





**Domanda 1/3 (punti 5/30)**

Apple ha recentemente lanciato sul mercato la sua versione dei *tag* *bluetooth*, chiamata AirTag. Ogni *tag* è un dispositivo fisico capace di connettersi a dispositivi Apple, come iPhone, iPad e MacBook, utilizzando il canale di comunicazione *bluetooth*. A intervalli regolari, ogni *tag* invia un segnale crittato contenente la sua posizione (latitudine e longitudine). Un iPhone che entra nel raggio di azione del tag, e che precedentemente era stato abbinato al *tag* stesso, rileva l’informazione con il proprio sensore in ascolto sul canale *bluetooth*. Per facilitarne la gestione, il *tag,* pur non eseguendo fisicamente sul sistema operativo del telefono, è visto come locale ad esso. L’iPhone, quindi, invia a sua volta la posizione del *tag* e il suo identificativo a tutti i dispositivi che si trovino nel suo raggio di azione *bluetooth*. In questo modo, tali telefoni formano una rete estesa, capace di rilevare *tag* anche molto distanti dai telefoni associati. Anche in questo caso, gli iPhone dialogano tra loro come se fossero locali. Infine, un telefono associato a un *tag* può richiedere a quest’ultimo di emettere un suono, utilizzando un opportuno comando, se collegato direttamente alla medesima rete *bluetooth*.

Si modelli tale sistema mediante un diagramma delle classi, comprensivo dei *design pattern* a esso pertinenti.

**Risposta**

Il diagramma delle classi comprende le seguenti classi:

* Tag: Rappresenta un dispositivo fisico AirTag. Ha un ID univoco, una posizione (latitudine e longitudine) e la capacità di inviare un segnale criptato contenente la sua posizione e il suo ID ai dispositivi vicini.
* Telefono: Rappresenta un dispositivo, come un iPhone, in grado di rilevare e comunicare con i dispositivi AirTag. È dotato di un sensore Bluetooth che ascolta i segnali dei tag vicini e di un elenco di tag accoppiati. Ha anche la capacità di richiedere a un tag di emettere un suono.
* TagNetwork: Rappresenta la rete estesa di telefoni che possono rilevare i tag anche quando sono lontani dai telefoni associati. Ha un elenco di telefoni e un elenco di tag che sono stati rilevati dalla rete.

Il diagramma delle classi include anche i seguenti modelli di progettazione:

* Il pattern Observer è utilizzato per consentire ai telefoni di ascoltare i segnali dei tag vicini e di rilevare automaticamente i nuovi tag quando entrano nel raggio d'azione.
* Il pattern Singleton è utilizzato per garantire che esista una sola istanza della classe TagNetwork, che rappresenta l'intera rete di telefoni e tag.
* Il pattern Factory viene utilizzato per creare nuove istanze della classe Tag, poiché ogni tag ha un ID e una posizione unici.

# Domanda 2/3 (punti 3/30)

Dato il sistema precedentemente descritto, si usi un diagramma di sequenza per modellare la collaborazione delle componenti coinvolte nell’invio da parte di un *tag* della propria posizione a un iPhone, e successivamente, la richiesta da parte di quest’ultimo di emettere un suono.

# Risposta

Il diagramma di sequenza comprende i seguenti oggetti:

* Tag: Rappresenta un dispositivo fisico AirTag. Invia a intervalli regolari un segnale criptato contenente la sua posizione e il suo ID ai dispositivi vicini.
* iPhone: Rappresenta un iPhone in grado di rilevare e comunicare con i dispositivi AirTag. È dotato di un sensore Bluetooth che ascolta i segnali dei tag vicini e di un elenco di tag accoppiati. Ha anche la capacità di richiedere a un tag di emettere un suono.

Il diagramma di sequenza comprende anche i seguenti messaggi:

* Il tag invia un segnale criptato contenente la sua posizione e il suo ID ai dispositivi vicini.
* L'iPhone riceve il segnale e rileva il Tag utilizzando il suo sensore Bluetooth.
* L'iPhone invia al Tag la richiesta di emettere un suono.
* Il Tag riceve la richiesta ed emette un suono.

