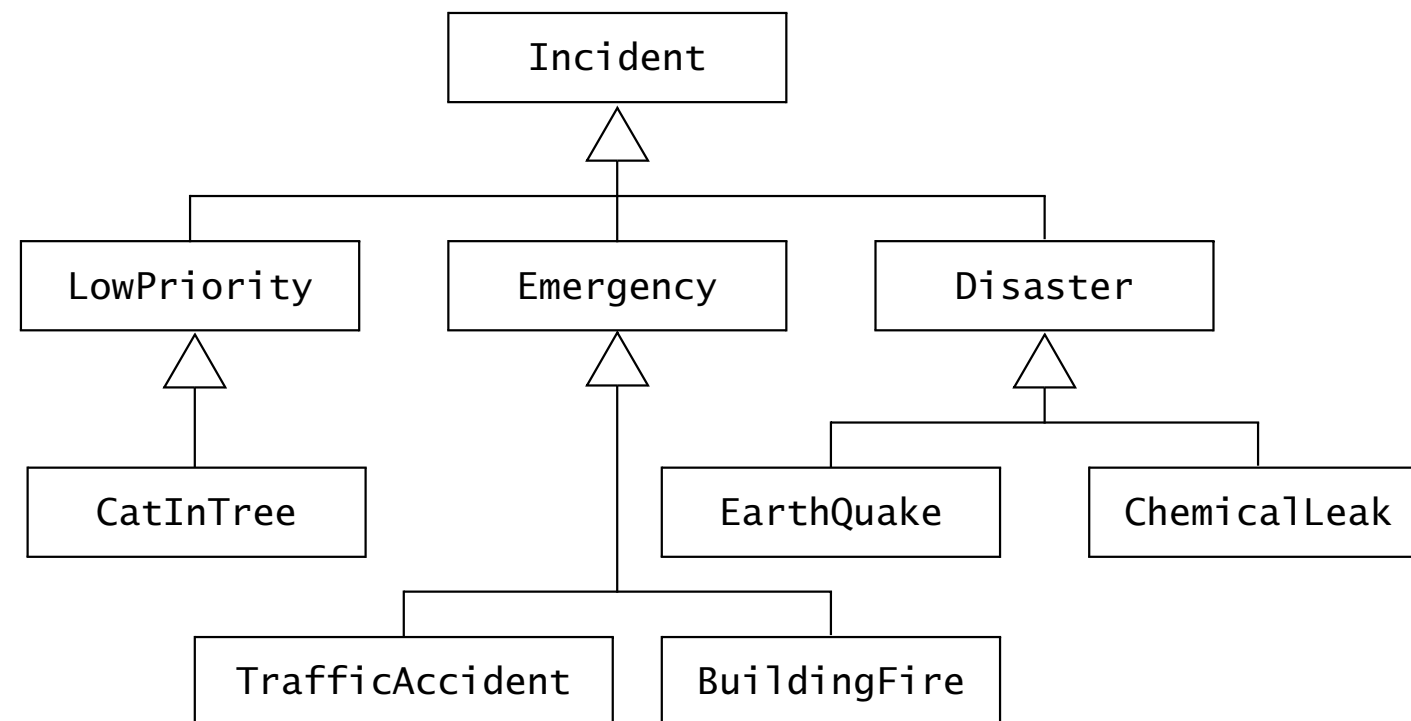
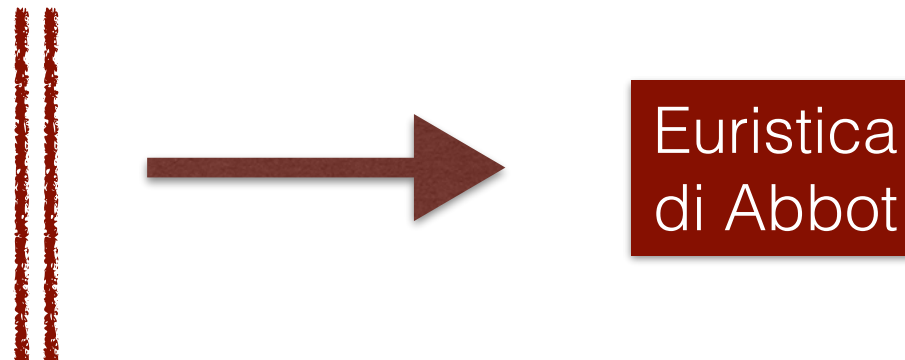


Figure 5-6 An example of a generalization hierarchy (UML class diagram). The top of the hierarchy represents the most general concept, whereas the bottom nodes represent the most specialized concepts.

