A l'etapa de disseny s'escullen les tecnologies per implantar el sistema, es dissenya lògica i físicament la capa de gestió de dades, es determina la capa del domini i es detallen tots els mòduls de la interfície d'usuari.

5.1- Selecció de la tecnologia

En aquest sistema la capa de presentació de dades està basada en HTML (HyperText Markup Language). L' HTML és el llenguatge que fan servir els diferents navegadors que actualment hi ha al mercat per tal de presentar les interfícies d'usuari de les pàgines web.

La capa del domini està encapsulada en pàgines PHP (PHP: HyperText PreProcessor). Es tracta d'un llenguatge de programació interpretat, lliure i força popular, utilitzat per generar contingut dinàmic al web. Les pàgines s'executen en el servidor web on hi ha el sistema d'informació que desenvolupem. La programació d'aquestes pàgines es pot fer en qualsevol llenguatge Script, un dels més utilitzat i que interpreten més navegadors és el JavaScript. Tot i que no permet tota l'extensa gamma de possibilitats que permet el llenguatge en el que està basat (Java), en tindrem suficient per executar les regles de negoci que necessitem.

L'accés a dades es basa en una capa d'accés al SGBD (Sistema de Gestió de la Base de Dades), on a partir d'un llenguatge intern es crearan sentències SQL. S'ha escollit com a SGBD relacional MySQL, és un programari multi-fil, multi-usuari i usa el llenguatge SQL. MySQL ha esdevingut molt popular gràcies a la seva velocitat en executar consultes i el seu suport de forma nativa per part del llenguatge PHP, en l'elaboració d'aplicacions web i en l'entorn del programari lliure.

Finalment, només esmentar que en la capa de presentació la majoria de pàgines tindran un format similar, de manera que pugui ser reutilitzable o, si més no, requereixin poc esforç de modificació. Per aquest motiu s'han usat les plantilles que ofereix el programa d'edició de pàgines web Adobe Dreamweaver.

5.2- Disseny de la capa de gestió de dades

La capa de gestió de dades sap on i com estan emmagatzemades les dades, però ignora com tractar-les. Es relaciona amb la capa de domini passant-li respostes i resultats, i rebent-ne les operacions de consulta i modificació de dades. S'ocupa de permetre al domini d'ignorar on són les dades i que determinats objectes del domini siguin persistents. Així doncs, dissenyem la base de dades tenint en compte les entitats que té el sistema i quines volem que es guardin de forma permanent.

El disseny lògic és el procés de construir un esquema que representa la informació, independentment del SGBD concret que s'utilitzarà. Per tal d'obtenir l'esquema lògic s'han de fer algunes conversions: eliminar les relacions de molts a molts, eliminar les relacions complexes, les recursives, les que tenen atributs, reconsiderar les relacions un a un i eliminar les relacions redundants.

Els esquemes es poden validar mitjançant la normalització, que garanteix que l'esquema resultant és més pròxim al model de l'empresa, consistent i té la màxima estabilitat.

Les restriccions d'integritat són les restriccions que s'imposen perquè la base de dades mai arribi a un estat inconsistent. Hi ha cinc tipus de restriccions: dades requerides, restriccions de domini, integritat d'entitats, integritat referencial i regles de negoci.

El disseny físic és el procés de crear les taules de la base de dades, la inserció de dades i consultes, així com les transaccions necessàries per al nostre sistema. S'avalua el contingut de tota la base de dades i es mira quines operacions es poden optimitzar, entre altres mètodes, creant índexs.

5.2.1- Disseny lògic

L'objectiu del disseny lògic és convertir els esquemes conceptuals en un esquema lògic global que s'ajusti al model de SGBD sobre el que s'implementarà el sistema. S'ha d'obtenir una representació que usi, del mode més eficient possible, els recursos que el model de SGBD posseeix per estructurar les dades i modelar les restriccions. Farem servir el model relacional ja que és un dels més extensos.

El model relacional no té les característiques d'abstracció que s'usen en el model conceptual, per tant, un primer pas a la fase del disseny lògic consisteix en la conversió de la representació a estructures de baix nivell.

Podem analitzar el diagrama entitat-relació obtingut a l'apartat 4.3 i desglossar-lo:

- Llistat d'especialitzacions. Per a cada especialització del disseny s'indicarà l'entitat principal i les de nivell inferior amb els seus atributs i caracteritzar l'especialització (disjunta, completa, etc.)

Entitat Principal: **Persona**

Atributs	Tipus d'atribut	Rang de valors / exemple
nom	Cadena de caràcters	'Anna'
cognom1	Cadena de caràcters	'Pinyot'
cognom2	Cadena de caràcters	'Garriga'
DNI	Numèric	47159338
sexe	Caràcter	Femení o masculí {F, M}
dataNaixement	Data	Format: aaaa-mm-dd On aaaa {1950-2050}
telèfon	Numèric	937147536
email	Cadena de caràcters	'annapinyot@gmail.com'

Especialització disjunta i completa. Una persona pot ser o bé del tipus pacient o bé del tipus metge.

Entitat de nivell inferior de Persona: **Pacient**

Atributs	Tipus d'atribut	Rang de valors / exemple
nomPare	Cadena de caràcters	'Joan'
cognomsPare	Cadena de caràcters	'Pinyot Garròs'
nomMare	Cadena de caràcters	'Maria'
cognomsMare	Cadena de caràcters	'Garriga Farràs'
mòbilPare	Numèric	612351145
mòbilMare	Numèric	690124421

Entitat de nivell inferior de Persona: **Metge**

Atributs	Tipus d'atribut	Rang de valors / exemple
Especialitat	Cadena de caràcters	'pediatre'
numCol·legiat	Numèric	10998
Mòbil	Numèric	629982304

L'entitat pacient és subclasse de persona i al mateix temps és superclasse de pacientAmbDiagnòstic i pacientTractat. L'especialització és incompleta, és a dir, podem tenir pacients que no tinguin diagnòstic i que no estiguin tractats (seran aquells pacients que encara no s'han visitat).

Entitat de nivell inferior de Pacient: **pacientAmbDiagnòstic**

Atributs	Tipus d'atribut	Rang de valors / exemple
Diagnòstic	Cadena de caràcters	'braquicefàlia coronal'
Observacions	Cadena de caràcters	'15 mm d'asimetria'

Entitat de nivell inferior de Pacient: **pacientTractat**

Atributs	Tipus d'atribut	Rang de valors / exemple
fiTractament	Data	Format: aaaa-mm-dd '2007-12-05'
Durada	Numèric	La unitat són setmanes 12

L'atribut durada correspon a la diferència de mesos entre la data de fiTractament d'aquesta instància i la data dataInici de la classe 'pacientEnTractament'.

Entitat Principal: **Visita**

Atributs	Tipus d'atribut	Rang de valors / exemple
Data	Data	Format: aaaa-mm-dd '2007-09-27'
horaInici	Hora	Format: hh-mm-ss '12-30-00'
horaFi	Hora	Format: hh-mm-ss '13-30-00'

Especialització disjunta i completa. Una visita pot ser 'primera visita', visita de 'seguiment', visita de 'control' o una visita per 'fer guix'.

Entitat de nivell inferior de Visita: **Primera Visita**

Atributs	Tipus d'atribut	Rang de valors / exemple
Preu (abstracte)	Numèric	El preu es defineix a les subclasses de Primera Visita.

Entitat de nivell inferior de Primera Visita: **Amb Guix**

Atributs	Tipus d'atribut	Rang de valors / exemple
Preu	Numèric	Els preus són en euros. 130

Entitat de nivell inferior de Primera Visita: **Sense Guix**

Atributs	Tipus d'atribut	Rang de valors / exemple
Preu	Numèric	Els preus són en euros. 100

Entitat de nivell inferior de Visita: **Visita de Seguiment**

Entitat de nivell inferior de Visita: **Visita de Control**

Aquestes dues entitats no tenen atributs, però cal identificar-les i distingir-les.

Entitat de nivell inferior de Visita: **Visita per fer Guix**

Atributs	Tipus d'atribut	Rang de valors / exemple
preuGuix	Numèric	Els preus són en euros. 30

Entitat Principal: **Casc**

Atributs	Tipus d'atribut	Rang de valors / exemple
Id	Cadena de caràcters	'10229380'

Especialització disjunta i completa. Un casc pot estar enviat ('motlle enviat'), 'pendent de rebre' o bé 'rebut'.

Entitat de nivell inferior de Casc: **Motlle enviat**

Atributs	Tipus d'atribut	Rang de valors / exemple
NumEnviament	Numèric	8209
Data	Data	'2007-08-27'

Entitat de nivell inferior de Casc: **Pendent de rebre**

Atributs	Tipus d'atribut	Rang de valors / exemple
NumSeguiment	Numèric	1294382
Data	Data	'2007-09-10'

Entitat de nivell inferior de Casc: **Rebut**

Atributs	Tipus d'atribut	Rang de valors / exemple
Data	Data	'2007-09-14'

- Llistat d'entitats (no es tornen a especificar les descrites anteriorment). Per cada entitat es mostraran els atributs que té, indicant-ne el tipus (numèric, caràcter, data, booleà, etc.) i un exemple o el rang de valors permès si l'atribut ho requereix.

