1506
UNIVERSITÀ
DEGLI STUDI
DI URBINO
CARLO BO

CORSO DI LAUREA IN
INFORMATICA APPLICATA
SCUOLA DI
SCIENZE TECNOLOGIE E FILOSOFIA DELL'INFORMAZIONE

Relazione del progetto d'esame di

PROGRAMMAZIONE E MODELLAZIONE AD OGGETTI

Studente: Kevin Berberolli

Matricola: 290248

Specifica del progetto

Il progetto consiste nella simulazione dello scenario di scarico e ritiro dei bagagli dal nastro trasportatore in un aeroporto, che consente ad un utente (osservatore) di effettuare la registrazione e ricevere notifiche da un provider. Gli osservatori effettueranno la registrazione con il provider e ogni volta che si verifica una condizione o un cambiamento di stato, il provider invia automaticamente una notifica a tutti gli osservatori che può essere di avvenuta ricezione, di errore o di completamento delle notifiche. Il provider può anche fornire informazioni sullo stato corrente degli osservatori.

Studio del problema

Ho deciso di scrivere un programma che esegue da solo, in quanto è una simulazione, in modo che l'utente possa limitarsi a leggere i dati stampati a video (monitor dei bagagli scaricati sul nastro e monitor dei bagagli ritirati dal nastro) e verificare il corretto funzionamento del programma.

Sono stati implementati i seguenti elementi:

- Un provider che corrisponde all'oggetto che invia le notifiche agli osservatori.

- Un osservatore che è un oggetto che riceve le notifiche da un provider.
  L'osservatore deve implementare tre metodi, che vengono tutti chiamati dal provider: un metodo che fornisce all'osservatore informazioni nuove o correnti, uno che informa l'osservatore che si è verificato un errore e infine uno che indica che il provider ha completato l'invio di notifiche.

- Un oggetto lista contenente i dati che il provider invia agli osservatori.

Scelte architetturali

a) Lo schema progettuale osservatore viene usato per implementare un sistema di informazioni per il ritiro dei bagagli in aeroporto. La classe *BaggageInfo* fornisce informazioni sui voli in arrivo e sui nastri in cui possono essere ritirati i bagagli per ogni volo.

b) Una classe *BaggageHandler* è responsabile della ricezione di informazioni sui voli in arrivo e sui nastri per il ritiro dei bagagli. Internamente, gestisce due raccolte:

- *observers*: una raccolta di client che riceveranno informazioni aggiornate.
- *flights*: una raccolta di voli e dei nastri ad essi assegnati.

