# ERP Hospital

## Diseño final

DANIEL LÓPEZ GARCÍA
RAFAEL NOGALES VAQUERO
LOTHAR SOTO PALMA
ELENA TORO PÉREZ
*Universidad de Granada*
8 de febrero de 2016

## Índice

# 1. Descripción General

# 2. Informacion

Los requisitos de datos enunciados en los requisitos funcionales son hipervínculos, puede pulsar en ellos para dirigirse automáticamente a uno de ellos.

## 2.1. Sistema de información

El objetivo es llegar a desarrollar un sistema de información útil para un hospital, ya que este tipo de organizaciones poseen áreas claramente divididas y que son heterogéneas, es decir, dichas áreas no tienen que ver entre si, como puede ser la gestión de personal o recursos humanos y la gestión de material por ejemplo. Se llevará a cabo la descripción de los requisitos previos para diseñar un sistema de información que se encargue de la organización de dichas áreas y en especial llevar una planificación y un control acerca de que procedimiento se tiene que aplicar sobre cada paciente en cada momento.

Las áreas funcionales que se tratarán de cubrir son las siguientes:

- Gestión de recursos humanos.
- Contabilidad y gestión de materiales.
- Gestión de asistencia.
- Logística.

# 3. Funcionalidades del sistema

## 3.1. Funcionalidad: Gestión de recursos humanos

### 3.1.1. Descripción

Este conjunto de módulos se encargará del manejo del personal, los turnos que se tienen que realizar en el hospital, el tipo de personal que es, por ejemplo si es enfermero, médico u otro y la información de los mismos entre otros.

Para ello, el responsable registra los datos del empleado(nombre, DNI, dirección, telefono), otros datos referentes a su contrato (tipo de empleado, numero de horas) y el turno en el que trabaja.

### 3.1.2. Requisitos funcionales

#### RF1: Dar de alta un nuevo empleado

- Actor: Empleado del sistema(encargado)
- Entrada: RD7
- Procesamiento: Crear un nuevo empleado, con un identificador(RD6).

- Salida: Ninguna

**RF2: Dar de baja a un empleado.**

- Actor: Empleado del sistema(encargado)
- Entrada: RD8
- Procesamiento: Eliminar los datos referentes al empleado.(RD6)
- Salida: Ninguna

**RF3: Cambiar turno de un trabajador.**

- Actor: Empleado del sistema(encargado)
- Entrada: RD8, RD9
- Procesamiento: Modificar el turno del empleado.
- Salida: Ninguna

**RF4: Modificar datos empleado.**

- Actor: Empleado del sistema(encargado)
- Entrada: RD8, RD6
- Procesamiento: Modificar los datos referentes al empleado.
- Salida: Ninguna

**RF5: Consultar datos de un empleado.**

- Actor: Empleado del sistema(encargado)
- Entrada: RD8
- Procesamiento: Obtener los datos del empleado identificado por RD8
- Salida: RD6

**RF6: Obtener todos los empleados de un tipo.**

- Actor: Empleado del sistema(encargado)
- Entrada: RD10
- Procesamiento: Obtener todos los empleados de un tipo
- Salida: RD6

**RF7: Obtener todos los empleados de un turno y tipo.**

- Actor: Empleado del sistema(encargado)
- Entrada: RD10, RD9
- Procesamiento: Obtener todos los empleados del mismo tipo(RD10) en un turno(RD9).
- Salida: RD6

## 3.2. Funcionalidad: Contabilidad y gestión de materiales

### 3.2.1. Descripción

Esta funcionalidad se encargará de todo lo relacionado con las facturas del hospital y los materiales. En lo que a gestión de material respecta vamos a diferenciar distintos tipos de material, ya sean clínicos, farmaceuticos, máquinas o bien alimentos, además el sistema será conocedor de la localización de dichos materiales en almacenes y el stock de estos. En lo que a contabilidad respecta analizamos las facturas separandolas por tipos ya sean facturas producidas por los materiales o bienes o facturas de terceros y por último permite pagarlas regularmente.

### 3.2.2. Requisitos funcionales

**RF1: Añadir Material:**

- Actor: Empleado del sistema(encargado)
- Entrada: RD1
- Procesamiento: Crea un nuevo material en el sistema, con su identificador (RD1)
- Salida: Ninguna

**RF2: Eliminar Material:**

- Actor: Empleado del sistema(encargado)
- Entrada: RD1
- Procesamiento: Se busca el material precisado y se elimina de la base de datos. (RD1)
- Salida: Ninguna

**RF3: Consultar materiales necesarios:**

- Actor: Empleado del sistema(encargado)
- Entrada: Ninguna
- Procesamiento: Crea una lista buscando materiales que no posean existencias en ningún almacen. (RD1, RD2)
- Salida: Lista con materiales no disponibles.

**RF4: Inventario de materiales:**

- Actor: Empleado del sistema(encargado)
- Entrada: Ninguna
- Procesamiento: Elabora una lista de los materiales que se encuentran en todos los almacenes del hospital. (RD1, RD2)
- Salida: Lista de materiales en Almacen.

**RF5: Consultar almacenes:**

- Actor: Empleado del sistema(encargado)
- Entrada: RD1
- Procesamiento: Elabora una lista de almacenes que poseen un determinado material. (RD1, RD2)
- Salida: Lista de almacenes que contienen un material. (RD1)

**RF6: Añadir factura:**

- Actor: Empleado del sistema(encargado)
- Entrada: RD4
- Procesamiento: Crea una nueva factura en el sistema, con su identificador. (RD4)
- Salida: Ninguna

**RF7: Pagar factura:**

- Actor: Empleado del sistema(encargado)
- Entrada: RD5
- Procesamiento: Usando una información de pago dependiendo del tercero o proveedor que proporcione el material o servicio, se paga una determinada factura.(RD4)
- Salida: Ninguna

**RF8: Consultar facturas de servicios:**

- Actor: Empleado del sistema(encargado)
- Entrada: Ninguna
- Procesamiento: Crea una lista de las facturas que están relacionadas con servicios. (RD4)
- Salida: Lista de facturas de servicios.

**RF9: Consultar facturas de Material:**

- Actor: Empleado del sistema(encargado)
- Entrada: Ninguna
- Procesamiento: Crea una lista de las facturas que están relacionadas con material. (RD4)
- Salida: Lista de facturas de material.

**RF10: Consultar facturas de Personal:**

- Actor: Empleado del sistema(encargado)
- Entrada: Ninguna
- Procesamiento: Crea una lista de las facturas que están relacionadas con personal. (RD4)
- Salida: Lista de facturas de personal.

**RF11: Consultar tipo de factura:**

- Actor: Empleado del sistema(encargado)
- Entrada: RD4
- Procesamiento: Obtiene el tipo de facura.
- Salida: Tipo de factura

## 3.3. Funcionalidad: Gestión de Asistencia

### 3.3.1. Descripción

Todo lo relacionado con la gestión de citas de los pacientes, tanto a nivel de médico de cabecera como de una consulta de especialidad.

### 3.3.2. Requisitos funcionales

## 3.4. Funcionalidad: Gestión de Asistencia

### 3.4.1. Descripción

Todo lo relacionado con la gestión de citas de los pacientes, tanto a nivel de médico de cabecera como de una consulta de especialidad.

### 3.4.2. Requisitos funcionales

**RF1: Crear cita**

- Actor: Empleado del sistema(encargado)

- Entrada: RD13 (Paciente)
- Procesamiento: Esta función es la encargada de llamar a las demás subfunciones. Debe llamar a crearCitaEspecialidad ó crearCitaMedicoCabecera, para elegir el tipo de cita que el Paciente RD13 desee.
- Salida:

**RF2: Ver primer hueco libre**

- Actor: Empleado del sistema(encargado)
- Entrada: Array de citas (huecos) del Médico RD16
- Procesamiento: Función que nos devuelve el primer hueco libre del Médico RD16.
- Salida: int

**RF3: Ocupar hueco**

- Actor: Empleado del sistema(encargado)
- Entrada: RD13 (Paciente), RD14 (Cita), RD16 (Médico)
- Procesamiento: Función que le asigna al Paciente RD13 la Cita RD14 elegida y el Médico RD16 obtenido en crearCitaEspecialidad ó crearCitaMedicoCabecera.
- Salida:

**RF6: Crear cita de especialidad**

- Actor: Empleado del sistema(encargado)
- Entrada: RD13 (Paciente), RD15 (Especialidad)
- Procesamiento: Función que selecciona la especialidad requerida por el Paciente RD13. Esta, a su vez, llama a siguienteMedicoLibre que devuelve el primer médico libre que encuentre de la especialidad requerida.
- Salida:

**RF7: Pedir cita de médico de cabecera**

- Actor: Empleado del sistema(encargado)
- Entrada: RD13 (Paciente)
- Procesamiento: Función que asigna una cita general con el primer médico libre de cabecera.
- Salida:

**RF9: Siguiente Médico Libre**

- Actor: Empleado del sistema(encargado)
- Entrada: RD15 (Especialidad)
- Procesamiento: Función que recorre el array de Médicos y devuelve el primer Médico libre de la Especialidad requerida.
- Salida: RD16 (Médico)

**RF10: Ingresar paciente:**

- Actor: Empleado del sistema(encargado)
- Entrada: RD13 (Paciente), RD17 (Sala)
- Procesamiento: Asigna un paciente a una determinada sala de tipo habitación.
- Salida: Ninguna

**RF11: Dar de alta paciente:**

- Actor: Empleado del sistema(encargado)
- Entrada: RD13 (Paciente), RD17 (Sala)
- Procesamiento: Libera una determinada sala de tipo habitación de un paciente.
- Salida: Ninguna

## 3.5. Funcionalidad: Logística

### 3.5.1. Descripción

Esta funcionalidad se encargará de todo lo relacionado con la gestion de ambulancias y sus rutas. En lo que a gestión de ambulancias se refiere vamos a diferenciar distintos tipos de ambulancias: las que tienen material de asistencia y las que no, además de las UVI moviles. El sistema sabrá decirnos que ambulancias están listas para partir y cuanto tiempo les falta para volver a las que hay de viaje.

### 3.5.2. Requisitos funcionales

**RF1: Consultar estado de las Ambulancias:**

- Actor: Empleado del sistema(encargado)
- Entrada: RD11
- Procesamiento: Consulta cuantas ambulancias hay disponibles y cuanto tiempo les falta aproximadamente para volver a las que hay haciendo una ruta.
- Salida: Tabla de estados de ambulancias y tiempos

**RF2: Ver rutas pentidentes:**

- Actor: Empleado del sistema(encargado)
- Entrada:
- Procesamiento: Consulta la lista de rutas que hay para el día actual y si ha habido una ambulancia que ya la haya antendido, o está siendo atendida.
- Salida: Lista de rutas no atendidas.

**RF3: Añadir rutas:**

- Actor: Empleado del sistema(encargado)
- Entrada: RD12
- Procesamiento: Añade la ruta de entrada a la lista de rutas.
- Salida: Ninguna

**RF4: Eliminar rutas:**

- Actor: Empleado del sistema(encargado)
- Entrada: RD12
- Procesamiento: Elimina la ruta de entrada de la lista de rutas.
- Salida: Ninguna

**RF5: Añadir Ambulancia:**

- Actor: Empleado del sistema(encargado)
- Entrada: RD11
- Procesamiento: Añade la ambulancia de entrada a la flota.
- Salida: Ninguna

**RF6: Eliminar Ambulancia:**

- Actor: Empleado del sistema(encargado)
- Entrada: RD11
- Procesamiento: Elimina la ambulancia de entrada de la flota.
- Salida: Ninguna

**RF7: Añadir Sala:**

- Actor: Empleado del sistema(encargado)
- Entrada: RD17
- Procesamiento: Crea una nueva sala en el sistema con un determinado tipo. (RD17)
- Salida: Ninguna

**RF8: Eliminar Sala:**

- Actor: Empleado del sistema(encargado)
- Entrada: RD17
- Procesamiento: Elimina una sala en el sistema. (RD17)
- Salida: Ninguna

# 4. Requisitos de datos

**RD1: Datos almacenados de Material**

- Número de identificación [Entero]
- Nombre del material [Cadena de caracteres]
- Tipo de material puede ser clínico, farmacéutico, maquinaria o alimenticio [Cadena de caracteres]
- Stock [Booleano]
- Proveedor [Entero]

**RD2: Datos almacenados de Almacen**

- Número de identificación [Entero]
- Capacidad [Booleano o Entero]

**RD3: Datos almacenados de Empresa externa o tercero**

- Número de identificación [Entero]
- Número de registro [Entero]
- Nombre Comercial [Cadena de caracteres]
- Domicilio de la compañia [Cadena de caracteres]
- Fecha de fundación [Fecha]
- Objeto de actividad [Cadena de caracteres] empresarial
- Teléfono [Entero]
- FAX [Entero]

**RD4: Datos almacenados de Factura**

- Número de identificación de la factura [Entero]
- Tipo de factura, puede ser de material, personal o servicios(terceros) [Cadena de caracteres]
- Nombre Proveedor [Cadena de caracteres]
- Factura fija [Booleano]
- Fecha límite de pago [Fecha]
- Fecha inicio servicio [Fecha]
- Fecha fin servicio [Fecha]
- Fecha pago automatico [Fecha] (antes fecha límite)
- Cantidad [Flotante]

**RD5: Datos almacenados para el pago de facturas**

- Número de identificación de la factura [Entero]
- Tipo de factura, puede ser de material, personal o servicios(terceros) [Cadena de caracteres]
- Nombre Proveedor [Cadena de caracteres]
- Cantidad [Flotante]

**RD6: Datos almacenados de Personal(Empleado)**

- Número de identificación empleado [Entero]
- DNI [Cadena de caracteres]
- Nombre [Cadena de caracteres]
- Apellidos [Cadena de caracteres]
- Fecha de Nacimiento [Fecha]
- Direccion [Cadena de caracteres]
- Tipo de empleado, puede ser medico, asistente ... [Cadena de caracteres]
- Teléfono [Entero]
- Numero horas [Flotante]
- Turno [Cadena de caracteres]
- Nómina [Flotante]

**RD7: Datos para el alto de un nuevo empleado.**

- Nombre [Cadena de caracteres]
- DNI [Entero]
- Direccion [Cadena de caracteres]
- Telefono [Entero]

- Tipo [Cadena de caracteres]
- Numero horas [Flotante]
- Turno [Cadena de caracteres]

**RD8: Datos para la consulta de un empleado**

- DNI [Entero]

**RD9. Datos almacenados de turno.**

- Nombre [Cadena de caracteres]
- Hora inicio
- Hora fin

**RD10. Datos almacenados de tipo de empleado.**

- Nombre (Medico, Asistencia y tecnicos, Administrativos)
- Especialidad

**RD11: Ambulancia (Vehículo)**

- Modelo de vehiculo [Cadena de caracteres]
- Matricula [Cadena de caracteres]
- Fecha de produccion [Fecha]
- Tipo de ambulancia: no asistencial, asistencial basica, UVI móvil [Cadena de caracteres]
- Kilometros recorridos [Entero]
- Está en ruta [booleano]

**RD12: Datos almacenados de Ruta**

- Ambulancia Asignada [Ambulancia (RD12)]
- Distancia [Flotante]
- Fecha de asistencia [Fecha]
- Periodicidad [Entero]
- Tipo de servicio: Basico, Rescate, UVI móvil [Cadena de caracteres]
- Atendida [booleano]

**RD13: Datos almacenados de Paciente** Persona que usa el servicio. Se describe mediante:

- Número de tarjeta sanitaria [Entero]
- Fecha de nacimiento [Fecha]
- DNI [Entero]
- RD15 asignado
- Habitación en planta [Cadena de caracteres]
- Sala de servicio [Cadena de caracteres]

**RD14: Datos almacenados de Cita** Es una tupla que indica una fecha en el calendario. Se describe mediante:

- Hora
- Día
- Mes
- Año

**RD15: Datos almacenados de Especialidad** Almacena los tipos de especialidades hospitalarias que se ofrecen. Se describe mediante:

- Tipo de especialidad [Cadena de caracteres]

**RD16: Datos almacenados de Médico** Persona que ofrece un servicio médico en el hospital. Se describe mediante:

- DNI [Entero]
- Nombre [Cadena de caracteres]
- Apellido 1 [Cadena de caracteres]
- Apellido 2 [Cadena de caracteres]
- Número identificador [Entero]
- Especialidad [Cadena de caracteres]

**RD17: Datos almacenados de Sala**

- Número de identificación [Entero]
- Tipo pueden ser: habitación, sala de operaciones o UCI. [Cadena de caracteres]
- Ocupado [Booleano]

# 5. Restricciones Semánticas

**RS1: Tipo de material:** El material puede ser clínico, farmacéutico, maquinaria o alimenticio.

**RS2: Actividad de empresa externa:** La actividad de las empresas externas pueden ser dar servicio o ser un proveedor.

**RS3: Tipo de Factura** Una factura pueden ser de material, un servicio o de personal.

**RS4: Cantidad de Materiales:** Solo se pueden eliminar materiales si la cantidad de este es 0.

**RS5: Tipo de personal** Un empleado puede tener un puesto determinado.

**RS6: Condición de Stock** Un material puede estar en un almacen o no, por lo que si está en algun almacen se considera que hay stock.