# Domanda 3/3 (punti 2/30)

Il sistema operativo iOS fornisce un’applicazione che visualizza, all’interno di una mappa, la posizione degli AirTag a esso associati. . In particolare, per ogni *tag* nella mappa, vengono visualizzate latitudine, longitudine e identificativo. Selezionando un *tag* sulla mappa ed entrando nelle informazioni di dettaglio, è possibile richiedere che esso emetta un suono, se questo è collegato direttamente tramite la rete *bluetooth*.

Utilizzando un diagramma dei casi d’uso, si modellino le esigenze sopra delineate. Non è richiesta alcuna descrizione testuale del diagramma.

# Risposta

Il diagramma dei casi d'uso comprende i seguenti attori:

* Utente: rappresenta la persona che utilizza l'applicazione iOS per visualizzare le posizioni degli AirTag e chiedere loro di emettere un suono.
* AirTag: Rappresenta un dispositivo AirTag fisico che è abbinato al dispositivo iOS dell'utente e può essere rilevato da quest'ultimo.
* Dispositivo iOS: Rappresenta il dispositivo iOS dell'utente, che ha installato l'applicazione e può rilevare e comunicare con i dispositivi AirTag.

Il diagramma dei casi d'uso comprende anche i seguenti casi d'uso:

* Visualizza posizioni AirTag: Rappresenta la possibilità per l'utente di visualizzare le posizioni degli AirTag su una mappa.
* Visualizza informazioni dettagliate sugli AirTag: Rappresenta la possibilità per l'utente di visualizzare informazioni dettagliate su uno specifico AirTag, tra cui latitudine, longitudine e identificatore.
* Richiesta di emissione di suoni da parte dell'AirTag: Rappresenta la possibilità per l'utente di richiedere a un AirTag di emettere un suono quando è collegato direttamente tramite la rete Bluetooth.

# Domanda 1/3 (punti 5/30)

Una *startup* statunitense sta sviluppando una applicazione che intende unificare tutte le librerie digitali disponibili sul mercato. L’applicazione si connette agli *account* dell’utente per tali librerie (Apple iCloud, Amazon Kindle, ecc.) così da recuperare la lista di libri in essi contenuti e renderli disponibili in un solo luogo virtuale. La rappresentazione di un libro,Book*,* è chiaramente unificata all’interno dell’applicazione così come il metodo di recupero. Ogni distributore di libri però ha la sua propria API, che deve così essere adattata per il corretto recupero delle corrispondenti informazioni. Per ogni servizio esterno e remoto, è presente un agente, che all’aggiunta/rimozione di un libro nella libreria originaria, notifica l’applicazione centrale, la quale recupera e aggiorna immediatamente le nuove informazioni disponibili.

Si modelli tale sistema mediante un diagramma delle classi, comprensivo dei *design pattern* a esso pertinenti.

# Risposta

Ecco un possibile diagramma delle classi per il sistema descritto:

[Libro]<>1--\*[Distributore di libri]

|

| 1

|

[Agente del libro]

|

| 1

|

[Applicazione centrale]

La classe Book rappresenta un libro all'interno del sistema e può includere attributi come titolo, autore ed editore, oltre a metodi per recuperare e aggiornare le informazioni sul libro.

La classe Distributore di libri rappresenta un distributore di libri esterno, come Apple iCloud o Amazon Kindle, e può includere attributi come il nome del distributore e l'endpoint API per accedere alle informazioni sul libro. Potrebbe anche includere metodi per recuperare e aggiornare le informazioni sui libri attraverso l'API.

La classe Book Agent rappresenta un agente responsabile della gestione della connessione a un particolare distributore di libri e può includere attributi quali il distributore di libri a cui è connesso e l'endpoint API per l'accesso alle informazioni sui libri. Potrebbe anche includere metodi per recuperare e aggiornare le informazioni sui libri e per inviare notifiche all'applicazione centrale quando un libro viene aggiunto o rimosso dalla libreria originale.

La classe Applicazione centrale rappresenta l'applicazione principale che unifica tutte le librerie digitali e potrebbe includere attributi come un elenco di tutti i libri presenti nel sistema e un elenco di tutti gli agenti librari. Potrebbe anche includere metodi per recuperare e aggiornare le informazioni sui libri e per aggiungere e rimuovere agenti librari.

