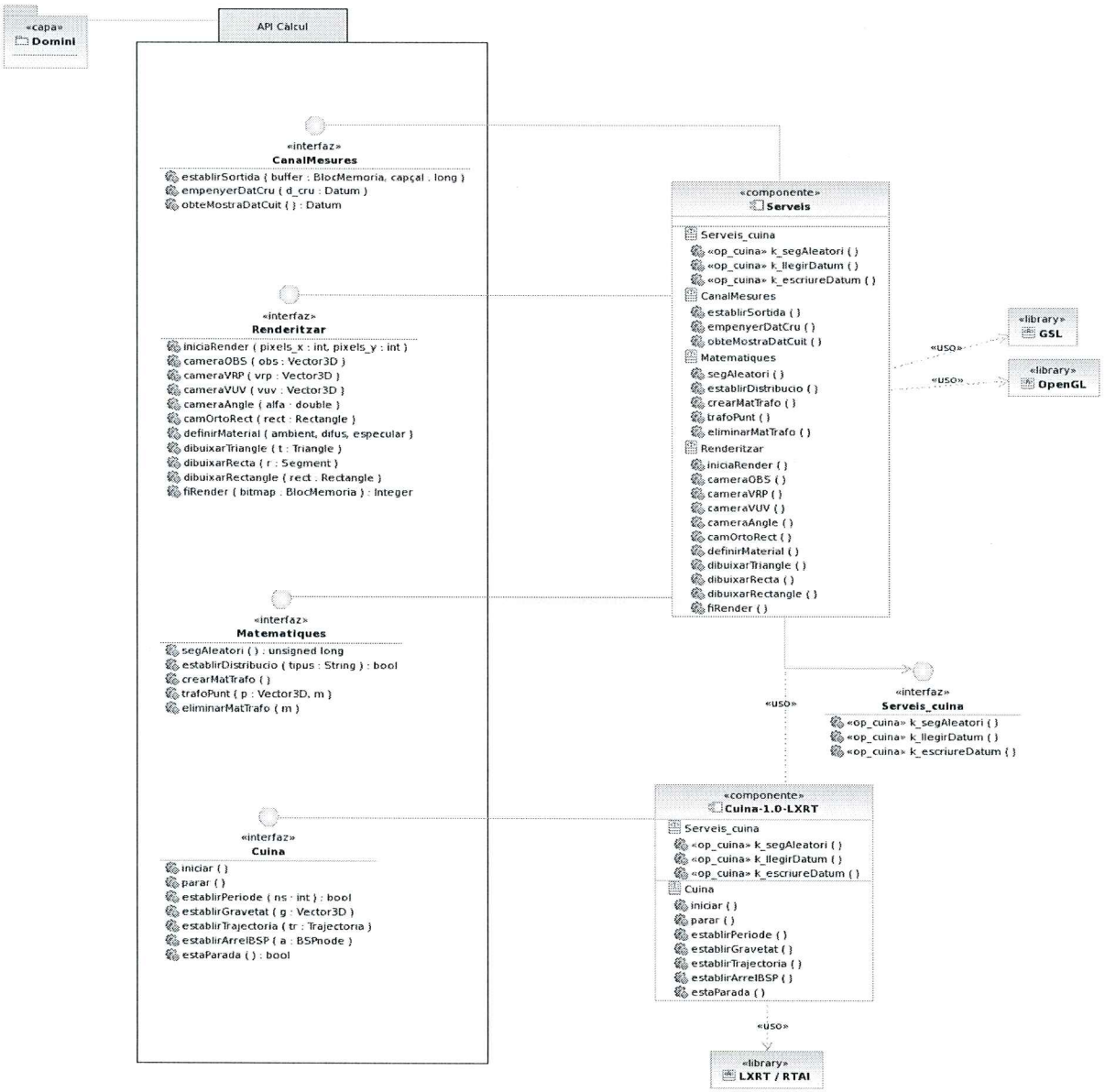
«subsystem»
Motor Calcul Temps Real

Disseny del subsistema: motor de càlcul en temps real

Essent aquest el cor del servidor, amb l'afegit de treballar en temps real, s'ha de perseguir un disseny òptim en eficiència. No seria adequat un disseny basat en objectes ja que la comunicació entre classes té una petita penalització de rendiment que no ens podem permetre treballant en temps real. Així doncs aquest subsistema estarà exterioritzat i s'utilitzarà a través d'un conjunt d'interfícies agrupades en una API.



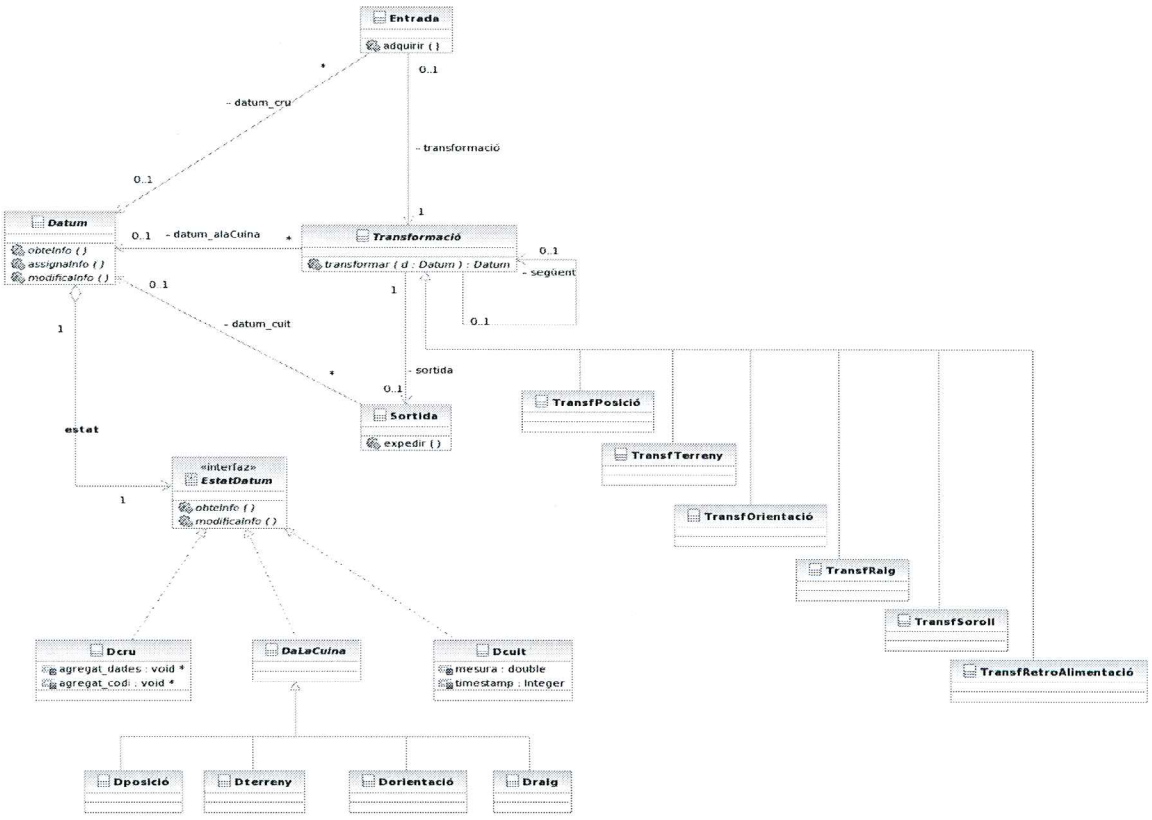
El disseny està basat en el patró arquitectònic del microkernel [RB011] formant així un motor de càlcul composat principalment per dos components: el nucli i el proveïdor de serveis. Fent un símil amb el món de la restauració el nucli del subsistema ha estat anomenat "Cuina" dins de la qual es treballa en temps real coent la teca mentre que el component "Serveis" posa a disposició el produït dins la cuina juntament amb altres utilitats relacionades.