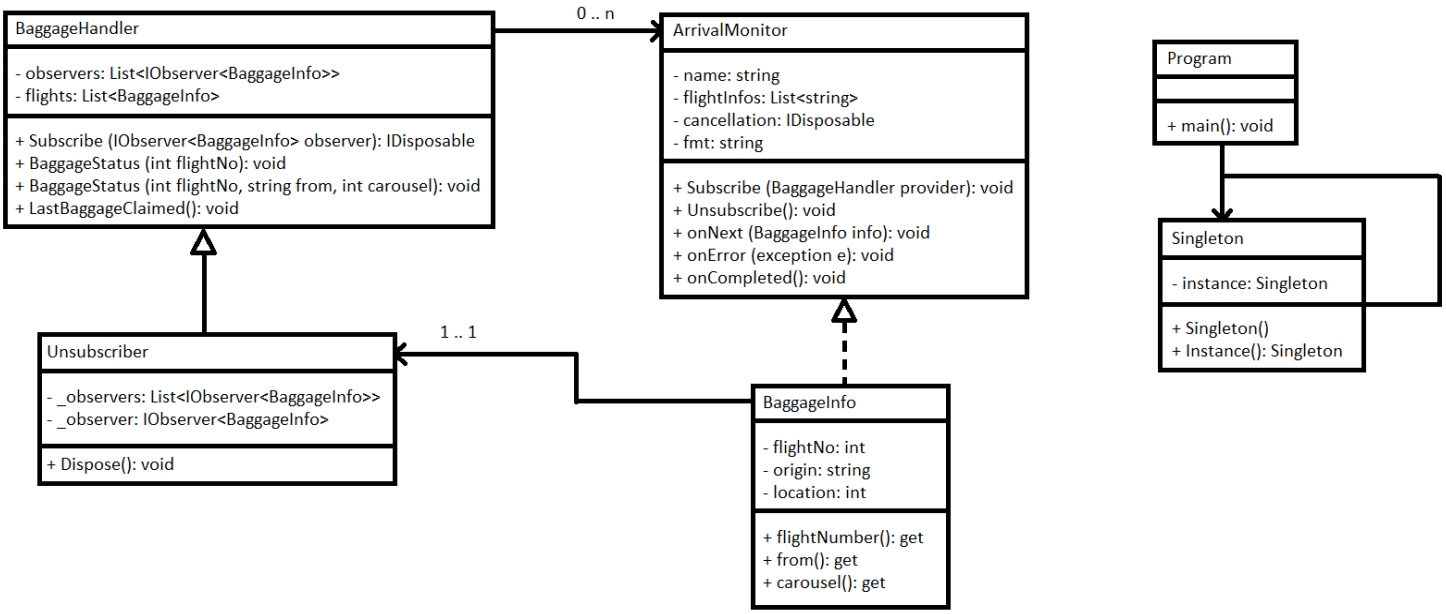
Il metodo *BaggageHandler.BaggageStatus* può essere chiamato per indicare che i bagagli di un volo stanno per essere scaricati o che lo scaricamento è terminato. Nel primo caso, al metodo vengono passati un numero di volo, l'aeroporto di provenienza e il nastro in cui vengono scaricati i bagagli. Nel secondo caso, al metodo viene passato solo un numero di volo. Per i bagagli scaricati, il metodo verifica se le informazioni di *BaggageInfo* passate al metodo sono presenti nella raccolta *flights*. Se non lo sono, il metodo aggiunge le informazioni e chiama il metodo *OnNex*t di ogni osservatore. Per i voli per cui è terminato lo scaricamento dei bagagli, il metodo controlla se le informazioni su tale volo sono archiviate nella raccolta *flights*.

Quando l'ultimo volo è atterrato e i relativi bagagli sono stati gestiti, viene chiamato il metodo *BaggageHandler.LastBaggageClaimed*. Questo metodo chiama il metodo *OnCompleted* di ogni osservatore per indicare che tutte le notifiche sono state completate e cancella la lista degli osservatori.

c) La classe ArrivalsMonitor fornisce informazioni relative al ritiro dei bagagli. Le informazioni vengono visualizzate in base al nome della città di provenienza.
I metodi ArrivalsMonitor sono contrassegnati come *virtual*, in modo che possano essere sottoposti a override da una classe derivata. Include i metodi *Subscribe* e *Unsubscribe*. Il metodo *Subscribe* consente alla classe di salvare l'implementazione di *IDisposable* restituita dalla chiamata a *Subscribe* in una variabile privata. Il metodo *Unsubscribe* consente alla classe di annullare la sottoscrizione delle notifiche chiamando l'implementazione di Dispose del provider. ArrivalsMonitor fornisce anche le implementazioni dei metodi *OnNext, OnError* e *OnCompleted* (spiegati precedentemente).

UML

Diagramma UML:



Documentazione sull'utilizzo

Il programma esegue senza bisogno di parametri in ingresso o passi particolari da seguire. Le informazioni verranno visualizzate in ordine alfabetico, in base al nome della città dell'aeroporto di provenienza e saranno divisi in due "monitor" in output, uno per lo scarico dei bagagli sul nastro e un altro per il ritiro degli stessi.

All'avviare della simulazione il punto di ingresso dell'applicazione crea un'istanza della classe *BaggageHandler*(provider), nonché due istanze della classe *ArrivalsMonitor* (observer1, observer2) e usa il metodo *BaggageStatus* per aggiungere o togliere informazioni sui bagagli dei voli in arrivo. In ognuno dei due casi, gli osservatori ricevono gli aggiornamenti e visualizzano le informazioni relative allo scarico e ritiro dei bagagli dal nastro.

Use case

Il provider indica se un osservatore desidera ricevere notifiche, quindi i chiamanti al metodo passano un'istanza dell'osservatore.

Ad un certo punto, un provider specificato può avere zero, uno o più osservatori. Il provider è responsabile dell'archiviazione dei riferimenti agli osservatori e della loro validità prima dell'invio delle notifiche.

Il provider può inviare tre tipi di notifiche all'osservatore (metodo notify).

L'osservatore notifica al provider che sta per ricevere le notifiche (metodo *subscribe*).