Entitat: **Adreça**

Atributs	Tipus d'atribut	Rang de valors / exemple
Carrer	Cadena de caràcters	'Centre'
Número	Numèric	33
Pis	Numèric	3
Porta	Numèric	2
Escala	Cadena de caràcters	'A'
codiPostal	Numèric	08211
Població	Cadena de caràcters	'Castellar del Vallès'
Província	Cadena de caràcters	'Barcelona'
País	Cadena de caràcters	'Espanya'

Entitat: **agenda visites**

Una agenda de visites està composta per anys. Cada any té mesos, i els mesos dies. La idea és poder determinar quin horari fa un metge per cada dia, assignant una franja horària de matí i una de tarda.

Entitat: **any**

Atributs	Tipus d'atribut	Rang de valors / exemple
Any	Any	'2007'

Entitat: **mes**

Atributs	Tipus d'atribut	Rang de valors / exemple
Mes	Mes	'octubre'

Entitat: **dia**

Atributs	Tipus d'atribut	Rang de valors / exemple
Dia	Dia	'02'

Entitat: **horari disponible**

Atributs	Tipus d'atribut	Rang de valors / exemple
horaInici	Hora	'10:30:00'
horaFi	Hora	'13:30:00'

Entitat: **seu**

Atributs	Tipus d'atribut	Rang de valors / exemple
nomSeu	Cadena de caràcters	'bilbao'

Entitat: **tipusTractament**

Atributs	Tipus d'atribut	Rang de valors / exemple
nomTractament	Cadena de caràcters	'plagiocefàlia'
preuTractament	Numèric	3000

Entitat: **pacientEnTractament**

Atributs	Tipus d'atribut	Rang de valors / exemple
dataInici	Data	'2007-01-20'
preuEspecífic	Numèric	2500
tractament Pagat?	Booleà	Cert

L'atribut 'tractamentPagat?' és cert si s'han fet dos pagaments amb valor igual a 'preuEspecífic' d'aquesta mateixa entitat.

Entitat: **pagament**

Atributs	Tipus d'atribut	Rang de valors / exemple
quantitat	Numèric	1000
mètode	Cadena de caràcters	'transferència'
data	Data	'2007-04-10'

L'atribut mètode pot tenir els valors següents: 'en metàl·lic', 'transferència', 'xec' o 'travels'.

Entitat: **informe**

	Atributs	Tipus d'atribut	Rang de valors / exemple
Ajustaments	Edat	Numèric	Edat en setmanes. 16
	PerímetreCranial	Numèric	PC en centímetres. 41
	Longitud	Numèric	L en centímetres. 13
	AmpladaMàxima	Numèric	AM en centímetres. 11
	Lideal	Numèric	LIdeal = AM x 100/80 – L real 0.75
	DiagonalMajor	Numèric	En centímetres. 15
	DiagonalMenor	Numèric	En centímetres. 14
	DismetriaBòvedaCranial	Numèric	En mil·límetres. 10
	NasionTragusEsquerre	Numèric	En centímetres. 9
	NasionTragusDret	Numèric	En centímetres. 8
	DismetriaBaseCranial	Numèric	En mil·límetres. 10
	LongitudOrellaOrella	Numèric	En centímetres. 24
	TragoATrago	Numèric	En centímetres. 10
	ÍndexCranialAxial	Numèric	Índex= AM x 100 / L 80
	Observacions	Cadena de caràcters	'plagio tipus 2'

Entitat: **història mèdica**

Atributs	Tipus d'atribut	Rang de valors / exemple
edatMareAlNaixement	Numèric	38
Prematur?	Booleà	Cert
SetmanesNaixement	Numèric	30
PesAlNaixement	Numèric	Les unitats del pes són grams. 2500

AlçadaAlNaixement	Numèric	Les unitats de l'alçada són cm. 50
PrimerFill?	Booleà	Cert
TipusPart	Cadena de caràcters	Rang: {'natural', 'cesària'} 'natural'
NombreFillsAlPart	Cadena de caràcters	'part múltiple de bessons'
OrdreAlNéixer	Numèric	2
AnomaliesCongènites	Cadena de caràcters	'llavi leporí'
PrecaucionsMèdiques	Cadena de caràcters	''
ComplicacionsNeonatals	Cadena de caràcters	'hipoglucèmia'
OperacióCranial?	Booleà	Cert
OperacióAmbSutura?	Booleà	Fals
DataOperació	Data	'2005-11-30'
EscànersFets	Cadena de caràcters	Rang: {'X-RAY','CT','MRI', ''} 'CT'
CapAnormalNaixement?	Booleà	Cert
DescripcióCapNaixement	Cadena de caràcters	''
QuiNotificaDeformitat	Cadena de caràcters	'una tieta'
QuanNotificaDeformitat	Cadena de caràcters	'a les dues setmanes de néixer'
PrimeraNotíciaDocBand	Cadena de caràcters	Rang: {'pediatre', 'anunci', 'internet', 'família/amic', 'altres'} 'internet'
PosicióDormirDesprésNéixer	Cadena de caràcters	Rang: {'cap a la dreta', 'cap a l'esquerra' } 'cap a l'esquerra'
PeríodeCanviPosició	Cadena de caràcters	'cada 4 hores'
ReposicionacióFeta?	Booleà	Fals
Torticollis?	Booleà	Fals
Etiologia	Cadena de caràcters	Rang: {'naixement' ,'posicional', 'úter', 'sinostosis', 'sindròmica'} 'posicional'
PromigHoresSeientCotxe	Numèric	Hores per dia. 2
PromigHoresBressol	Numèric	Hores per dia. 6
PromigHoresCotxet	Numèric	Hores per dia. 1
AplanamentParietalOccipital	Cadena de caràcters	Rang: {'bilateral', 'D>E', 'E>D', 'D', 'E', '-'} 'bilateral'
Raça	Cadena de caràcters	Rang: {'blanc', 'negre', 'hispà', 'asiàtic', 'orient mitjà', 'altres'}

		'asiàtic'
Comentaris	Cadena de caràcters	''

Entitat: **centre**

Atributs	Tipus d'atribut	Rang de valors / exemple
preu1visitaAmbGuix	Numèric	130
preu1visitaSenseGuix	Numèric	100
preuVisitaFerGuix	Numèric	30

- Llistat d'interrelacions. Per cada interrelació del digrama s'explicarà una breu descripció del que representa, el tipus (1-1, N-N, 1-N).

Interrelació	Relaciona	Descripció	Tipus
viu a	Persona – Adreça	Una persona viu a una adreça. A una adreça hi poden viure vàries persones.	N – 1
té agenda	Metge – Agenda visites	Un metge té una agenda de visites. Una agenda correspon a un sol metge.	1 – 1
té història	Pacient – Història	Un pacient té assignada una història mèdica. Una història correspon a un pacient.	1 – 1
consta de	Història - Informe	Una història consta de diversos informes.	1 – N
pertany a	Informe - Història	Un informe pertany a una història mèdica.	N - 1
es redacta	Visita – Informe	A una visita es pot redactar un informe com a conclusió d'aquesta.	1 – 0..1
es fa a	Visita – Seu	Una visita es duu a terme a una seu. A una seu s'hi poden fer vàries visites.	N – 1
feta per	Visita – Metge	Una visita és feta per un o dos metges. Un metge pot tenir vàries visites assignades.	N – 1..2
Correspon	Història – 1era Visita	Una història correspon a una primera visita. Vàries històries es poden haver creat a la primera visita.	N – 1
fa guix	Pacient – Visita de tipus Fer guix	A un pacient només se l'hi pot fer una visita de fer guix un sol cop. Molts pacients poden haver fet una visita on	N – 1

		s'hagi fet el guix.	
fa seguiment	PacientTractat – Visita de tipus De Seguiment	Un pacient que ha estat tractat pot fer fins a 3 visites de seguiment. Molts pacients tractats poden haver fet visites de seguiment.	N – 0..3
fa control	PacientAmbDiagnòstic – Visita de tipus De control	Un pacient que tingui el diagnòstic fet pot anar fent vàries visites de control. Diversos pacients amb diagnòstic poden tenir assignada una visita de control.	N – N
fa tractament	PacientAmbDiagnòstic – PacientEnTractament	Un pacient amb diagnòstic pot fer fins a dos tractaments. Un pacient en tractament correspon a un sol pacient amb diagnòstic.	1 – 0..2
Segueix	PacientEnTractament – tipusTractament	Un pacient en tractament segueix un sol tractament. El mateix tractament pot ser seguit per varis pacients.	N - 1
té assignat	PacientEnTractament – Casc	Un pacient en tractament té assignat un únic casc. Un casc correspon a un sol pacient que estigui fent un tractament.	1 – 1
Efectua	PacientEnTractament – Pagament	Un pacient en tractament pot fer fins a dos pagaments. Cada pagament correspon a un pacient.	1– 0..2
pagada?	Visita de tipus 1era Visita – Pagament	La primera visita pot estar o no pagada. Si existeix el pagament, correspon a una primera visita.	1 – 0..1
pagar guix	Visita de tipus Fer guix – Pagament	Una visita de fer guix pot estar o no pagada. Si el guix està pagat, el pagament correspon a una visita de fer guix.	1 – 0..1

Amb la informació anterior s'han identificat les entitats i de les interrelacions que les associen. La informació es troba dividida en blocs de dades relacionats de forma lògica. Les taules són objectes fonamentals d'una base de dades relacional, en elles es basa el sistema d'emmagatzematge de la informació i el sistema de recuperació.

Tot seguit es pot procedir al disseny físic, on es mostraran els passos per a la creació de les taules de la base de dades.

