//las operaciones que tienen … al final, son más grandes pero no pintó describirlas.
//faltan bastantes casos, pero los mínimos están todos.

Caso de Uso: Alta Perfil

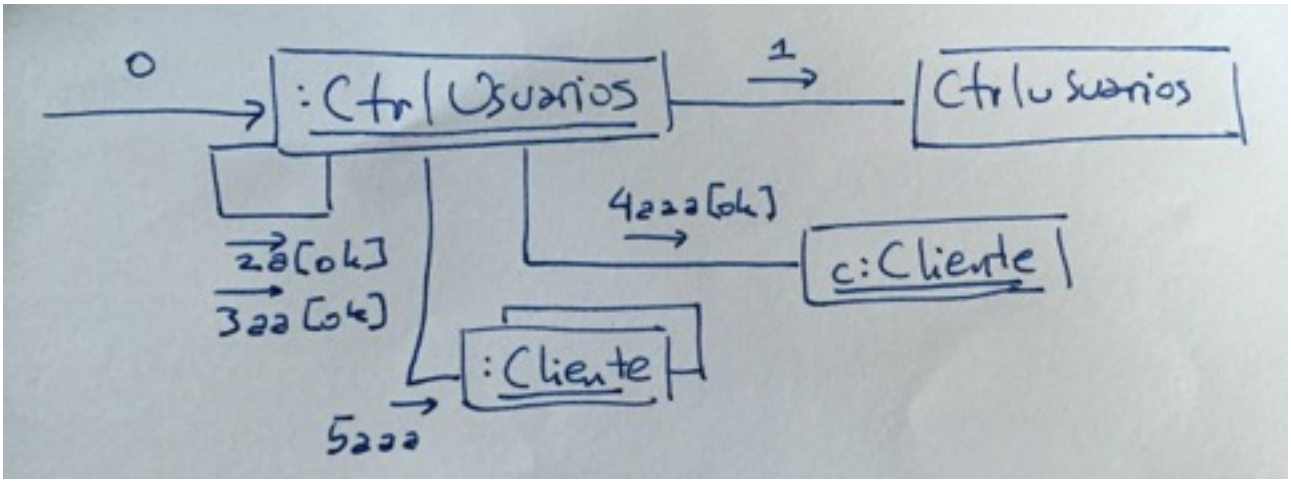
El caso de uso comienza cuando el administrador desea dar de alta a un nuevo usuario en el sistema. Para ello indica su nickname, nombre, apellido, correo electrónico, fecha de nacimiento e imagen (opcional). Además indica si es Cliente o Artista. Si es Artista, puede incluir una breve biografía y un link a su sitio web de promoción. Finalmente el sistema da de alta al usuario. Si el nickname o el correo electrónico ya está reclamado por algún otro usuario, o si algún campo está vacío o tiene un formato incorrecto el sistema avisa al administrador. Si el administrador lo desea puede corregir los campos o cancelar el alta.

Análisis:DSS

[alt]
cliente
AltaCliente
[loop ingresó cosas mal]
[opt] quiere
AltaCliente
artista
AltaArtista
[loop ingresó cosas mal]
[opt] quiere
AltaArtista

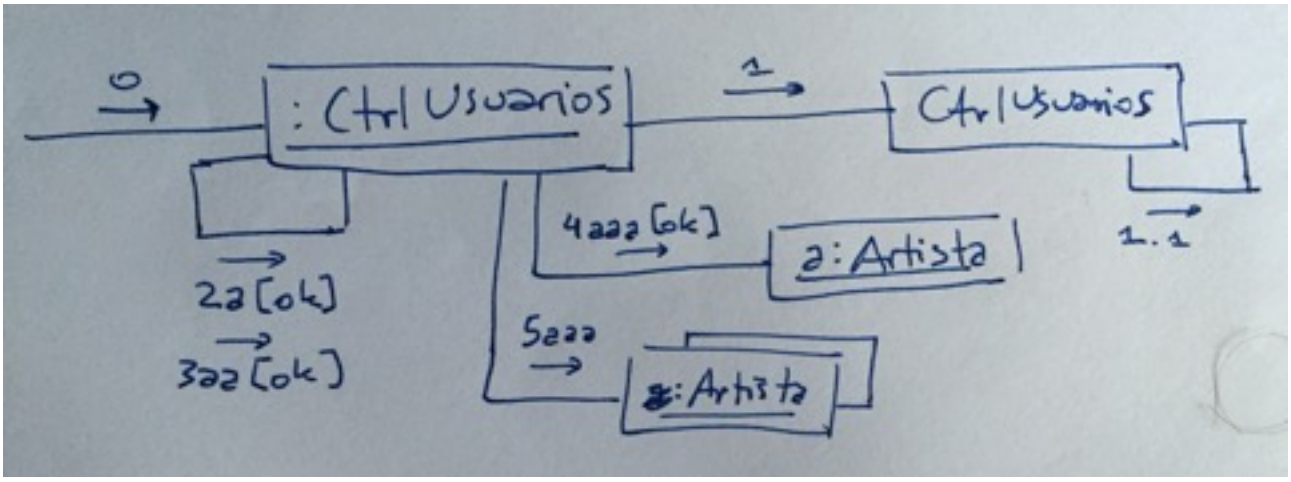
Diseño: DCOMM

Operación: AltaCliente



0:) AltaCliente(d:DataCliente)
1:) ok:=ValidarDatosUsuario(d) //movido a clase usuario
2a[ok]:) ok:=!ExisteUsuarioCorreo(d.correo)…
3aa[ok]:) ok:=!ExisteUsuarioNick(d.nick)…
4aaa[ok]:) create(d)
5aaa:) add(c)

Datatypes Relacionados: DataUsuario, DataCliente.



Operación: AltaArtista

0:) AltaArtista(d:DataArtista)
1:) ok:=ValidarDatosArtista(d) //movido a clase artista
1.1:) ok:=ValidarDatosUsuario(d) //movido a clase usuario
2a[ok]:) ok:=!ExisteUsuarioCorreo(d.correo)…
3aa[ok]:) ok:=!ExisteUsuarioNick(d.nick)…
4aaa[ok]:) create(a)
5aaa:) add(a)

Datatypes Relacionados: DataUsuario, DataArtista.

Caso de Uso: Alta Album

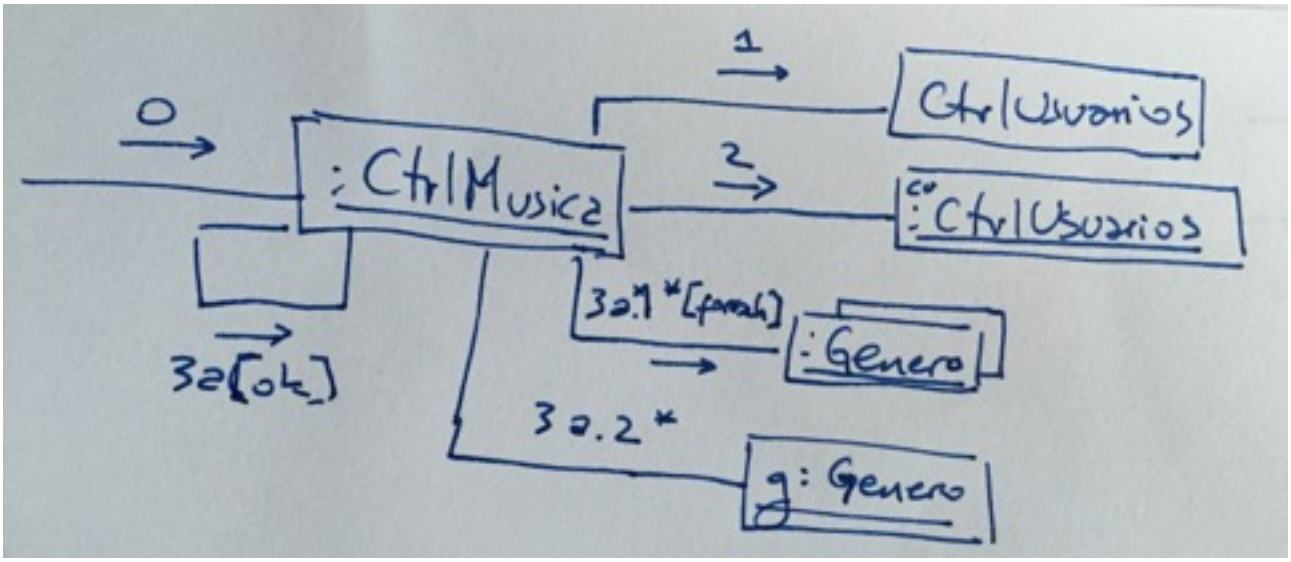
El caso de uso comienza cuando el administrador desea crear un nuevo álbum de un artista. Para ello primero indica el artista y, en caso de existir, indica el nombre del álbum, el año de creación, los géneros a los que aplica y una imagen (opcional). Además, indica la información de cada una de los temas que componen el álbum: nombre, duración y su ubicación en el álbum. Finalmente, el administrador determina el archivo de música o la dirección web desde dónde se puede escuchar el tema. Si los datos son correctos el sistema da de alta el álbum.

