

POLITECNICO DI MILANO

Facoltà di Ingegneria Industriale e dell'Informazione

Corso di Laurea in Ingegneria Informatica



Progetto di Ingegneria del Software 2

TravelDream

DD

Professore: **Luca Mottola**

Autori:
Emma Balgera
Sara Biancini
Pietro Bressana

Anno Accademico 2013-2014

Questo documento rappresenta la seconda deliverable. Insieme ad esso è stato fornito anche il RASD aggiornato.

Scopo del Design Document (DD) è mostrare in modo dettagliato la struttura del prodotto. Ovviamente, tutte le scelte effettuate nella stesura del precedente documento saranno rispettate, descrivendole in modo più preciso e a “basso livello”, in particolare, il capitolo 2.3 del RASD, cioè quello delle assunzioni e delle scelte di progettazione. Per questo motivo, saranno introdotti i concetti in modo incrementale, dalla descrizione dell’architettura fino a mostrare il comportamento e funzionamento di TravelDream, tramite appositi schemi.

Saranno utilizzati: UX Diagram, BCE, ER Model, EER Model, Logical Model e Diagrammi di Sequenza.

Indice

1. Errata Corrige	4
2. Introduzione	5
2.1. Definizioni, acronimi e abbreviazioni.....	5
3. Architettura	6
3.1. Presentation Tier.....	7
3.2. Business Tier	7
3.3. Data Tier.....	7
3.4. Design Pattern.....	8
4. Design	9
4.1. Progettazione concettuale: ER Model	10
4.2. Progettazione concettuale: EER Model	13
4.3. Progettazione logica: Logic Model.....	18
4.4. Modello di navigazione: UX Model.....	19
4.5. Altri diagrammi	25
5. Progetto per JEE.....	35
Indice delle figure	39

1. Errata Corrige

E' stato aggiornato il DD alla versione 1.2.

In questa nuova versione sono state apportate le seguenti modifiche:

- ER Model
- EER Model
- Logic Model
- Modifica UXModel Registrazione di un visitatore e autenticazione di un cliente.
- Modifica UXModel Ricerca dei pacchetti da parte di un cliente.
- Modifica UXModel Gestione dei pacchetti da parte di un cliente.
- Modifica UXModel Gestione dei componenti base da parte di un impiegato.
- Modifica UXModel Gestione di pacchetti predefiniti da parte di un impiegato.
- Modifica UXModel Accettazione o rifiuto alla partecipazione a un viaggio da parte di un amico.
- Modifica UXModel Accettazione o rifiuto di una proposta regalo da parte di un amico.
- Modifica BCE Model del visitatore.
- Modifica BCE Model del cliente 1.
- Modifica BCE Model del cliente 2.
- Modifica BCE Model del cliente 3.
- Modifica BCE Model del cliente 4.
- Modifica BCE Model dell'impiegato 1.
- Modifica BCE Model dell'impiegato 2.
- Modifica BCE Model dell'impiegato 3.
- Modifica BCE Model dell'impiegato 4.
- Modifica EntityBeans.

2. Introduzione

2.1. Definizioni, acronimi e abbreviazioni

- Design pattern: “soluzione progettuale generale di un problema ricorrente”. È una descrizione o un modello da applicare per risolvere un problema che può presentarsi in diverse situazioni durante la progettazione e lo sviluppo del software.
- Browser: programma installato in un sistema operativo che permette l’accesso alla rete internet attraverso protocolli standard. Questo software mostra pagine web interpretando del codice ricevuto da un server. Esempi di codice con cui possono essere scritte sono: HTML, CSS, Javascript ecc...
- JEE: la Java Platform, Enterprise Edition o Java EE è una piattaforma software di programmazione Java. La piattaforma fornisce API e un ambiente di runtime per lo sviluppo e l’esecuzione di software per le imprese, compresi i servizi di rete e web, e di altre grandi applicazioni di rete a più livelli. La Java EE estende la Java 2 Platform, Standard Edition (Java SE). La piattaforma incorpora un design basato in gran parte su componenti modulari in esecuzione su un server di applicazioni.
- EJB: gli Enterprise JavaBeans (EJB) sono i componenti che implementano, lato server, la logica di business all’interno dell’architettura Java EE. Le specifiche per gli EJB definiscono diverse proprietà che questi devono avere, tra cui la persistenza, il supporto alle transazioni, la gestione della concorrenza e della sicurezza e l’integrazione con altre tecnologie, come JMS, JNDI, e CORBA. Lo standard attuale è EJB 3 ed è stato completato nella primavera del 2006.
- Session Bean: nella Java Platform è un tipo di Enterprise Bean. Una Session Bean implementa una business task ed è ospitata in un EJB container.
- Entity Bean: è un tipo di Enterprise JavaBean, un componente J2EE server-side che rappresenta dati persistenti gestiti in un database.

3. Architettura

Nel RASD non è stato precisato come sarà realizzato il lato client. Da specifica **il committente ha proposto queste due possibili soluzioni:**

- **Un'applicazione web, alla quale l'utente potrà accedere con qualunque browser.**
- **Un'applicazione Java da scaricare ed installare sul proprio computer.**

Dopo un'accurata analisi, sono stati evidenziati i seguenti punti fondamentali per la scelta:

1. L'applicazione di un'agenzia viaggi deve operare solo attraverso il web senza avere accesso al computer locale, a meno di possibili funzioni di upload/download.
2. Gli utenti sono distribuiti su molti sistemi operativi, comprese diverse distribuzioni Linux (anche se in modo meno rilevante)
3. Il numero di utenti di "smartphone" e "tablet" è cresciuto così tanto da indurci a sviluppare un'applicazione che possa essere supportata anche da questi dispositivi mobile, che hanno sistemi operativi differenti, non supportano gli stessi software/linguaggi di programmazione e spesso richiedono strumenti di sviluppo e requisiti differenti.

Possibili soluzioni ai punti elencati sopra:

1. **Realizzare un'applicazione web** accessibile tramite browser.
2. **Rendere indipendente il lato client dal sistema operativo (es. tramite un'applicazione Java)**
3. **Rendere il lato client compatibile con i principali dispositivi mobile.**

Di conseguenza, **la migliore soluzione è realizzare l'interfaccia come un'applicazione web, alla quale l'utente potrà accedere con qualunque browser, indipendentemente dal sistema operativo e dal tipo di dispositivo (mobile o fisso).**

L'architettura del sistema è quella chiamata Three-Tier ("a tre strati/livelli"). Grazie ad essa vi è una separazione tra l'interfaccia utente (Presentation Tier), la logica funzionale (Business Tier) e la persistenza dei dati (Data Tier) in più strati. Essi possono risiedere sullo stesso server oppure su più macchine.

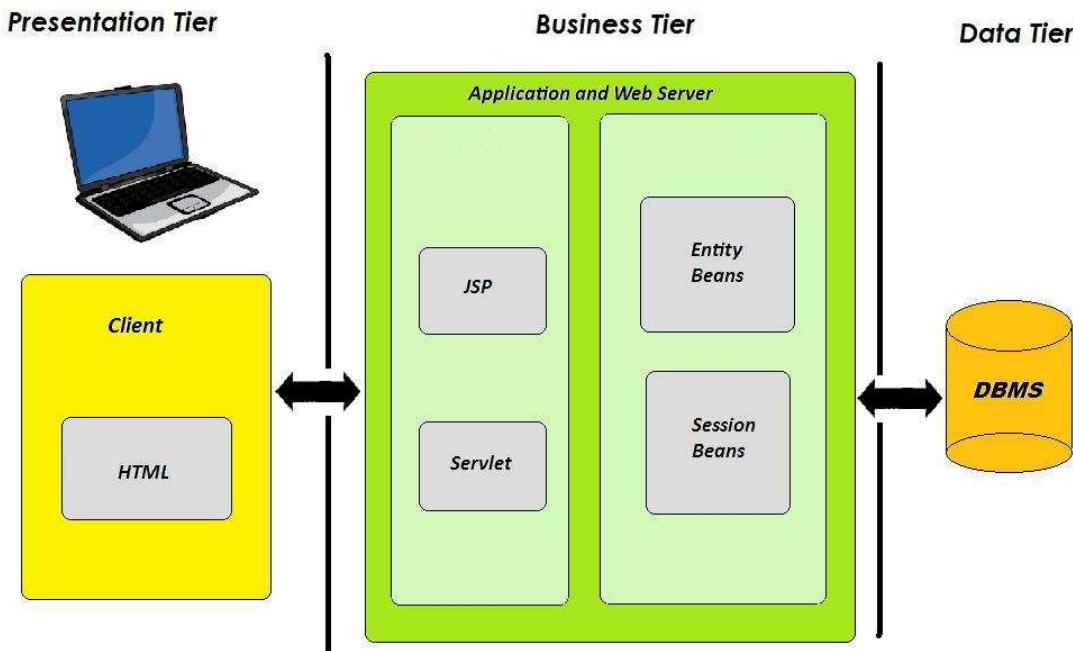


Fig. 3.1 – Architettura Three-Tier

3.1. Presentation Tier

Strato che si occupa della presentazione, cioè della visualizzazione dei dati. Essi sono mostrati tramite un'interfaccia grafica per far sì che l'utente possa interagire col sistema tramite funzioni predefinite e semplici da usare. Alcune delle tecnologie utilizzate per raggiungere tale obiettivo sono JSP, HTML, JavaScript, e JSF.

3.2. Business Tier

E' lo strato più importante del sistema. Gestisce la logica, cioè fornisce l'accesso al database, e costituisce il cuore della logica di business. La tecnologia utilizzata è EJB, poiché rappresenta un requisito di progetto fornito dal committente.

3.3. Data Tier

Strato che si occupa di gestire la persistenza dei dati, cioè il database. Il team di sviluppo ha scelto MySQL, un database di tipo relazionale.

3.4. Design Pattern

La suddivisione tra visualizzazione (interfaccia utente), controllo (che fornisce informazioni alla GUI, gestisce la logica e utilizza i dati) e modello (che rappresenta i dati) permette di far uso del Design Pattern MVC (Model – View – Controller).

Lo schema seguente (Fig. 3.2) mostra le interazioni dirette tra i vari moduli software e quelle indirette (frecce sottili). Le prime sono effettuate tramite associazioni dirette (richieste), le seconde tramite quelle indirette, spesso con Design pattern.

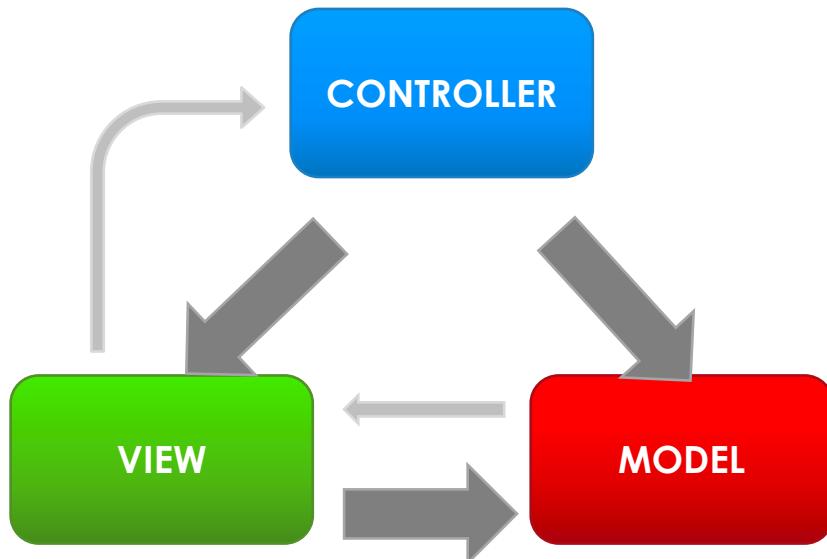


Fig. 3.2 – Architettura Three-Tier

4. Design

Questo capitolo descrive la fase di design, cioè la progettazione dell'intero software. Inizialmente sarà mostrata la definizione ad alto livello della struttura del database. In seguito, verrà mostrata la progettazione logica, sempre riferita al database.

Garantita la persistenza dei dati ad alto livello, il team si è concentrato sul modello di navigazione. L'obiettivo è di scendere ad un livello più basso e dettagliato della descrizione del sistema in termini di visualizzazione/navigazione delle pagine web.

Per fornire una descrizione più dettagliata e per ricollegarsi alla divisione in tre livelli descritta nel capitolo precedente, si è deciso di adottare anche i diagrammi di analisi (BCE).

Completate queste fasi e definito nel dettaglio ogni comportamento del sistema, sarà sviluppata la progettazione di TravelDream basata su JEE (rispettando il vincolo imposto dal committente), cioè definendo gli Entity Bean e i Session Bean.

Infine, per una maggiore chiarezza e per evitare interpretazioni errate di alcune scelte di progettazione, sono stati allegati alcuni Sequence Diagram.

Attenzione: è ovvio che ogni passo della fase di design debba rispettare tutte le scelte progettuali effettuate nel RASD.

4.1. Progettazione concettuale: ER Model

La fase principale della progettazione di un database consiste nell'individuazione delle entità e delle relazioni.

Essendo un modello ad alto livello, è facilmente leggibile ed interpretabile.

Per comprenderne il significato sono necessarie alcune informazioni sulla notazione usata:

- **Entità:** rappresenta una classe di oggetti che hanno caratteristiche comuni.
Nello schema ER è rappresentata come un rettangolo con all'interno un nome.
- **Relazione:** identifica un legame tra due o più entità.
E' rappresentata come un rombo che ha all'interno un verbo o un sostantivo, ed è collegato alle entità che deve "legare".
- **Attributo:** elemento contenuto in un'entità.
Nello schema ER è rappresentato come pallino vuoto collegato ad un'entità, che possiede un nome.
- **Attributo chiave:** attributo che identifica univocamente gli elementi appartenenti alla stessa entità.
E' indicato come un pallino pieno collegato ad un'entità, che possiede un nome.

