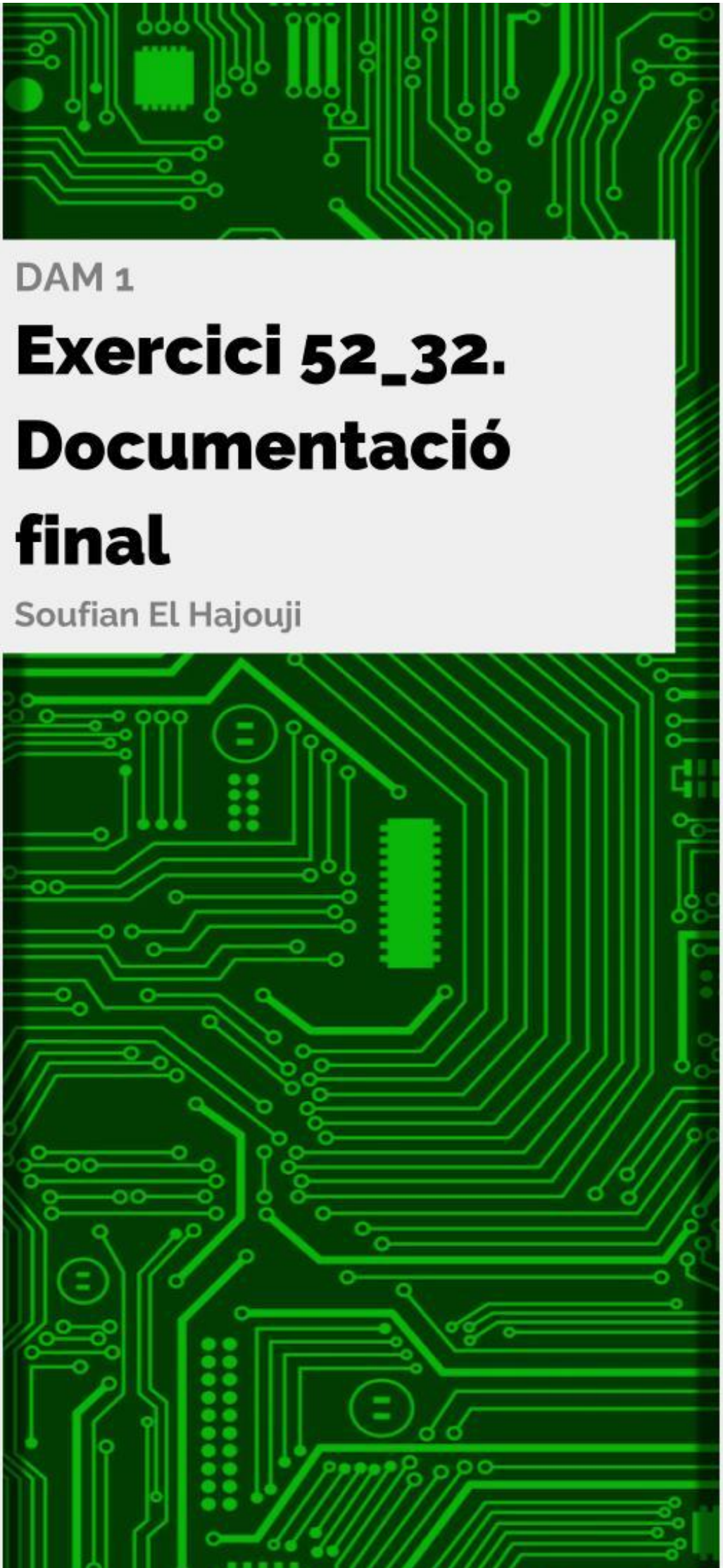

DAM 1

Exercici 52_32. Documentació final

Soufian El Hajouji

ÍNDEX

Exercici 52_12. Documentació de la refacció.....2
Exercici 52_16. Documentem l'extracció.....4
Exercici 52_18. Documentem el moviment.....5
Exercici 52_22. Actualització de la documentació.....7
Exercici 52_25. Canvis en els diagrames UML.....9
Exercici 52_27. Canvis en els diagrames UML.....11
Exercici 52_32. Documentació final..... 13

Exercici 52_12. Documentació de la refacció

Enunciat

En aquest exercici començarem a documentar els canvis que realitzarem al nostre projecte.

Crea un document una mica formal (amb portada, taula de continguts, secció d'introducció i conclusions) i afegeix una secció on incloguis el diagrama de classes, i el de seqüència d'inici del projecte de refacció.