Conjunt de serveis a disposar

1. Interfície -CanalMesures-

Controlador del flux de dades de telemetria de l'escànner. Aquest flux es generat mitjançant el patró arquitectònic del canal el qual pot ser abstret com una tovera que, seqüencialment, transforma les dades des de un valor inicial cap a un valor final mitjançant un seguit d'etapes. L'element que es transportat dins del canal és el Datum, unitat d'informació bàsica del flux que partint d'un estat cru (entrada) es va transformant fins a arribar a cuit (sortida). Llavors pot ser expedit o be reutilitzat (retroalimentació).

1.1. Aplicació del patró arquitectònic del canal: Vista estàtica [RB011]



En aquesta representació s'utilitza la forma de classe i relació UML per representar les entitats principals de l'arquitectura, no obstant aquestes entitats poden ésser materialitzades en components de naturalesa mes eficient.

Entitat	Documentació
DaLaCuina	Macro-estat del Datum quan aquest està patint les transformacions de la cuina.
Datum	Contenidor d'informació susceptible a sofrir transformacions. És la unitat bàsica d'intercanvi de dades entre el domini del servidor i aquest subsistema.
Dcru	Estat del Datum abans d'entrar a la cuina. Es pot generar un Datum en aquest estat en qualsevol moment a partir del domini del servidor.
Dcuit	Estat del Datum després de sortir de la cuina. Té com a atributs resultants la distància mesurada per telemetria i l'instant de temps en que s'ha llegit.
Dorientacio	Estat en que es queda un Datum després d'aplicar la transformació pròpia de TransfOrientacio a un Datum en estat Dterreny
Dposicio	Estat en que es queda un Datum després d'aplicar la transformació pròpia de TransfPosicio a un Datum en estat Dcru
Draig	Estat en que es queda un Datum després d'aplicar la transformació pròpia de TransfRaig a un Datum en estat Dorientacio
Dterreny	Estat en que es queda un Datum després d'aplicar la transformació pròpia de TransfTerreny a un Datum en estat Dposicio
Entrada	Punt d'accés a la tovera on s'esperen Datum en estat Dcru
EstatDatum	Abstracció dels diferents estats en que es pot trobar el Datum
Sortida	Final de la tovera a través de la qual s'expedeixen Datum en estat Dcuit
Transformacio	Interfície comú per a totes les transformacions dels Datum
TransfPosicio	Transformació consistent en calcular la nova posició del vehicle dins de l'entorn en funció de l'acceleració i el temps
TransfTerreny	Transformació consistent en calcular l'elevació del terreny en funció de la posició del vehicle dins de l'entorn.
TransfOrientacio	Transformació consistent en calcular l'orientació en que queda el vehicle al trepitjar el terreny
TransfRaig	Transformació consistent en calcular l'impacte d'un raig segons la posició i orientació d'un sistema de referència dins de l'entorn.
TransfSoroll	Transformació consistent en afegir l'error corresponent a una mesura.
TransfRetroAlimentacio	Transformació consistent en convertir un Datum en estat Dcuit a estat Dcru.

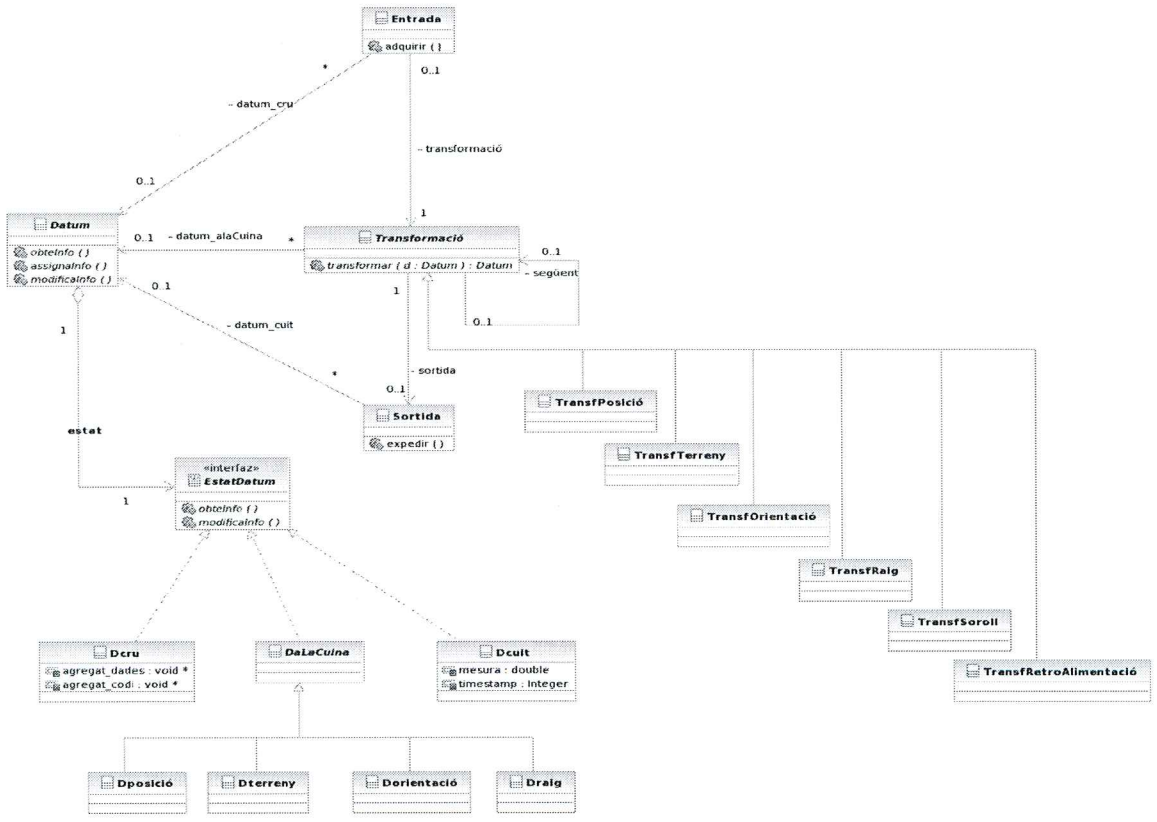
El disseny està basat en el patró arquitectònic del microkernel [RB011] formant així un motor de càlcul composat principalment per dos components: el nucli i el proveïdor de serveis. Fent un símil amb el món de la restauració el nucli del subsistema ha estat anomenat "Cuina" dins de la qual es treballa en temps real coent la teca mentre que el component "Serveis" posa a disposició el produït dins la cuina juntament amb altres utilitats relacionades.

Conjunt de serveis a disposar

1. Interfície -CanalMesures-

Controlador del flux de dades de telemetria de l'escànner. Aquest flux es generat mitjançant el patró arquitectònic del canal el qual pot ser abstret com una tovera que, seqüencialment, transforma les dades des de un valor inicial cap a un valor final mitjançant un seguit d'etapes. L'element que es transportat dins del canal és el Datum, unitat d'informació bàsica del flux que partint d'un estat cru (entrada) es va transformant fins a arribar a cuit (sortida). Llavors pot ser expedit o be reutilitzat (retroalimentació).

1.1. Aplicació del patró arquitectònic del canal: Vista estàtica [RB011]



En aquesta representació s'utilitza la forma de classe i relació UML per representar les entitats principals de l'arquitectura, no obstant aquestes entitats poden ésser materialitzades en components de naturalesa mes eficient.

