

Títol: Dashboard amb estadístiques configurables a nivell d'usuari a partir de la gestió de pacients en TAO

Volum: 1/1

Alumne: Josep Itarte Aguiló

Director: Oscar Galceran

Ponent: Claudia Patricia Ayala Martinez

Departament: ESSI

Gener de 2013

DADES DEL PROJECTE

Títol del Projecte: *Dashboard amb estadístiques configurables a nivell d'usuari a partir de la gestió de pacients en Teràpia Anticoagulant Oral*

Nom de l'estudiant: *Josep Itarte Aguiló*

Titulació: *Enginyeria Tècnica en Informàtica de Sistemes*

Crèdits: *22.5*

Modalitat: *B*

Empresa: *Systemlab Technologies S.A.*

Director: *Oscar Galceran*

Ponent: *Claudia Patricia Ayala Martinez*

Departament: *Enginyeria de Serveis i Sistemes d'Informació (ESSI)*

MEMBRES DEL TRIBUNAL (nom i signatura)

Presidenta: *Maria Dolors Costal Costa*

Vocal: *Rubén Tous Liesa*

Secretaria: *Claudia Patricia Ayala Martinez*

QUALIFICACIÓ

Qualificació numèrica:

Qualificació descriptiva:

Data:

Agraïments

Durant el temps que ha durat el projecte he rebut el suport de molta gent. Hi ha hagut moments bons i dolents, però sempre hi han sigut. Voldria agrair a aquestes persones la seva paciència i comprensió.

En primer lloc, a la meva família i als meus amics pel suport rebut, el seu recolzament al llarg del projecte i per haver-me fet veure que mai és massa tard.

No puc obviar als meus companys de Systemlab per donar-me consells quan els necessitava i, en especial, a l'Oscar Galceran per dirigir el projecte, haver-me ajudat i guiat.

A la Claudia Ayala per acceptar el projecte i per la seva ajuda i comprensió.

I en especial a la Marta per la paciència, tot el suport que m'ha donat i per estar sempre al meu costat.

Índex de continguts

1. Introducció.....	11
1. 1 Introducció general.....	11
1. 2 Descripció del projecte.....	11
1. 2. 1 Context empresarial.....	11
1. 2. 2 Context inicial.....	12
1. 2. 3 Necessitats.....	13
1. 2. 4 Objectius.....	14
1. 3 Decisió tecnològica.....	14
1. 4 Entorn de treball.....	15
1. 5 Glossari.....	15
1. 6 Estructura del document.....	16
2. Estat de l'art.....	19
3. Planificació.....	23
3. 1 Fases del projecte.....	23
3. 1. 1 Contextualització i captura de requeriments.....	24
3. 1. 2 Especificació	24
3. 1. 3 Disseny.....	24
3. 1. 4 Implementació i verificació	24
3. 1. 5 Depuració de la memòria.....	24
3. 2 Planificació inicial.....	25
4. Anàlisi de requeriments.....	27
4. 1 Característiques generals.....	27
4. 2 Requeriments funcionals.....	28
4. 2. 1 Obtenció i tractament de la informació.....	28
4. 2. 2 Representació de la informació.....	29
4. 3 Requeriments no funcionals.....	30
4. 3. 1 Eficiència.....	30
4. 3. 2 Usabilitat i interfície d'usuari.....	30
4. 3. 3 Seguretat.....	30

4. 4	Requeriments de no tècnics.....	30
5.	Especificació.....	33
5. 1	Actors del sistema.....	33
5. 2	Diagrama de casos d'ús.....	34
5. 3	Especificació de casos d'ús.....	36
5. 3. 1	Casos d'ús dels usuaris de tipus Client.....	37
5. 3. 2	Casos d'ús dels usuaris tipus Gestor d'Estadístiques.....	42
5. 4	Model conceptual.....	45
6.	Disseny.....	47
6. 1	Arquitectura de l'aplicació.....	47
6. 2	Afectació dels requeriments en el disseny.....	47
6. 3	Disseny de les capes.....	48
6. 3. 1	Capa de Presentació.....	49
6. 3. 2	Capa de Negoci.....	52
6. 3. 3	Capa de Persistència.....	56
6. 4	Diagrames de seqüència dels casos d'ús.....	59
6. 4. 1	Diagrama de seqüència. Veure resultats estadístics.....	61
6. 4. 2	Diagrama de seqüència. Afegir nova gràfica al mòdul.....	62
6. 4. 3	Diagrama de seqüència. Eliminar gràfica al mòdul.....	63
6. 4. 4	Diagrama de seqüència. Redimensionar gràfica.....	64
6. 4. 5	Diagrama de seqüència. Ressituar gràfica.....	65
7.	Implementació.....	67
7. 1	Capa de Persistència.....	67
7. 1. 2	Noves taules a la base de dades.....	67
7. 1. 3	Oracle, procediments i tasques programades (Jobs).....	71
7. 1. 4	Fitxer de configuració d'usuari.....	74
7. 2	Capa de Negoci.....	75
7. 3	Capa de Presentació.....	77
7. 4	Verificació.....	83
8.	Anàlisi econòmic.....	86
8. 1	Detall del cost inicial.....	86
8. 1. 1	Cost del programari i maquinari.....	87

8. 1. 2 Cost dels recursos humans.....	88
8. 1. 3 Cost inicial total.....	89
8. 2 Detall del cost final.....	90
8. 2. 1 Cost del programari i maquinari.....	90
8. 2. 2 Cost dels recursos humans.....	91
8. 2. 3 Cost final total.....	91
8. 3 Comparativa del cost inicial envers el final.....	92
9. Conclusions.....	93
9. 1 Objectius aconseguits.....	93
9. 2 Planificació final.....	93
9. 2. 1 Desviacions de temps.....	94
9. 2. 2 Duració real del projecte.....	95
9. 3 Futures ampliacions.....	95
9. 4 Experiència personal.....	96
10 Annex.....	98
10. 1 Manual d'usuari.....	98
10. 2 Eines emprades per a la redacció de la memòria.....	103
10. 3 Creació del paquet de procediments SQL.....	103

Índex de figures

Figura 1: Mapa general d'Anthem/HytGold.....	21
Figura 2: Cicle de desenvolupament en cascada.....	23
Figura 3: Taula de les tasques amb les estimacions inicials.....	25
Figura 4: Diagrama de Gantt de la planificació inicial.....	26
Figura 5: Diagrama PERT de la planificació inicial.....	26
Figura 6: Diagrama mostrant les diferències entre tipus d'usuari.....	34
Figura 7: Diagrama amb els nous casos d'ús.....	35
Figura 8: Diagrama de casos d'ús pels usuaris de tipus "Client".....	35
Figura 9: Diagrama de casos d'ús pels usuaris "Gestors d'estadístiques".....	36
Figura 10: Contracte "Veure resultats estadístics".....	37
Figura 11: Contracte "Afegir nova gràfica".....	39
Figura 12: Contracte "Eliminar gràfica del mòdul".....	40
Figura 13: Contracte "Redimensionar gràfica".....	41
Figura 14: Contracte "Ressituar gràfica al mòdul".....	42
Figura 15: Contracte "Calcul estadístic periòdic".....	43
Figura 16: Contracte "Incorporar estadístiques".....	44
Figura 17: Nous elements UML al model de dades.....	45
Figura 18: Nou model de dades complet.....	46
Figura 19: Disseny arquitectura en tres capes.....	49
Figura 20: Patró Façana aplicat a la capa de Presentació.....	51
Figura 21: Patró Façana en la capa de Presentació detallat.....	52
Figura 22: Exemple de patró Façana en la capa de Negoci (1).....	55
Figura 23: Exemple de patró Façana en la capa de Negoci (2).....	56
Figura 24: Disseny de la capa de Persistència.....	59
Figura 25: Diagrama de seqüència genèric.....	60
Figura 26: Diagrama de seqüència. Veure resultats estadístics.....	62
Figura 27: Diagrama de seqüència. Afegir nova gràfica.....	63
Figura 28: Diagrama de seqüència. Eliminar gràfica.....	64
Figura 29: Diagrama de seqüència. Redimensionar gràfica.....	65

Figura 30: Diagrama de seqüència. Ressituar gràfica.....	66
Figura 31: Esquema d'EJB 2.1.....	77
Figura 32: Estructura de les representacions gràfiques.....	77
Figura 33: Exemple de gràfica de tipus sectors.....	78
Figura 34: Implementació dels gràfics més detallada.....	79
Figura 35: Implementació de les línies de gràfiques.....	80
Figura 36: Implementació de la configuració del dashboard.....	81
Figura 37: Una línia de gràfiques d'exemple.....	81
Figura 38: Implementació del diàleg que mostra les gràfiques.....	82
Figura 39: Exemple de diàleg amb les gràfiques disponibles.....	83
Figura 40: Procés de verificació de les estadístiques.....	84
Figura 41: Diagrama de Gantt. Temps real del projecte.....	95
Figura 42: Diagrama PERT. Temps real del projecte.....	95
Figura 43: Pantalla de Login de l'aplicació.....	98
Figura 44: Vista general de la pantalla inicial dels usuaris.....	99
Figura 45: Opció de menú Estadístiques.....	99
Figura 46: Pantalla general en mode configuració.....	100
Figura 47: Afegir una línia nova.....	101
Figura 48: Canviant la mida dels gràfics.....	101
Figura 49: Diàleg d'usuari mostrant les estadístiques disponibles.....	102
Figura 50: Exemple de gràfic amb opcions.....	103

1. Introducció

1.1 Introducció general

Aquest document és la memòria descriptiva del Projecte Final de Carrera de títol *Dashboard amb estadístiques gràfiques configurables a nivell d'usuari amb informació generada a partir d'un programari de gestió de pacients en teràpia anticoagulant*, desenvolupat per Josep Itarte a l'empresa Systelab Technologies SA.

A grans trets, el propòsit del projecte és la creació i la integració d'una nova funcionalitat en un programari real i present en mercats internacionals, concretament en el sector del diagnòstic mèdic.

La memòria descriu de manera global tots els aspectes que s'han hagut de tenir en compte per assolir els objectius del projecte. S'explica quin és el context del qual es parteix, quines són les necessitats que han motivat la creació del projecte i quines són les tasques que s'han realitzat al llarg de tot el procés de creació.

1.2 Descripció del projecte

La missió d'aquest apartat és mostrar d'una manera genèrica el context inicial des del qual es parteix i quins són els principals objectius que caldrà haver assolit a la finalització del projecte.

1.2.1 Context empresarial

Systelab Technologies és una empresa catalana que crea i distribueix productes de les tecnologies de la informació. Pertany al grup empresarial anomenat Grup Werfen i es dedica principalment a crear productes per un sector orientat a la medicina. El propòsit d'aquests productes de programari és el de millorar l'atenció als pacients i facilitar la

gestió de la informació per als facultatius en hospitals, clíniques i altres consultoris mèdics.

Seguint una estratègia empresarial, Systelab Technologies ha decidit millorar un dels seus productes que actualment ja es troba en el mercat. Aquest producte, tot i estar al mercat i en funcionament en diversos països, és un projecte de programari en constant desenvolupament i manteniment, amb l'objectiu d'adaptar-lo i integrar-lo a nous sistemes clients.

Aquesta decisió és clau per la creació del PFC, ja que el projecte pertany a tot un seguit de millores que formaran una nova versió del producte.

1. 2. 2 Context inicial

En el moment de començar el projecte final de carrera, l'empresa disposa en el mercat d'un producte de programari destinat a resoldre les necessitats de la gestió de pacients en Teràpia Anticoagulant Oral (TAO). Aquest producte va néixer com a projecte empresarial fa aproximadament 10 anys i, en aquest moment, els clients encara demanen canvis i millores per a noves instal·lacions. Un d'aquests canvis és el de la construcció d'una nova interfície gràfica del programari per fer-la més usable i agradable pels usuaris.

El programa està en funcionament a Itàlia i a diverses comunitats autònomes d'Espanya, amb un gran nombre de pacients actius¹, (actualment sota una teràpia) en augment. El programa l'utilitzen diàriament diversos perfils de persones de l'àmbit de la medicina. Des d'hematòlegs, infermeres, caps de centre mèdic, administradors de sistemes, fins als mateixos pacients que poden consultar les seves dades per Internet.

Actualment, el programa incorpora moltes funcionalitats, des de bàsiques fins a molt complexes, però l'operació principal consisteix en la gestió del procediment que fan els metges als pacients TAO, des de que es presenten a les consultes mèdiques fins que marxen amb una dosificació farmacològica personalitzada.

El procés comença quan un pacient té una cita per una visita al seu centre mèdic, aleshores és cridat a infermeria i es fa una analítica d'INR (Raó Internacional

1 200.000 pacients actius segons dades de desembre de 2012. Font: Systelab Technologies S.A.

Normalitzada) mitjançant un analitzador. S'obté un resultat i a partir d'aquest, si és un pacient estable, el mateix programa suggereix una dosificació i repartició dels medicaments. En cas contrari, quan el pacient no és estable, és el metge qui calcula la dosi. En ambdós casos, els resultats són sempre validats per un metge. Un cop el pacient té la visita validada, se li imprimeix un informe que conté tota la informació necessària per a que pugui seguir amb el seu tractament juntament amb la planificació de les seves properes visites.

Aquesta gestió genera internament una gran quantitat de dades que aporten molta informació. Aquesta es tractarà i es presentarà de forma agradable als usuaris (hematòlegs, metges...) per tal de que aquests observin de primera mà els resultats de la seva gestió dels seus pacients.

1. 2. 3 Necessitats

En el moment inicial, l'empresa té com a principal repte la creació d'una nova interfície gràfica per l'actual sistema. Per aquest motiu es contempla l'opció d'afegir noves funcionalitats al programari actual que permetin dotar-lo amb més eines i utilitats pels usuaris.

La responsabilitat del PFC rau en la creació d'una d'aquestes noves funcionalitats i que a continuació es detalla. Caldrà dotar al sistema d'una nova funcionalitat que permeti als usuaris veure els resultats d'una sèrie d'estadístiques personals en la pantalla principal del programa (dashboard). Aquestes estadístiques explotaran al màxim les dades que es generen a partir de la gestió de pacients (procediment que s'ha descrit en l'anterior apartat). El seguit de processos estadístics s'executaran diàriament per tal que els usuaris tinguin les dades sempre actualitzades. Per tal de no afectar al rendiment del sistema en hores de treball, els processos s'executaran durant la nit. Aquestes estadístiques estaran personalitzades a nivell d'usuari, de centre mèdic i instal·lació. Per tant, per mantenir-ho caldrà emmagatzemar els resultats a les bases de dades del sistema. Amb aquestes dades tractades es mostrarà la informació representada gràficament als usuaris de manera que sigui d'utilitat.

1. 2. 4 Objectius

L'objectiu del projecte final de carrera no és crear un nou producte sinó construir una nova funcionalitat i integrar-la dins d'un producte informàtic actualment existent. Els objectius principals d'aquesta nova funcionalitat són, per una banda, el tractament de les dades que es generen durant el propi procés de gestió i, per altra, mostrar aquesta informació tractada al sistema per que sigui d'utilitat pels usuaris. Si aquests objectius s'assoleixen satisfactòriament, indirectament també es complirà un tercer objectiu que es basa en que els usuaris guanyin confiança amb el programari i també millorin els resultats en la seva feina diària.

De manera més esquemàtica, es podrien enumerar els següents objectius:

- Creació i integració d'una nova funcionalitat en un sistema real
- Transformació de les dades del sistema en informació valuosa pels usuaris
- Creació d'estadístiques per tractar dades del sistema
- Mostra dels resultats de les dades de manera gràfica als usuaris

1. 3 Decisió tecnològica

Com s'ha esmentat en apartats anteriors, el projecte final de carrera forma part d'un paquet de millores sobre un programari existent. Aquest programari, on s'haurà d'integrar el projecte, ha estat en constant manteniment i desenvolupament al llarg de molts anys, la qual cosa implica que s'ha rebut una herència tecnològica que no s'ha pogut canviar ni discutir.

A banda de les restriccions pròpies del programari existent també hi ha unes altres que tenen a veure amb decisions tant a nivell d'empresa com a nivell dels clients.

L'opinió dels clients pot ser tant a nivell de funcionalitat com a nivell de les seves infraestructures que és on acabarà instal·lada l'aplicació.

És per aquestes raons que el desenvolupament del projecte en algun moment haurà de seguir unes pautes marcades per respectar restriccions tant de caràcter tecnològic com dels clients.

En els següents capítols, però en gran mesura en l'apartat d'anàlisi de requeriments, es poden veure el seguit de restriccions que cal respectar.

1. 4 Entorn de treball

Durant tot el transcurs del projecte final de carrera s'ha treballat en un entorn empresarial. Aquest fet fa que l'entorn de treball sigui radicalment diferent al que es podria trobar, per exemple, en un laboratori de càlcul d'investigació. Una empresa té un seguit d'avantatges però també inconvenients.

L'entorn de treball durant tot el projecte final de carrera ha estat en el sí d'un projecte en desenvolupament on paral·lelament hi treballen més persones. Incloent-me a mi mateix, l'equip de desenvolupadors consta de cinc persones, i l'equip de qualitat consta de dues persones. A part, també hi ha participat el cap de projecte, que és el mateix director del PFC. El fet de poder treballar conjuntament amb personal altament qualificat enriqueix molt més les tasques que es fan a la feina i donen més valor a la feina diària, ja que permet aprendre molt més i millor.

Per altra banda, durant els preliminars del PFC es va haver de treballar conjuntament amb els propis clients de l'empresa, per arribar a acords sobre els requeriments que havia de complir l'aplicació. Tot i no ser cap molèstia, aquest fet fa que no es tingui la darrera decisió en alguns aspectes del projecte. Com es diu sovint, el client sempre té la raó.

1. 5 Glossari

En aquest glossari es pot trobar informació d'utilitat per una millor comprensió de la memòria. Es pot veure un llistat amb les definicions d'algunes paraules clau que s'utilitzen al llarg de la memòria.

Terme	Significat
API	<i>Interfície de Programació d'Aplicacions (Application Programming Interface)</i> que conté un conjunt de declaracions que defineix un component informàtic el qual es pot utilitzar. També s'anomena de manera comú com a llibreria de programació.
EJB	<i>Enterprise Java Bean</i> . Especificació d'un component informàtic per implementar les parts que afecten al servidor d'aplicacions distribuïdes.
Java	Llenguatge de programació creat per Sun Microsystems. L'any 2006 es va alliberar Java com a programari lliure de codi obert.
UML	<i>Unified Modeling Language</i> . Llenguatge de modelat que s'utilitza per

	especificar, dissenyar i documentar programari orientat a objectes.
SQL	<i>Structured Query Language</i> . Llenguatge estàndard de comunicació amb les bases de dades. Permet accedir a les dades i realitzar operacions i consultes de manera estàndard.
DAO	<i>Data Access Object</i> . Component del programari que subministra una interfície comuna entre l'aplicació i altres dispositius per tal d'accedir a les dades.
Script	Fitxer que conté una sèrie d'instruccions o comandes per tal de que siguin executats de manera conjunta.
BD (DB)	Sigles que volen dir <i>Base de Dades o Data Base</i> .
J2EE	<i>Java 2 Enterprise Edition</i> . Plataforma de programació per desenvolupar i executar programari escrit en llenguatge Java amb una arquitectura distribuïda en nivells, basada en components de programari i executada en un servidor d'aplicacions.

1. 6 Estructura del document

El present document s'organitza en un total de nou capítols, on cadascun d'ells representa un estat en el cicle del desenvolupament del programari. Tot seguit es descriuen els seus continguts.

Capítol 1. Introducció

En el primer capítol de la memòria es fa una descripció general de tot el que té a veure amb el projecte final de carrera. El propòsit del capítol és definir d'una manera clara quins són els objectius del projecte, quines àrees es treballaran i quin és el context del qual es parteix.

Capítol 2. Estat de l'art

En aquesta secció es descriu més en profunditat quin és l'estat del projecte actualment, des de quin punt es parteix i què es pretén guanyar un cop finalitzat el treball.

Capítol 3. Planificació inicial

La planificació inicial posa sobre la taula una primera aproximació de les tasques que caldrà realitzar en el projecte, en quin ordre seran completades i quins són els terminis estimats per cadascuna d'elles.

També es podrà observar un diagrama on es mostren les tasques en format gràfic.

Capítol 4. Anàlisi de requeriments

En aquest capítol es defineixen els requeriments essencials que haurà de complir el projecte final de carrera. En primer lloc s'expliquen d'una manera global però tot seguit es fa un anàlisi de cadascun d'ells, classificant-los segons el seu tipus: funcional, no funcional i no tècnic.

Capítol 5. Especificació

La secció de l'especificació té com a objectiu definir de manera formal accions que hauran de poder fer els usuaris amb el nou sistema. Per aquest motiu, el capítol inclou una descripció dels actors del sistema, dels casos d'ús on poden participar i el model conceptual de totes les representacions que estan involucrades en el projecte.

Capítol 6. Disseny

El capítol de disseny està molt lligat al capítol 5, ja que bàsicament el que sí pot trobar és una descripció tècnica i detallada de l'aplicació amb l'objectiu d'acomplir l'especificació anterior. A grans trets, en aquesta secció s'analitza l'arquitectura de l'aplicació i els patrons de disseny que hi intervenen.

Capítol 7. Implementació i verificació

La missió d'aquest capítol rau en explicar com s'ha acabat construït la nova funcionalitat, de quina manera s'ha tingut en compte l'anàlisi i el disseny realitzat en els anteriors passos i exemplificar de la manera més gràfica i rigorosa possible quins recursos i quines eines s'han fet servir per assolir l'objectiu.

Capítol 8. Anàlisi de costos

Aquest apartat neix de la necessitat de voler conèixer quins són els costos en projectes d'aquesta envergadura. S'analitzen uns costos inicials estimats, tenint en compte el nombre d'hores previstes, el nombre de recursos, etc, i es compara amb els costos obtinguts un cop acabat el projecte i discutint-ne quines han sigut les desviacions.

Capítol 9. Conclusions

Aquest capítol és de caire personal i és on s'avalua de manera més subjectiva el que ha comportat fer aquest projecte. A part, també es fa una darrera reflexió on s'analitza quina ha sigut la planificació final del projecte, i també es proposen possibles futures ampliacions.

2. Estat de l'art

En el món actual, el negoci basat en el programari, i especialment en sectors crítics i d'alt risc com és el de la salut, és altament complicat i amb una competència ferotge. Actualment hi ha diverses empreses que, com si juguessin una partida d'escacs, intenten guanyar nous clients oferint cada cop un programari amb una major qualitat i més adaptable.

El projecte final de carrera ha de ser capaç d'agregar-se en el nucli d'un programari complex d'una manera senzilla i que, per tant, no impacti negativament sobre el treball que ja s'ha fet. Tot seguit es detalla la informació sobre l'estat actual del *programari mare* d'on ha de néixer el projecte final de carrera.

El projecte Anthema/HytGold² (programari mare) és un producte de programari que va començar fa més de deu anys. Aquest fet és clau ja que genera unes implicacions molt importants, sobretot de caire tecnològic. El projecte, amb el propòsit d'instal·lar-se en el major nombre de clients possible, ha estat en permanent evolució al llarg dels anys. Ara és l'hora, per tant, d'aprofitar i millorar-lo.

En el moment de desenvolupar el projecte final de carrera, el programari mare Anthema/HytGold es troba immers en unes tasques de millora de la interfície gràfica. L'objectiu d'aquestes millores és el de reformar tota l'aparença gràfica i reformular l'organització de la informació a nivell visual. A banda, també s'està treballant en algunes noves funcionalitats. El projecte final de carrera queda inclòs dins aquestes tasques de millora.

És molt important que, un cop finalitzat el projecte final, el projecte Anthema tingui un salt qualitatiu considerable pel que fa a informació mostrada als usuaris. Aquest fet serà clau per diferenciar el producte de les solucions de la competència, i per a que els usuaris i clients tinguin més confiança en el producte.

² Anthema és el nom comercial del programa a Itàlia, i HytGold és el nom que rep a Espanya i Portugal.

Com es veurà en els propers capítols, una de les principals escomeses que ha de tenir el projecte final de carrera és la d'aprofitar informació de la pròpia base de dades d'un sistema (a partir d'un tractament estadístic), per tal de mostrar-la de manera gràfica, útil i elegant.

Per aquest motiu s'introdueix a continuació un petit resum de les funcionalitats que té el sistema del qual es pretén absorbir la informació. D'aquesta manera es podrà detectar i analitzar quina és la informació més valuosa pels usuaris, i tenir-la en compte a l'hora d'implementar el projecte.

El programa Anthema/HytGold basa el seu funcionament en la gestió de la teràpia anticoagulant de pacients. Per aconseguir-ho, el programari és utilitzat per perfils de l'àmbit de la medicina, experts en teràpia anticoagulant. Aquests usuaris s'han d'encarregar de que els pacients tinguin un servei mèdic correcte, que rebin una dosificació farmacològica detallada i que la seva vida normal no es vegi alterada.

A continuació, s'explica quina és la funcionalitat bàsica del programari, que té a veure amb els passos que ha de seguir un pacient des del moment en que es presenta en un centre mèdic per una visita fins que en surt amb la seva dosificació de medicaments.

- Citat
- Esperant resultat
- Esperant dosificació
- Esperant Validació
- Validat
- Imprès

Aquesta funcionalitat és la pedra angular del programari, donat que és on es gestiona la teràpia dels pacients i d'on s'obté informació valuosa pels treballadors del sector.

Tot i això, l'aplicació no només s'encarrega d'aquesta gestió, sinó que també disposa de moltes altres funcionalitats, sobretot en l'àmbit de les comunicacions. Per exemple,

actualment són un client ric i un client web).

Un cop vist en més detall quines dades administra el programari ja es pot tenir una idea més clara de les dades que caldrà tractar, i quina relació tenen amb el programa. En els propers episodis comença el cicle de l'enginyeria del programari, descrivint d'una manera més tècnica cadascuna de les tasques que caldrà resoldre per assolir els objectius.

3. Planificació

Per definir la planificació inicial s'ha tingut en compte en primer lloc la quantitat de tasques que cal realitzar. Aquestes tasques van lligades al cicle de desenvolupament de programari. En segon lloc, s'ha volgut quantificar inicialment el temps que serà necessari destinar per aconseguir aquestes tasques.

3.1 Fases del projecte

En aquest apartat es detallen quines són les fases del projecte que s'han tingut en compte en la planificació inicial. Com es pot veure en la següent imatge, el cicle de desenvolupament que s'ha triat és el cicle en cascada. Aquest cicle conté una sèrie d'estats o passos i que només des d'un estat es pot tornar al seu immediatament anterior o següent.

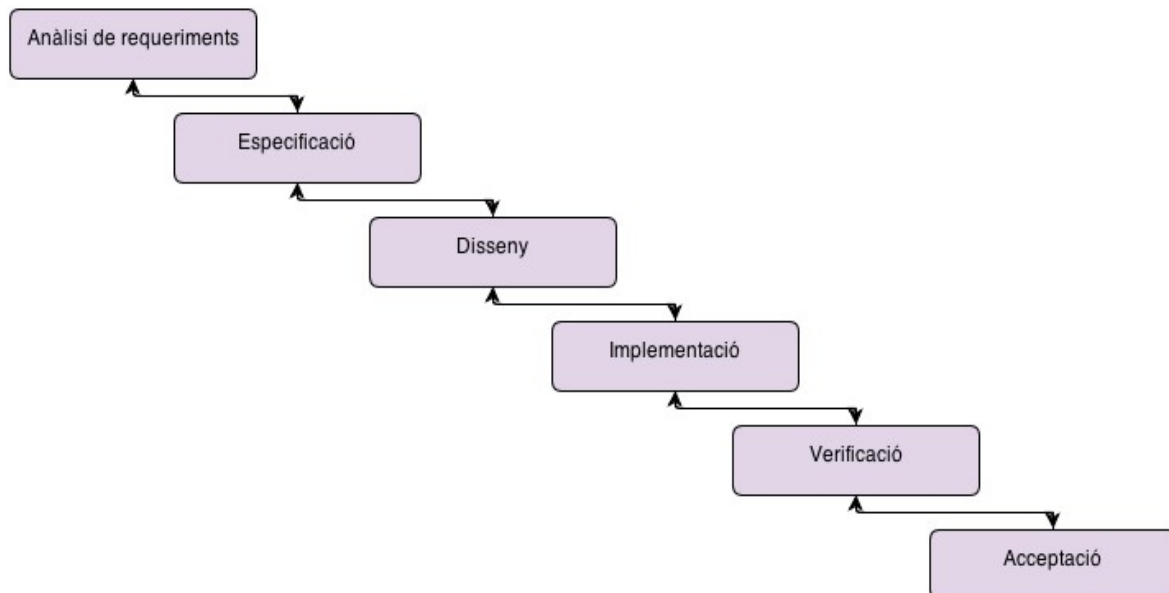


Figura 2: Cicle de desenvolupament en cascada

3. 1. 1 Contextualització i captura de requeriments

En aquesta primera fase del projecte cal tenir clar què es vol fer i fins on cal arribar en termes globals del projecte. El projecte final de carrera no és solament una petició directa de client, si no que també neix com a procés de millora contínua que s'ha decidit portar a terme des de Systemlab Technologies. Aquest fet fa que calgui definir una sèrie de requeriments que cobreixin unes necessitats.