Análisis:DSS

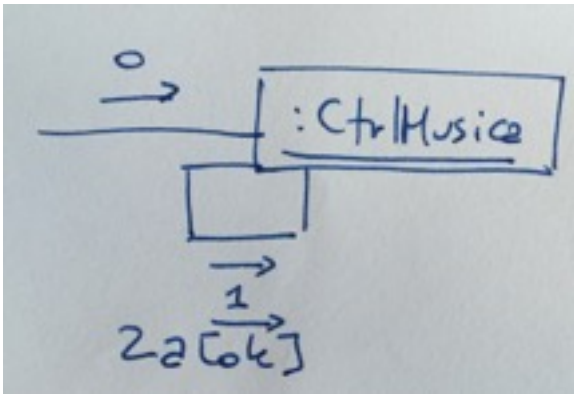
ExisteArtista() : [bool, List<Genero>]
[alt] existe=true
IngresaTema //recuerda tema
[loop quiere más temas]
IngresaTema //recuerda tema
ListarGeneros // para que no se los tenga que saber de memoria.
AltaAlbum
//no se pide pero debería haber opción de cancelar.

Diseño: DCOMM

Operación: ExisteArtista



0:) ExisteArtista(nomArtista:string) : [ok: bool, lst: List<Genero>]
1:) cu:=getInstancia()
2:) ok:=BuscarUsuarioNick(nomArtista).EsArtista()…
3a[ok]:) lst:=ListarGeneros()
3a.1*[foreach]:) g:=next()
3a.2*:) lst.add(g.getNombre())

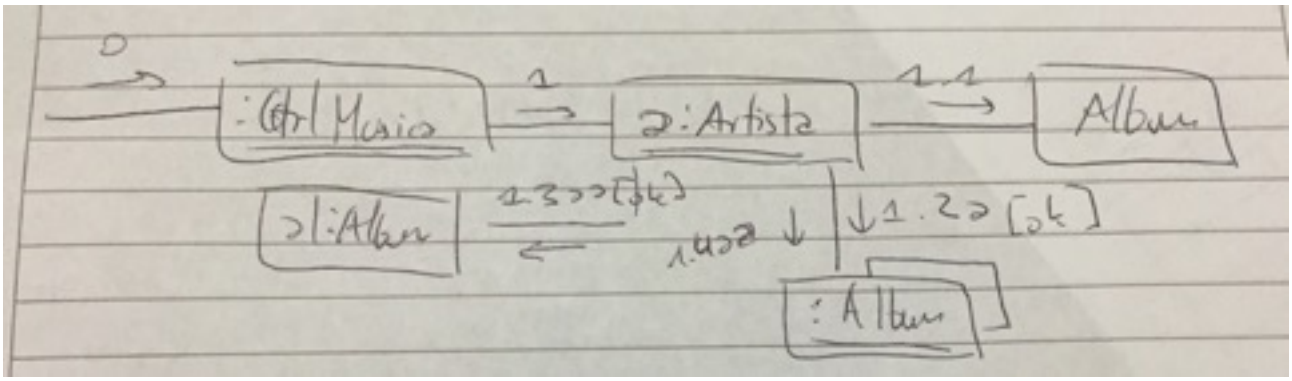


Operación: IngresaTema

0:) IngresaTema(d:DataTema)
1:) ok:=ValidarTema(d) //si ok, recuerda el tema

Datatypes Relacionados: DataTema, DataTemaWeb, DataTemaArchivo

Operación: AltaAlbum



//pre: se recuerdan List<Tema> lst y Artista a
0:) AltaAlbum(d:DataAlbum)
1:) AltaAlbum(d,lst)
1.1:) ok:=ValidarAlbum(d,lst)
1.2a[ok] :) ok:=exists(d.nombre)
1.3aa[ok] :) create(d,lst)
1.4aa :) add(al)

Datatypes Relacionados: DataAlbum

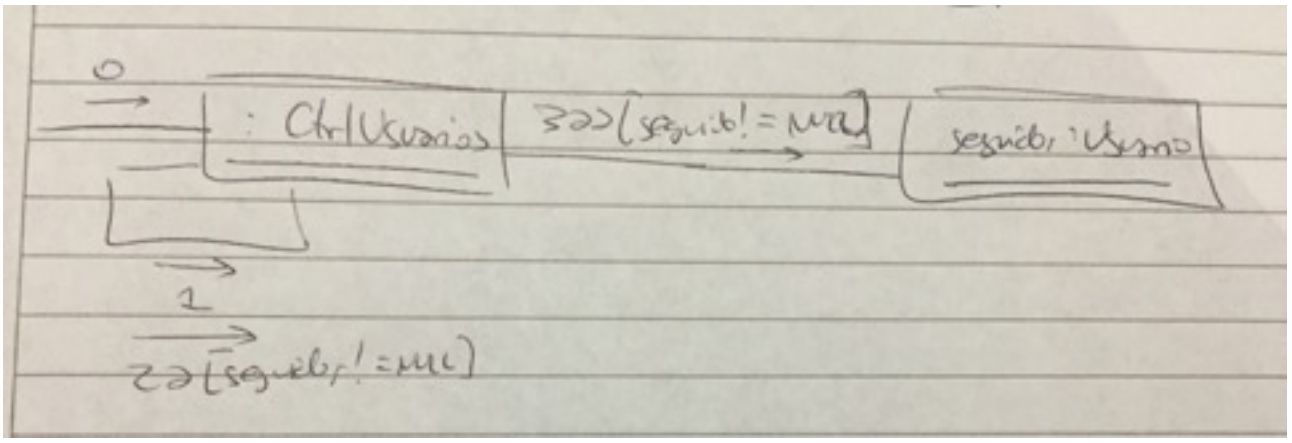
Caso de Uso: Seguir Usuario

El caso de uso comienza cuando el administrador desea indicar que un cliente comienza a seguir a un usuario. Para ello indica el cliente que desea realizar el seguimiento y el usuario (cliente/artista) al que desea seguir, y el sistema relaciona ambos usuarios.