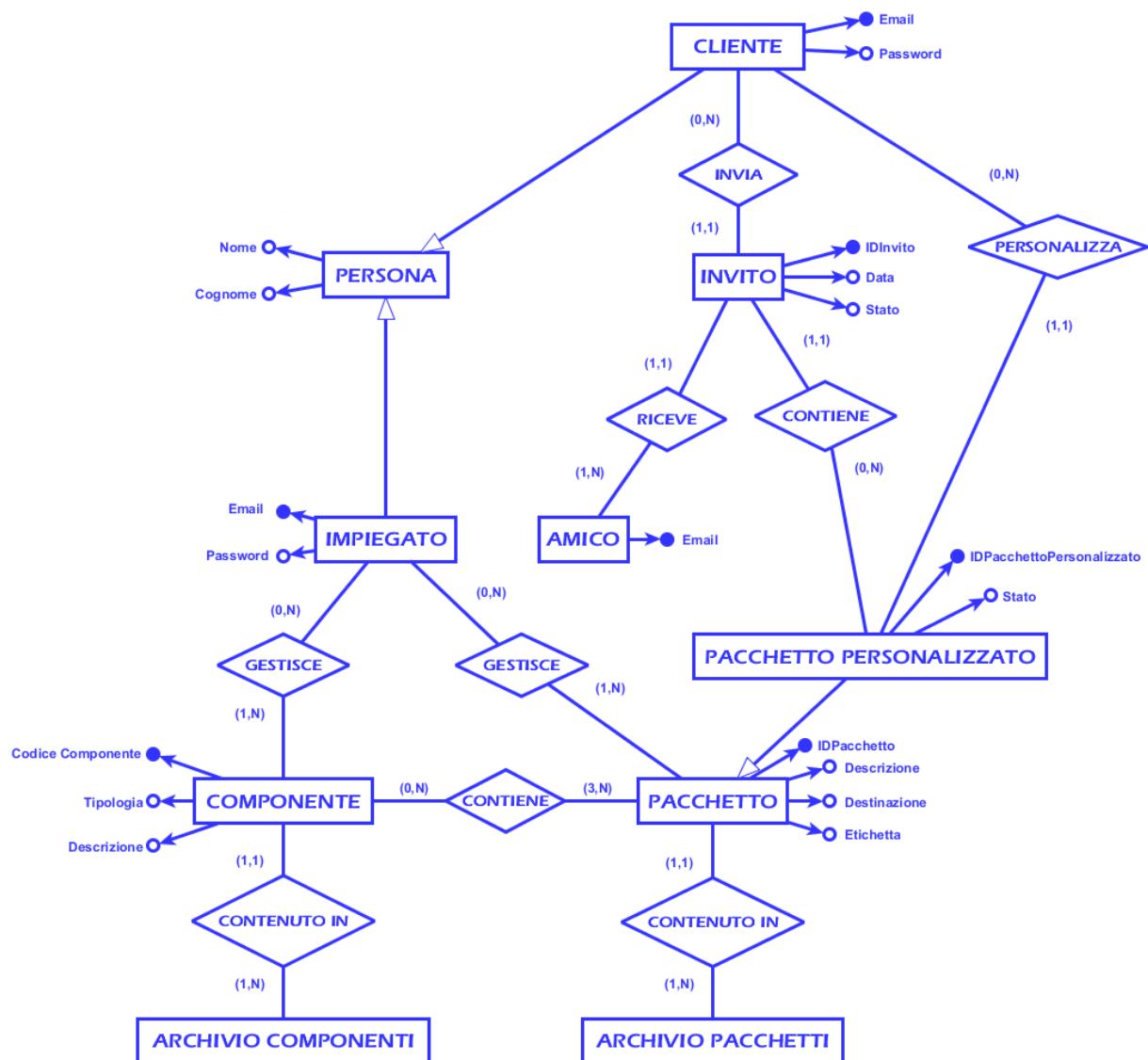


Fig. 4.1 – Schema ER

4.1.1. Entità

Persona

Entità generica che non sarà rappresentata negli schemi successivi. L'unico scopo è di esprimere che "Impiegato" e "Cliente" condividono alcuni attributi.

Gli attributi sono:

- Nome: nome reale di una persona
- Cognome: cognome reale di una persona.

Cliente

Entità che rappresenta l'utilizzatore registrato del sistema ed estende l'entità Persona. **E' importante notare che la classe Visitatore non è stata rappresentata nel database, in quanto non è necessario memorizzare i suoi dati.**

Gli attributi sono:

- Email: chiave primaria dell'entità Cliente. Sarà utilizzata per eseguire le ricerche di clienti.
- Password: parola chiave che permette il Login, insieme all'Email.

Amico

Entità che rappresenta un cliente o un visitatore a cui un cliente invia un invito per partecipare ad un viaggio.

- Email: chiave primaria dell'entità Amico. Sarà utilizzata dal cliente per inviare un invito per un viaggio.

Impiegato

Entità che estende Persona e che identifica un impiegato di TravelDream.

- Email: chiave primaria che identifica univocamente un impiegato.
- Password: codice segreto che permette ad un impiegato di accedere all'area riservata, insieme all'Email.

Componente

Entità che rappresenta un componente base che può essere di tre tipologie diverse: volo, hotel ed escursione.

- CodiceComponete: chiave primaria che identifica in modo univoco un componente base. Sarà utilizzato per eseguire ricerche di componenti tramite codice.
- Tipologia: attribuisce ad un componente una delle possibili tipologie. Sarà utilizzato per eseguire ricerche di componenti tramite filtro.
- Descrizione: attributo che specifica le informazioni generali del componente base.

Pacchetto

Entità che rappresenta un pacchetto predefinito.

- IDPacchetto: chiave primaria che identifica univocamente un pacchetto base. Sarà utilizzato per effettuare ricerche di pacchetti predefiniti tramite identificativo.
- Destinazione: nome della metà turistica che si riferisce al dato pacchetto predefinito. Sarà utilizzato per effettuare ricerche di pacchetti predefiniti tramite destinazione.
- Etichetta: attributo che permette la ricerca di un pacchetto predefinito tramite etichetta.
- Descrizione: attributo che specifica le informazioni generali del pacchetto predefinito.

Pacchetto Personalizzato

Entità che rappresenta un pacchetto personalizzato da un cliente.

- IDPacchettoPersonalizzato: chiave primaria che identifica in modo univoco un pacchetto personalizzato di un cliente.
- Stato: attributo che rappresenta lo stato del pacchetto personalizzato di un cliente.

Invito

Entità che rappresenta un invito di un cliente autenticato al sito TravelDream inviato ad un amico.

- IDInvito: chiave primaria che identifica un invito.
- Data: attributo che rappresenta la data di invio di un invito
- Stato: attributo che rappresenta lo stato attuale di un invito: accettato o rifiutato. L'attributo rimane vuoto nell'intervallo di tempo che trascorre tra l'invio dell'invito e la sua scadenza o la ricezione di una risposta.

Archivio Componenti

Entità che rappresenta la lista di tutti i componenti base presenti nel sistema.

Archivio Pacchetti

Entità che rappresenta la lista di tutti i pacchetti predefiniti presenti nel sistema.

4.1.2. Relazioni binarie

- **Gestisce**: un impiegato “gestisce” da 0 a N componenti, mentre un componente “è gestito” da 1 a N impiegati.
- **Gestisce**: un impiegato “gestisce” da 0 a N pacchetti, mentre un pacchetto “è gestito” da 1 a N impiegati.
- **Contiene**: un pacchetto “contiene” da 3 a N componenti, mentre un componente “è contenuto” da 0 a N pacchetti.
- **Contenuto In**: l'archivio componente “contiene” da 1 a N componenti, mentre un componente “è contenuto” in uno ed un solo archivio componenti.
- **Contenuto In**: l'archivio pacchetti “contiene” da 1 a N pacchetti, mentre un pacchetto “è contenuto” in uno ed un solo archivio pacchetti.
- **Riceve**: un amico “riceve” da 1 a N inviti, mentre un invito “è ricevuto” da uno ed un solo amico.
- **Invia**: un cliente “invia” da 0 a N inviti, mentre un invito “è inviato” da uno ed un solo cliente.
- **Contiene**: un invito “contiene” uno ed un solo pacchetto personalizzato, mentre un pacchetto personalizzato “è contenuto” da 0 a N inviti.
- **Personalizza**: un cliente “personalizza” da 0 a N inviti, mentre un invito “è personalizzato” da uno ed un solo cliente.

4.2. Progettazione concettuale: EER Model

Durante la progettazione concettuale, si è scelto di realizzare anche l'EER (Enhanced ER o Extended ER) che “estende” l'ER Model.

In questo schema si mettono in evidenza le cardinalità delle relazioni tramite una simbologia differente rispetto l'ER. Inoltre, **questo modello mostra più nel dettaglio le entità, indentificando il tipo degli attributi e le chiavi esterne.**

Simboli delle entità:

- Chiave gialla: indica la chiave primaria
- Rombo rosso: indica una chiave esterna (non utilizzato)
- Rombo bianco: attributo che può essere NULL
- Rombo azzurro: attributo che non può essere NULL

Le relazioni si distinguono in:

- “Nonidentifying Relationship” quando la chiave esterna non fa parte della chiave primaria della tabella figlia (non utilizzata in questo schema).
- “Identifying Relationship” quando la chiave esterna fa parte della chiave primaria nella tabella figlia.

Cardinalità delle relazioni:

- Indica una cardinalità “1”
- Indica una cardinalità “N”

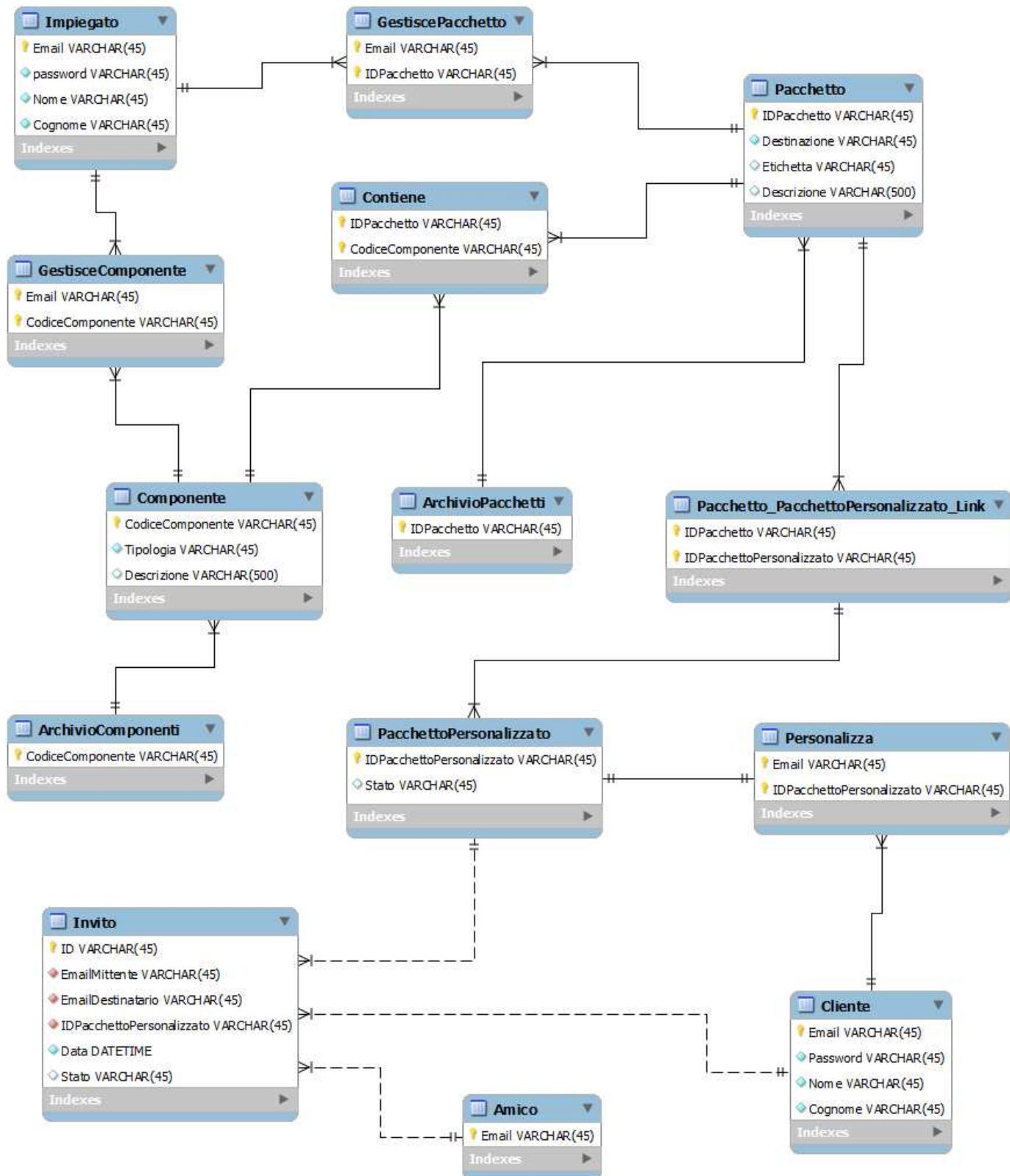


Fig. 4.2 – EER Model

Le differenze fondamentali dallo schema ER precedente sono:

- La relazione “riceve” tra le entità amico ed invito e la relazione “invia” tra entità cliente ed invito, non sono presenti nello schema perché l’entità invito contiene come chiave primaria l’e-mail del mittente e quella del destinatario. Questo permette di sostituire tali relazioni con delle semplici associazioni tra entità.
- Le relazioni “contiene” tra componente ed archivio componente, e tra pacchetto ed archivio pacchetti non sono presenti nello schema perché sostituite con semplici associazioni 1 a molti.
- La generalizzazione pacchetto, pacchetto personalizzato è stata rappresentata inserendo una nuova entità pacchetto_pacchetto personalizzato link che associa ad ogni pacchetto personalizzato uno ed un solo pacchetto.

Ora segue la spiegazione dettagliata di ogni entità dello schema EER e dei loro attributi.

Cliente

Entità che rappresenta un cliente dell’applicazione TravelDream, cioè un utente che si è registrato al sistema. Non vi è alcuna differenza con lo schema ER a parte il fatto che “eredita” gli attributi da Persona. Per maggiore completezza sono riportati tutti gli attributi dell’entità.

- Email: chiave primaria che costituisce, insieme alla password, le informazioni di login del cliente
- Password: parola chiave che permette, insieme alla e-mail, di autenticarsi con il sistema.
- Nome: nome proprio del cliente del sistema.
- Cognome: cognome del cliente.

Amico

Entità che rappresenta un cliente o un visitatore a cui un cliente invia un invito per partecipare ad un viaggio.

- Email: chiave primaria dell’entità Amico; sarà utilizzata dal cliente per l’invio di un invito.

Impiegato

Entità che rappresenta un impiegato di TravelDream.

Poiché “eredita” tutti gli attributi di Persona (nello schema ER), avrà i seguenti attributi:

- Email: chiave primaria che costituisce, insieme alla password impiegato, le informazioni di Login dell’impiegato.
- Password: parola chiave che permette, insieme all’email dell’impiegato, l’autenticazione al sistema.
- Nome: nome proprio dell’impiegato di TravelDream.
- Cognome: cognome dell’impiegato.

Gestisce Componente

Entità che rappresenta i componenti base che vengono gestiti da un determinato impiegato.

- Email: chiave primaria che identifica in modo univoco un impiegato di TravelDream.
- CodiceComponente: chiave primaria che identifica in modo univoco un componente base.

Gestisce Pacchetto

Entità che rappresenta i pacchetti predefiniti che vengono gestiti da un determinato impiegato.

- Email: chiave primaria che identifica in modo univoco un impiegato di TravelDream.
- IDPacchetto: chiave primaria che identifica in modo univoco un pacchetto predefinito.

Componente

Entità che rappresenta un componente base che può essere di tre tipologie diverse: volo, hotel ed escursione.

- CodiceComponete: chiave primaria che identifica in modo univoco un componente base. Sarà utilizzato per eseguire ricerche di componenti tramite codice.
- Tipologia: attribuisce ad un componente una delle possibili tipologie. Sarà utilizzato per eseguire ricerche di componenti tramite filtro.
- Descrizione: attributo che specifica le informazioni generali del componente base.

Pacchetto

Entità che rappresenta un pacchetto predefinito.