3. 1. 2 Especificació

Un cop definits els requeriments caldrà fer un anàlisi de sistema per especificar el que serà la nova funcionalitat de la manera més detallada possible. Per fer-ho, caldrà generar la documentació necessària.

3. 1. 3 Disseny

El procés del disseny es basarà en el refinament de l'especificació obtinguda en la fase anterior. Caldrà obtenir un disseny complet amb tota la documentació. També caldrà dissenyar l'estructura de la base de dades.

3. 1. 4 Implementació i verificació

És en aquesta fase quan es pot començar a implementar la nova funcionalitat. En aquest punt caldrà conèixer les tecnologies rebudes en herència del projecte i també dominar les possibles noves tecnologies.

Un cop implementat caldrà verificar de manera eficaç la nova funcionalitat. Aquest punt és molt important ja que es podran detectar possibles errors de codificació abans de posar la nova funcionalitat en producció.

3. 1. 5 Depuració de la memòria

Com a darrer pas, caldrà revisar i corregir tots els documents definits fins al moment i agrupar-los en el que serà la memòria final del projecte.

3.2 Planificació inicial

Com s'ha esmentat en l'apartat anterior, el projecte final de carrera es basa en un sistema bàsic de desenvolupament de programari. Per aquest motiu, inicialment es va fer una planificació basada en els grans apartats del cicle de creació de programari (veure Figura 2). A banda d'aquestes tasques genèriques, també es va planificar una tasca que consistia en escriure, de la manera més acurada possible, tota la documentació generada en tots aquests apartats. L'objectiu d'aquesta tasca era arribar al final del projecte amb el mínim de càrrega de treball de documentació per generar la memòria.

En els inicis del projecte, es va analitzar quin podia ser el termini màxim d'entrega del projecte, ja que calia acomplir les necessitats de l'empresa envers els clients. Per aquesta raó es va decidir que la millor manera de dur a terme el projecte era dedicant-li jornades completes de vuit hores per ajustar i apropar al màxim el termini d'entrega.

Com es veu en les següents imatges, en la planificació també s'ha considerat les vacances d'estiu, ja que l'empresa es manté tancada tres setmanes. A banda, també es considera la tasca de "redacció de memòria" com a unió entre les tasques "Memòria" i "Depuració memòria" observables en el diagrama de Gantt.

Tasca	Data inici	Data final	Dies
Definició de requeriments	02/07/12	06/07/12	5
Especificació	09/07/12	20/07/12	10
Disseny	23/07/12	10/08/12	15
Vacances estiu	13/08/12	31/08/12	15
Implementació	03/09/12	28/09/12	20
Verificació	01/10/12	05/10/12	5
Redacció memòria	08/07/12	19/10/12	80

Figura 3: Taula de les tasques amb les estimacions inicials

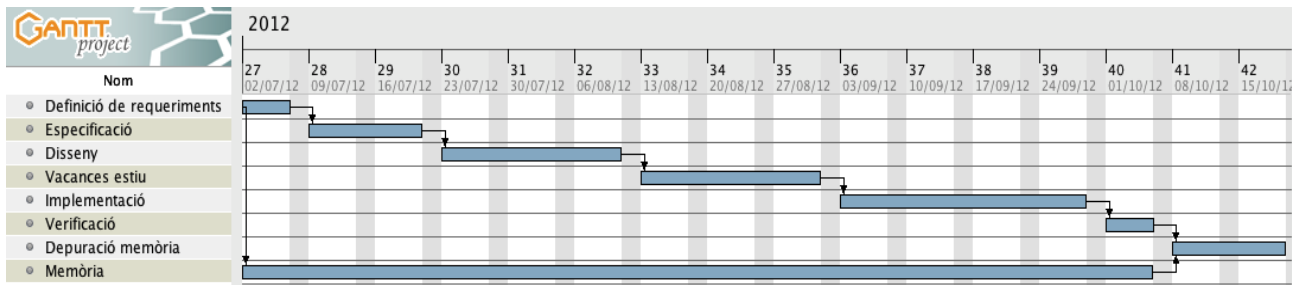


Figura 4: Diagrama de Gantt de la planificació inicial

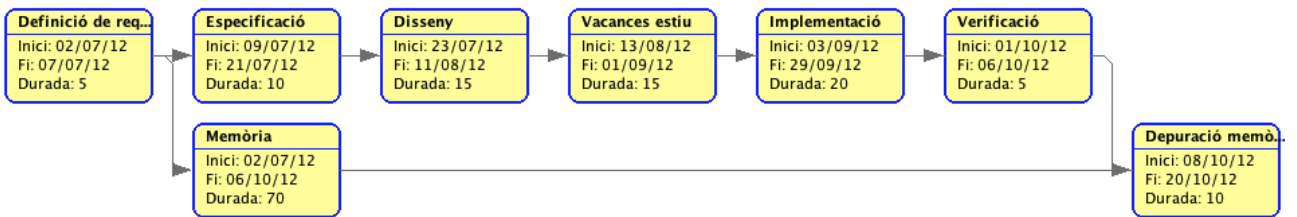


Figura 5: Diagrama PERT de la planificació inicial

4. Anàlisi de requeriments

El següent capítol es basa en la captura i l'anàlisi detallat dels requeriments del projecte final de carrera. L'elaboració d'un bon anàlisi de requeriments serà clau ja que, qualsevol serrell mal definit, pot impactar de manera greu en qualsevol de les següents fases del projecte. Aquest capítol, per tant, és la pedra angular del cicle de desenvolupament. Caldrà doncs deixar ben clars els límits i particularitats del que serà la nova funcionalitat.

Per altra banda, la definició dels requeriments ha de ser el resultat de diverses discussions tant amb els gestors del projecte com dels clients. És molt important que tots els interessats del projecte (stakeholders) estiguin d'acord i sàpiguen de primera mà quins són els objectius i limitacions del projecte final de carrera abans de començar qualsevol tasca d'anàlisi. Una bona definició de requeriments estalviarà que, en tasques de desenvolupament, s'hagi de modificar feina ja feta.

4.1 Característiques generals

En els capítols introductoris de la memòria s'ha volgut emfatitzar que l'objectiu del projecte és el d'aportar valor a un programari. Tot seguit es descriuran les idees amb les que es pretén arribar al propòsit final.

Tant els clients com els caps de projecte de l'empresa necessiten un sistema que aporti informació als usuaris de l'aplicació. Aquesta informació té a veure amb els resultats de la gestió que fan diàriament amb pacients en teràpia anticoagulant. Per donar un petit exemple, un usuari (metge) fa una visita a un pacient. Durant aquesta visita, el pacient es fa una analítica i obté un resultat. Amb aquests resultats, l'usuari (metge) proposa una dosi d'un fàrmac concret per al pacient i finalitza la visita. Per tant, tota aquesta informació queda traçada al sistema i serà necessari treure'n profit.

Doncs, per acomplir l'objectiu del projecte, caldrà que s'aprofiti una sèrie d'informació desada al sistema per, un cop treballada amb unes estadístiques definides, mostrar-la als

usuaris. Tot seguit s'exposen els punts principals:

- Obtenir informació valuosa a nivell d'usuari de les bases de dades de l'aplicació.
- Tractar i depurar la informació executant processos estadístics
- Transformar la informació obtinguda a objectes gràfics fàcilment comprensibles.
- Crear i adaptar un mòdul on l'usuari pugui gestionar com es representa la informació gràfica generada de manera senzilla.
- Cal que el sistema recordi la configuració personal i mostri la informació, tal i com l'usuari desitja.
- Adaptar el sistema actual per afegir-hi el nou mòdul.
- Actualitzar i generar la informació diàriament i sense que impacti en el rendiment de l'aplicació.

En els propers apartats s'especifica amb més detall els requeriments dels punts anteriors.

4. 2 Requeriments funcionals

Tot seguit es descriuen els requeriments funcionals del projecte. Per aclarir la finalitat de cada requeriment s'ha optat per separar-los en els següents punts.

4. 2. 1 Obtenció i tractament de la informació

- **Obtenció de dades**

Cal definir quina és la informació que es vol obtenir de les bases de dades del sistema per després mostrar-la als usuaris. Les dades cal que reflecteixin la tasca diària dels usuaris de l'aplicació. Serà necessari que s'analitzi el model propi de la base de dades del sistema i s'identifiquin les taules on es desa la informació crítica.

- **Tractament i transformació de les dades**

Un cop s'hagi localitzat en el sistema on es desa la informació, caldrà crear diversos procediments estadístics per tal d'exploitar aquests dades.

- **Manteniment de la informació**

Caldrà crear una nova estructura a la base de dades per tal d'emmagatzemar informació sobre quins processos s'han executat i també desar-ne els darrers resultats en part calculats. Cal tenir el control de quins procediments s'executen, en quin moment i quin resultat han generat.

- **Actualització de la generació de la informació**

El sistema haurà de ser capaç, de manera automàtica, de refrescar i actualitzar les dades obtingudes a partir dels processos estadístics. Caldrà definir una tasca periòdica que executi diàriament les estadístiques per cada usuari. Aquest procés cal que no es produeixi en horari laboral, ja que no ha d'impactar en el temps de resposta de l'aplicació.

4. 2. 2 Representació de la informació

Caldrà crear una interfície d'usuari que es mostri cada cop que l'usuari entri a l'aplicació. Aquesta interfície, que funcionarà com un quadre de comandaments, ha de possibilitar a l'usuari gestionar-ne el contingut i la distribució dels continguts.

La interfície d'usuari estarà recollida en una pantalla principal, que oferirà els continguts calculats pel sistema en funció de l'usuari. Per tant, el contingut podrà diferir en funció de les dades pròpies de l'usuari.

L'usuari podrà escollir quins gràfics mostrar a partir d'un llistat de possibles estadístiques. L'usuari també haurà de poder moure els gràfics, eliminar-los o redimensionar-los.

També caldrà crear una sèrie de gràfics de diferents tipus i que siguin capaços de representar diferents tipus d'estadístiques.

A banda de les característiques anteriors, el sistema ha de ser capaç de representar la informació tal i com l'usuari ha configurat per darrer cop. És a dir, els usuaris poden modificar la distribució de la informació segons el seu parer. El sistema ha de mostrar la informació tal i com els usuaris volen, desant les dades de configuració al sistema.

4. 3 Requeriments no funcionals

En aquest punt es detallen els requeriments no funcionals del sistema. Aquests requeriments són els que el projecte haurà de complir per garantir els aspectes que no tenen a veure amb la funcionalitat. Són necessaris ja que, tot i aconseguir una bona funcionalitat, pot quedar inutilitzable si el sistema no és usable, és insegur o és totalment ineficient. Tot seguit se n'especifiquen els més importants.

4. 3. 1 Eficiència

Cal que la resposta per part del sistema a les peticions de l'usuari sigui ràpida i fiable. El fet d'implementar la nova funcionalitat ha de garantir que, tant la nova eina com les funcionalitats que ja formen part del programa, funcionin de manera eficient i que el rendiment no es vegi penalitzat.

4. 3. 2 Usabilitat i interfície d'usuari

És necessari que la interfície d'usuari sigui clara, intuïtiva i fàcil d'utilitzar. Cal que el temps d'aprenentatge per part de l'usuari amb la nova funcionalitat sigui el més curt possible.

4. 3. 3 Seguretat

És clau que les dades que es mostrin a l'usuari siguin correctes i que no impliquin cap risc per a la gestió dels pacients. Apart també caldrà seguir una sèrie de directrius d'empresa com el compliment de lleis sobre privacitat de dades personals.

4. 4 Requeriments de no tècnics

Un cop analitzats els requisits funcionals i els no funcionals cal tenir en compte els requeriments no tècnics. La majoria d'aquests requisits venen imposats per diversos motius, un dels més importants és la pròpia arquitectura de l'aplicació on s'ha d'integrar la nova funcionalitat. Altres requisits són dictaminats per la pròpia empresa i els seus clients.

i) Tecnologies de programari

Tal i com ja s'ha detallat en anteriors capítols, la funcionalitat en que es basa el projecte final de carrera ha de poder construir-se i integrar-se en el sí d'un programari existent (Anthem/HytGold). Aquest fet fa que calgui seguir una sèrie de directrius i s'hagin de seguir una sèrie de mecanismes per poder garantir la perfecta integració amb el programari.

Els requisits a nivell de programari són els següents:

- La nova funcionalitat ha d'integrar-se en una aplicació basada en l'estàndard Java EE i utilitza com a tecnologia de negoci Enterprise Java Beans 2.1 (EJB 2.1).
- Com a motor de persistència cal adaptar-se a un marc de treball propietat de l'empresa (anomenat Framework DAO) i serà necessari utilitzar com a gestor de bases de dades Oracle Database.
- La part gràfica de l'aplicació ha d'implementar-se utilitzant la API Swing de Java versió 1.6 i per crear gràfiques caldrà utilitzar una llibreria comercial de programari Java anomenada Monarch Graphs.
- A banda d'aquests requisits, és també necessari que el programari sigui multi-idioma i que utilitzi el mètode d'internacionalització i18n³.
- Per últim cal deixar constància que l'aplicació ha de ser capaç de funcionar amb els servidors d'aplicacions d'Oracle OC4J i també Jboss.

De com s'acompleixen aquests requisits se'n parlarà més extensament en els propers capítols, on es farà una descripció més tècnica dels conceptes enunciats anteriorment.

ii) Tecnologies de maquinari

El principal requisit a nivell de maquinari és que l'aplicació no sigui excessivament pesada ja que cal que es pugui instal·lar en equips informàtics clients que no són molt potents.

iii) Polítiques d'empresa

En el desenvolupament del projecte serà necessari seguir una sèrie polítiques d'empresa. Entre les polítiques, les més destacables són:

- Política de privacitat: acatar d'una manera ferma la llei de protecció de dades, ja

3 Internacionalització i18n. Criteri estàndard d'internacionalització d'aplicacions de programari.

que el sistema treballa amb dades privades i confidencials. La LOPD és d'obligat compliment.

- Com a segona política i lligada en gran mesura amb l'anterior, cal desnaturalitzar la base de dades en el moment de desenvolupar. És a dir, durant el procés d'implementar l'aplicació i fer proves, serà necessari utilitzar una base de dades correcta i real. Com que aquestes dades són privades és obligatori manipular la base de dades desnaturalitzant-ne les dades privades utilitzant uns processos d'enciptació.

5. Especificació

En aquest apartat es definirà el comportament del sistema a implementar. Per a fer-ho, caldrà en primer lloc tenir en compte els requeriments obtinguts en la l'apartat anterior. Tenint present aquests requeriments, caldrà aleshores definir quins són els actors que interaccionen amb el sistema, quins casos d'ús poden realitzar i en quins escenaris.

Per tant, tot seguit es podrà veure un conjunt de documentació tècnica que defineix els actors del sistema, els casos d'ús i la seva descripció. A part, també es mostra el model de dades necessari per acomplir amb els requeriments del sistema i els diagrames de seqüència.

5. 1 Actors del sistema

Com s'ha dit anteriorment, el projecte final de carrera ha de construir una nova funcionalitat que s'integri en un sistema ja creat. Aquest sistema és una aplicació on ja hi ha una distinció de rols entre els usuaris que hi poden accedir. Per aquest motiu, és necessari definir quins actors s'adapten amb cada perfil d'usuari del sistema.

En l'aplicació hi poden entrar una sèrie d'usuaris que tenen unes limitacions segons una sèrie de rols que tenen assignats. Utilitzarem aquest rol per definir els actors que poden interactuar amb la nova funcionalitat. A part dels usuaris que poden participar en l'aplicació també cal tenir en compte que hi ha un altre tipus d'usuari que s'encarregarà de preparar les estadístiques que s'executaran diàriament i també programarà la seva execució en els moments requerits. Com que seria molt poc viable que un usuari s'encarregués de fer aquestes tasques s'utilitzarà una eina que permetrà programar el propi gestor de bases de dades per a que executi les estadístiques a unes determinades hores.

Concretament, es tindran en compte dos tipus d'usuari ben diferenciats. Els usuaris de tipus client, que són els usuaris que poden entrar a l'aplicació i que poden actuar amb les

estadístiques amb el nou mòdul. I els usuaris gestors de les estadístiques, que són els encarregats de preparar les estadístiques, la programació d'execució diària i també preparar el sistema, sobretot la base de dades, per poder oferir als usuaris client unes estadístiques calculades. Tot seguit es pot observar un diagrama on es mostra l'arquitectura dels usuaris:

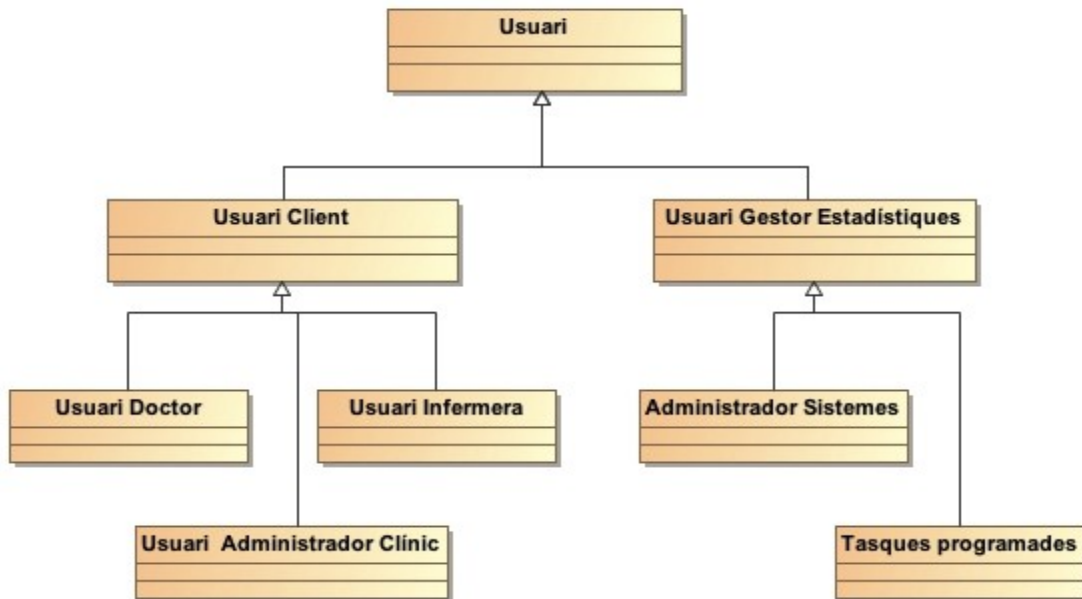


Figura 6: Diagrama mostrant les diferències entre tipus d'usuari

5. 2 Diagrama de casos d'ús

En aquest apartat es poden observar els casos d'ús definits per cada tipus d'usuari del sistema. Cal tenir en compte, com s'ha dit anteriorment, que en el sistema existeixen dos tipus d'usuari ben diferents.

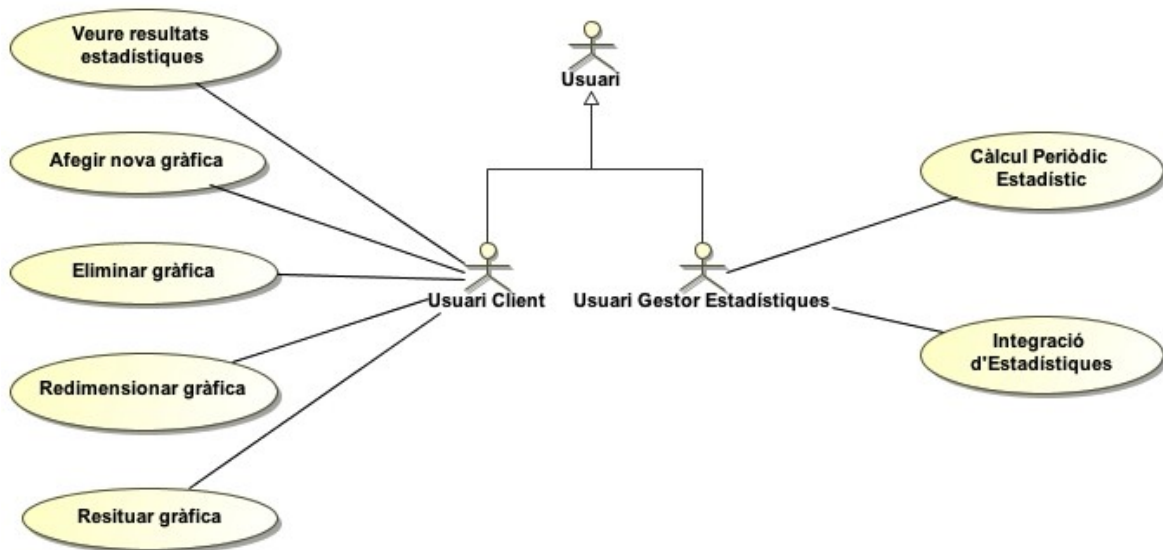


Figura 7: Diagrama amb els nous casos d'ús

La figura anterior mostra un diagrama de casos d'ús on s'exposen les funcionalitats que tenen a la seva disposició els dos tipus d'usuari.

Entrant més en detall, a continuació es mostra el diagrama de casos d'ús pels usuaris de tipus Client, que són els que poden accedir a l'aplicació.

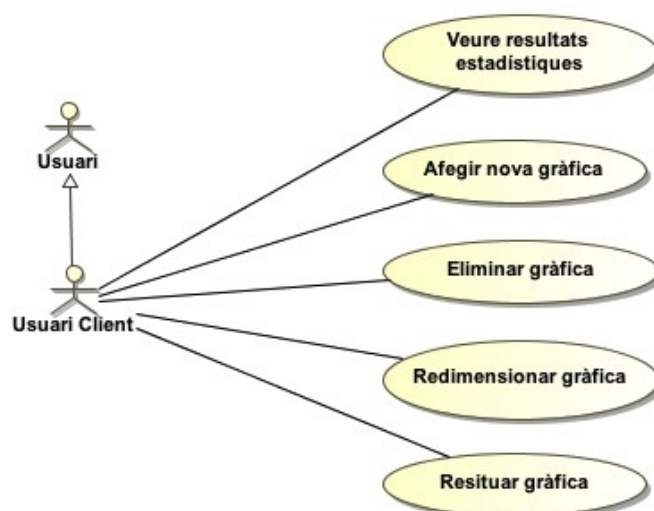


Figura 8: Diagrama de casos d'ús pels usuaris de tipus "Client"

Com es veu en la figura, la funcionalitat pròpia dels usuaris en el nou mòdul es basa en la gestió i representació de les gràfiques generades. L'usuari de tipus Client pot afegir noves estadístiques al mòdul, redimensionar-les, situar-les en les posicions que vulgui, etc.

Per altra banda, els usuaris que s'encarreguen de preparar el sistema per les execucions de les estadístiques periòdiques disposen dels següents casos d'ús.

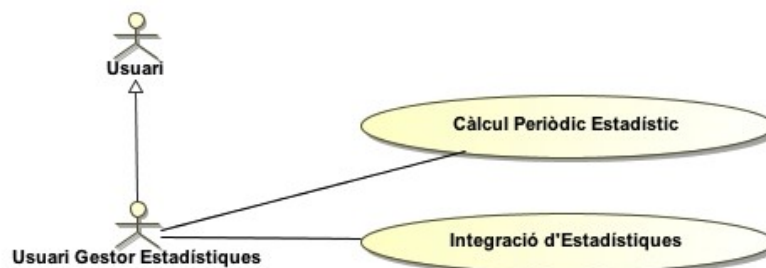


Figura 9: Diagrama de casos d'ús pels usuaris "Gestors d'estadístiques"

Aquests usuaris disposen d'un nombre reduït de casos d'ús ja que solament han de preparar el sistema per que s'executin els processos periòdicament. Bàsicament, han d'incloure les estadístiques al sistema i preparar-lo per que s'executin i es calculin cada cert període de temps.

5. 3 Especificació de casos d'ús

Tot seguit s'enumeren i s'especifiquen els casos d'ús descrits anteriorment. En primer lloc s'especificaran els casos d'ús referents als usuaris de tipus Client i a continuació als usuaris de tipus Gestors d'Estadístiques.

5.3.1 Casos d'ús dels usuaris de tipus Client

5.3.1.1 Veure resultats estadístics

Cas d'ús: Veure resultats estadístics.

Actors: Usuari genèric

Descripció: L'usuari accedeix al mòdul de l'aplicació on es mostren els resums gràfics de les seves aportacions al sistema.

Curs dels esdeveniments:

Actor	Resposta del sistema
1. L'usuari accedeix l'aplicació i automàticament accedeix a la pàgina de visualització de les estadístiques d'usuari.	
	2. El sistema mostra una sèrie de gràfiques que es corresponen amb l'usuari.

Curs alternatiu:

2. El sistema no troba cap resultat estadístic calculat en el sistema i no mostra cap gràfica.

Diagrama de seqüència:

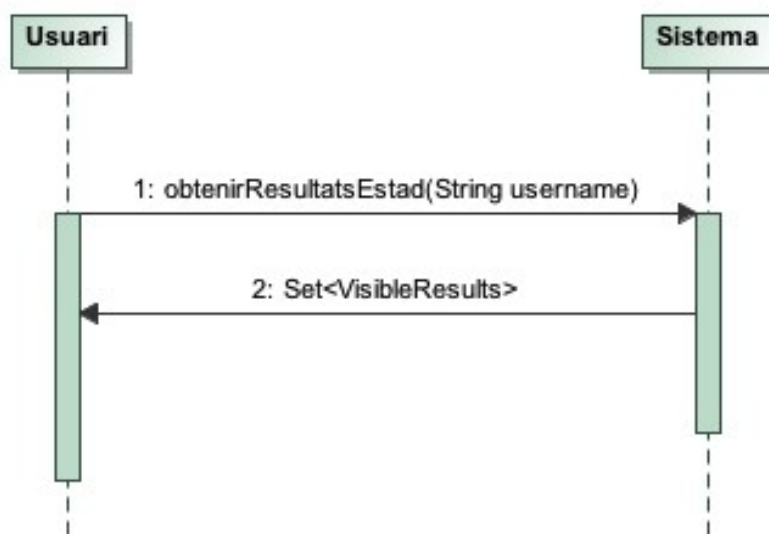


Figura 10: Contracte "Veure resultats estadístics"

5. 3. 1. 2 Afegir gràfica al mòdul d'estadístiques

Cas d'ús: Afegir gràfica al mòdul d'estadístiques.

Actors: Usuari genèric

Descripció: L'usuari vol afegir un altre gràfica amb dades basades en la seva gestió en el mòdul d'estadístiques.

Curs dels esdeveniments:

Actor	Resposta del sistema
1. L'usuari selecciona l'opció d'afegir un resultat gràfic al mòdul d'estadístiques.	
	2. El sistema mostra un llistat on apareixen una sèrie de possibles gràfiques que l'usuari pot incorporar al seu mòdul.
3. L'usuari selecciona una gràfica.	
	4. El sistema incorpora la nova gràfica seleccionada al mòdul de l'usuari.

Curs alternatiu:

3a . L'usuari selecciona més d'una gràfica del llistat de possibles gràfiques a escollir.

4a. El sistema mostra un missatge dient que només es pot seleccionar una gràfica.

3b. L'usuari no selecciona cap gràfica del llistat i tanca el llistat.

4b. El sistema no incorpora cap nova gràfica al mòdul de l'usuari.

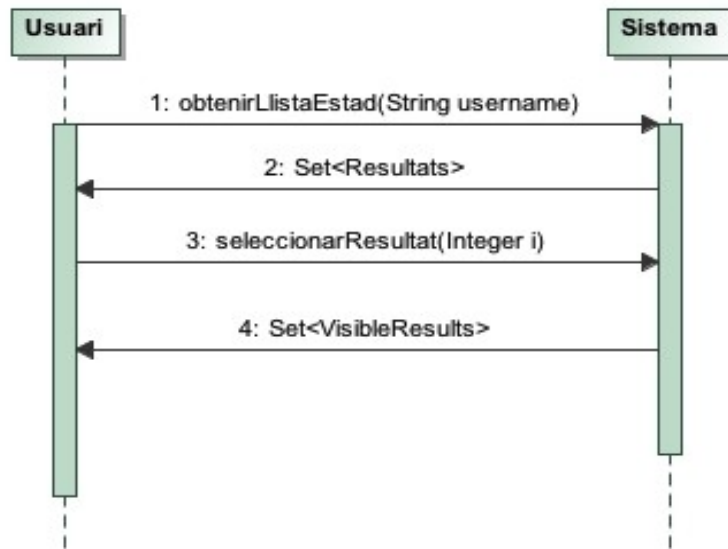
Diagrama de seqüència:

Figura 11: Contracte "Afegir nova gràfica"

5. 3. 1. 3 Eliminar gràfica del mòdul d'estadístiques

Cas d'ús: Eliminar gràfica del mòdul d'estadístiques.

Actors: Usuari genèric

Descripció: L'usuari vol eliminar una gràfica del seu mòdul personal d'estadístiques.