Pots aprofitar la secció de conclusions per descriure les teves reflexions respecte l'estat actual del projecte.

Aquest document l'aniràs completant a mida que vagis avançant en els exercicis.

Introducció

Aquest document descriu el procés de refacció d'un sistema de gestió de lloguer de vehicles. Aquest sistema ja existeix i ha estat desenvolupat però, degut a les necessitats dels usuaris, es requereix una revisió completa i una millora en la seva estructura per aconseguir una millor eficiència i mantenibilitat.

El sistema actual consisteix en tres classes: Client, Lloguer i Vehicle. La classe Client conté informació dels clients com el NIF, el nom i el telèfon. La classe Vehicle conté informació dels vehicles com la marca, el model i la categoria, i la classe Lloguer conté informació sobre els lloguers, incloent el vehicle llogat i la durada del lloguer.

Diagrama de Classes

El diagrama de classes del sistema de gestió de lloguer de vehicles és el següent:

```
class Vehicle {
   + {static} BASIC = 1: int
   + {static} GENERAL = 2: int
   + {static} LUXE = 3: int
   - model: String
   - marca: String
   - categoria: int
   + Vehicle(marca, model, categoria)
}

class Lloguer {
   - dies: int
   + Lloguer(vehicle, dies)
}

class Client {
   - nif: String
   - nom: String
   - telèfon: String
```

```
    - lloguers: ArrayList<Lloguer>
    + Client(nif, nom, telefon)
    + getNif(): String
    + getNom(): String
    + getTelefon(): String
    + informe(): String
}

Vehicle <-left- Lloguer
Lloguer <-left- Client: *
```

Diagrama de Seqüència

El diagrama de seqüència a continuació mostra la seqüència d'inici del projecte de refacció, que comença amb un usuari que sol·licita un informe sobre un client.

```
actor Usuari
Usuari -> unClient: informe()
unClient -> unLloguer: getVehicle()
note left: per cada lloguer
unClient -> unVehicle: getCategoria()
unClient -> unLloguer: getDies()
```

Conclusions

En resum, el sistema de gestió de lloguer de vehicles existent necessita una refacció completa per millorar la seva eficiència i mantenibilitat. Els diagrames de classes i seqüència són una eina útil per ajudar en la planificació del projecte de refacció, i el diagrama de classes mostra les classes que s'han de desenvolupar i les seves relacions.

Exercici 52_16. Documentem l'extracció

Enunciat

Afegeix una nova secció a la documentació on descriuràs els canvis realitzats al codi amb el moviment de mètodes de l'exercici anterior.

Aquest canvi segurament ha modificat el diagrama de classes i/o de seqüència. Redibuixa els diagrames que hagin canviat perquè reflexin la nova situació.

Aprofita la secció de conclusions per descriure les teves reflexions respecte aquest canvi.

s'ha afegit un nou mètode a la classe Client anomenat "quantitatPerLloguer", que permet saber el preu que ha de pagar cada client pel lloguer. A més, s'ha omplert el mètode "informe" a la mateixa classe, que proporciona informació detallada sobre els lloguers realitzats per un client determinat.

El diagrama de classes també s'ha actualitzat per reflectir aquestes noves funcionalitats. Ara, la classe Client té un mètode addicional i la classe Lloguer té una relació bidireccional amb la classe Client.

Finalment, el diagrama de seqüència també s'ha actualitzat per reflectir la crida al mètode "quantitatPerLloguer" per part del client. Això és important perquè ara el client pot saber el preu que ha de pagar per cada lloguer realitzat.

Diagrama de Classes actualitzat:

class Vehicle {
+ {static} BASIC = 1: int
+ {static} GENERAL = 2: int
+ {static} LUXE = 3: int
- model: String
- marca: String
- categoria: int
+ Vehicle(marca, model, categoria)
}

class Lloguer {
- dies: int
+ Lloguer(vehicle, dies)
}

class Client {
- nif: String
- nom: String
- telèfon: String

- lloguers: ArrayList<Lloguer>
+ Client(nif, nom, telefon)
- {static} quantitatPerLloguer(Lloguer): double
+ informe(): String
}

Vehicle <-left- Lloguer
Lloguer <-left- Client: *

Vehicle <-left- Lloguer
Lloguer <-left- Client

Diagrama de Seqüència actualitzat:

actor Usuari
Usuari -> unClient: informe()
unClient -> unLloguer: getVehicle()
note left: per cada lloguer
unClient -> unVehicle: getCategoria()
unClient -> unLloguer: getDies()
unClient -> unClient : quantitatPerLloguer()

Exercici 52_18. Documentem el moviment

Enunciat

Afegeix una nova secció a la documentació on descriuràs els canvis realitzats al codi amb el moviment de mètodes de l'exercici anterior.

