

SAPIENZA Università di Roma
Facoltà di Ingegneria - Corso di Laurea in Ingegneria Informatica
Corso di Progettazione del Software A.A. 2009/2010
Prova al calcolatore del 18 febbraio 2011

L'applicazione da progettare riguarda la gestione dei terremoti che possono verificarsi in una data zona. L'applicazione ha lo scopo di eseguire un'insieme di attività, in un dato ordine, le quali hanno lo scopo di:

- dare il primo supporto alle popolazioni colpite,
- salvare le persone intrappolate negli edifici,
- dirigere le persone agli ospedali più vicini,
- effettuare una prima analisi sull'agibilità degli edifici

La zona da supervisionare è divisa in aree, ognuna delle quali contiene un insieme di edifici, alcuni dei quali sono ospedali. Per ogni edificio di ogni area, viene fatto un sopralluogo per valutarne l'agibilità e per determinare il numero di persone che vi sono rimaste intrappolate. Le persone rimaste negli edifici possono essere vive oppure defunte. Una persona viva può essere ferita, nel qual caso viene trasportata in un ospedale per una diagnosi.

Il processo di supervisione prevede che un operatore della Protezione Civile inserisca le informazioni sulle aree colpite; tali informazioni vengono successivamente memorizzate nel sistema. Successivamente, viene gestita l'emergenza nelle diverse aree identificate. Per velocizzare le operazioni, le aree possono essere gestite in parallelo da diversi operatori. Inizialmente l'operatore addetto, non appena è in grado di gestire una certa area, inserisce la lista degli edifici da controllare. Successivamente viene effettuato un sopralluogo per ogni edificio, valutandone l'agibilità, determinando le persone defunte e vive ed, eventualmente, decidendo gli ospedali presso cui ricoverare i feriti. Il capo della Protezione Civile è, inoltre, interessato ad effettuare alcune statistiche: la frazione di defunti (il numero di persone defunte diviso quello di persone presenti negli edifici) e la frazione dei feriti (il numero delle persone ferite diviso il numero complessivo). Una persona è considerata ferita se è stata ricoverata in qualche ospedale.

Il diagramma delle classi è raffigurato in Figura 1, insieme alla descrizione della responsabilità sulle associazioni e gli invarianti da garantire