5.2.2- Disseny físic

A l'etapa del disseny físic s'ha de crear les taules de la base de dades, definir la inserció de dades i consultes, determinar les transaccions necessàries i intentar optimitzar les consultes fetes, si és necessari, amb la creació d'índexs.

El disseny s'ha d'adaptar al SGBD que s'ha escollit per tenir en compte les seves limitacions i avantatges.

Començarem amb la creació de les taules de la base de dades, indicant els passos a seguir:

- Escollir una clau primària per a cada classe.

UML utilitza OO (Orientació a Objectes) i, per tant, fa servir OID (Object Identifier) per a identificar objectes. Una primera opció és fer servir la clau externa del model UML.
Podem tenir varis problemes:

- no existeix clau externa
- existeix clau externa però té molts atributs
- volem poder modificar tots els atributs d'un objecte sense que en canviï la clau

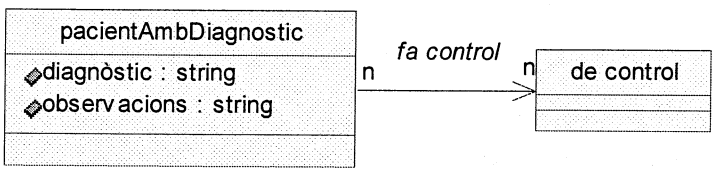
La solució és:

- afegir una columna sense significat que es pugui fer servir per identificar la fila

El valor d'aquesta columna sol ser un comptador que s'incrementa cada cop que afegim una fila. No estan concebuts per a consultar, sinó per a comparar.

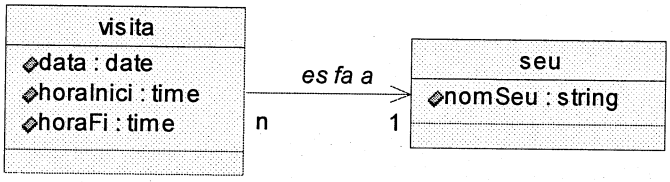
- Tenim els següents tipus d'associacions binàries.

  - tipus M:N



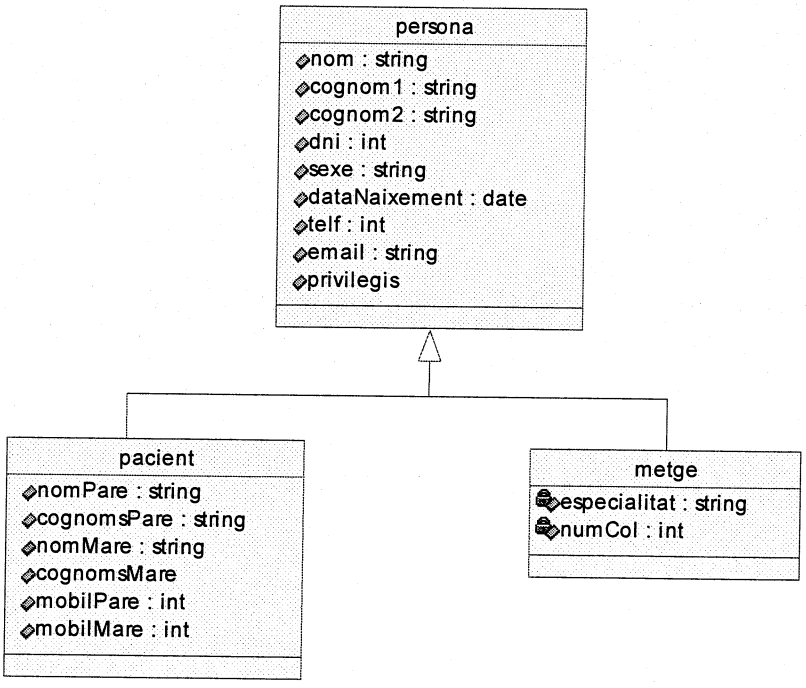
La solució és crear una taula amb la relació que uneix les dues classes, de tal manera que la seva clau primària serà (idPacientAmbDiagnòstic, idControl) on idPacientAmbDiagnòstic és clau forana de la classe pacientAmbDiagnòstic i idControl és clau forana de la classe deControl.

- tipus 1:N



La solució per relacionar les dues classes és afegir una clau forana a la classe de multiplicitat N que referenciï la classe de multiplicitat 1.

- Especialització / generalització



Les classes de nivell inferior tenen una clau primària que alhora és clau forana de la seva superclasse. És a dir, pacient té un identificador de pacient i alhora aquest referència a l'identificador de la classe persona. El mateix passa amb metge.

Tota taula ha de complir les formes normals:

a) Una taula està en primera FN si i només si qualsevol atribut és atòmic (no descomposable, no agregat o grup repetitiu).

b) Una taula està en segona FN si està en primera FN i tot atribut no clau depèn de tota la clau.

c) Una taula està en tercera FN si està en segona FN i tot atribut no clau depèn d'un altre atribut no clau, és a dir, que no hi ha transitivitat.

Seguint els passos descrits anteriorment obtenim les següents taules. La clau primària és l'atribut o conjunt d'atributs de la relació que identifiquen unívocament cadascuna de les tuples de la relació. La clau forana és l'atribut o conjunts d'atributs d'una relació que coincideixen amb els valors de la clau primària d'una altra relació. D'aquesta manera a partir d'una taula podem referenciar a altres relacions.

Adreça (id, carrer, número, pis, porta, escala, codiPostal, població, província, país)
Clau primària: (id)

Persona (nom, cognom1, cognom2, DNI, sexe, dataNaixement, telf, email, adreça)
Clau primària: (nom, cognom1, cognom2)
Clau forana: (adreça → Adreça (idAdreça))

Pacient (id, nomPare, cognomsPare, nomMare, cognomsMare, mobilPare, mobilMare, ferGuix, ferGuix, VisitaPrimera)
Clau primària: (id)
Clau forana: (id → Persona (idPersona))
Clau forana: (ferGuix → VisitaFerGuix (idFerGuix))
Clau forana: (visitaPrimera → visitaPrimera (idVisitaPrimera))

Metge (id, especialitat, numCol legiat, mobil)
Clau primària: (id)
Clau forana: (id → Persona (idPersona))

AgendaVisites (idMetge)

Clau primària: (idMetge)

Clau forana: (metge → Metge (idMetge))

Any (any, agenda)

Clau primària: (any, agenda)

Clau forana: (agenda → AgendaVisites (idAgenda))

Mes (mes, any)

Clau primària: (mes, any)

Clau forana: (any → Any (idAny))

Dia (dia, mes)

Clau primària: (dia, mes)

Clau forana: (mes → Mes (idMes))

HorariDisponible (dia, horaInici, horaFi)

Clau primària: (dia, horaInici, horaFi)

Clau forana: (dia → Dia (idDia))

PacientAmbDiagnostic (id, diagnòstic, observacions)

Clau primària: (id)

Clau forana: (id → Pacient (idPacient))

PacientTractat (id, fiTractament, durada)

Clau primària: (id)

Clau forana: (id→ Pacient (idPacient))

PacientEnTractament (idPacientAmbDiagnostic, dataInici, preuEspecífic, tractPagat?, tipusTractament, casc, 1pagament, 2pagament)

Clau primària: (idPacient, dataInici)

Clau forana: (idPacient → pacientAmbDiagnòstic (id))

Clau forana: (tipusTractament → tipusTractament (id))

Clau forana: (casc → Casc (id))

Clau forana: (1pagament → Pagament (idPagament))

Clau forana: (2pagament → Pagament (idPagament))

TipusTractament (nomTractament, preuTract)

Clau primària: (nomTractament)

Casc (id)

Clau primària: (id)

MotlleEnviat (id, numEnviament, data)

Clau primària: (id)

Clau forana: (id → Casc (idCasc))

PendentDeRebre (id, numSeguiment, data)

Clau primària: (id)

Clau forana: (id → Casc (idCasc))

Rebut (id, data)

Clau primària: (id)

Clau forana: (id → Casc (idCasc))

Seu (nomSeu)

Clau primària: (nomSeu)

Visita (id, data, horaInici, horaFi, informe, metge1, metge2, seu)

Clau primària: (id)

Clau forana: (informe → Informe (idInforme))

Clau forana: (metge1 → Metge (idMetge))

Clau forana: (metge2 → Metge (idMetge))

Clau forana: (seu → Seu (idSeu))

VisitaFerGuix (id, preuGuix, pagament)

Clau primària: (id)

Clau forana: (id → Visita (idVisita))

Clau forana: (pagament → Pagament (idPagament))

VisitaPrimera (id, preu, tipus, pagament) on tipus={'amb guix','sense guix'}

Clau primària: (id)

Clau forana: (id → Visita (idVisita))

Clau forana: (pagament → Pagament (idPagament))

VisitaSeguiment (id)

Clau primària: (id)

Clau forana: (id → Visita (idVisita))

VisitaControl (id)

Clau primària: (id)

Clau forana: (id → Visita (idVisita))

FaSeguiment (pacientTractat, visitaSeguiment)

Clau primària: (pacientTractat, visitaSeguiment)

Clau forana: (pacientTractat → pacientTractat (idPacient))

Clau forana: (visitaSeguiment → VisitaSeguiment (idVisita))

FaControl (pacientAmbDiagnostic, visitaControl)

Clau primària: (pacientAmbDiagnòstic, visitaControl)