- IDPacchetto: chiave primaria che identifica in modo univoco un pacchetto predefinito. Sarà utilizzato per effettuare ricerche di pacchetti predefiniti tramite identificativo.
- Destinazione: nome della metà turistica che si riferisce al dato pacchetto predefinito. Sarà utilizzato per effettuare ricerche di pacchetti predefiniti tramite destinazione.
- Etichetta: attributo che permette la ricerca di un pacchetto predefinito tramite etichetta.
- Descrizione: attributo che specifica le informazioni generali del pacchetto predefinito.

Contiene

Entità che rappresenta quali componenti base compongono un determinato pacchetto predefinito.

- IDPacchetto: chiave primaria che identifica in modo univoco un pacchetto predefinito.
- CodiceComponete: chiave primaria che identifica in modo univoco un componente base.

Pacchetto Personalizzato

Entità che rappresenta un pacchetto personalizzato da un cliente.

- IDPacchettoPersonalizzato: chiave primaria che identifica in modo univoco un pacchetto personalizzato di un cliente.
- Stato: attributo che rappresenta lo stato del pacchetto personalizzato di un cliente.

Invito

Entità che rappresenta un invito di un cliente autenticato al sito TravelDream inviato ad un amico.

- EmailMittente: chiave primaria che identifica un cliente che ha inviato un invito ad un amico
- EmailDestinatario: chiave primaria che identifica un amico a cui è stato inviato un invito per visionare un pacchetto viaggio.
- IDPacchettoPersonalizzato: chiave primaria che identifica un pacchetto personalizzato a cui un invito si riferisce.
- Data: attributo che rappresenta la data di invio di un invito
- Stato: attributo che rappresenta lo stato attuale di un invito: accettato o rifiutato. L'attributo rimane vuoto nell'intervallo di tempo che trascorre tra l'invio dell'invito e la sua scadenza o la ricezione di una risposta.

Pacchetto Pacchetto Personalizzato Link

Entità che rappresenta la generalizzazione tra pacchetto e pacchetto personalizzato.

- IDPacchetto: chiave primaria che identifica in modo univoco un pacchetto predefinito.
- IDPacchettoPersonalizzato: chiave primaria che identifica in modo univoco un pacchetto personalizzato di un cliente.

Personalizza

Entità che rappresenta i pacchetti personalizzati di un determinato cliente di TravelDream.

- Email: chiave primaria che costituisce che identifica in modo univoco un cliente di TravelDream.
- IDPacchettoPersonalizzato: chiave primaria che identifica in modo univoco un pacchetto personalizzato.

Archivio Componenti

Entità che rappresenta la lista di tutti i componenti base presenti nel sistema.

- CodiceComponente: chiave primaria che identifica in modo univoco un componente base.

Archivio Pacchetti

Entità che rappresenta la lista di tutti i pacchetti predefiniti presenti nel sistema.

- IDPacchetto: chiave primaria che identifica univocamente un pacchetto predefinito.

4.3. Progettazione logica: Logic Model

La progettazione logica è una traduzione dello schema ER. Essa mostra le entità con gli attributi e le chiavi, ma si concentra soprattutto sul descrivere i collegamenti concettuali tra le tabelle. Nello schema seguente sono indicate le chiavi primarie in giallo. Le chiavi esterne, rappresentate come chiavi primarie, non sono indicate in modo esplicito nello schema sottostante.

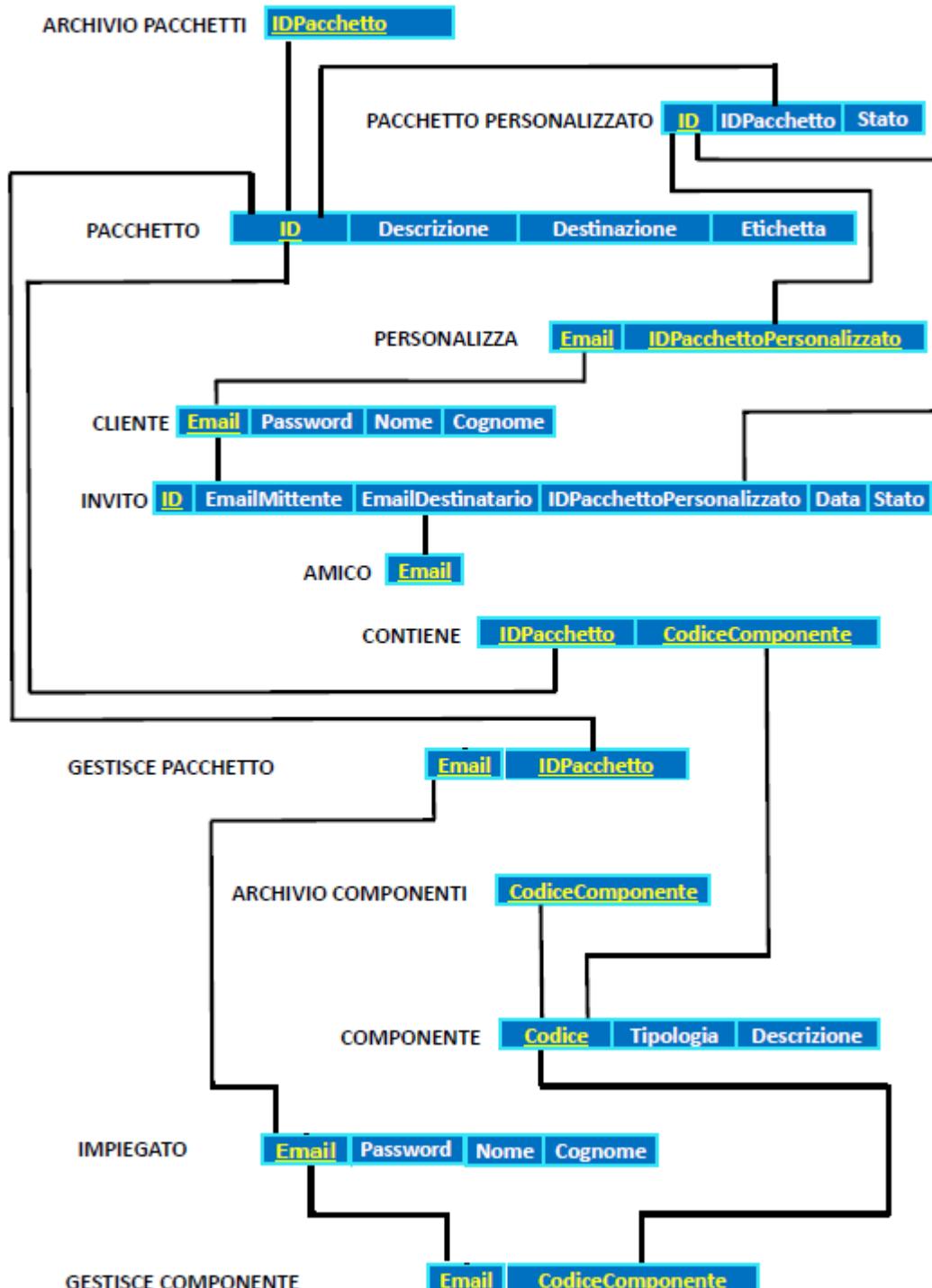


Fig. 4.3 – Schema Logico

4.4. Modello di navigazione: UX Model

Un modello di navigazione, detto User eXperience Model (“UX Model”), rappresenta la navigazione tra le pagine di TravelDream. Ci sono <<screen>>, <<input form>> con nome, attributi e metodi ed aggregazioni, composizioni e metodi.

Il simbolo “+” rappresenta una lista di elementi con gli attributi specificati nella classe oppure all'esterno tramite la relazione di aggregazione (rombo vuoto).