**RS7: Capacidad de un almacen** Un almacen tiene una determinada capacidad (puede ser representada como un booleando verificando si el almacen esta completo o no o un entero con un número determinado de objetos).

**RS8. Tipo de emleado** Cada empleado pertenece a un único tipo.

**RS9. Número de turnos** Un empleado pertenece a uno o más turnos.

**RS10: Médico de cabecera único** Cada paciente tiene asociado un único médico de cabecera.

**RS11: Número de citas** Cada paciente sólo puede pedir una cita con un médico de cabecera cada vez. Es decir, hasta pasada la primera cita, no puede pedir la siguiente.

**RS12: Número de pacientes** Cada médico de cabecera puede tener asociados varios pacientes.

**RS13: Igualdad de citas** No puede haber dos citas iguales (misma fecha y hora) con el mismo médico.

**RS14: Solicitud de citas** Un paciente puede pedir tantas citas como desee, con diferentes médicos de especialidad.

**RS15: Solicitud de consulta a otro médico** Un paciente, aunque tenga asignado un médico de cabecera, puede pedir una consulta con otro médico de medicina general.

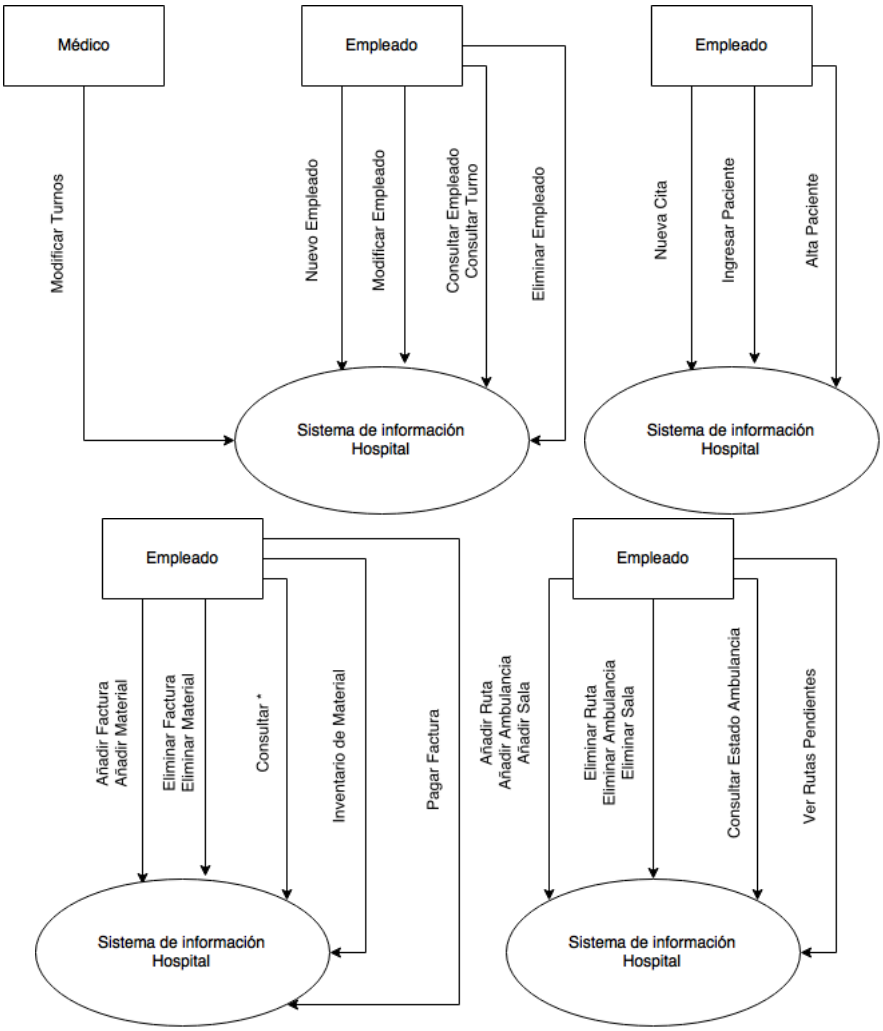
**RS16: Tipo de Sala** Una sala puede pertenecer a diferentes tipos, habitación, UCI o sala de operación.

**RS17: Número de habitaciones por paciente** Un paciente puede estár asignado a dos habitaciones y además puede tener una sala donde se este aportando el servicio necesario para el mismo.

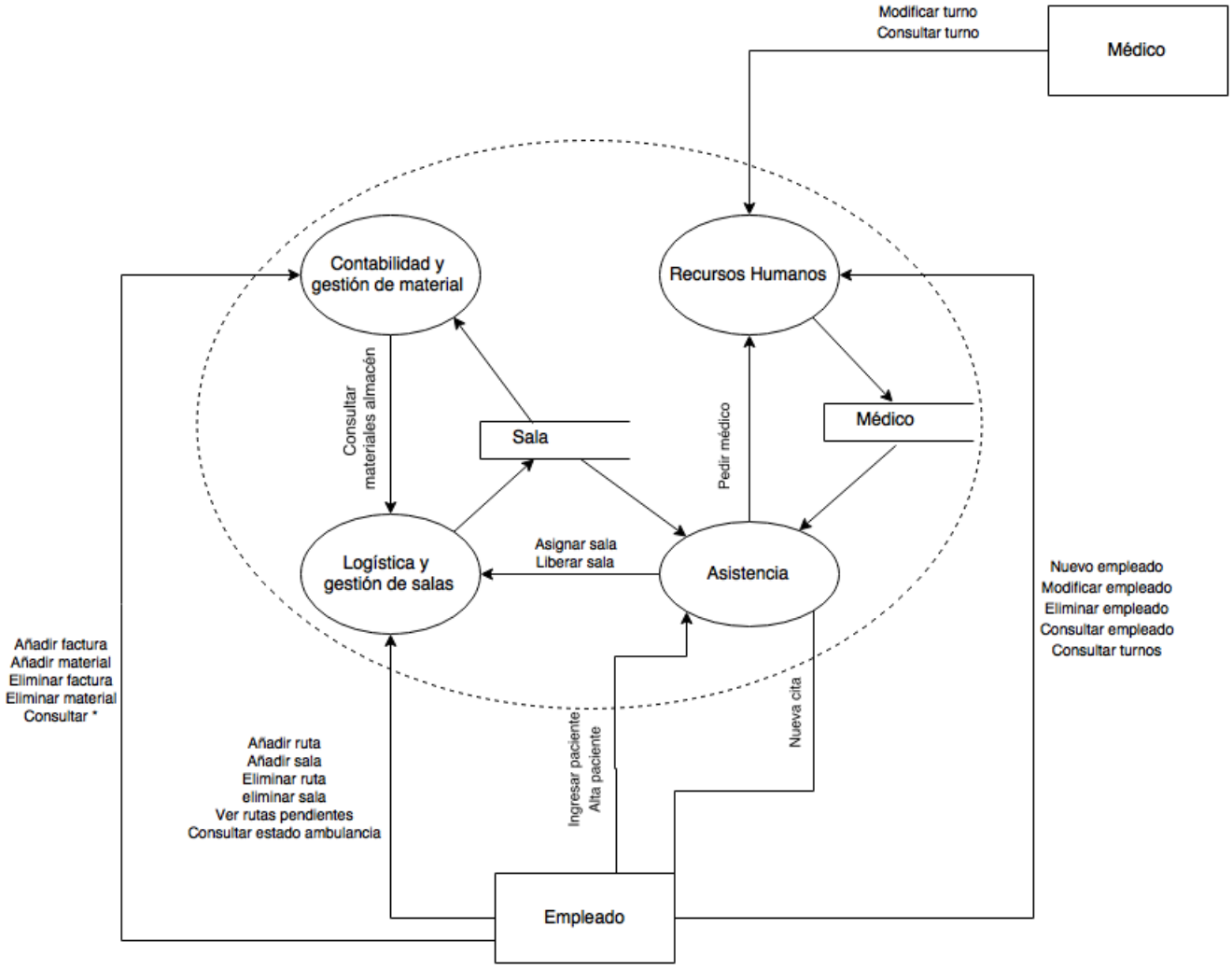
**RS18: Dependencia Materiales:** No se puede eliminar un tercero ni una sala si esta asociado a un material.