Curs dels esdeveniments:

Actor	Resposta del sistema
1. L'usuari selecciona l'opció d'editar estadístiques	
	2. El sistema mostra les opcions que pot fer l'usuari amb la gràfica
3. L'usuari selecciona l'opció d'eliminar.	
	4. El sistema esborra del mòdul de l'usuari la gràfica que aquest ha seleccionat.

Curs alternatiu:

3a. L'usuari decideix no esborrar cap estadística i no selecciona l'opció d'esborrar.

4a. El sistema no elimina cap estadística i torna a l'estat inicial.

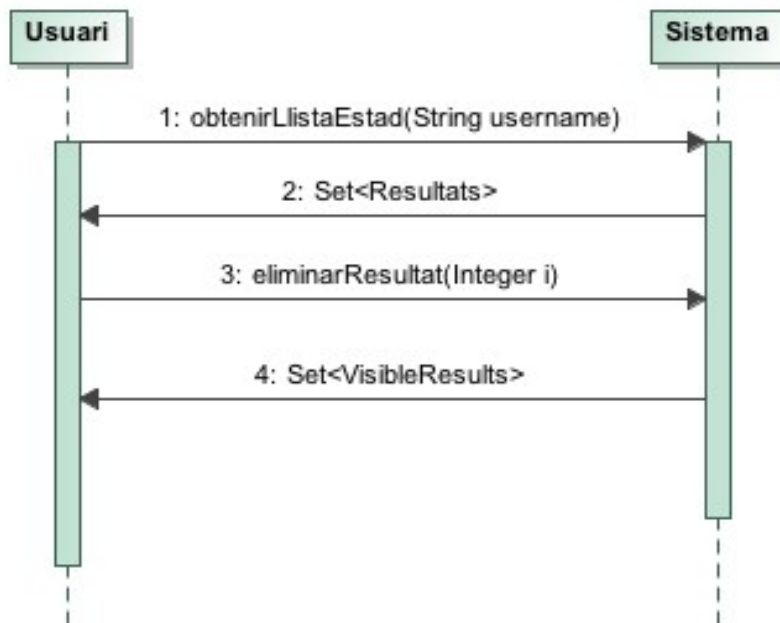
Diagrama de seqüència:

Figura 12: Contracte "Eliminar gràfica del mòdul"

5. 3. 1. 4 Redimensionar gràfica del mòdul d'estadístiques

Cas d'ús: Redimensionar gràfica del mòdul d'estadístiques.

Actors: Usuari genèric

Descripció: L'usuari vol redimensionar una gràfica del seu mòdul personal d'estadístiques.

Curs dels esdeveniments:

Actor	Resposta del sistema
1. L'usuari selecciona l'opció d'editar estadístiques	
	2. El sistema mostra les opcions que pot fer l'usuari amb la gràfica
3. L'usuari selecciona l'opció redimensionar per canviar la mida de la gràfica.	
	4. El sistema canvia la mida a la gràfica.

Curs alternatiu:

3a. L'usuari decideix no modificar la gràfica.

4a. El sistema no canvia res i torna a l'estat inicial.

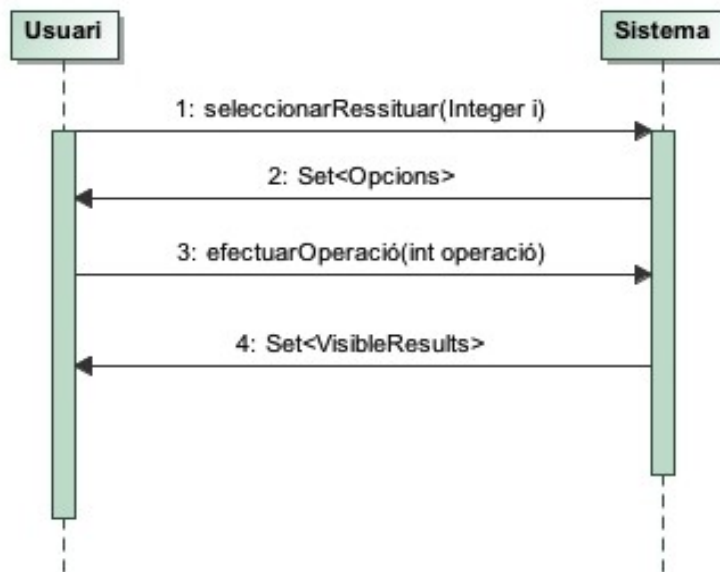
Diagrama de seqüència:

Figura 13: Contracte "Redimensionar gràfica"

5. 3. 1. 5 Ressituar gràfica del mòdul d'estadístiques

Cas d'ús: Ressituar gràfica dins mòdul d'estadístiques.

Actors: Usuari genèric

Descripció: L'usuari vol canviar de lloc una gràfica del seu mòdul personal d'estadístiques.

Curs dels esdeveniments:

Actor	Resposta del sistema
1. L'usuari selecciona l'opció d'editar estadístiques.	
	2. El sistema mostra les opcions que pot fer l'usuari amb la gràfica.
3. L'usuari selecciona l'opció de moure la gràfica.	
	4. El prepara el mòdul per que l'usuari pugui situar la gràfica en un altre emplaçament
5. L'usuari canvia de lloc la gràfica.	
	6. El sistema mostra la gràfica en la nova ubicació.

Curs alternatiu:

3a. L'usuari decideix no moure la gràfica.

4a. El sistema no canvia res i torna a l'estat inicial.

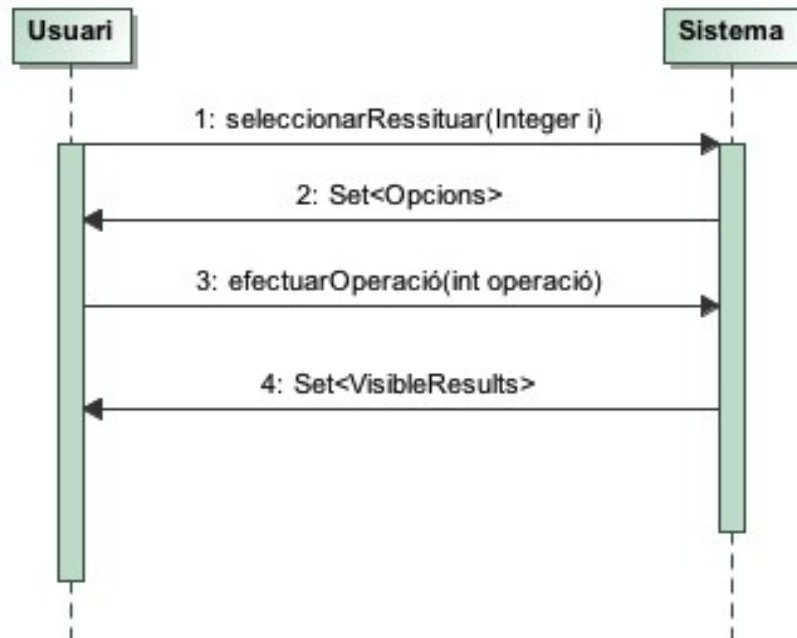
Diagrama de seqüència:

Figura 14: Contracte "Ressituar gràfica al mòdul"

5. 3. 2 Casos d'ús dels usuaris tipus Gestor d'Estadístiques

5. 3. 2. 1 Càlcul periòdic d'estadístiques

Cas d'ús: Càlcul periòdic d'estadístiques.

Actors: Programari automatitzat

Descripció: Un programari que permeti programar tasques haurà d'executar una sèrie de procediments estadístics periòdicament.

Curs dels esdeveniments:

Actor	Resposta del sistema
1. El programari llença processos estadístics.	
	2. El sistema recull les dades i les emmagatzema internament.

Diagrama de seqüència:

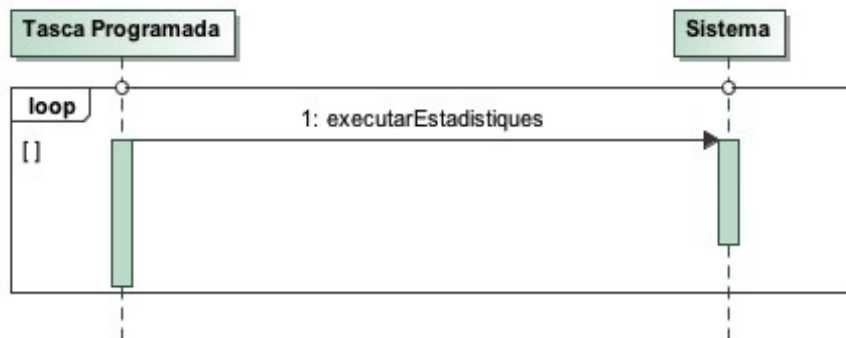


Figura 15: Contracte "Calcul estadístic periòdic"

5. 3. 2. 2 Integració d'estadístiques

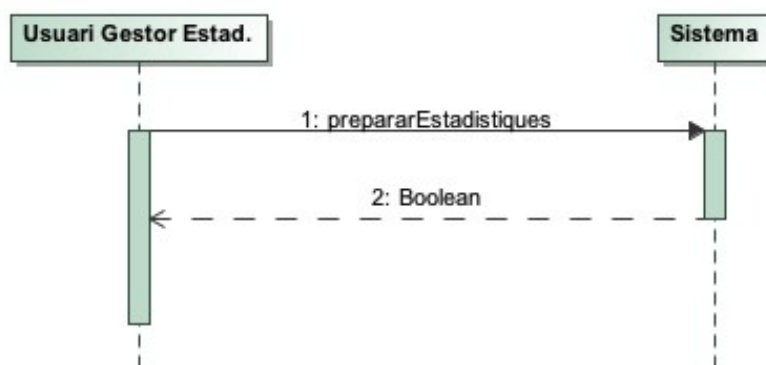
Cas d'ús: Integració d'estadístiques.

Actors: Usuari Gestor d'estadístiques

Descripció: L'usuari ha de ser capaç d'integrar en les bases de dades del sistema una sèrie d'estadístiques, mitjançant uns procediments automàtics, dissenyades tenint en compte als usuaris de tipus Client

Curs dels esdeveniments:

Actor	Resposta del sistema
1. L'usuari llença uns processos per incorporar noves estadístiques a les bases de dades del sistema.	
	2. El sistema emmagatzema les estadístiques que el procediment periòdic haurà d'executar.

Diagrama de seqüència:*Figura 16: Contracte "Incorporar estadístiques"*

5. 4 Model conceptual

Tot seguit s'introduirà el nou model de dades que servirà per complir els requeriments i els casos d'ús definits anteriorment.

Per poder acomplir les noves necessitats s'ha intentat especificar un model de dades que sigui capaç d'integrar-se en el sistema ja existent sense que aquest tingui cap impacte negatiu. Per a fer-ho, s'han tingut en compte les dues principals necessitats:

- Manteniment dels resultats dels processos estadístics
- Manteniment de la configuració d'usuari de representació de gràfiques

Tot seguit s'il·lustren els nous elements UML que caldrà incorporar en el sistema:

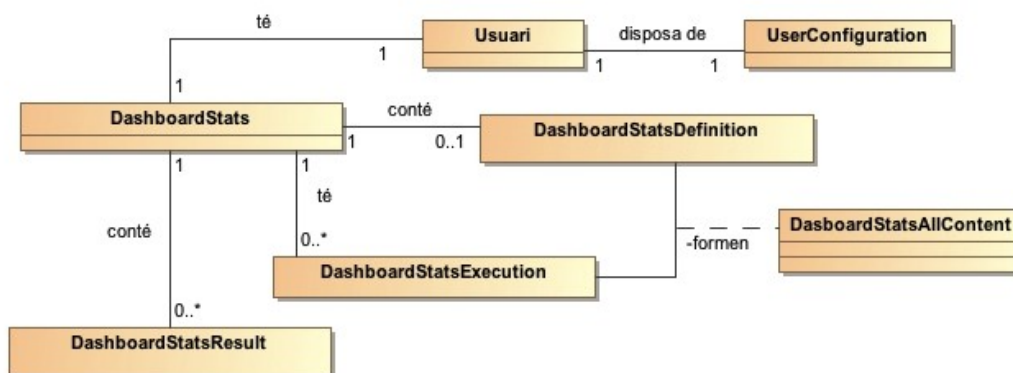


Figura 17: Nous elements UML al model de dades

Com es veu en l'anterior diagrama UML, s'han incorporat uns nous elements que permetran, per una banda, mantenir i obtenir els resultats de les estadístiques a nivell d'usuari, com també gestionar-ne la configuració pròpia d'usuari.

Per simplificar el model, s'ha optat per separar en diversos elements els resultats de les estadístiques de les seves definicions com també de les seves execucions.

Tot seguit es mostra el model amb els atributs i mètodes necessaris:

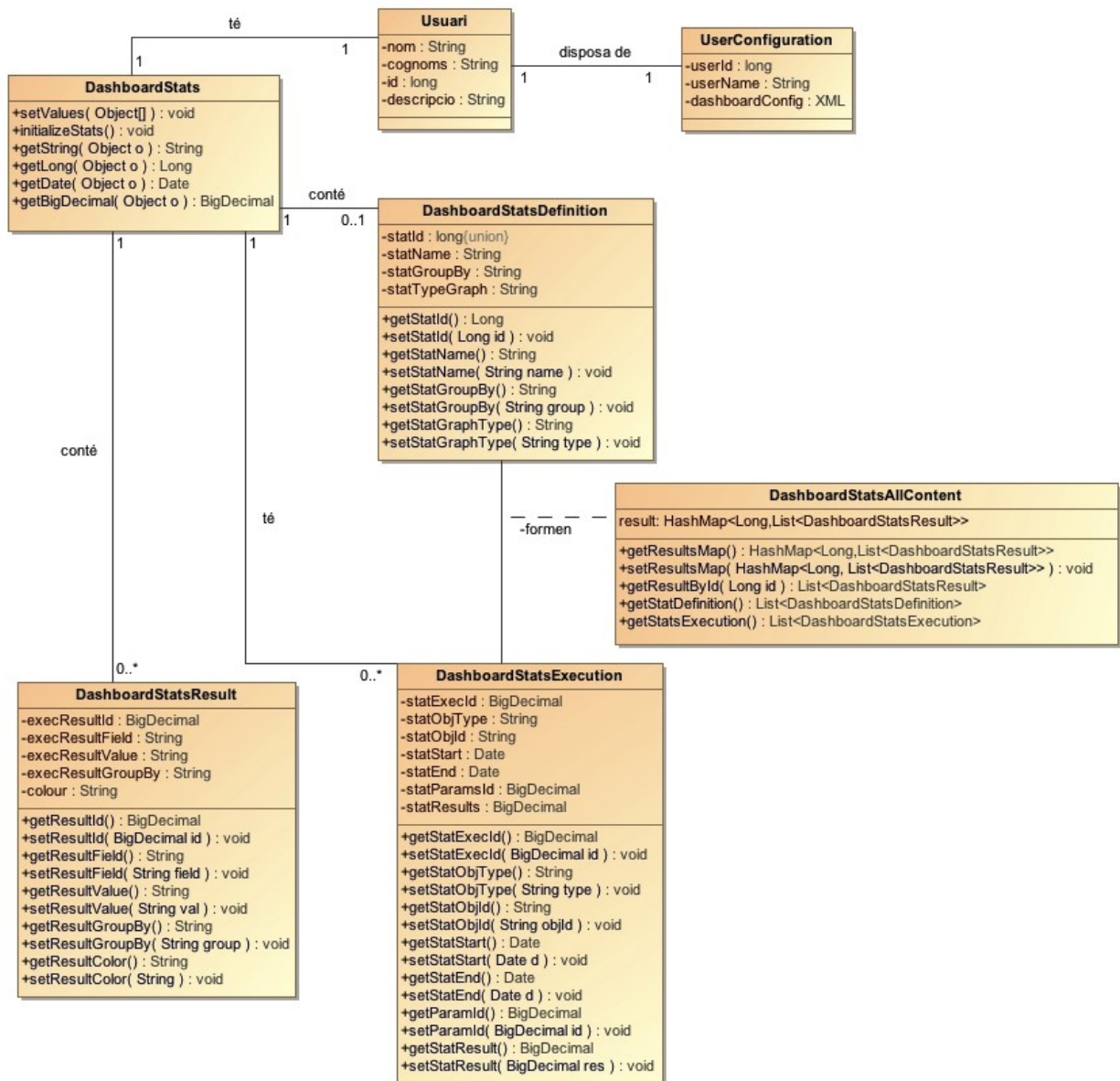


Figura 18: Nou model de dades complet

6. Disseny

En aquest apartat es detalla l'arquitectura de l'aplicació on caldrà integrar el nou mòdul i els patrons de disseny que es fan servir. Al tractar-se d'una aplicació de gran envergadura, s'intentarà esquematitzar al màxim les explicacions de les parts més importats d'aquesta.

6. 1 Arquitectura de l'aplicació

L'aplicació on es construirà la nova funcionalitat es basa en una arquitectura distribuïda que ofereix els seus serveis a través d'Internet o d'una xarxa local. Tot i les avantatges de tenir una aplicació distribuïda pel que fa a balanceig de càrrega, escalabilitat dels recursos, etc, sovint l'aplicació s'instal·la en un mateix servidor que concentra la majoria de les funcionalitats.

Com a base sòlida de l'aplicació s'utilitza una plataforma Java per a garantir una arquitectura distribuïda. En concret, es fa servir la plataforma Java EE (Java Enterprise Edition) que es basa en components de programari i que ofereix un gran volum d'avantatges.

El fet d'utilitzar aquest model fa que calgui seguir uns patrons de disseny per adaptar les necessitats pròpies a la plataforma d'arquitectura distribuïda. El patró més important i que impacta de manera clara amb la futura implementació és el patró de les tres capes, que separa en tres nivells ben diferenciats.

En concret, l'aplicació consta de les capes: de Presentació, de Negoci i de Persistència. Al llarg d'aquest capítol s'analitzaran en profunditat.

6. 2 Afectació dels requeriments en el disseny

Prèviament a descriure les capes en que es divideix el disseny de l'aplicació cal també analitzar de quina manera afecten els requeriments, la major part imposats per l'herència del projecte, al disseny que s'està descrivint en aquests moments.

Un dels requeriments demanats pels clients és que necessiten un mètode d'accés eficient

a l'aplicació. Això es refereix a que sovint l'aplicació s'instal·la en centres hospitalaris on els equips informàtics són molt limitats o tenen restringits molts recursos. Per altra banda, aquest programa també s'instal·la en uns equips més potents, normalment de doctors, que necessiten molta potència perquè han de fer servir moltes funcionalitats. Aquest fet fa que l'aplicació disposi de dos clients amb els quals accedir al servidor d'aplicacions. Un d'ells és una aplicació web a la qual s'accedeix per mitjà dels navegadors, orientat a equips amb poca potència ja que es deriva la major part del càlcul al servidor. El segon client és un client ric d'escriptori i que ofereix més funcionalitats que no pas el client de navegador. Amb aquest client es necessita un major nombre de recursos en els equips clients per garantir el bon funcionament del programari.

Com a efecte col·lateral del que s'acaba d'explicar, com que un dels accessos és una aplicació d'escriptori, és necessari un procediment per a que el programa client s'actualitzi periòdicament amb les millores i canvis de les versions més noves. Aquest fet fa que es disposi d'un servei basat en un navegador que permet als usuaris connectar-se al servidor d'aplicacions i descarregar la versió més nova del programa. Això és un fet que cal tenir en compte en el disseny de l'aplicació.

Com a darrera implicació dels requeriments, cal esmentar la necessitat que l'aplicació sigui multi-idioma. Per això cal que la nova funcionalitat estigui disponible en els idiomes necessaris pels clients. Concretament, cal que estigui traduït a l'anglès, a l'espanyol, a l'italià i al portuguès.

6.3 Disseny de les capes

Un cop descrits els requisits que poden impactar de manera més agressiva al disseny, es planteja tot seguit la descripció detallada de l'arquitectura en tres capes de l'aplicació.

El patró de disseny en tres capes té com a principal objectiu fer que qualsevol canvi sigui senzill i que no afecti a totes les parts del disseny sino només en algunes. Això vol dir que facilita les propietats d'extensibilitat, portabilitat, mantenibilitat i reestructurabilitat.

Com s'ha dit anteriorment, els components de programari es divideixen en tres parts, i aquests components només es poden comunicar amb els elements o bé de la mateixa capa o amb els elements de les capes contigües.

A continuació es detallen els objectius de cadascuna de les capes.

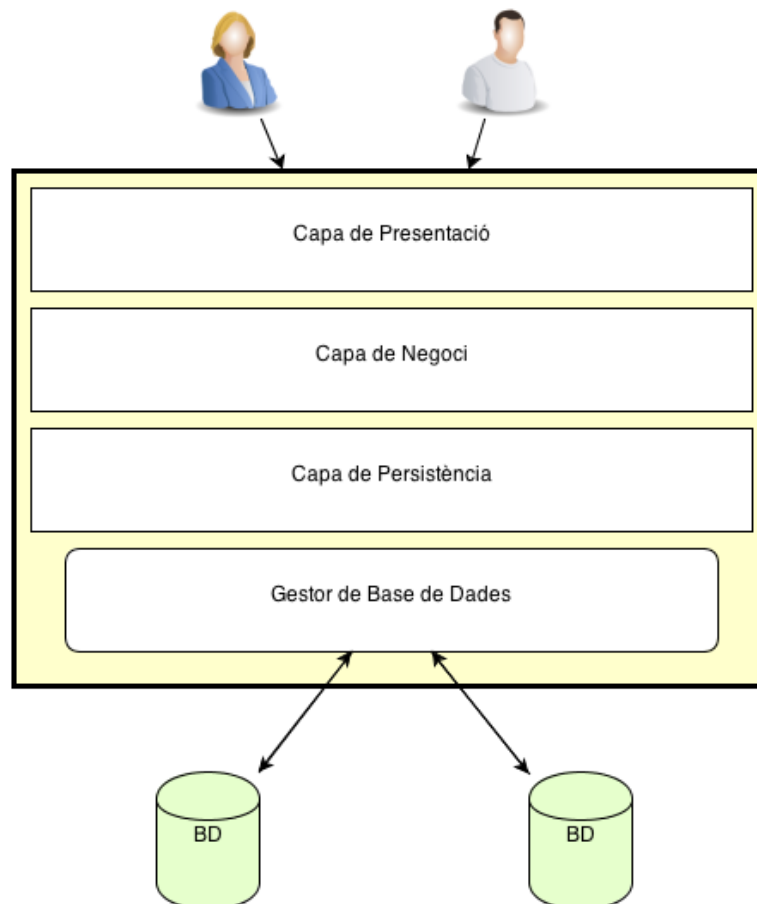


Figura 19: Disseny arquitectura en tres capes

6.3.1 Capa de Presentació

La capa de Presentació sap com haurà de representar les dades a l'usuari però desconeix quins procediments cal efectuar per tal de donar les respostes pertinents als usuaris. Les relacions d'aquesta capa són les següents:

- **Relació amb els usuaris:** permet rebre les peticions dels usuaris. Per una banda, necessita saber quines dades ha de mostrar i també quines opcions selecciona l'usuari. Per altra banda, també necessita saber com ha de mostrar les dades. Per exemple, ha de permetre a l'usuari seleccionar un nou tipus de gràfica i també ha

de poder representar-la correctament.

- **Relació amb la capa de Negoci:** ha de comunicar a aquesta capa quines han sigut les peticions de l'usuari i també ha de rebre'n les respostes per poder representar-les.

6. 3. 1. 1 Marcs de treball utilitzats

Per tal de resoldre les necessitats de la capa de presentació s'han utilitzat diverses eines que han servit per simplificar la feina de desenvolupament. Es poden resumir en els punts següents:

- **Java Swing:** consisteix en una sèrie d'eines Java que proporcionen elements ja desenvolupats per tal de crear interfícies gràfiques d'usuari. Per simplificar, Java Swing és una biblioteca gràfica per a Java, que incorpora diferents elements que els desenvolupadors poden reutilitzar. Aquest elements són els que clàssicament es poden trobar en qualsevol interfície d'usuari, com botons, camps de text, taules, menús desplegable, etc.
- **MonarchCharts:** Monarch Charts és també una llibreria de components (com Java Swing) però creada i distribuïda per l'empresa Singleton Labs. Els components dels que disposa es basen en representacions gràfiques de tota mena, permetent afegir en programari Java gràfiques complexes però d'una manera més senzilla que si s'haguessin de desenvolupar des de zero.

6. 3. 1. 2 Patrons de disseny

- **Patró Façana:** En la capa de Presentació s'aplica el patró de disseny Façana per garantir que totes les peticions d'aquesta capa que tenen com a destí a la capa de Negoci responen a una funcionalitat comuna. Per exemple, la nova funcionalitat necessitarà efectuar una sèrie de peticions sobre les dades de les estadístiques. D'aquesta manera, totes les peticions que engloba aquesta funcionalitat passaran per un mateix punt.

- **Patró MVC** (als components de formulari): s'han dissenyat els components Java per tal de que puguin disposar d'una validació pròpia sense haver d'accedir a la capa de Negoci. Aquest fet és rellevant ja que la capa de Presentació ja efectua una primera validació i parseig de la informació de les dades.
- **Internacionalització**:: responent també a un requeriment, aquesta capa haurà de ser disponible en quatre idiomes: Castellà, Anglès, Portuguès i Italià. S'ha dissenyat, per tant, una funcionalitat que, en funció de l'usuari i la seva pròpia configuració, és capaç de disposar de l'aplicació en l'idioma que desitgi.

6. 3. 1. 3 Diagrama del patró Façana

Tot seguit es pot veure una imatge que mostra el patró façana aplicat en la capa de presentació.

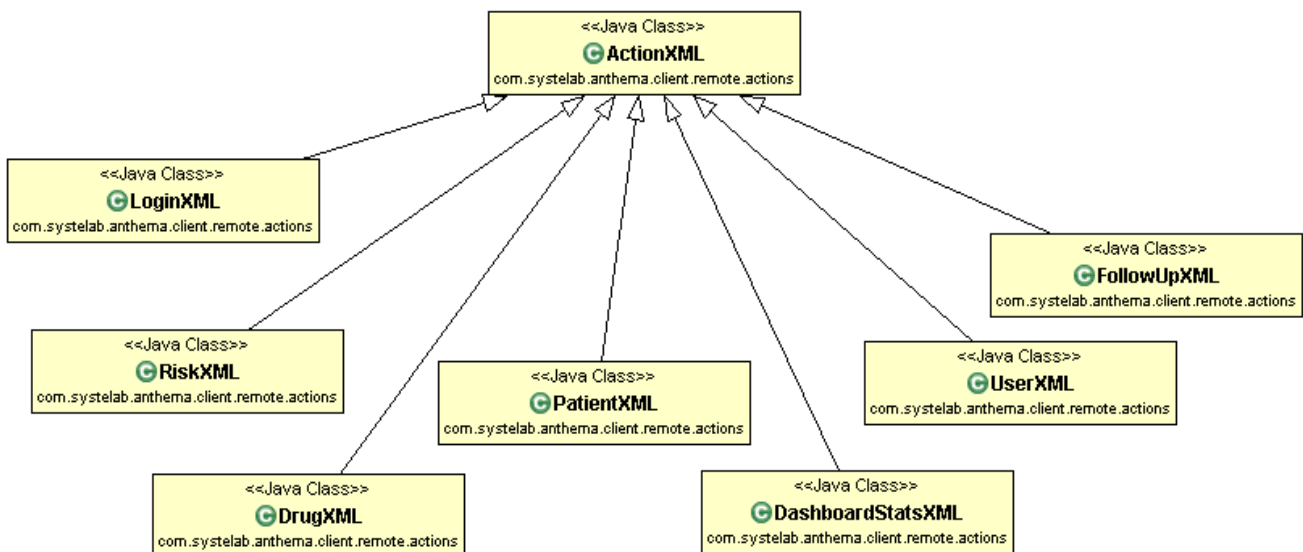


Figura 20: Patró Façana aplicat a la capa de Presentació

Com es veu en el diagrama, existeix una entitat genèrica anomenada ActionXML de la qual neixen les altres entitats específiques per cada funcionalitat. Per exemple, apareix l'entitat UserXML que és la que s'encarrega de gestionar les crides a la capa de Negoci de tot el que tingui a veure amb la gestió d'usuaris. La resta d'entitats responen a altres

funcionalitats de l'aplicació.