Análisis: DSS
AltaSeguir

Diseño: DCOMM
Operación: AltaSeguir

0:) AltaSeguir(nomSeguidor, nomSeguido : string)
1:) seguidor:=BuscarUsuarioNick(nomSeguidor)



2a[seguidor!=NULL]:) seguido:= BuscarUsuarioNick(nomSeguido)
3aa[seguido!=NULL]:) Seguir(seguido)
…

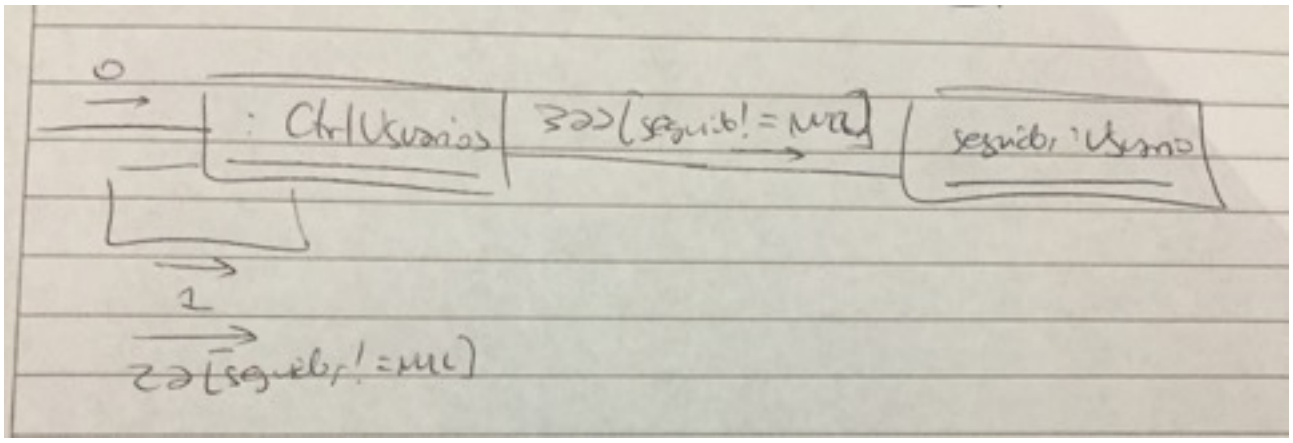
//si seguido es un artista le tiene que registrar en el artista.

Caso de Uso: Dejar de Seguir Usuario

El caso de uso comienza cuando el administrador desea indicar que un cliente deja de seguir a un usuario. Para ello indica el cliente que desea finalizar el seguimiento y el usuario (cliente/artista) al que no desea seguir más, y el sistema termina con la relación.

Análisis: DSS
BajaSeguir

Diseño: DCOMM
Operación: BajaSeguir



0:) BajaSeguir(nomSeguidor, nomSeguido : string)
1:) seguidor:=BuscarUsuarioNick(nomSeguidor)
2a[seguidor!=NULL]:) seguido:= BuscarUsuarioNick(nomSeguido)
3aa[seguido!=NULL]:) seguidor.DejarDeSeguir(seguido)
…

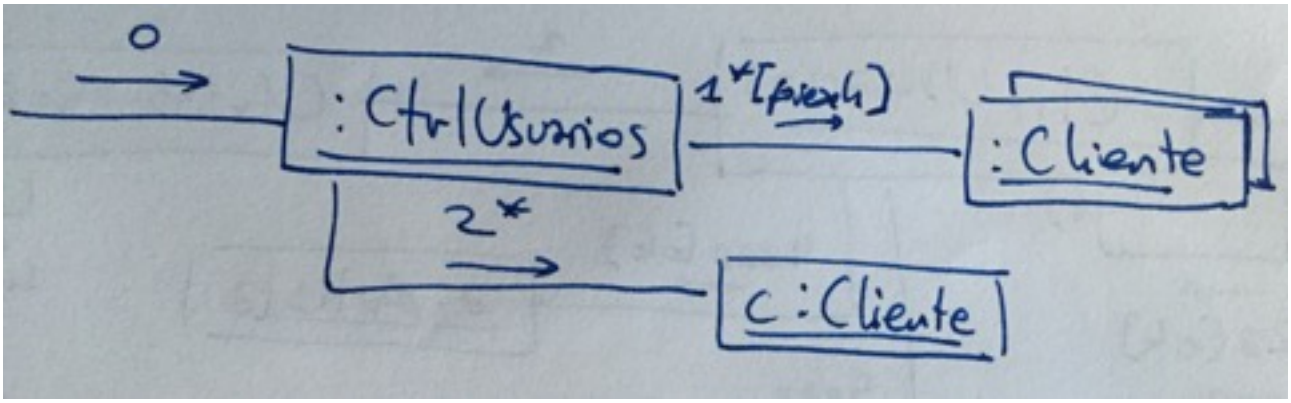
Caso de Uso: Consulta Cliente

El caso de uso comienza cuando el administrador desea consultar el perfil de un cliente. Para ello el sistema muestra la lista de todos los clientes y el administrador elige uno. Luego, el sistema muestra todos los datos básicos del cliente, incluyendo, sí tiene, su imagen asociada. Además, se muestran los nickname de todos los seguidores que tiene, de los usuarios a los que sigue (identificando si son clientes o artistas), se muestran las listas de reproducción que creó y las preferencias que tiene guardadas.

Análisis: DSS
ListarClientes
ConsultaCliente

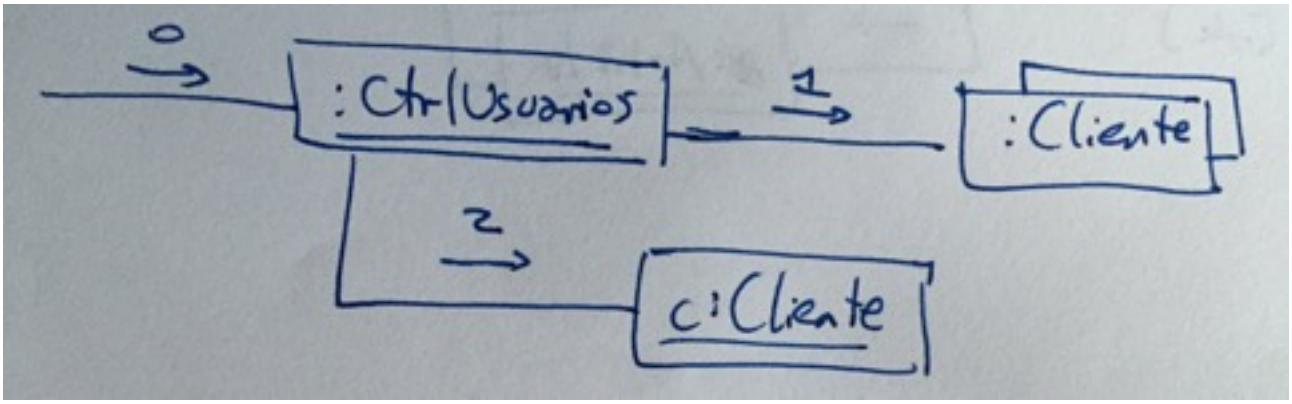
Diseño: DCOMM

Operación: ListarClientes



0:) lst:= ListarClientes() : List<String>
1*[foreach]:) c:=next()
2*:) lst.add(c.getNombre())

Operación: ConsultaCliente



0:) ConsultaCliente(nombre:string) : DataClienteExt
1:) c:=find(nombre)
2:) lst.add(c.getDataExt())… //getDataExt tiene lógica bastante pero fácil. recorrer cosas.

Datatypes relacionados: DataClienteExt

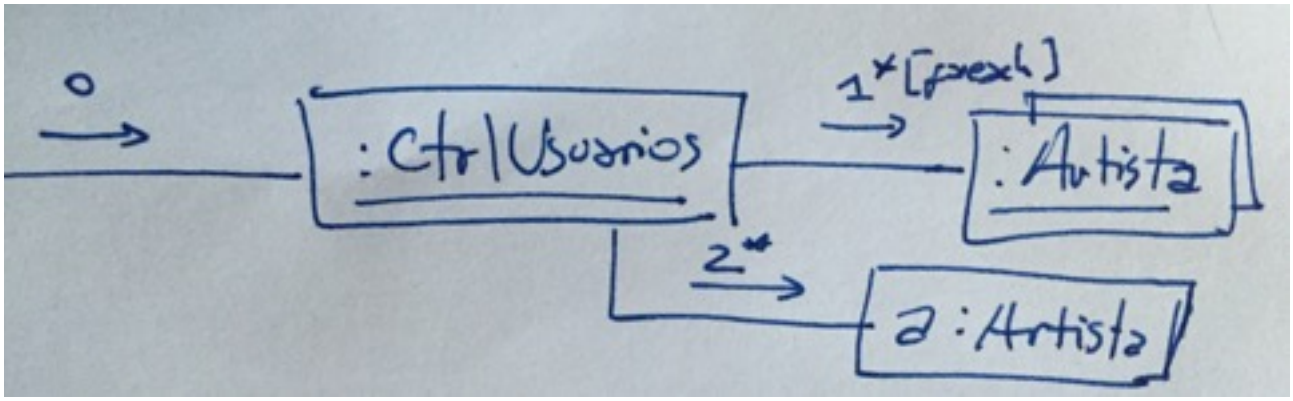
Caso de Uso: Consulta Artista

El caso de uso comienza cuando el administrador desea consultar el perfil de un artista. Para ello el sistema muestra la lista de todos los artistas y el administrador elige uno. Luego, el sistema muestra todos los datos básicos del artista, incluyendo, sí tiene, su imagen asociada, biografía y dirección de su página web. Además, se muestra el número de seguidores que tiene y los nickname de todos ellos, así como todos sus álbumes publicados.