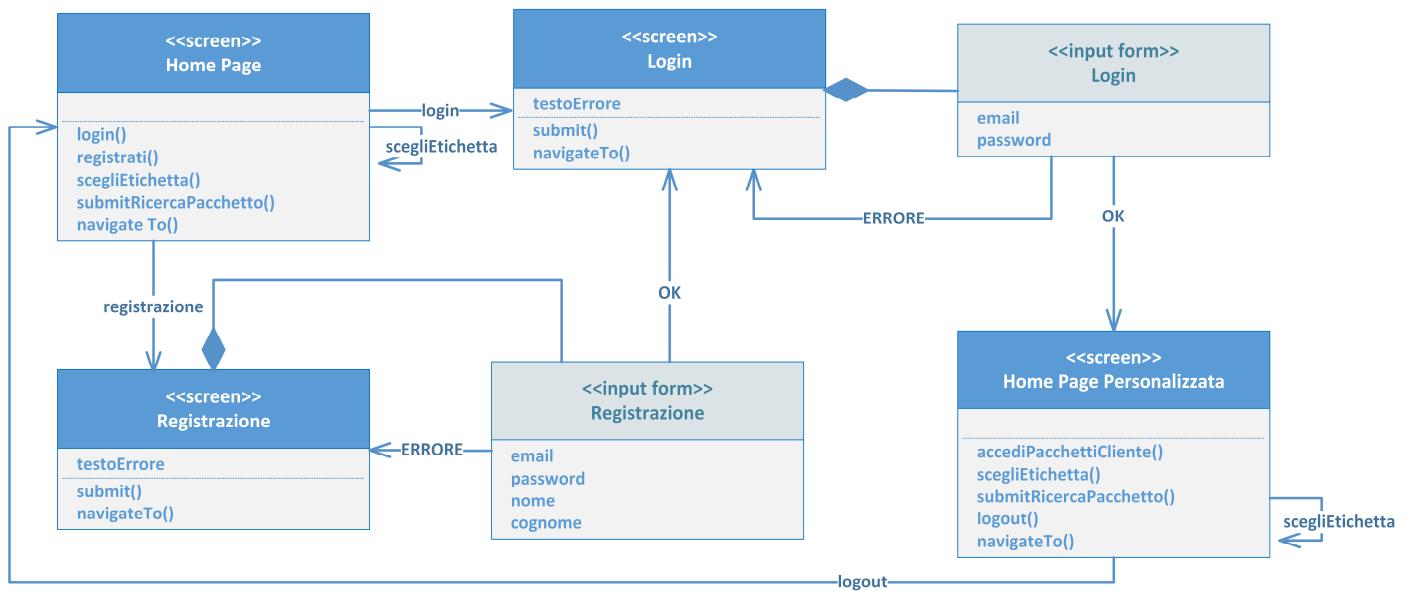


Fig. 4.4 – UXModel Registrazione di un visitatore e autenticazione di un cliente

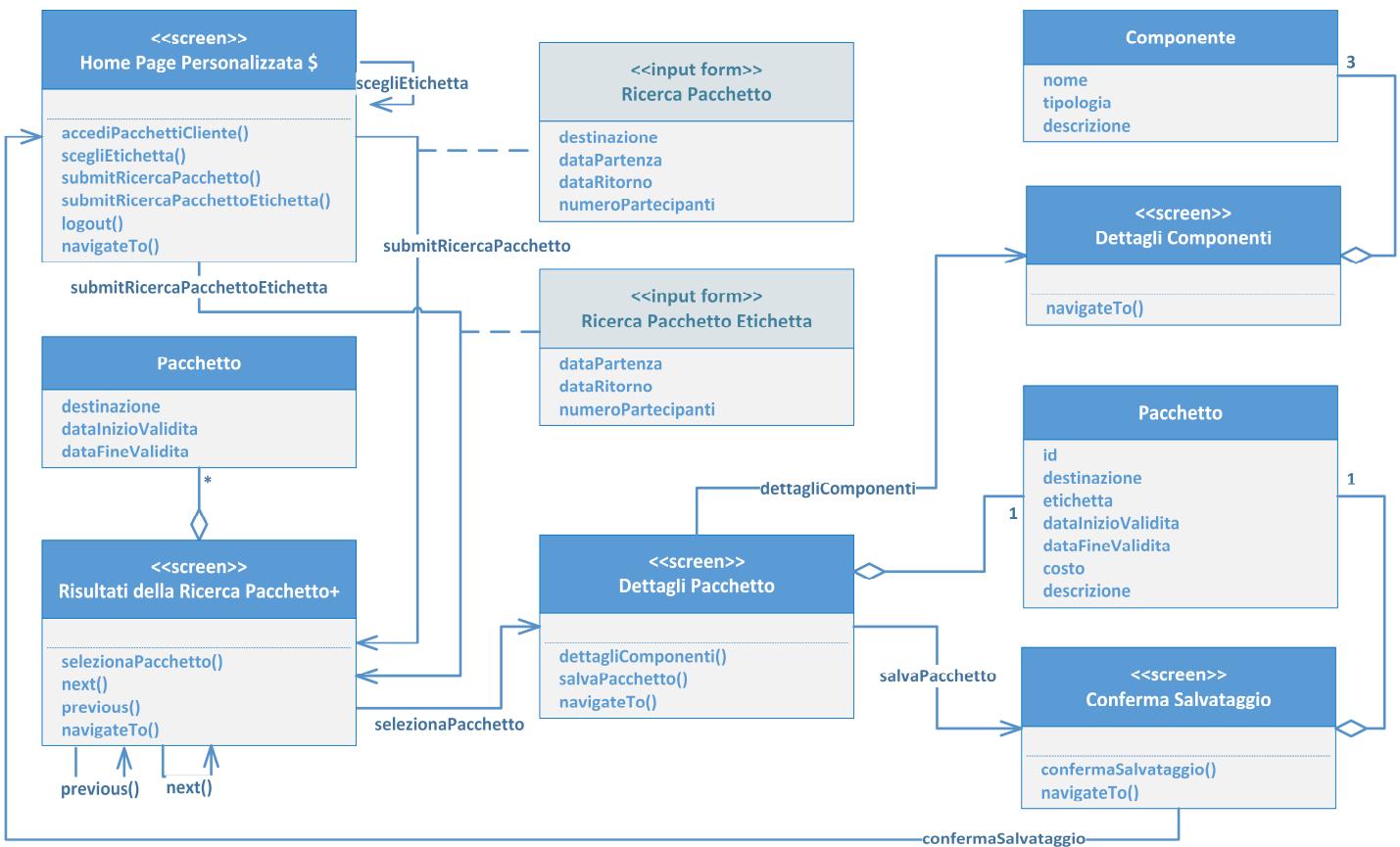


Fig. 4.5 – UXModel Ricerca dei pacchetti da parte di un cliente

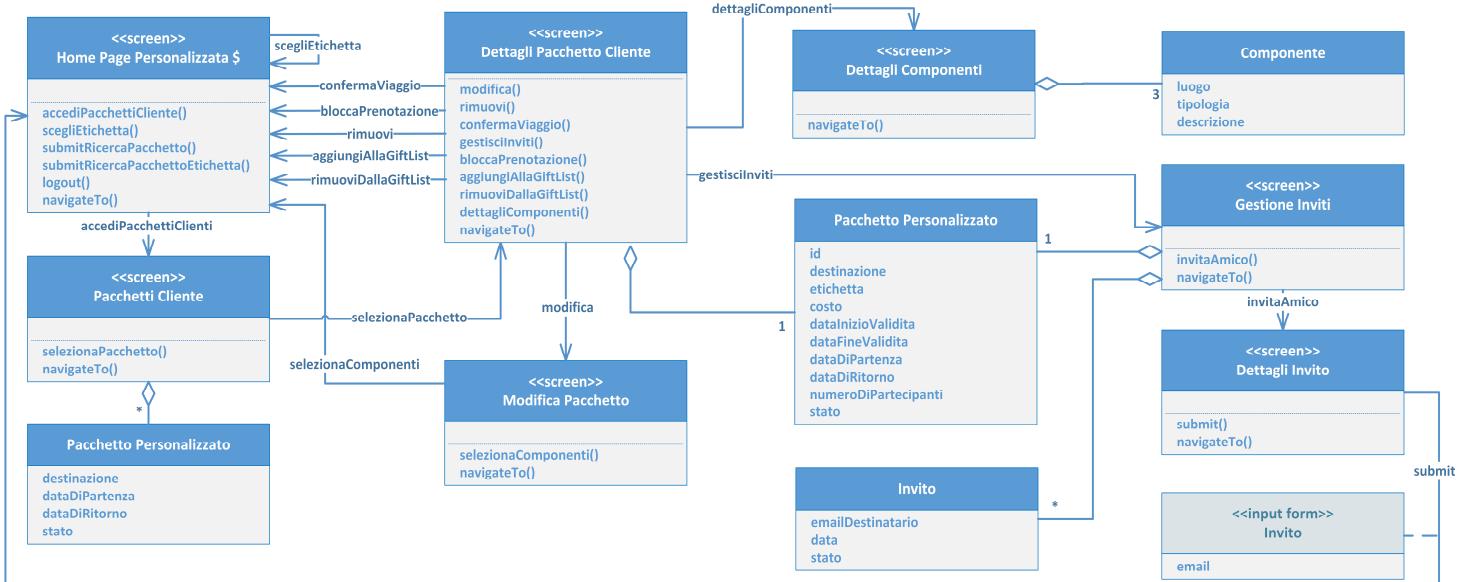
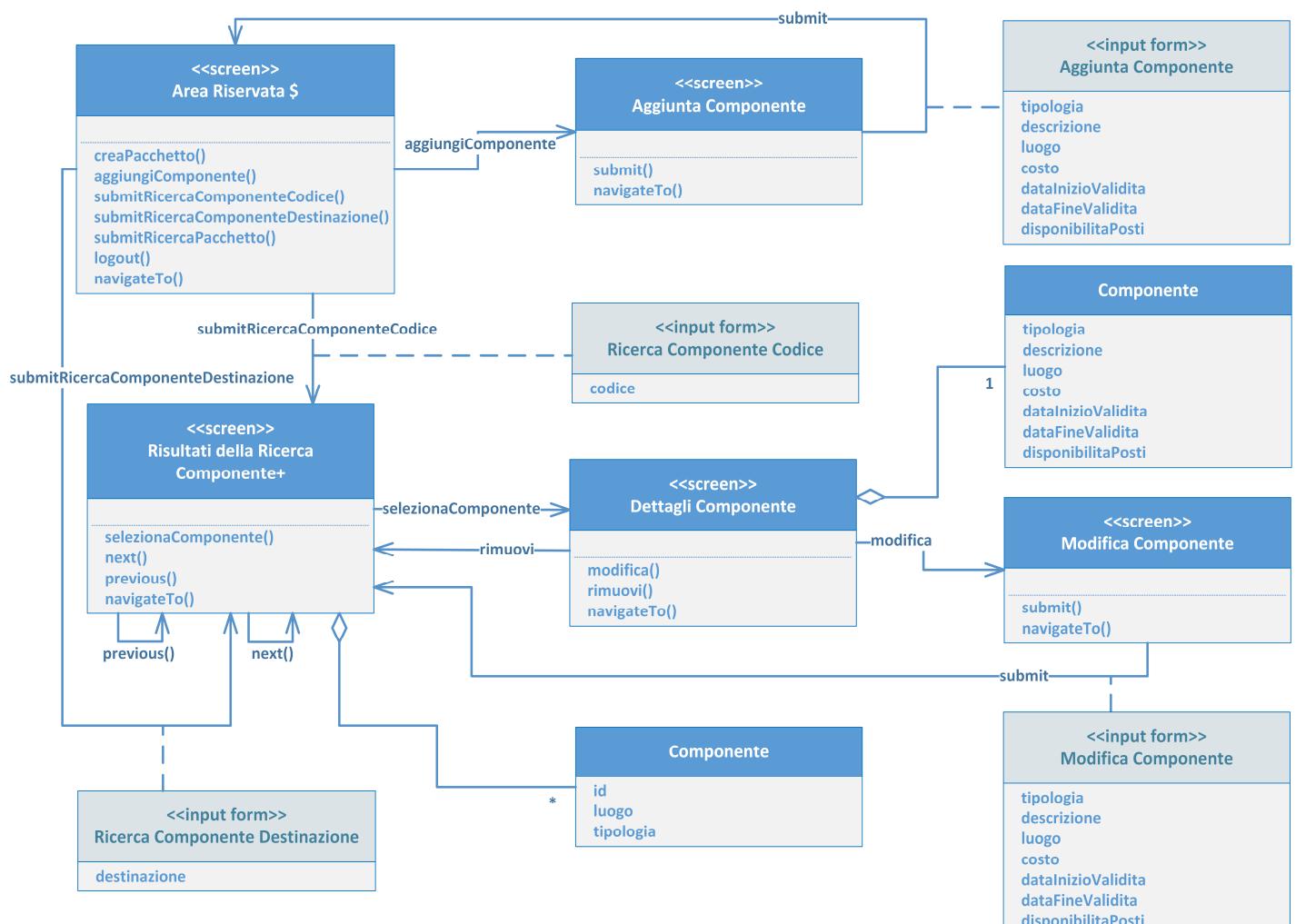
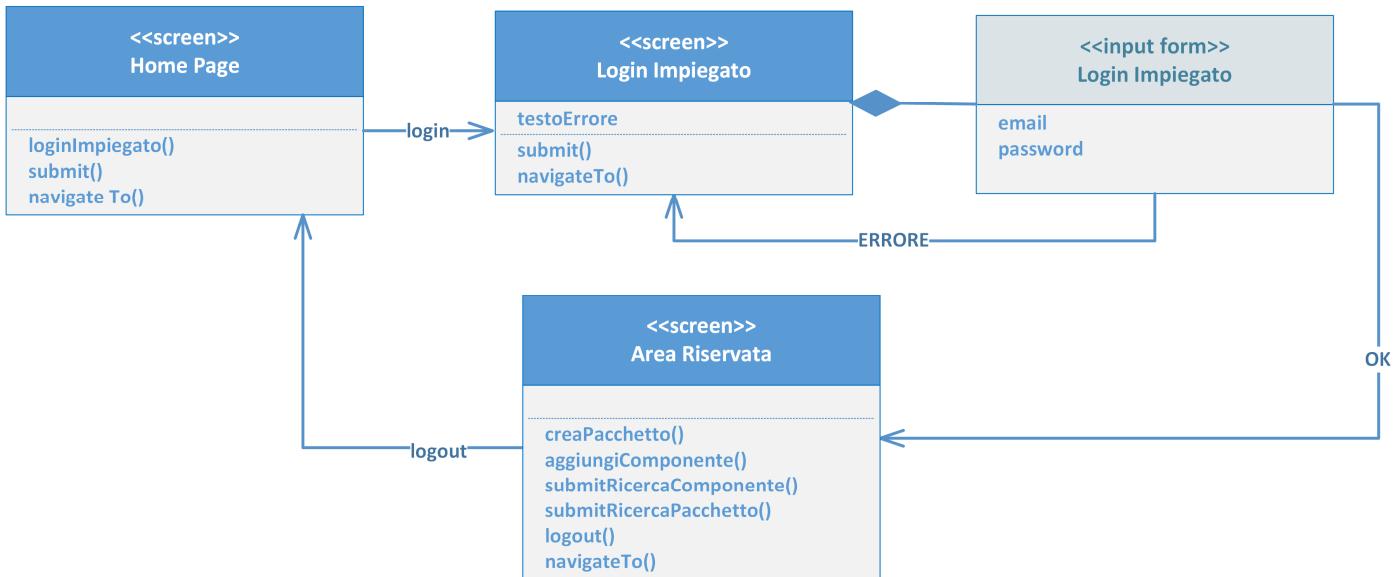


Fig. 4.6 – UXModel Gestione dei pacchetti da parte di un cliente



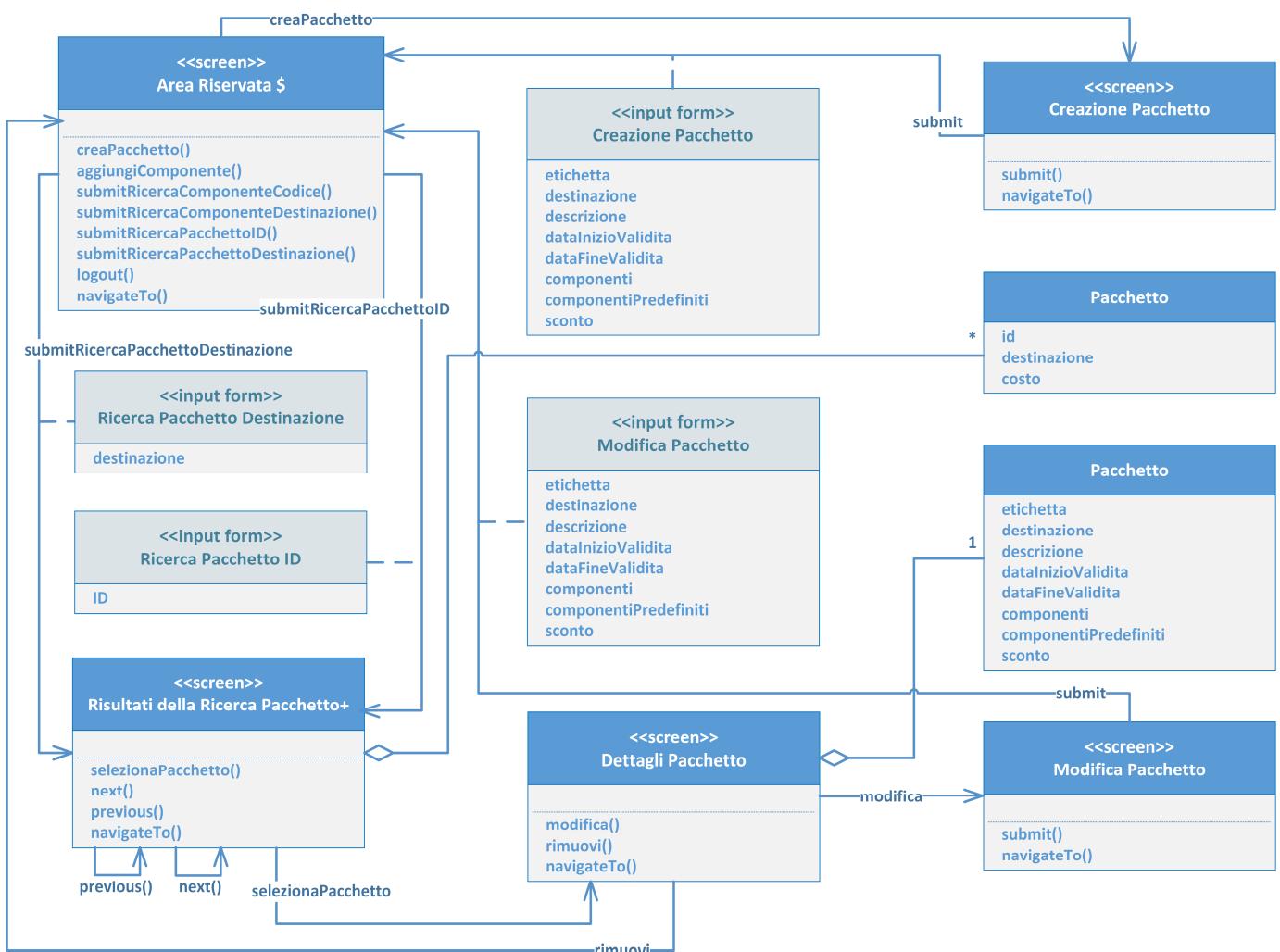


Fig. 4.9 – UXModel Gestione di pacchetti predefiniti da parte di un impiegato

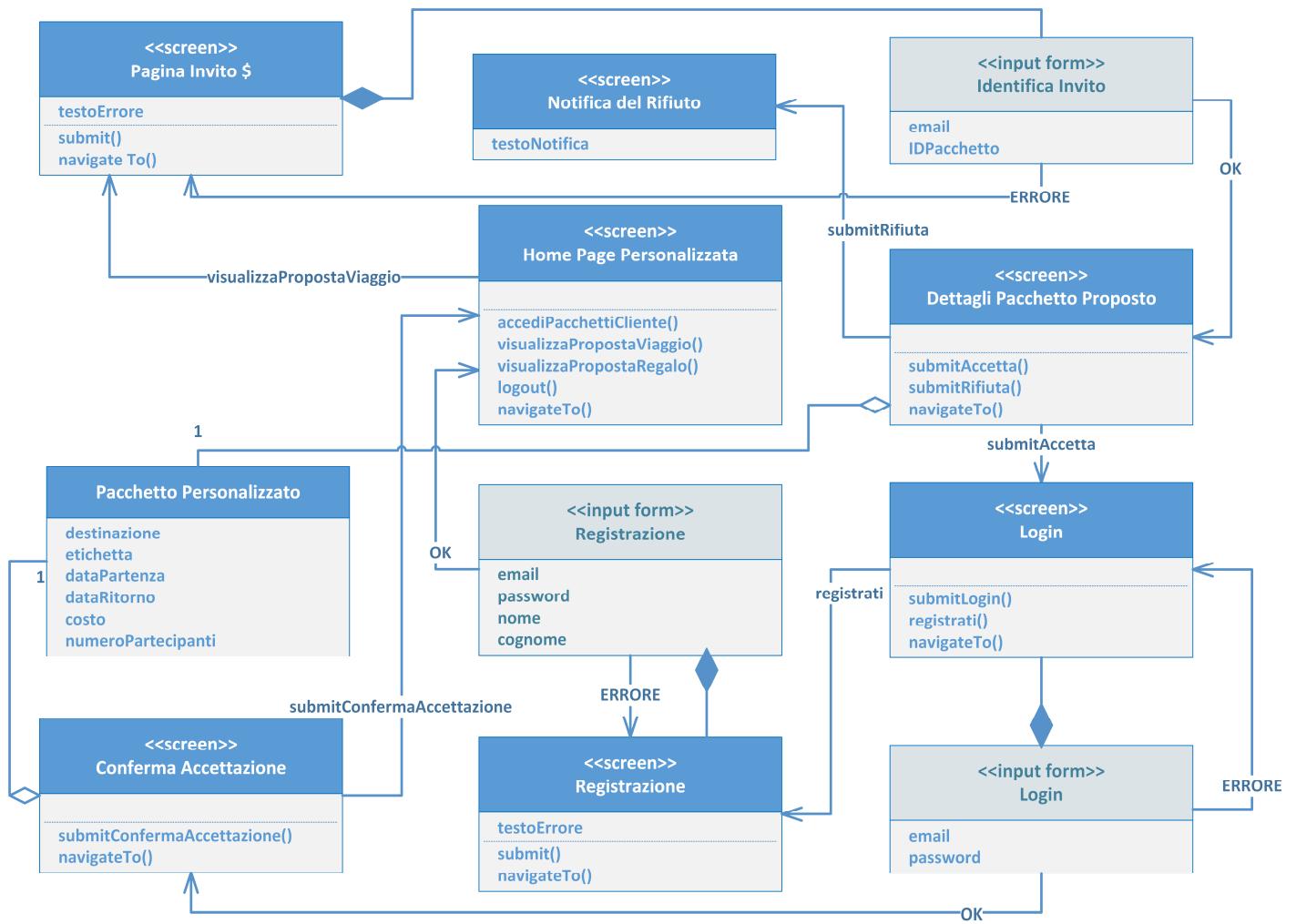


Fig. 4.10 – UXModel Accettazione o rifiuto alla partecipazione a un viaggio da parte di un amico