Attività di analisi

- Identificare gli oggetti entità
 - Identificare gli oggetti boundary
 - Identificare gli oggetti control
- 
- Euristica di Abbot
- Mappare i casi d'uso sugli Oggetti attraverso Sequence Diagram
 - Modellare le interazioni tra gli Oggetti attraverso CRC card (class, responsibilities and collaborations)
 - Identificare le associazioni
 - identificare le aggregazioni
 - Identificare gli attributi
 - Modellare le relazioni di Ereditarietà
 - Modellare il comportamento State-Dependent di ogni Oggetto



Abbot's heuristic



Dai casi d'uso agli Object

- A partire dai casi d'uso si identificano i possibili *object*
- *Analisi del linguaggio naturale*
- Un set di euristiche intuitive per identificare oggetti attributi e associazioni a partire dalla specifica dei requisiti
- L'euristica mappa una parte del linguaggio
- Per esempio: nomi, verbi avere, verbi essere, aggettivi

Table 5-1 Abbott’s heuristics for mapping parts of speech to model components [Abbott, 1983].

Part of speech	Model component	Examples
Proper noun	Instance	Alice
Common noun	Class	Field officer
Doing verb	Operation	Creates, submits, selects
Being verb	Inheritance	Is a kind of, is one of either
Having verb	Aggregation	Has, consists of, includes
Modal verb	Constraints	Must be
Adjective	Attribute	Incident description

<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>	<p>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.</p> <p>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.</p> <p>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.</p>
<i>Exit condition</i>	5. The FieldOfficer receives the acknowledgment and the selected response.

Figure 5-7 An example of use case, ReportEmergency (one-column format).

Heuristics for identifying entity objects

- Terms that developers or users need to clarify in order to understand the use case
- Recurring nouns in the use cases (e.g., Incident)
- Real-world entities that the system needs to track (e.g., FieldOfficer, Dispatcher, Resource)
- Real-world activities that the system needs to track (e.g., EmergencyOperationsPlan)
- Data sources or sinks (e.g., Printer).

Dai casi d'uso agli Object

- L'analisi del linguaggio naturale ha il vantaggio di focalizzarsi sui termini dell'utente
- Ha lo svantaggio di essere molto legato allo stile di scrittura dell'analista
- Il linguaggio naturale è di per sé un mezzo impreciso
- Il modello degli oggetti rischia di essere impreciso
- Per ovviare a questo inconveniente l'analista o lo sviluppatore devono rifrasare, chiarificare le specifiche dei requisiti

Dai casi d'uso agli Object

- Altro svantaggio del linguaggio naturale: ci sono molti più nomi rispetto alle classi che poi risultano essere rilevanti
- Sinonimi o attributi
- In genere l'euristica di Abbott funziona bene per generare una lista iniziale di oggetti “candidati”
- Descrivere brevemente gli oggetti permette allo sviluppatore di chiarire i concetti usati ed evitare misunderstanding

Una prima ipotesi di Entity in base all'esempio precedente

Table 5-2 Entity objects for the ReportEmergency use case.

Dispatcher	Police officer who manages Incidents. A Dispatcher opens, documents, and closes Incidents in response to Emergency Reports and other communication with FieldOfficers. Dispatchers are identified by badge numbers.
EmergencyReport	Initial report about an Incident from a FieldOfficer to a Dispatcher. An EmergencyReport usually triggers the creation of an Incident by the Dispatcher. An EmergencyReport is composed of an emergency level, a type (fire, road accident, other), a location, and a description.
FieldOfficer	Police or fire officer on duty. A FieldOfficer can be allocated to, at most, one Incident at a time. FieldOfficers are identified by badge numbers.
Incident	Situation requiring attention from a FieldOfficer. An Incident may be reported in the system by a FieldOfficer or anybody else external to the system. An Incident is composed of a description, a response, a status (open, closed, documented), a location, and a number of FieldOfficers.

Identificare gli oggetti Boundary

- Gli oggetti boundary rappresentano l'interfaccia del sistema con gli attori
- In ogni caso d'uso ogni attore interagisce almeno con un oggetto boundary
- Gli oggetti boundary collezionano le informazioni dagli attori e li traducono in un form che può essere usato sia dagli oggetti entity che dagli oggetti control
- Gli oggetti boundary modellano ^{anche} interfacce utente
- Comunque NON descrivono in dettaglio l'aspetto visuale o visivo delle interfacce utente

Identificare gli oggetti Boundary

- Per esempio un menu o una barra di scorrimento sono troppo dettagliate
- Lo sviluppatore può discutere le interfacce utente attraverso quelli che si chiamano mock-up
- I mock-up possono poi essere definiti nelle interfacce grafiche finali

Euristica per identificare gli oggetti Boundary

Heuristics for identifying boundary objects

- Identify user interface controls that the user needs to initiate the use case (e.g., `ReportEmergencyButton`).
- Identify forms the users needs to enter data into the system (e.g., `EmergencyReportForm`).
- Identify notices and messages the system uses to respond to the user (e.g., `AcknowledgmentNotice`).
- When multiple actors are involved in a use case, identify actor terminals (e.g., `DispatcherStation`) to refer to the user interface under consideration.
- Do not model the visual aspects of the interface with boundary objects (user mock-ups are better suited for that).
- *Always* use the end user's terms for describing interfaces; do not use terms from the solution or implementation domains.