# 6. Diseño

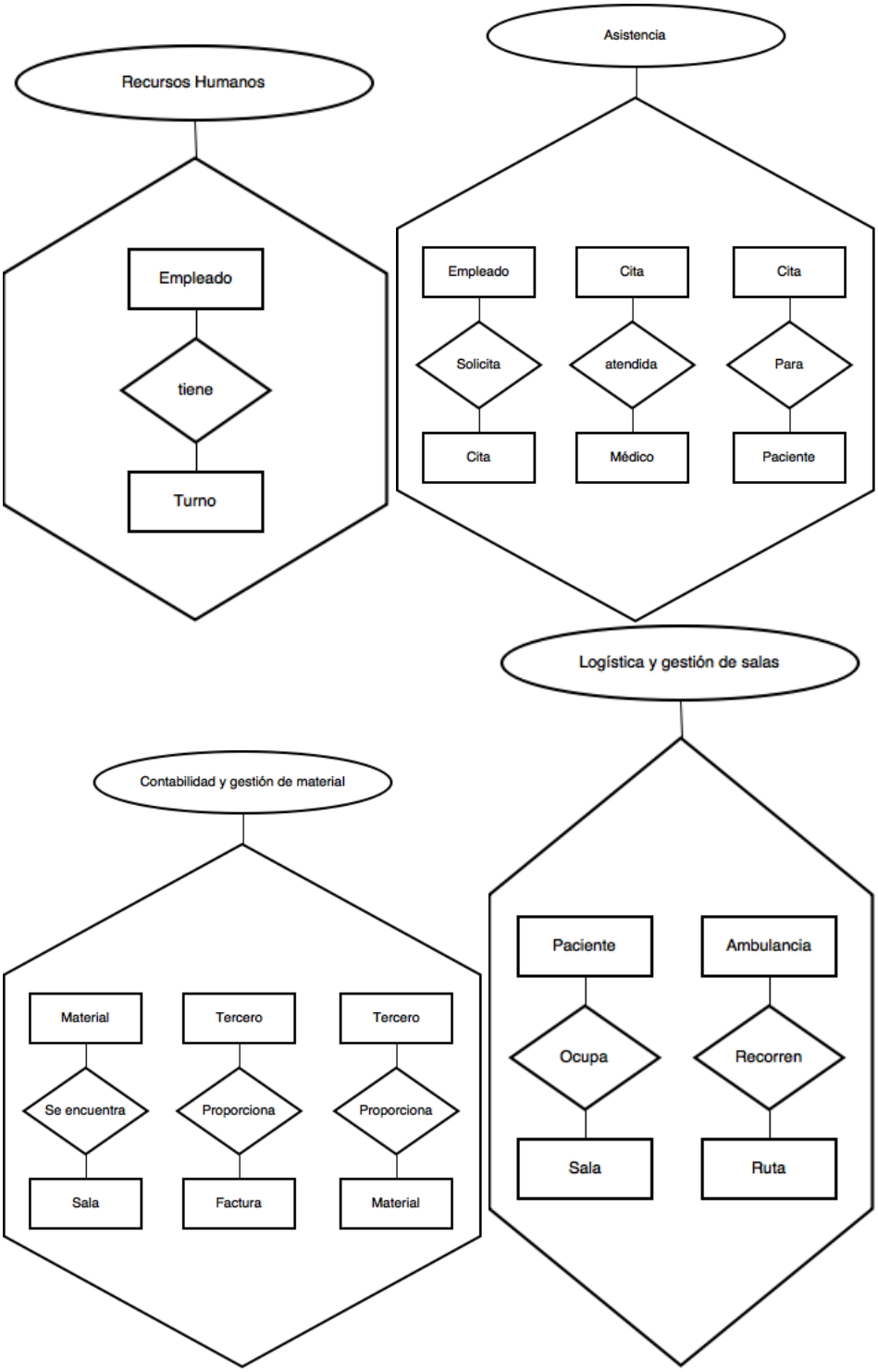
## 6.1. Diagrama armazón inicial

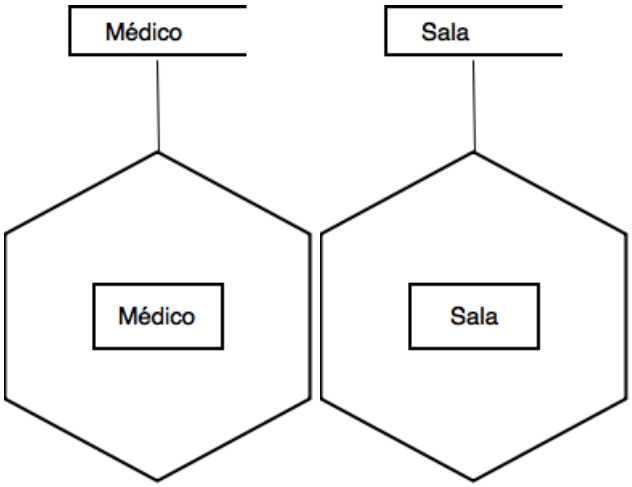


## 6.2. Diagrama armazón F

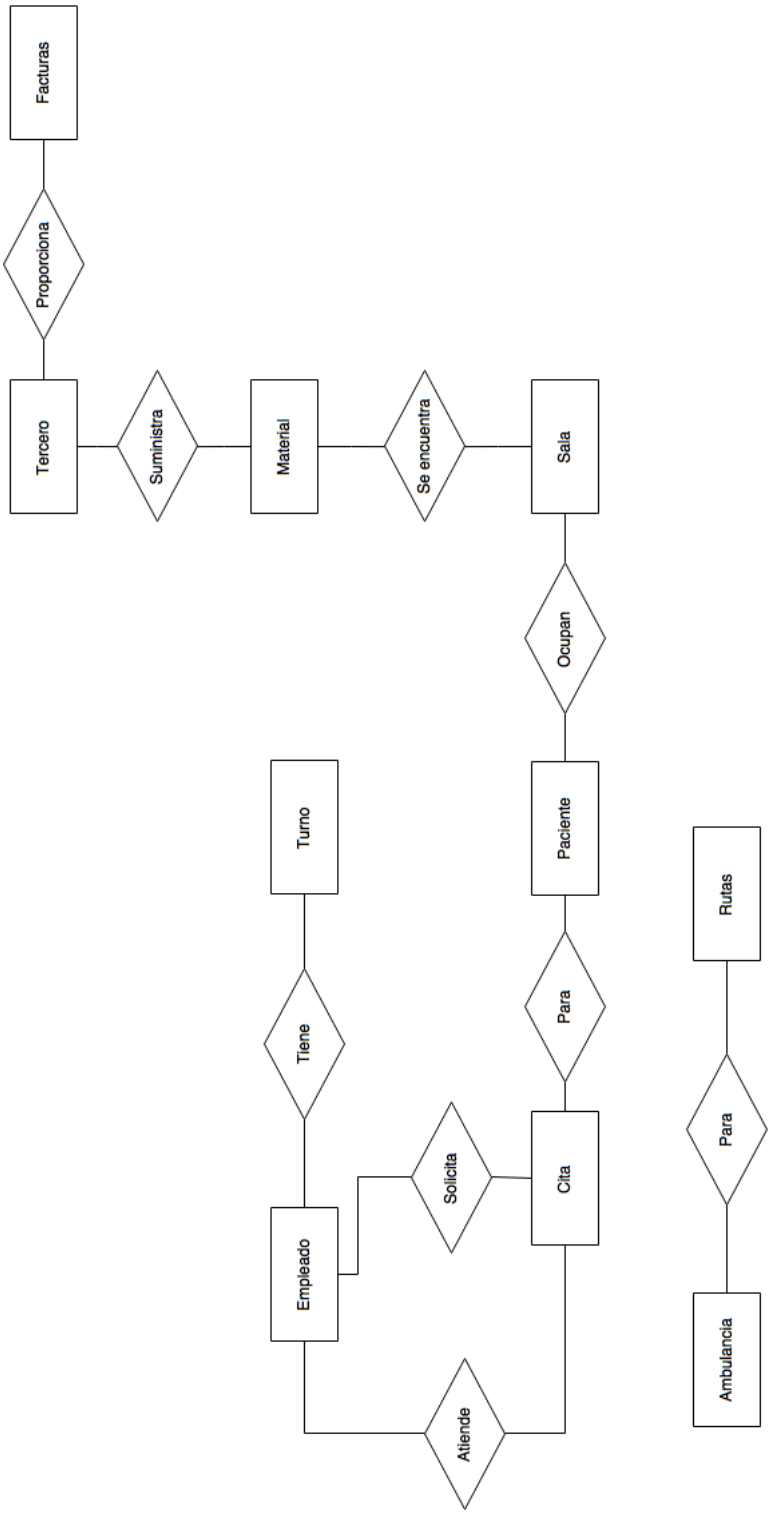


## 6.3. Esquemas externos para el diagrama armazón F



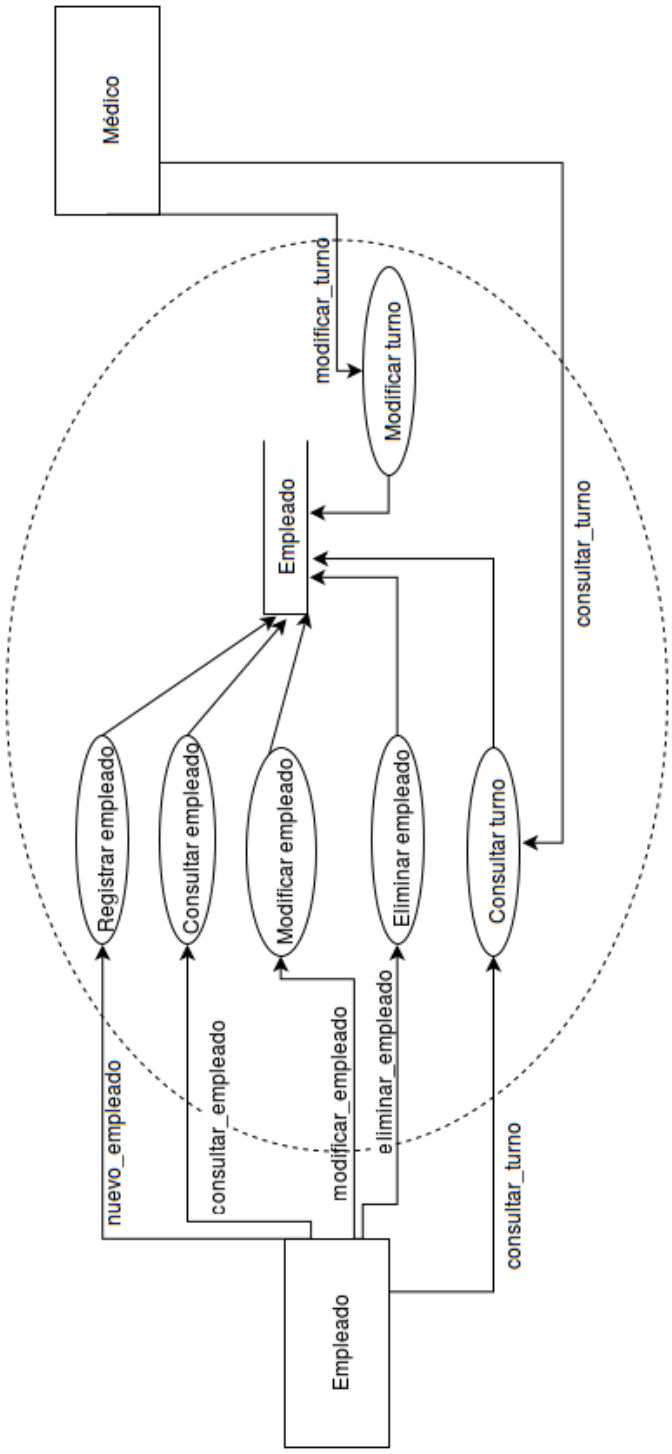

Médico
Sala
Médico
Sala

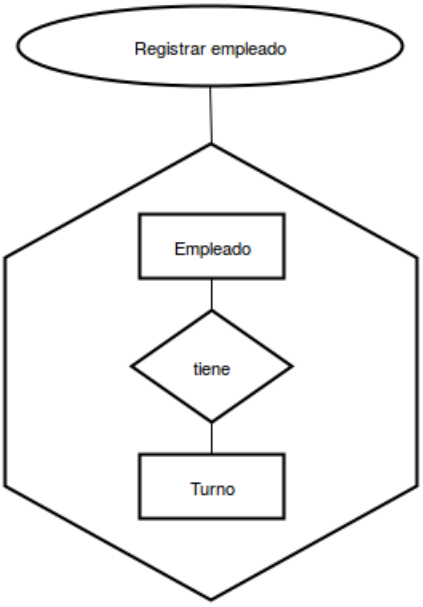
## 6.4. Diagrama armazón D

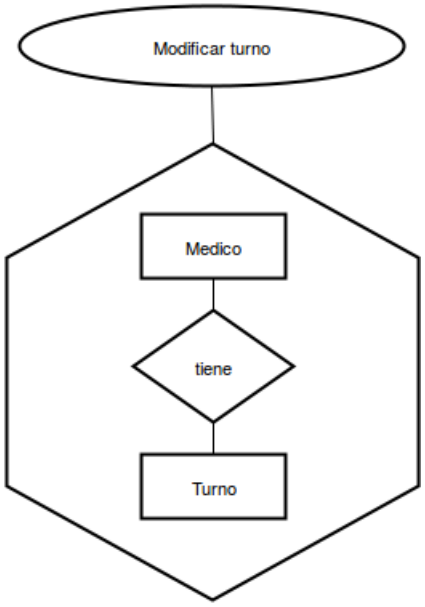


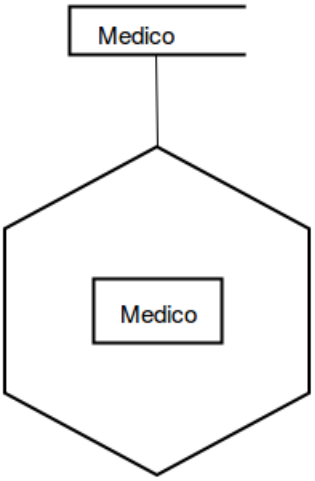
## 6.5. Primer Refinamiento

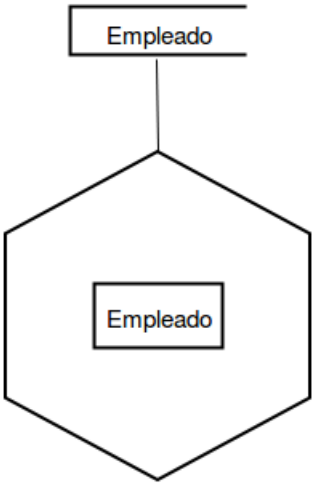
### 6.5.1. Refinamiento parcial de recursos humanos



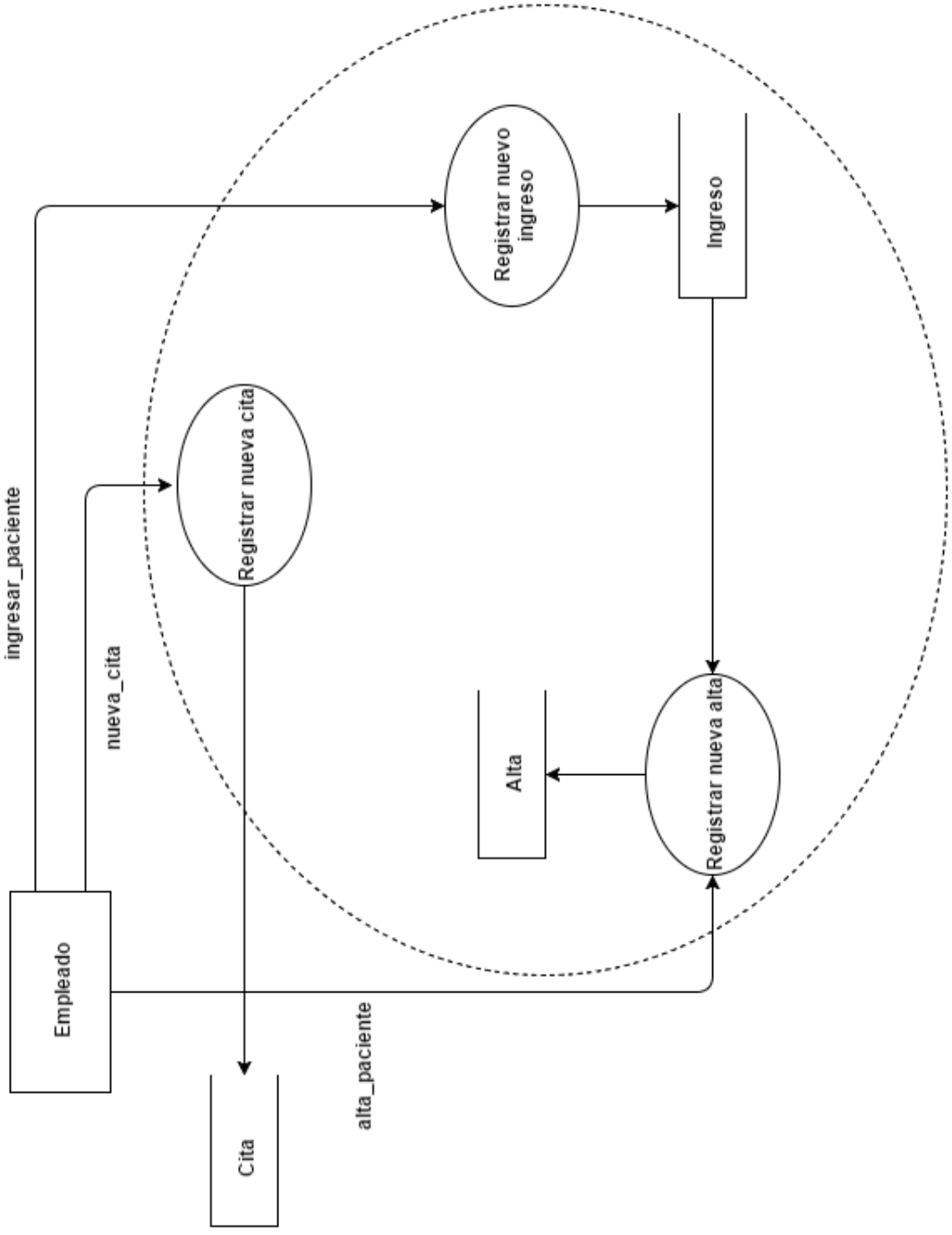

Registrar empleado
Empleado
tiene
Turno

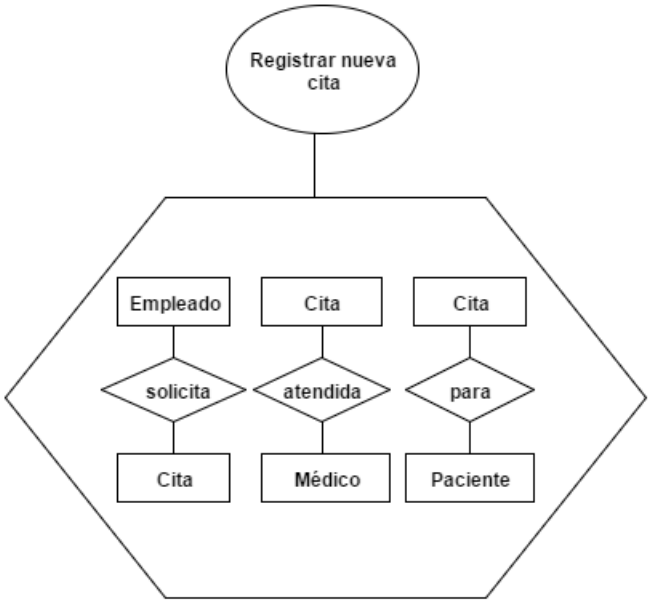

Modificar turno
Medico
tiene
Turno

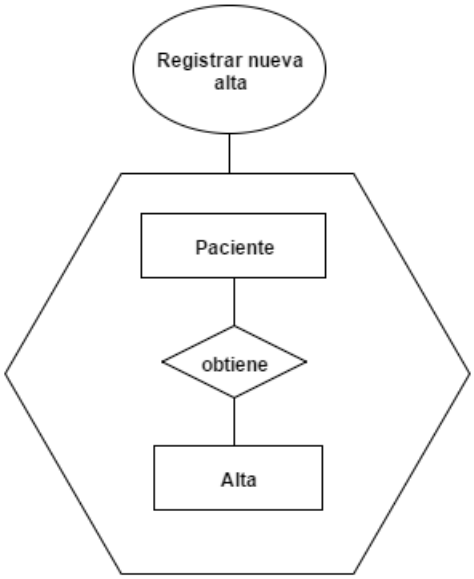

Medico
Medico

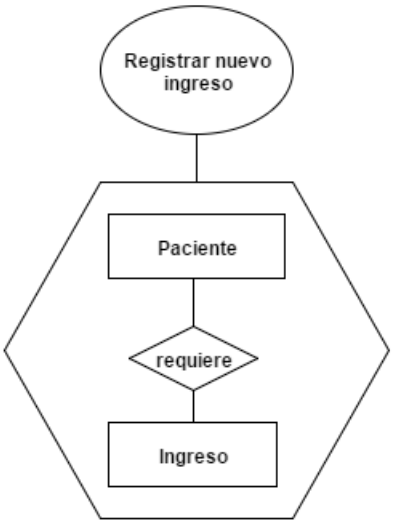

Empleado
Empleado

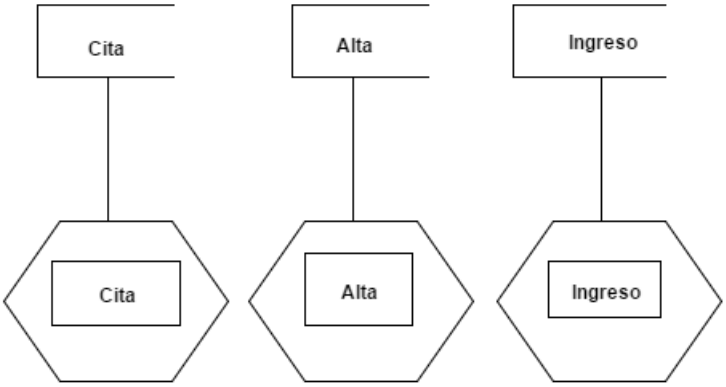
### 6.5.2. Refinamiento parcial de asistencia