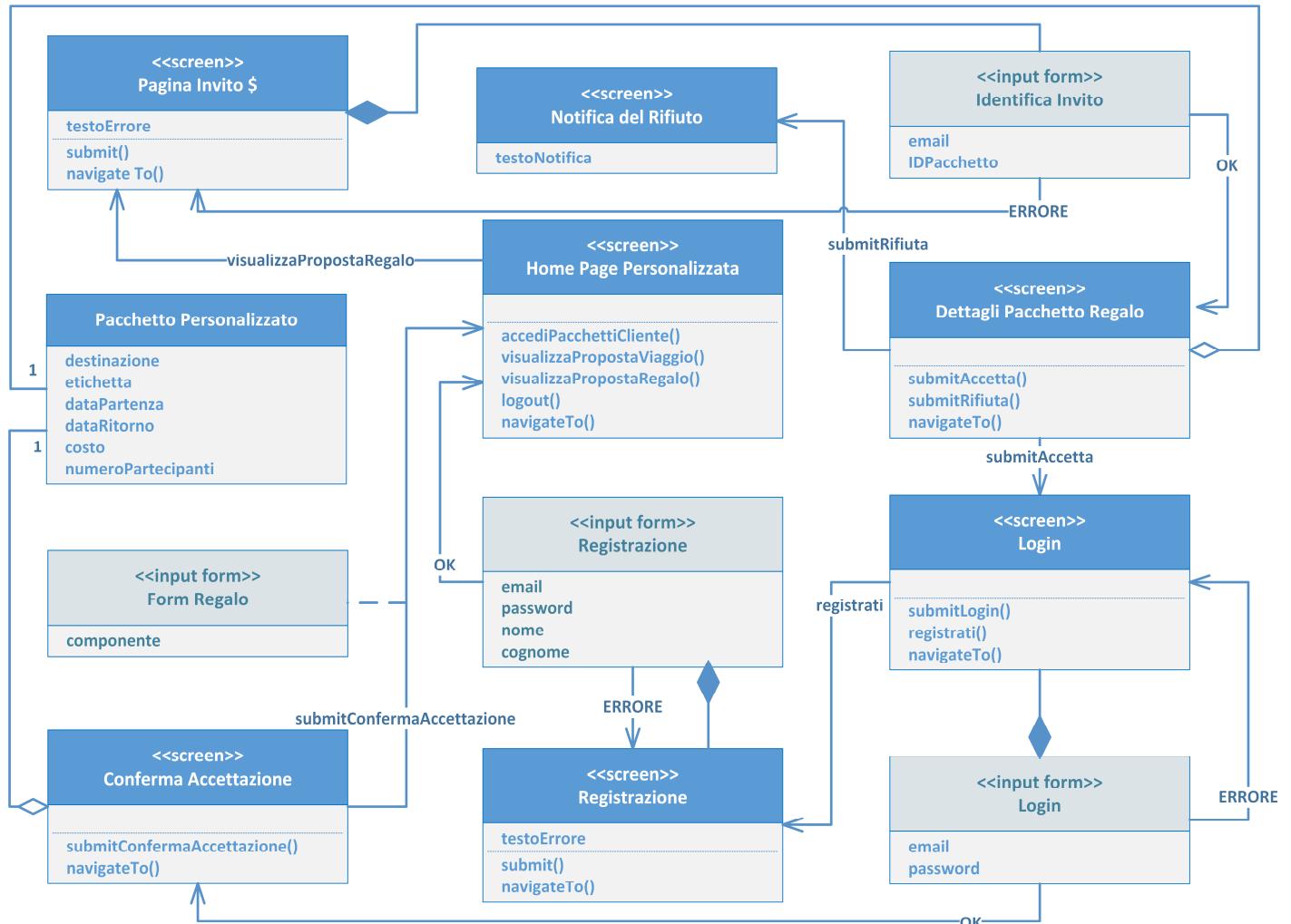


Fig. 4.11 – UXModel Accettazione o rifiuto di una proposta regalo da parte di un amico

4.5. Altri diagrammi

4.5.1. Diagramma di analisi (BCE)

Il diagramma Boundary-Control-Entity divide la presentazione del sistema in (boundary), controllo (control) e dati (entity).

Questi diagrammi usano il seguente standard:

- Classi: rappresentate come rettangoli con il tipo (<<control>>, <<entity>> o <<boundary>>), il nome, gli attributi e i metodi.
 - <<boundary>>: classe che fa parte dell'interfaccia utente
 - <<entity>>: classe che rappresenta la persistenza dei dati
 - <<control>>: classe che rappresenta la logica del sistema
- Relazioni: associazioni tra classi specificando le cardinalità
- Frecce: indicano

Abbiamo optato per una divisione dei BCE per attore.

Inoltre, per motivi di visualizzazione, il BCE di ogni attore è stato diviso in parti.