Euristica per identificare gli oggetti Boundary

Table 5-3 Boundary objects for the ReportEmergency use case.

AcknowledgmentNotice	Notice used for displaying the Dispatcher's acknowledgment to the FieldOfficer.
DispatcherStation	Computer used by the Dispatcher.
ReportEmergencyButton	Button used by a FieldOfficer to initiate the ReportEmergency use case.
EmergencyReportForm	Form used for the input of the ReportEmergency. This form is presented to the FieldOfficer on the FieldOfficerStation when the "Report Emergency" function is selected. The EmergencyReportForm contains fields for specifying all attributes of an emergency report and a button (or other control) for submitting the completed form.
FieldOfficerStation	Mobile computer used by the FieldOfficer.
IncidentForm	Form used for the creation of Incidents. This form is presented to the Dispatcher on the DispatcherStation when the EmergencyReport is received. The Dispatcher also uses this form to allocate resources and to acknowledge the FieldOfficer's report.

Identificare gli oggetti Control

- Gli oggetti Control sono responsabili del coordinamento tra gli oggetti boundary ed entity
- Di solito gli oggetti Control non hanno un corrispettivo concreto nel mondo reale
- Esiste spesso una relazione molto stretta tra un caso d'uso ed un oggetto Control
- In genere un oggetto Control creato all'inizio del caso d'uso cessa di esistere alla sua fine
- È responsabile di collezionare informazioni dagli oggetti boundary e inviarli agli oggetti entity

Euristica per identificare gli oggetti Control

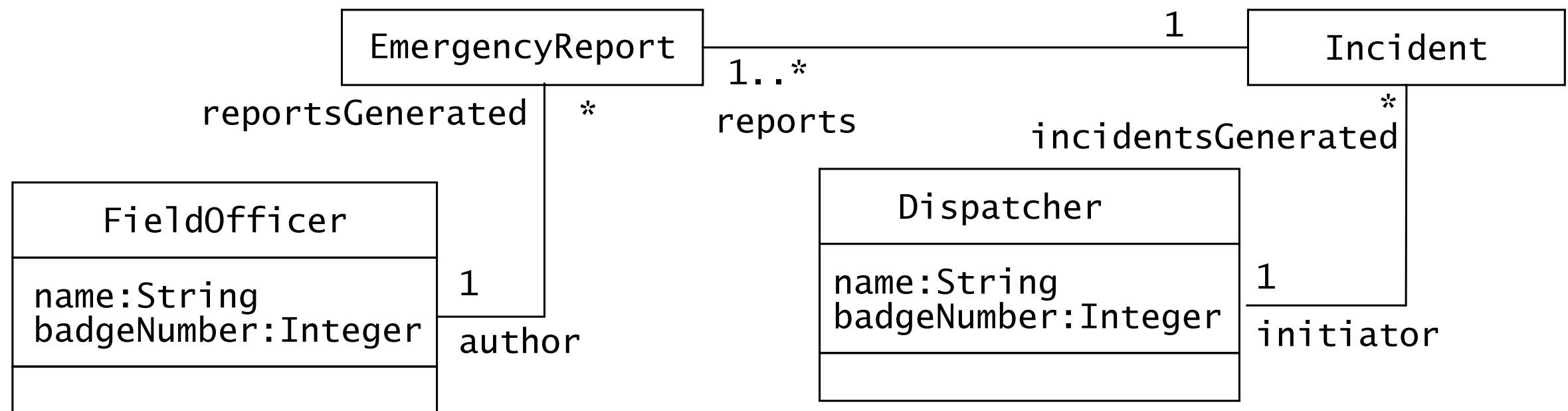
Heuristics for identifying control objects

- Identify one control object per use case.
- Identify one control object per actor in the use case.
- The life span of a control object should cover the extent of the use case or the extent of a user session. If it is difficult to identify the beginning and the end of a control object activation, the corresponding use case probably does not have well-defined entry and exit conditions.

Table 5-4 Control objects for the ReportEmergency use case.