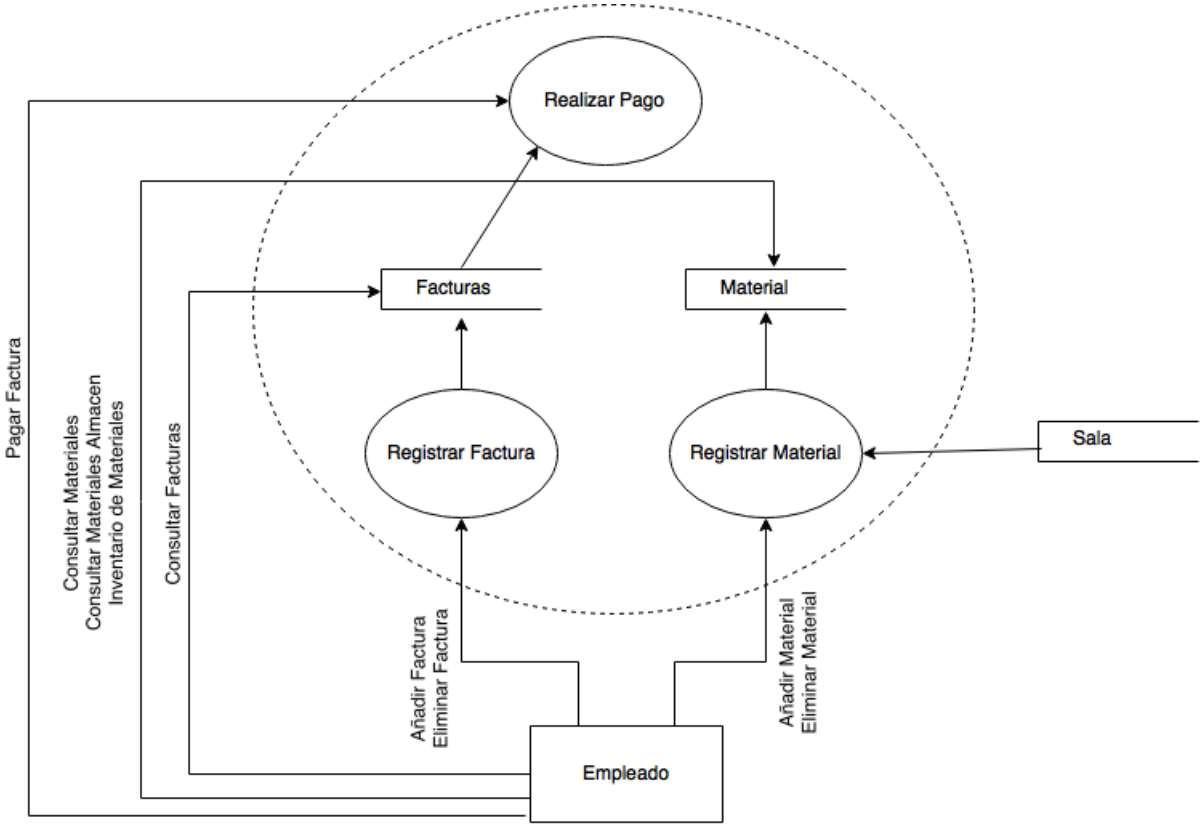

Registrar nueva cita
Empleado
Cita
Cita
solicita
atendida
para
Cita
Médico
Paciente

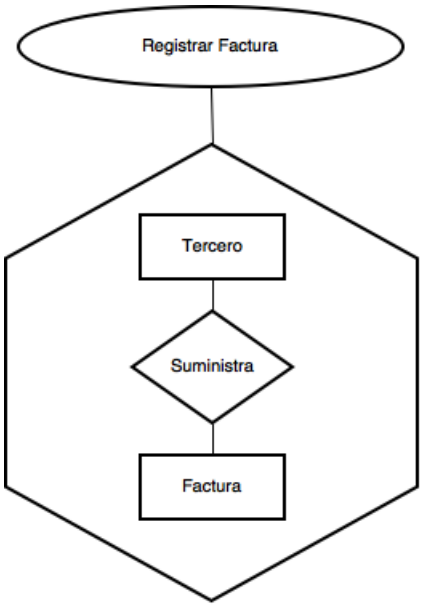

Registrar nueva alta
Paciente
obtiene
Alta

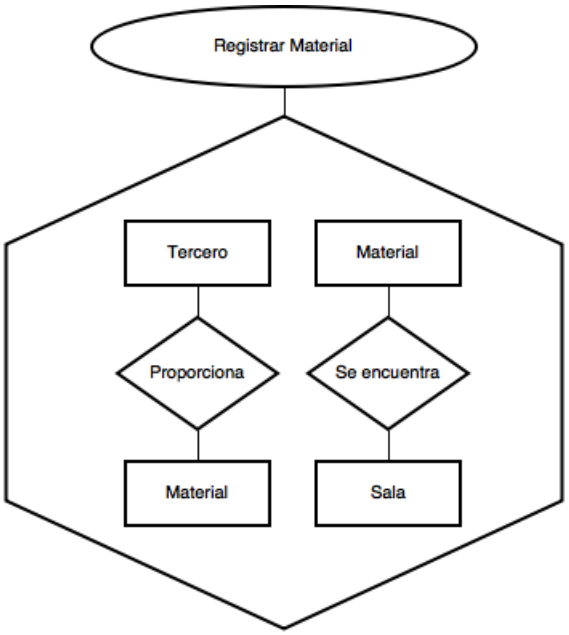

Registrar nuevo ingreso
Paciente
requiere
Ingreso

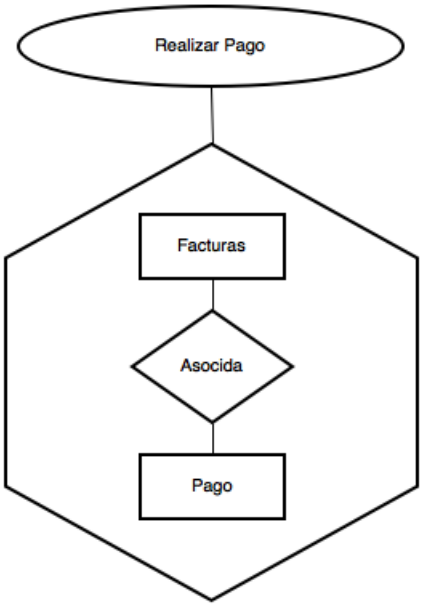

Cita
Alta
Ingreso
Cita
Alta
Ingreso

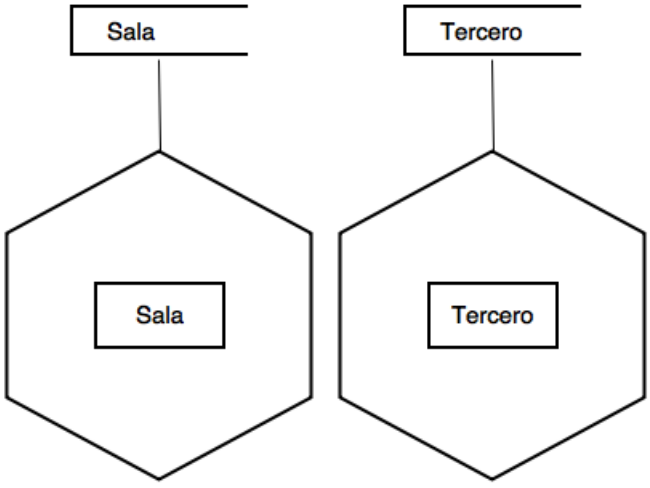
### 6.5.3. Refinamiento parcial de contabilidad y gestión de material