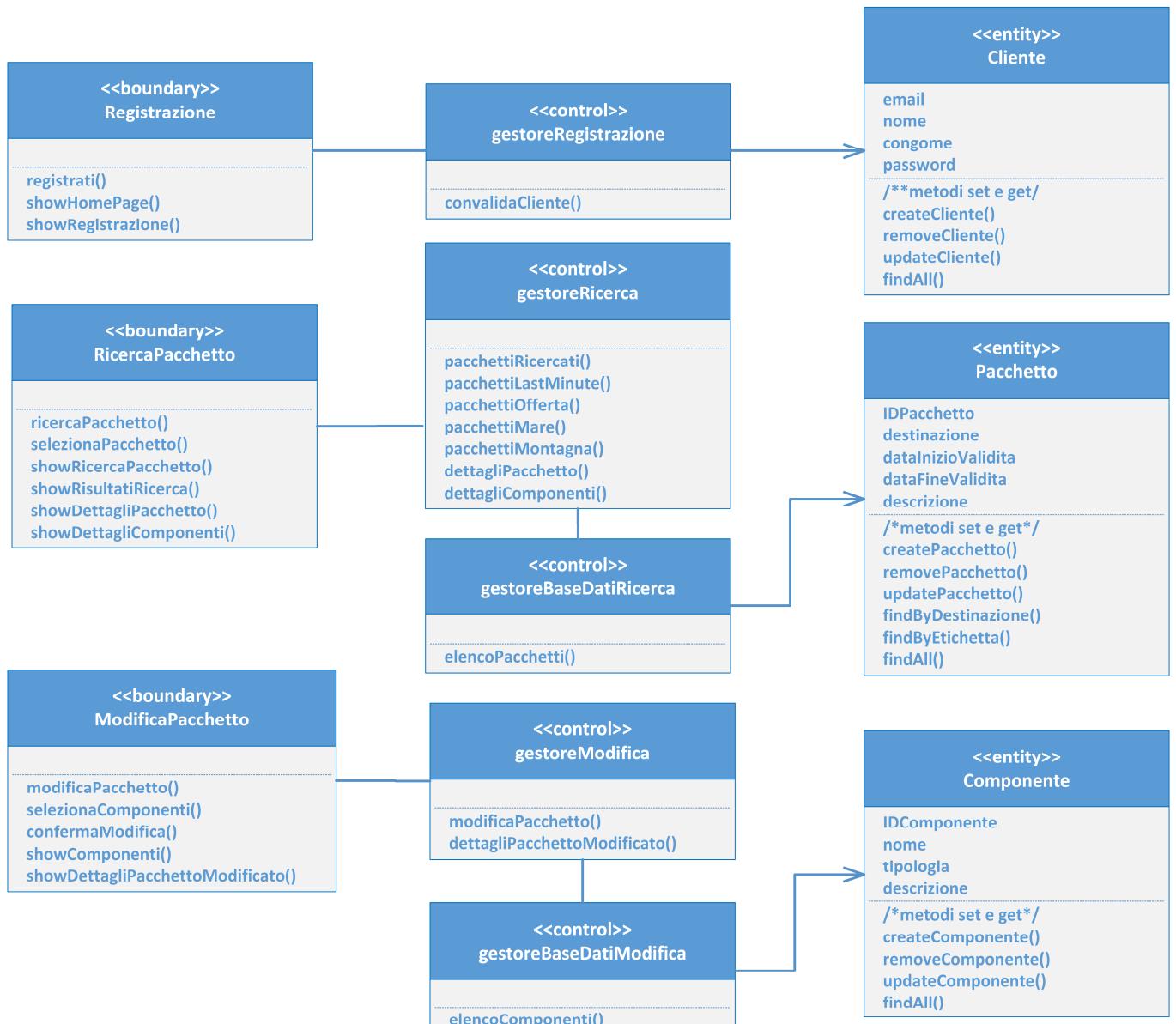


Fig. 4.12 – BCE Model del visitatore

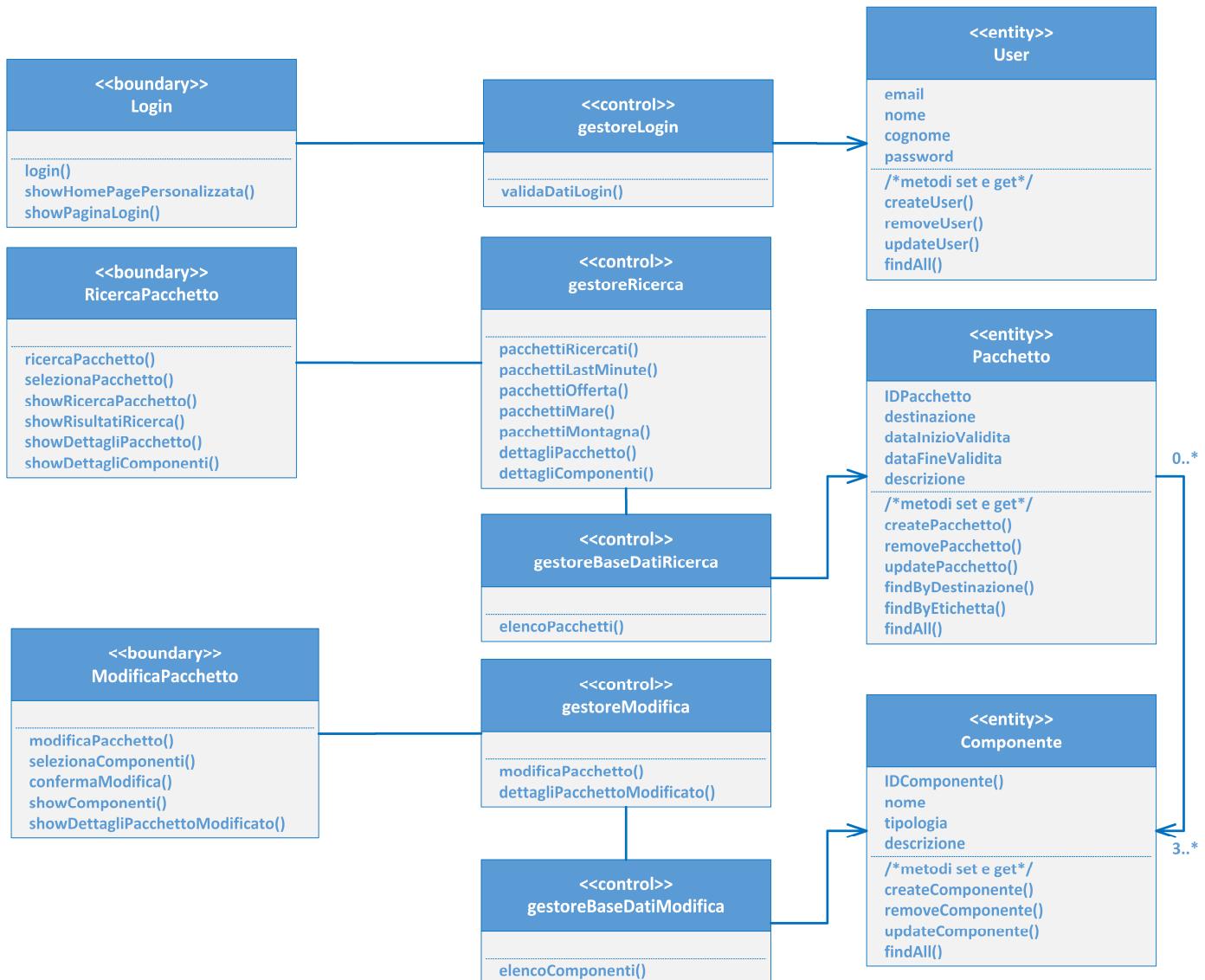


Fig. 4.13 – BCE Model del cliente 1

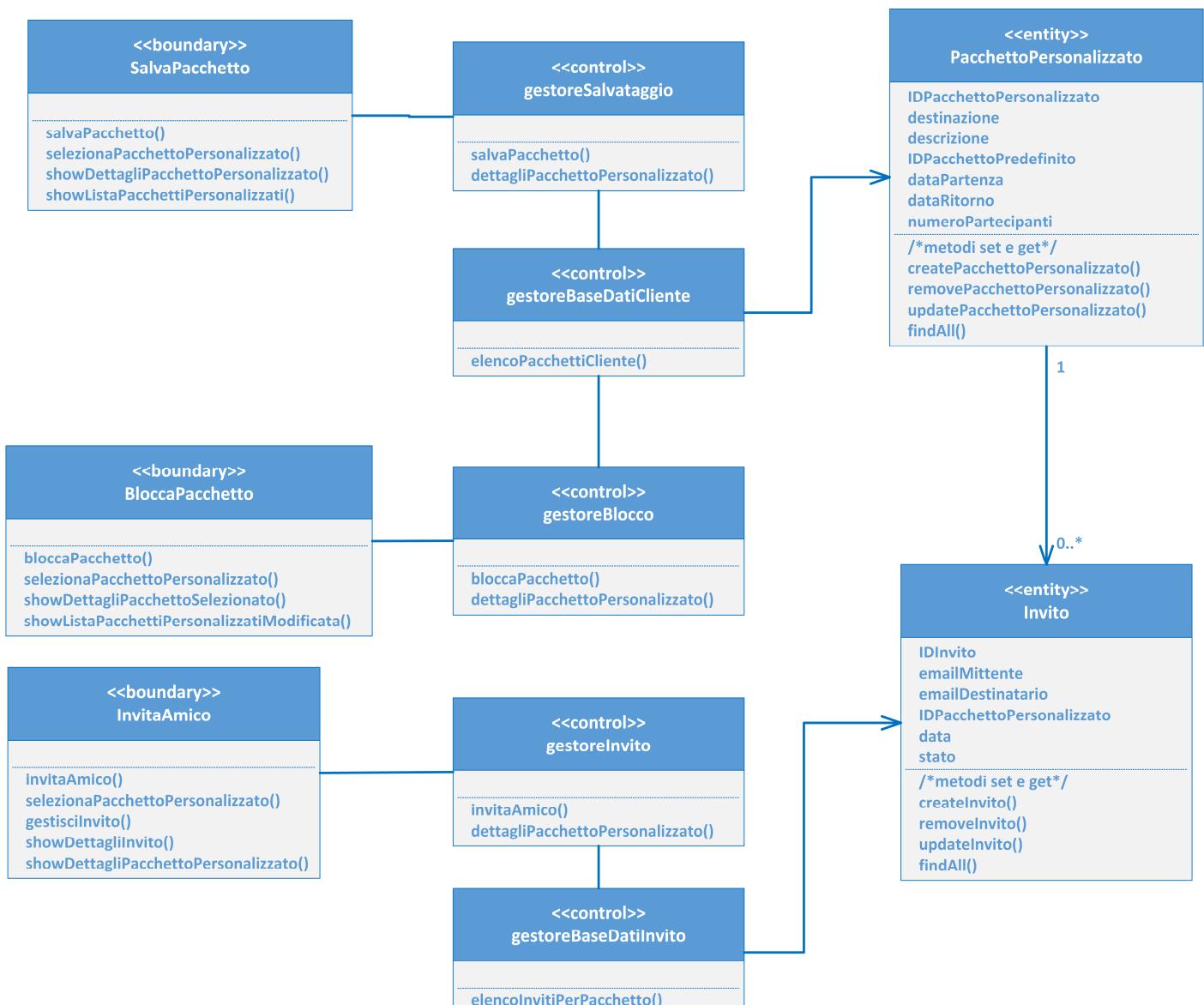


Fig. 4.14 – BCE Model del cliente 2

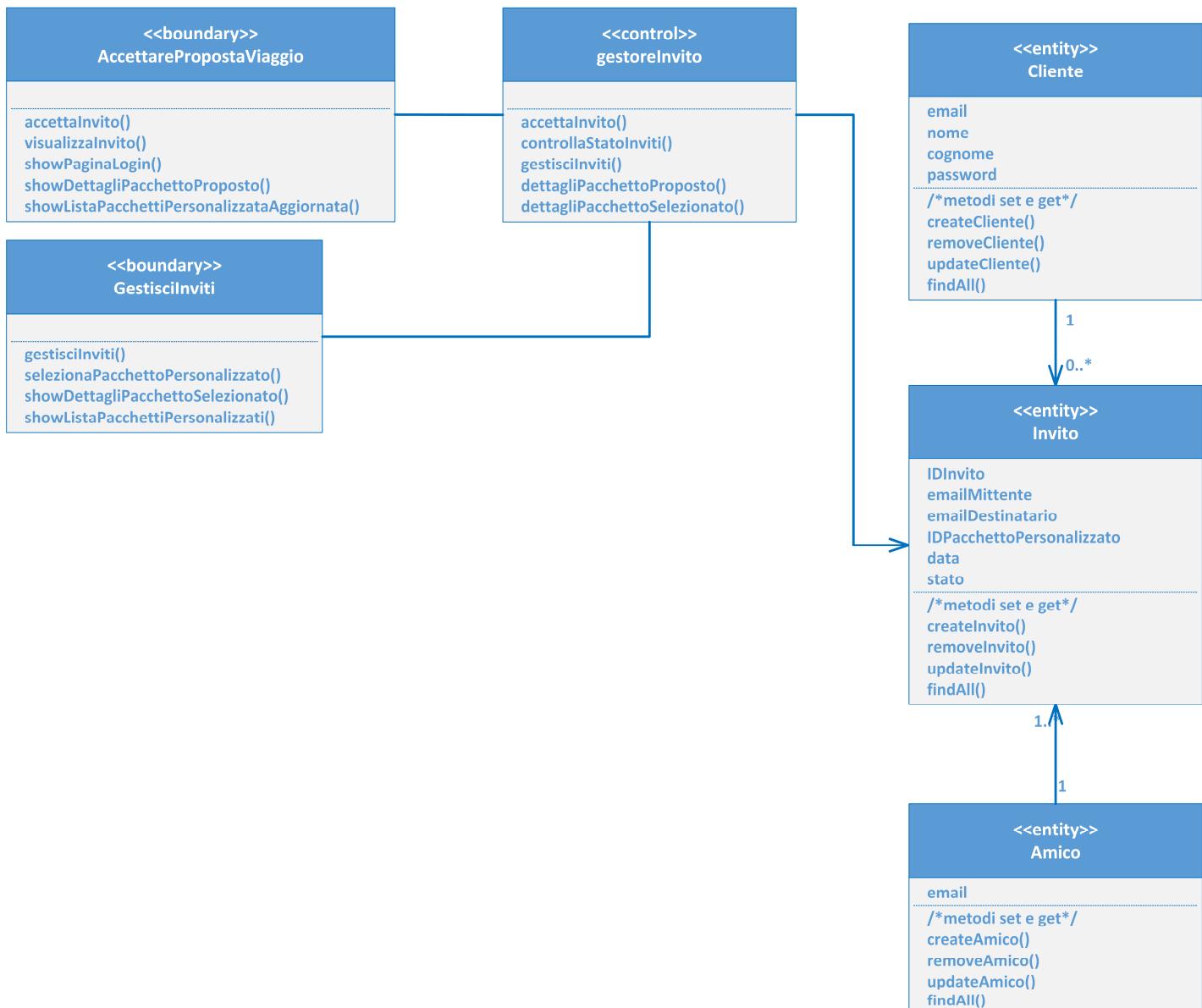


Fig. 4.15 – BCE Model del cliente 3

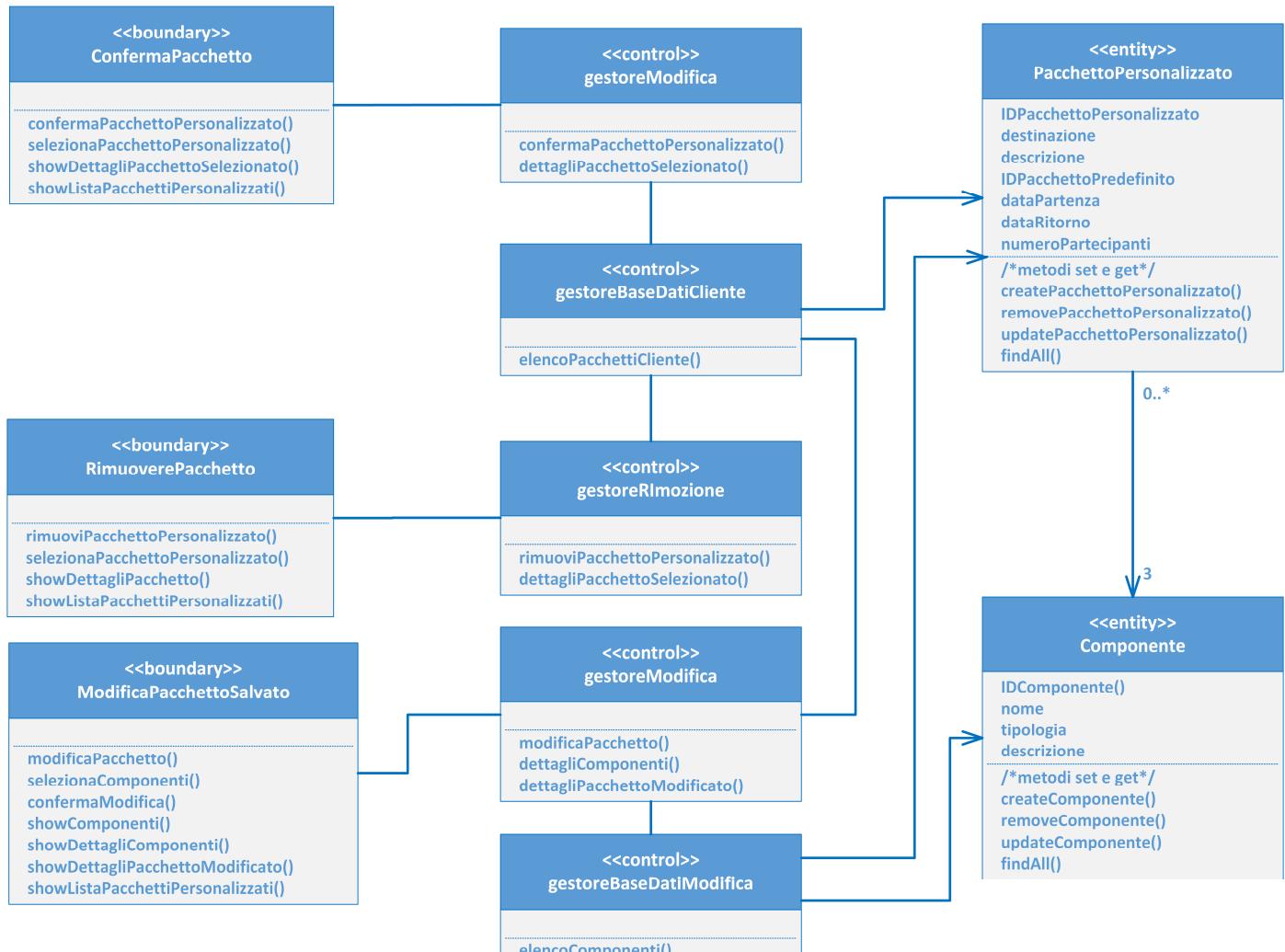


Fig. 4.16 – BCE Model del cliente 4

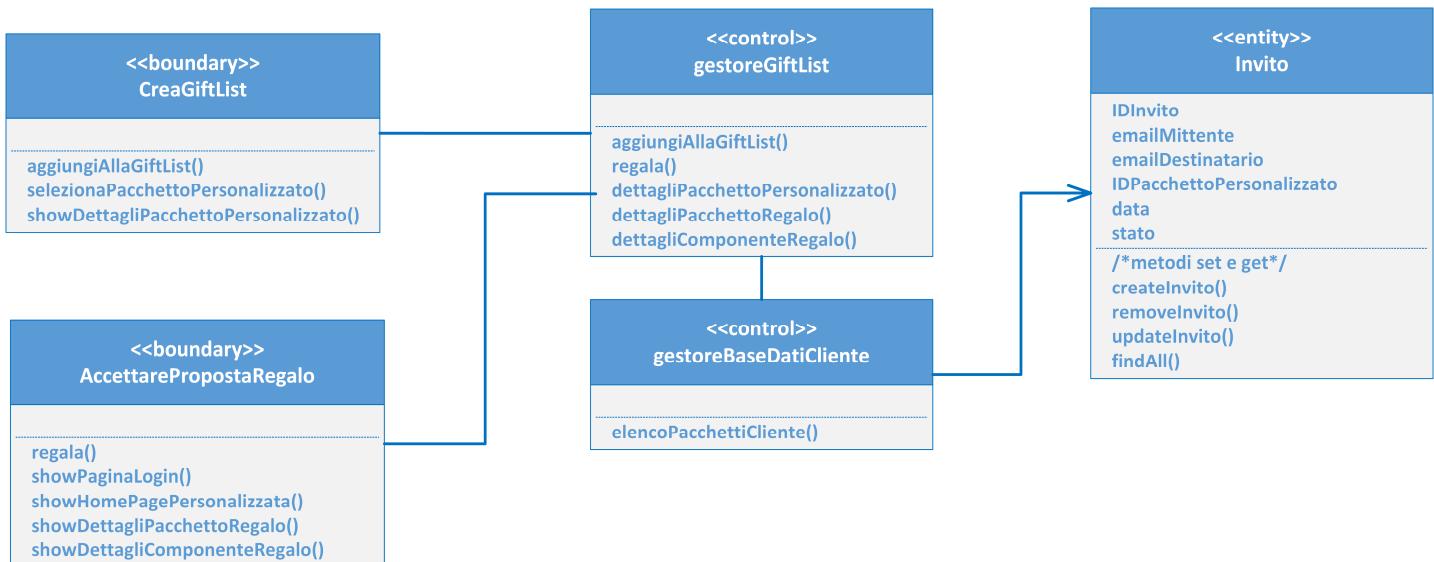


Fig. 4.17 – BCE Model del cliente 5

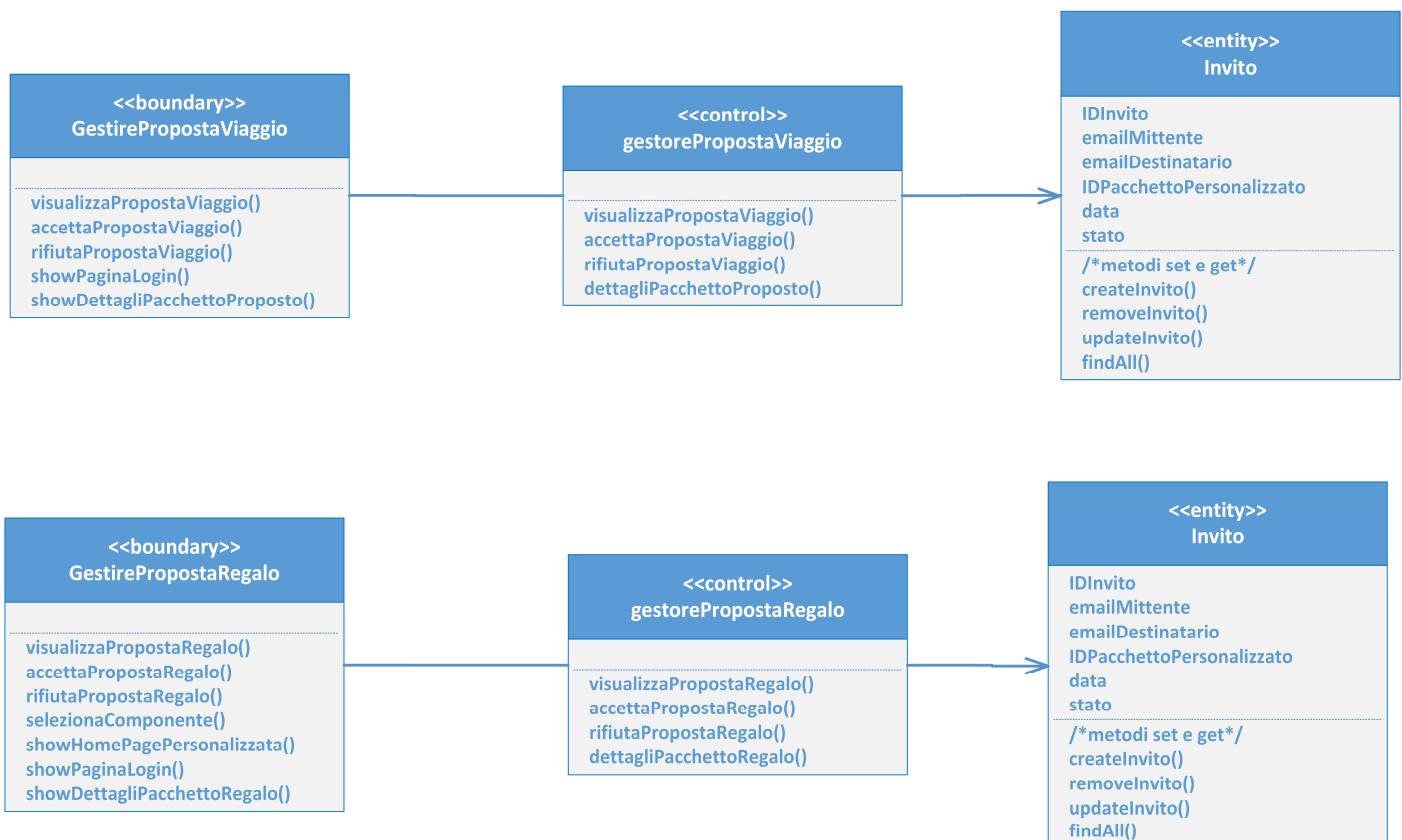


Fig. 4.18 – BCE Model dell'amico

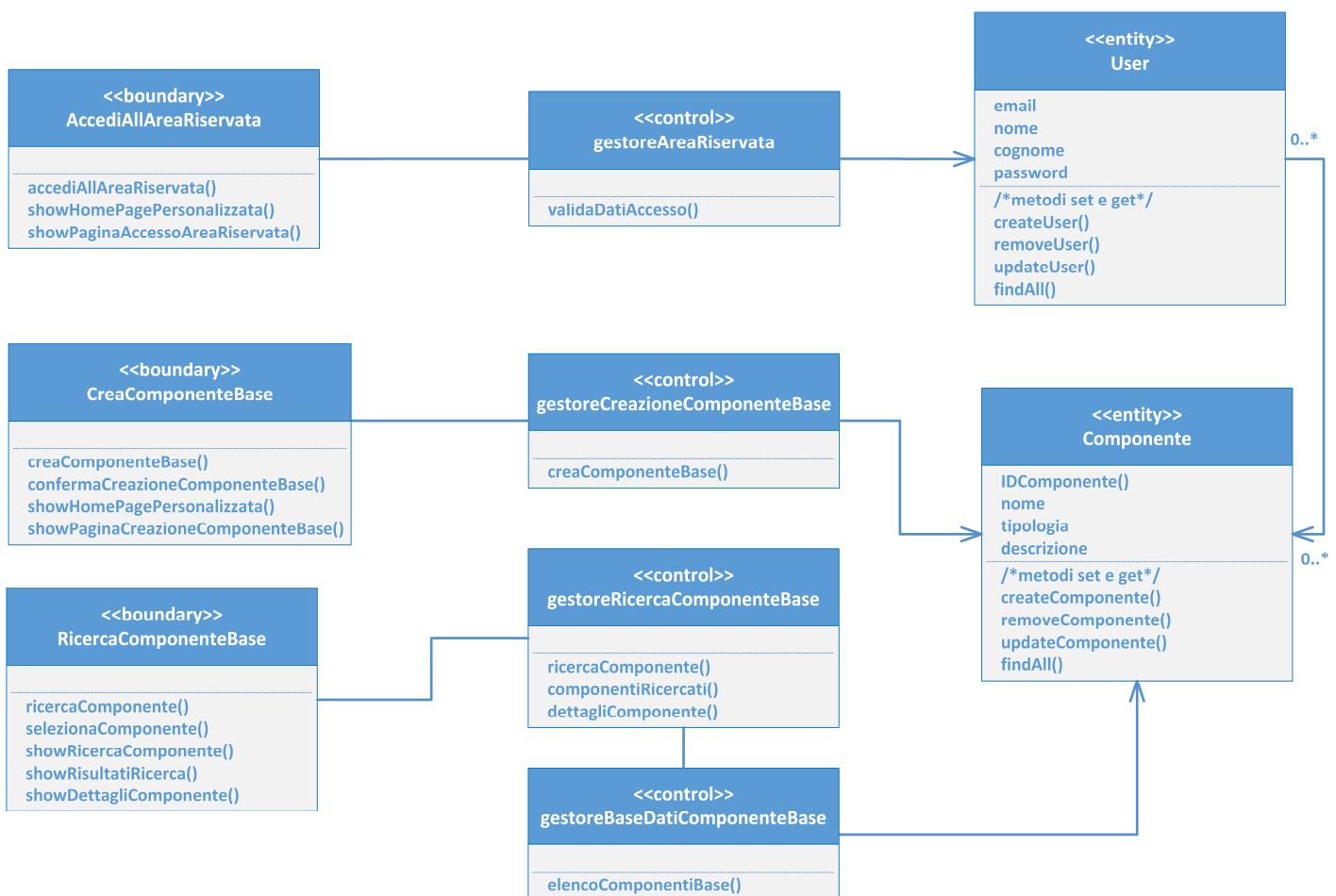


Fig. 4.19 – BCE Model dell'impiegato 1

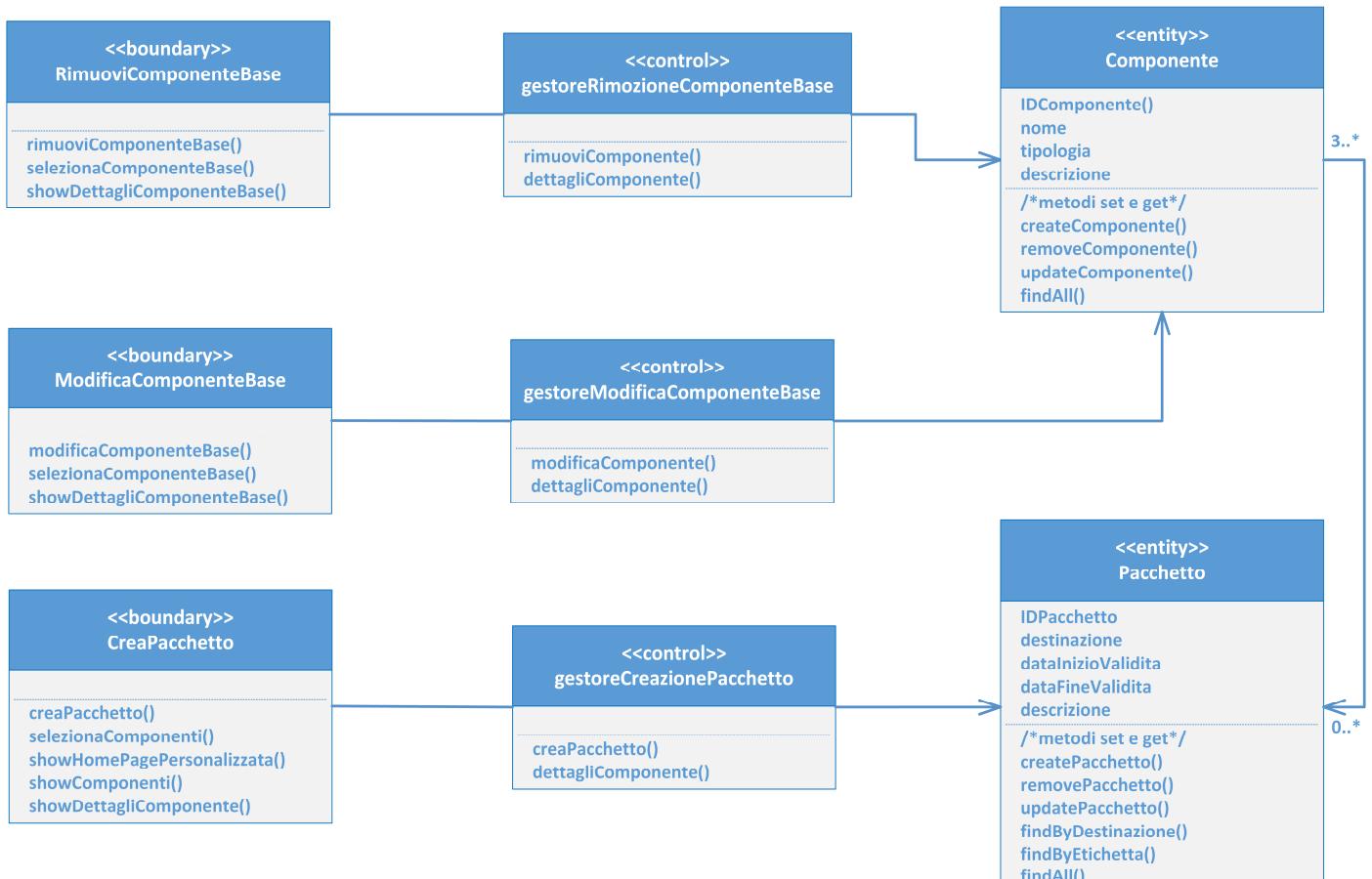