Entitat	Documentació
DaLaCuina	Macro-estat del Datum quan aquest està patint les transformacions de la cuina.
Datum	Contenidor d'informació susceptible a sofrir transformacions. És la unitat bàsica d'intercanvi de dades entre el domini del servidor i aquest subsistema.
Dcru	Estat del Datum abans d'entrar a la cuina. Es pot generar un Datum en aquest estat en qualsevol moment a partir del domini del servidor.
Dcuit	Estat del Datum després de sortir de la cuina. Té com a atributs resultants la distància mesurada per telemetria i l'instant de temps en que s'ha llegit.
Dorientacio	Estat en que es queda un Datum després d'aplicar la transformació pròpia de TransfOrientacio a un Datum en estat Dterreny
Dposicio	Estat en que es queda un Datum després d'aplicar la transformació pròpia de TransfPosicio a un Datum en estat Dcru
Draig	Estat en que es queda un Datum després d'aplicar la transformació pròpia de TransfRaig a un Datum en estat Dorientacio
Dterreny	Estat en que es queda un Datum després d'aplicar la transformació pròpia de TransfTerreny a un Datum en estat Dposicio
Entrada	Punt d'accés a la tovera on s'esperen Datum en estat Dcru
EstatDatum	Abstracció dels diferents estats en que es pot trobar el Datum
Sortida	Final de la tovera a través de la qual s'expedeixen Datum en estat Dcuit
Transformacio	Interfície comú per a totes les transformacions dels Datum
TransfPosicio	Transformació consistent en calcular la nova posició del vehicle dins de l'entorn en funció de l'acceleració i el temps
TransfTerreny	Transformació consistent en calcular l'elevació del terreny en funció de la posició del vehicle dins de l'entorn.
TransfOrientacio	Transformació consistent en calcular l'orientació en que queda el vehicle al trepitjar el terreny
TransfRaig	Transformació consistent en calcular l'impacte d'un raig segons la posició i orientació d'un sistema de referència dins de l'entorn.
TransfSoroll	Transformació consistent en afegir l'error corresponent a una mesura.
TransfRetroAlimentacio	Transformació consistent en convertir un Datum en estat Dcuit a estat Dcru.

El patró arquitectònic del canal simplifica algorismes que fàcilment poden ser descomposts en una sèrie d'etapes (Transformacions) les quals treballen sobre elements aïllats (Datum). En conseqüència, l'arquitectura resulta fàcilment adaptable per operar en paral·lel de forma segmentada millorant de manera substancial el rendiment del subsistema.

1.2. Operativa de -CanalMesures-

Servei	Acció que realitza
establirSortida	Direccionar tot el flux de dades resultant (datums cuits) cap a un espai de memòria on s'expedeix informació. Es com canviar de lloc la sortida d'una tovera.
empenyerDatCru	Aportació d'un Datum en estat cru. Aquest serà l'única llavor del futur flux ja que el subsistema es retroalimenta.
obteMostraDatCuit	Com que el flux de sortida pot arribar a ser molt intensiu és útil aquesta operació per mostrejar-lo a una freqüència mes baixa.

2. Interfície -Renderitzar-

Es basa en la llibreria gràfica OpenGL (rutines de renderitzat fora de pantalla "OSrender") per a calcular les imatges a color pròpies de les càmeres presents en l'entorn.

Servei	Acció que realitza
cameraOBS	Situa la càmera en el punt d'observació indicat (punt OBS)
cameraVRP	Enfoca al punt indicat (punt VRP)
cameraVUV	Especifica la vertical de la càmera (vector VUV)
camOrtoRect	Si s'utilitza aquesta operació llavors la configuració de la càmera passa a ser ortonormal. El seu camp de visió tindrà forma d'hexaedre la secció del qual estarà definida pel paràmetre "rect" i la seva altura serà el mòdul del vector entre els punts OBS i VRP.
definirMaterial	Indica les característiques del material per a calcular la il·luminació
dibuixarTriangle	Primitiva de representació dels models tridimensionals.
dibuixarRectangle	Primitiva per a representar plans
fiRender	Retorna un vector de ternes RGB corresponents als pixels de la imatge realitzada.

Servei	Acció que realitza
iniciaRender	Reserva els recursos necessaris per obtenir imatges de les dimensions indicades

3. Interfície -Matemàtiques-

Es basa en la llibreria científica GSL (inicials en anglès de GNU Scientific Library) per a realitzar calculs de generació de nombres pseudoaleatoris de qualitat i també per treballar amb matrius.

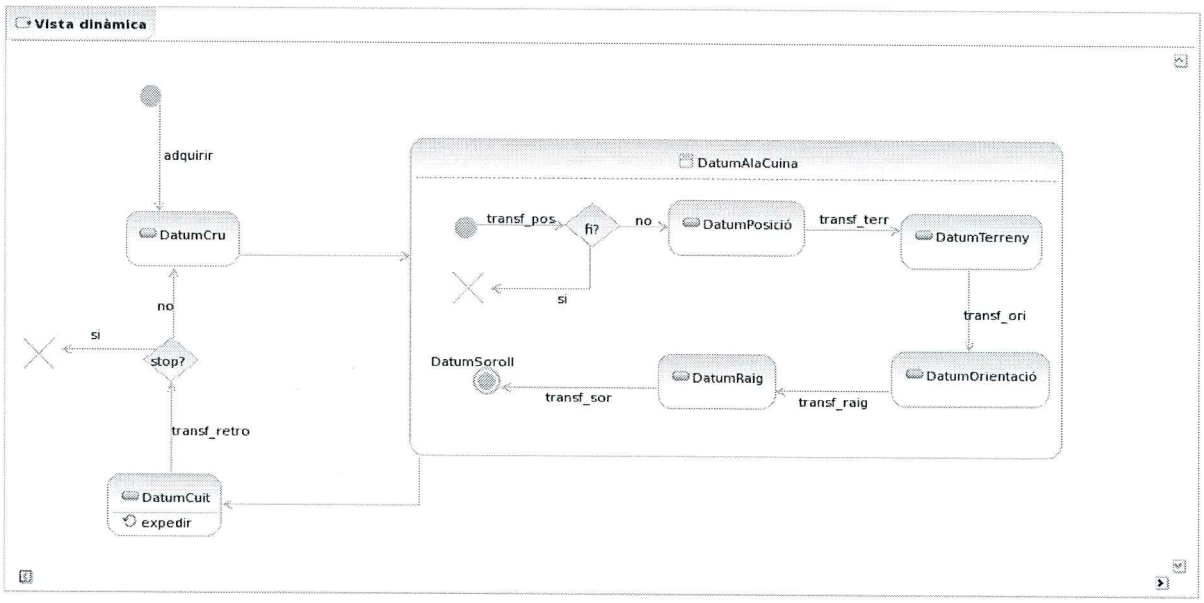
Servei	Acció que realitza
crearMatTrafo	Reserva els recursos per a una nova matriu de transformació homogènia i retorna un identificador per treballar-hi posteriorment.
eliminarMatTrafo	Allibera els recursos ocupats per la matriu de transformació homogènia.
establirDistribucio	Canvia el model de distribució estadísitica que segueix la variable aleatoria.
segAleatori	Retorna nou valor de la variable aleatoria. Mai es repeteix una sèrie perquè es guarda de manera persistent l'ultima llavor utilitzada.
trafoPunt	Transforma un punt segons la matriu de transformació homogènia indicada.

4. Interfície privada -ServeisCuina-

A través d'aquesta interfície especial la cuina accedeix de la manera mes eficient possible als serveis existents.

El "kernel" del subsistema: La cuina

A l'interior de la cuina es pot veure l'aplicació final del patró arquitectònic del canal. És aquí on tenen lloc les transformacions dels Datum. A cada període de temps predeterminat, precís i constant esdevé un canvi d'estat del Datum segons el següent diagrama.



En el diagrama d'estats s'aprecia la retroalimentació.

Per tal de proveir aquesta base de temps real, es proposa utilitzar un sistema operatiu Linux amb l'extensió de temps real RTAI*.