Aquest canvi segurament ha modificat el diagrama de classes i/o de seqüència. Redibuixa els diagrames que hagin canviat perquè reflexin la nova situació.

Aprofita la secció de conclusions per descriure les teves reflexions respecte aquest canvi.

Amb l'extracció de la funcionalitat de càlcul de preu de lloguer per part del client en el nou mètode quantitatPerLloguer i canviant-li el nom a quantitat, s'ha millorat la cohesió de la classe Lloguer, ja que ara té la responsabilitat exclusiva d'obtenir informació relacionada amb els seus propis lloguers. Això també ajuda a mantenir un acoblament més baix entre les diferents classes del sistema.

Per tant, s'ha modificat la classe Lloguer per afegir el nou mètode “quantitat” i s'ha actualitzat el mètode informe perquè proporcioni informació detallada sobre els lloguers realitzats per un client determinat. A més a més, s'ha actualitzat el diagrama de classes per reflectir aquestes noves funcionalitats, on la classe Lloguer té una relació bidireccional amb la classe Client.

Diagrama de Classes actualitzat:

class Vehicle {
+ {static} BASIC = 1: int
+ {static} GENERAL = 2: int
+ {static} LUXE = 3: int
- model: String
- marca: String
- categoria: int
+ Vehicle(marca, model, categoria)
}

class Lloguer {
- dies: int
+ Lloguer(vehicle, dies)
+ quantitat(): double
}

class Client {
- nif: String
- nom: String
- telèfon: String
- lloguers: ArrayList<Lloguer>
+ Client(nif, nom, telefon)
+informe(): String
}

Vehicle <-left- Lloguer
Lloguer <-left- Client: *

Vehicle <-left- Lloguer
Lloguer <-left- Client

El diagrama de seqüència: també s'ha actualitzat per reflectir la crida al mètode quantitat per part del client, ja que això és important perquè ara el client pot saber el preu que ha de pagar per cada lloguer realitzat.

actor Usuari
Usuari -> unClient: informe()
unClient -> unLloguer: getVehicle()
note left: per cada lloguer
unClient -> unVehicle: getCategoria()
unClient -> unLloguer: getDies()
unClient -> unLloguer: quantitat()
En general, aquesta extracció de funcionalitat ha millorat la cohesió de la classe Client i ha contribuït a reduir l'acoblament entre les diferents classes del sistema.

Exercici 52_22. Actualització de la documentació

Enunciat

Afegeix una nova secció a la documentació on descriuràs els canvis realitzats al codi amb el moviment de mètodes de l'exercici anterior.

Aquest canvi segurament ha modificat el diagrama de classes i/o de seqüència. Redibuixa els diagrames que hagin canviat perquè reflexin la nova situació.

Aprofita la secció de conclusions per descriure les teves reflexions respecte aquest canvi.

Amb l'extracció de la funcionalitat de càlcul de les bonificacions de lloguer del mètode "informe()" per part del client en el nou mètode "bonificacionsDeLloguer" i canviant-li el nom a "bonificacions" a la classe Lloguer, s'ha millorat la cohesió de la classe Lloguer, ja que ara té la responsabilitat exclusiva d'obtenir informació relacionada amb els seus propis lloguers. Això també ajuda a mantenir un acoblament més baix entre les diferents classes del sistema.

Per tant, s'ha modificat la classe Lloguer per afegir el nou mètode "bonificacions" i s'ha actualitzat el mètode "informe" perquè proporcioni informació detallada sobre els lloguers realitzats per un client determinat. A més a més, s'ha actualitzat el diagrama de classes per reflectir aquestes noves funcionalitats, on la classe Lloguer té una relació bidireccional amb la classe Client.