Clau forana: (pacientAmbDiagnòstic → pacientAmbDiagnòstic (idPacient))

Clau forana: (visitaControl → visitaControl (idVisita))

Pagament (id, quantitat, mètode, data)

Clau primària: (id)

Informe (id, ajustaments, observacions, visita, història)

Clau primària: (id)

Clau forana: (visita → Visita (idVisita))

Clau forana: (història → Història (idHistòria))

ajustaments = (Edat, PerímetreCranial, Longitud, AmpladaMàxima, Lideal, DiagonalMajor, DiagonalMenor, DismetriaBòvedaCranial, NasionTragusEsquerre, NasionTragusDret, DismetriaBaseCranial, LongitudOrellaOrella, TragoATrago, ÍndexCranialAxial)

HistòriaMèdica (idPacient, dadesHistòria, primeraVisita)

Clau primària: (idPacient)

Clau forana: (idPacient → Pacient (id))

Clau forana: (primeraVisita → VisitaPrimera (id))

dadesHistòria = (edatMareAlNaixement, Prematur?, SetmanesNaixement, PesAlNaixement, AlçadaAlNaixement, PrimerFill?, TipusPart, NombreFillsAlPart, OrdreAlNéixer, AnomaliesCongènites, PrecaucionsMèdiques, ComplicacionsNeonatals, OperacióCranial?, OperacióAmbSutura?, DataOperació, EscànersFets, CapAnormalNaixement?, DescripcióCapNaixement, QuiNotificaDeformitat, QuanNotificaDeformitat,

PrimeraNotíciaDocBand, PosicióDormirDesprésNéixer, PeríodeCanviPosició, ReposicionacióFeta?, Torticollis?, Etiologia, PromigHoresSeientCotxe, PromigHoresBressol, PromigHoresCotxet, AplanamentParietalOccipital, Raça)

Centre (id, preu1visitaAmbGuix, preu1visitaSenseGuix, preuFerGuix)
Clau primària: (id)

Per verificar que la informació de la base de dades és correcta cal complir les següents regles d'integritat:

- De les entitats: cap atribut de la clau primària pot contenir valors nuls (NULL).
- Referencials: una base de dades relacional no pot tenir valors de la clau forana sense concordança.
- Dels dominis semàntics: tot valor d'un atribut ha de pertànyer al domini sobre el qual s'ha definit l'atribut.

Un cop revisat el model relacional obtingut podrem procedir a la seva implementació.

5.3- Disseny de la capa de domini

La capa del domini sap com satisfer les peticions de l'usuari però ignora on es guarden les dades i com es presenten a l'usuari. S'ocupa d'assabentar-se dels esdeveniments, controlar-ne la validesa, executar les accions encomanades per obtenir-ne el resultat i comunicar la resposta.

Actua d'intermediari entre la capa de presentació de la capa de dades. Es relaciona amb la capa de presentació passant-li respostes i resultats dels esdeveniments i consultes que rep. Es relaciona amb la capa de gestió de dades passant-li operacions de consulta i modificacions de dades, i rebent-ne les respostes i resultats.

Cal definir els contractes per les operacions del sistema. Els contractes descriuen els efectes que sorgeixen al aplicar una operació.

Operació: alta_pacient (nom, cognom1, cognom2, dni, sexe, dataNaixement, telf, email, nomPare, cognomsPare, nomMare, cognomsMare, mobilPare, mobilMare, carrer, número, pis, porta, escala, codiPostal, població, província, país, idPrimeraVisita, idFerGuix)

Responsabilitats

Introdueix un pacient al sistema.

Si el pacient encara no ha fet ni primera visita ni guix, els camps idPrimeraVisita i idFerGuix són nuls.

Cas d'ús:

1.1- donar_alta_pacient

Precondicions: -

Postcondicions:

El pacient amb identificador (nom, cognom1, cognom2) s'ha introduït al sistema.

Si el pacient ha fet primera visita o una visita on s'ha fet guix, s'associaran aquestes visites a aquest pacient.

Sortida:

El sistema desglossa els atributs d'entrada per a crear les instàncies Persona, Pacient i Adreça.

Si no existeix cap Persona amb l'identificador (nom, cognom1, cognom2) llavors es creen les classes Persona i Pacient. Retorna un '0' indicant que l'operació s'ha dut amb èxit.

Si l'Adreça no existeix la crea. Si existeix, l'associa a aquesta Persona.

Si existeix la Persona retorna un '-1' indicant que no s'ha pogut dur a terme l'operació.

Operació: modificar_pacient (nom, cognom1, cognom2, dni, sexe, dataNaixement, telf, email, nomPare, cognomsPare, nomMare, cognomsMare, mobilPare, mobilMare, carrer, número, pis, porta, escala, codiPostal, població, província, país, idPrimeraVisita, idFerGuix)

Responsabilitats

Modifica les dades d'un pacient existent al sistema.

Cas d'ús:

1.2- modificar_dades_pacient

Excepcions:

Els camps: nom, cognom1, cognom2 no es poden modificar.

Precondicions:

El pacient amb identificador (nom, cognom1, cognom2) existeix al sistema.

Postcondicions:

S'han modificat les dades del Pacient i Persona amb identificador (nom, cognom1, cognom2) i l'Adreça relacionada amb aquest pacient.

Sortida: -

Operació: consultar_pacient (nom, cognom1, cognom2)

Responsabilitats

Mostra les dades d'un pacient existent al sistema.

Cas d'ús:

1.3- consultar_dades_pacient

Precondicions:

El pacient amb identificador (nom, cognom1, cognom2) existeix al sistema.

Postcondicions: -

Sortida: dades_persona, dades_pacient, dades_adreça

El sistema mostra les dades corresponents a la Persona i Pacient amb (nom, cognom1, cognom2), cerca l'Adreça que té relacionada i també en mostra les dades.

Operació: cercar_pacient (atributs_per_cercar_pacient)

Responsabilitats

Cerca els pacients que coincideixen amb els atributs d'entrada i en retorna una llista amb els identificadors trobats. Els possibles atributs_per_cercar_pacient estan definits al glossari.

Cas d'ús:

1.4- cercar_pacient_per_atribut

Precondicions: -

Postcondicions: -

Sortida:

Set {idPacient}. Es retorna una llista amb els identificadors dels pacients candidats a la cerca.

Si no se'n ha trobat cap, retorna un '-1'.

Operació: obrir_història (idPacient, dadesHistòria)

Responsabilitats

S'adjunta una història mèdica a un pacient.

Cas d'ús:

1.5- obrir_història_mèdica

Precondicions:

El pacient amb identificador (idPacient) existeix al sistema.

Postcondicions:

El pacient amb idPacient té una història associada.

Sortida:

Si el pacient amb identificador (idPacient) ja té una història associada retorna un '-1'.

Si el pacient amb identificador (idPacient) no té cap primeraVisita associada retorna un '-2'.

Sinó, es crea una instància de HistòriaMèdica amb les (dadesHistòria) i s'associa al pacient amb (idPacient).

Operació: modificar_història (idPacient, dades_història_noves)

Responsabilitats

Es modifiquen els valors antics de la història del pacient idPacient per les dades_història_noves.

Cas d'ús:

1.6- modificar_història_mèdica

Precondicions:

El pacient amb identificador (idPacient) existeix al sistema i té història mèdica.

Postcondicions:

Els valors de la història del pacient idPacient han estat modificats.

Sortida: -

Operació: consultar_història (idPacient)

Responsabilitats

Mostra les dades de la història mèdica del pacient idPacient.

Cas d'ús:

1.7- consultar_història_mèdica

Precondicions:

El pacient amb identificador (idPacient) existeix al sistema i té HistòriaMèdica

Postcondicions: -

Sortida: dades_història

El sistema mostra les dades de la HistòriaMèdica corresponents al Pacient amb identificador (idPacient).

Operació: alta_metge (nom, cognom1, cognom2, dni, sexe, dataNaixement, telf, email, especialitat, numCol, mobil, carrer, número, pis, porta, escala, codiPostal, població, província, país)

Responsabilitats

Introdueix un metge al sistema.

Cas d'ús:

2.1- donar_alta_metge

Precondicions: -

Postcondicions:

El metge amb identificador (nom, cognom1, cognom2) s'ha introduït al sistema. Es crea una Agenda de Visites buida per a aquest metge.

Sortida:

El sistema desglossa els atributs d'entrada per a crear les instàncies Persona, Metge i Adreça.

Si no existeix cap Persona amb l'identificador (nom, cognom1, cognom2) llavors es creen les classes Persona, Metge i una Agenda de Visites buida per aquest metge. Retorna un '0' indicant que l'operació s'ha dut amb èxit.

Si l'Adreça no existeix la crea. Si existeix, l'associa a aquesta Persona.

Si existeix la Persona retorna un '-1' indicant que no s'ha pogut dur a terme l'operació.

Operació: modificar_metge (nom, cognom1, cognom2, dni, sexe, dataNaixement, telf, email, especialitat, numCol, mobil, carrer, número, pis, porta, escala, codiPostal, població, província, país)

Responsabilitats

Modifica les dades d'un metge existent al sistema.

Cas d'ús:

2.2- modificar_dades_metge

Excepcions:

Els camps: nom, cognom1, cognom2 no es poden modificar.

Precondicions:

El metge amb identificador (nom, cognom1, cognom2) existeix al sistema.

Postcondicions:

S'han modificat les dades del Metge i Persona amb identificador (nom, cognom1, cognom2) i l'Adreça relacionada amb aquest metge.