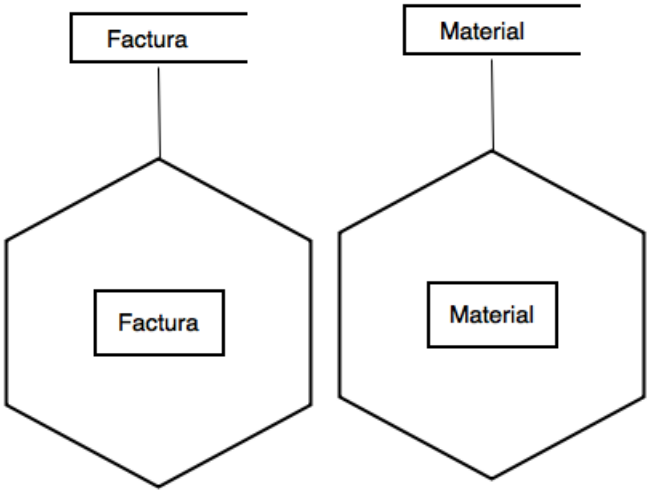



Registrar Factura
Tercero
Suministra
Factura

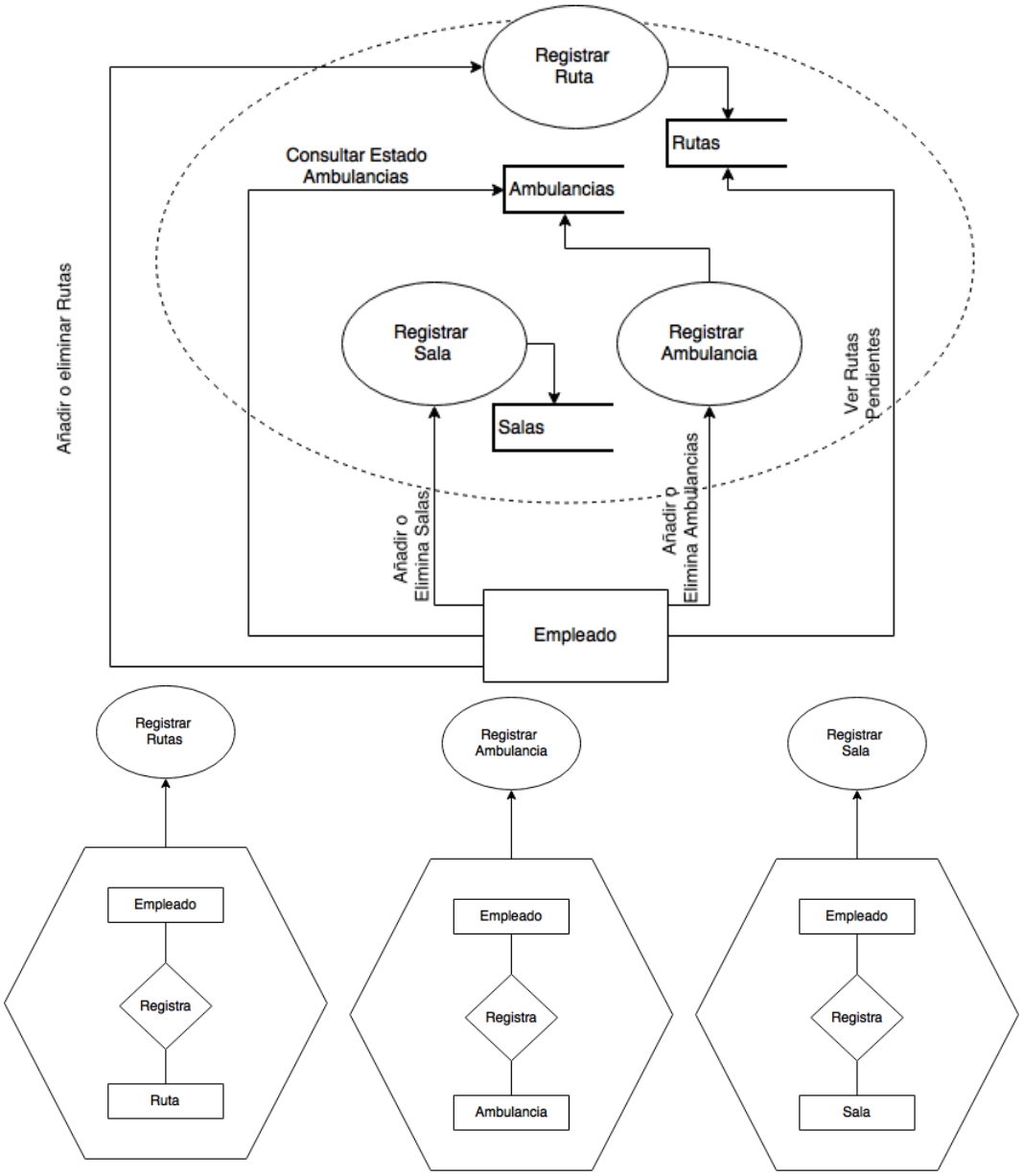

Registrar Material
Tercero
Material
Proporciona
Se encuentra
Material
Sala


Realizar Pago
Facturas
Asocida
Pago

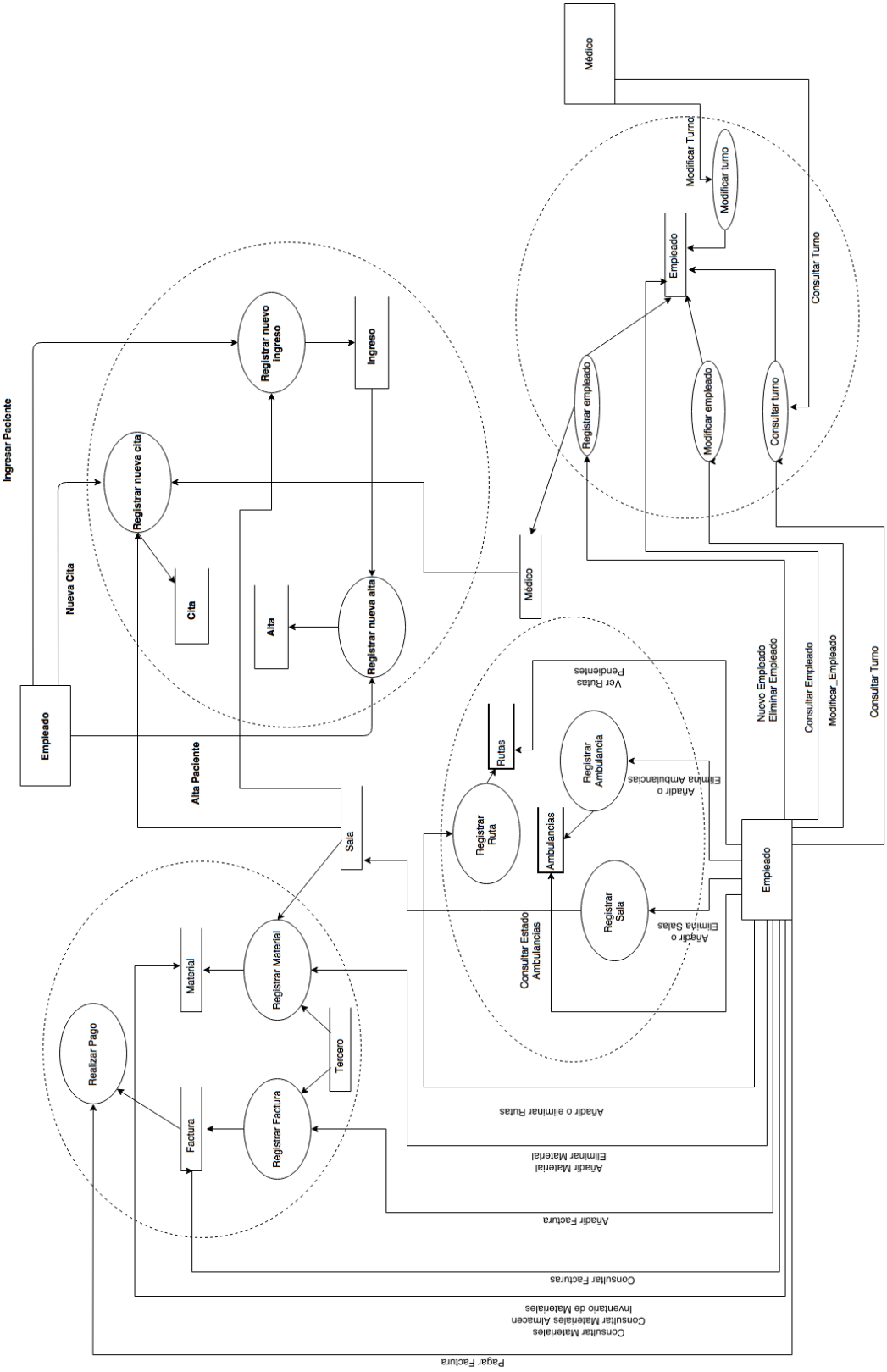

Sala
Sala
Tercero
Tercero


Factura
Factura
Material
Material

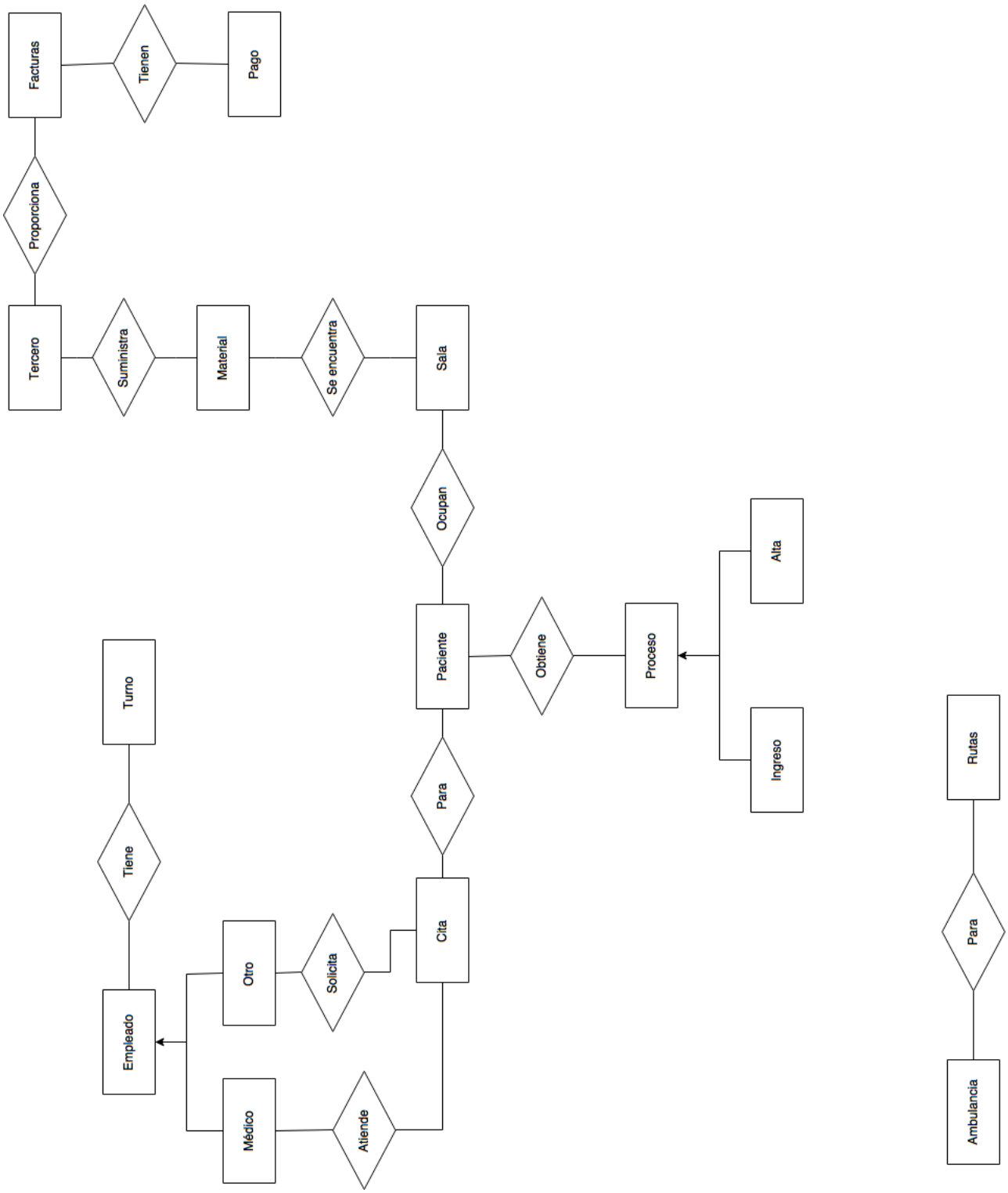
### 6.5.4. Refinamiento parcial de logística y gestion de salas



### 6.5.5. Diagrama armazón F refinado

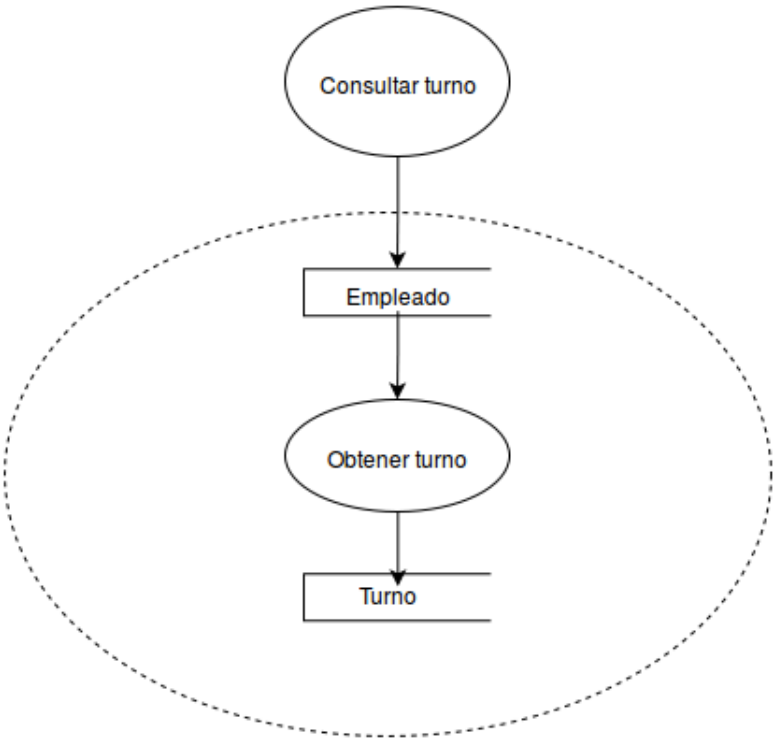


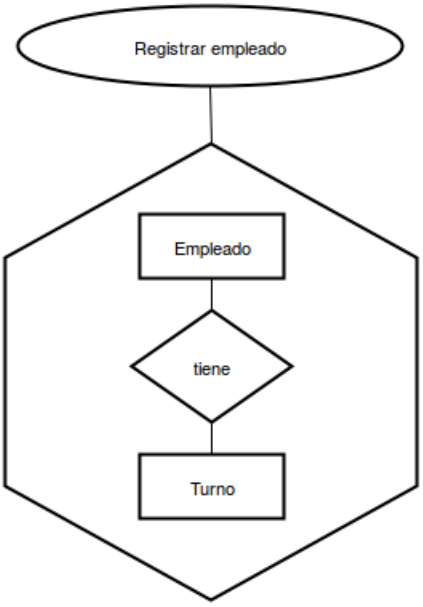
### 6.5.6. Diagrama armazón D

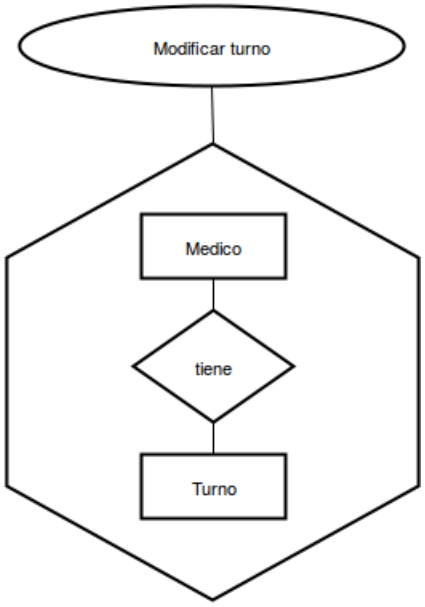


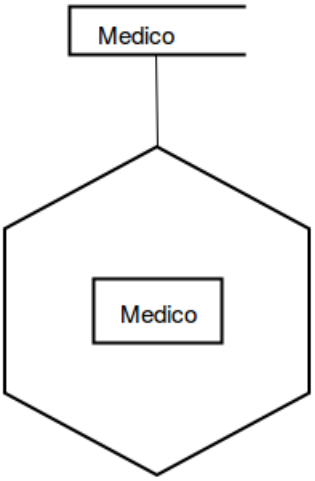
## 6.6. Segundo Refinamiento

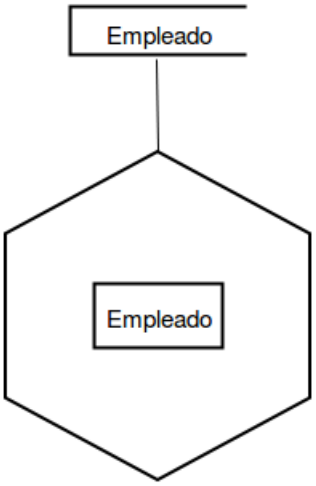
### 6.6.1. Refinamiento parcial de recursos humanos