5. Interfície -Cuina-

L'objectiu principal d'aquesta interfície és introduir els paràmetres que resten constants durant tot el procés de càlcul així com oferir el comandament de la cuina

Servei	Acció que realitza
establirArrelBSP	Introdueix l'estructura geomètrica del món la qual és constant durant tot el procés de càlcul
establirGravetat	El vector indica tant la direcció com la intensitat de la força de gravetat que es tindrà en compte durant tot el procés de càlcul
establirPeriode	Indica el lapse de temps que tarda l'escànner en llegir una mesura
establirTrajectoria	Introdueix la trajectòria sobre la que es desplaça el vehicle la qual és constant durant tot el procés de càlcul

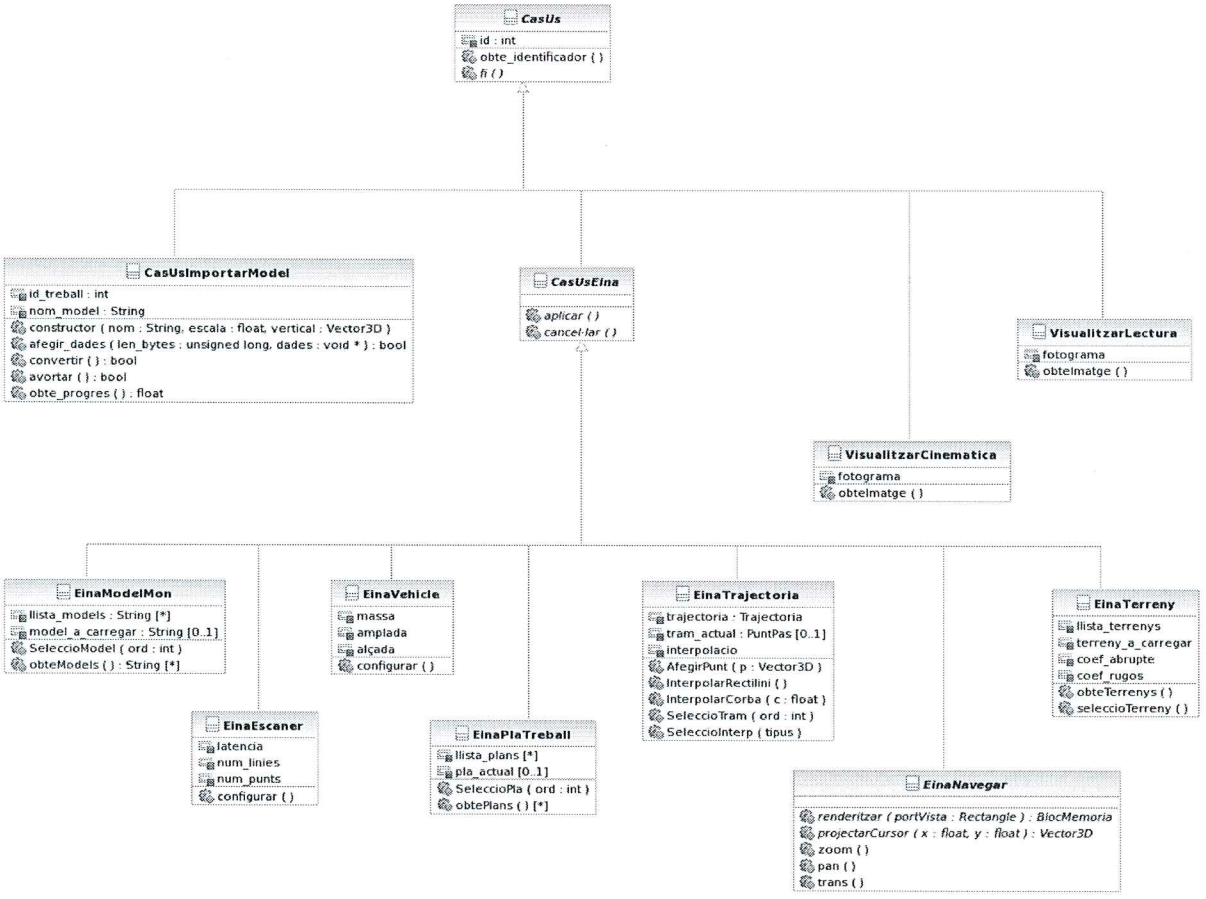
*Vegeu annex A

Servei	Acció que realitza
estaParada	Consulta l'estat de funcionament de la cuina.
iniciar	Provoca el naixement del flux de mesures a través del canal extedint un Datum cuit cada període establert.
parar	Interromp el flux de mesures.

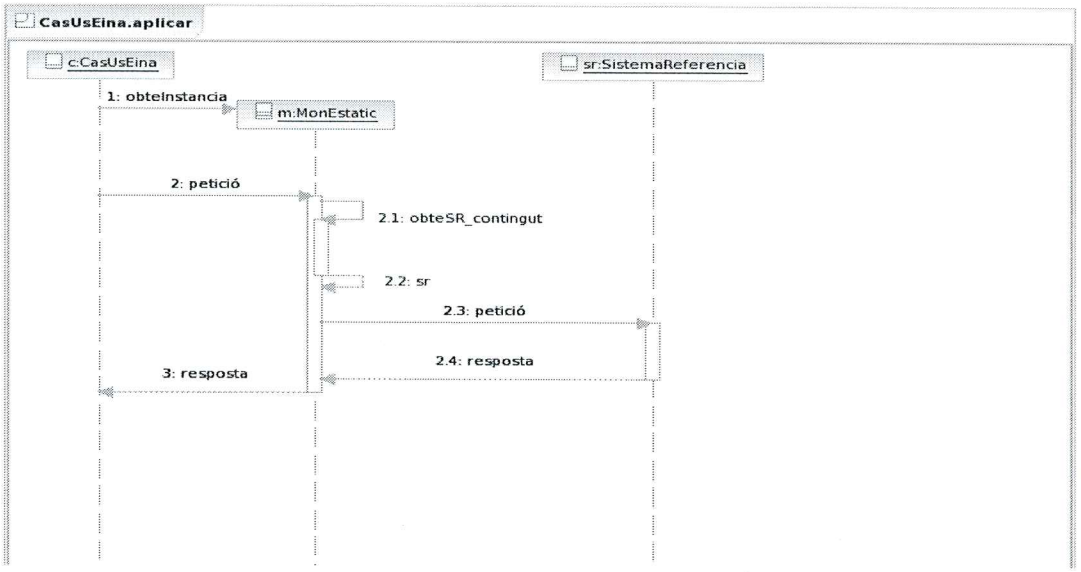
Controladors de la capa de Domini

Es divideixen en dos grups, un segueix el patró del controlador de transacció i l'altre segueix el patró del controlador de cas d'ús. La diferència mes notòria és que el controlador de cas d'ús manté un estat durant l'execució les diverses accions mentre que el controlador transacció es crea, executa la seva única acció i després de retornar els resultats es destrueix la instància.