Per tant, el patró façana és útil en la banda client ja que permet diferenciar quines operacions s'efectuaran en la capa de Negoci. Per exemple, tot seguit es poden veure les operacions per dues d'aquestes entitats:

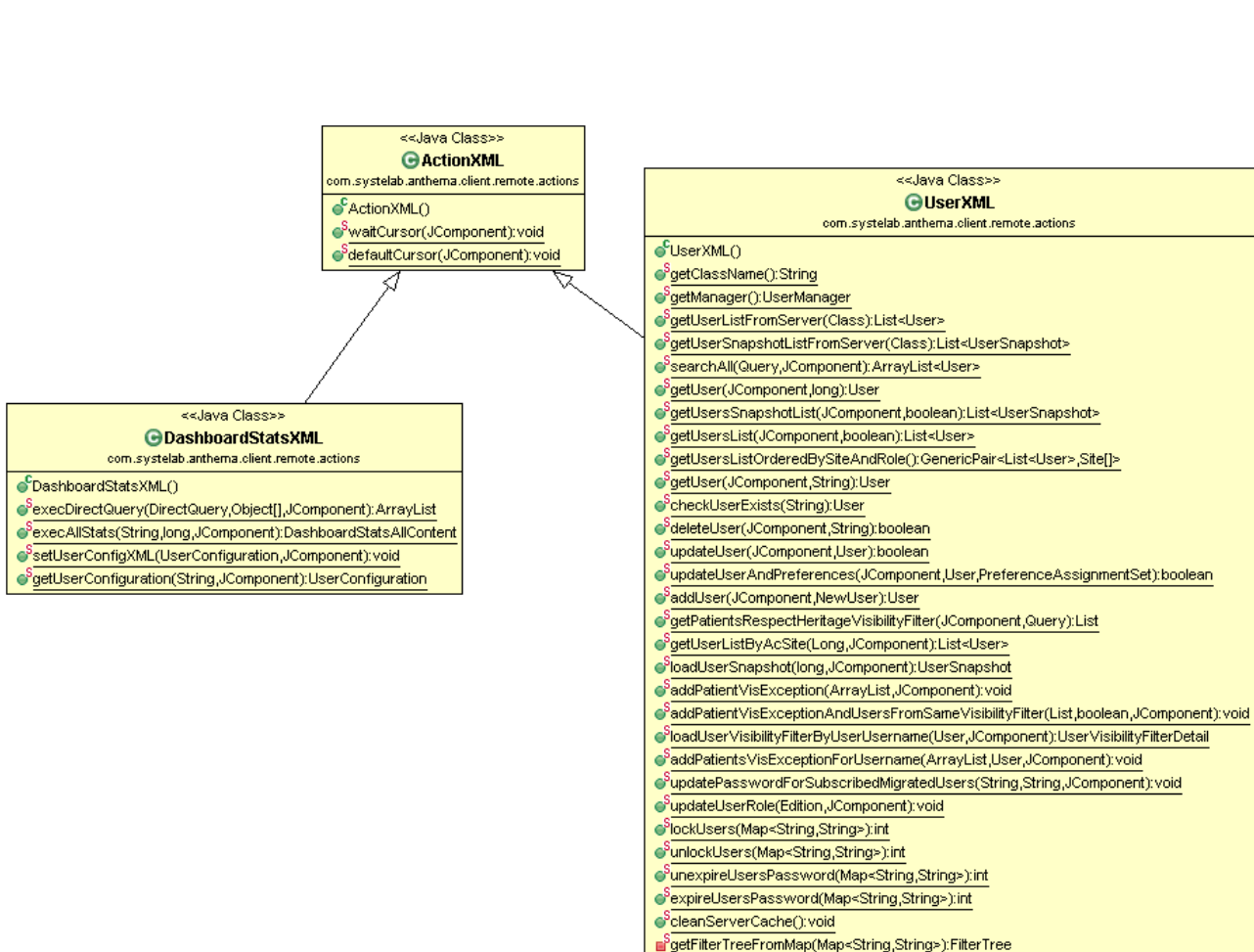


Figura 21: Patró Façana en la capa de Presentació detallat

En aquest cas, el patró façana diferencia clarament les operacions que pot gestionar UserXML amb DashboardStatsXML. Mentre que el primer pot operar sobre els usuaris de l'aplicació, l'altre pot executar consultes referents a les estadístiques.

6. 3. 2 Capa de Negoci

La capa de Negoci (o de Domini) sap com satisfer les peticions de l'usuari, però desconeix on són les dades i tampoc com es presenten a l'usuari. Les relacions són:

- **Relació amb la capa de Presentació:** rep els esdeveniments de l'usuari a través d'aquesta capa i també retorna les dades transformades i els resultats.
- **Capa de Persistència:** li comunica les operacions de consulta i modificació de dades i en rep els resultats.

6.3.2.1 Marcs de treball utilitzats

Per dissenyar la capa de Negoci s'ha aprofitat el marc de treball basat en la solució consistent en Enterprise Java Beans⁴, que com es veurà en el capítol d'implementació, proporciona diverses avantatges respecte a altres marcs de treballs i també aïlla als desenvolupadors d'aspectes complexos com són les transaccions, control de concurrència, etc.

Les entitats destacables d'aquesta capa són els següents:

- **Manager:** aquesta entitat consisteix en implementar les característiques pròpies dels Enterprise Java Beans de tipus sense Estat ni Sessió. En aquestes entitats s'implementen les operacions purament de lògica de negoci.
- **Query:** aquestes entitats són les encarregades de mantenir les operacions o consultes que caldrà que la capa de persistència apliqui a la base de dades, com també els paràmetres necessaris per executar-les correctament. Estan basades en arbres binaris de dades i disposen de mètodes per poder representar operacions lògiques i operacions de comparació.
- **DirectQuery:** aquesta entitat està dissenyada per quan la pròpia entitat Query no compleix amb les necessitats. Bàsicament, la diferència entre els dos objectes és que l'objecte Query representa les accions clàssiques de base de dades (operacions de consulta o modificació), i en canvi DirectQuery és capaç de llençar paquets de procediments de base de dades o fins i tot sentències SQL directament sense passar per un parseig posterior.

4 EJB. Solució de disseny basada en Java. (Veure Glossari al capítol 1).

- **Model de dades:** aquesta entitat té un caire més genèric i representa tot el model de dades necessari per a que la capa de Negoci pugui treballar correctament.

6. 3. 2. 2 Patrons de disseny

En la capa de negoci s'han utilitzat els següents patrons de disseny.

- **Patró Façana:** aquest patró s'utilitza per separar i diferenciar les operacions disponibles en la lògica de negoci tenint en compte de quina funcionalitat que tracta. Per exemple, la gestió de la lògica relacionada amb usuaris disposarà d'una sèrie d'operacions que no tenen res a veure amb les operacions que fan referència a medicaments. Per això es separen aquestes operacions segons la temàtica en un sol punt d'entrada. D'aquesta manera s'acota millor cada funcionalitat i el disseny és més simple i mantenible.

Un dels principals motius pels quals l'aplicació utilitza una especificació basada en els Enterprise Java Beans és que permet diferenciar en diferents EJBs la funcionalitat comuna. Per posar-ne un exemple, l'aplicació aplica el patró façana per diferenciar la gestió de l'agenda del sistema amb la gestió dels pacients i usuaris del sistema. Això es pot fer definint un EJB per cada funcionalitat. Fent això també s'aconsegueix aplicar un patró Façana en el sistema.

6. 3. 2. 3 Diagrames del patró Façana.

Tal com s'ha descrit en l'anterior apartat, el sistema aplica el patró façana diferenciant cada funcionalitat comuna i situant-la en un EJB específic i diferent. Tot seguit es pot veure la representació del patró observant-ne dos EJB.

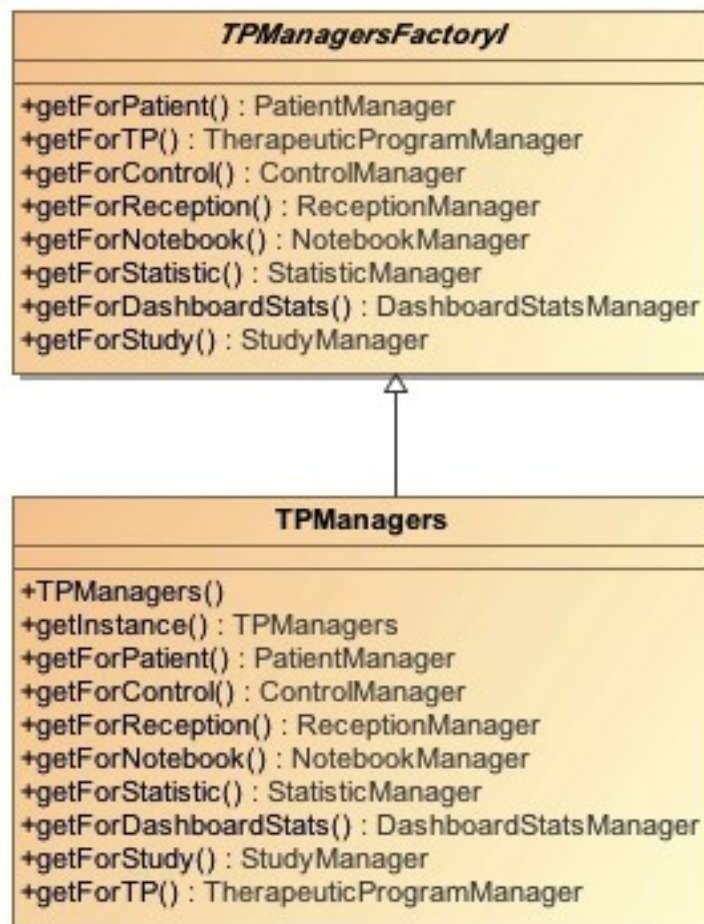


Figura 22: Exemple de patró Façana en la capa de Negoci (1)

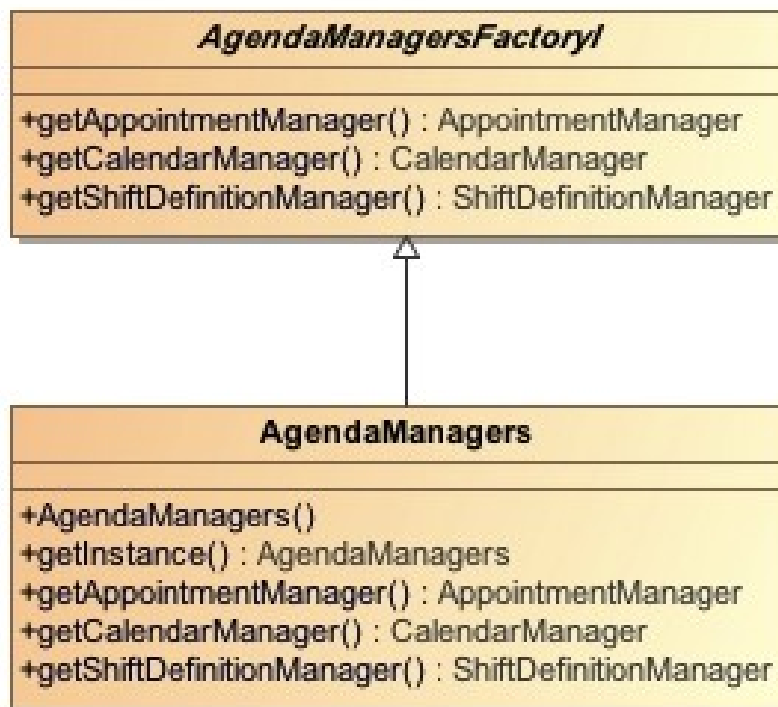


Figura 23: Exemple de patró Façana en la capa de Negoci (2)

Com es pot veure en la imatge anterior, existeixen dos Managers que concentren funcionalitat que no tenen a veure un amb l'altre. A part, també s'hi ha afegit la construcció d'aquestes entitats que s'aconsegueix mitjançant Factorys.

6. 3. 3 Capa de Persistència

La capa de Persistència coneix on són les dades, com s'emmagatzemen i com s'hi pot accedir, però no té coneixement sobre com tractar-les. Aquesta capa es relaciona amb:

- **Relació amb la capa de Negoci:** li retornarà els resultats i les respostes a les preguntes o operacions que aquesta capa ha efectuat. Normalment, les operacions són de tipus, o bé *consulta* o bé de *modificació*.
- **Relació amb el gestor de bases de dades:** la capa envia les operacions al gestor de bases de dades per a que aquest les executi.

6. 3. 3. 1 Marcs de treball utilitzats

Aquesta capa es basa en un marc de treball creat per la pròpia empresa i que s'anomena Framework DAO. Aquest disseny es va efectuar als inicis de l'any 2000, justament quan el projecte s'iniciava. En aquesta època no es coneixien altres eines de persistència de dades que oferissin tot el que el projecte demanava en aquell moment. Els desenvolupadors, per tant, van decidir crear un sistema propi de persistència que satisfés les demandes del projecte. El marc de treball té una complexitat elevada però al mateix temps permet totes les accions que podria oferir qualsevol eina actual com Hibernate.

El disseny d'aquest marc de treball consisteix en una sèrie d'entitats que, unides entre elles, permeten proveir a la capa de Negoci les dades que aquesta sol·licita. Concretament:

- **Persistor**: és l'entitat que s'encarrega d'obtenir la connexió amb la base de dades i proveir a la capa de negoci d'una sèrie de mètodes per poder accedir a les taules de la base de dades. Concretament, la funció principal és la d'efectuar un *mapeig* entre els objectes Java amb les Entitats Relacionals del model de dades. A més a més, aquesta entitat disposa d'un sistema de filtres de visibilitat per tal que les accions que es facin a la base de dades puguin ser parametritzades en funció de l'usuari. I també genera una sèrie de traces de logs que es desen en la mateixa de dades. Aquest fet és important per mantenir un control de les accions que s'han dut a terme. Com ja es veurà en l'apartat d'implementació, aquesta part important de la capa es basa en una implementació d'EJB (Enterprise Java Beans) de tipus Estat i sense Sessió.
- **SQLQueryParser**: aquesta entitat, com el seu nom indica, s'encarrega de parsejar l'objecte Query que prové de la capa de Negoci. Bàsicament el que fa és preparar les operacions que es llençaran contra la base de dades utilitzant els paràmetres que provenguin de la capa de Negoci.
- **VisibilityFilter**: com ja s'ha dit anteriorment, el VisibilityFilter permet reescriure les

operacions o consultes de base de dades afegint noves clàusules en funció d'uns paràmetres concrets.

- **Schema:** les entitats de tipus Schema s'utilitzen per definir els noms de les taules i de les columnes de la base de dades.
- **DAO:** l'objecte DAO o Data Access Object és la peça fonamental de la capa de Persistència. És l'encarregat d'executar les consultes o operacions a la base de dades i també, a partir del resultat, preparar un objecte Java amb les dades per poder-lo tractar a la capa de Negoci.
- **DirectQueryDAO:** aquesta entitat és molt semblant a l'entitat DAO però està orientada a satisfer les consultes de tipus DirectQuery i no de Query. L'objectiu però, és el mateix: aconseguir que la base de dades retorni a la capa de Negoci el que ha demanat.

A banda d'aquestes entitats, per completar el disseny de la capa de persistència cal també destacar la preparació i les tasques que haurà de fer la pròpia base de dades. Tot i que no està relacionada amb la part de codificació, la base de dades necessita unes estructures de dades concretes i una sèrie de procediments per poder extreure'n les dades. Aquestes estructures es detallaran en profunditat en el capítol d'implementació, concretament en la part de la capa de persistència.

En la següent imatge es pot veure un diagrama on apareixen els objectes anteriorment descrits:

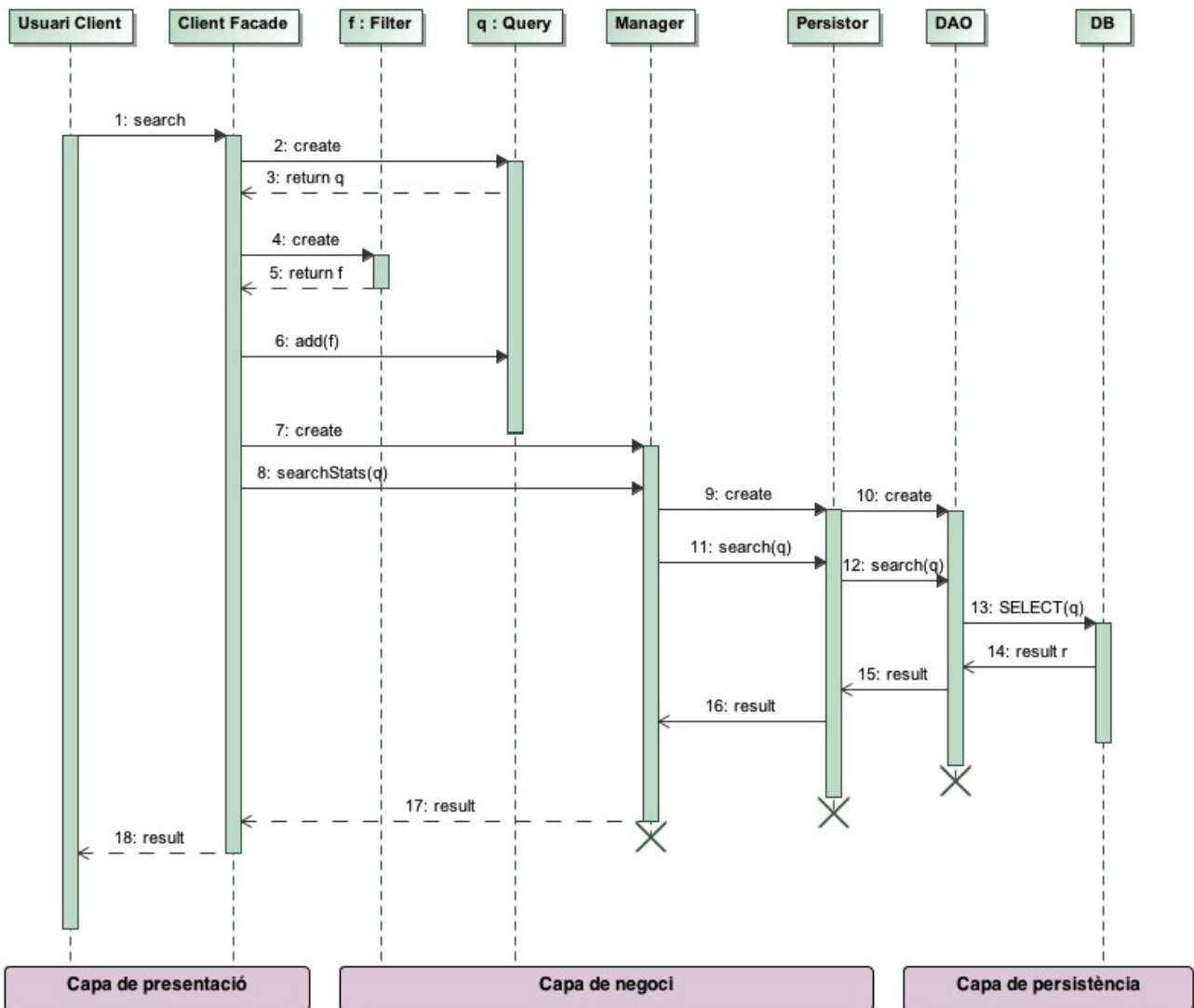


Figura 25: Diagrama de seqüència genèric.

Com es pot veure en l'anterior imatge, s'observa com des de la banda client s'accedeix a la Façana Client. En aquest moment es preparen els elements necessaris per a que la capa de negoci pugui fer la seva funció. Concretament, es prepara en primer lloc l'objecte Query que és qui conté la pregunta a efectuar, com també l'objecte Filtre, que és l'encarregat de contenir els paràmetres necessaris.

En aquest punt ja intervé la capa de Negoci, que rep els paràmetres de client i decideix, també en funció del patró Façana, quina entitat de la capa de Persistència haurà de cridar. Per tant, el Manager de la capa de Negoci cridarà al Persistor de la capa de Persistència que convingui.

Aquest Persistor crearà l'objecte DAO que serà l'encarregat d'accedir a la base de dades i

obtenir el resultat. Un cop disposa del resultat, la informació torna enrere capa per capa. A la inversa de com ha arribat la petició, la capa de Persistència comunica a la capa de Negoci que ja té disponible el resultat. I Aquesta, retorna a la banda del client els valors obtinguts mitjançant la operació sol·licitada.

6. 4. 1 Diagrama de seqüència. Veure resultats estadístics

La següent imatge mostra el diagrama de seqüència pel cas d'ús de veure resultats estadístics. Com es pot veure, requereix de l'existència de dues entitats Persistor, una per obtenir la configuració pròpia de l'usuari i l'altra, un cop obtinguda la configuració, cercar les gràfiques que li pertocuen.

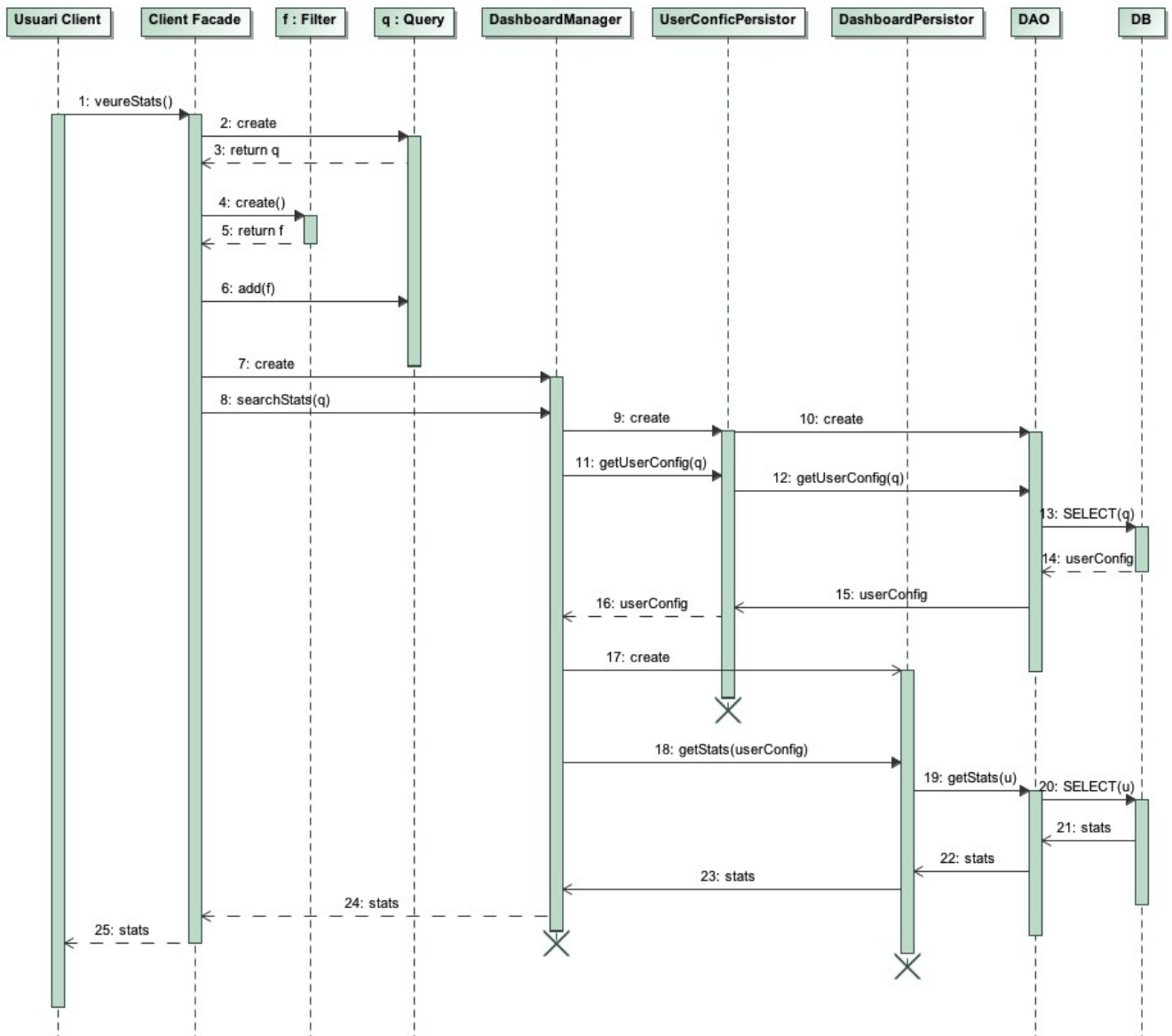


Figura 26: Diagrama de seqüència. Veure resultats estadístics

Una altra particularitat d'aquest diagrama es pot veure en l'ús que es fa de l'objecte DAO, ja que utilitzant-ne només un podem efectuar les dues operacions de cerca.

6. 4. 2 Diagrama de seqüència. Afegir nova gràfica al mòdul

A continuació es pot veure el diagrama pel cas d'ús d'afegir una nova gràfica al mòdul principal de l'usuari.

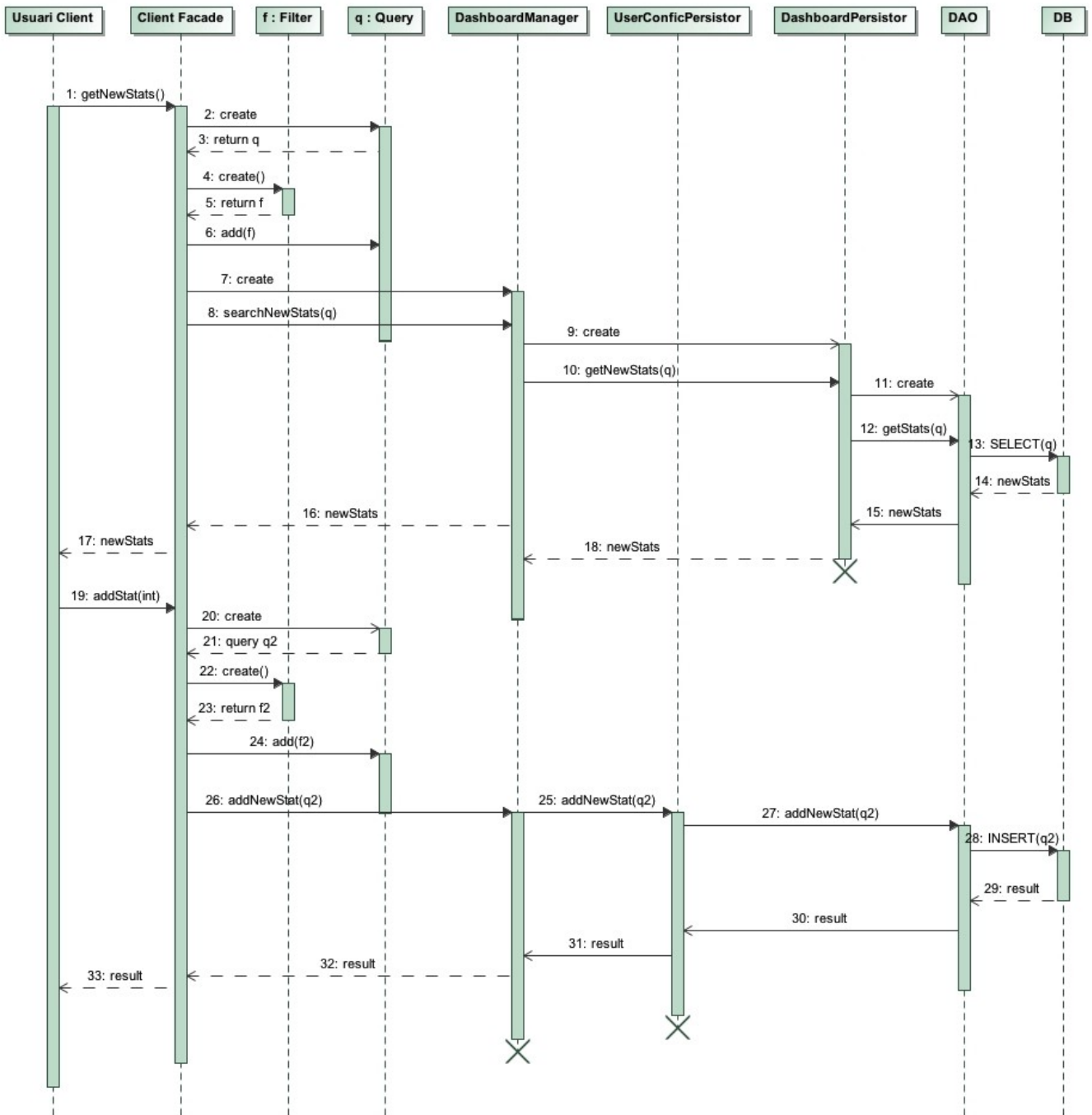


Figura 27: Diagrama de seqüència. Afegir nova gràfica

6. 4. 3 Diagrama de seqüència. Eliminar gràfica al mòdul

A continuació es pot veure el diagrama pel cas d'ús d'eliminar una gràfica del mòdul principal de l'usuari.