ReportEmergencyControl	Manages the ReportEmergency reporting function on the FieldOfficerStation. This object is created when the FieldOfficer selects the “Report Emergency” button. It then creates an EmergencyReportForm and presents it to the FieldOfficer. After submitting the form, this object then collects the information from the form, creates an EmergencyReport, and forwards it to the Dispatcher. The control object then waits for an acknowledgment to come back from the DispatcherStation. When the acknowledgment is received, the ReportEmergencyControl object creates an AcknowledgmentNotice and displays it to the FieldOfficer.
ManageEmergencyControl	Manages the ReportEmergency reporting function on the DispatcherStation. This object is created when an EmergencyReport is received. It then creates an IncidentForm and displays it to the Dispatcher. Once the Dispatcher has created an Incident, allocated Resources, and submitted an acknowledgment, ManageEmergencyControl forwards the acknowledgment to the FieldOfficerStation.

esempio di diagramma delle classi



Mappare gli oggetti sui sequence

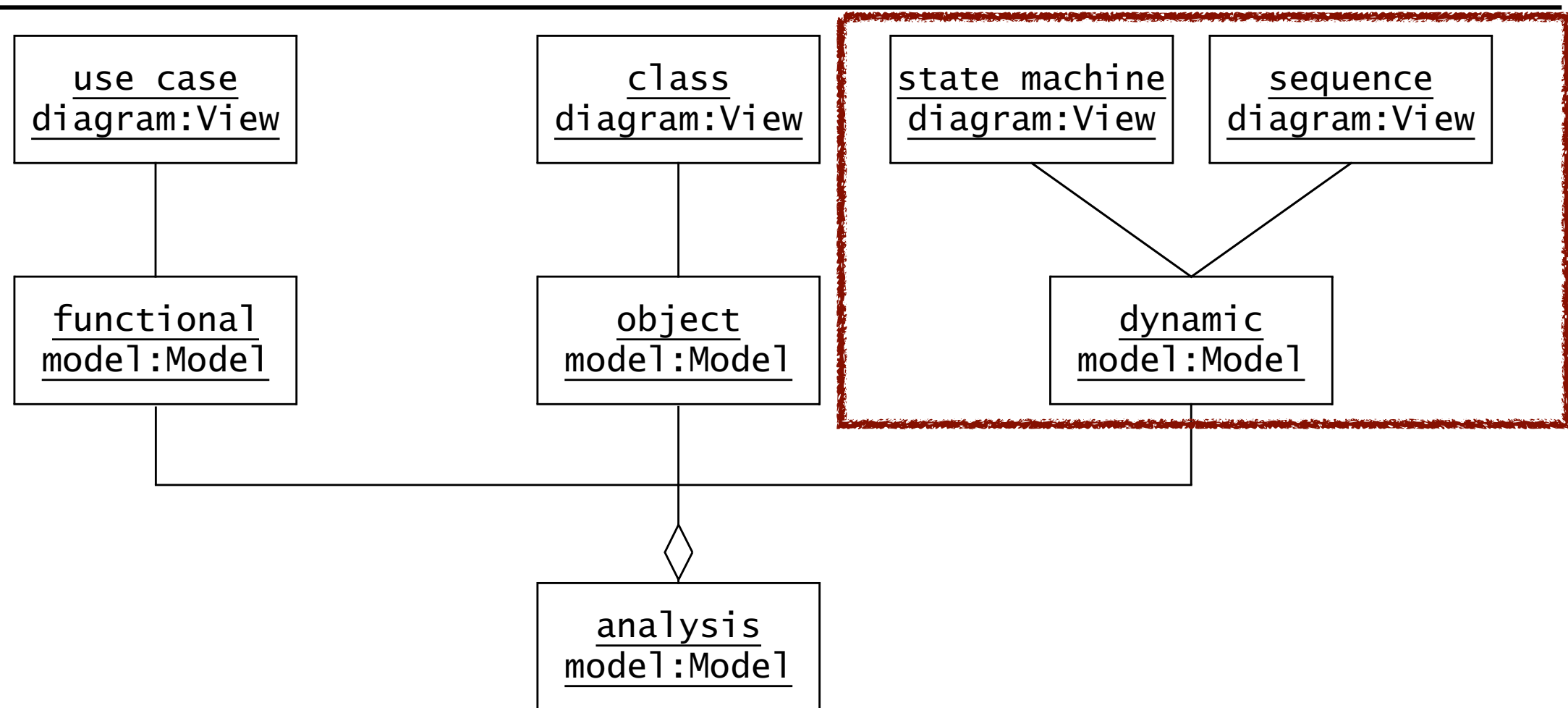


Figure 5-3 The analysis model is composed of the functional model, the object model, and the dynamic model. In UML, the functional model is represented with use case diagrams, the object model with class diagrams, and the dynamic model with state machine and sequence diagrams.