Sortida: -

Operació: consultar_metge (nom, cognom1, cognom2)

Responsabilitats

Mostra les dades d'un metge existent al sistema.

Cas d'ús:

2.3- consultar_dades_metge

Precondicions:

El metge amb identificador (nom, cognom1, cognom2) existeix al sistema.

Postcondicions: -

Sortida: dades_persona, dades_metge, dades_adreça

El sistema mostra les dades corresponents a la Persona i Metge amb (nom, cognom1, cognom2), cerca l'Adreça que té relacionada i també en mostra les dades.

Operació: cercar_metge (atributs_per_cercar_metge)

Responsabilitats

Cerca els metges que coincideixen amb els atributs d'entrada i en retorna una llista amb els identificadors trobats. Els possibles atributs_per_cercar_metge estan definits al glossari.

Cas d'ús:

2.4- cercar_metge_per_atribut

Precondicions: -

Postcondicions: -

Sortida:

Set {idMetge}. Es retorna una llista amb els identificadors dels metges candidats a la cerca.

Si no se'n ha trobat cap, retorna un '-1'.

Operació: assignar_horari (idMetge, any, mes, dia, horaInici, horaFi)

Responsabilitats

Per al metge amb idMetge s'assigna una franja horària entre horaInici i horaFi al seu horari, el al dia, mes i any indicats.

Cas d'ús:

2.5- assignar_horari_a_metge

Precondicions:

El metge amb idMetge existeix al sistema.

Postcondicions:

El metge té assignat un horari de disponibilitat per al dia, mes, any i franja horària indicada.

Sortida:

Es retorna un '0' si s'ha pogut assignar l'horari correctament.

Es retorna un '-1' si el metge ja tenia un horari assignat pels atributs d'entrada.

Operació: modificar_horari (idMetge, anyVell, mesVell, diaVell, horaIniciVell, horaFiVell, anyNou, mesNou, diaNou, horaIniciNou, horaFiNou)

Responsabilitats

Per al metge amb idMetge es modifiquen els valors antics pels nous indicats als atributs d'entrada.

Cas d'ús:

2.6- modificar_horari_metge

Precondicions:

El metge amb idMetge existeix al sistema. Els atributs (anyVell, mesVell, diaVell, horaIniciVell i horaFiVell) existeixen a l'agenda del metge.

Postcondicions:

L'horari amb valors antics és reemplaçat pels valors nous.

Sortida:

Es retorna un '0' si s'ha pogut modificar l'horari correctament.

Es retorna un '-1' si el metge ja tenia un horari assignat pels atributs d'entrada nous.

Operació: consultar_horari (idMetge, any, mes, dia)

Responsabilitats

Per al metge amb idMetge es mostra l'horari del dia indicat.

Cas d'ús:

2.7- consultar_horari_metge

Precondicions:

El metge amb idMetge existeix al sistema.

Postcondicions: -

Sortida:

Set {horaInici, horaFi}

Es retorna un '-1' si el metge no té cap horari assignat per al dia indicat.

Operació: programar_visita (data, horaInici, horaFi, idInforme, idMetge1, idMetge2, idSeu, tipus)

Responsabilitats

Es programa una visita. Segons el tipus de visita indicada, s'introduiran més atributs:

Tipus = VisitaFerGuix / atributs addicionals: (preuGuix, idPagament, idPacient)

Tipus = VisitaPrimera / atributs addicionals: (preu, tipus, idPagament, idPacient)

Tipus = VisitaSeguiment / atributs addicionals: (idPacientTractat)

Tipus = VisitaControl / atributs addicionals: (idPacientAmbDiagnostic)

Cas d'ús:

3.1- programar_visita

Precondicions:

El metge amb idMetge1 i idMetge2 existeixen al sistema.

La seu amb idSeu existeix al sistema.

El metge amb idMetge1 i idMetge2 tenen disponible a la seva agenda l'hora entre horaInici i horaFi de la data indicada.

Els preus que conté la taula Centre no tenen valors nuls.

Tipus VisitaFerGuix: el pacient (idPacient) existeix al sistema i no té cap instància de VisitaFerGuix.

Tipus VisitaPrimera: el pacient (idPacient) existeix al sistema i no té cap instància de VisitaPrimera.

Tipus VisitaSeguiment: el pacient (idPacientTractat) existeix al sistema i no té més de dues instàncies de VisitaSeguiment.

Tipus VisitaControl: el pacient (idPacientAmbDiagnostic) existeix al sistema.

Postcondicions:

Per a tots els casos:

Es crea una instància de Visita amb data, horaInini, horaFi, pels metges amb idMetge1, idMetge2 a la seu amb idSeu.

Si no hi ha informe el camp idInforme és nul, si n'hi ha s'indica l'identificador d'aquest.

S'anota a l'agenda dels idMetge1 i idMetge2 que a la (data, horaInici, horaFi) hi ha visita.

Si tipus = VisitaFerGuix:

Es crea una instància de VisitaFerGuix. Si idPagament no és nul, s'indica amb quin pagament està relacionada aquesta visita. Al pacient amb idPacient, s'afegeix l'identificador d'aquesta visita creada al camp idFerGuix.

Si tipus = VisitaPrimera:

Es crea una instància de VisitaPrimera. Si idPagament no és nul, s'indica amb quin pagament està relacionada aquesta visita. Al pacient amb idPacient s'afegeix l'identificador de la visita creada al camp idVisitaPrimera.

Si tipus = VisitaSeguiment:

Es crea una instància de VisitaSeguiment. Es crea una instància de FaSeguiment indicant el identificador del pacient tractat amb idPacientTractat i el de la visita de seguiment acabada de crear.

Si tipus = VisitaControl:

Es crea una instància de VisitaControl. Es crea una instància de FaControl indicant el identificador del pacient amb diagnòstic amb idPacientAmbDiagnostic i el de la visita de control acabada de crear.

Sortida:

Si tot ha anat correctament es retorna un '0'. Si hi ha hagut errors, un '-1'.

Operació: modificar_visita (idVisita, atributs_visita, atributs_segons_visita)

Responsabilitats

Es mostren els atributs de la visita que té per identificador (idVisita) amb l'opció de modificar-los. Són reemplaçats pels atributs_visita i atributs_segons_visita, definits posteriorment.

Atributs_visita = (data, horaInici, horaFi, idInforme, idMetge1, idMetge2, idSeu)

Si (idVisita) correspon a:

VisitaFerGuix: atributs_segons_visita = (preuGuix, idPagament)

VisitaPrimera: atributs_segons_visita = (preu, tipus, idPagament)

Cas d'ús:

3.2- modificar_visita

Precondicions:

La visita amb (idVisita) existeix al sistema.

Postcondicions:

Els valors de la visita amb idVisita han estat modificats pels nous atributs i atributs_segons_visita.

Si s'ha modificat el camp (data, horaInici, horaFi, idMetge1 o idMetge2) es procedirà a fer l'operació 'eliminar_visita' i es farà la de 'programar_visita' amb els atributs no modificats.

Sortida:

Si s'ha modificat correctament es retorna un '0'. Si hi ha hagut errors, un '-1'.

Operació: consultar_visita (idVisita)

Responsabilitats

Es mostren les dades corresponents a la visita amb idVisita.

Cas d'ús:

3.3 – consultar_dades_visita

Precondicions:

La visita amb idVisita existeix al sistema.

Postcondicions: -

Sortida: (data, horaInici, horaFi, pacient, metge1, metge2, seu)

Operació: eliminar_visita (idVisita)

Responsabilitats

La visita amb idVisita queda desprogramada i s'elimina del sistema.

Cas d'ús:

3.4 – eliminar_visita

Precondicions:

La visita amb idVisita existeix al sistema.

La data de la visita que es vol eliminar és més gran que la data actual (no conté informe).

Postcondicions:

La visita amb idVisita s'elimina del sistema. S'elimina la visita a les agendes dels metges implicats.

Sortida: -

Operació: cercar_visita (atributs_per_cercar_visita)

Responsabilitats

Cerca les visites que coincideixen amb els atributs d'entrada i en retorna una llista amb els identificadors trobats. Els possibles atributs_per_cercar_visita estan definits al glossari.

Cas d'ús:

3.5 – cerca_visita_per_atribut

Precondicions: -

Postcondicions: -

Sortida:

Set {idVisita}. Es retorna una llista amb els identificadors de les visites candidates a la cerca.

Si no se'n ha trobat cap, retorna un '-1'.

Operació: redactar_informe (ajustaments, observacions, idVisita, idHistòria)

Responsabilitats

Es crea l'informe amb les dades dels ajustaments i les observacions. S'associa a la visita amb idVisita i a la història amb idHistòria.

Cas d'ús:

3.6 – redactar_informe

Precondicions:

La visita amb idVisita i la història amb idHistòria existeixen al sistema.

Postcondicions:

S'associa l'informe creat amb la visita amb idVisita i s'afegeix a la història amb idHistòria.

Sortida:

Es retorna un '-1' si la visita amb idVisita ja tenia un informe associat.

Operació: modificar_informe (idInforme, ajustaments, observacions)

Responsabilitats

Es modifiquen les dades de l'informe amb idInforme pels ajustaments i les observacions noves.

Cas d'ús:

3.7 – modificar_informe

Precondicions:

L'informe amb idInforme existeix al sistema.