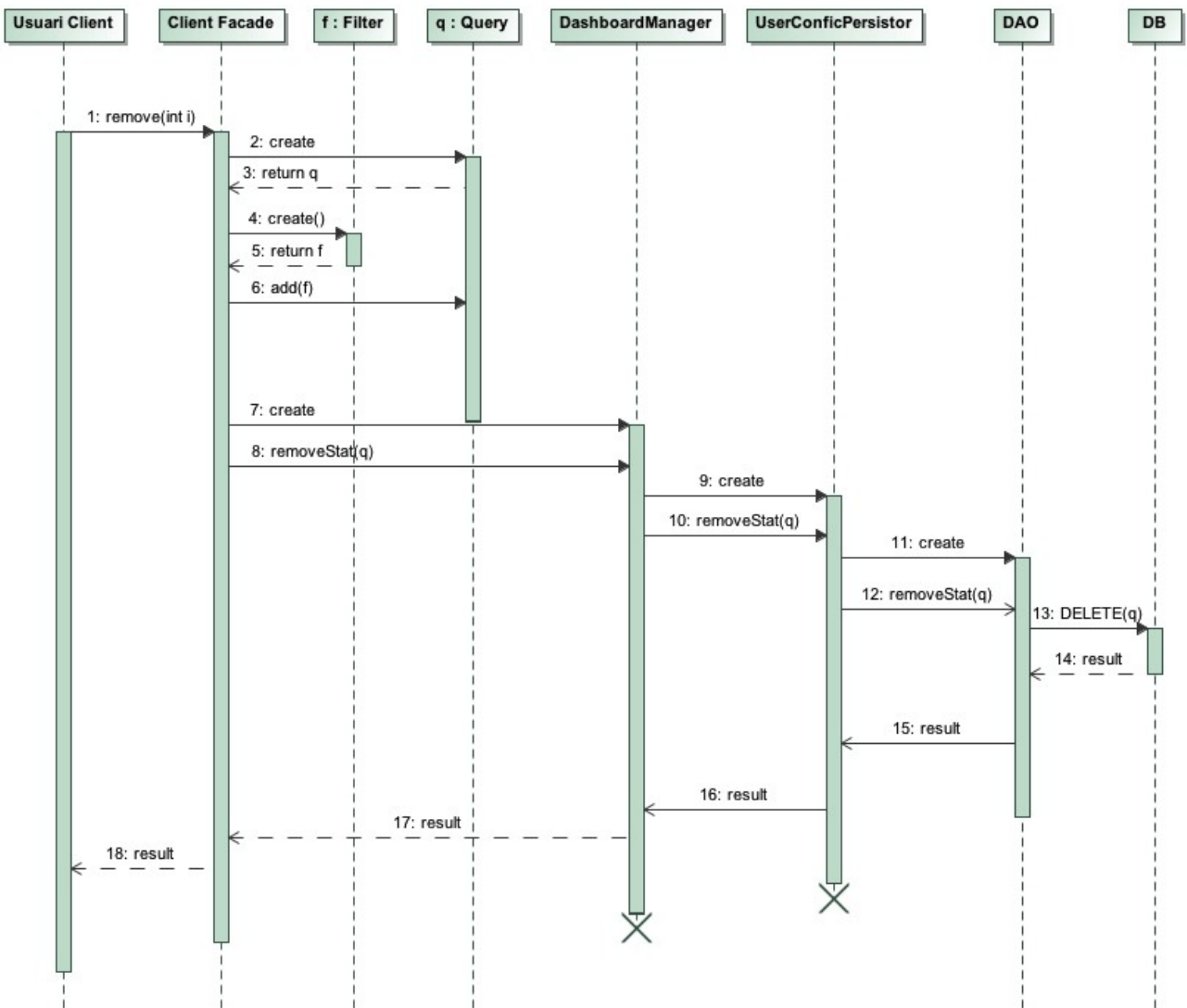


Figura 28: Diagrama de seqüència. Eliminar gràfica

6. 4. 4 Diagrama de seqüència. Redimensionar gràfica

A continuació es pot veure el diagrama pel cas d'ús de redimensionar una gràfica al mòdul principal de l'usuari.

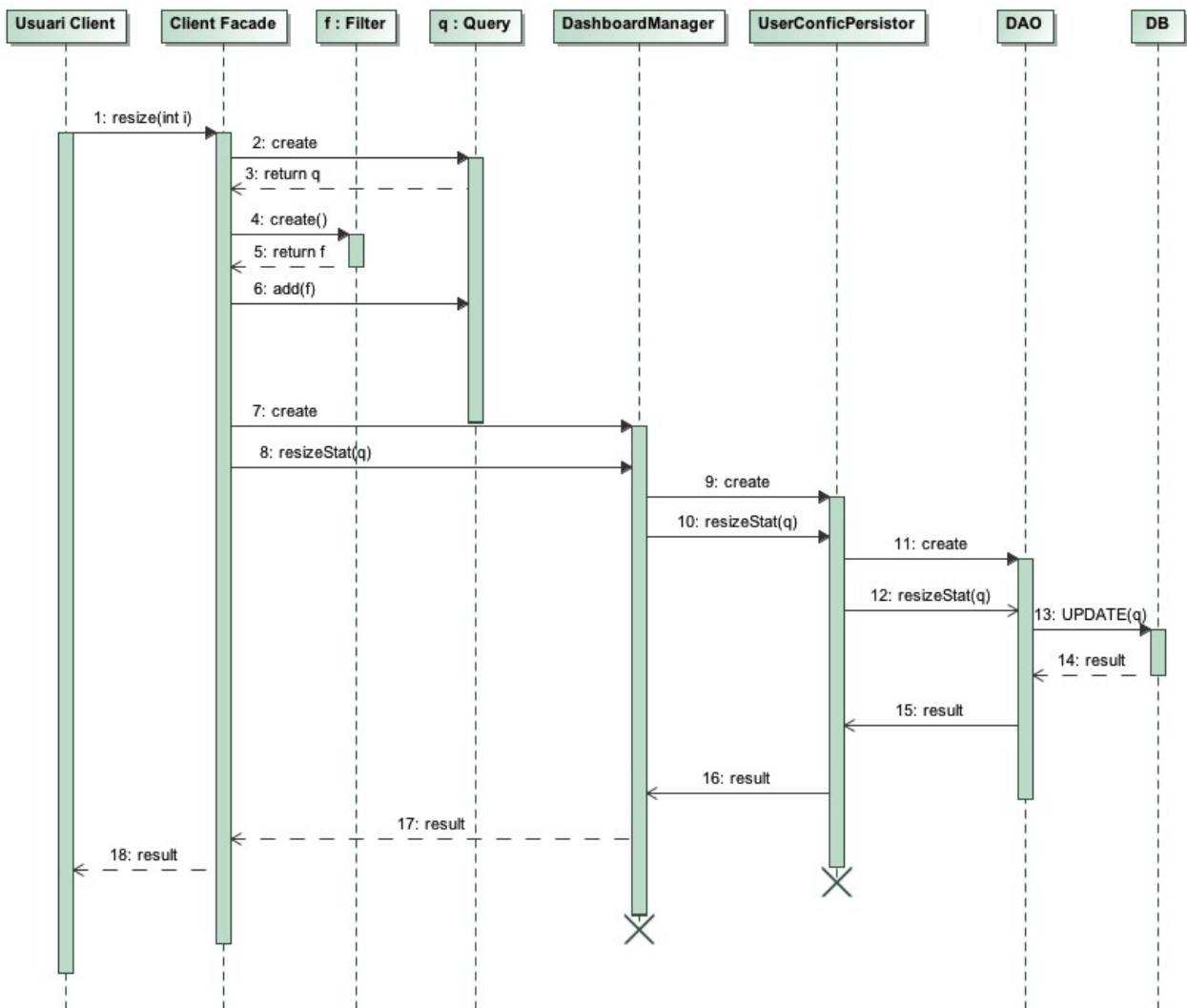


Figura 29: Diagrama de seqüència. Redimensionar gràfica

6. 4. 5 Diagrama de seqüència. Ressituar gràfica

A continuació es pot veure el diagrama pel cas d'ús d'afegir una nova gràfica al mòdul principal de l'usuari.

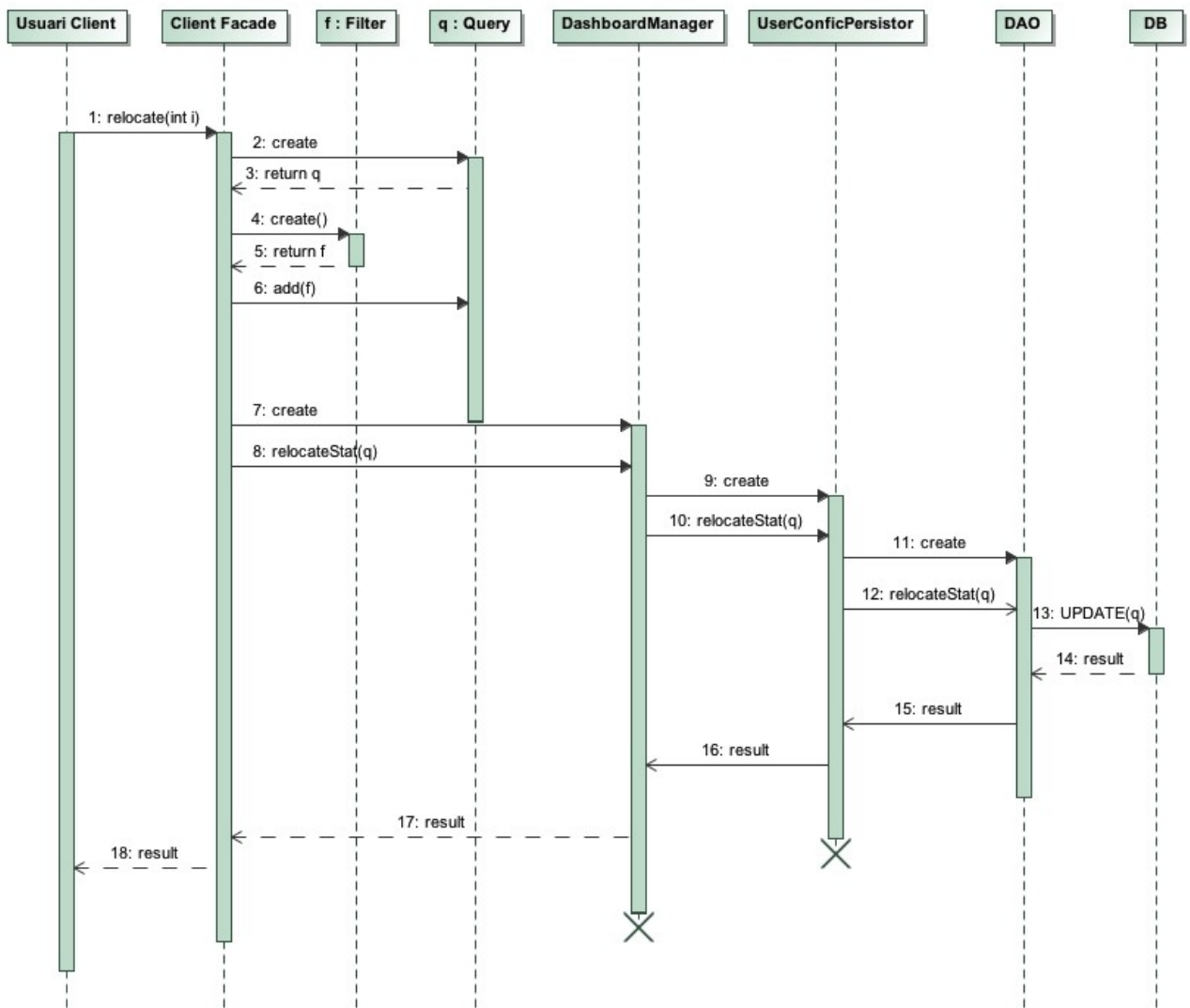


Figura 30: Diagrama de seqüència. Ressituar gràfica

7. Implementació

És en aquest punt on es dona a conèixer amb més profunditat com s'ha implementat el disseny explicat en el punt anterior. La explicació es basarà en la separació per capes de l'aplicació i detallarà les tasques i els aspectes més importants de cadascuna. Com a característica comuna d'entre totes les capes de disseny, cal dir que el llenguatge utilitzat ha estat Java.

7. 1 Capa de Persistència

Els canvis efectuats en la capa de Persistència han estat, en primer lloc, la creació d'una estructura de dades per mantenir els càlculs de les estadístiques i, en segon lloc, la implementació d'un sistema de tasques programades que permeti executar els processos periòdicament. Com es veurà al llarg d'aquest apartat, s'han creat un conjunt de taules a la base de dades per mantenir tant les estadístiques com els seus resultats, i també s'ha creat un paquet de procediments SQL que es llençaran periòdicament utilitzant una sèrie de tasques programades per omplir les estructures amb dades calculades.

7. 1. 2 Noves taules a la base de dades

- **Per mantenir la configuració de l'usuari**

Amb l'objectiu de mantenir la configuració de les gràfiques que es mostren a l'usuari i tota la distribució i repartiment en la seva pantalla principal d'estadístiques queda definida la següent taula. La taula, a part de la configuració de la plana principal (dashboard) també s'ha utilitzat per desar-hi configuracions d'altres parts de l'aplicació, com per exemple, la configuració que l'usuari fa de les columnes que mostren els llistats de l'aplicació.

Els objectes que s'hi desen són de tipus BLOB i consisteixen en un fitxer XML. Més endavant es detallarà què conté aquest fitxer.

Taula de configuració d'usuari: APP_USER_CONFIGURATION

Nom de la columna	Tipus	NULL	Comentari
USER_USERNAME	VARCHAR2	No	Identificador de l'usuari
USER_DASHBOARD_XML	BLOB	Sí	Configuració del dashboard de l'usuari
USER_CONFIG_XML	BLOB	Sí	Configuració genèrica d'usuari de l'aplicació
AVK_CONFIG_XML	BLOB	Sí	Configuració d'usuari de teràpia clàssica
HOSPITALIZATION_CONFIG_XML	BLOB	Sí	Configuració d'usuari de les pantalles d'hospitalització
LONGTERM_CONFIG_XML	BLOB	Sí	Configuració dels aspectes de teràpia de llarg termini de l'aplicació

- **Per mantenir els resultats i executar les estadístiques**

i) Taula de definició d'estadístiques: STATS_DEFINITION

El seu objectiu és definir de manera concreta les característiques generals de cada estadística.

Nom de la columna	Tipus	NULL	Comentari
STATS_ID	NUMBER	No	Identificador de l'estadística
STATS_NAME	VARCHAR2	Sí	Nom de l'estadística
STATS_SCRIPT	VARCHAR2	Sí	Script de l'estadística (sentència SQL)
STATS_PARAM_DEF_ID	NUMBER	Sí	Identificador de la definició de Paràmetres
STATS_RESULT_DEF_ID	NUMBER	Sí	Identificador de la definició dels resultats
STATS_CALCULATED_BY	VARCHAR2	Sí	Tipus d'objecte utilitzat per l'estadística. Possibles valors: User, Center o Installation
STATS_USER_ROLE	VARCHAR2	Sí	Si l'estadística es calcula per Usuari, aquest camp permet definir el seu rol.
STATS_OBJECT_ID_IN	VARCHAR2	Sí	Si l'estadística es calcula per centre o usuari, permet filtrar pels usuaris o centres descrits en aquest camp, separats per comes.
STATS_GROUP_BY	VARCHAR2	Sí	El criteri de Group By de l'estadística
STATS_TYPE_GRAPH	VARCHAR2	Sí	Conté el tipus de gràfic a mostrar. Possibles valors: Lineal, Pie
STATS_UNITS	VARCHAR2	Sí	Conté les unitats dels resultats. Possibles valors: Number, Percentage
STATS_IN_REAL_TIME	VARCHAR2	Sí	Execució en temps real o no.

ii) Taula de definició dels paràmetres d'execució: STATS_EXEC_PARAMS

La taula s'utilitza per mantenir els paràmetres que es s'empraran en les execucions de les estadístiques, així com les seves propietats com son l'ordre, el seu valor, etc.

Nom de la columna	Tipus	NULL	Comentari
STATS_EXEC_PARAM_ID	NUMBER	No	Identificació dels paràmetres de l'estadística
STATS_EXEC_PARAM_NAME	VARCHAR2	Sí	Nom dels paràmetres
STATS_EXEC_PARAM_POSITION	NUMBER	No	Posició del paràmetre en l'script de l'estadística
STATS_EXEC_PARAM_VALUE	VARCHAR2	Sí	Valor del paràmetre per la seva execució a l'script

iii) Taula per definir els resultats de les execucions: STATS_EXEC_RESULTS

Bàsicament la seva escomesa és la de mantenir els camps dels resultats de les execucions, així com les seves característiques.

Nom de la columna	Tipus	NULL	Comentari
STATS_EXEC_RES_ID	NUMBER	No	Identificació dels resultats de l'execució
STATS_EXEC_RESULT_FIELD	VARCHAR2	No	Nom del camp en l'execució de l'estadística.
STATS_EXEC_RESULT_VALUE	VARCHAR2	Sí	Valor del resultat del camp de l'execució de l'estadística
STATS_EXEC_GROUP_BY	VARCHAR2	No	Nom de la sèrie dels resultats de l'execució
RESULT_ORDER	NUMBER	Sí	Indica l'ordre dels resultats de l'execució

iv) Taula de control d'execucions: STATS_EXECUTION

Aquesta taula manté la informació de les successives execucions de les estadístiques i les relaciona amb els seus paràmetres d'entrada i els seus resultats.

Nom de la columna	Tipus	NULL	Comentari
STATS_EXEC_ID	NUMBER	No	Identificador de l'execució de l'estadística
STATS_ID	NUMBER	Sí	Identificador de l'estadística executada
STATS_OBJECT_TYPE	VARCHAR2	Sí	Tipus d'objecte usat pel càlcul
STATS_OBJECT_ID	VARCHAR2	Sí	Identificador de l'objecte d'estadístiques executat. Si l'estadística s'ha calculat per centre, haurà de contenir l'ID del centre.
STATS_DATE	DATE	Sí	Data de l'execució de l'estadística
STATS_START_TIMESTAMP	TIMESTAMP	Sí	Timestamp de l'inici de l'execució
STATS_END_TIMESTAMP	TIMESTAMP	Sí	Timestamp de finalització del càlcul
STATS_EXEC_PARAMS_ID	NUMBER	Sí	Identificació del paràmetre d'execució que conté el valor dels paràmetres en l'execució de l'estadística
STATS_EXEC_RESULT_ID	NUMBER	Sí	Identificador dels resultats estadístics que contenen els resultats de les execucions

v) Taula de definició de paràmetres: STATS_PARAM_DEF

Taula que permet mantenir paràmetres ja configurats per les execucions de les estadístiques. Els valors d'aquesta taula poden ser utilitzats des de la taula STATS_EXEC_PARAMS que és on es preparen per les execucions.

Nom de la columna	Tipus	NULL	Comentari
STATS_PARAM_ID	NUMBER	No	Identificador del paràmetre de l'estadística
STATS_PARAM_NAME	VARCHAR2	Sí	Nom del paràmetre que caldrà utilitzar en l'script de l'estadística.
STATS_PARAM_POSITION	NUMBER	No	Posició del paràmetre a utilitzar en l'escript de l'estadística
STATS_PARAM_VALUE	VARCHAR2	Sí	Conté el valor del paràmetre a usar en l'script de l'estadística

vi) Taula de definició de resultats: STATS_DEF_RES

Aquesta taula té un funcionament similar que STATS_PARAM_DEF però basant-se en els resultats. Per tant, aquesta taula permet preparar camps de resultat per a que després es puguin utilitzar des de la taula STATS_EXEC_RESULTS.

Nom de la columna	Tipus	NULL	Comentari
STATS_RES_DEF_ID	NUMBER	No	Identificador del resultat de l'estadística
STATS_RES_NAME	VARCHAR2	Sí	Nom del resultat dins l'script de l'estadística
STATS_RES_POSITION	NUMBER	No	Posició del resultat en l'script de l'estadística
STATS_RES_COLOR	VARCHAR2	Sí	Color a utilitzar per pintar el resultat.

7. 1. 3 Oracle, procediments i tasques programades (Jobs)

Tot el sistema d'estadístiques no tindria sentit si aquestes no es poguessin executar de manera periòdica i totalment autònoma. En aquest apartat es veuen quines eines s'han fet servir per a que, utilitzant les noves taules anteriorment descrites, es puguin executar les estadístiques de manera periòdica.

- **Oracle Database**

Abans d'entrar en les característiques pròpies de la base de dades és necessari conèixer quin és el producte utilitzat per dur a terme les tasques de persistència de dades.

Oracle Database és un producte creat per Oracle Corporation que consisteix en un Sistema de Gestió de Bases de Dades Relacional (SGBDR). Aquest sistema és un dels més complets i potents del mercat actual i té com a principals fites les següents característiques:

- *Suport en les transaccions*
- *Estabilitat*
- *Escalabilitat*
- *Suport multi-plataforma*

Tot i que tan sols és una petita pinzellada del que és Oracle, queda clar que aquest sistema serà bàsic per poder mantenir un sistema d'estadístiques com el que s'ha creat.

- **Paquet de procediments**

Una altra de les característiques d'Oracle Database és la possibilitat que ofereix de crear i preparar un paquet de procediments amb disponibilitat d'execució permanent. Aquest fet és molt important ja que permetrà poder executar les estadístiques en qualsevol moment.

El paquet de procediments disposa d'una sèrie de mètodes que el que fan és obtenir les dades (les estadístiques) de les taules anteriorment detallades i preparar-les utilitzant les definicions i els paràmetres.

Els mètodes són els següents:

Mètode	Declaració SQL	Explicació
Execute_Statistic	procedure Execute_Statistic (Statistic_Id in Number);	Aquest mètode rep un "enter" com a paràmetre que és l'identificador de l'estadística
Execute_All_Statistics	procedure Execute_All_Statistics;	Permet executar totes les estadístiques en una sola crida
Immediate_For_User	function Immediate_For_User(Statistic_Id In Number, Stat_user_id In Number) return Number;	Permet l'execució d'una estadística immediata, és a dir, no programada per usuari
All_Immediate_For_User	function All_Immediate_For_User(Stats_u ser_id In Number) return varchar2;	Permet executar totes les estadístiques per usuari no programades

Un cop observades les declaracions dels mètodes del paquet de procediments es mostra de quina manera es poden executar.

Si es volgués executar un mètode en concret, tant des d'un programari client com des de la mateixa consola de la base de dades, es podria fer de la següent forma:

Execució d'una estadística en concret (identificador = 3)

```
execute New_Stats.Execute_Statistic(3);
```

Execució de totes les estadístiques en una crida

```
execute New_Stats.Execute_All_Statistics();
```

Per si es té més curiositat sobre la implementació dels mètodes del paquet de procediments, en el tercer apartat de l'Annex es pot veure el codi dels procediments.

- **Tasques programades (Jobs)**

Oracle Database també ofereix una solució interessant per programar execucions d'operacions SQL. Concretament, ofereix uns elements anomenats Oracle Jobs que permeten definir un calendari per unes tasques en concret i la mateixa base de dades s'encarregarà d'executar-les.

Els elements que cal tenir en compte per poder configurar correctament un Job d'Oracle són els següents:

- **What:** bàsicament és el procediment que vol que s'executi a una freqüència en concret.
- **Next_date:** és la data i la hora en que s'executarà per primer cop la tasca programada.
- **Interval:** el període de temps que es vol que es torni a executar el procediment.

Tot seguit es mostra un exemple de com es podria configurar un Job.

```
1: DECLARE
2:   X NUMBER;
3: BEGIN
4:   SYS.DBMS_JOB.SUBMIT
5:   (
6:     job      => X
7:     ,what    => 'New_Stats.Execute_All_Statistics();'
8:     ,next_date => to_date('02/02/2013 22:30:00', 'mm/dd/yyyy hh24:mi:ss')
9:     ,interval => 'NEXT_DAY(TRUNC(SYSDATE), 'SUNDAY')'
10:    ,no_parse => FALSE
11:   );
12:   :JobNumber := to_char(X);
13: END;
```

Com es veu en l'exemple anterior, en la línia número 7 es declara el procediment a executar, en la número 8 es formalitza la data de la primera execució. Concretament, es diu que la primera execució es farà el segon dia del més de febrer de 2013 a dos quarts d'onze de la nit (22:30h). Per últim, es declara com a exemple que la tasca s'executi cada diumenge.

7. 1. 4 Fitxer de configuració d'usuari

Per desar la configuració que pot efectuar el propi usuari en el dashboard d'estadístiques, és necessari una estructura de dades que sigui senzill de tractar i que no ocupi molt espai de dades.

Per poder fer-ho s'ha dissenyat un fitxer XML que és manté en la taula *APP_USER_CONFIGURATION* en el camp *USER_DASHBOARD* en forma de dades BLOB⁵.

Aquest fitxer utilitza l'especificació d'XML la versió 1.0 i fa servir un encoding UTF-8. El seu objectiu és estructurar la pantalla d'estadístiques com si fos un taulell de files i columnes on, en cada fila, poguessin aparèixer columnes variables.

Per exemple, una possible representació lògica podria ser:

	Títol de la pantalla general d'estadístiques		
<i>Fila 0</i>	Gràfic #2	Gràfic #3	(espai buit)
<i>Fila 1</i>	Gràfic #4	Gràfic #5	Gràfic #11

Com es pot veure en la representació anterior, la pantalla quedarà dividida en dues files (fila 0 i fila 1) i que a la vegada, les dues files disposen d'un nombre diferent de gràfics.

Per poder mantenir aquestes dades d'una manera senzilla es va dissenyar el següent model de fitxer XML:

```
<?xml version='1.0' encoding='UTF-8' ?>
<dashboard>
  <line height='248'>
    <chart id='2'/>
    <chart id='3'/>
  </line>
  <line height='255'>
    <chart id='4'/>
    <chart id='5'/>
    <chart id='11'/>
  </line>
</dashboard>
```

5 BLOB: Binary Large Objects. Element de base de dades que permet emmagatzemar fitxers de gran tamany que canvien de forma dinàmica.

Com es veu en l'exemple de fitxer XML, es declara en primer lloc la capçalera del fitxer XML (`<?xml version='1.0' encoding='UTF-8' ?>`) un tag⁶ general que és on comença i acaba la configuració del dashboard (`<dashboard></dashboard>`) i els elements que hi pertanyen.

L'esquema consisteix en primer lloc en declarar una línia, que té una propietat on diu l'alçada que ha de tenir en píxels (`<line height='248'>`) i dins d'aquest tag s'hi especifiquen els gràfics juntament amb el seu identificador (`<chart id='4'/>`).

Aquest fitxer es valida d'una manera eficient en l'aplicació i permet mantenir la configuració que fa cada usuari de la seva pantalla d'estadístiques.

7.2 Capa de Negoci

Com s'ha introduït en l'apartat de disseny, la capa de Negoci està basada en la solució d'Oracle Enterprise Java Beans. En aquest punt s'entrarà més en detall i s'explicarà en què ens ajuden els EJBs i quines són les seves característiques més importants.

Enterprise Java Beans

Els EJB són una API (Interfície de Programació d'Aplicacions) que formen part de l'estàndard de construcció d'aplicacions empresarials de J2EE (Java 2 Enterprise Edition), propietat d'Oracle Corporation i originalment creat per Sun Microsystems. L'especificació detalla com, des de la banda del servidor, es proveeixen els objectes a la banda de client. Aquests objectes reben el nom d'EJB.

L'ús dels EJB és molt important ja que estalvia molta feina als desenvolupadors. Com a característiques més importants es troben:

- Comunicació remota per CORBA
- Gestió de les transaccions

⁶ Etiqueta emprada per la codificació dels fitxers XML.

- Control de la concurrència
- Servei de noms i de directori (JNDI)
- Seguretat
- Ubicació de components en un servidor d'aplicacions

Existeixen diversos tipus d'EJB (de sessió, de missatge i d'entitat) però per simplificar, a continuació només s'explicarà en detall el tipus d'EJB que realment s'ha fet servir en el desenvolupament del projecte.

EJB de Sessió sense Estat:

Els Enterprise Java Bean de sessió sense estat (*session stateless*) s'encarreguen de gestionar el flux de la informació en el servidor. Com es va dir en el capítol de disseny, els EJB funcionen com una façana al client on cadascun d'ells ofereix una sèrie de serveis.

El funcionament dels EJB és el següent:

Els EJB es distribueixen dins de servidors d'aplicacions (contenidors d'EJB) i cal que cadascun d'ells faciliti una classe d'implementació Java i dues interfícies. Aquestes dues interfícies s'anomenen "Home" i "Remote" i especifiquen les signatures dels mètodes remots dels EJB.

- *Interfície Home*: permet al codi client manipular mètodes de classe dels EJB que no estan associats a cap instància en particular.
- *Interfície Remote*: especifica els mètodes d'instància encarregats de realitzar les operacions.

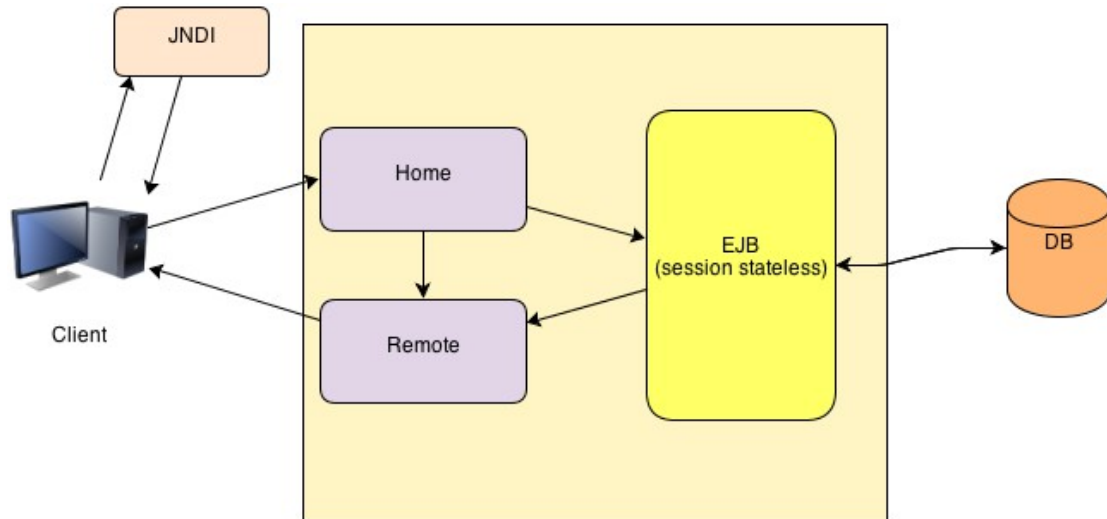


Figura 31: Esquema d'EJB 2.1

7.3 Capa de Presentació

Com també s'ha detallat en l'apartat de disseny, per la capa de Presentació s'han utilitzat diferents eines o solucions per tal de crear una interfície d'usuari usable i senzilla.