Análisis: DSS
ListarArtistas
ConsultaArtista

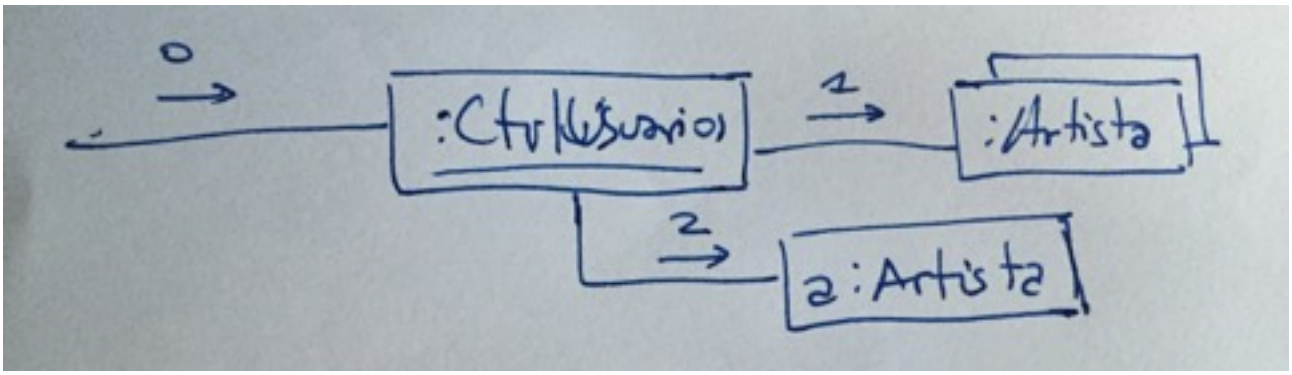
Diseño: DCOMM

Operación: ListarArtistas



0:) lst:= ListarArtistas() : List<String>
1*[foreach]:) a:=next()
2*:) lst.add(a.getNombre())

Operación: ConsultaArtista



0:) ConsultaArtsta(nombre:string) : DataArtistaExt
1:) a:=find(nombre)
2:) lst.add(c.getDataExt())… //getDataExt tiene lógica bastante pero fácil. recorrer cosas. tiene que conocer sus álbumes el artista. y sus followers.

Datatypes relacionados: DataArtistaExt

Caso de Uso: Consulta Álbum

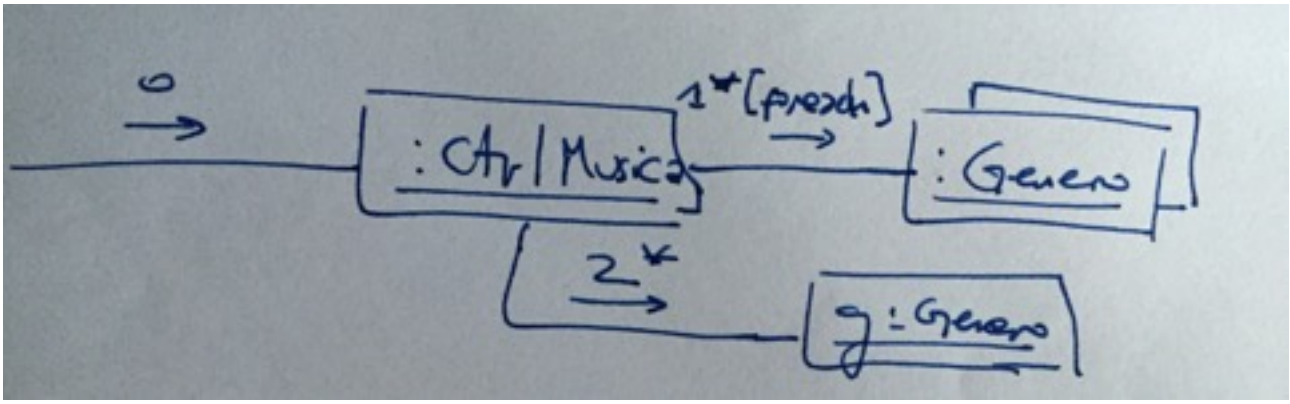
El caso de uso comienza cuando el administrador desea consultar un álbum. Para ello el administrador indica si desea consultar por género o artista y el sistema lista los géneros (o artistas) existentes y el administrador selecciona uno de los álbumes correspondientes a dicho género (o artista). Luego, el sistema muestra el nombre del álbum, año de creación, los géneros a los que aplica, su imagen y la información detallada de cada una de los temas que componen el álbum (permitiendo descargar el archivo de música o ver la dirección web ingresada).

Análisis: DSS

[alt]
quiere genero
ListarGeneros
ListarAlbumesDeGenero
quiere artista
ListarArtistas
ListarAlbumesDeArtista
ConsultaAlbum
[loop] quiere descargar temas
DescargarTema

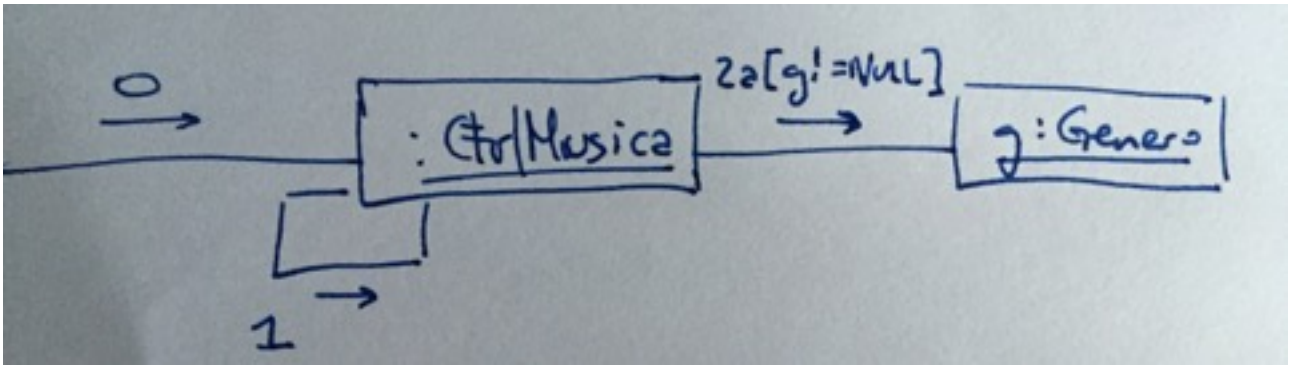
Diseño: DCOMM

Operación: ListarGeneros



0:) lst:=ListarGeneros() : List<string>
1*[foreach]:) g:=next()
2*:) lst.add(g.getNombre())