Diagrama de Classes actualitzat:

class Vehicle {
+ {static} BASIC = 1: int
+ {static} GENERAL = 2: int
+ {static} LUXE = 3: int
- model: String
- marca: String
- categoria: int
+ Vehicle(marca, model, categoria)
}

class Lloguer {
- dies: int
+ Lloguer(vehicle, dies)
+ quantitat(): double
+ bonificacions(): int
}

class Client {
- nif: String
- nom: String
- telèfon: String

- lloguers: ArrayList<Lloguer>
+ Client(nif, nom, telefon)
+informe(): String
}

Vehicle <-left- Lloguer
Lloguer <-left- Client: *

Vehicle <-left- Lloguer
Lloguer <-left- Client

El diagrama de seqüència:

actor Usuari
Usuari -> unClient: informe()
unClient -> unLloguer: getVehicle()
note left: per cada lloguer
unClient -> unVehicle: getCategoria()
unClient -> unLloguer: getDies()
unClient -> unLloguer: quantitat()
unClient -> unLloguer: bonificacions()

Exercici 52_25. Canvis en els diagrames UML

Enunciat

Afegeix una nova secció a la documentació on descriuràs els canvis realitzats al codi amb l'eliminació de les variables total i bonificacions de Client.informe().

Redibuixa els diagrames de classe i de seqüència del projecte tal i com està després d'aquests canvis.

Pots aprofitar la secció de conclusions per descriure les teves reflexions respecte aquest canvi.

Amb la eliminació de la variable "total" i la creació del nou mètode "importTotal()" a la classe Client, s'ha millorat la cohesió d'aquesta classe, ja que ara és responsable d'obtenir la informació del preu total dels lloguers realitzats pel client. Això ajuda a mantenir un acoblament més baix entre les diferents classes del sistema.

De manera similar, amb l'eliminació de la variable "bonificacions" i la creació del nou mètode "bonificacionsTotal()" a la classe Client, s'ha millorat la cohesió d'aquesta classe, ja que ara és responsable d'obtenir la informació total de les bonificacions dels lloguers realitzats pel client. Això també ajuda a mantenir un acoblament més baix entre les classes.

Per tant, s'ha modificat la classe Client per afegir els nous mètodes "importTotal()" i "bonificacionsTotal()", i s'ha actualitzat el diagrama de classes per reflectir aquesta nova funcionalitat.

Diagrama de Classes actualitzat:

class Vehicle {
+ {static} BASIC = 1: int
+ {static} GENERAL = 2: int
+ {static} LUXE = 3: int
- model: String
- marca: String
- categoria: int
+ Vehicle(marca, model, categoria)
}

class Lloguer {
- dies: int
+ Lloguer(vehicle, dies)
+ quantitat(): double
+ bonificacions(): int
}

class Client {
- nif: String

- nom: String
- telèfon: String
- lloguers: ArrayList<Lloguer>
+ Client(nif, nom, telefon)
+ informe(): String
- importTotal(): double
- bonificacionsTotal(): int
}

Vehicle <-left- Lloguer
Lloguer <-left- Client: *

Vehicle <-left- Lloguer
Lloguer <-left- Client

El diagrama de seqüència:

actor Usuari
Usuari -> unClient: informe()
unClient -> unLloguer: getVehicle()
note left: per cada lloguer
unClient -> unVehicle: getCategoria()
unClient -> unLloguer: getDies()
unClient -> unLloguer: quantitat()
unClient -> unLloguer: bonificacions()
unClient -> unClient : importTotal()
unClient -> unClient : bonificacionsTotal()

Exercici 52_27. Canvis en els diagrames UML

Enunciat

Com a darrer exercici, actualitza els diagrames UML de la documentació amb la situació final del codi.

Troba una manera clara de documentar l'evolució dels darrers canvis. Si no se t'acut una de millor, et proposo que segueixis el patró habitual: una secció per canvi.

S'ha creat un mètode privat per a cada part de l'informe: composaCapsalera(), composaDetall() i composaPeu(). Aquesta estructura permet separar clarament les funcionalitats i millora la llegibilitat i mantenibilitat del codi. Cada mètode pot contenir la lògica específica per construir la seva part corresponent de l'informe. Al final, els resultats són concatenats en l'ordre adequat per generar l'informe complet.