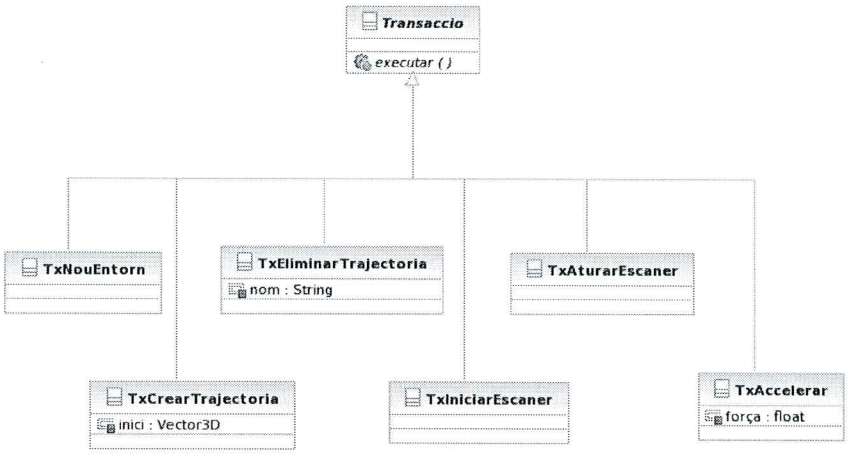
- *Controladors de cas d'ús*



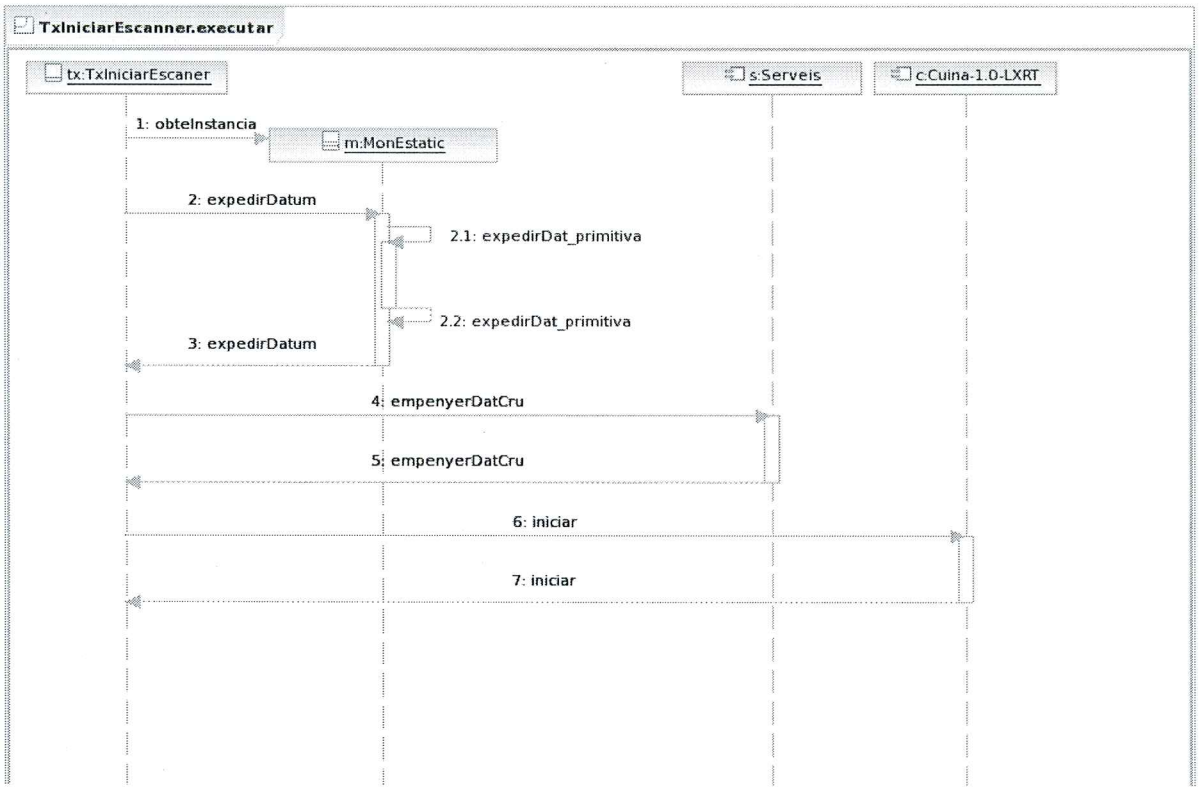
- *Dinàmica típica durant l'edició de l'entorn*



- *Controladors de transacció*



- *Dinàmica típica de l'ús del subsistema: motor de càlcul en temps real*



Gestió dels errors

Tenint en compte la convenció de tractament dels errors (apartat 7.2.1) la capa de domini llençarà una excepció tipus "EcapaDomini" en cas de fallada.

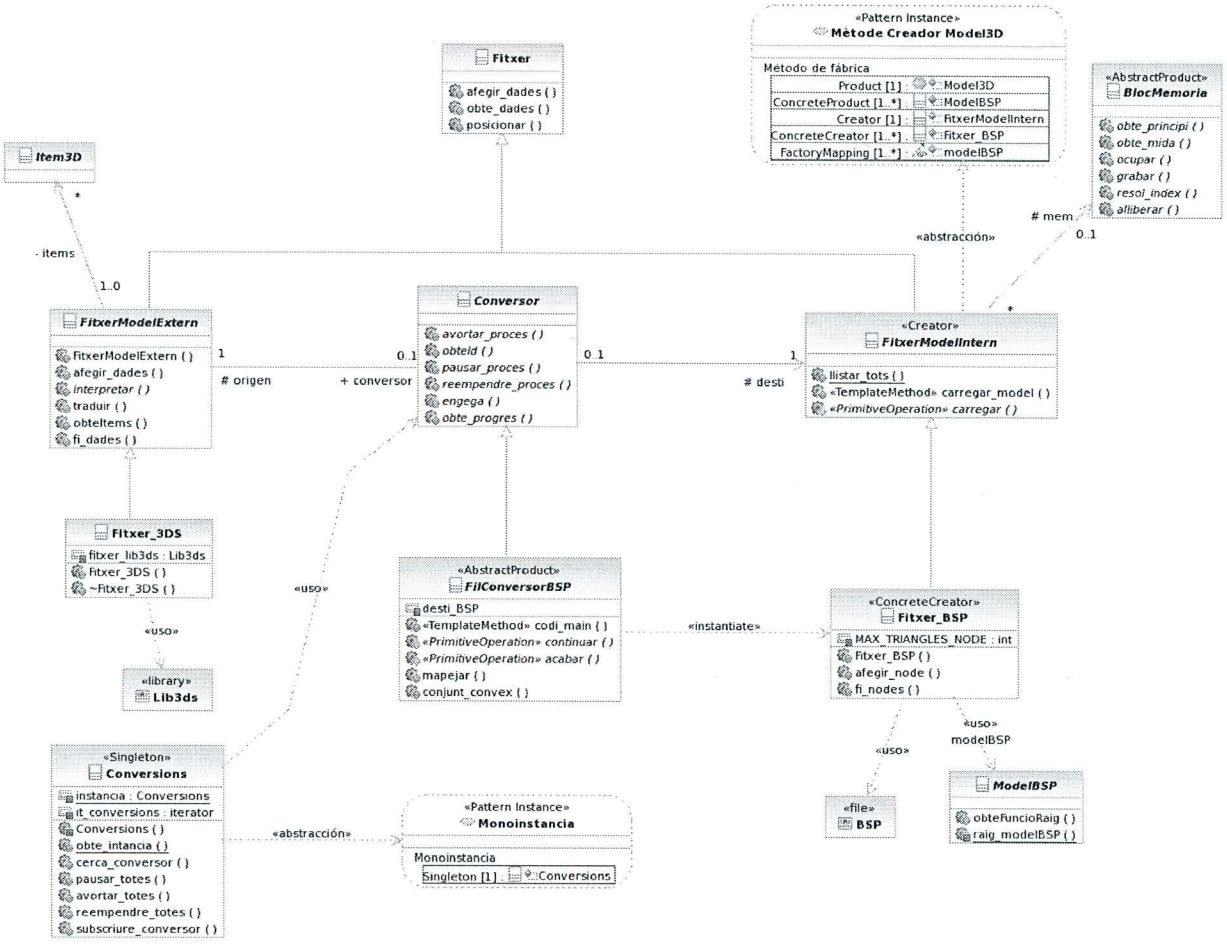
7.4.4 Capa de Dades

Responsabilitat principal de la capa

En primer lloc la capa té l'objectiu de fer persistents els models tridimensionals basats en conjunts d'Item3D.

El disseny d'aquesta capa té dos vessants. Per una banda interpretar fitxers de models en format 3DS i per l'altra traduir segons l'estructura BSP i fer-la persistent mitjançant un segon fitxer. No caldria dir que el disseny ha de permetre futures ampliacions, tant en el tipus del fitxer de model extern com el de l'intern, preferiblement mitjançant el mecanisme de l'herència.

