

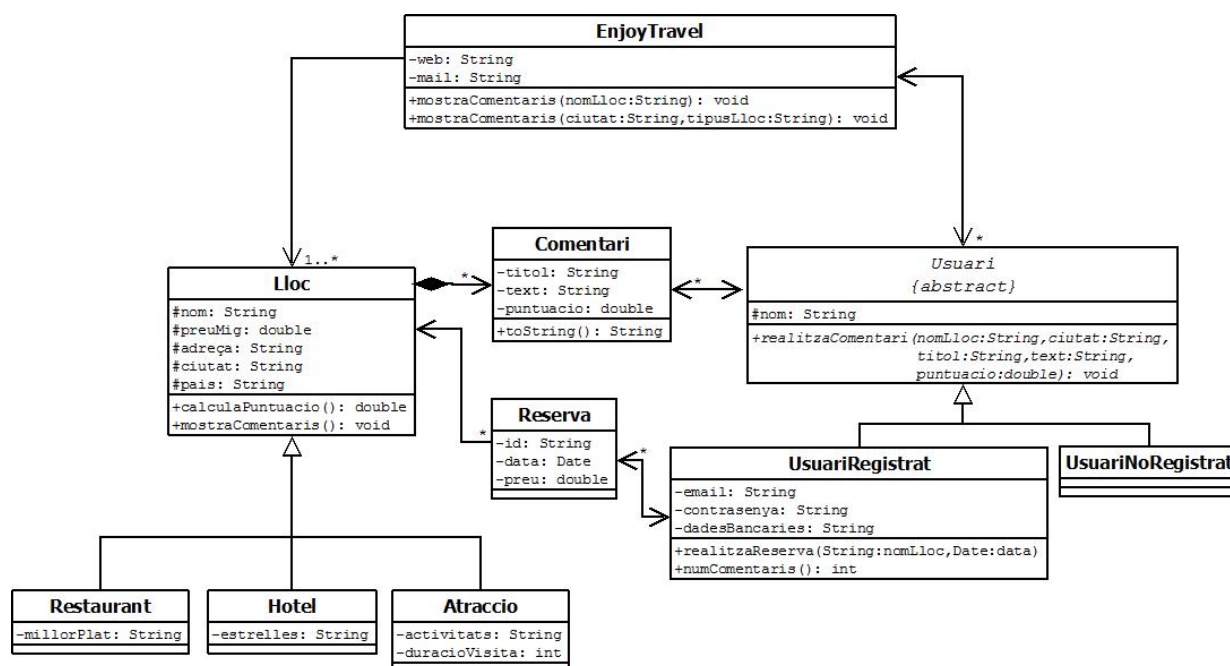
Metodologia i Programació Orientada a Objectes – Examen Parcial Grup 60

Tardor 2013-2014 (20 de Novembre de 2013)

Es vol realitzar la web EnjoyTravel de crítica i reserva de viatges, que permeti a viatgers de tot el món consultar les crítiques d'altres viatgers, respecte als hotels, restaurants i atraccions que ofereixen les ciutats que visitarà, abans de realitzar el seu viatge. Dita web també permetrà als seus usuaris realitzar les reserves que necessiti mentre planifica el viatge, així com realitzar la seves pròpies crítiques.

La web estarà formada pels llocs on han estat els viatgers: restaurants, hotels i atraccions, així com els comentaris que han realitzat els usuaris respecte a dits llocs. A la web podran accedir tot tipus de viatgers, que seran els usuaris, entre els que distingirem els que estan registrats i els que no ho estan, pels darrers el nom de l'usuari serà "anònim". Tant el usuaris registrats com els no registrats podran realitzar i consultar crítiques, mentre que només els registrats podran realitzar reserves.

El diagrama UML resultat de les fases d'anàlisi de requisits i disseny es mostra a continuació.



1. (2 punts) A partir del diagrama UML respon justificadament a les següents preguntes:

- En el diagrama UML pots identificar algun mètode polimòrfic? En cas afirmatiu quines classes implementen dit mètode?

El mètode `realitzaComentari` és polimòrfic, ja que a l'estar a la classe abstracta `Usuari` l'han d'implementar les classes filles, i ha de generar un comentari segons les particularitats de cada tipus d'usuari.

L'implementaran les classes UsuariRegistrat i UsuariNoRegistrat.

- Un usuari no registrat pot realitzar comentaris, però no reserves. La presència o absència de quines relacions indiquen a l'UML que pot fer comentaris? I que no pot fer reserves?

Un usuari registrat pot realitzar comentaris, ja que al ser UsuariRegistrat subclasse de la classe Usuari hereta la relació entre Usuari i Comentari.

Un usuari no registrat no pot realitzar reserves, ja que la classe UsuariNoRegistrat no té relació amb Reserva, ni la classe Usuari de la qual hereta, tampoc té relació amb Reserva.

- L'associació escollida entre les classes Lloc i Comentari ha estat una composició. Per què creus que s'ha escollit aquest tipus de relació? Seria millor haver escollit una agregació?

S'ha escollit una composició o agregació forta ja que quan es destrueix el contenidor els elements continguts també es destrueixen. En aquest cas, al destruir un Lloc també es destruïen els seus comentaris. En cas d'haver escollit una agregació això no succeiria, i mantindríem els comentaris d'un lloc que no existeix.