Concretament s'ha emprat Java Swing i Monarch Graphs, que s'han detallat en capítols anteriors. Tot i això, a continuació s'explicarà de manera visual el disseny implementat amb aquestes eines, i quin ha sigut el resultat final.

Per resoldre la implementació de les representacions gràfiques dels resultats estadístics s'ha optat per seguit aquest model:

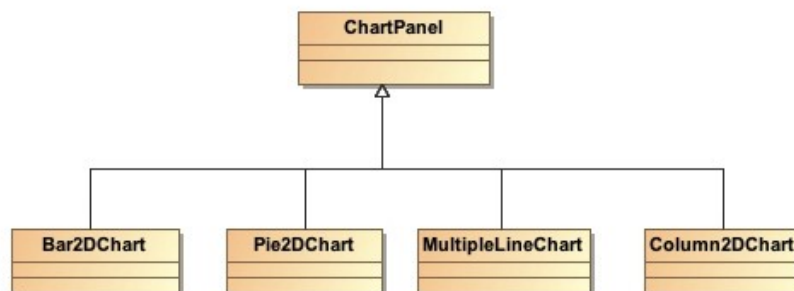


Figura 32: Estructura de les representacions gràfiques

Com es veu en la figura anterior, hi ha quatre tipus de gràfica que hereten d'una classe genèrica.

Un possible exemple, en aquest cas del tipus de gràfica de sectors podria ser el següent (Pie2DChart):

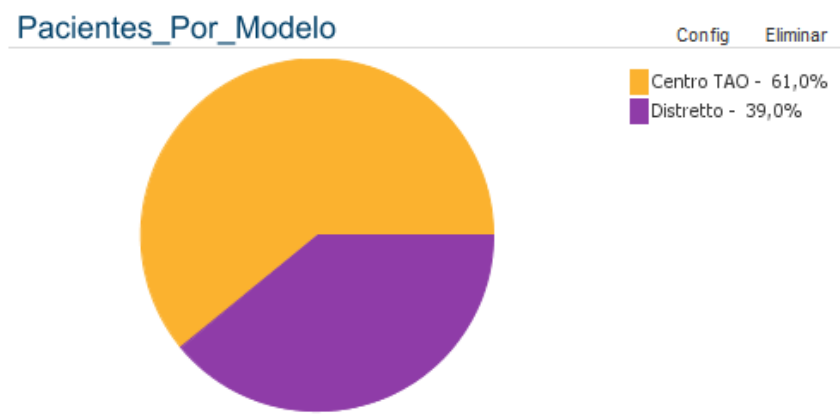


Figura 33: Exemple de gràfica de tipus sectors

Com mostra la figura anterior, cada gràfica disposa d'un títol, de les opcions de Configuració i Eliminar, i també de la llegenda amb els percentatges de les opcions.

L'opció d'Eliminar el que fa és fer la gràfica no visible en el mòdul d'estadístiques. I l'opció de Configuració permet substituir el gràfic per un altre.

A continuació es pot veure més en detall el disseny implementat en el cas dels gràfics.

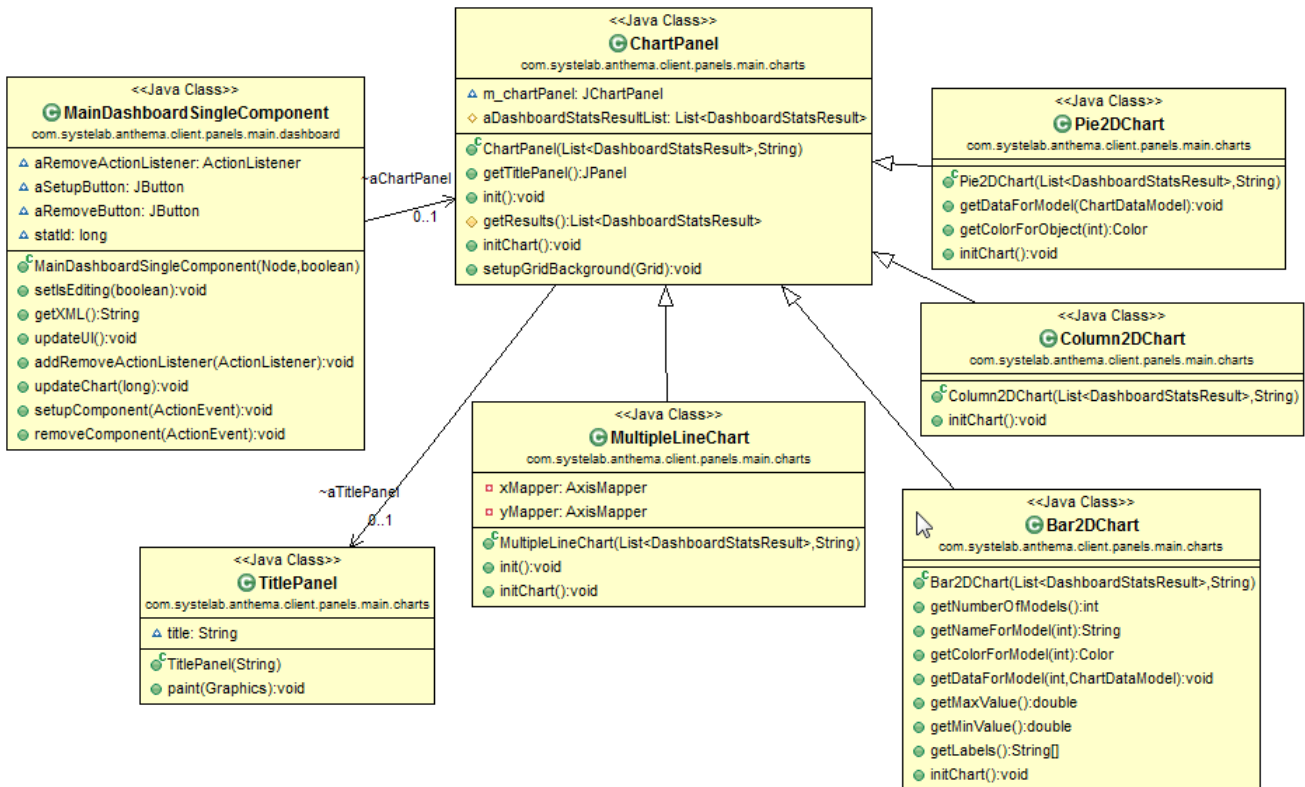


Figura 34: Implementació dels gràfics més detallada

I tot seguit s'exposa el disseny implementat del que visualment s'observa com una línia de gràfics. Com es pot veure, es basa en un panell anomenat MainDashboardLinePanel i que es comunica amb altres panells. El conjunt de panells disposen d'una sèrie de mètodes que són les accions que poden arribar a oferir.

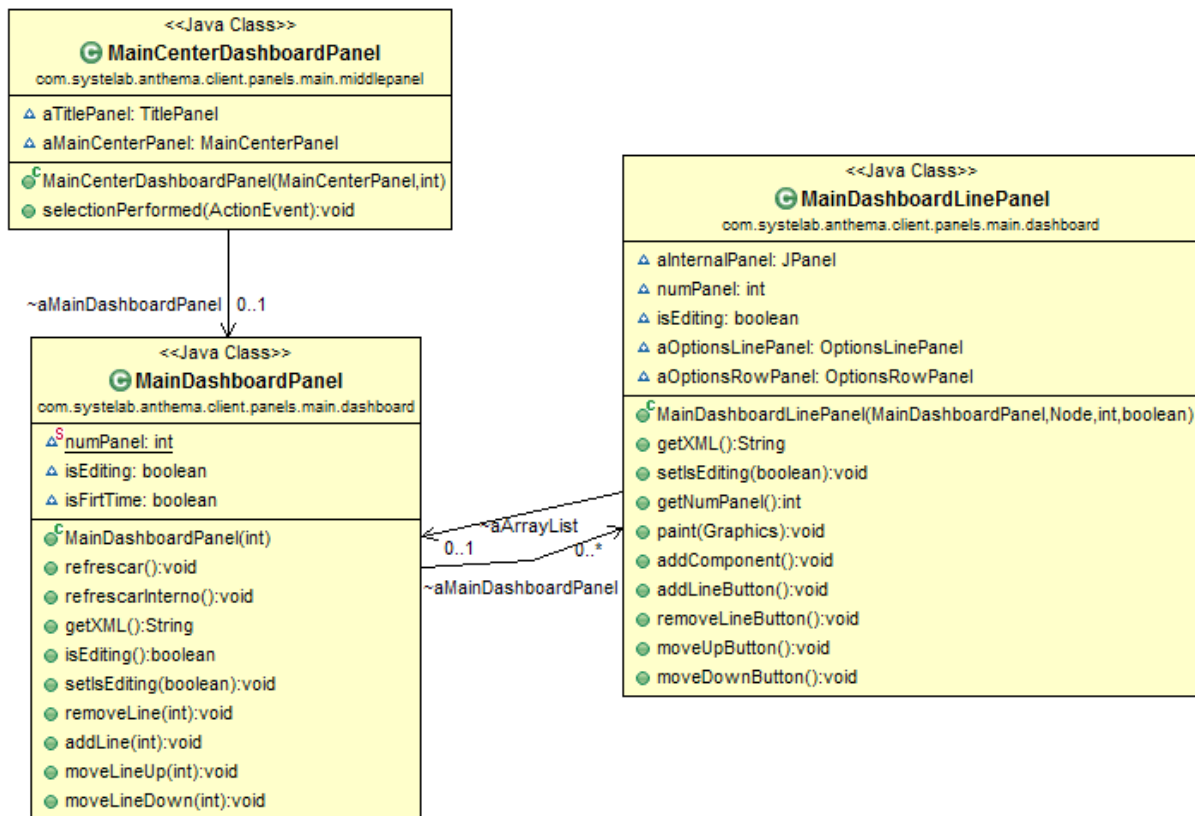


Figura 35: Implementació de les línies de gràfiques

A part, també s'han implementat les funcions de configuració d'aquests gràfics. Per fer-ho, s'ha implementat el següent disseny:

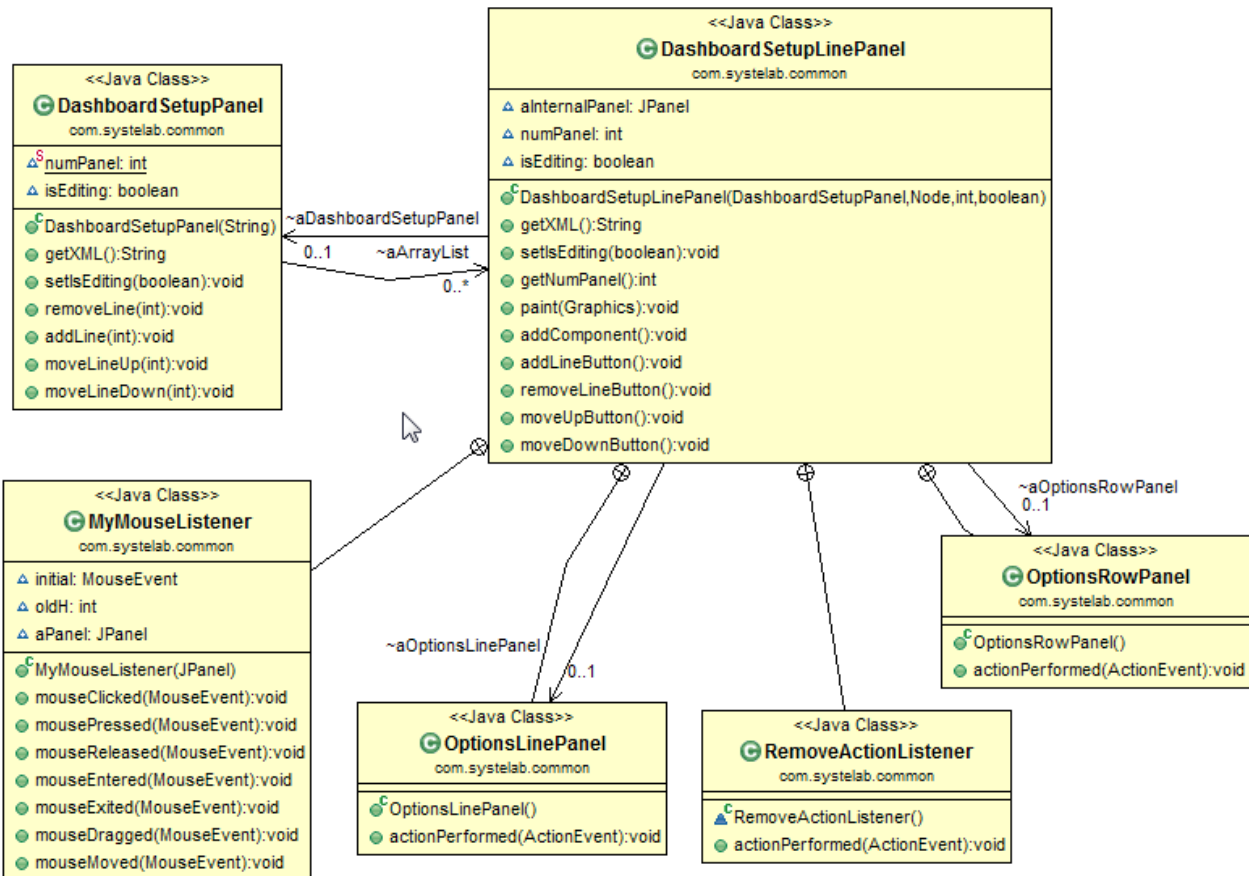


Figura 36: Implementació de la configuració del dashboard

Com s'ha especificat en el capítol de disseny, la representació de les gràfiques ha de consistir en una sèrie de files on, cadascuna d'elles conté un seguit de gràfiques. Tot i això, en la implementació s'ha definit cada fila com a una línia de gràfiques.

Amb la creació d'aquestes línies i la seva configuració s'obté, per exemple, un resultat com el següent:

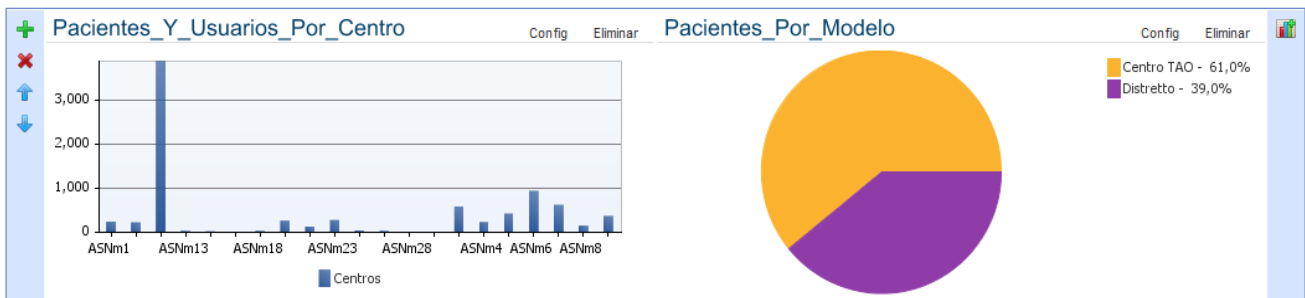


Figura 37: Una línia de gràfiques d'exemple.

Per altra banda, també s'ha creat un petit diàleg que permet mostrar a l'usuari el llista de

gràfiques que té disponibles. S'aconsegueix basant-se en el següent disseny:

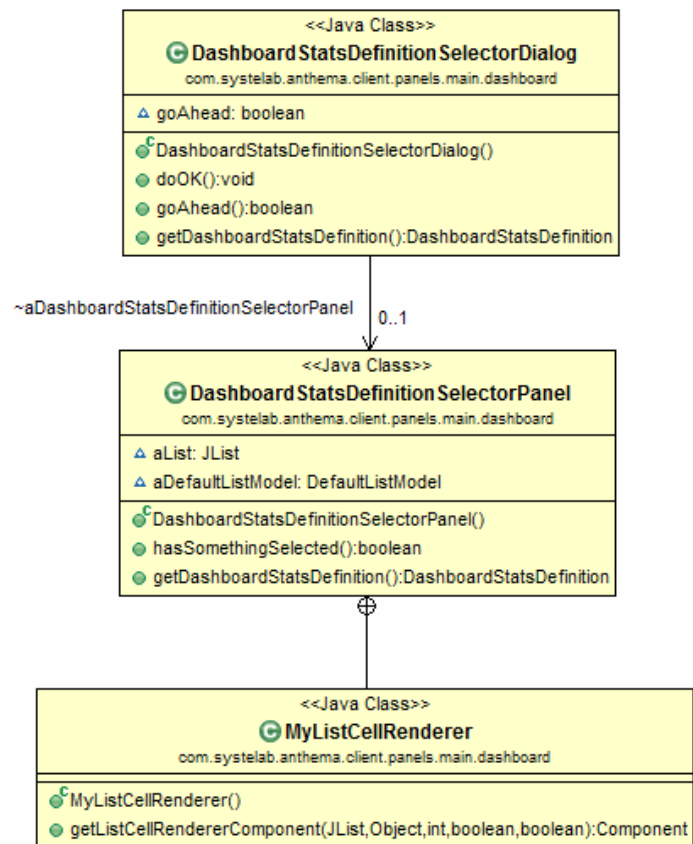


Figura 38: Implementació del diàleg que mostra les gràfiques

Un possible exemple del diàleg seria el següent:

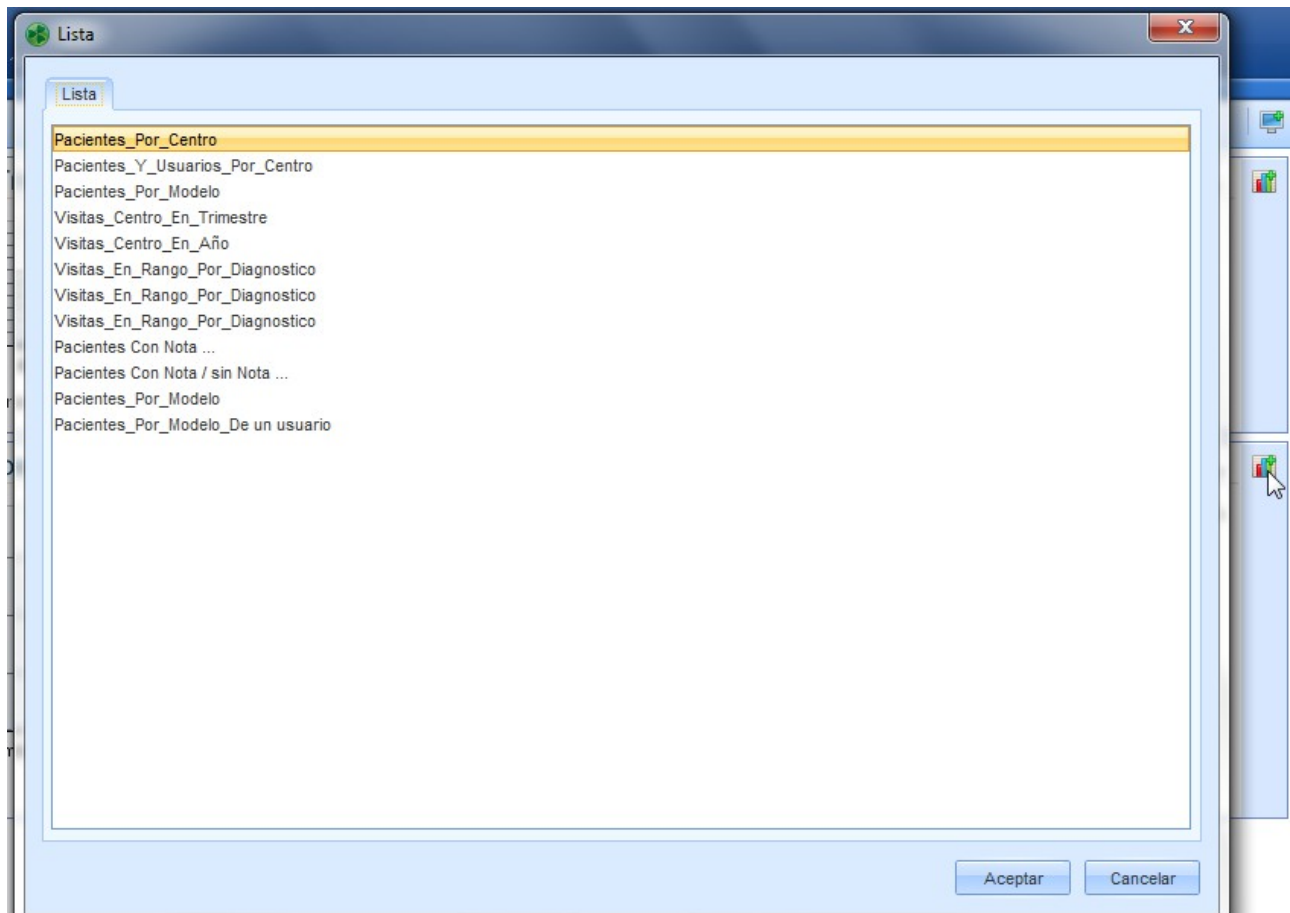


Figura 39: Exemple de diàleg amb les gràfiques disponibles

7.4 Verificació

En aquest apartat s'explica quines proves s'han fet per tal d'assegurar que la implementació és correcta i que es compleixen els objectius que es van definir a l'inici del projecte.

Com s'ha pogut veure al llarg de la memòria, la implementació s'ha basat en dos grans aspectes. En primer lloc, s'han dissenyat uns procediments per tal d'executar uns processos estadístics i també s'han creat unes estructures de dades amb l'objectiu de desar les dades calculades. En segon lloc, s'ha construït una interfície gràfica per tal de mostrar aquests resultats als usuaris. Les proves, per tant, s'han separat tenint en compte els punts anteriorment explicats.

- **Verificació de l'execució dels processos estadístics**

Per provar que l'execució dels procediments funciona correctament s'ha creat una bateria de consultes SQL que s'han llençat contra la base de dades i s'han desat els resultats. Després, s'han configurat les mateixes estadístiques utilitzant les noves estructures de dades però s'han llençat utilitzant el nou sistema (executat pels procediments de la base de dades). Finalment s'han comparat els resultats verificant que són iguals.

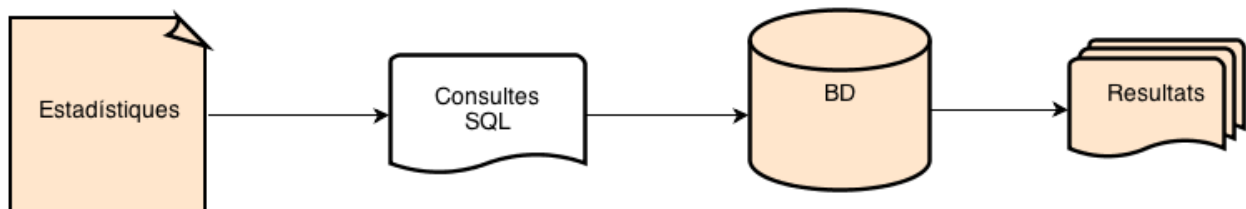
Simplificant obtenim que, en el primer cas, es segueixen aquests passos:

1. Creació de consultes SQL a partir de les estadístiques
2. Execució de les consultes directament a la BD
3. Obtenció dels resultats

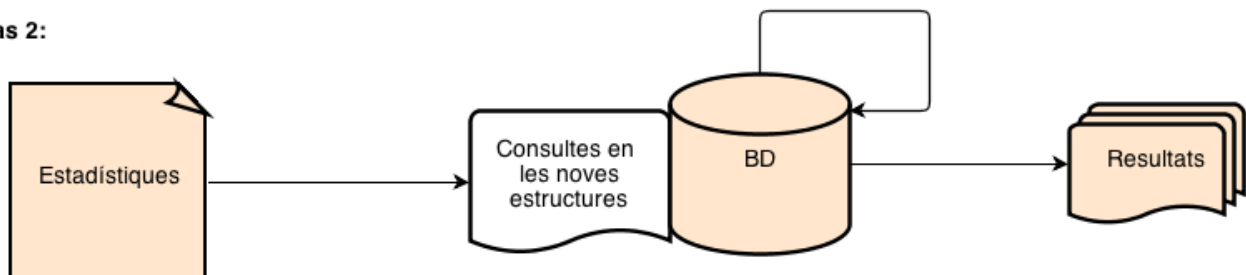
En el segon cas, els passos són:

1. Importació de les estadístiques a les estructures del nou sistema
2. Execució d'aquestes usant el nou procediment
3. Comparativa de resultats i comprovació d'errors

Cas 1:



Cas 2:



- **Verificació de les dades mostrades per pantalla**

Aquest part és la més complexa de verificar, ja que es tracta d'obtenir les dades calculades de la base de dades i mostrar-les per pantalla. En aquest punt s'ha tingut en compte, a part d'obtenir les dades correctes, que aquestes es representessin de manera agradable i funcional a l'usuari, ja que era un dels principals objectius del projecte.

Com a principals criteris que s'han tingut en compte:

- Dades correctes (comparant-les amb les consultes directes a BD)
- Representació correcta de les dades en les gràfiques
- Manteniment de la configuració local de l'usuari
- Colors correctes

A part d'aquestes validacions, també s'ha verificat el compliment dels diferents casos d'ús definits en l'inici del projecte. Per concloure, s'enumeren els casos d'ús i s'explica el resultat obtingut.

Cas d'ús	Resultat
Veure estadístiques	Es pot veure el conjunt d'estadístiques disponibles per usuari
Afegir nova gràfica	El sistema permet afegir una nova gràfica al mòdul.
Eliminar gràfica	També es verifica que es puguin eliminar gràfiques del mòdul.
Redimensionar gràfica	La mida de les gràfiques també pot canviar-se.
Ressituar gràfica	Es permet moure de posició les gràfiques de l'usuari.

La verificació detallada anteriorment ha garantit que es pogués comprovar que s'assoleixen tots els objectius proposats en l'inici del projecte. Tot i això, en el capítol 9 de la memòria, s'expliquen quins eren els objectius i fins a quin punt s'ha arribat en cadascun d'ells. A part, també es fa un anàlisi de les possibles futures millores que podrien desenvolupar-se per donar encara més valor al projecte.

8. Anàlisi econòmic

En aquest punt s'expliquen les principals conclusions pel que fa a l'apartat econòmic del projecte. Es veu quin ha estat el cost inicial previst just abans d'implementar el projecte i l'anàlisi amb el que ha estat el cost final. S'hi poden veure també quines divergències hi ha hagut i quines desviacions han sigut les més significatives.

8.1 Detall del cost inicial

El cost inicial del projecte vindrà condicionat per la quantitat d'hores estimades per cada tasca i també quina serà la participació de cada perfil de treballador. Com que el projecte es basa en el cicle complet d'enginyeria del programari al llarg del procés hi intervenen diferents perfils tècnics. Per exemple, hi apareix la figura del Gestor de projectes en la captura dels requeriments, els analistes tècnics que apareixen en les funcions d'anàlisi i disseny, o també els programadors en la part de la implementació.

Encara que tots els perfils s'hagin dut a terme per una sola persona, si es té en compte que en el mercat, aquests perfils tenen diferents facturacions, també caldrà especificar-ho en l'anàlisi de costos.

Per deixar clar quins perfils es tenen en compte, es mostra la següent taula amb les tasques assignades i el preu per hora aproximat.

Perfil	Tasques	Preu/hora
Cap de projectes	Apareix en la presa de requeriments. En la resta de les etapes ha de fer un seguiment del progrés de les tasques.	50 €/h
Analista funcional	Apareix en totes les etapes excepte la d'implementació.	40 €/h
Analista tècnic	Té el paper més destacat en la presa de requeriments.	40 €/h
Desenvolupador	Les seves tasques es centren en la part d'implementació, detecció d'errors i verificació.	30 €/h

8. 1. 1 Cost del programari i maquinari

El cost de programari i maquinari inicial ha de tenir en compte el valor tant dels equips informàtics com el preu de les llicències de programari que es fan servir per desenvolupar el projecte.

- *Cost inicial de maquinari:*

Equip	Preu
PC IntelCore i5	700 €
Perifèrics	200 €
Servidor de proves	500 €
Total	1.400 €

Tot i això, el preu total de la taula no és realment el total del cost, ja que l'empresa disposa d'un pla de reciclatge d'equips que consisteix en amortitzar-los al cap de tres anys. Per aquest motiu, cal tenir en compte el preu en funció del temps que s'han utilitzat els equips i no sobre el preu total.