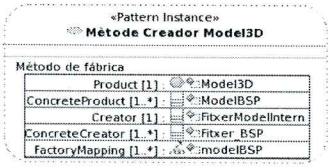
Diagrama de la capa de Dades



Documentació de les classes

Classe	Comentari
BlocMemoria	Proporciona espais fixes en la memòria RAM (espais no paginables pel SO).
Conversor	Abstracció del procés de codificar un conjunt d'elements Item3D al format intern del sistema.
Conversions	Mono instància de agrupa tots els processos de conversió
FilConversorBSP	Procés de transformar un conjunt d'Item3D, concretament un conjunt de Triangle, al format intern anomenat BSP.
Fitxer	Representa un fitxer físic. Permet crear, modificar i esborrar.
Fitxer_3DS	Fitxer en el format extern propi de l'aplicació "3D Studio"
Fitxer_BSP	Fitxer en format intern BSP
FitxerModelExtern	Abstracció de tots els fitxers que tenen un format extern al sistema.
FitxerModelIntern	Abstracció de tots els fitxers que tenen un format propi del sistema
Item3D	Classe del mòdul de geometria que representa a qualsevol cos tridimensional de dimensions finites.
ModelBSP	És el resultat de carregar un fitxer tipus BSP.

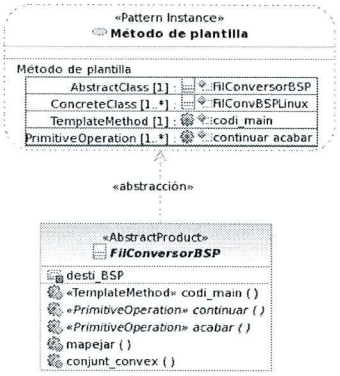
Futures ampliacions del tipus "suportar un nou format de fitxer extern" o bé "implementar una altra estructura interna de Model3D" es realitzarien mitjançant el mecanisme de l'herència de les classes FitxerModelExtern i FitxerModelIntern, respectivament.



Aplicació del patró de disseny

Els objectes Model3D, concretament els ModelBSP, son creats a través del mètode de fàbrica resident en la classe FitxerModelIntern "carregar()→Model3D". Aquest, derivat en la classe Fitxer_BSP és l'encarregat de descodificar el contingut del fitxer i instanciar un objecte de la classe ModelBSP amb la informació obtinguda.

Aplicació del patró de disseny



En aquest cas s'ha aplicat el patró de disseny "Mètode de plantilla" sobre la classe FilConversorBSP que, a la vegada, és de la família de productes abstractes introduïda pel patró de la "Fàbrica abstracta" (vegeu apartat 7.4.1). És un procés que ha de córrer de forma concurrent en un fil apart perquè la seva execució en el cas de l'ordenació de triangles segons el mètode BSP necessita un temps quadràtic respecte del nombre de triangles. D'aquesta manera es pot tenir diversos processos treballant en mode "batch".

Tornant al patró "Mètode de plantilla", seria convenient que el procés es pogues interrompre per tant es dividirà l'operació de conversió en:

- Primitiva: `FilConversorBSP::continuar()→Booleà`

  Mètode abstracte que pregunta si el fil ha rebut el senyal d'acabar per la força, cada sistema operatiu ho implementa a la seva manera.

- *Primitiva:* `FilConversorBSP::acabar()`

  Mètode abstracte que finalitza el fil, cada sistema operatiu ho implementa a la seva manera.

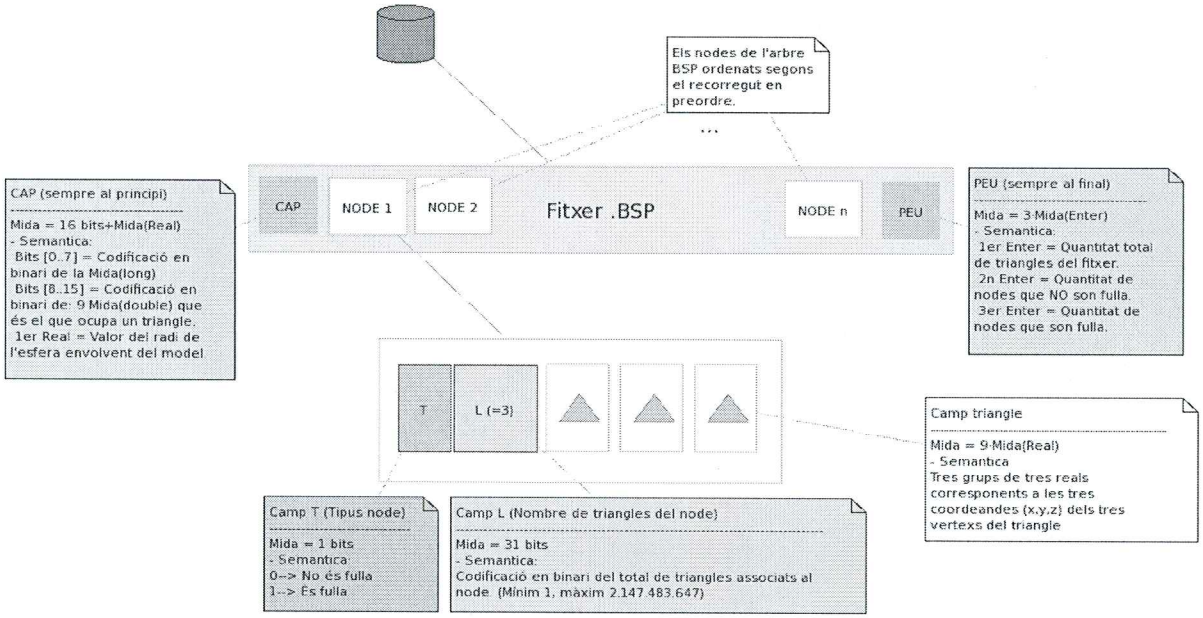
- Mètode plantilla: `FilConversorBSP::codi_main()`

Codi del procés de generar i emmagatzemar un arbre BSP a partir d'un conjunt desordenat de triangles (vegeu apartat 4.4.1), això és independent del sistema operatiu.

«file»
BSP

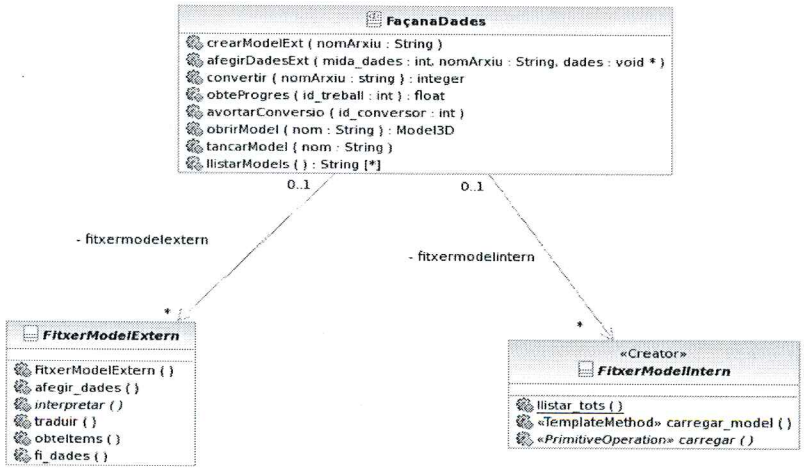
Format del fitxer BSP

Tenint en compte que els models basats en l'ordenació de triangles BSP han de ser persistents en un fitxer, s'haurà de definir el seu format tot descrivint els camps en que està dividit.



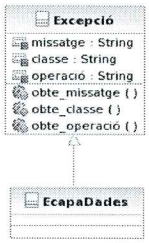
Controladors de la capa de Dades

Gràcies a que l'operativa de la capa està centrada en la gestió de models resulta adequat aplicar el patró controlador "Façana" definint d'aquesta manera la interfície d'accés a la capa de dades.



Com ho és la de la capa de Dades, la missió d'aquesta façana és gestionar el servei de càrrega/descàrrega d'objectes Model3D.

Gestió dels errors



Tenint en compte la convenció de tractament dels errors (apartat 7.2.1) la capa de dades llençarà una excepció tipus “EcapaDades” en cas de fallada.