Postcondicions:

S'actualitza la informació que conté l'informe idInforme.

Sortida: -

Operació: consultar_informe (idInforme)

Responsabilitats

Es mostren les dades que conté l'informe amb idInforme.

Cas d'ús:

3.8 – consultar_informe

Precondicions:

L'informe amb idInforme existeix al sistema.

Postcondicions: -

Sortida: (ajustaments, observacions)

Operació: nombre_visites_metge (idMetge, data1, data2)

Responsabilitats

Mostra el nombre de visites que ha fet un metge entre dues dates.

Cas d'ús:

3.9- nombre_visites_per_un_metge

Precondicions:

El metge amb identificador (idMetge) existeix al sistema i (data2 > data1).

Postcondicions: -

Sortida: nombre_visites

El sistema calcula quantes visites ha fet el metge amb idMetge entre la data1 i data2.

Operació: pacients_visitats (idMetge, data1, data2)

Responsabilitats

Retorna els identificadors dels pacients que ha visitat un metge entre dues dates.

Cas d'ús:

3.10- pacients_visitats_per_un_metge

Precondicions:

El metge amb identificador (idMetge) existeix al sistema i (data2 > data1).

Postcondicions: -

Sortida: set {idPacient}

El sistema mostra quins pacients han estat visitats pel metge amb idMetge entre la data1 i data2.

Operació: nombre_visites_pacient (idPacient, data1, data2)

Responsabilitats

Mostra el nombre de visites en què s'ha visitat un pacient entre dues dates.

Cas d'ús:

3.11- nombre_visites_per_pacient

Precondicions:

El pacient amb identificador (idPacient) existeix al sistema i (data2 > data1).

Postcondicions: -

Sortida: nombre_visites

El sistema calcula quantes visites té el pacient amb identificador idPacient entre la data1 i data2.

Operació: nombre_visites_seu (idSeu, data1, data2)

Responsabilitats

Mostra el nombre de visites que s'han fet en una seu entre dues dates.

Cas d'ús:

3.12- nombre_visites_per_seu

Precondicions:

La seu amb identificador (idSeu) existeix al sistema i (data2 > data1).

Postcondicions: -

Sortida: nombre_visites

El sistema calcula quantes visites s'han fet a la seu amb idSeu entre la data1 i data2.

Operació: introduir_pagament (quantitat, mètode, data, idVisita, idPacient)

Responsabilitats

Introdueix el pagament al sistema i indica a quina visita o a quin pacient correspon.

Cas d'ús:

4.1 – introduir_pagament

Precondicions:

idVisita o idPacient no són nuls. Mètode = {'metàl·lic, 'transferència', 'xec', 'travel'}

Postcondicions:

S'ha afegit el pagament al sistema i s'ha relacionat amb la visita o pacient corresponent.

Sortida:

- Si (idVisita != null) llavors es mira si és una VisitaFerGuix o una VisitaPrimera i s'associa aquest pagament a la visita. Es retorna un '0' indicant que s'ha efectuat l'operació correctament. Si idVisita no és ni VisitaFerGuix ni VisitaPrimera es retorna un '-1'.

- Si (idPacient != null) ha de correspondre a un pagament d'un pacientEnTractament. Si no es correspon amb aquest tipus de pacient es retorna un '-2'. Si el pacientEnTractament té els dos camps que indiquen els pagaments complerts, llavors es retorna un '-3'. Si hi ha els dos camps buits s'associa el pagament creat amb el pacient i comprova que 'quantitat' sigui igual al 50% del preu del tractament que està fent. Si només té un camp buit, es comprova que la quantitat dels dos pagaments d'aquest pacient sigui igual a la totalitat del tractament que està fent. Si ho té tot pagat, l'atribut de 'tractPagat?' del pacient és cert. Es retorna un '0' indicant que s'ha efectuat l'operació correctament.

Operació: modificar_pagament (idPagament, quantitat, mètode, data)

Responsabilitats

Modifica les dades d'un pagament pels valors nous d'entrada.

Cas d'ús:

4.2 – modificar_pagament

Precondicions:

El pagament amb idPagament existeix al sistema.

Postcondicions:

Es modifiquen les dades del pagament amb identificador idPagament.

Sortida: -

Operació: eliminar_pagament (idPagament)

Responsabilitats

Elimina un pagament del sistema i recalcula el valor 'tractPagat?' del pacient en qüestió.

Cas d'ús:

4.3 – eliminar_pagament

Precondicions:

El pagament amb idPagament existeix al sistema.

Postcondicions:

Si el pagament correspon a una VisitaPrimera o VisitaFerGuix, s'elimina el pagament i les relacions amb aquestes classes.

Si correspon a una part del tractament d'un pacientEnTractament, s'elimina el pagament i es recalcula l'atribut 'tractPagat?' d'aquest pacient.

Sortida: -

Operació: cercar_pagaments (idPacient)

Responsabilitats

Mostra tots els pagaments que ha efectuat un pacient.

Cas d'ús:

4.4 – cercar_pagaments_d'un_pacient

Precondicions:

El pacient amb idPacient existeix al sistema.

Postcondicions: -

Sortida: set {idPagament}

Es cerca si el pacient ha fet pagaments per les visites en què ha assistit.

Si el pacient és un pacientEnTractament es cerquen els pagaments que ha fet pel tractament.

Es retorna un conjunt buit si el pacient no ha fet cap pagament.

Operació: consultar_pagament (idPagament)

Responsabilitats

Mostra les dades d'un pagament.

Cas d'ús:

4.5 - consultar_pagament_d'un_pacient

Precondicions:

El pagament amb idPagament existeix al sistema.

Postcondicions: -

Sortida: (quantitat, mètode, data)

Operació: establir_tipus_tractament (nomTractament, preu)

Responsabilitats

Introdueix el tractament amb nomTractament al sistema.

Cas d'ús:

5.1- establir_tipus_de_tractament

Precondicions:

El tractament amb nomTractament no existeix al sistema.

Postcondicions:

S'ha afegit el tractament amb nomTractament al sistema.

Sortida: -

Operació: consultar_tipus_tractament (nomTractament)

Responsabilitats

Mostra les dades (el preu) del tractament nomTractament.

Cas d'ús:

5.2- consultar_tipus_de_tractament

Precondicions:

El tractament amb nomTractament existeix al sistema.

Postcondicions: -

Sortida: preuTractament

Operació: modificar_tipus_tractament (nomTractament, preuNou)

Responsabilitats

Modifica el preu antic del tractament nomTractament pel preuNou.

Cas d'ús:

5.3- modificar_tipus_de_tractament

Precondicions:

El tractament amb nomTractament existeix al sistema.

Postcondicions:

S'ha modificat el preu del tractament pel preu nou.

Sortida: -

Operació: pacient_amb_diagnostic (idPacient, diagnostic, observacions)

Responsabilitats

S'indica que el pacient amb idPacient és un pacient que té diagnòstic.

Cas d'ús:

5.4- pacient_amb_diagnòstic

Precondicions:

El pacient amb idPacient no té diagnòstic (no és ni un pacientTractat ni un pacientAmbDiagnòstic ni un pacientEnTractament).

Postcondicions:

Es crea la instància de pacientAmbDiagnostic amb les dades 'diagnostic' i 'observacions'.

Sortida: -

Operació: fer_tractament (idPacient, dataInici, preuEspecific, idTipusTractament, idCasc, idPagament1, idPagament2)

Responsabilitats

S'indica que el pacient amb idPacient és un pacient que fa tractament.

Cas d'ús:

5.5- fer_tractament

Precondicions:

El pacient amb idPacient és un pacientAmbDiagnostic i no ha fet més d'un tractament.

Ha d'existir un tipus de tractament que es correspongui amb el camp 'idTipusTractament'.

Els camps idCasc, idPagament1 i idPagament2 poden ser nuls.

Postcondicions:

Es crea la instància de PacientEnTractament amb les dades d'entrada.

Sortida: -

Operació: fi_tractament (idPacient, dataFi)

Responsabilitats

S'indica que el pacient amb idPacient és un pacient que ha acabat el tractament.

Cas d'ús:

5.6- pacient_tractat

Precondicions:

El pacient amb idPacient és un pacientEnTractament.

Postcondicions:

S'elimina la instància de pacientAmbDiagnostic que té per identificador idPacient i es crea una instància de pacientTractat amb identificador idPacient. Es calcula la durada del tractament amb la dataInici del pacientEnTractament corresponent i la dataFi d'entrada.

Sortida: -

Operació: assignar_casc (idPacient, idCasc, numEnviament, data)

Responsabilitats

Associa el casc amb idCasc al pacient amb idPacient.

Cas d'ús:

5.7- assignar_casc_a_pacient

Precondicions:

El pacient amb idPacient és un pacientEnTractament i no té cap casc assignat.

El casc amb identificador idCasc no existeix al sistema.

Postcondicions:

Es crea una instància de 'casc' amb (idCasc) i s'associa al pacientEnTractament amb identificador idPacient. Es crea una instància de 'motlleEnviat' amb (idCasc, numEnviament, data)

Sortida: -

Operació: modificar_estat_casc (idCasc, estatNou, atributs_nou_estat)

Responsabilitats

Canvia l'estat del casc segons l'estat al que passarà.

Cas d'ús:

5.8- modificar_estat_casc

Precondicions:

EstatNou = {'motlle enviat', 'pendent de rebre', 'rebut'}

El casc amb identificador idCasc existeix al sistema.