Operación: ListarAlbumesDeGenero

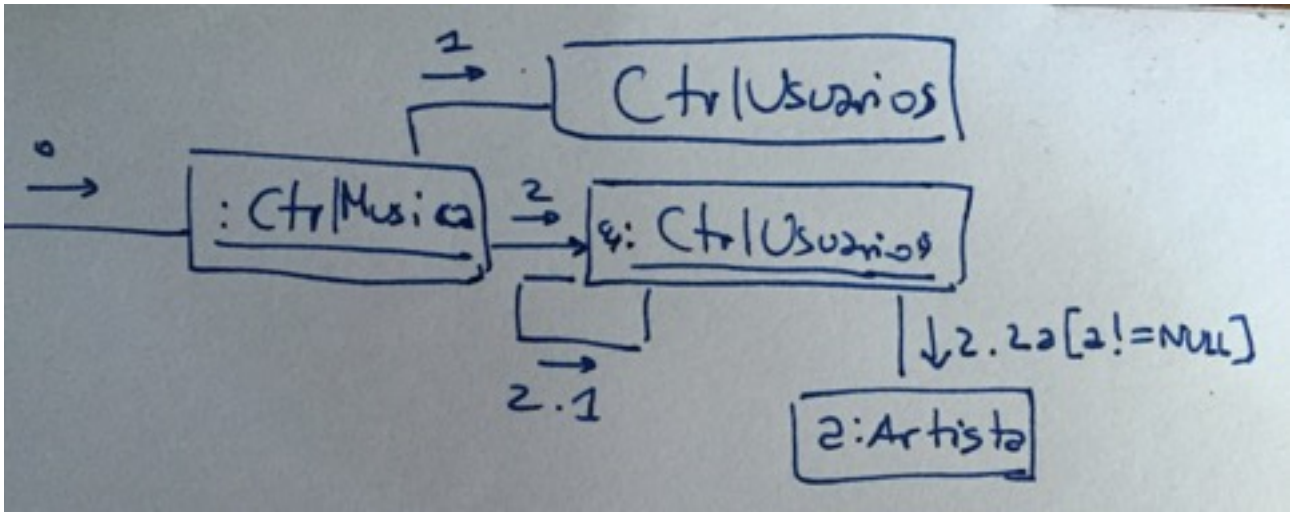


0:) ListarAlbumesDeGenero(nomGenero: string) : List<string>
1:) g:=BuscarGenero(nomGenero)…
2a[g!=NULL]:) ListarAlbumes()…

Operación: ListarArtistas

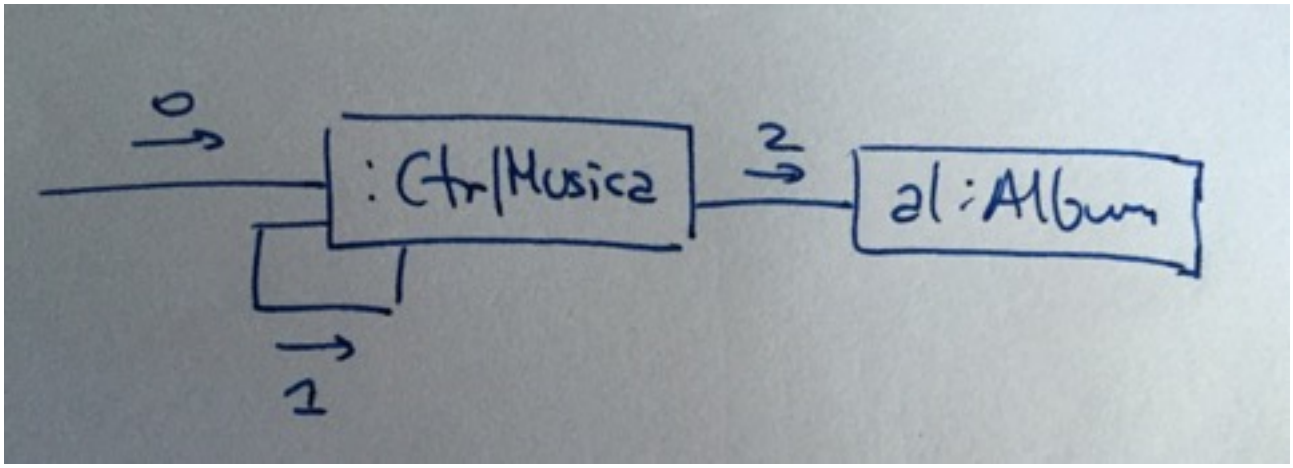
comparte de CU: Listar Artista

Operación: LIstarAlbumesDeArtista



0:) ListarAlbumesDeArtista(nomArtista:string) : List<string>
1:) cu:=getInstancia()
2:) ListarAlbumesDeArtista(nomArtista)
2.1:) a:=BuscarArtista(nomArtista)…
2.2a[a!=NULL]:) ListarAlbumes()…

Operación: Consulta Album



//se recuerda género o artista del álbum, dependiendo lo que haya elegido el Admin.
0:) ConsultaAlbum(nomAlbum:string) : DataAlbumExt
1:) al:=BuscarAlbum(nomAlbum, g)… OR BuscarAlbum(nomAlbum, a)…
2:) al.getData()

Datatypes relacionados: DataAlbumExt, DataAlbum, DataTema

Operación: DescargarTema

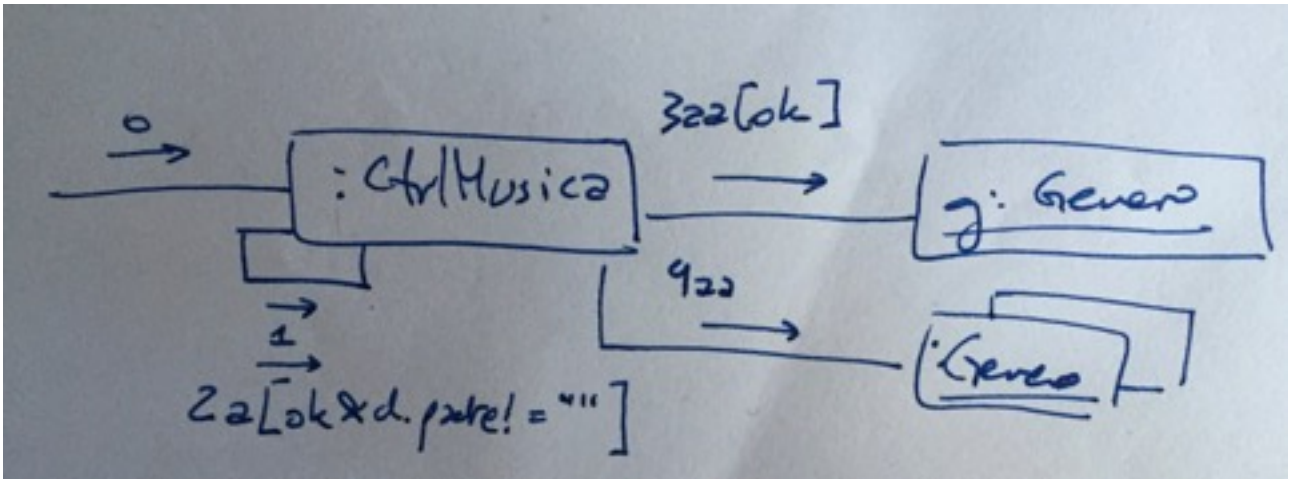
FALTA DCOMM de DescargarTema, pero no debería ser difícil. va a depender de la implementación, probablemente.

Caso de Uso: Alta Género

El caso de uso comienza cuando el administrador desea crear un nuevo género en el sistema. Para ello indica su nombre y opcionalmente su padre. Finalmente el sistema registra el género indicado. En el caso de que no se elija un padre será ubicado por debajo de un género por defecto de nombre Género, que se encuentra al tope de la jerarquía. Si los datos son correctos, el sistema da de alta el nuevo género con los datos especificados anteriormente.