8 Construcció (Iteració C1)

El contingut d'aquesta etapa és el codi C++ i Java dels prototips i de l'aplicació. Tot això es troba en el CD que acompanya aquesta memòria. El sistema de fitxers ha estat generat mitjançant l'entorn de desenvolupament Rational Software Architect versió 7.0 excepte la carpeta "Prototip" el codi de la qual ha estat desenvolupat amb el Visual C++ versió 6.0. No obstant per compilar-ho només es necessari tenir instal·lat el gcc (mínim versió 3) amb l'eina makefile i el JDK de Java (mínim versió 1.5).

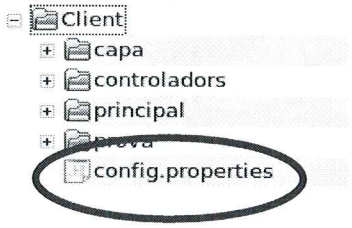
La implementació de l'aplicació es parcial. La seva continuació es podria realitzar en mode ascendent (primer servidor després client) o descendent (primer client i després servidor) ja que ambdós opcions tenen implementat el seu començament.

- Implementació descendent: Capa de presentació del client

Client	4,0 KB
capa	4,0 KB
dadescom	4,0 KB
domini	4,0 KB
presentacio	4,0 KB
DialegAlgorismes.java	4,7 KB
DialegAvaluacions.java	4,4 KB
DialegConectar.java	3,4 KB
DialegDesar.java	4,0 KB
Dialeg.java	4,8 KB
DialegObrir.java	4,2 KB
DialegVehicle.java	4,3 KB
EinaCanviMon.java	2,6 KB
EinaNavegar.java	3,4 KB
EinaSensor.java	4,0 KB
EinaTrajectoria.java	2,1 KB
EinaVehicle.java	6,3 KB
Finestra.java	20,6 KB
Fons.java	889 Bytes
Importacions.java	1,9 KB
PanellGraficAnimat.java	818 Bytes
PanellGrafic.java	658 Bytes
PanellInicial.java	1,1 KB
VistaSensor.java	593 Bytes

Classes corresponents a les vistes

- Suport al idioma de la interfície gràfica del client



El fitxer senyalat emmagatzema totes les frases que apareixen a la interfície gràfica del client. Suposant que es vulgues traduir l'aplicació només faria falta modificar aquest fitxer de text, no caldria tornar a compilar l'aplicació.

- Implementació ascendent: Capa de dades del servidor

El servidor és la part de l'aplicació mes avançada. La capa de dades està al 100% amb controladors inclosos. A més a més els moduls de geometria i de la fabrica abstracta també estan operatius.

- Generació automàtica de codi

Per als usuaris de l'entorn Rational Software Architect, s'adjunta la carpeta Modelatge, la qual és el projecte de disseny del sistema en llenguatge UML que pot ser portat a codi de forma automàtica.

- Prototip

A la carpeta "Prototip" del CD hi ha petits programes que implementen les estructures de dades i algorismes explicats en l'apartat d'anàlisi.

9 Conclusió

S'ha de dir que els resultats obtinguts pel que fa a especificació, disseny i construcció del programari corresponen als esperats en el moment de plantejar l'objectiu del projecte. La tasca inicial d'especificació ha estat notòriament dificultosa donat que no hi havia cap pauta prèvia a seguir. És àmpliament sabut que en el desenvolupament d'una aplicació la formalització dels requisits sempre comporta un esforç extraordinari, es a dir, sovint passa que costa més desxifrar *que* ha de fer l'aplicació que no pas *com* ho farà. No obstant la utilització de metodologies estandarditzades, com ho és el RUP, ha estat de gran ajuda en moments d'encallament i, un cop més, demostren ésser fonamentals per a concebre un projecte complex de manera clara i ordenada, factors claus per a l'èxit.

Algorismes SLAM a part, cal esmentar l'adquisició personal de nous coneixements en el camp del modelatge d'entorns de realitat virtual per a la simulació així com de l'arquitectura de sistemes d'alt rendiment, concretament d'aquells que han de treballar en temps real.

Durant l'etapa de disseny s'ha decidit donar forma a la solució basant-se en l'arquitectura client / servidor. D'aquesta manera la càrrega funcional present en l'especificació ha estat repartida entre aquests dos rols, seguint la filosofia del principi "dividir i vèncer". Resumidament, la part client ha absorbit la responsabilitat d'oferir una interfície d'usuari amigable i senzilla juntament amb la de gestionar la persistència dels entorns editats. D'altra banda pesa sobre el servidor la dura tasca de simular en temps real comportaments físics de sensors, vehicles i altres cossos delegant a la infraestructura telemàtica el procés de difusió de dades per tal d'alimentar els consumidors finals: els algorismes SLAM, els quals són avaluats a través de la col·laboració entre ambdós client i servidor.

Per últim queda oberta la tasca de construcció o implementació la qual necessita una major assignació de recursos, tant temporals com humans. Malgrat això, val a dir que el conjunt del disseny i els prototips realitzats permetran, en el futur, una represa ràpida del projecte tot mantenint el compromís de qualitat.

En resum, tant l'especificació i l'anàlisi com el disseny realitzats han suposat complir perfectament els objectius inicials, sempre tenint en compte els recursos i equip de que s'ha pogut disposar per tal de desenvolupar el projecte.

10 Futures millores

Hi ha molts aspectes que han estat retallats per motius de manca de recursos però que haurien d'ésser tinguts en compte de cara a ampliacions futures ja que el disseny del sistema ha previst que sigui factible:

i. Múltiples sessions concurrents en el servidor per a permetre el treball de diversos clients a la vegada.

ii. Simulació en mons virtuals dinàmics, es a dir, que a més a més del vehicle hi hagi altres cossos en moviment.

iii. Simulació d'excursions pel món virtual en mode lliure, es a dir, sense estar lligades a trajectòries definides a priori.

iv. Incorporar sensors que captin imatges d'intensitats i en color.

v. Vehicle amb esmorteïdors.

vi. Compressió de les dades que són transmeses a través de la xarxa per tal de que sigui possible utilitzar internet com a canal de comunicacions.

vii. Ampliar el ventall de formats a partir dels quals es pot importar models de móns virtuals.

viii. Tenir en compte aspectes referents a la seguretat en front a atacs des de la xarxa.

ix. Poder efectuar càrregues parcials de models tridimensionals molt molt grans.

11 Annex A: Instal·lació de Linux RTAI

Introducció:

Aquesta és la guia per a la instal·lació i utilització del sistema operatiu Linux amb l'extensió de temps real RTAI (www.rtai.org). El conjunt resulta perfectament capaç de treballar a temps real fins i tot en aplicacions d'usuari.

L'explicat a continuació es basa en la meva experiència intentant instal·lar RTAI Linux en un PC

Resum dels passos a efectuar

1. Aconseguir una distribució de Linux
2. Descarregar el programari RTAI
3. Descarregar la versió del kernel apropiada
4. Aplicar el pegat i compilar el kernel
5. Configurar i instal·lar el RTAI

Detall de cada pas.

1. Aconseguir una distribució de Linux

En principi qualsevol distribució és vàlida. L'únic requisit indispensable és que tingui el compilador gcc versió 3 o superior, a efectes de poder compilar el codi font del kernel de Linux.

2. Descarregar el programari RTAI

El principal objectiu de RTAI és prendre el control de la màquina sense que el sistema operatiu se n'adoni. Per aconseguir-ho s'ha d'aplicar un pegat al codi font del sistema operatiu i compilar-lo de nou. En acabat poden conviure les aplicacions de temps real amb les que no ho son.

A la pàgina www.rtai.org es pot descarregar de forma gratuïta l'ultima versió estable la qual bé empaquetada dins d'un arxiu comprimit (.tar.bz2). D'ara en endavant es farà referència a la carpeta de destí on hi ha desempaquetat l'arxiu com:

~/RTAI/

Es recomana descarregar l'ultima versió estable.