Postcondicions:

Es comprova l'estat del casc amb identificador idCasc. Si coincideix amb l'estat nou, permet modificar els

atributs que conté. Si els estats són diferents:

Si estatNou = 'pendent de rebre' es crea la classe PendentDeRebre i permet introduir els valors (numSeguiment, data). Elimina el MotlleEnviat amb identificador idCasc.

Si estatNou = 'Rebut' es crea la classe Rebut amb l'identificador idCasc i s'eliminen, si existeixen, les classes amb idCasc MotlleEnviat i PendentDeRebre.

Sortida: -

Operació: consultar_estat_casc (idCasc)

Responsabilitats

Mostra l'estat i els atributs del casc amb idCasc.

Cas d'ús:

5.9- consultar_estat_casc

Precondicions:

El casc amb identificador idCasc existeix al sistema.

Postcondicions: -

Sortida: {estat_casc, atributs}

Segons el tipus d'instància del casc amb identificador idCasc:

estat_casc = {'motlle enviat', 'pendent de rebre', 'rebut'}

Operació: afegir_seu (nomSeu)

Responsabilitats

Introdueix la seu amb nomSeu al sistema.

Cas d'ús:

6.1- afegir_seu

Precondicions:

La seu amb nomSeu no existeix al sistema.

Postcondicions:

S'ha afegit la seu amb nomSeu al sistema.

Sortida: -

Operació: consultar_seus ()

Responsabilitats

Mostra totes les seus que hi ha al sistema.

Cas d'ús:

6.2- consultar_seus

Precondicions: -

Postcondicions: -

Sortida: set {nomSeu}

Operació: eliminar_seu (nomSeu)

Responsabilitats

Elimina la seu amb nomSeu del sistema.

Cas d'ús:

6.3- eliminar_seu

Precondicions:

La seu amb nomSeu existeix al sistema.

Postcondicions:

El sistema deixa de tenir la seu amb nomSeu.

Sortida: -

Operació: establir_preu_1visita (preuNou, ambGuix?)

Responsabilitats

Estableix el preu de la primera visita segons sigui amb guix o sense.

Cas d'ús:

6.4- establir_preu_primera_visita

Precondicions: -

Postcondicions:

El valor antic del preu de la primera visita és reemplaçat pel preu nou, segons si s'indica que és una visita amb guix o sense.

Sortida: -

Operació: consultar_preu_1visita (ambGuix?)

Responsabilitats

Mostra el preu de la primera visita segons sigui amb guix o sense.

Cas d'ús:

6.5- consultar_preu_primera_visita

Precondicions: -

Postcondicions: -

Sortida:

Si existeix es retorna el (preuPrimeraVisita)

Si no ha estat introduït prèviament, es retorna un '-1'.

Operació: preu_fer_guix (preuNou)

Responsabilitats

Estableix el preu d'una visita on només es fa guix.

Cas d'ús:

6.6- establir_preu_fer_guix

Precondicions: -

Postcondicions:

El valor antic del preu d'una visita on només es fa guix és reemplaçat pel preu nou.

Sortida: -

Operació: preu_1_visita (ambGuix?)

Responsabilitats

Mostra el preu d'una visita on només s'ha fet guix.

Cas d'ús:

6.7- consultar_preu_fer_guix

Precondicions: -

Postcondicions: -

Sortida:

Si existeix es retorna el (preuFerGuix)

Si no ha estat introduït prèviament, es retorna un '-1'.

5.4- Disseny de la interfície d'usuari

La capa de presentació sap com presentar les dades a l'usuari però ignora quines transformacions cal fer per donar resposta a les peticions d'aquest. Es relaciona amb els usuaris rebent-ne esdeveniments i presentant-los respostes i resultats. S'ocupa d'ordenar l'execució d'accions a la capa del domini i de tractar finestres, botons, diàlegs, menús i llistats a nivell d'interfície.

El format de les dades d'entrada serà controlat per expressions regulars a la capa de presentació mitjançant scripts JavaScript. D'aquesta manera es garanteix que la informació introduïda per l'usuari al sistema està en un format adient al tipus d'atribut esperat. S'utilitzaran les següents per validar:

- valors numèrics: /^[0-9]+$/
- cadenes de caràcters: /^[a-zA-Z-'\s]+$/
- email: /^[\w\-\.\+]+\@[a-zA-Z0-9\.\-]+\.[a-zA-z0-9]{2,4}$/
- valors alfanumèrics: /^[0-9a-zA-Z]+$/

El disseny intern del comportament de la capa de presentació es farà a partir de cada operació detectada als controladors de domini. Es tracta d'obtenir, per a cada operació, un disseny de com es durà a terme la seva execució per a resoldre totes les responsabilitats implicades.

5.4.1- Disseny detallat. Lògica interna de cada mòdul

El disseny detallat s'ha realitzat en format digital, d'aquesta manera es pot provar el prototip de l'aplicació, provar les navegabilitats i la interacció amb l'usuari. Es veu clarament com està estructurada la informació i quins són els passos a seguir per a dur a terme les diferents funcionalitats.

Per a provar el prototip cal tenir instal·lat un servidor local per tal d'executar les pàgines PHP i activar la base de dades mySQL.

La implementació és el procés de traduir el disseny a un programa executable. El prototip de l'aplicació és la base per acabar de fer tota la futura implementació.

Un cop estigui feta s'haurà de dur a terme diverses proves:

- **Proves d'unitats.**

Provar components individuals. Per exemple, les operacions de donar d'alta pacients, metges, seus. Operacions de modificar instàncies, de consultar dades ja introduïdes i d'eliminar-ne.

- **Proves de mòduls.**

Provar col·leccions relacionades de components dependents. Per exemple, es proven aquelles operacions que depenen d'altres: assignar un horari a un metge, donar d'alta visites, pagaments de tractaments, etc.

- **Proves de subsistemes.**

Integrar els mòduls en subsistemes i es proven. L'objectiu és testejar la interfície. Són proves sobretot de validació al moment d'introduir les dades i es prova mòdul per mòdul totes les operacions possibles.

- **Prova del sistema.**

Testejar el sistema complet. Un cop funciona tot al servidor local es pujarà al servidor del lloc que ens proporciona l'allotjament web.

- **Prova d'acceptació.**

Proves amb el client per comprovar si el software obtingut és acceptable.

Persona:

Una persona pot ser un metge o un pacient. L'identificador d'una persona serà una clau formada per aquests tres valors: nom, primer cognom i segon cognom. La clau no pot ser el DNI perquè la majoria de nens (els pacients) amb pocs mesos de vida no en disposen.

Paràmetres per a la cerca d'una persona:

Una persona es pot buscar pels atributs: nom, primer cognom, segon cognom, DNI, sexe, data de naixement (mes/any), telèfon, mòbil, e-mail o adreça.

Pacient:

Persona que es sotmet al tractament.

Camps obligatoris per donar d'alta un pacient:

Els atributs següents són obligatoris: nom, primer cognom i segon cognom.

Paràmetres per a la cerca d'un pacient:

Un pacient es pot buscar pels atributs de persona més: nom del pare, cognoms del pare, nom de la mare, cognoms de la mare, estat del pacient (pacient amb diagnòstic / pacient en tractament/ pacient tractat), durada del tractament, si té el tractament pagat o no, seu on ha estat visitat, pel metge que ha estat visitat, els mesos o any que ha estat visitat.

Metge:

Persona que treballa al centre mèdic.

Camps obligatoris per donar d'alta un metge:

Els atributs següents són obligatoris: nom, primer cognom i segon cognom.

Paràmetres per a la cerca d'un metge:

Un metge es pot buscar pels atributs de persona més: especialitat, número de col·legiat, pel nombre de visites (superiors o inferiors a una xifra) en un rang de temps.

Seu:

Lloc on el centre mèdic duu a terme visites.

Visita:

Una visita representa un pacient que va al metge en una data i hora determinades per a obtenir una valoració del seguiment del tractament. Aquesta valoració es pot representar en un 'informe'.

Camps obligatoris per donar d'alta una visita:

Els atributs següents són obligatoris: pacient, data, hora d'inici, hora final, seu i metge (mínim un i màxim dos).

Paràmetres per a la cerca d'una visita:

Una visita es pot buscar pels atributs: data, hora, metge, pacient, seu.

Pagament:

Un pagament representa quan un pacient paga una quantitat a favor del centre mèdic en una data concreta.

Camps obligatoris per donar d'alta un pagament:

Els atributs següents són obligatoris: quantitat, mètode, data i a quina visita o pacient correspon.

Paràmetres per a la cerca d'un pagament:

Un pagament es pot buscar pels atributs: pacient, data, mètode.

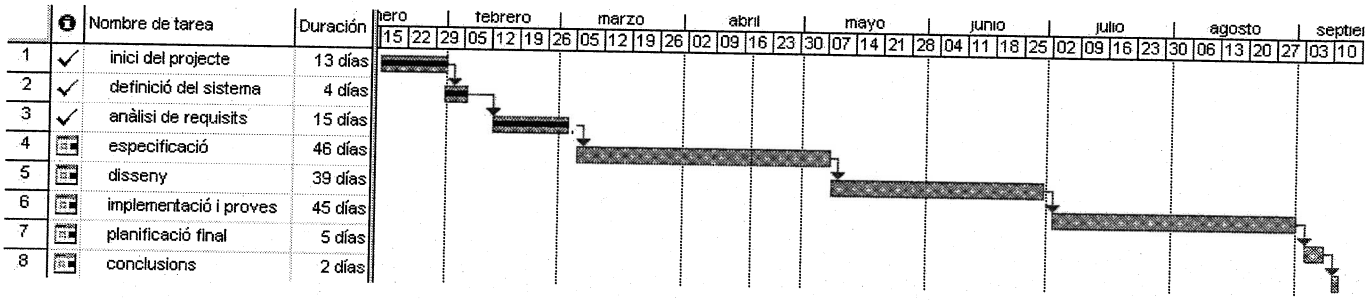
Història mèdica:

Tot pacient que hagi tingut una primera visita té associada una història mèdica. Aquesta recull tots els paràmetres mèdics necessaris per al posterior seguiment del pacient.