Análisis: DSS
[opt]
ListarGeneros
AltaGenero

Diseño: DCOMM
Operacion: ListarGeneros
Comparte de ConsultaAlbum
Operacion: AltaGenero



0:) AltaGenero(d:DataGenero)
1:) ok:=!ExisteGenero(d.nombre)
2a[ok&d.padre!=“”]:) ok:=ExisteGenero(d.padre)
3aa[ok]:) create(d)
4aa:) add(g)

Datatypes relacionados: DataGenero

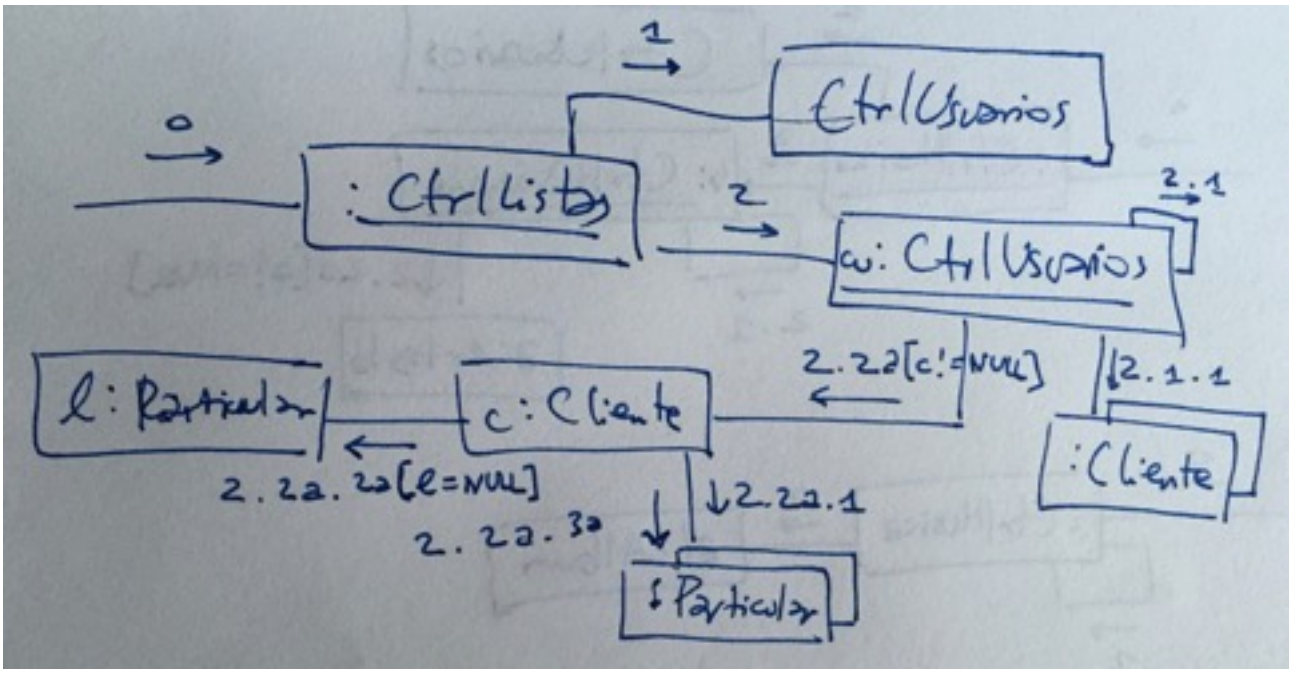
Caso de Uso: Alta de Lista

El caso de uso comienza cuando el administrador desea crear una nueva lista de reproducción. Para ello indica si la lista es por defecto o particular, el nombre de la lista y una imagen (opcional). Si la lista es por defecto, debe indicar además a qué género pertenece. En caso contrario, debe indicar el cliente propietario de la lista. Si los datos son correctos, el sistema crea la lista y en caso de ser particular, la crea privada.

Análisis: DSS
[alt]
particular
ListarClientes
AltaListaParticular
defecto
ListarGeneros
AltaListaDefecto

Operación: ListarClientes
comparte de ConsultaCliente

Operación: AltaListaParticular



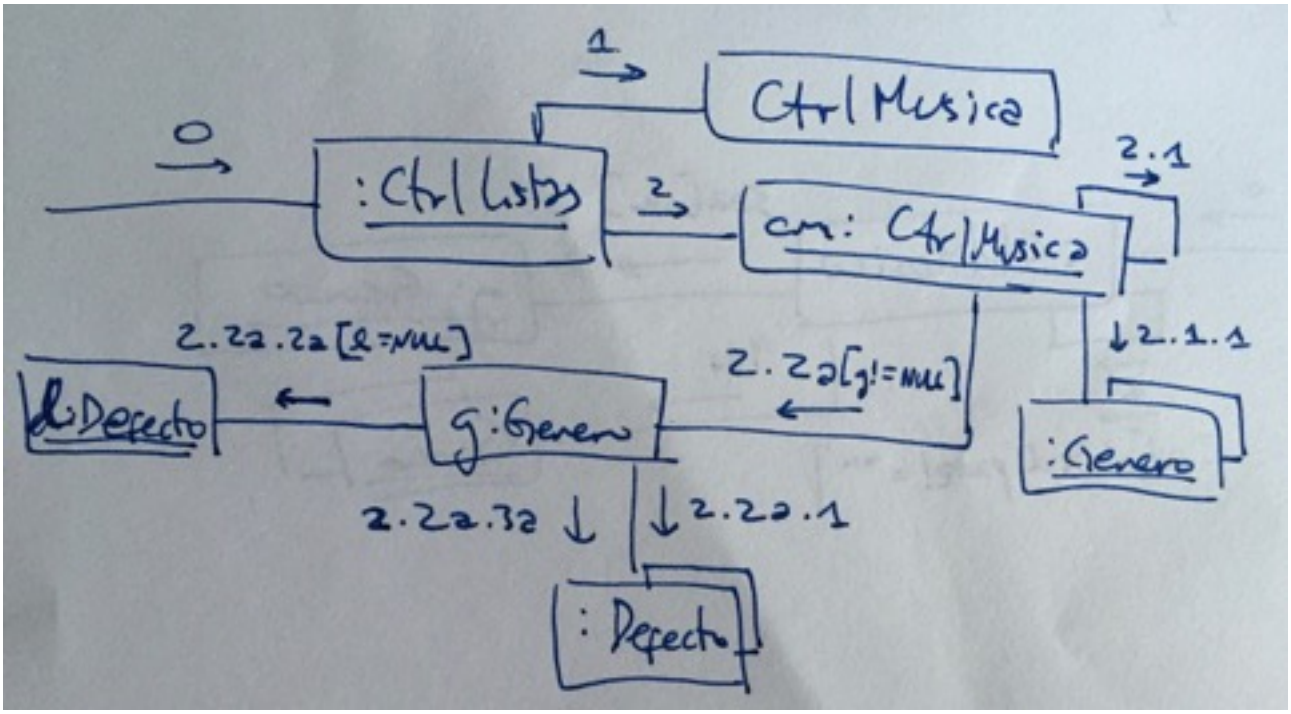
0:) AltaListaParticular(d:DataParticular)
1:) getInstancia()
2:) CrearLista(d)
2.1:) BuscarCliente(d.nomCliente)
2.1.1:) c:=find(d.nomCliente)
2.2a[c!=NULL]:) AltaLista(d)
2.2a.1:) l:=find(d.nombre)
2.2a.2a[l=NULL]:) create(d)
2.2a.3a:) add(l)

Datatypes relacionados: DataParticular, DataLista

Operación: ListarGeneros

comparte de ConsultaAlbum

Operación: AltaListaDefecto



0:) AltaListaDefecto(d:DataDefecto)
1:) getInstancia()
2:) CrearLista(d)
2.1:) BuscarGenero(d.genero)
2.1.1:) g:=find(d.genero)
2.2a[g!=NULL]:) AltaLista(d)
2.2a.1:) l:=find(d.nombre)
2.2a.2a[l=NULL]:) create(d)
2.2a.3a:) add(l)

Datatypes relacionados: DataDefecto, DataLista

Caso de Uso: Agregar Tema a Lista

El caso de uso comienza cuando el administrador desea agregar un nuevo tema a una lista de reproducción. Para ello indica el usuario y la lista a la cual se agregará el tema, o sólo el nombre de la lista en caso de ser una lista por defecto. Luego, selecciona un tema (según sea de una lista propietaria pública, por defecto, o de un álbum) y el sistema lo agrega a la lista.

[alt]
- particular
  - ListarClientes
  - ListarListasDeCliente
  - ElegirLista
- defecto
  - ListarListasDefecto
  - ElegirLista

[alt]
- la fuente es una lista pública
  - ListarClientes
  - ListarListasDeCliente
  - ListarTemasDeLista
- la fuente es una lista por defecto
  - ListarListasDefecto
  - ListarTemasDeLista
- la fuente es un álbum
  - ListarArtistas
  - ListarAlbumesDeArtista
  - ListarTemasDeAlbum

AgregarTemaLista(tema) //la lista se la acuerda.