In termini di design pattern, la classe Book Agent potrebbe essere considerata un pattern Facade, in quanto fornisce un'interfaccia semplificata per l'accesso alle informazioni sui libri da parte dei distributori di libri. La classe Book Distributor potrebbe essere considerata un pattern Bridge, in quanto disaccoppia l'interfaccia di accesso alle informazioni sui libri dall'implementazione di tale interfaccia. La classe Central Application può essere considerata un pattern Mediator, in quanto media la comunicazione tra gli agenti librari e i distributori di libri.

# Domanda 2/3 (punti 3/30)

Dato il sistema descritto nel quesito precedente, usate un diagramma di sequenza per modellare la collaborazione delle componenti coinvolte nella recupero di un nuovo libro da un *account* Apple iCloud, a seguito di una notifica da parte dell’agente.

# Risposta

Ecco un possibile diagramma di sequenza per lo scenario descritto:

[Agente per i libri] --notifica--> [Applicazione centrale]

[Applicazione centrale] --recupera le informazioni sul libro da Apple iCloud--> [Distributore di libri]

[Distributore libri] --restituisce le informazioni sul libro--> [Applicazione centrale]

[Applicazione centrale] -aggiorna le informazioni sul libro nel sistema--> [Libro]

[Applicazione centrale] --notifica agli agenti librari la presenza di un nuovo libro--> [Agente librario]

L'agente del libro invia una notifica all'applicazione centrale che segnala l'aggiunta di un nuovo libro alla libreria Apple iCloud.

L'applicazione centrale recupera le informazioni sul libro dal distributore di libri Apple iCloud utilizzando il suo endpoint API.

Il distributore di libri restituisce le informazioni sul libro all'applicazione centrale.

L'applicazione centrale aggiorna le informazioni sul libro nel sistema e invia una notifica agli agenti del libro per informarli della disponibilità di un nuovo libro.

# Domanda 3/3 (punti 2/30)

Naturalmente, l’applicazione già descritta dispone anche di una sua propria UI. Attraverso tale UI, l’utente può registrare un nuovo servizio esterno (Apple iCloud, Amazon Kindle, ecc.), così come effettuare ricerca dei libri presenti in locale, simultaneamente per titolo, autore, anno di pubblicazione, casa di pubblicazione. La lista di libri risultante dalla ricerca (ove non vuota), riporta e visualizza il nome del libro, l’autore, e una copertina, opportunamente scaricata dalla libreria originaria.

Modellare le esigenze sopra delineate utilizzando un diagramma dei casi d’uso. Non è richiesta descrizione testuale del diagramma.

# Risposta

Ecco un possibile diagramma dei casi d'uso per il sistema descritto:

[Utente]

|

| 1

|

[UI]

|

| 1..\*

|

[Applicazione centrale]

|

| 1..\*

|

[Distributore di libri]

|

| 1..\*

|

[Agente del libro]

|

| 1..\*

|

[Libro]

L'attore Utente rappresenta la persona che utilizza l'applicazione.

Il caso d'uso UI rappresenta l'interfaccia utente dell'applicazione.

Il caso d'uso Applicazione centrale rappresenta l'applicazione principale che unifica tutte le librerie digitali.

Il caso d'uso Book Distributor rappresenta un distributore di libri esterno, come Apple iCloud o Amazon Kindle.

Il caso d'uso Book Agent rappresenta un agente responsabile della gestione della connessione a un particolare distributore di libri.

Il caso d'uso Book rappresenta un libro all'interno del sistema.

Il caso d'uso UI comprende i seguenti sottocasi d'uso:

Registrazione del distributore di libri: Questo caso d'uso rappresenta il processo di registrazione di un nuovo distributore di libri con l'applicazione.

Ricerca libri: Questo sottocaso d'uso rappresenta il processo di ricerca di libri all'interno del sistema per titolo, autore, anno di pubblicazione e casa editrice.

Il caso d'uso dell'applicazione centrale comprende i seguenti casi d'uso secondari:

Recupera informazioni sui libri: Questo sottocaso d'uso rappresenta il processo di recupero delle informazioni sui libri da un distributore di libri attraverso il corrispondente agente librario.

Aggiornare le informazioni sul libro: Questo sottocaso d'uso rappresenta il processo di aggiornamento delle informazioni sui libri nel sistema.

Il caso d'uso Distributore di libri comprende i seguenti sottocasi d'uso:

Fornire informazioni sui libri: Questo sottocaso d'uso rappresenta il processo di fornitura di informazioni sui libri all'applicazione centrale attraverso la sua API.