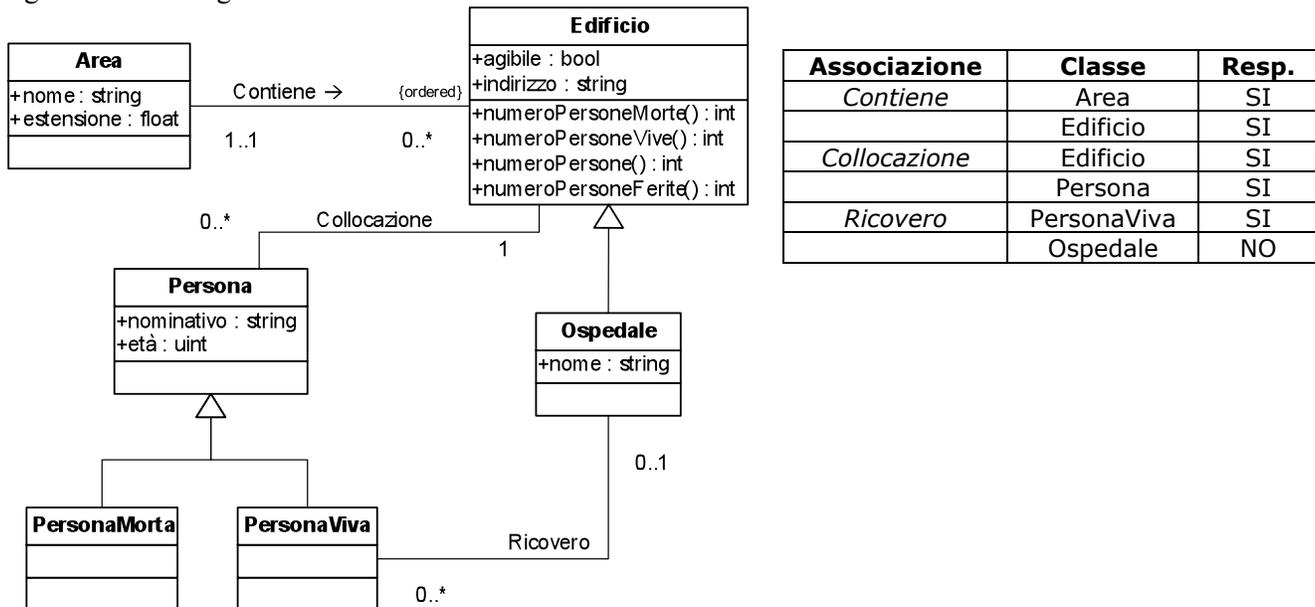
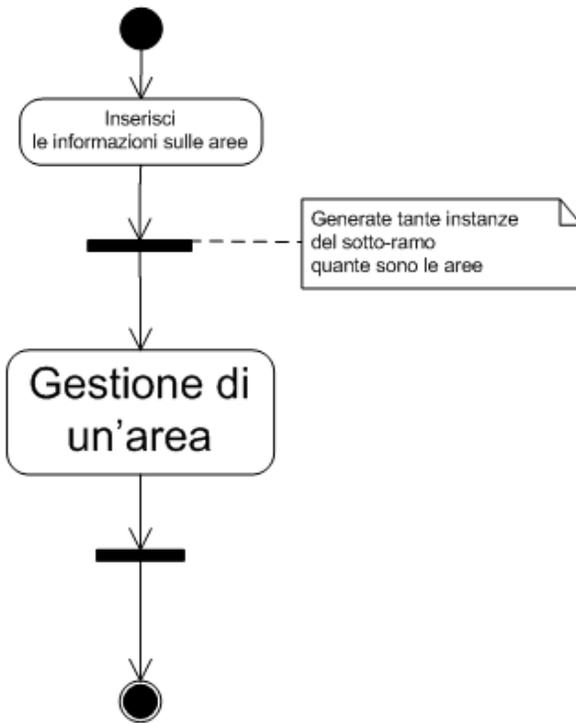


Figura 1: Diagramma delle Classi, Responsabilità



```

VariabiliAttività GestioneEmergenza
  var: aree: List<Area>
Fine
  
```

```

InizioSpecificaAttivitàIO
  operazione: inserisciInfoAree
  in: --
  out: areeIO: List<Area>
  
```

Figura 2: Diagramma dell'attività principale e specifica OCL

I metodi dichiarati nella classe **emergenza.Edificio** possono essere specificati in OCL come segue:

```

InizioSpecificaOperazioniClasse Edificio
Context Edificio::numeroPersoneVive():(int)
pre: --
post: restituisce un numero di persone ritrovate vive nell'edificio
Context Edificio::numeroPersoneMorte():(int)
pre: --
post: restituisce un numero di persone ritrovate morte nell'edificio
Context Edificio::numeroPersone():(int)
pre: --
post: restituisce un numero di persone complessivo ritrovate nell'edificio
Context Edificio::numeroPersoneFerite():(int)
pre: --
post: restituisce un numero di persone ritrovate ferite nell'edificio
  