CUListarTemas
AgregarTemaLista(tema, lista)

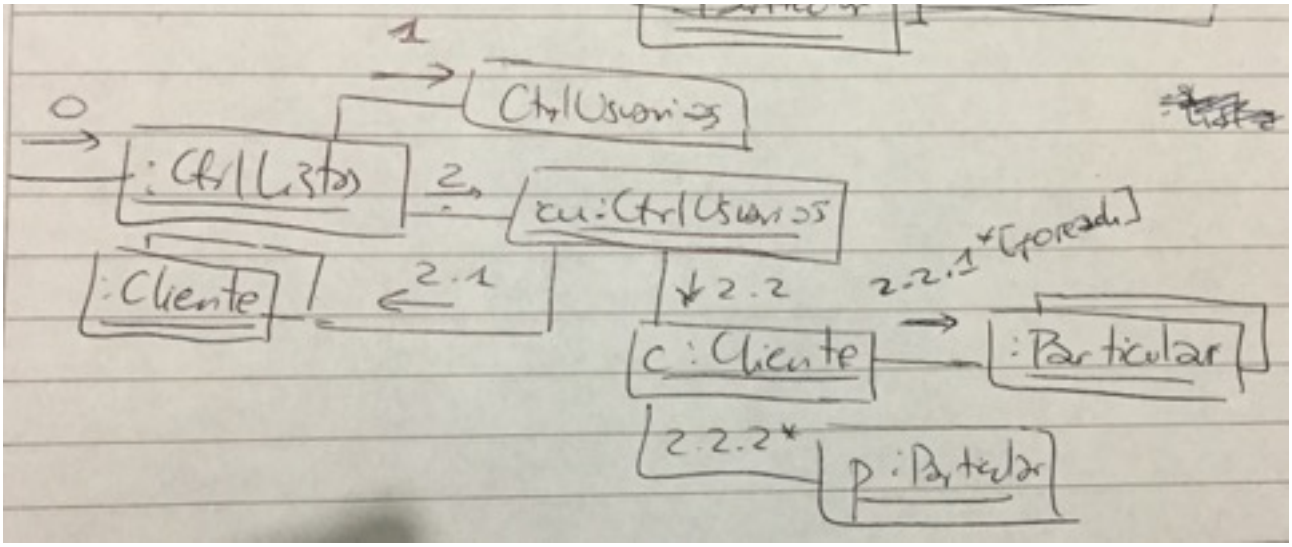
Caso de Uso: Quitar Tema de Lista

Análisis: DSS
[alt]
- particular
  - ListarClientes
  - ListarListasDeCliente //recuerda cliente
- defecto
  - ListarListasDefecto

ListarTemasLista(Lista l) //recuerda lista
RemoverTemaLista(tema)

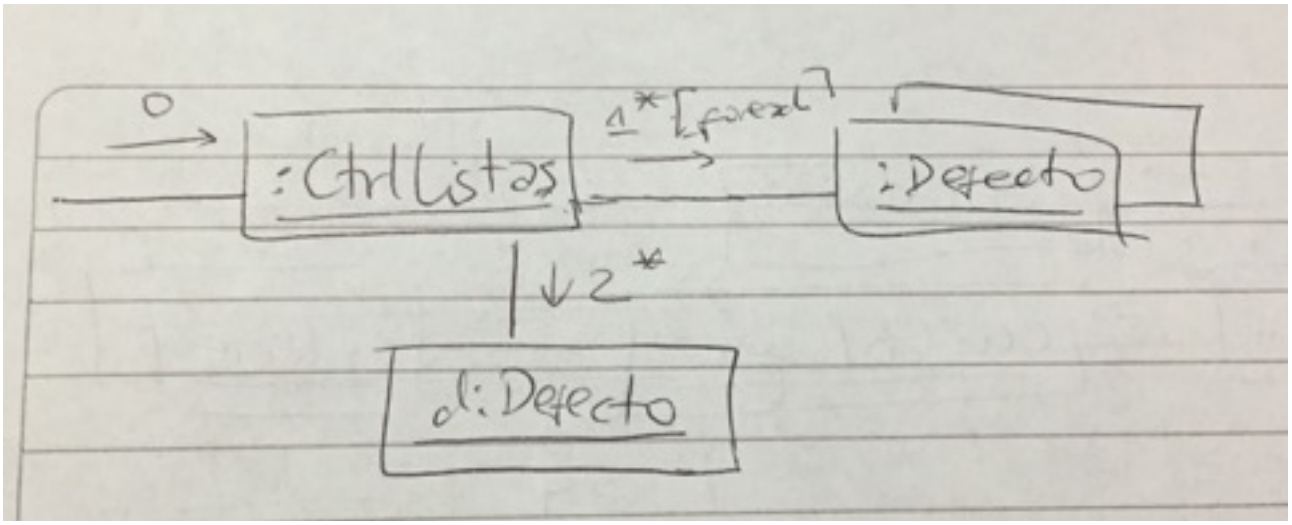
- **Operación: ListarClientes**
  - Compartida.
- **Operación: ListarListasDeCliente**

0:) ListarListasdeCliente(nomCliente : string) : List<DataLista>



1:) cu:=getInstancia()
2:) ListarListasDeCliente(nomCliente)
2.1:) c:=find(nomCliente)
2.2:) ListarListas()
2.2.1*[foreach]:) p:=next()
2.2.2*:) getData()

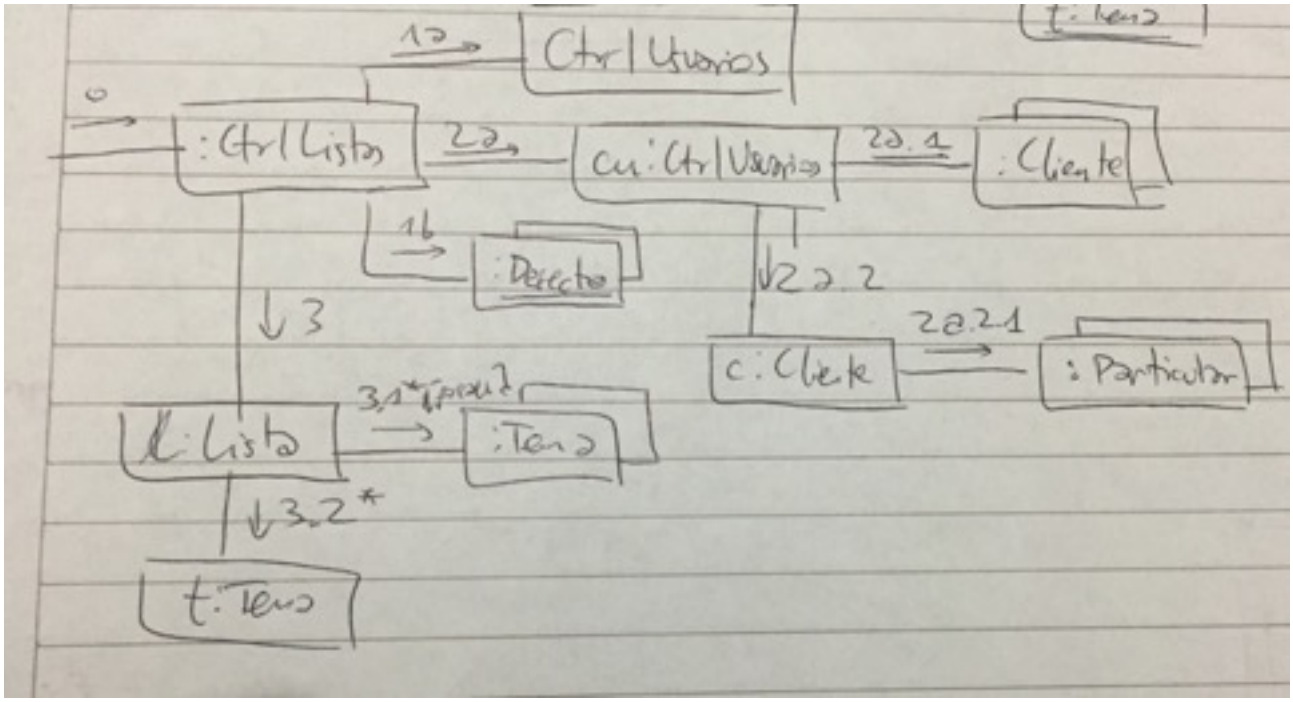
Operación: ListarListasDefecto



0:) ListarListasDefecto(): List<DataDefecto>
1:) d:=next()
2:) getData()