El preu dels equips es basa en el seu ús: 8 hores al dia durant 20 dies al mes. Tenint en compte la regla d'amortització en tres anys, s'obté:

$$\text{Preu hora maquinari} = \text{preu maquinari} / \text{hores treballades en 3 anys}$$

Si s'apliquen els valors reals s'obté el següent:

$$\text{Preu hora maquinari} = 1.400 \text{ €} / (3 \text{ anys} * 11 \text{ mesos/any} * 20 \text{ dies/mes} * 8\text{h/dia}) =$$

$$\text{Preu hora maquinari} = 1.400 \text{ €} / 5.280 \text{ h} = \mathbf{0.27 \text{ €/h}}$$

Si tenim en compte el nombre d'hores dedicades en el projecte, es pot concloure que el preu final del maquinari és:

$$\text{Cost total del maquinari} = \mathbf{0.27 \text{ €/h} * 440 \text{ h} = 116 \text{ €}}$$

- Cost inicial del programari:

Producte	Preu
Llicència Oracle Enterprise	2000 €
Llicència Team Foundation Software (TFS)	1500 €
Llicència Monarch Charts	300 €
Total:	3.800 €

Com succeeix amb el cas del maquinari, en els costos de tipus programari també s'amortitzen al cap de tres anys per política d'empresa. Per tant, seguint amb el mateix patró, es calcula el valor del programari:

Preu hora programari = $3.800 \text{ €} / (3 \text{ anys} * 11 \text{ mesos/any} * 20 \text{ dies/mes} * 8\text{h/dia}) =$

Preu hora programari = $3.800 \text{ €} / 5.280 \text{ h} = 0.72 \text{ €/h}$

Si tenim en compte el nombre d'hores dedicades en el projecte, es pot concloure que el preu final del maquinari és:

Cost total del programari = $0.72 \text{ €/h} * 440 \text{ h} = 316 \text{ €}$

Un cop analitzats els costos inicials, tant de programari com del maquinari, es pot obtenir quin és el valor inicial general en aquest aspecte.

Cost inicial maq. i prog. = cost maquinari + cost programari = $116 + 316 = 432 \text{ €}$

El cost inicial estimat per a maquinari i programari és de 432 €.

8. 1. 2 Cost dels recursos humans

El cost dels recursos humans té una relació directa amb les hores que dediquen els perfils a les seves tasques. Tot seguit es detallen les hores que dediquen aquests perfils i el preu total de les seves jornades.

Cal dir que les hores són les estimades inicialment i no les reals. En primer lloc, es mostren les hores totals per cada tasca i a continuació es separen en funció del perfil i

se'n calcula el cost.

Tasca	Jornades	Hores
Definició de requeriments	5	40
Especificació	10	80
Disseny	15	120
Implementació	20	160
Verificació	5	40
Total:	55 jornades	440 hores

Un cop vistes les jornades estimades inicials, es reparteixen les hores de les jornades en funció de les tasques entre els perfils que les executen.

Perfil	Preu/hora	Hores	Cost (€)
Cap de projecte	50 €/h	40	2.000
Analista funcional	40 €/h	80	3.200
Analista tècnic	40 €/h	120	4.800
Desenvolupador	30 €/h	200	6.000
Total:		440 h	16.000 €

El cost dels recursos humans inicial puja fins a 16.000 €.

8. 1. 3 Cost inicial total

Tenint en compte els càlculs fets en els anteriors apartats, el cost inicial total és el següent:

Cost inicial estimat = Cost inicial recursos humans + cost inicial programari i maquinari

Aplicant els valors:

Cost inicial estimat = 16.000 € + 432 € = 16.432 €

En conclusió, el cost inicial estimat és de 16.432 €.

8. 2 Detall del cost final

En aquest punt es pot observar quin ha estat el cost real un cop finalitzat el projecte. Com es podrà veure en el següent capítol (anàlisi de la planificació), hi ha hagut una sèrie de variacions que han impactat clarament en el cost final del projecte. És el moment, doncs, de veure si aquests canvis de planificació han comportat un sobrecost respecte al valor estimat inicialment o si pel contrari, ha impactat positivament i realment el projecte ha sortit més econòmic del que s'esperava.

Les hores finals han variat sensiblement en comparació a les inicials. La següent taula mostra les jornades finals.

Tasca	Jornades	Hores
Definició de requeriments	6	48
Especificació	5	40
Disseny	15	120
Implementació	14	112
Verificació	5	40
Total:	45 jornades	360 hores

El fet de que hi hagi divergències entre les hores estimades i les reals impactarà en el cost final del projecte.

En concret, les hores finals són 360 i les planificades originalment 440, per tant, s'ha reduït el projecte en 80 hores. Cal tenir present que en ambdós casos no intervenen les hores de redacció de la memòria.

8. 2. 1 Cost del programari i maquinari

El cost del maquinari no ha variat respecte al planificat inicialment. Com que l'empresa amortitza els equips al cap de 3 anys, i no hi ha hagut canvis de maquinari imprevistos, es pot mantenir el valor inicial com a cost del programari i maquinari final.

Per tant, el cost final per a maquinari i programari és de 432 €.

8. 2. 2 Cost dels recursos humans

El cost dels recursos humans si que ha patit algun canvi, ja que hi ha hagut una disminució considerable de les hores dedicades al final del projecte.

En la següent taula es veuen les hores finals dedicades en funció del perfil.

Perfil	Preu/hora	Hores	Cost (€)
Cap de projecte	50 €/h	48	2.400
Analista funcional	40 €/h	40	1.600
Analista tècnic	40 €/h	120	4.800
Desenvolupador	30 €/h	152	4.560
Total:		360 h	13.360 €

Tenint en compte que les hores, en alguns perfils, han disminuït, el cost real ha sigut significativament inferior al cost inicialment estimat.

El cost dels recursos humans real un cop acabat el projecte és de 13.360 €.

8. 2. 3 Cost final total

Tal com ha succeït amb el càlcul total inicial, ara es farà de la mateixa manera pel calcular el cost final real, un cop conegudes les desviacions del projecte.

$$\text{Cost final total} = \text{Cost final recursos humans} + \text{cost final programari i maquinari}$$

Aplicant els valors:

$$\text{Cost final total} = 13.360 \text{ €} + 432 \text{ €} = 13.792 \text{ €}$$

En conclusió, el cost final del projecte és de 13.792 €.

8. 3 Comparativa del cost inicial envers el final

Com es pot observar, hi ha una diferència clara entre el cost estimat del projecte amb el cost real. En gran mesura, aquesta desviació té a veure en que, en el moment d'estimar les tasques, en l'inici del projecte, es va pensar que en algunes d'aquestes hi hauria una dedicació en jornades que finalment no hi ha hagut.

Concretament, sense comptar els períodes festius ni tampoc la dedicació a la redacció de la memòria del projecte, es pot concloure que inicialment es va estimar un total de 440 hores, però que finalment s'hi han dedicat 360 hores. Això representa un estalvi de 80 hores. Aquest fet fa que el càlcul inicial estimat fos de 16.432 € i, en canvi, el càlcul final és d'un total de 13.792 €.

Per concloure, cal remarcar que hi ha hagut un estalvi respecte al càlcul inicial de 2.640 €. Aquest fet és important, ja que tot i havent estimat les tasques abans de començar el projecte, hi ha hagut una desviació temporal important. En aquest cas, però, el canvi ha sigut beneficiós, ja que ha comportat un estalvi d'hores en algun dels perfils del projecte.

9. Conclusions

En aquest darrer capítol es mostren les conclusions un cop finalitzat el projecte final de carrera. Es fa un repàs de la que ha sigut la planificació final i es compara amb la que es va planificar inicialment, es descriuen els objectius aconseguits i les possibles futures ampliacions que es podrien fer.

9.1 Objectius aconseguits

Els objectius que es van marcar en l'inici del projecte s'han pogut aconseguir en gran part. El repte era solucionar totes les necessitats que, inicialment van ser proposades per l'empresa, que es van contrastar i detallar amb els clients, i cal dir que s'han assolit. Aquest fet, però, no vol dir que no hi hagi la possibilitat de realitzar futures millores. Més endavant s'enumeraran algunes tasques que es podrien fer per millorar encara més la funcionalitat implementada.

Si es mira enrere, els principals objectius eren:

- Creació i integració d'una nova funcionalitat en un sistema real
- Transformar dades del sistema en informació valuosa pels usuaris
- Creació d'estadístiques per tractar dades del sistema
- Mostrar els resultats de les dades de manera gràfica als usuaris

Tots els objectius s'han pogut aconseguir.

9.2 Planificació final

Els primers passos del PFC van consistir en definir un conjunt de tasques generals a fer i també estimar d'una manera aproximada el temps que caldria dedicar a cadascuna d'elles. Tot seguit es mostra la comparativa entre el temps que es va predir a l'inici del projecte amb el temps dedicat realment en cadascuna de les parts del projecte.

Tasca	Jornades estimades	Jornades reals	Diferència
Definició de requeriments	5	6	+1
Especificació	10	5	-5
Disseny	15	16	+1
Vacances d'estiu	15	15	0
Implementació	20	14	-6
Verificació	5	5	0
Redacció memòria	80	83	+3
Total:	150	144	-6
Total (sense festius):	135	129	-6

Com es pot veure en la taula anterior, les jornades estimades a l'inici del projecte varen ser excessivament pessimistes, ja que finalment el projecte ha finalitzat abans.

9.2.1 Desviacions de temps

Intentar predir el que trigarà un projecte d'enginyeria del programari és realment complicat. Sempre hi ha tasques que es poden allargar i d'altres que es poden resoldre amb menys temps del previst. A continuació s'intentarà explicar on hi ha hagut desviacions i explicar-ne les raons.

Una de les tasques que han reduït els terminis estimats inicialment ha estat l'especificació. Com es pot veure en la taula del capítol anterior, hi ha hagut un decreixement de 5 jornades respecte el temps inicial. Aquest fet ha sigut en gran part produït per una bona feina de definició de requeriments, ja que es van deixar molt clars els objectius i ja es van donar algunes pinzellades de com hauria de ser l'especificació i disseny final del projecte.

Per altra banda, una tasca que sí que ha sobrepassat els temps marcats ha estat la redacció de la memòria del projecte. Al llarg del desenvolupament del projecte s'ha fet feina de redactat de la memòria, sobre tot de les parts inicials com poden ser la definició de requeriments i els casos d'ús del projecte. Tot i això hi ha hagut un increment d'hores pel fet de crear una bona documentació tècnica de les parts de disseny i implementació.

9. 2. 2 Duració real del projecte

Tot resumint el que s'ha dit fins ara, inicialment es va estimar que el projecte tindria una durada aproximada de 135 jornades. Tanmateix, per les raons que ja s'han comentat, el projecte ha pogut reduir el seu temps total a 129 jornades, guanyant 6 dies.

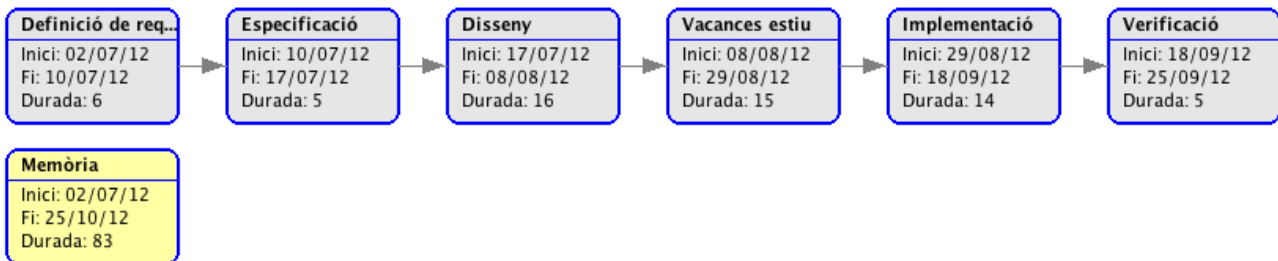
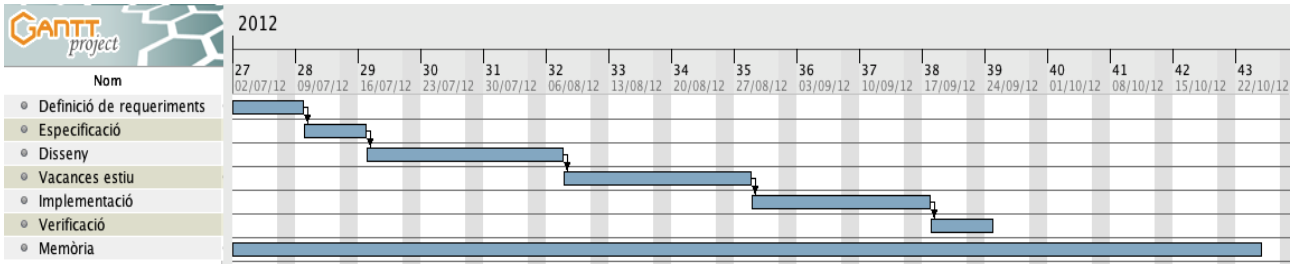


Figura 42: Diagrama PERT. Temps real del projecte

9. 3 Futures ampliacions

Tot i que el projecte ha pogut assolir tots els reptes declarats en l'anàlisi de requeriments, hi ha algunes tasques que es podrien fer en el futur per millorar i oferir encara més funcionalitats al projecte. Per altra banda, aquestes possibles millores no són cabdals ni obligatòries, però millorarien l'experiència d'usuari en gran mesura.

• **Millora de l'aspecte gràfic:**

L'aspecte gràfic de l'aplicació ha millorat en gran mesura després d'afegir-ne la nova funcionalitat al sistema. Tot i això es podrien efectuar les següents fites:

- **Afegir noves gràfiques:** seria bo que en el futur es poguessin afegir noves

tipologies de gràfica per potenciar encara més el motor d'estadístiques. Una possibilitat seria afegir gràfiques en tres dimensions.

- **Afegir més colors als gràfics:** similar al que s'ha dit, també milloraria l'experiència d'usuari oferir aquestes gràfiques amb diferents games de colors, i que fins i tot l'usuari pogués triar-ne de nous.
- **Més informació sobre les gràfiques:** a més a més, també es podria millorar el sistema fent que els usuaris poguessin veure encara més informació de les dades que es mostren en les gràfiques. Per exemple, el conjunt de mostres tractades, altres percentatges, etc.

- **Millora de la integració d'estadístiques al sistema:**

Aquestes millores tenen a veure amb la facilitat d'incloure noves estadístiques als sistemes. Es podria resumir en una nova millora:

- **Procediment visual per afegir noves estadístiques al sistema:** La idea seria implementar un petit programari que, de manera visual, es poguessin incloure en les bases de dades dels sistemes noves estadístiques de manera senzilla. Ara mateix, tal i com està implementat el nou sistema, cal que un tècnic executi unes sentències a la base de dades. Amb un programari es milloraria aquest procés i també seria molt més senzill detectar-ne possibles errors i disposar d'un millor manteniment.

9. 4 Experiència personal

Un cop acabat el projecte i veient el que s'ha aconseguit finalment, no puc evitar sentir-me satisfet de la feina feta. Tot i que el camí no ha sigut fàcil, la recompensa és ben valuosa i gratificant.

Al treballar en un projecte real, i el més important, el fet d'integrar un nou mòdul dins d'un programari que està actualment en el mercat, ha comportat una gran responsabilitat, ja

que qualsevol petit error podria comprometre al projecte i també donar mal nom al producte. Per sort no ha sigut així i per tant estic doblement satisfet.

Durant el procés del projecte final de carrera, he hagut d'exercir diversos perfils segons les tasques que havia de realitzar. Aquest fet ha sigut molt important, ja que he pogut constatar de primera mà quines problemàtiques tenen cadascun d'aquests perfils en la seva feina diària. Una de les més complexes i noves que he pogut observar és el tracte directe amb els clients. Cal tenir molta destresa a l'hora de tractar els clients i intentar extreure el que realment necessiten o volen i saber-ho traslladar posteriorment a uns bons requeriments.

Tot i això, cal dir que fent el PFC he hagut també de, no només desenvolupar el producte, si no també crear una bona documentació. La part tècnica de la memòria ha sigut tot un repte i crec que treballant-hi he après molt.

Deixant de banda els detalls tècnics, no puc deixar de valorar positivament el fet de treballar al costat dels companys de feina que, tot i que el PFC ha recaigut en mi totalment, els hi he pogut demanar consell i m'han donat suport en els moments de més complexitat.

Tot i l'esforç i el desgast que comporta el projecte final de carrera, realment val la pena i em sento orgullós de la feina aconseguida i de tot el que he après al llarg d'aquest trajecte.

10 Annex

En aquest annex es pot trobar informació que no tenia lloc dins de la memòria.

10.1 Manual d'usuari

En aquest apartat es donaran les nocions bàsiques per poder utilitzar la nova funcionalitat de l'aplicació. Cal dir que, el nou mòdul s'ha creat amb la intenció de que sigui fàcil per l'usuari i que hi hagi una corba d'aprenentatge lleugera.

Segons la configuració actual del programari, la pantalla inicial que veurà l'usuari un cop faci *login* a l'aplicació és la nova pantalla d'estadístiques. Per tant, l'usuari en primer lloc haurà de fer *login*:



Figura 43: Pantalla de Login de l'aplicació

Un cop verificades les credencials de l'usuari, aquest accedirà automàticament a la pantalla d'estadístiques on podrà veure els resultats en funció del seu usuari i la seva gestió diària.

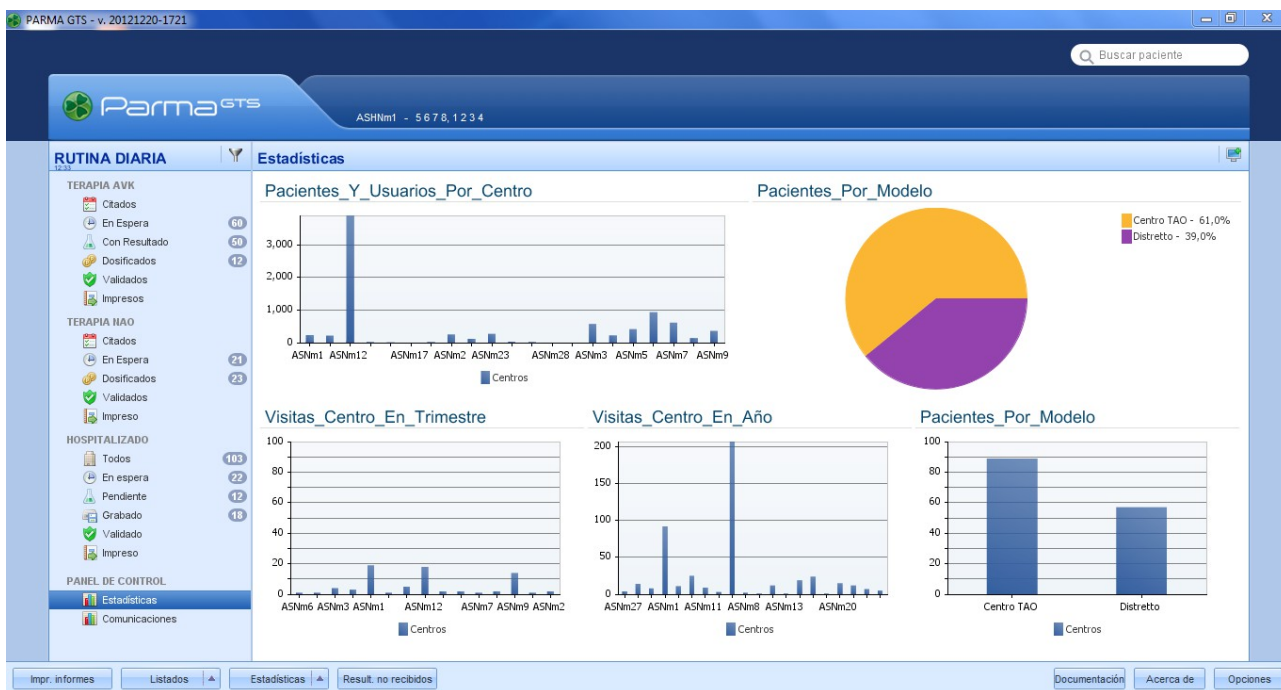



Figura 44: Vista general de la pantalla inicial dels usuaris

Com es pot observar, l'opció de menú de l'esquerra *Estadístiques* està seleccionada automàticament. Si l'usuari comença a treballar en altres parts de l'aplicació, només cal que cliqui a l'opció *d'Estadístiques* per que pugui tornar de nou a la pantalla de gràfics.



Figura 45: Opció de menú Estadístiques

En aquest moment l'usuari ja pot configurar-se la pantalla al seu gust. Per fer-ho, ha de clicar sobre el botó *editar* que apareix a la part superior dreta de la finestra . Clicant aquest botó apareixen les opcions de configuració disponibles:

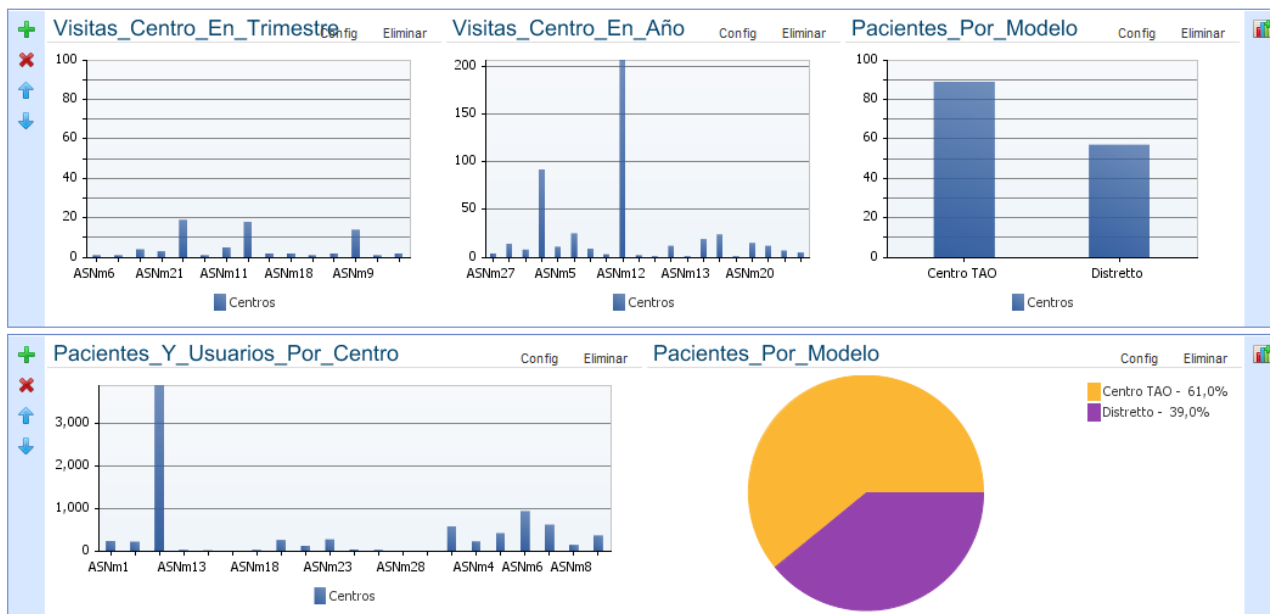






Figura 46: Pantalla general en mode configuració

Com es pot apreciar, la finestra queda dividida en dues *línies*, que cadascuna conté una sèrie de gràfiques i que totes dues també disposen d'una sèrie d'opcions. Els gràfics que hi apareixen també tenen les seves opcions de configurar o eliminar.

La imatge mostra una sèrie d'icones en la part esquerra de cada línia que el que fan és:

- Afegir una nova línia buida de gràfiques 
- Eliminar la línia de gràfiques 
- Moure la línia cap a dalt 
- Moure la línia cap a baix 

Per exemple, si s'afegeix una línia de gràfics quedaria com la següent imatge.

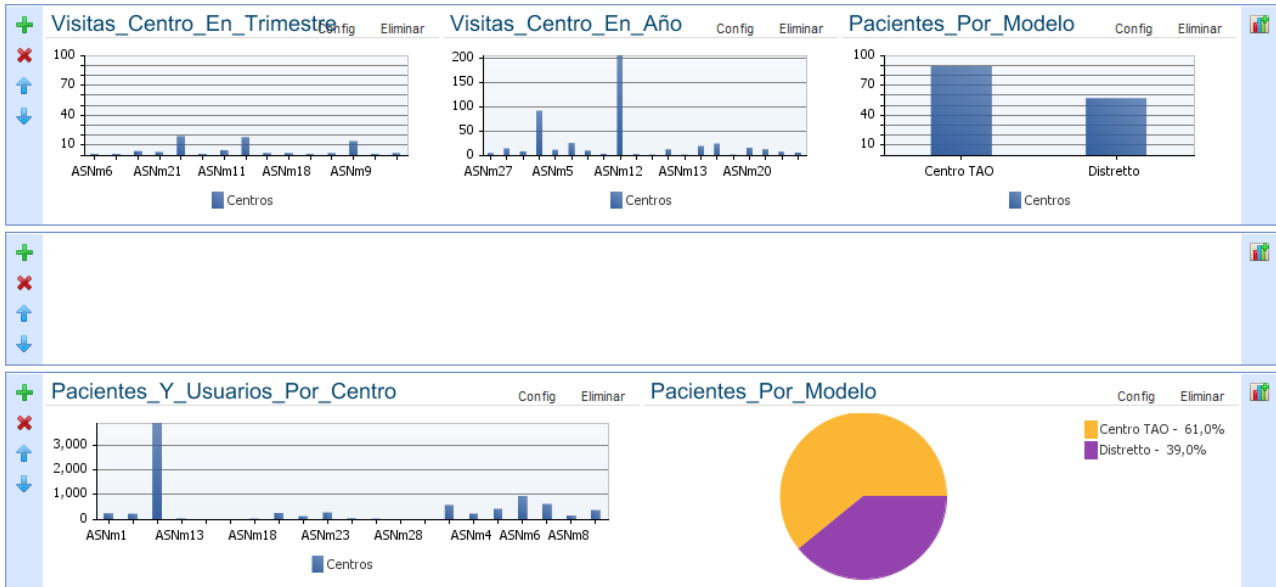


Figura 47: Afegir una línia nova

A part d'afegir-ne, també es poden redimensionar. Si es posa el cursor entre les ratlles que separen les línies de gràfics, es permet canviar la mida d'aquestes línies i per tant, també la mida dels gràfics.

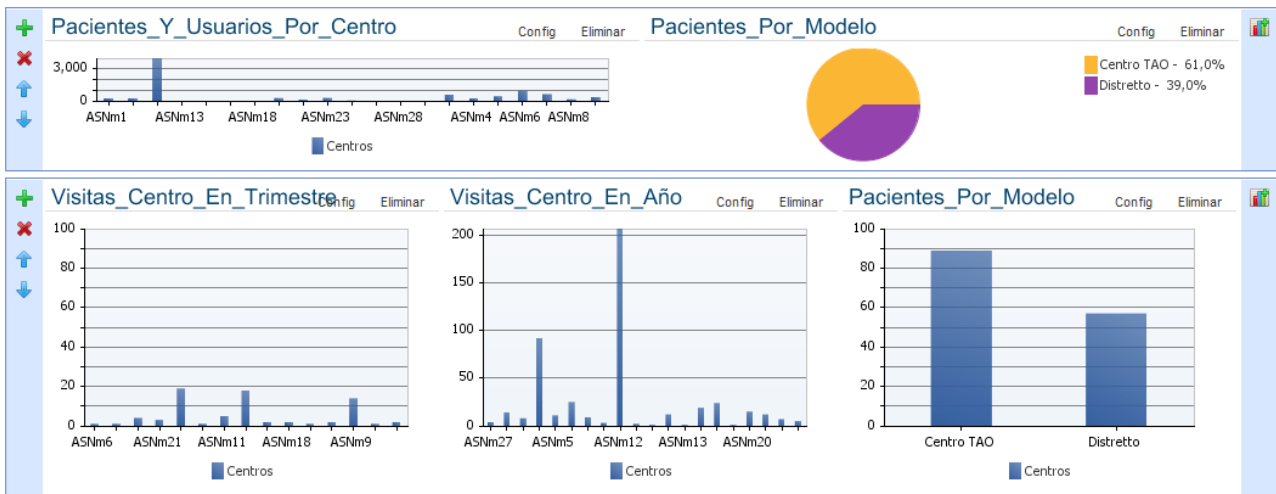


Figura 48: Canviant la mida dels gràfics

Però això no és tot. Com s'ha explicat al llarg d'aquesta memòria, també es poden afegir noves gràfiques. L'usuari ho pot fer utilitzant el botó que apareix al costat dret de cada línia.



Clicant sobre la icona s'obre un nou diàleg d'usuari que mostra un llistat amb les possibles estadístiques que té al seu abast.