Informe:

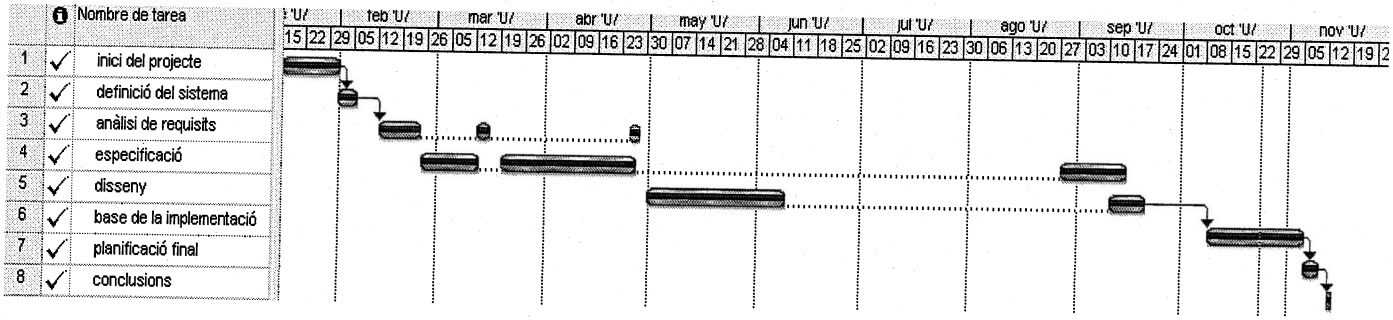
Un informe recull la informació presa durant una visita. Cada informe que es genera a les visites forma part de la història mèdica d'un pacient.

El següent diagrama de Gantt mostra la planificació inicial de com es pensava que es realitzaria el projecte:



A l'hora de la veritat hi ha hagut canvis. En un principi s'havia previst un desenvolupament del software de forma seqüencial, és a dir, el cicle de vida clàssic. És un model lineal que quan finalitza una fase en comença una altra, i el que realment s'ha seguit ha estat un model evolutiu. La manera de treballar ha estat iterativa, retrocedint per tal d'afegir noves funcionalitats i tornant a avançar. A cada cicle s'ha aconseguit una documentació més sofisticada del producte.

El següent diagrama mostra la planificació final:



L'etapa d'anàlisi de requisits i d'especificació s'ha anat fent de forma saltejada, han sorgit nous requisits que s'han considerat importants al moment en què s'estava especificant. Durant el disseny també s'ha retocat part de l'especificació. Un cop donades per finalitzades totes les fases fins al disseny es va decidir no dur a terme tota la implementació del projecte, sinó fer-ne només un prototip. Això va ser degut a la complexitat que aquesta comporta i el poc temps del que es disposa. Així doncs, es deixa obert el projecte per a una futura implementació.

Un cop tenim la planificació temporal cal fer un anàlisi econòmic per veure quin cost té aquest projecte. Primer de tot calculem en hores el que ha suposat cada etapa:

Fase	Duració en dies	Hores (8h/dia)
Definició del sistema	5	40
Anàlisi de requisits	15	120
Especificació	50	400
Disseny	20	160
Base de la implementació	30	60
Planificació final	2	16
Conclusions	1	8
	TOTAL	**804 hores**

Els recursos emprats per al projecte es poden dividir en recursos humans (personal necessari) i tecnològics (hardware, software i llicències).

Recursos humans:

Rol	Què fa?	Sou
Analista	És el responsable de l'anàlisi de requisits, l'especificació, el disseny de l'aplicació i les planificacions del projecte.	50 € / hora
Programador	És el responsable de la implementació de l'aplicació.	30 € / hora
Tester	És el responsable del testeig i la integració de l'aplicació, també recull les dades de les execucions.	20 € / hora

Tot seguit veurem l'ocupació de cadascun d'ells i veurem la despesa total vinculada als recursos humans necessaris.

Rol	Hores treballades	Renumeració total
Analista	744	37.200 €
Programador	40	1.200 €
Tester	20	400 €
	COST TOTAL DE DESPESES EN PERSONAL	**38.800 €**

Recursos tecnològics:

El següent quadre mostra les despeses associades a la tecnologia bàsica per al desenvolupament del sistema:

Element	Quantitat	Per a què el necessitem?	Què val?
PC	1	Essencial per treballar i allotjar l'aplicació a un servidor local. S'ha calculat un preu estàndard.	1.200 €
Apache + mySQL	1	Necessitem la base de dades.	0 €
Dreamweaver	1	Per implementar l'aplicació cal l'editor de php.	480 €
Windows XP	1	S'utilitza aquest sistema operatiu.	250 €
Mesos d'allotjament web amb servidor PHP i mySQL	2	Durant les fases d'implantació i testeig del software cal ubicar el software a un servidor web que suporti PHP i tingui mySQL.	40 €
Mesos de connexió a internet	2	Cal connexió a internet per accedir a l'aplicació ubicada a un servidor.	60 €
		COST TOTAL DE DESPESES TECNOLÒGIQUES	**2.030 €**

Cost total:

El cost total es calcula simplement sumant els recursos humans i els recursos tecnològics.

Tipus de cost	Despesa
Recursos humans	38.800 €
Recursos tecnològics	2.030 €
COST TOTAL	**40.830 €**

9- Conclusions i relació del projecte amb la carrera

El projecte és un recull de conceptes i tècniques estudiades a les assignatures de la branca d'enginyeria del software. De cada una d'elles s'ha agafat allò que era necessari:

- **Enginyeria del software I:** procés i metodologies més importants per al desenvolupament de software, elements necessaris per a l'especificació i notació UML.

ESI m'ha aportat els coneixements necessaris per a veure la realitat des de l'òptica d'un enginyer, modelitzant, entenent i especialment conceptualitzant tot l'entorn en un diagrama, una imatge de la realitat. Aquest diagrama plasma l'entorn, els actors que hi desenvolupen un paper, els conceptes importants i la relació entre ells. Es tracta d'essencialitzar el complex flux de treball d'un negoci, per a quedar-nos amb tot allò que es necessari per tal de dur a terme la informatització.

- **Enginyeria del software II:** patrons de disseny orientats a objectes i elements de l' UML específics per al disseny.

ES2 m'ha proporcionat les eines necessàries per estructurar el model de dades en patrons de disseny. Els patrons de disseny són solucions generals a problemes comuns en el disseny d'un programa, aquestes solucions ja preestablertes són de gran ajut, tant sols cal adaptar-les a la situació específica.

- **Projecte d'enginyeria del software i bases de dades:** gestió de projectes tot seguint processos d'anàlisi i de disseny metòdics, disciplinat i sistemàtics.

- **Enginyeria de requisits:** especificar formalment els requisits funcionals.

PESBD suggereix un fil de treball pel model de desenvolupament del software. A cada fase es conclouen unes tasques que permeten anar definint el sistema adaptant-lo al cas específic. El procés de treball condueix a corregir aquells requisits que en un principi no es contemplaven. Les tècniques explicades a Enginyeria de Requisits m'han permès captar què és clau i què no i quina importància té per al client un requisit concret.

La càrrega de treball de l'estudiant s'estima en 20 hores per crèdit. D'aquesta manera, per un projecte de 37.5 crèdits es preveuen 750 hores de treball. A la planificació final d'aquest projecte s'han calculat 804 hores. Es va decidir no continuar amb la implementació del projecte perquè s'hagués sobrepassat considerablement aquest nombre d'hores.

Des del món laboral me'n he adonat dels coneixements que he anat adquirint durant el meu pas per la FIB. No m'ha servit conèixer els nombres de Carmichael. Tampoc m'ha aportat res a curt termini saber multiplexar un processador de 64 bits. I saber que l'ENIAC va ser presentat al febrer del 1945 no ha fet que m'augmentessin el sou.

Ara bé, des de la distància, me n'adono que he completat els meus estudis *d'Enginyeria Informàtica*; i que és una enginyera informàtica? Doncs bé, per mi, abans que informàtica sóc enginyera, algú capaç de fer servir els recursos disponibles per a proposar solucions adients als problemes que es plantegen. En quant a "informàtica" m'aferro al sentit més etimològic de la paula, referent a informació, a gestió de la informació.

Així doncs, em considero una persona capaç d'aportar solucions a problemes en el camp de la informació. En aquest sentit la FIB m'ha **ensenyat a aprendre,** a no tenir por dels problemes que se'm presentin, sabent dividir-los fins a un punt on la seva dificultat sigui accessible, i intentar aplicar patrons estudiats i coneguts, tal i com la cèlebre definició d'intel·ligència assenyala: *La intel·ligència resideix en la capacitat de detectar i aplicar patrons.*