Operación: ListarTemasLista

0:) ListarTemasLista(nomLista : string) : List<DataTema>
1a[se recuerda cliente] :) getInstancia()
2a :) ListarTemasLista(nomCliente: string, nomLista)
2a.1 :) c:= find(nomCliente)

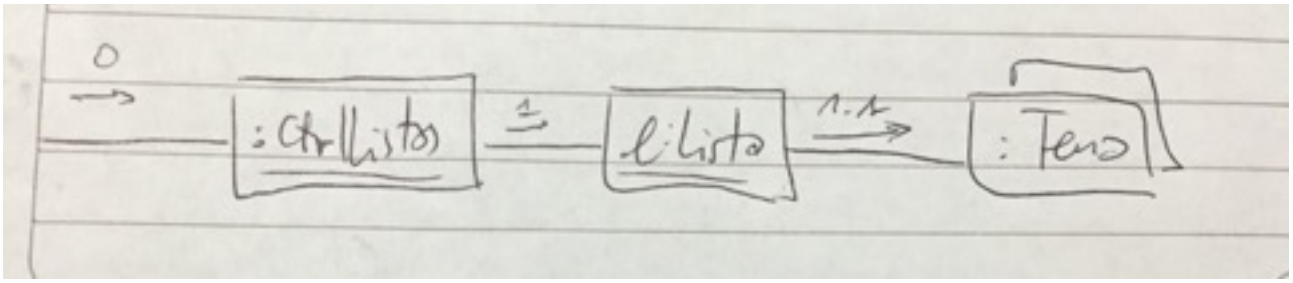


2a.2 :) ListarTemasLista(nomLista)
2a.2.1 :) l:=find(nomLista)
1b[no se recuerda cliente] :) l:=find(nomLista)
3:) ListarTemas()
3.1*[foreach] :) t:=next()
3.2* :) getData()

post: recuerda la lista.

Operación: RemoverTemaLista

pre: se recuerda lista l



0:) RemoverTemaLista(nomTema, albumTema)
1:) RemoverTema(nomTema, albumTema)
1.1:) remove(nomTema, AlbumTema)

Caso de Uso: Publicar Lista

El caso de uso comienza cuando el administrador desea hacer pública una lista de reproducción propietaria. Para ello indica el usuario y la lista que se hará pública y el sistema la hace pública.

Análisis: DSS

ListarClientes
ListarListasDeCliente //recuerda cliente
PublicarLista

Diseño: DCOMM

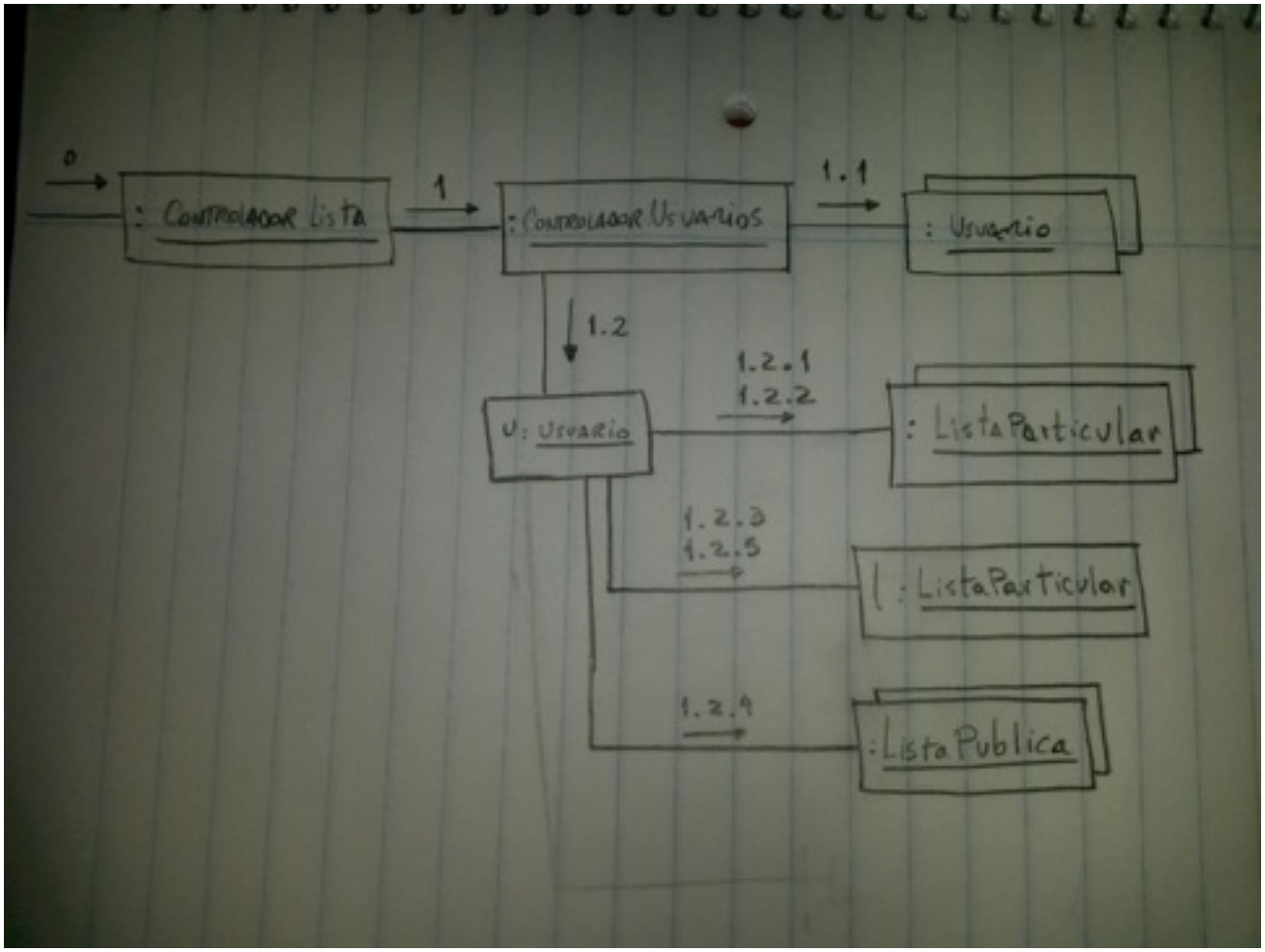
Operación: ListarClientes

comparte de ConsultaCliente

Operación: ListarListasDeCliente

comparte.

Operación: PublicarLista



0 : PublicarLista(nombreLista, nickname: String)
1 : PublicarLista(nombreLista, nickname: String)
1.1 : u := find(nickname: String): Usuario
1.2 : publicarLista(nombreLista: String)
1.2.1 : l := find(nombreLista: String): ListaParticular
1.2.2 : remove(l)

1.2.3 : lpub := devolverComoPublica(): ListaPublica
1.2.4 : Add(lpub)
1.2.5 : Destroy()

Datatypes:

[DataCliente]-:>[DataUsuariol
-nick:string
-nombre:string
-apellido:string
-correo:string
-fNac:Date
-img]

[DataUsuario]<:-[DataArtistal
-bio:string
-url:string]

[DataTemal
-nombre:string
-duracion:int?
-numero:int]
[DataTema]<:-[DataTemaWebl-url:string]
[DataTema]<:-[DataTemaArchivol-archivo:string?]

[DataAlbuml
-nombre:string
-año:int
-generos:Set<String>
-img:string]

[DataClienteExtl
…
]

[DataArtistaExtl
…
]

[DataAlbum]<:-[DataAlbumExtl
-temas:List<DataTema>]

[DataGenerol
-nombre:string
-padre:string]

[DataLista]<:-[DataParticularl
-nomCliente:string
]

[DataLista]<:-[DataDefectol
-genero:string
]

[DataListal
-nombre:string
-img:string]