3. Descarregar la versió del kernel apropiada.

Sigui quina sigui la distribució de Linux que tinguem instal·lada no es aconsellable utilitzar la font del kernel sovint inclosa ja que la majoria de distribucions incorporen pegats que interfereixen i fan incompatible l'adaptació amb RTAI.

Les diferents versions del codi font del kernel de Linux "en estat pur" (sense pegats) es pot trobar a l'adreça www.kernel.org

La versió de kernel que hem d'aconseguir depen de la versió de RTAI de que disposem. Per esbrinar-ho cal veure els kernel que són compatibles. En el cas de la versió 3.1 el directori de pegats té:

```
ls ~/RTAI/rtai-core/arch/i386/patches
```

```
hal16-2.4.24.patch
hal16-2.4.25.patch
hal16-2.4.26.patch
hal16-2.4.27.patch
hal6-2.6.7.patch
hal7-2.6.7.patch
hal7-2.6.8.1.patch
```

Ja es pot endevinar que les versions 2.4.24 fins 2.4.27 estan suportades així com la 2.6.7 i 2.6.8.1.

El prefix "hal" significa "hardware abstraction layer" que té que veure amb la manera amb la qual RTAI pren el control del sistema deixant al Linux ben cregut de que opera sol.

El més aconsellable seria descarregar-se la versió del kernel més actual de les suportades pel RTAI.

4. Aplicar el pegat RTAI sobre el kernel del Linux

Hi ha pegats pel kernel del Linux que el modifiquen per diverses necessitats, sovint per donar suport a perifèrics. Pegats com aquests són necessaris per fer que el kernel treballi amb l'extensió RTAI.

Per aplicar els pegats s'utilitza la comanda "patch" executada a l'arrel de l'estructura de directoris que contenen el kernel. Per exemple suposem que tenim descarregat i desempaquetat el kernel de Linux versió 2.6.8.1 al directori

```
~/linux-2.6.8.1/
```

Situats en quest directori utilitzaríem la comanda "patch" de la següent manera:

```
patch -p1 < ~/RTAI/rtai-core/arch/i386/patches/hal7-2.6.8.1.patch
```

Prèviament, seria convenient utilitzar la comanda "patch" amb el paràmetre "--dry-run" el qual comprova que el pegat en qüestió s'acomoda sense errors sobre el codi font del kernel sense arribar a tocar-lo.

Un cop aplicat el pegat arriba l'hora de confgurar i compilar de nou el kernel. Les comandes necessàries serien

```
cd ~/linux-2.6.8.1/
cp arch/i386/defconfig .config
make oldconfig
make menuconfig
```

(Si estem en un entorn gràfic en comptes de `make menuconfig` utilitzar `make xconfig`)

En aquest punt cal configurar totes les opcions del kernel de manera que doni suport a tots els dispositius de la nostra màquina. El nou kernel tindrà el sufix 2.6.8.1-tempsreal el qual servira per diferenciar-lo d'altres versions que tinguem instal·lades en l'equip.

Seguint amb la construcció del kernel:

```
make clean
make bzImage
make modules
```

En aquest punt ja hi hauria el codi compilat del kernel. Ara seria l'hora d'instal·lar-lo, per suposat, com a superusuari.

```
su
make modules-install
mkinitrd /boot/initrd-2.6.8.1-tempsreal.img 2.6.8.1-tempsreal
cp arch/i386/boot/bzImage /boot/vmlinuz-2.6.8.1-tempsreal
```

```
cp System.map /boot/System.map-2.6.8.1-tempsreal
ln -s /boot/System.map-2.6.8.1tempsreal /boot/System.map
```

Suposant que el gestor de l'arrencada es el grub s'hauria d'editar el fitxer: /boot/grub/menu.lst

Per tal d'arrencar amb nou kernel hauriem d'afegir una entrada nova similar a:

```
title linux-2.6.8.1-tempsreal
kernel (hd0,0)/vmlinuz-2.6.8.1-tempsreal root=/dev/hda2 devfs=mount
resume=/dev/hda3
initrd (hd0,0)/initrd-2.6.8.1-tempsreal.img
```

Es possible que segons la configuració del PC s'hagi d'adaptar el fragment anterior canviant els identificadors de dispositiu hd0, hda2, hda3.

L'última acció que queda per fer és reiniciar el PC i verificar que el nou kernel arrenca bé.

5. Configurar i instal·lar el RTAI

El RTAI es configura de manera similar al procés seguit de configurar el kernel del Linux. Es recomana realitzar la construcció del RTAI en un directori apart. Sigui aquest ~/RTAI/rtai-build/

Les comandes a executar serien:

```
cd ~/RTAI/rtai-build/
make -f ~/RTAI/makefile menuconfig
```

(Si es disposa d'un entorn gràfic: make -f ~/RTAI/makefile xconfig)

Un cop configurades les opcions desitjades es compila el RTAI mitjançant la instrucció

```
make
```

Un cop finalitza el procés de compilació toca instal·lar-ho (com a superusuari)

```
su
```

```
make install
```

Per últim caldria reiniciar i comprovar que no hi ha errors durant l'arrrencada.

Per defecte el RTAI queda instal·lat a la carpeta /usr/realtime

Tanmateix seria convenient executar un test per comprovar que RTAI funciona. Com a superusuari executaríem

```
cd /usr/realtime/testsuite/kern/latency
./run
```

El test mostra uns conjunts de latències que no impressionen gaire fins que un no s'adona que totes les unitats de temps estan en nanosegons i no varien en funció de la càrrega que altres processos puguin exercir sobre el sistema operatiu. Això és temps real.

12 Glossari

BSP
Binary Space Partition

Càmera PAN
Moviment lateral de la càmera

Cubic splines
Corbes de interpolació que utilitzen polinomis de grau tres

Frustum
Camp de visió d'una càmera representat en forma de piràmide truncada

GSL
GNU Scientific Library

GRUB
GRand Unified Bootloader

HAL
Hardware Abstraction Layer

Multicast
En una xarxa, sistema d'enviament de paquets punt-multipunt

OBS
Observer Point

PITCH
Rotació d'un sistema de referència tridimensional sobre l'eix Y

RAM
Random Access Memory

ROLL
Rotació d'un sistema de referència tridimensional sobre l'eix X

RTAI
Real Time Application Interface

RTP
Real Time Protocol

RUP
Rational Unified Process

SLAM
Simultaneous Localisation And Mapping

SOAP
Single Object Access Protocol

TCP/IP
Transport Control Protocol / Internet Protocol

UDP
User Datagram Protocol

UML
Unified Modeling Language

VRP
Viewer Reference Point

VUV
View Up Vector

YAW
Rotació d'un sistema de referència tridimensional sobre l'eix Z

13 Bibliografia

[RB001]:

SOFTWARE DEVELOPMENT AND DOCUMENTATION, MIL-STD-498, U.S. DEPARTMENT OF DEFENSE, DECEMBER 1994

[RB002]:

IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION, VOL. 17, NO. 3, JUNE 2001

[RB003]:

Comen, Thomas H Leiserson, Charles E. and Rivest, Ronald L.: Introduction to Algorithms

[RB004]:

BSP Trees and Polygon removal in real time 3D rendering

[RB005]:

Wikipedia: http://www.wikipedia.org

[RB006 en línia]

http://real.uwaterloo.ca/~sbirkett/syde312-2005/cubic%20splines.pdf

[RB007 en línia]

"Chapter 5. Integral de Línea" http://www.uhu.es/07021/ficheros/Temas/ampte5.pdf

[RB008 en línia]

"La regla de Simpson compuesta"

http://ochoa.mat.ucm.es/~fmacia/Docencia/MN/Matlab/cuad/simp.pdf

[RB009 en línia]

"Perlin Noise" http://freespace.virgin.net/hugo.elias/models/m_perlin.htm

[RB010]

"Computación y sistemas Vol.8 Núm. 2, pp. 132-149"

[RB011]

Bruce Powel Douglas: "Real time design patterns: Robust Scalable Architecture for Real-Time Systems" 2002