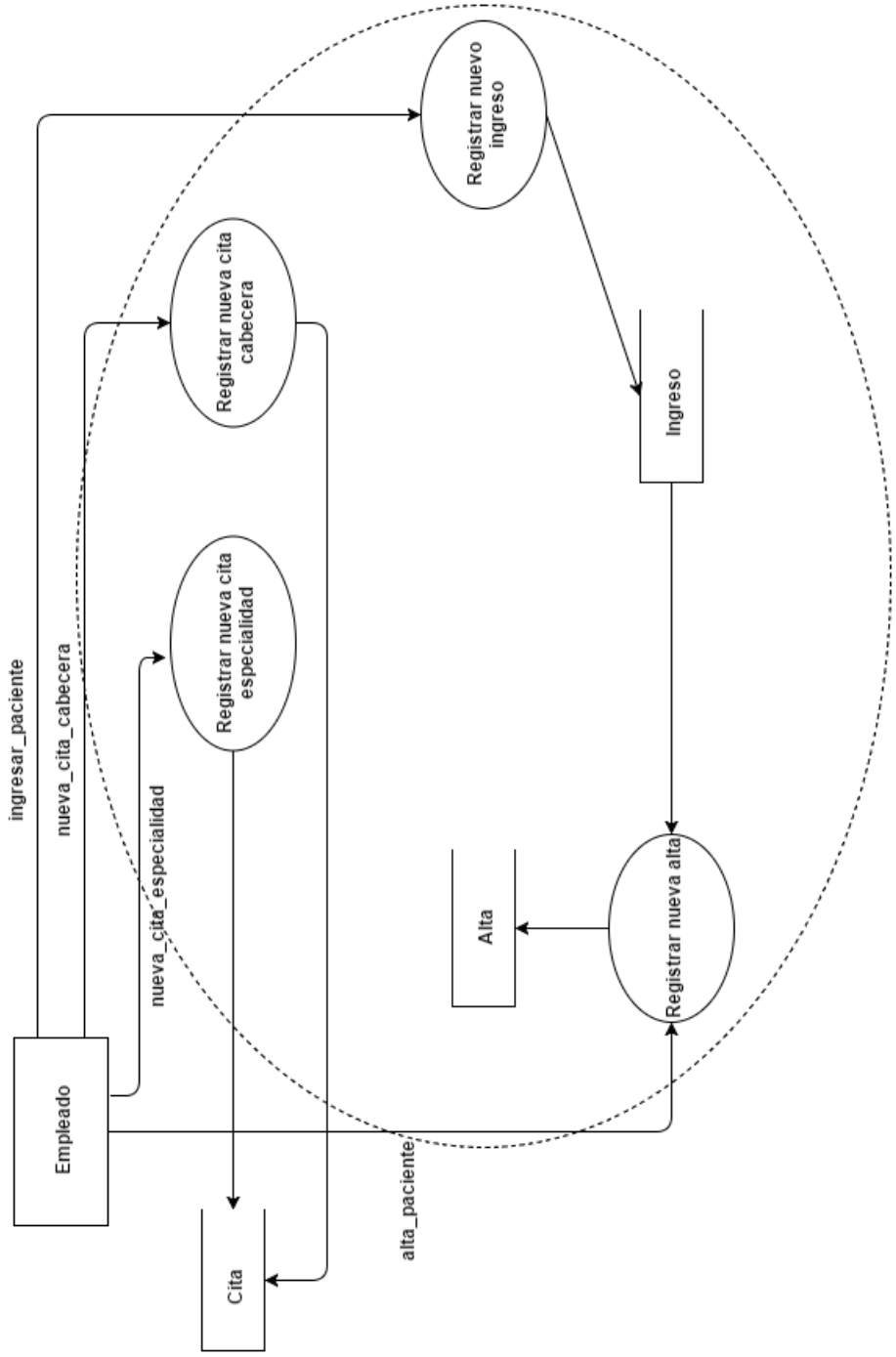

Registrar empleado
Empleado
tiene
Turno

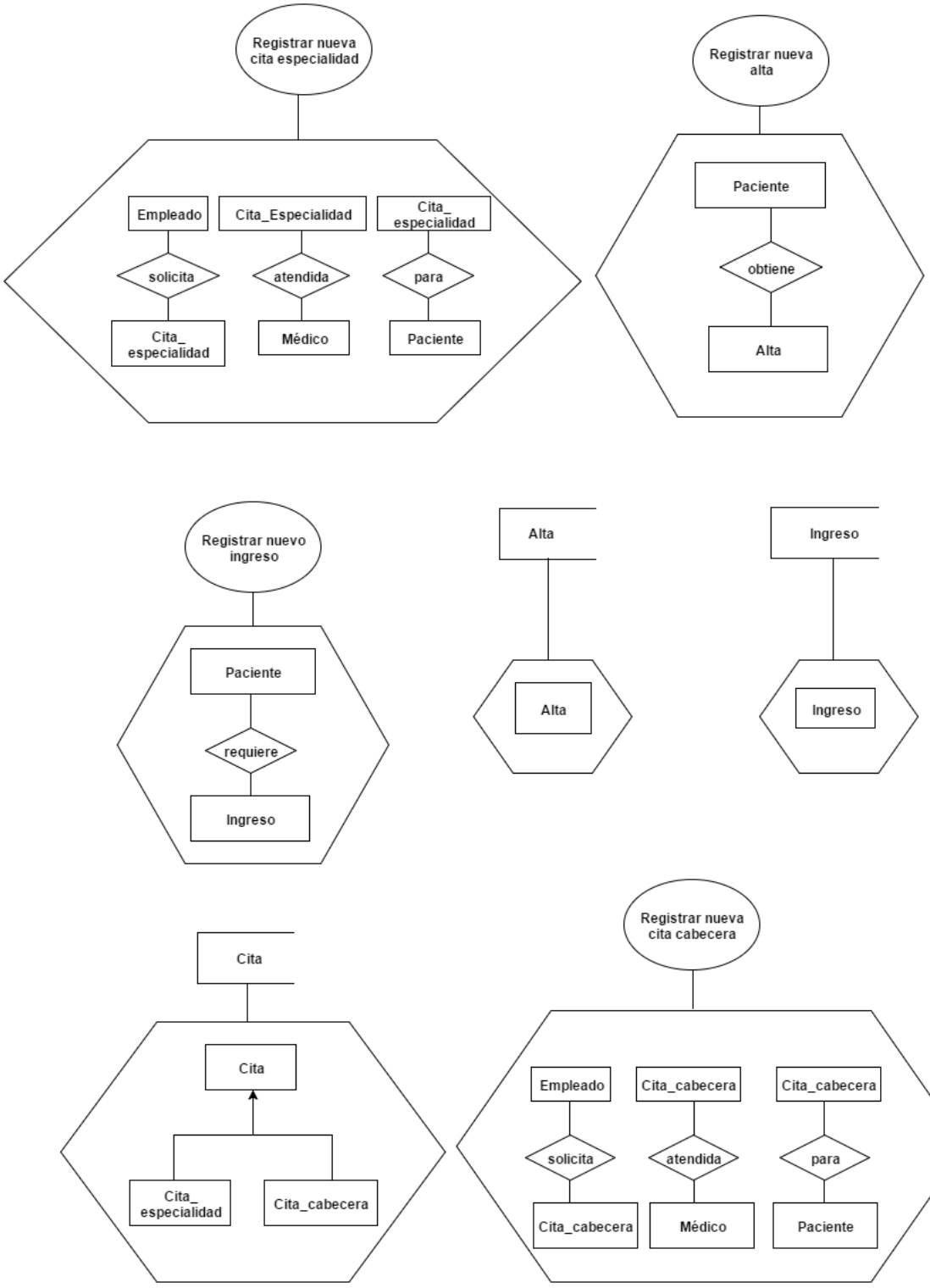

Modificar turno
Medico
tiene
Turno


Medico
Medico

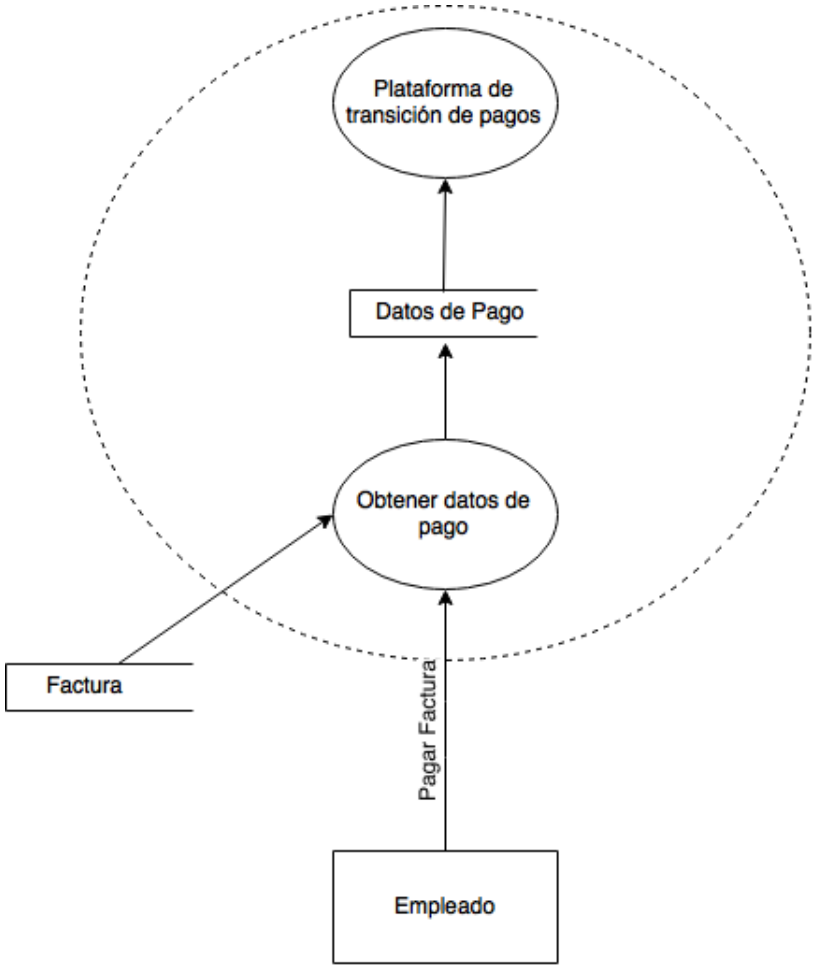

Empleado
Empleado

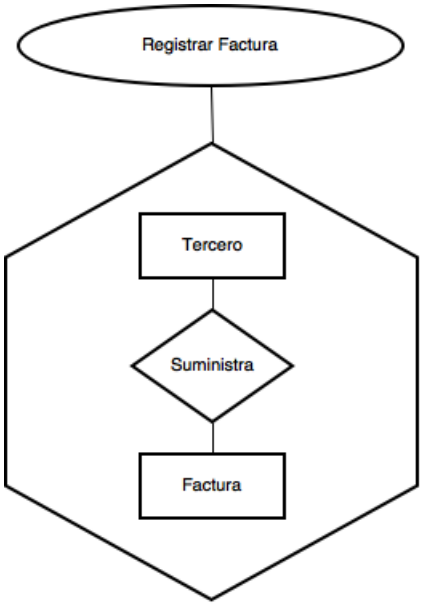
### 6.6.2. Refinamiento parcial de asistencia

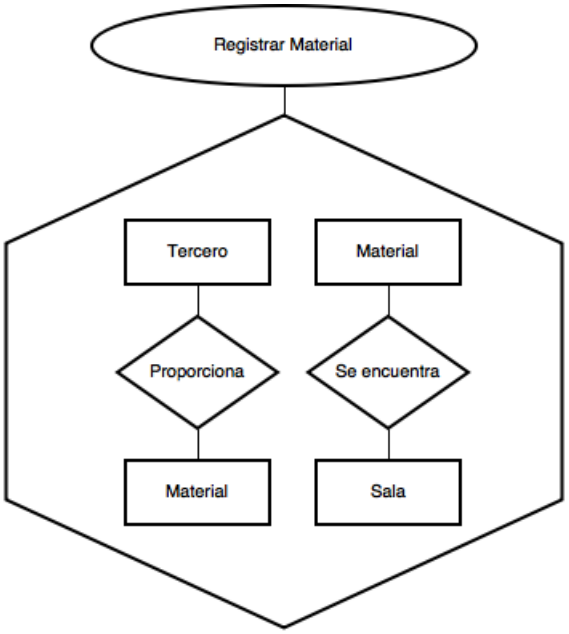


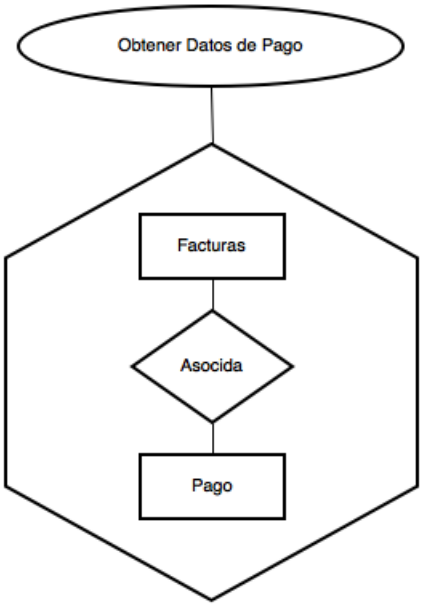

Registrar nueva cita especialidad
Empleado
Cita_Especialidad
Cita_ especialidad
solicita
atendida
para
Cita_ especialidad
Médico
Paciente
Registrar nueva alta
Paciente
obtiene
Alta
Registrar nuevo ingreso
Paciente
requiere
Ingreso
Alta
Alta
Ingreso
Ingreso
Cita
Cita
Cita_ especialidad
Cita_cabecera
Registrar nueva cita cabecera
Empleado
Cita_cabecera
Cita_cabecera
solicita
atendida
para
Cita_cabecera
Médico
Paciente

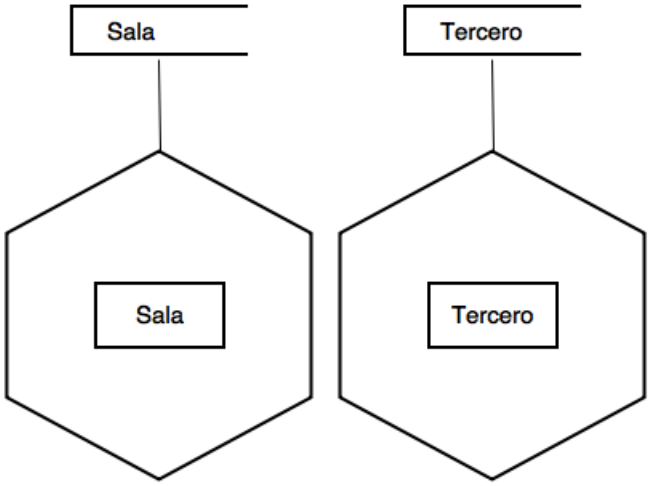
### 6.6.3. Refinamiento parcial de contabilidad y gestión de material