Fig. 4.20 – BCE Model dell'impiegato 2

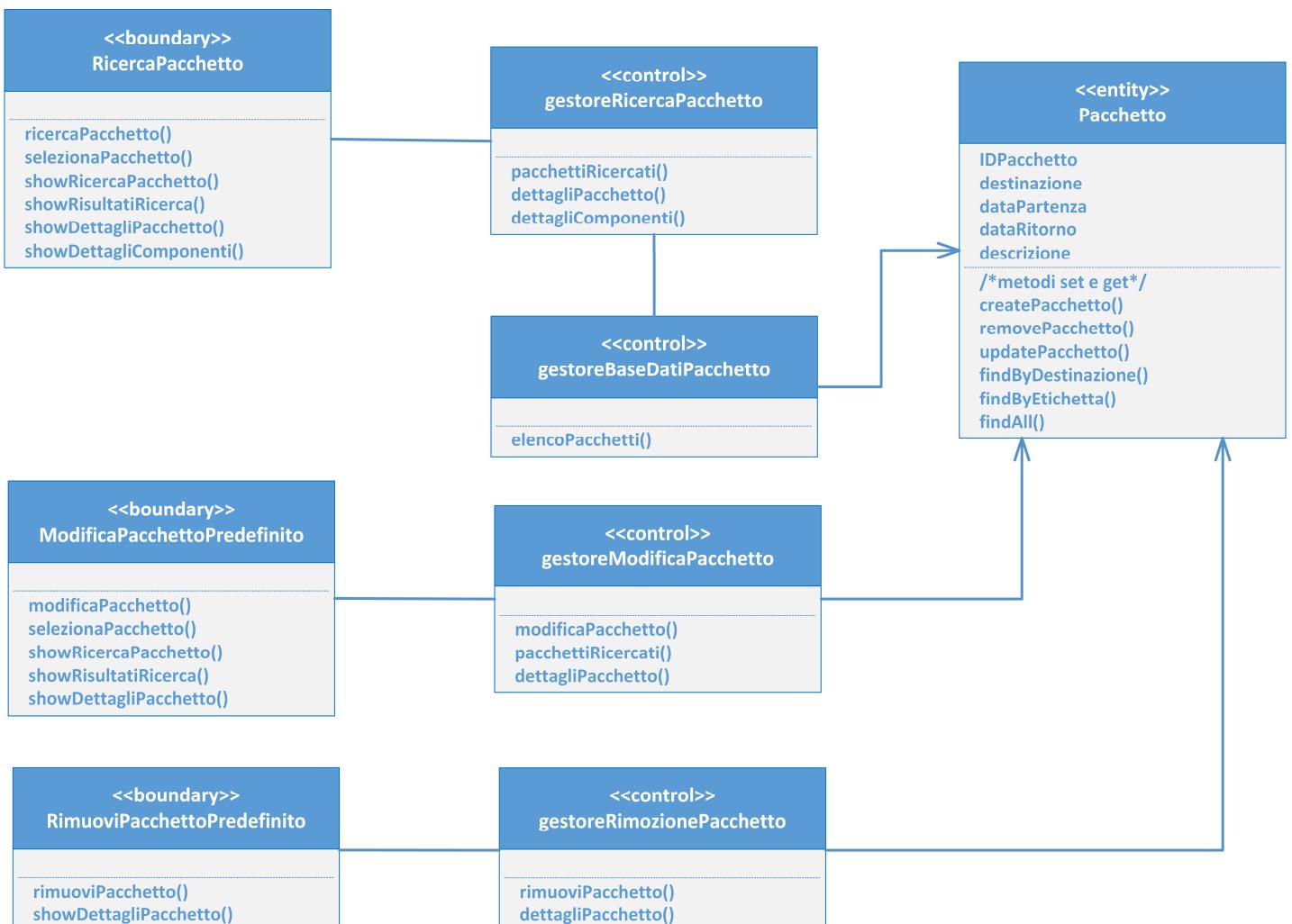


Fig. 4.21 – BCE Model dell'impiegato 3

5. Progetto per JEE

Dopo aver progettato il sistema ad alto livello, è necessario scendere in quello più basso in cui sono specificati gli elementi che saranno implementati direttamente tramite JEE.

Prima sono definiti gli “Entity Beans” per rappresentare i dati e in seguito i “Session Bean” che rappresentano i controlli.

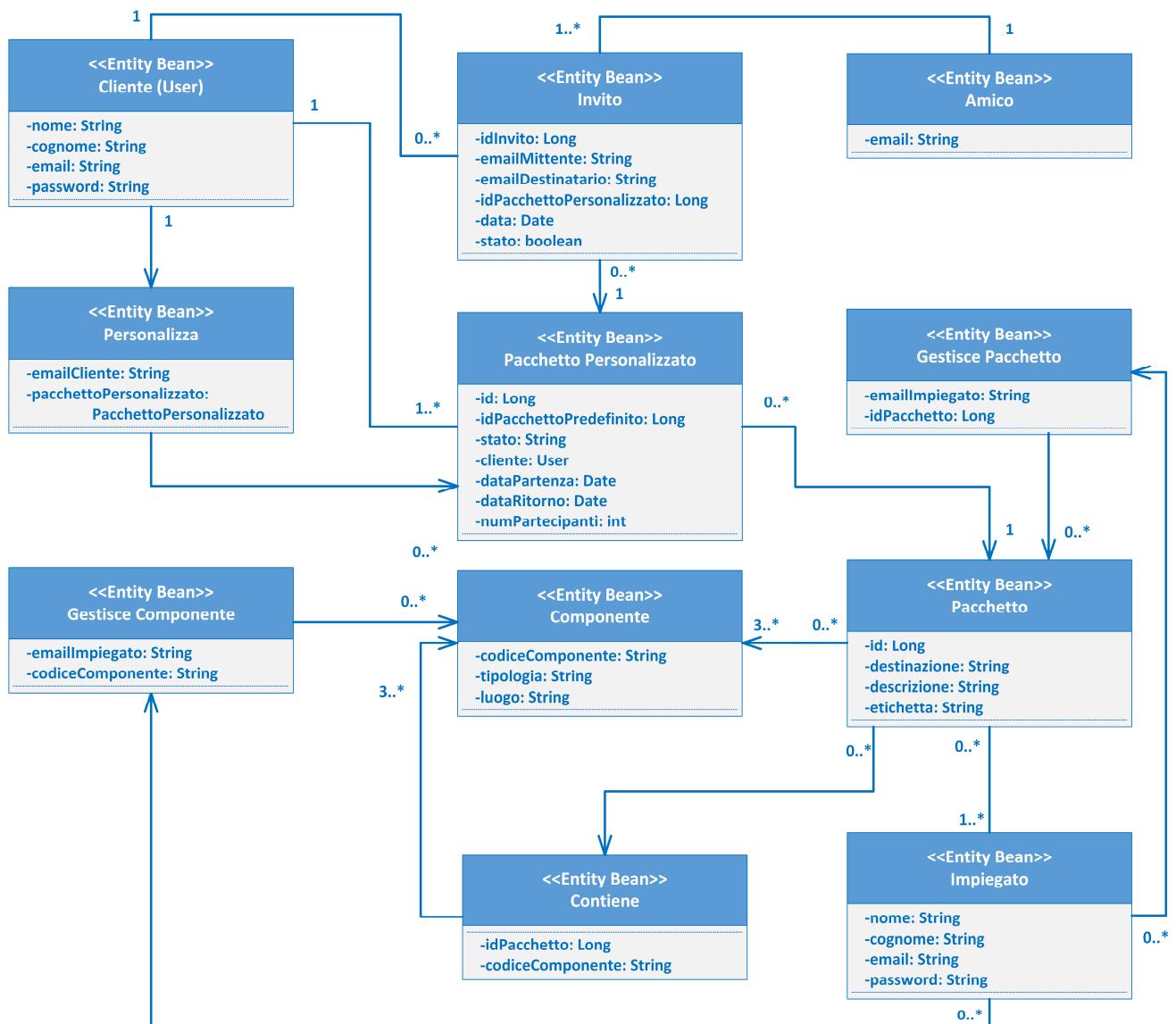


Fig. 5.1 - EntityBeans

Di solito, i “Session Bean” coincidono con i <<control>> del diagramma BCE, compresi i metodi. Un Session Bean può essere di tipo Stateless o Stateful. Sotto sono specificati i nomi e i tipi.