- Podem conèixer la puntuació dels diferent llocs als que han estat els usuaris de la Web EnjoyTravel? Creus que falta l'atribut puntuació? En cas contrari, explica com conèixer la puntuació d'un lloc (restaurant, hotel o atracció).

Sí, ja que el mètode calculaPuntuacio() de la classe Lloc calcula la puntuació per a un lloc de la web EnjoyTravel

No caldria un atribut, ja que des de la classe Lloc podem accedir a tots els comentaris que han realitzat els usuaris per a un lloc i calcular la mitjana a través del mètode calculaPuntuacio() de la classe Lloc

2. **(2,5 punts)** Escriu el codi Java per a totes les classes del diagrama UML. Codifica només la part corresponent a la definició les classes, és a dir els seus atributs i la implementació de les relacions entre classes.

No es necessari que defineixis els constructors ni els mètodes.

```
public class EnjoyTravel {
    private String web;
    private String email;
    private Map<String,Lloc> llocs;
    private Set<Usuari> usuaris;
}

public class Lloc {
    protected String nom;
    protected double preuMig;
    protected String adreça;
    protected String ciutat;
```

```

        protected String pais;
        private List<Comentari> comentaris;
    }

    public class Restaurant extends Lloc{
        private String millorPlat;
    }

    public class Hotel extends Lloc{
        private String estrelles;
    }

    public class Atraccio extends Lloc{
        private String activitats;
        private int duracioVisita;
    }

    public abstract class Usuari{
        protected String nom;
        protected EnjoyTravel travel;
        protected List<Comentari> comentaris;
    }

    public class UsuariRegistrat extends Usuari{
        private String email;
        private String contrasenya;
        private String dadesBancaries;
        private List<Reserva> reserves;
    }

    public class UsuariNoRegistrat extends Usuari{...}

    public class Comentari {
        private String titol;
        private String text;
        private double puntuacio;
        private Usuari usuari;
    }

    public class Reserva {
        private String id;
        private Date data;
        private double preu;
        private Lloc lloc;
        private UsuariRegistrat usuariRegistrat;
    }

```

3. **(1 punts)** Implementa el mètodes constructors de la classe abstracta Usuari i de les seves subclasses UsuariRegistrat i UsuariNoRegistrat.

```

public abstract class Usuari {
    public Usuari(String nom){
        this.nom=nom;
        comentaris=new ArrayList<Comentari>();
    }
}

public class UsuariNoRegistrat extends Usuari{
    public UsuariNoRegistrat(String nom){
        super(nom);
    }
}

public class UsuariRegistrat extends Usuari{

```

```

    public UsuariRegistrat(String nom, String email, String contrasenya){
        super(nom);
        this.email=email;
        this.contrasenya=contrasenya;
        reserves=new ArrayList<Reserva>();
    }
}

```

4. **(1,5 punt)** Implementa el mètode de la classe Lloc:

+calculaPuntuacio():double

Aquest mètode retorna la puntuació promig d'un lloc en funció de tots els comentaris rebuts. Suposa que els mètodes d'accés (getters) de la classe Comentari ja estan implementats.

```

public class Lloc {
    public double calculaPuntuacio(){
        double puntuacio=0.0;
        Iterator<Comentari> it=comentarios.iterator();
        while(it.hasNext()){
            puntuacio+=it.next().getPuntuacio();
        }
        return puntuacio/comentarios.size();
    }
}

```

5. **(1 punt)** Implementa el mètode de la classe EnjoyTravel:

+ mostraComentarios(nomLloc:String):void

Aquest mètode mostrarà tots els comentaris que han realitzat els usuaris per al lloc que se li passa com a paràmetre.

Aquest mètode farà ús de +mostraComentarios():void de la classe Lloc que també haureu d'implementar.

```

public class EnjoyTravel {
    public void mostraComentarios(String nomLloc){
        llocs.get(nomLloc).mostraComentarios();
    }
}
public class Lloc {
    public void mostraComentarios(){
        Iterator<Comentari> it=comentarios.iterator();
        while(it.hasNext()){
            System.out.println(it.next().toString());
        }
    }
}

```

6. **(2 punts)** Implementa el mètode de la classe UsuariRegistrat:

+realitzaComentari(nomLloc:String, ciutat:String, titol:String, text:String, puntuacio:double):void

Aquest mètode rep com a paràmetres el nom del lloc que l'usuari vol comentar, la ciutat a la que es troba, i el títol, text i puntuació del comentari introduït per l'usuari.

Aquest mètode crearà el comentari, i l'inclourà entre els comentaris del lloc i els de l'usuari.

Suposa que la classe Lloc ja té implementat el constructor:

```
public Lloc(String nom, String ciutat);
```

Suposa que la classe Comentari ja té implementat el constructor:

```
        public Comentari(String titol, String text, double puntuacio);
```

Suposa que els mètodes d'accés (getters) de les classes EnjoyTravel i Lloc estan implementats.

```
public class UsuariRegistrat extends Usuari{
    public void realitzaComentari(String nomLloc, String ciutat, String
    titol, String text, double puntuacio){
        Lloc lloc=travel.getLlocs().get(nomLloc);
        if(lloc==null )
            lloc=new Lloc(nomLloc, ciutat);
        else if(lloc.getCiutat()!=ciutat)
            lloc=new Lloc(nomLloc, ciutat);
        Comentari c=new Comentari(titol, text, puntuacio);
        lloc.getComentaris().add(c);
        comentaris.add(c);
    }
}
```