IES-2onprob.Esq.Comportament-QT2...

Useernaamee29

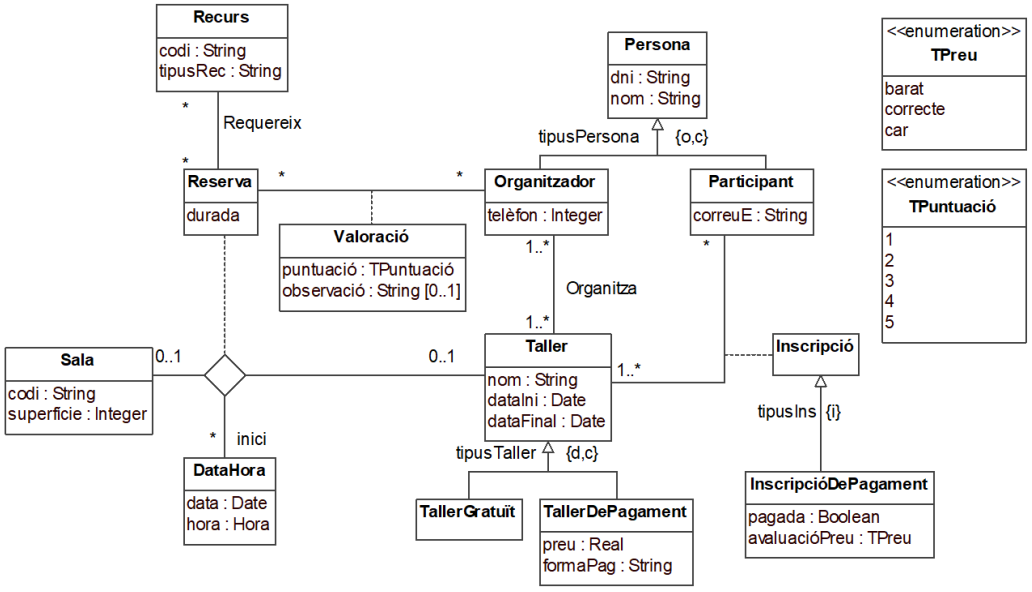
Introducción a la ingeniería del software

2º Grado en Ingeniería Informática

Facultad de Informática de Barcelona (Fib)
Universidad Politécnica de Catalunya

7a entrega d'IES – 11 de novembre del 2022 – QT2223

Considereu un sistema per a la gestió dels tallers que s'ofereixen a un centre cívic. Als tallers, que poden ser gratuïts o de pagament, s'hi poden inscriure persones participants i el sistema guarda totes les inscripcions, amb informació addicional en el cas de les inscripcions a tallers de pagament. A més, el centre cívic disposa d'un conjunt de sales que es poden reservar per a la realització dels tallers.



Restriccions Textuals

1. Claus externes: (Taller, nom), (Persona, dni), (Sala, codi), (DataHora, data+hora), (Recurs, codi)
2. La data d'inici d'un taller és anterior o igual a la seva data de finalització
3. Un organitzador d'un taller no pot ser participant d'aquell mateix taller
4. Una inscripció és de pagament si i només si correspon a un taller de pagament
5. Les reserves d'una mateixa sala no es poden solapar temporalment
6. Les reserves d'un mateix taller no es poden solapar temporalment
7. Una reserva ha de tenir una data que estigui entre la data d'inici i de finalització del taller de la reserva
8. Un recurs no pot ser requerit per dues o més reserves que se solapin temporalment
9. Un organitzador no pot valorar una reserva que correspon a un taller que ell no organitza

El sistema a desenvolupar no pot modificar les dades de Persona, Organitzador, Participant, Taller, DataHora, Sala i Valoració, ja que les gestiona un altre sistema. El sistema ha de permetre efectuar les funcionalitats següents:

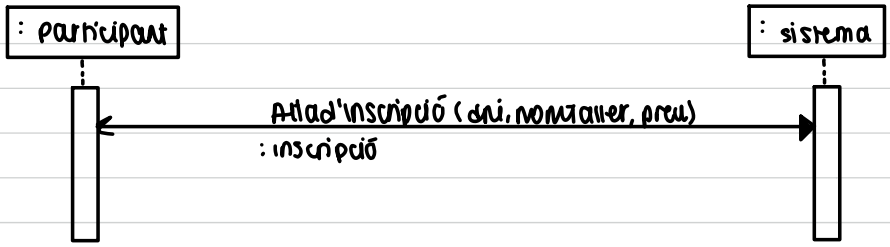
Alta d'Inscripció: Quan un participant vol inscriure's a un taller ell mateix introdueix el seu dni i el nom del taller al que vol inscriure's. Si el taller és de pagament, per defecte es considerarà inscripció no pagada i el participant haurà d'indicar l'avaluació que fa del seu preu. Considereu que una inscripció només es pot fer abans del començament del taller. Feu que la interacció necessària per dur a terme aquesta funcionalitat requereixi només un esdeveniment.

Alta de Reserva: Quan un organitzador d'un taller vol fer una reserva de sala, introdueix al sistema les dades necessàries per fer-ho. Per cada recurs que es requereixi a la reserva, haurà d'informar-lo. Si el recurs és nou s'haurà de donar d'alta. Feu que la interacció necessària per dur a terme aquesta funcionalitat requereixi més d'un esdeveniment.

Consulta Reserves Exitoses: Quan un administrador vol consultar les reserves més exitoses d'un taller de pagament indica al sistema el nom del taller. El sistema retorna, per a cada reserva amb més de 5 valoracions amb puntuació de 5 d'aquest taller: el codi de la sala reservada, la data i hora de la reserva i el conjunt de correus de participants inscrits. Aquesta funcionalitat solament pot demanar-se si hi ha com a mínim 3 inscripcions al taller pagades i amb avaluació de barat.

Es demana que feu mitjançant la notació UML els diagrames de seqüència i contractes en OCL de les operacions corresponents a les funcionalitats anteriors.

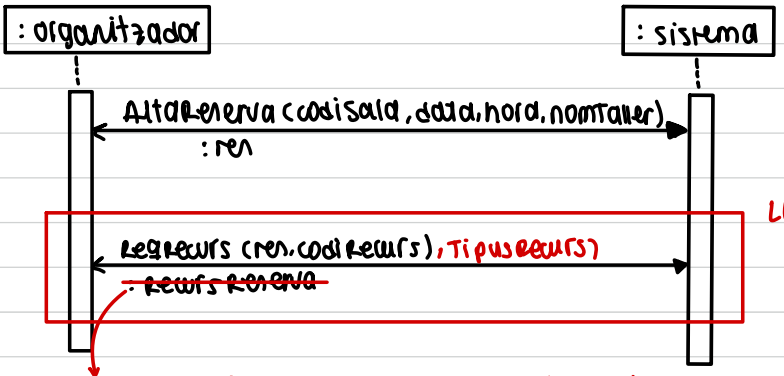
Alta d'inscripció



and t.dataInici ≥ today())

Context sistema:: Alta d'inscripció (dni: string, nomTaller: string, price: Tpreu): Inscripcio
pre: Participant.allInstances() → (p | dni = dni and Taller.allInstances() → (t | t.nom = nomTaller)
post: Inscripcio.allInstances() → Exists (i | i.oclIsNew() and i.participant.dni = dni and
i.Taller.nom = nomTaller and if i.taller.oclIsTypeOf(TallerDePag)
~~if i.tipusIns = = InscripcioDePagament then i.avaluacioPreu = price~~
~~and i.pagada = false endif and result = i~~)
then i.oclIsTypeOf(InscripcioDePagament) and i.oclAsType(InscripcioDePagament).
pagada = false and i.oclAsType(InscripcioDePag).avaluacioPreu = avuiPreu
endif)