- GestoreRegistrazione (stateless)
- GestoreLogin (stateless)
- GestoreAreaRiservata (stateless)
- GestoreRicerca (stateless)
- GestoreBaseDatiRicerca (stateless)
- GestoreModifica (stateless)
- GestoreBaseDatiModifica (stateless)
- GestoreRimozione (stateless)
- GestoreSalvataggio (stateless)
- GestoreBaseDatiCliente (stateless)
- GestoreBlocco (stateless)
- GestoreInvito (stateless)
- GestoreBaseDatiInvito (stateless)
- GestoreGiftList (stateless)
- GestoreBaseDatiInvito (stateless)
- GestorePropostaViaggio (stateless)
- GestorePropostaRegalo (stateless)
- GestoreCreazioneComponenteBase (stateless)
- GestoreRicercaComponenteBase (stateless)
- GestoreBaseDatiComponenteBase (stateless)
- GestoreRimozioneComponenteBase (stateless)
- GestoreModificaComponenteBase (stateless)
- GestoreCreazionePacchetto (stateless)
- GestoreRicercaPacchetto (stateless)
- GestoreBaseDatiPacchetto (stateless)
- GestoreModificaPacchetto (stateless)
- GestoreRimozionePacchetto (stateless)
- GestoreAggiuntaComponentiAPacchetto (stateless)
- GestoreModificaComponenteInPacchetto (stateless)
- GestoreRimozioneComponenteDaPacchetto (stateless)
- GestoreModificaDettagliPacchetto (stateless)

5.1.1. Diagrammi di sequenza

Per descrivere meglio la sequenza di operazioni che avvengono nel sistema sono stati allegati alcuni diagrammi di sequenza.

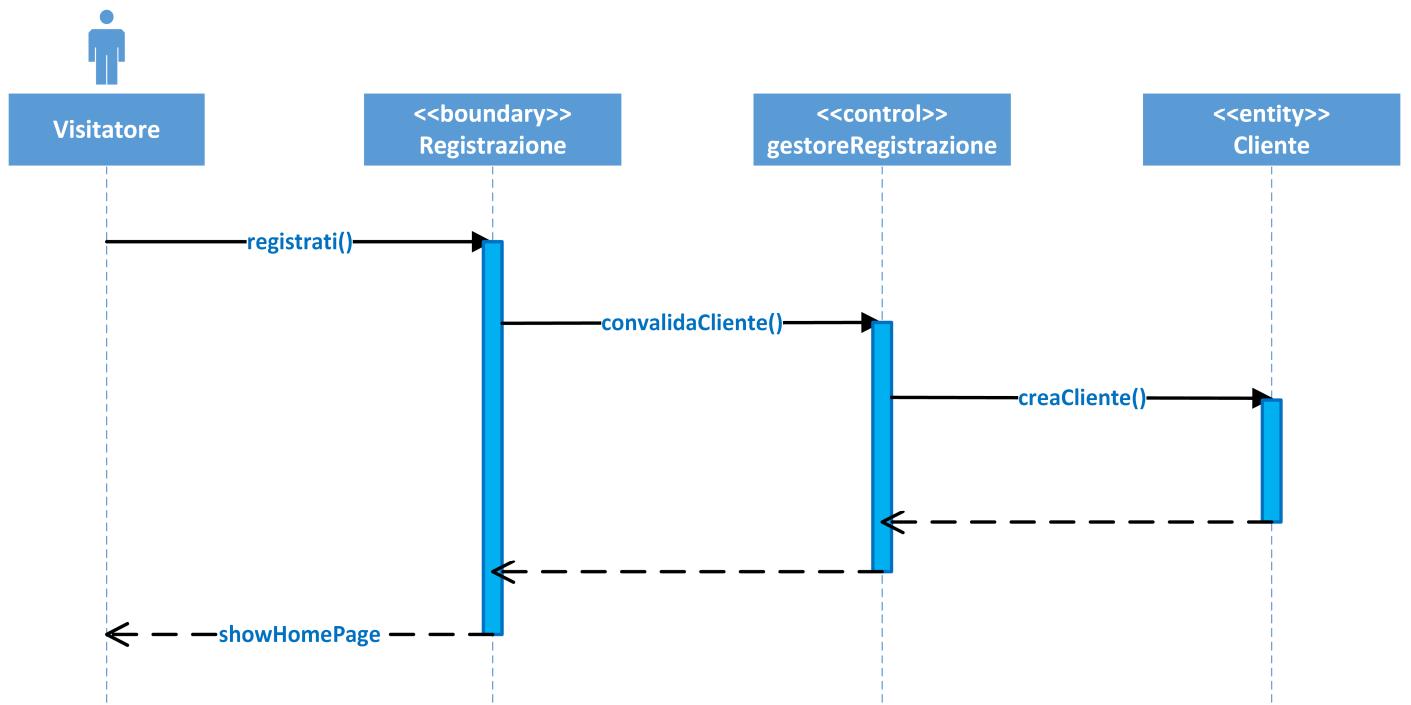


Fig. 5.2 - Sequence Diagram registrazione da parte di un visitatore

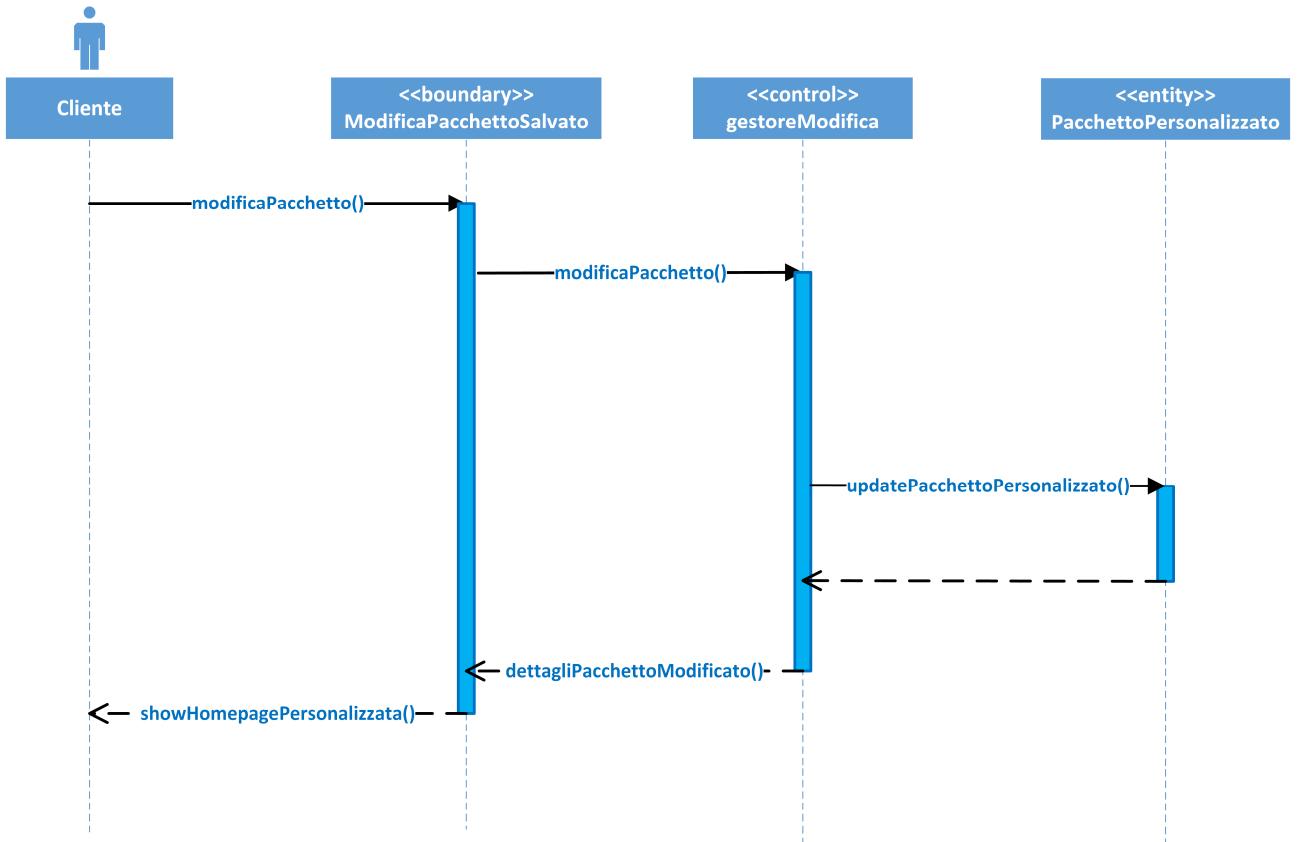


Fig. 5.3 - Sequence Diagram modifica pacchetto personalizzato da parte di un cliente

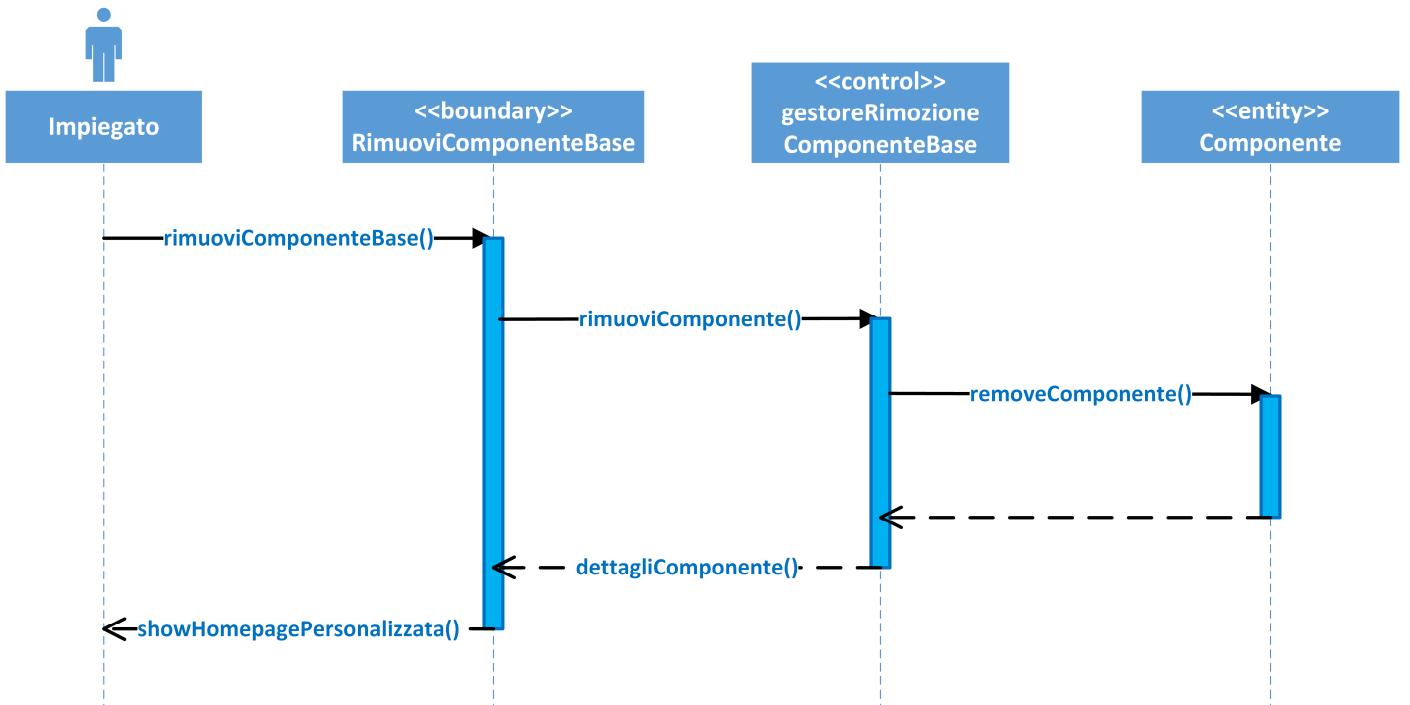


Fig. 5.4 - Sequence Diagram rimozione di un componente da parte di un impiegato

Indice delle figure

Fig. 3.1 – Architettura Three-Tier	7
Fig. 3.2 – Architettura Three-Tier	8
Fig. 4.1 – Schema ER	10
Fig. 4.2 – EER Model.....	14
Fig. 4.3 – Schema Logico	18
Fig. 4.4 – UXModel Registrazione di un visitatore e autenticazione di un cliente	19
Fig. 4.5 – UXModel Ricerca dei pacchetti da parte di un cliente	20
Fig. 4.6 – UXModel Gestione dei pacchetti da parte di un cliente	20
Fig. 4.7 – UXModel Accesso all'area riservata da parte di un impiegato	21
Fig. 4.8 – UXModel Gestione dei componenti base da parte di un impiegato.....	21
Fig. 4.9 – UXModel Gestione di pacchetti predefiniti da parte di un impiegato.....	22
Fig. 4.10 – UXModel Accettazione o rifiuto alla partecipazione a un viaggio da parte di un amico	23
Fig. 4.11 – UXModel Accettazione o rifiuto di una proposta regalo da parte di un amico	24
Fig. 4.12 – BCE Model del visitatore.....	26
Fig. 4.13 – BCE Model del cliente 1.....	27
Fig. 4.14 – BCE Model del cliente 2.....	28
Fig. 4.15 – BCE Model del cliente 3.....	29
Fig. 4.16 – BCE Model del cliente 4.....	30
Fig. 4.17 – BCE Model del cliente 5.....	31
Fig. 4.18 – BCE Model dell'amico.....	31
Fig. 4.19 – BCE Model dell'impiegato 1	32
Fig. 4.20 – BCE Model dell'impiegato 2	33
Fig. 4.21 – BCE Model dell'impiegato 3	34
Fig. 5.1 – EntityBeans.....	35
Fig. 5.2 – Sequence Diagram registrazione da parte di un visitatore	37
Fig. 5.3 – Sequence Diagram modifica pacchetto personalizzato da parte di un cliente.....	38
Fig. 5.4 – Sequence Diagram rimozione di un componente da parte di un impiegato.....	38