Diagrama de Classes actualitzat:

class Vehicle {
+ {static} BASIC = 1: int
+ {static} GENERAL = 2: int
+ {static} LUXE = 3: int
- model: String
- marca: String
- categoria: int
+ Vehicle(marca, model, categoria)
}

class Lloguer {
- dies: int
+ Lloguer(vehicle, dies)
+ quantitat(): double
+ bonificacions(): int
}

class Client {
- nif: String
- nom: String
- telèfon: String
- lloguers: ArrayList<Lloguer>
+ Client(nif, nom, telefon)
+ informe(): String
- importTotal(): double
- bonificacionsTotal(): int
- composaCapsalera(): String
- composaDetall(): String

- composaPeu(): String
}

Vehicle <-left- Lloguer
Lloguer <-left- Client: *

Vehicle <-left- Lloguer
Lloguer <-left- Client

El diagrama de seqüència:

actor Usuari
Usuari -> unClient: informe()
unClient -> unLloguer: getVehicle()
note left: per cada lloguer
unClient -> unVehicle: getCategoria()
unClient -> unLloguer: getDies()
unClient -> unLloguer: quantitat()
unClient -> unLloguer: bonificacions()
unClient -> unClient : importTotal()
unClient -> unClient : bonificacionsTotal()
unClient -> unClient : composaCapsalera()
unClient -> unClient : composaDetall()
unClient -> unClient : composaPeu()

Exercici 52_32. Documentació final

Enunciat

Afegeix una nova secció a la documentació on descriuràs els canvis realitzats al codi amb l'extracció dels mètodes que composen l'informe.

Redibuixa els diagrames de classe i de seqüència del projecte tal i com està després d'aquests canvis.

Pots aprofitar la secció de conclusions per descriure les teves reflexions respecte aquest canvi.

S'ha creat una constant estàtica final amb el nom de EUROS_PER_UNITAT_DE_COST que substitueix el valor constant 30 per a la variable de cost per unitat a la classe Client.

S'han creat variables per a cada categoria (BASIC, GENERAL, LUXE) amb els noms de QUANTITAT_BASIC, QUANTITAT_GENERAL, DIES_BASIC , DIES_GENERAL, MULTIPLE_BASIC , MULTIPLE_GENERAL , MULTIPLE_LUXE a la classe lloguer.

En aquest cas, s'ha canviat el nom del mètode Lloguer.quantitat() a sumaQuantitat() i la variable quantitat s'ha canviat per la variable quantitatTotal a la classe Lloguer.

S'ha creat un mètode privat per a cada part de l'informe en HTML: composaCapsaleraHTML(), composaDetallHTML() i composaPeuHTML() a la classe Client.

Diagrama de Classes actualitzat:

class Vehicle {
+ {static} BASIC = 1: int
+ {static} GENERAL = 2: int
+ {static} LUXE = 3: int
- model: String
- marca: String
- categoria: int
+ Vehicle(marca, model, categoria)
}

class Lloguer {
- dies: int
+ Lloguer(vehicle, dies)
+ sumaQuantitats(): double
+ bonificacions(): int
- {static} QUANTITAT_BASIC = 3: int
- {static} DIES_BASIC = 3: int
- {static} DIES_GENERAL = 2: int
- {static} MULTIPLE_BASIC = 1.5: double

- {static} QUANTITAT_GENERAL = 4: int
- {static} MULTIPLE_GENERAL = 2.5: double
- {static} MULTIPLE_LUXE = 6: int

}

class Client {
- nif: String
- nom: String
- telèfon: String
- lloguers: ArrayList<Lloguer>
+ Client(nif, nom, telefon)
+ informe(): String
- importTotal(): double
- bonificacionsTotal(): int
- composaCapsalera(): String
- composaDetall(): String
- composaPeu(): String
+ informeHTML(): String
- composaCapsaleraHTML(): String
- composaDetallHTML(): String
- composaPeuHTML(): String
+ {static} EUROS_PER_UNITAT_DE_COST = 30: double

}

Vehicle <-left- Lloguer
Lloguer <-left- Client: *

Vehicle <-left- Lloguer
Lloguer <-left- Client

El diagrama de seqüència:

actor Usuari
Usuari -> unClient: informe()
Usuari -> unClient: informeHTML()
unClient -> unLloguer: getVehicle()
note left: per cada lloguer
unClient -> unVehicle: getCategoria()
unClient -> unLloguer: getDies()
unClient -> unLloguer: sumaQuantitats()
unClient -> unLloguer: bonificacions()
unClient -> unClient : importTotal()
unClient -> unClient : bonificacionsTotal()

```
unClient -> unClient : composaCapsalera()
unClient -> unClient : composaDetall()
unClient -> unClient : composaPeu()

unClient -> unClient : composaCapsaleraHTML()
unClient -> unClient : composaDetallHTML()
unClient -> unClient : composaPeuHTML()
```