Alta de Reserva



LOOP - una reserva pot tenir molts recurs

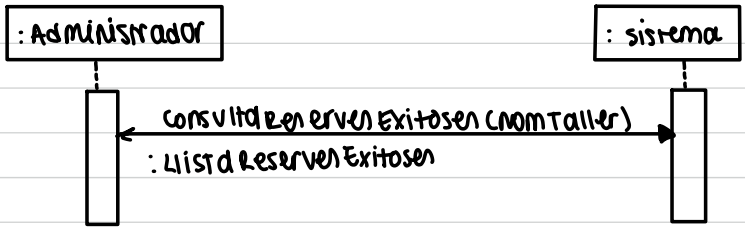
• Es retorna si s'ha de utilitzar posteriorment

context sistema:: AltaReserva (codiSala: string, data: Date, hora: Time, nomTaller: string): res
pre: sala.allInstances() → exists (s | s.codi = codiSala)
~~Data+Hora.allInstances() → exists (dh | dh.data = data and dh.hora = hora)~~ → no fa falta serà un int
Taller.allInstances() → exists (t | t.nom = nomTaller)
post: Reserva.allInstances() → exists (r | r.oclIsNew() and r.sala.codi = codiSala and
vincles { r.Taller.nom = nomTaller and r.DataHora.data = data and result = r)
and r.DataHora.hora = Hora and r.duracio = duracio → atribut

oclIsNew i no un boolea!!

context sistema:: RegRecurs (res: Reserva, codiRecurs: string): RecursReserva → tipus recurs: string
pre: ~~Reserva.allInstances() → exists (r | r = res)~~ → no fa falta, ja la reserva ja està declarada
~~Recurs.allInstances() → exists (rec | rec.codi = codiRecurs)~~ → si no existeix en crea de nou
post: if not Recurs.allInstances()@PRE → exists (r | r.codi = codiR) then
RecursallInstances() → exists (r | r.codi = codiR and r.tipusRec = tipusRec and r.oclIsNew())
endif and r.Recurs.codi → includes (codiR)

consulta reserves exitoses



context sistema:: consultaReservesExitoses(nomTaller: string):
set(tupletype(codisala: string, data: Date, hora: time, correus: string)) ~~llista reserves exitoses~~ → correus: set(string)

pre: Taller.allInstances() → exists(t | t.nom = nomTaller) and ~~Inscripcio.allInstances()~~ t.inscripcio
→ select(i | if i.tipusIns = "InscripcioDePagament" then i.oclAsType(InscripcioDe
pagament).pagada = true and i.oclAsType(InscripcioDepagament).avaluacioPreu
= TPreu::barat endif) → size() > 5

body: let resultat: set(reserva) = Reserva.allInstances() → select(r | r.taller.nom =
= nomTaller and r.valoracio → select(v | v.puntuacio = TpuntuacioPossible::5)) → size() > 5
in result = resultat → collect(r | tuple{
codiSala = r.sala.codi,
data = r.dataHora.data,
hora = r.dataHora.hora,
correus = r.taller.participant.correu
})

pre: Taller.allInstances() → exists(t | t.nom = nomTaller and t.inscripcio → select(i |
i.oclIsTypeOf(InscripcioDePagament) and i.oclAsType(InscripcioDepagament).pagada
= true and i.oclAsType(InscripcioDepagament).avaluacioPreu = TPreu::barat)
→ size() ≥ 3)

B) body: let reserves: set(reserva) = Reserva.allInstances() → select(r | r.valoracio →
select(v | v.puntuacio = T.puntuacio::5) → size() ≥ 5)
in result = reserves → collect(r | tuple{codi = r.sala.codi,
data = r.datahora.data,
hora = r.datahora.hora,
correus = r.taller.participant.correu → asSet()})

declaració variables (reserves)

construir tuples

elimina els repetits

asSet() que no sigui clau externa i vol no repetits