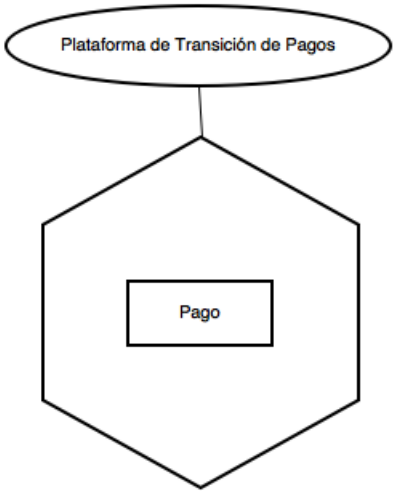


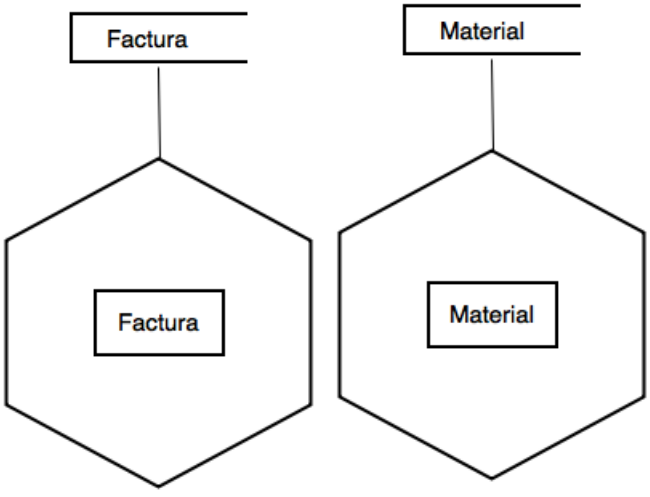

Registrar Factura
Tercero
Suministra
Factura


Registrar Material
Tercero
Material
Proporciona
Se encuentra
Material
Sala

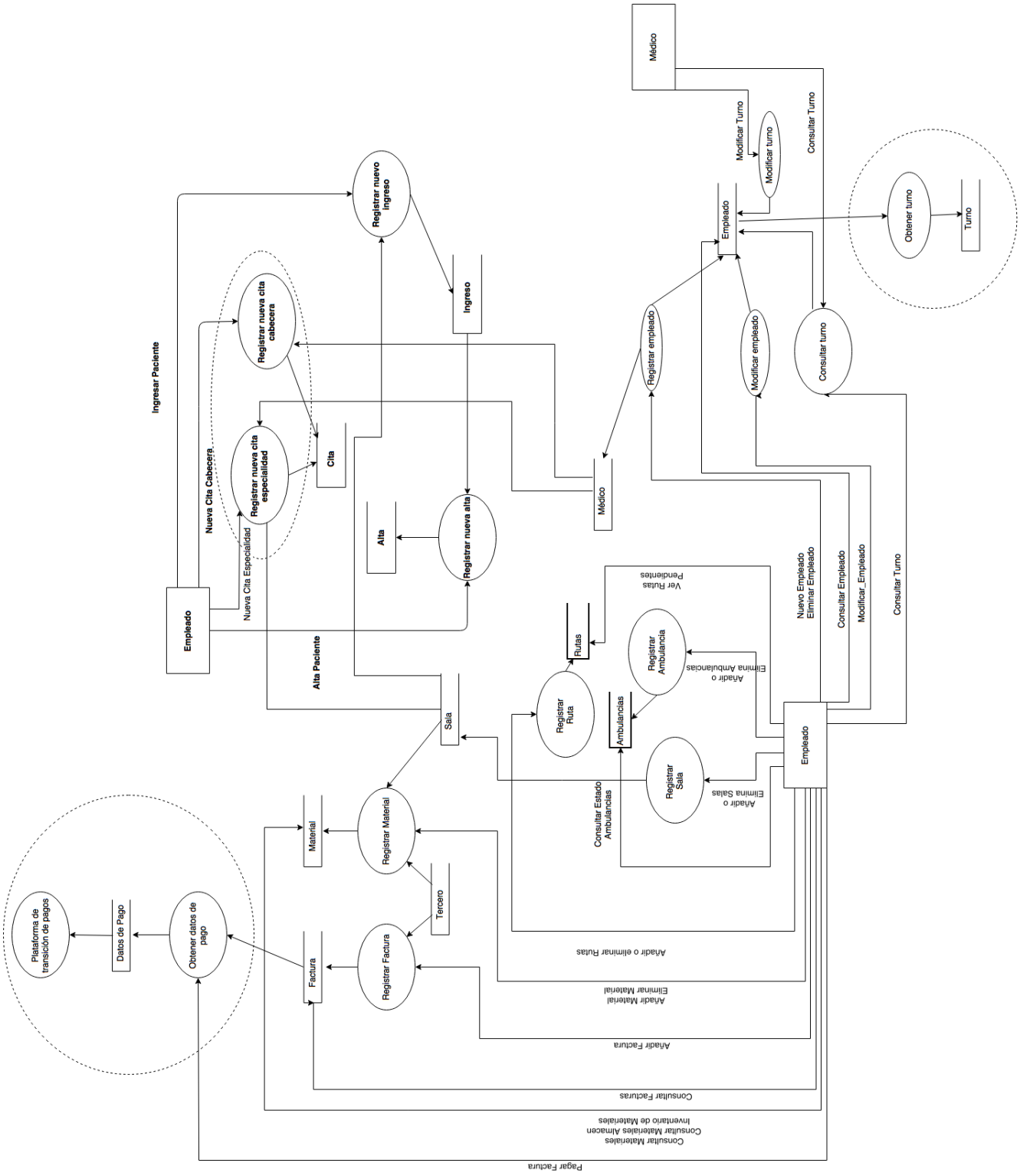

Obtener Datos de Pago
Facturas
Asocida
Pago


Sala
Sala
Tercero
Tercero

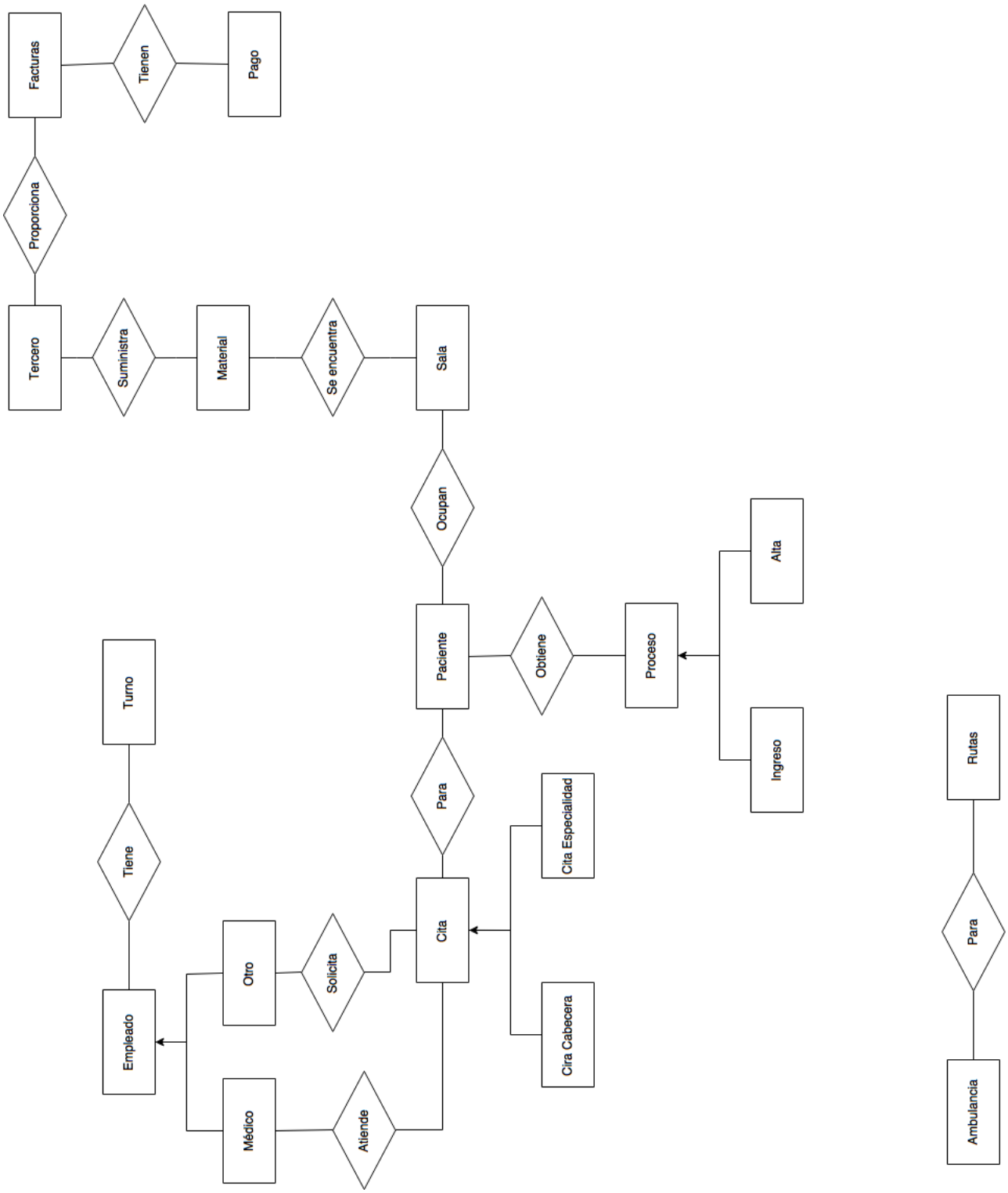

Plataforma de Transición de Pagos
Pago


Factura
Factura
Material
Material

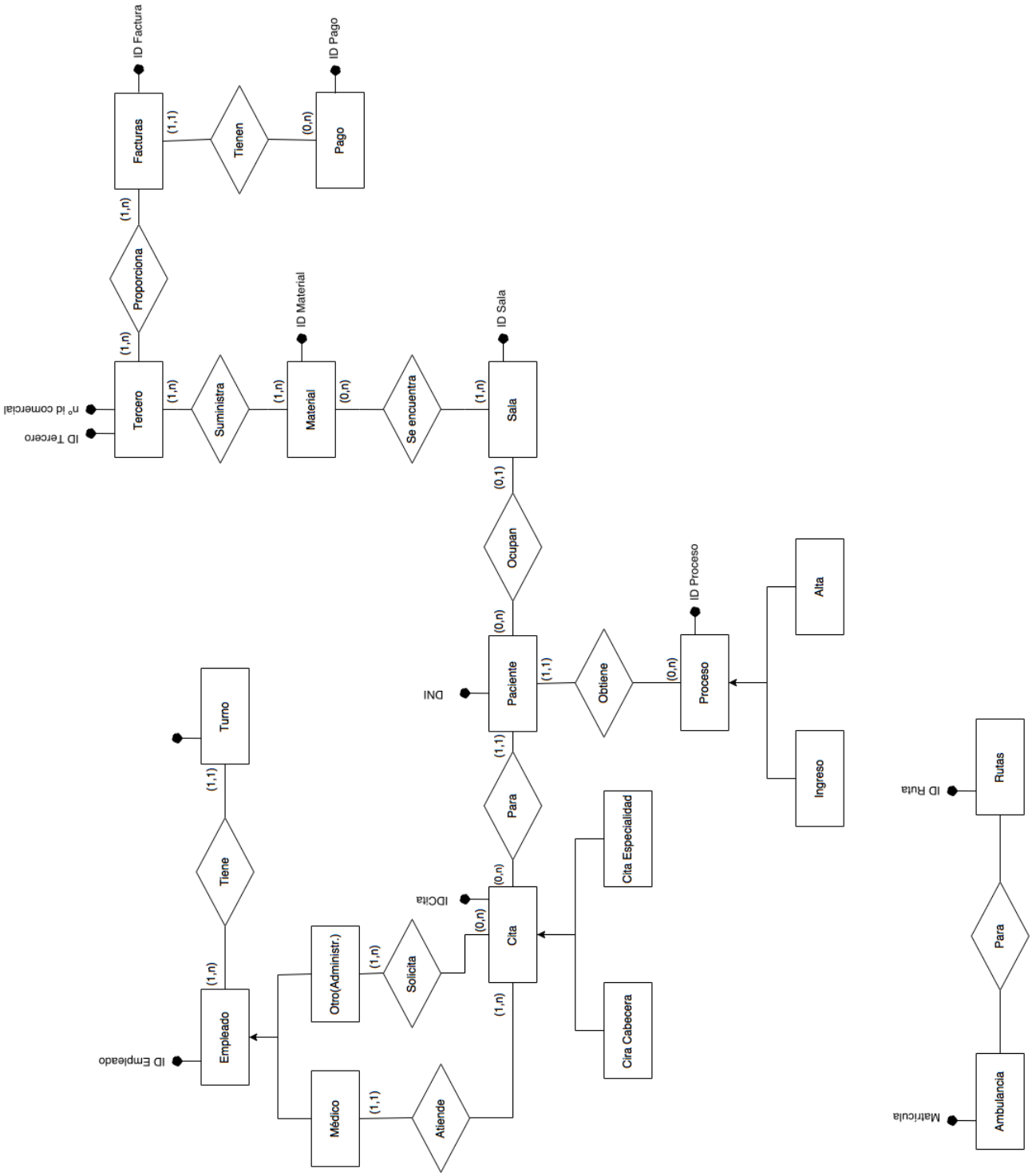
### 6.6.4. Diagrama armazón F final



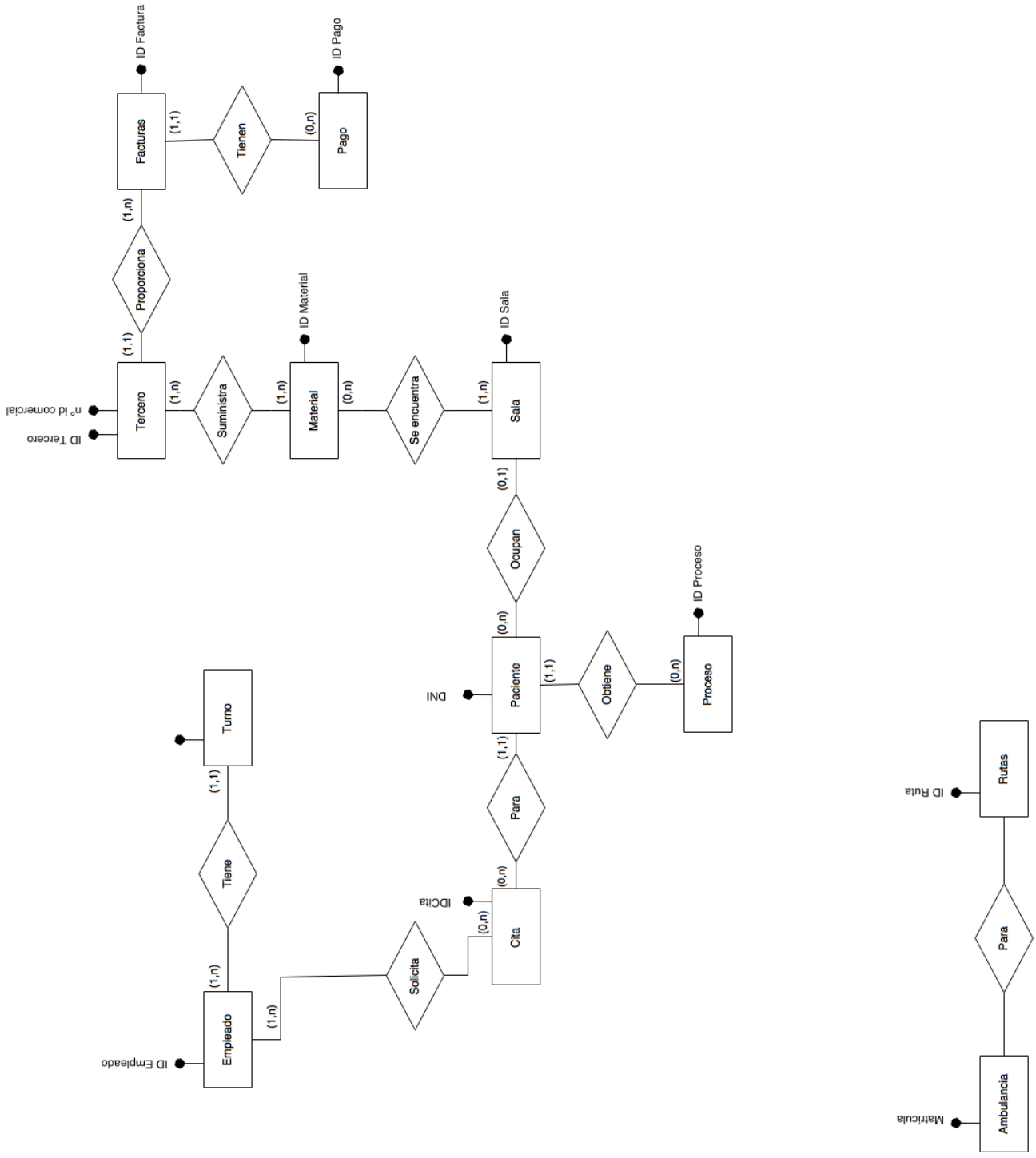
### 6.6.5. Diagrama armazón D



# 7. Diagrama E/R. Diagrama Conceptual inicial



# 8. Diagrama E/R. Diagrama Conceptual completo



# 9. Operaciones de Datos

Las operaciones con datos se han llevado a cabo usando funciones PL/SQL aunque se podrían haber usado procedimientos el motivo viene detallado en la sección de descripción de la solución implementada, aun así el código en PL/SQL es muy básico, procedemos a explicar cual han sido nuestras operaciones:

## 9.1. Recursos humanos

### 9.1.1. Inserción

```
CREATE OR REPLACE EDITIONABLE FUNCTION "X4742492"."INSERTAEMPLEADO" (v_nombre in VARCHAR,
v_apellidos in VARCHAR, v_dni in VARCHAR, v_direccion in VARCHAR,
v_tipo in VARCHAR, v_tlf in NUMBER, v_nh in NUMBER, v_nom in NUMBER,v_turno in VARCHAR2)
RETURN NUMBER AS
v_result number := "X4742492"."EMPLEADO_SEQ".nextval+1;
BEGIN
INSERT INTO EMPLEADO (nombre,apellidos,dni,direccion,tipo,telefono,horas,NOMINA,TURNO) VALUES
(v_nombre,v_apellidos,v_dni,v_direccion,v_tipo,v_tlf,v_nh,v_nom,v_turno);
RETURN v_result;
END INSERTAEMPLEADO;
```

### 9.1.2. Borrado

```
CREATE OR REPLACE EDITIONABLE FUNCTION "X4742492"."BORRAEMPLEADO" (V_ID IN NUMBER )
RETURN NUMBER AS
v_result number := 0;
BEGIN
DELETE FROM EMPLEADO WHERE ID = V_ID;
v_result := 1;
RETURN v_result;
END BORRAEMPLEADO;
```

### 9.1.3. Consulta

Para las consultas se ha usado sentencias SQL, tales como

```
SELECT * FROM EMPLEADOS
```

para obtener los empleados existentes en la base de datos.

### 9.1.4. Modificado

Se hace uso de 2 procedimientos para modificar tuplas de la base de datos. Uno de ellos para cambiar el turno de un empleado, y otro para modificar los datos. El código de ambos es el siguiente:

```
create or replace FUNCTION CAMBIATURNO (V_TURNO IN VARCHAR2 , V_ID IN NUMBER )
RETURN NUMBER AS
v_result number := 0;
BEGIN
UPDATE EMPLEADO SET TURNO=V_TURNO WHERE ID=V_ID;
v_result := 1;
```

```
RETURN v_result;
END CAMBIATURNO;

create or replace FUNCTION MODIFICAREMPLEADO(V_ID IN NUMBER,v_nombre in VARCHAR,v_apellidos in VAR
RETURN NUMBER AS
v_result number :=0;
BEGIN
UPDATE EMPLEADO SET NOMBRE=v_nombre,APELLIDOS=v_apellidos,dni=v_dni,DIRECCION=v_direccion,tipo=v_t
WHERE ID=V_ID;
v_result := 1;
RETURN v_result;
END MODIFICAREMPLEADO;
```

## 9.2. Asistencia

Las operaciones en esta funcionalidad van a consistir basicamente insertar, consultar y borrar Citas.