```

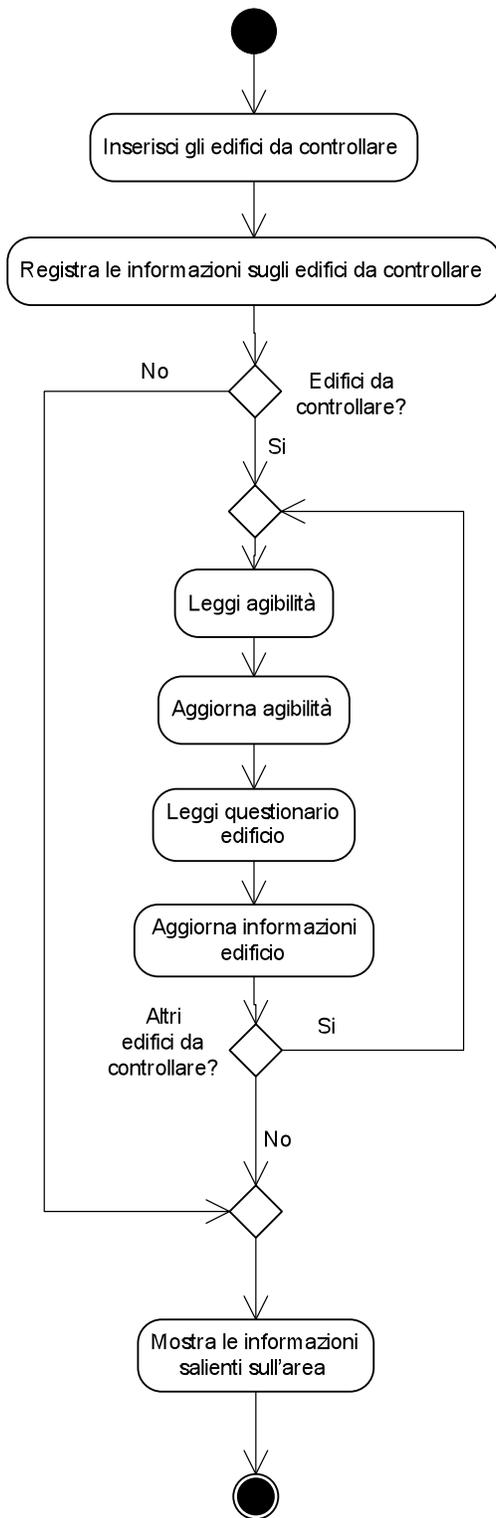
Il diagramma delle attività è rappresentato in Figura 2. L'attività **Gestione di un'area** è composta e il corrispondente diagramma delle attività è rappresentato in Figura 3. Il diagramma delle attività utilizza alcuni tipi di dato:

- **RecordPersona** che definisce quattro proprietà
 - **nominativo: String**
 - **eta: int**
 - **vivo: boolean**
 - **ospedale: Ospedale**
 - La proprietà **ospedale** può valere anche *null* se la persona in questione è morta oppure è viva ma non ferita. Ovviamente non può accadere che esista una persona per cui `vivo=false` e `ospedale!=null`

Tutte le classi che codificano il diagramma delle attività sono attestate nel package **processo**. Le attività di I/O e i task sono codificate in classi che hanno lo stesso nome di quello usato nella specifica OCL. Il diagramma in

Figura 2 è codificato nella classe **processo.GestioneEmergenza** e il sotto-diagramma è codificato nella classe **processo.GestioneArea**.

- **Al fine di modificare il codice fornito per implementare le funzionalità, si suggerisce di intervenire esclusivamente sulle seguenti classi:**
 - **emergenza.PersonaViva**
 - **emergenza.Edificio**
 - **emergenza.ManagerContiene**
 - **emergenza.TipoLinkRicovero**
 - **processo.GestioneArea**
 - **processo.AggiornaAgibilita**
 - **processo.RegistraInfoEdifici**
 - **gui.PrincipaleListener**



Ogni istanza dell'attività GestioneArea prende come input un Area che serve ad identificare l'area su cui si agisce.

VariabiliAttività GestioneArea
var: area: area
var: edifici: List<Edificio>
 Fine

InizioSpecificaAttivitàIO
operazione: inserisciEdifici
in: nomeArea: String
out: edifIO: List<String>
operazione: leggiAgibilita
in: indirizzo: String
out: agibile: boolean
operazione: leggiQuestionario
in: indirizzo: String
out: recordList: List<RecordPersona>
operazione: mostraInfoSalienti
in: recordArea: RecordArea
out: --
 Fine

InizioSpecificaAttivitàAtomica registraInfoEdifici
 (edifici: List<Edificio>, area: Area)
pre:
post: Aggiunge gli edifici nella lista *edifici* all'area *area*.
 Fine

InizioSpecificaAttivitàAtomica aggiornaAgibilita
 (edificio: Edificio, agibilita:boolean)
pre:
post: Aggiorna l'agibilità dell'edificio *edificio* secondo quanto specificato dalla variabile *agibilita*
 Fine

InizioSpecificaAttivitàAtomica aggiornaInfoEdifici
 (edificio: Edificio) (quest: List<RecordPersona>)
pre:
post: associa ogni persona nella lista *quest* ad un edificio e, se tale persona è viva e ferita, associa tale persona all'ospedale.
 Fine

Figura 3: Diagramma dell'attività composta di **Gestione Area**