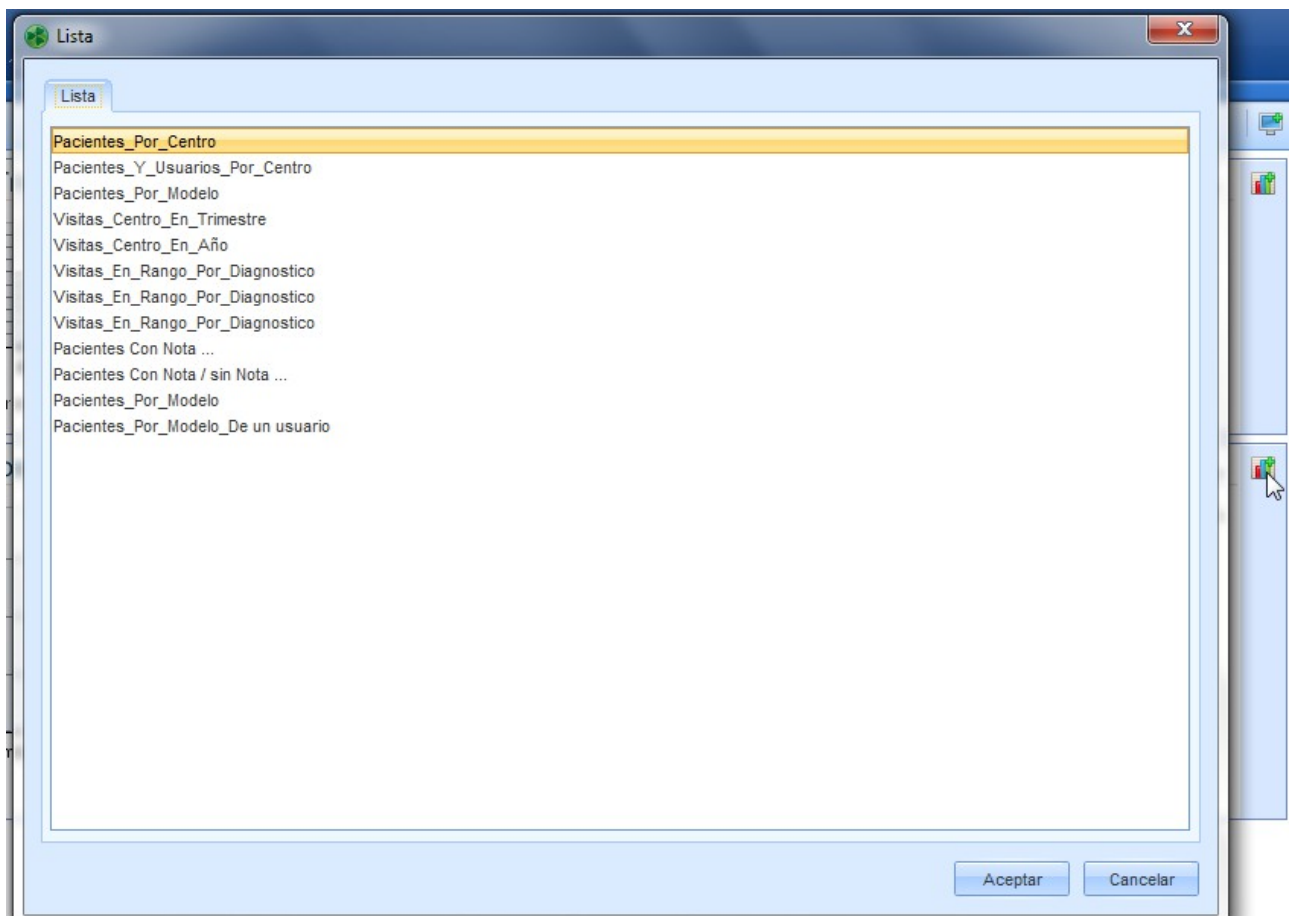


Figura 49: Diàleg d'usuari mostrant les estadístiques disponibles

Si l'usuari selecciona una estadística, s'afegirà en la línia corresponent. En el cas de l'exemple anterior, s'afegirà en la segona línia. (veure el cursor situat en el botó de la segona línia).

Els gràfics també tenen les seves opcions de configuració:

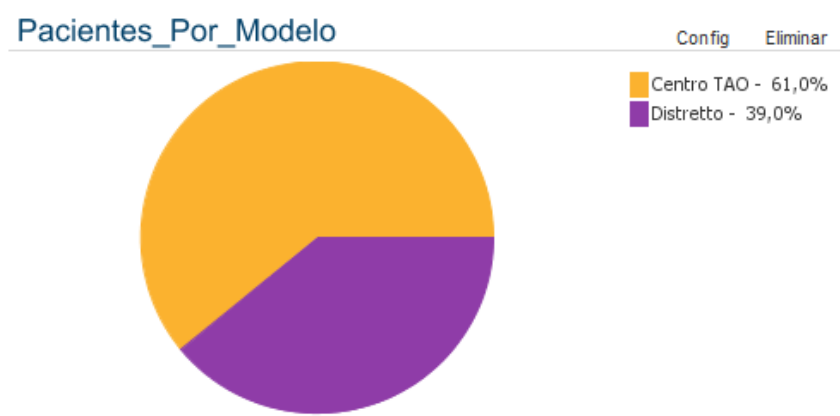



Figura 50: Exemple de gràfic amb opcions

Com es veu en la imatge anterior, el gràfic té un parell d'opcions en la part superior. El botó *Eliminar* treu la gràfica de la línia, i l'opció *Config* permet obrir el diàleg amb la llista d'estadístiques a mostrar. Si l'usuari en seleccionés una altra, la gràfica actual desapareixeria i es mostraria la nova.

Finalment, quan l'usuari està satisfet amb la seva configuració cal que torni a prémer el botó inicial. 

D'aquesta manera les opcions de configuració desapareixen i l'usuari ja pot seguir treballant.

10. 2 Eines emprades per a la redacció de la memòria

Per la redacció de la memòria del PFC s'han utilitzat les següents eines de programari:

- **Paquet LibreOffice:** Per la creació dels documents.
- **Diagram.ly** (Extensió de Chrome): Creació d'alguns diagrames i figures.
- **SQL Developer:** Observar les taules de la base de dades.
- **MagicDraw:** Creació de figures d'enginyeria del programari.
- **GanttProject:** Per la confecció dels diagrames de Gantt de la planificació.

10. 3 Creació del paquet de procediments SQL

Tot seguit es mostra el codi font dels procediments SQL que executen les estadístiques periòdicament.

```

create or replace
PACKAGE New_Stats
AS
  procedure Execute_Statistic (Statistic_Id in Number);
  procedure Execute_All_Statistics;
  function Immediate_For_User(Statistic_Id In Number, Stat_user_id In Number) return Number;
  function All_Immediate_For_User(Stats_user_id In Number) return varchar2;
End New_Stats;
/

create or replace
PACKAGE BODY New_Stats
AS
  -- Procedure to execute the statistic
  procedure Execute_Statistic (Statistic_Id in Number)
  is
    Stats_Def Stats_Definition%Rowtype;
    --Stats_Res_Columns Stats_Res_Def%Rowtype;
    Exists_Stat Number(4);
    Execution_Id Number(19,0);
    Execution_Parameter_Id Number(19,0);
    Execution_Result_Id Number (19,0);
    Calculated_by Stats_Definition.Stats_Calculated_By%Type;
    sentence Stats_Definition.Stats_Script%Type;
    script_Parameters_Substituted varchar2(4000);
    Param_Evaluated varchar2(100);
    Param_To_Replace varchar2 (20);
    User_Role_Execution varchar2(30);

    type Object_ID is record (Obj_Id Number(19,0));
    type Objects_ID is table of Object_ID;
    Objects_ID_Res Objects_ID;

  begin
    select count(*) into Exists_Stat from Stats_Definition where Stats_Id = Statistic_Id;
    if Exists_Stat > 0 then
      -- Get the definition of the statistic
      select * into Stats_Def from Stats_Definition where Stats_Id = Statistic_Id;
      -- dbms_output.put_line('Exists');

      -- Verify the calculated by field in order to do an execution for each item found. In case that Calculated
      by is different from installation, the first parameter of the script should
      -- be the identifier of the element to be used for calculation. This is an extra parameter considered as
      parameter 0 and is not in the list of parameters definition

      Calculated_by := Stats_Def.Stats_Calculated_By;
      -- dbms_output.put_line('Calculated By: ' || Calculated_by);

      if upper(Calculated_by) = 'INSTALLATION' then
        -- Get the identifiers of the execution
        select Stats_Exec_Id_Seq.nextval into Execution_Id from dual;
        select Stats_Exec_Param_Id_Seq.nextval into Execution_Parameter_Id from dual;
        select Stats_Exec_Res_Id_Seq.nextval into Execution_Result_Id from dual;

        -- Create the record of the stat execution
        insert into Stats_Execution (Stats_Exec_Id, Stats_Id, Stats_Object_Type, Stats_Object_Id,

```

```

Stats_Date, Stats_Start_Timestamp, Stats_End_Timestamp, Stats_Exec_Params_Id,
Stats_Exec_Result_Id)
  select Execution_Id, Stats_Def.Stats_Id, Calculated_by, null, trunc(sysdate), systimestamp, null,
  case
    when Stats_Def.Stats_Param_Def_Id is null then null
    else Execution_Parameter_Id
  end, Execution_Result_Id from dual;

  script_Parameters_Substituted := Stats_Def.Stats_script;
  -- Copy the parameters of the execution if there are parameters
  if Stats_Def.Stats_Param_Def_Id is not null then

    for Stats_Params_In in (select * from Stats_Params_Def where Stats_Param_Id =
Stats_Def.Stats_Param_Def_Id order by Stats_Param_Position)
    loop
      -- Evaluate the param
      sentence := 'select ' || Stats_Params_In.Stats_Param_Value || ' from dual';
      execute immediate sentence into Param_Evaluated;

      sentence := 'insert into Stats_Exec_Params (Stats_Exec_Param_Id, Stats_Exec_Param_Name,
Stats_Exec_Param_Position,Stats_Exec_Param_Value) select ' || Execution_Parameter_Id ||
      ', ' || Stats_Params_In.Stats_Param_Name || ', ' || Stats_Params_In.Stats_Param_Position || ', ' ||
|| Param_Evaluated || ' from dual';
      -- dbms_output.put_line('Sentence insert: ' || sentence );
      execute immediate sentence;

      select '%param_' || Stats_Params_In.Stats_Param_Position || '%' into Param_To_Replace from
dual;

      select replace(script_Parameters_Substituted, Param_To_Replace, Param_Evaluated) into
script_Parameters_Substituted from dual;

    end loop;
  end if;

  -- dbms_output.put_line('Script substituted: ' || script_Parameters_Substituted);

  -- Drop the temp table if it exists
  upgradefunctions.Drop_Table ('Temp_Stat_Result_' || Stats_Def.Stats_Id);

  -- Execute the statistic script
  sentence := 'create table Temp_Stat_Result_' || Stats_Def.Stats_Id || ' as ' ||
script_Parameters_Substituted;
  -- dbms_output.put_line('Sentence: ' || sentence );
  execute immediate sentence;

  -- Get the results of the temp table and copy them to the exec_results table
  for Stats_Res_Columns in (select * from Stats_Res_Def where Stats_Res_Def_Id =
Stats_Def.Stats_Result_Def_Id order by Stats_Res_Position)
  loop
    sentence := 'insert into Stats_Exec_Results select ' || Execution_Result_Id || ', ' ||
Stats_Res_Columns.Stats_Res_Name || ', ' || Stats_Res_Columns.Stats_Res_Name || ', Serie from
Temp_Stat_Result_' || Stats_Def.Stats_Id;
    -- dbms_output.put_line('Sentence insert: ' || sentence );
    execute immediate sentence;
  end loop;

  -- Drop the temp table if it exists

```

```

upgradefunctions.Drop_Table ('Temp_Stat_Result_' || Stats_Def.Stats_Id);

-- Update the record of the stat execution
update Stats_Execution set Stats_End_Timestamp = systimestamp where Stats_Exec_Id =
Execution_Id;
else

-- Verify if the statistic should be executed by center
if upper(Calculated_by) = 'CENTER' then
-- Create an execution for each center in the system

sentence := 'select ac_site_id from ac_site';
if (Stats_Def.Stats_Object_id_In is not null) then
sentence := sentence || ' where ac_site_id in (' || Stats_Def.Stats_Object_id_In || ')';
end if;

end if;

-- Verify if the statistic should be executed by User
if upper(Calculated_by) = 'USER' then

sentence := 'select user_id from app_user where user_patient_id is null ';

if (Stats_Def.Stats_User_Role is not null) then
sentence := sentence || ' and user_role = ' || to_char(Stats_Def.Stats_User_Role) || ";
end if;
if (Stats_Def.Stats_Object_id_In is not null) then
sentence := sentence || ' and user_id in (' || Stats_Def.Stats_Object_id_In || ')';
end if;

end if;

execute immediate sentence bulk collect into Objects_ID_Res;

for i in Objects_ID_Res.First..Objects_ID_Res.LAST
loop
-- Get the identifiers of the execution
select Stats_Exec_Id_Seq.nextval into Execution_Id from dual;
select Stats_Exec_Param_Id_Seq.nextval into Execution_Parameter_Id from dual;
select Stats_Exec_Res_Id_Seq.nextval into Execution_Result_Id from dual;

-- Create the record of the stat execution
insert into Stats_Execution (Stats_Exec_Id, Stats_Id, Stats_Object_Type, Stats_Object_Id,
Stats_Date, Stats_Start_Timestamp, Stats_End_Timestamp, Stats_Exec_Params_Id,
Stats_Exec_Result_Id)
select Execution_Id, Stats_Def.Stats_Id, Calculated_by, Objects_ID_Res(i).Obj_Id, trunc(sysdate),
systimestamp, null,
case
when Stats_Def.Stats_Param_Def_Id is null then null
else Execution_Parameter_Id
end, Execution_Result_Id from dual;

select replace(Stats_Def.Stats_script, '%param_0%', Objects_ID_Res(i).Obj_Id) into
script_Parameters_Substituted from dual;

-- Copy the parameters of the execution if there are parameters
if Stats_Def.Stats_Param_Def_Id is not null then
for Stats_Params_In in (select * from Stats_Params_Def where Stats_Param_Id =

```

```

Stats_Def.Stats_Param_Def_Id order by Stats_Param_Position)
  loop
    -- Evaluate the param
    sentence := 'select ' || Stats_Params_In.Stats_Param_Value || ' from dual';
    execute immediate sentence into Param_Evaluated;

    sentence := 'insert into Stats_Exec_Params (Stats_Exec_Param_Id, Stats_Exec_Param_Name,
Stats_Exec_Param_Position,Stats_Exec_Param_Value) select ' || Execution_Parameter_Id ||
    ', ' || Stats_Params_In.Stats_Param_Name || ', ' || Stats_Params_In.Stats_Param_Position || ', ' ||
|| Param_Evaluated || ' from dual';
    -- dbms_output.put_line('Sentence insert: ' || sentence );
    execute immediate sentence;

    select '%param_' || Stats_Params_In.Stats_Param_Position || '%' into Param_To_Replace from
dual;
    select replace(script_Parameters_Substituted, Param_To_Replace, Param_Evaluated) into
script_Parameters_Substituted from dual;

    end loop;
  end if;

-- dbms_output.put_line('Script substituted: ' || script_Parameters_Substituted);

-- Drop the temp table if it exists
upgradefunctions.Drop_Table ('Temp_Stat_Result_' || Stats_Def.Stats_Id);

-- Execute the statistic script
sentence := 'create table Temp_Stat_Result_' || Stats_Def.Stats_Id || ' as ' ||
script_Parameters_Substituted;
--dbms_output.put_line('Sentence: ' || sentence );
execute immediate sentence;

-- Get the results of the temp table and copy them to the exec_results table
for Stats_Res_Columns in (select * from Stats_Res_Def where Stats_Res_Def_Id =
Stats_Def.Stats_Result_Def_Id order by Stats_Res_Position)
  loop
    sentence := 'insert into Stats_Exec_Results select ' || Execution_Result_Id || ', ' ||
Stats_Res_Columns.Stats_Res_Name || ', ' || Stats_Res_Columns.Stats_Res_Name || ', Serie from
Temp_Stat_Result_' || Stats_Def.Stats_Id;
    --dbms_output.put_line('Sentence insert: ' || sentence );
    execute immediate sentence;
  end loop;

-- Drop the temp table if it exists
upgradefunctions.Drop_Table ('Temp_Stat_Result_' || Stats_Def.Stats_Id);

-- Update the record of the stat execution
update Stats_Execution set Stats_End_Timestamp = systimestamp where Stats_Exec_Id =
Execution_Id;
  end loop;
end if;
else
  dbms_output.put_line('Not present');
end if;
end;

-- Execute all the statistics in the system
procedure Execute_All_Statistics

```

```

is
  sentence varchar2 (100);
  Execution_Id Number(19,0);
begin
  select Stats_Exec_Id_Seq.nextval into Execution_Id from dual;
  insert into Stats_Execution (Stats_Exec_Id, Stats_Id, Stats_Object_Type, Stats_Object_Id, Stats_Date,
Stats_Start_Timestamp, Stats_End_Timestamp, Stats_Exec_Params_Id, Stats_Exec_Result_Id)
  select Execution_Id, null, 'All', null, trunc(sysdate), systimestamp, null, null, null from dual;

  for Statistics_Id in (select stats_id from Stats_Definition)
  loop

    sentence := 'call New_Stats.Execute_Statistic (' || Statistics_Id.Stats_id || ')';
    -- dbms_output.put_line('Sentence: ' || sentence );

    execute immediate sentence;

  end loop;

  update Stats_Execution set Stats_End_Timestamp = systimestamp where Stats_Exec_Id =
Execution_Id;
end;

-- Execute an immediate statistic for a user
function Immediate_For_User(Statistic_Id In Number, Stat_User_id In Number) return Number is
  Stats_Def Stats_Definition%Rowtype;
  --Stats_Res_Columns Stats_Res_Def%Rowtype;
  Exists_Stat Number(4);
  Execution_Id Number(19,0);
  Execution_Parameter_Id Number(19,0);
  Execution_Result_Id Number (19,0);
  Calculated_by Stats_Definition.Stats_Calculated_By%Type;
  sentence Stats_Definition.Stats_Script%Type;
  script_Parameters_Substituted varchar2(4000);
  Param_Evaluated varchar2(100);
  Param_To_Replace varchar2 (20);
  User_Role_Execution varchar2(30);

  type Object_ID is record (Obj_Id Number(19,0));
  type Objects_ID is table of Object_ID;
  Objects_ID_Res Objects_ID;

begin
  select count(*) into Exists_Stat from Stats_Definition where Stats_Id = Statistic_Id and
Stats_In_Real_Time = '1';
  if Exists_Stat > 0 then
    -- Get the definition of the statistic
    select * into Stats_Def from Stats_Definition where Stats_Id = Statistic_Id;
    -- dbms_output.put_line('Exists');

    -- Verify the calculated by field in order to do an execution for each item found. In case that Calculated
by is different from installation, the first parameter of the script should
    -- be the identifier of the element to be used for calculation. This is an extra parameter considered as
parameter 0 and is not in the list of parameters definition

    Calculated_by := Stats_Def.Stats_Calculated_By;
    -- dbms_output.put_line('Calculated By: ' || Calculated_by);

```

```

if upper(Calculated_by) = 'INSTALLATION' then
  -- Get the identifiers of the execution
  select Stats_Exec_Id_Seq.nextval into Execution_Id from dual;
  select Stats_Exec_Param_Id_Seq.nextval into Execution_Parameter_Id from dual;
  select Stats_Exec_Res_Id_Seq.nextval into Execution_Result_Id from dual;

  -- Create the record of the stat execution
  insert into Stats_Execution (Stats_Exec_Id, Stats_Id, Stats_Object_Type, Stats_Object_Id,
Stats_Date, Stats_Start_Timestamp, Stats_End_Timestamp, Stats_Exec_Params_Id,
Stats_Exec_Result_Id)
  select Execution_Id, Stats_Def.Stats_Id, Calculated_by, null, trunc(sysdate), systimestamp, null,
  case
    when Stats_Def.Stats_Param_Def_Id is null then null
    else Execution_Parameter_Id
  end, Execution_Result_Id from dual;

  script_Parameters_Substituted := Stats_Def.Stats_script;
  -- Copy the parameters of the execution if there are parameters
  if Stats_Def.Stats_Param_Def_Id is not null then

    for Stats_Params_In in (select * from Stats_Params_Def where Stats_Param_Id =
Stats_Def.Stats_Param_Def_Id order by Stats_Param_Position)
    loop
      -- Evaluate the param
      sentence := 'select ' || Stats_Params_In.Stats_Param_Value || ' from dual';
      execute immediate sentence into Param_Evaluated;

      sentence := 'insert into Stats_Exec_Params (Stats_Exec_Param_Id, Stats_Exec_Param_Name,
Stats_Exec_Param_Position,Stats_Exec_Param_Value) select ' || Execution_Parameter_Id ||
      ', ' || Stats_Params_In.Stats_Param_Name || ', ' || Stats_Params_In.Stats_Param_Position || ', ' ||
      Param_Evaluated || ' from dual';
      -- dbms_output.put_line('Sentence insert: ' || sentence );
      execute immediate sentence;

      select '%param_' || Stats_Params_In.Stats_Param_Position || '%' into Param_To_Replace from
dual;

      select replace(script_Parameters_Substituted, Param_To_Replace, Param_Evaluated) into
script_Parameters_Substituted from dual;

    end loop;
  end if;

  -- dbms_output.put_line('Script substituted: ' || script_Parameters_Substituted);

  -- Drop the temp table if it exists
  upgradefunctions.Drop_Table ('Temp_Stat_Result_' || Stats_Def.Stats_Id);

  -- Execute the statistic script
  sentence := 'create table Temp_Stat_Result_' || Stats_Def.Stats_Id || ' as ' ||
script_Parameters_Substituted;
  -- dbms_output.put_line('Sentence: ' || sentence );
  execute immediate sentence;

  -- Get the results of the temp table and copy them to the exec_results table
  for Stats_Res_Columns in (select * from Stats_Res_Def where Stats_Res_Def_Id =
Stats_Def.Stats_Result_Def_Id order by Stats_Res_Position)

```

```

loop
    sentence := 'insert into Stats_Exec_Results select ' || Execution_Result_Id || ', ' ||
Stats_Res_Columns.Stats_Res_Name || ', ' || Stats_Res_Columns.Stats_Res_Name || ', Serie from
Temp_Stat_Result_' || Stats_Def.Stats_Id;
    -- dbms_output.put_line('Sentence insert: ' || sentence);
    execute immediate sentence;
end loop;

-- Drop the temp table if it exists
upgradefunctions.Drop_Table ('Temp_Stat_Result_' || Stats_Def.Stats_Id);

-- Update the record of the stat execution
update Stats_Execution set Stats_End_Timestamp = systimestamp where Stats_Exec_Id =
Execution_Id;
else
    -- Verify if the statistic should be executed by center
    if upper(Calculated_by) = 'CENTER' then
        -- Create an execution for each center in the system

        sentence := 'select ac_site_id from ac_site where ac_site_id in (select user_ac_site from app_user
where user_id = ' || to_char(Stat_User_id) || ')';
        if (Stats_Def.Stats_Object_Id_In is not null) then
            sentence := sentence || ' and ac_site_id in (' || Stats_Def.Stats_Object_Id_In || ')';
        end if;

    end if;

    -- Verify if the statistic should be executed by User
    if upper(Calculated_by) = 'USER' then

        sentence := 'select user_id from app_user where user_patient_id is null and user_id = ' ||
to_char(Stat_User_id) ;

        if (Stats_Def.Stats_User_Role is not null) then
            sentence := sentence || ' and user_role = ' || to_char(Stats_Def.Stats_User_Role) || ";
        end if;
        if (Stats_Def.Stats_Object_Id_In is not null) then
            sentence := sentence || ' and user_id in (' || Stats_Def.Stats_Object_Id_In || ')';
        end if;

    end if;

    execute immediate sentence bulk collect into Objects_ID_Res;

    for i in Objects_ID_Res.First..Objects_ID_Res.LAST
    loop
        -- Get the identifiers of the execution
        select Stats_Exec_Id_Seq.nextval into Execution_Id from dual;
        select Stats_Exec_Param_Id_Seq.nextval into Execution_Parameter_Id from dual;
        select Stats_Exec_Res_Id_Seq.nextval into Execution_Result_Id from dual;

        -- Create the record of the stat execution
        insert into Stats_Execution (Stats_Exec_Id, Stats_Id, Stats_Object_Type, Stats_Object_Id,
Stats_Date, Stats_Start_Timestamp, Stats_End_Timestamp, Stats_Exec_Params_Id,
Stats_Exec_Result_Id)
        select Execution_Id, Stats_Def.Stats_Id, Calculated_by, Objects_ID_Res(i).Obj_Id, trunc(sysdate),
systimestamp, null,

```

```

case
  when Stats_Def.Stats_Param_Def_Id is null then null
  else Execution_Parameter_Id
end, Execution_Result_Id from dual;

select replace(Stats_Def.Stats_script, '%param_0%', Objects_ID_Res(i).Obj_Id) into
script_Parameters_Substituted from dual;

-- Copy the parameters of the execution if there are parameters
if Stats_Def.Stats_Param_Def_Id is not null then
  for Stats_Params_In in (select * from Stats_Params_Def where Stats_Param_Id =
Stats_Def.Stats_Param_Def_Id order by Stats_Param_Position)
  loop
    -- Evaluate the param
    sentence := 'select ' || Stats_Params_In.Stats_Param_Value || ' from dual';
    execute immediate sentence into Param_Evaluated;

    sentence := 'insert into Stats_Exec_Params (Stats_Exec_Param_Id, Stats_Exec_Param_Name,
Stats_Exec_Param_Position, Stats_Exec_Param_Value) select ' || Execution_Parameter_Id ||
', ' || Stats_Params_In.Stats_Param_Name || ', ' || Stats_Params_In.Stats_Param_Position || ', ' ||
Param_Evaluated || ' from dual';
    -- dbms_output.put_line('Sentence insert: ' || sentence);
    execute immediate sentence;

    select '%param_' || Stats_Params_In.Stats_Param_Position || '%' into Param_To_Replace from
dual;
    select replace(script_Parameters_Substituted, Param_To_Replace, Param_Evaluated) into
script_Parameters_Substituted from dual;

  end loop;
end if;

-- dbms_output.put_line('Script substituted: ' || script_Parameters_Substituted);

-- Drop the temp table if it exists
upgradefunctions.Drop_Table ('Temp_Stat_Result_' || Stats_Def.Stats_Id);

-- Execute the statistic script
sentence := 'create table Temp_Stat_Result_' || Stats_Def.Stats_Id || ' as ' ||
script_Parameters_Substituted;
--dbms_output.put_line('Sentence: ' || sentence);
execute immediate sentence;

-- Get the results of the temp table and copy them to the exec_results table
for Stats_Res_Columns in (select * from Stats_Res_Def where Stats_Res_Def_Id =
Stats_Def.Stats_Result_Def_Id order by Stats_Res_Position)
  loop
    sentence := 'insert into Stats_Exec_Results select ' || Execution_Result_Id || ', ' ||
Stats_Res_Columns.Stats_Res_Name || ', ' || Stats_Res_Columns.Stats_Res_Name || ', Serie from
Temp_Stat_Result_' || Stats_Def.Stats_Id;
    --dbms_output.put_line('Sentence insert: ' || sentence);
    execute immediate sentence;
  end loop;

-- Drop the temp table if it exists
upgradefunctions.Drop_Table ('Temp_Stat_Result_' || Stats_Def.Stats_Id);

-- Update the record of the stat execution

```

```

        update Stats_Execution set Stats_End_Timestamp = systimestamp where Stats_Exec_Id =
Execution_Id;
    end loop;
    end if;
else
    dbms_output.put_line('Not present');
end if;

    return Execution_Id ;
end Immediate_For_User;

-- Execute all immediate statistics for a user
-- Execute all immediate statistics for a user
function All_Immediate_For_User(Stats_user_id In Number) return varchar2 is
    sentence varchar2 (500);
    Execution_Id Number(19,0);
    Result_Range varchar2 (50);
begin
    select Stats_Exec_Id_Seq.nextval into Execution_Id from dual;
    insert into Stats_Execution (Stats_Exec_Id, Stats_Id, Stats_Object_Type, Stats_Object_Id, Stats_Date,
Stats_Start_Timestamp, Stats_End_Timestamp, Stats_Exec_Params_Id, Stats_Exec_Result_Id)
    select Execution_Id, null, 'All_Immediate', Stats_User_Id, trunc(sysdate), systimestamp, null, null, null
from dual;

    for Statistics_Id in (select stats_id from Stats_Definition where Stats_In_Real_Time = '1')
    loop
--        sentence := 'call New_Stats.Immediate_For_User(' || Statistics_Id.Stats_id || ', ' || Stats_User_id || ')';
        sentence := 'declare
            Stat_Execution_ID number;
            begin
                Stat_Execution_ID := New_Stats.Immediate_For_User (' || Statistics_Id.Stats_id || ', ' ||
Stats_User_id || ') ;
            end;';

--        dbms_output.put_line('Sentence: ' || sentence );

        execute immediate sentence ;

    end loop;

    Result_Range := 'Results from: ' || Execution_Id;

    update Stats_Execution set Stats_End_Timestamp = systimestamp where Stats_Exec_Id =
Execution_Id;

    return Result_Range;
end All_Immediate_For_User;

End New_Stats;
/

```