### 9.2.1. Inserción

```
create or replace FUNCTION REGISTRARCITA(V_ID IN NUMBER, V_POS IN NUMBER) RETURN NUMBER AS
v_result number := 0;
BEGIN
  INSERT CITA (HORA,FECHA) VALUES (...)
  UPDATE EMPLEADO SET
    CITA1= CASE WHEN v_pos=0 THEN 0 ELSE CITA1 END,
    CITA2= CASE WHEN v_pos=1 THEN 0 ELSE CITA2 END,
    CITA3= CASE WHEN v_pos=2 THEN 0 ELSE CITA3 END,
    CITA4= CASE WHEN v_pos=3 THEN 0 ELSE CITA4 END
   WHERE  ID=V_ID;
   v_result := 1;
  RETURN v_result;
END REGISTRARCITA;
```

### 9.2.2. Consulta

Para las consultas no se ha usado PL/SQL. Se han ejecutado sobre la base de datos las siguientes operaciones:

`SELECT * FROM CITA, PACIENTE WHERE (PACIENTE.CITA_ASIGNADA = CITA.ID)`

Se usa para recuperar todas las citas y pacientes en las cuales coincidan los identificativos de las citas y representarlas en una tabla en el programa.

`SELECT * FROM PACIENTE`

Se usa para recuperar todos los pacientes y representarlos en una tabla en el programa.

`SELECT * FROM SALA`

Se usa para recuperar todas las salas y representarlas en una tabla en el programa.

`SELECT * FROM Medico where(Medico.Especialidad...`

Se usa para recuperar todos los médicos de una determinada Especialidad y representarlos en una tabla en el programa.

`SELECT * FROM Medico`

Se usa para recuperar todos los médicos y representarlos en una tabla en el programa.

## 9.3. Contabilidad y gestión de material

Las operaciones en esta funcionalidad consisten en basicamente insertar, consultar y borrar facturas y materiales salvo la operación pagar facturas que crea un pago en el sistema y elimina la factura considerando que esta ya se ha pagado.

### 9.3.1. Consulta

Para las consultas no se ha usado PL/SQL se ha ejecutado sobre la base de datos las siguientes operaciones:

`SELECT * FROM FACTURA`

Se usa para recuperar todas las facturas y representarlas en una tabla en el programa.

`SELECT * FROM TERCERO`

Se usa para recuperar todos los terceros y representarlos en una tabla en el programa.

`SELECT * FROM PAGO`

Se usa para recuperar todos los pagos y representarlos en una tabla en el programa.

`SELECT * FROM MATERIAL`

Se usa para recuperar todos los materiales y representarlos en una tabla en el programa.

`SELECT * FROM FACTURA,TERCERO where(FACTURA.idTercero = TERCERO.id AND FACTURA.id=idFactura)`

Se usa para obtener una determinada factura para un tercero en específico.

`SELECT MAX(ID) FROM TERCERO`

Se usa para obtener la última tupla insertada en la tabla de terceros.

### 9.3.2. Modificado

Tan solo se ha usado una modificación en esta funcionalidad para cambiar el stock de un determinado material en la base de datos:

```
UPDATE MATERIAL SET STOCK=ntotal WHERE ID=idMaterial
```

### 9.3.3. Inserción

Las funciones de inserción se han realizado en PL/SQL, una para cada tipo de objeto que se fuera a usar en este caso el único que varía un poco es la inserción de materiales:

```
create or replace FUNCTION INSERTAFACTURA(v_descripcion in VARCHAR, v_proveedor in VARCHAR,
v_tipo in VARCHAR, v_cantidad in FLOAT, v_idtercero in NUMBER)
RETURN NUMBER
AS
    v_result number := 0;
BEGIN
    INSERT INTO FACTURA (descripcion,nombreproveedor,tipo,cantidad,fechalimite,IDTERCERO)
    VALUES (v_descripcion,v_proveedor,v_tipo,v_cantidad,TO_DATE(SYSDATE), v_idtercero);
    v_result := 1;
RETURN v_result;
END;
```

```
create or replace FUNCTION INSERTATERCERO
(v_nombreComercial in VARCHAR, v_nRegistro in VARCHAR, v_iban in VARCHAR, v_telefono in NUMBER,
v_fax in NUMBER, v_descripcion in VARCHAR, v_direccion in VARCHAR)
RETURN NUMBER
AS
    v_result number := 0;
BEGIN
    INSERT INTO TERCERO (NOMBRECOMERCIAL,NREGISTRO,IBAN,TELEFONO,FAX,DESCRIPCION,DIRECCION)
    VALUES (v_nombreComercial,v_nRegistro,v_iban,v_telefono,v_fax,v_descripcion,v_direccion);
    v_result := 1;
RETURN v_result;
END;


create or replace FUNCTION INSERTAMATERIAL
(v_NOMBRE in VARCHAR, v_TIPO in VARCHAR,
v_DESCRIPCION in VARCHAR, v_STOCK in NUMBER, v_IDTERCERO in NUMBER, v_IDSALA in NUMBER)
RETURN NUMBER
AS
    v_result MATERIAL.id%type;
BEGIN
    INSERT INTO MATERIAL (NOMBRE,TIPO,DESCRIPCION,STOCK)
    VALUES (v_NOMBRE,v_TIPO,v_DESCRIPCION,v_STOCK);
    SELECT MAX(ID) INTO v_result FROM MATERIAL;
    INSERT INTO TSUMINISTRAM (IDTERCERO,IDMATERIAL)
    VALUES (v_IDTERCERO,v_result);
RETURN v_result;
END;


create or replace FUNCTION INSERTAPAGO(v_IBAN in VARCHAR,
v_concepto in VARCHAR, v_destinatario in VARCHAR, v_cantidad in FLOAT, v_idFactura in NUMBER)
RETURN NUMBER
AS
    v_result number := 0;
BEGIN
    INSERT INTO PAGO (IBAN,cantidad,destinatario,concepto,fechapago,IDFACTURA)
    VALUES (v_IBAN,v_cantidad,v_destinatario,v_concepto,TO_DATE(SYSDATE),v_idFactura);
    v_result := 1;
RETURN v_result;
END;
```

### 9.3.4. Borrado

```
create or replace FUNCTION BORRAMATERIAL (
  V_ID IN NUMBER
) RETURN NUMBER AS
v_result number := 0;
BEGIN
  DELETE FROM MATERIAL WHERE ID = V_ID;
  v_result := 1;
  RETURN v_result;
END BORRAMATERIAL;


create or replace FUNCTION BORRATERCERO (
  V_ID IN NUMBER
) RETURN NUMBER AS
v_result number := 0;
BEGIN
  DELETE FROM TERCERO WHERE ID = V_ID;
  v_result := 1;
  RETURN v_result;
END BORRATERCERO;
```

## 9.4. Logística y gestión de salas

### 9.4.1. Inserción

Inserción de ambulancias en PL/SQL:

```
create or replace FUNCTION INSERTARAMBULANCIA
(
  V_MATRICULA IN AMBULANCIAS.MATRICULA%TYPE
, V_MODELO IN AMBULANCIAS.MODELO%TYPE
, V_MARCA IN AMBULANCIAS.MARCA%TYPE
, V_POTENCIA IN AMBULANCIAS.POTENCIACV%TYPE
, V_ULTIMAREVISION IN AMBULANCIAS.ULTIMAREVISION%TYPE
, V_FECHAFABRICACION IN AMBULANCIAS.FECHAFABRICACION%TYPE
, V_DISPONIBLE IN AMBULANCIAS.DISPONIBLE%TYPE
, V_RUTAACTUAL IN AMBULANCIAS.RUTA%TYPE
) RETURN NUMBER AS
    v_result number := 0;
BEGIN
  INSERT INTO AMBULANCIAS(Matricula, Modelo, Marca, PotenciaCV, UltimaRevision,
                          FechaFabricacion, Disponible, Ruta)
  VALUES(V_MATRICULA, V_MODELO, V_MARCA, V_POTENCIA, V_ULTIMAREVISION, V_FECHAFABRICACION, V_DISPO
  v_result := 1;
  RETURN v_result;
END INSERTARAMBULANCIA;
```

### 9.4.2. Consulta

Para las consultas no se ha usado PL/SQL. Se han ejecutado sobre la base de datos las siguientes operaciones:

`SELECT MAX(ID) FROM SALA`

Se usa para encontrar la sala con mayor ID

`SELECT * FROM SALA`

Se usa para recuperar todas las salas y representarlas en una tabla en el programa.

`SELECT * FROM AMBULANCIAS`

Se usa para recuperar todas las ambulancias y representarlas en una tabla en el programa.

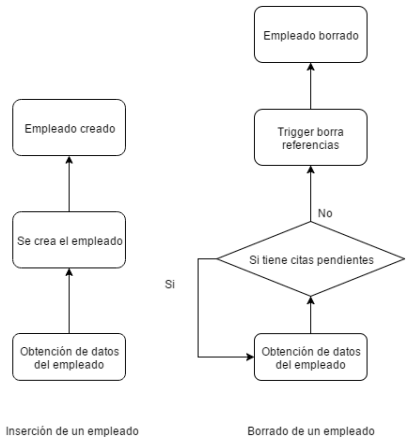
`SELECT IDSALA FROM MSEENCUENTRAS WHERE IDMATERIAL=idmat`

Selecciona para un material dado (idmat) las salas en las que se encuentra.

# 10. Esquemas de operación y navegación

## 10.1. Recursos humanos

Las operaciones más importantes son la inserción y el borrado de los empleados:

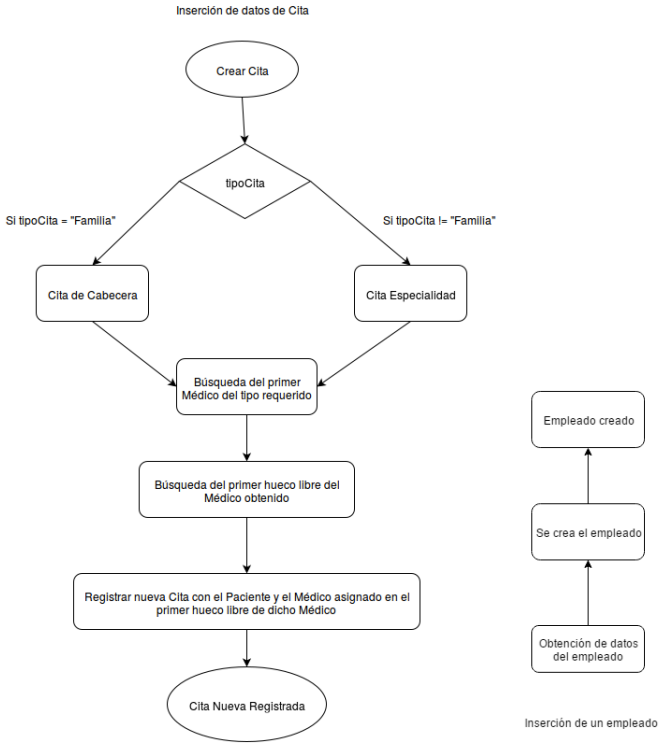



La operación de borrado hace uso de un trigger para eliminar las referencias existentes entre los pacientes y el médico que tienen asignado por defecto. El código es el siguiente:

```
create or replace TRIGGER TR_BORRA_EMPLEADO
BEFORE DELETE ON EMPLEADO
for each row
BEGIN
UPDATE PACIENTE SET MEDICO_ASIGNADO=null WHERE MEDICO_ASIGNADO =:old.ID;
END ;
```

El paciente pasa a no tener médico asignado, por lo que sería deseable disponer de otro procedimiento que le asignara un nuevo médico.

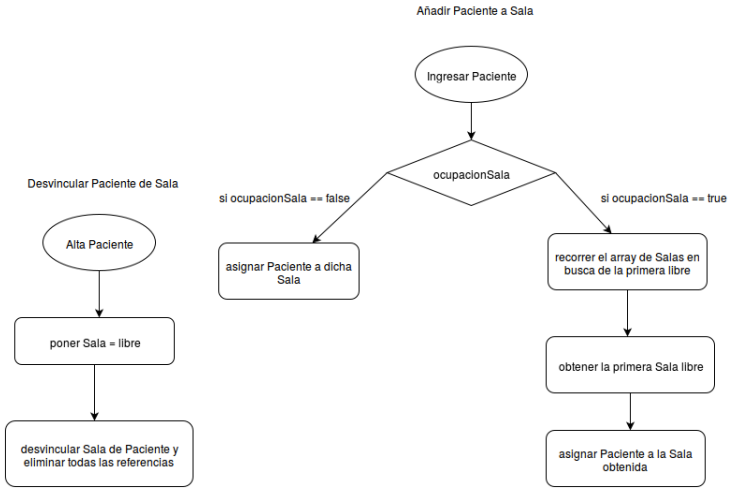
## 10.2. Asistencia




La operación borrarCita no está implementada pero debe hace uso de un trigger para el borrado de referencias no necesarias cuando se proceda a eliminar dicha Cita.

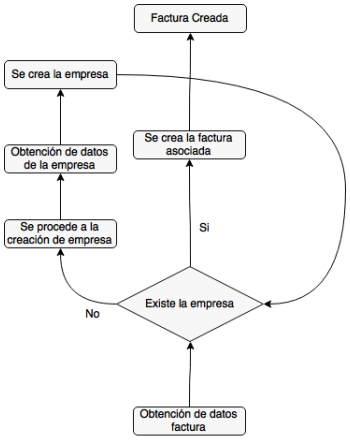
Es decir, dicho trigger se encarga de eliminar las referencias entre las tablas de Paciente, Salas y la de Médicos, eliminando las referencias que haya entre ellos asociadas a la Cita a eliminar.

Esquemas de navegación para dar de alta e ingresar un paciente:

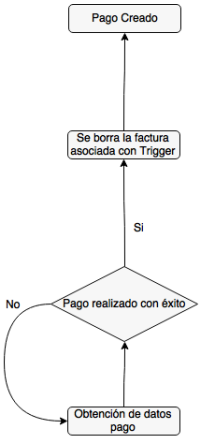


## 10.3. Contabilidad y gestión de material

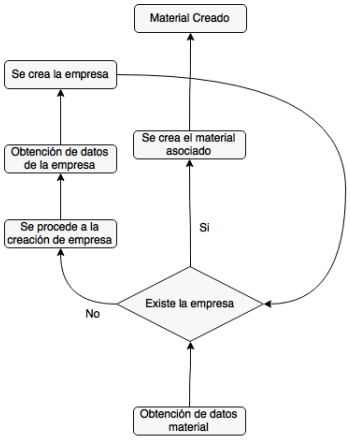
En las operaciones de inserción de esta funcionalidad sin lugar a dudas la más importante o de mayor relevancia es la inserción de un material y una factura:



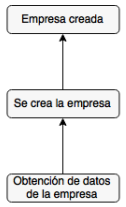
Inserción de una Factura



Inserción de un pago

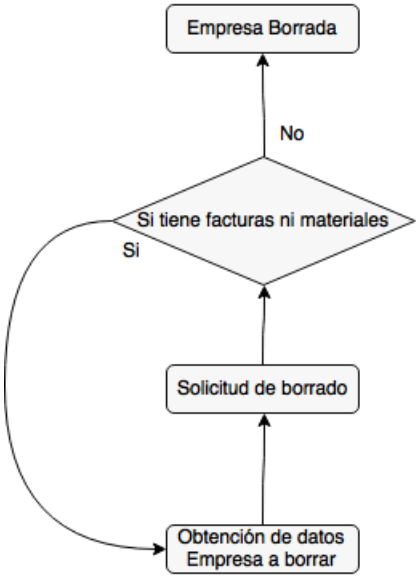


Inserción de un Material

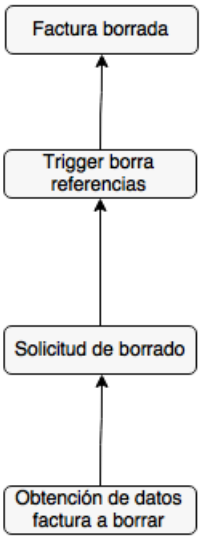


Inserción de un Empresa

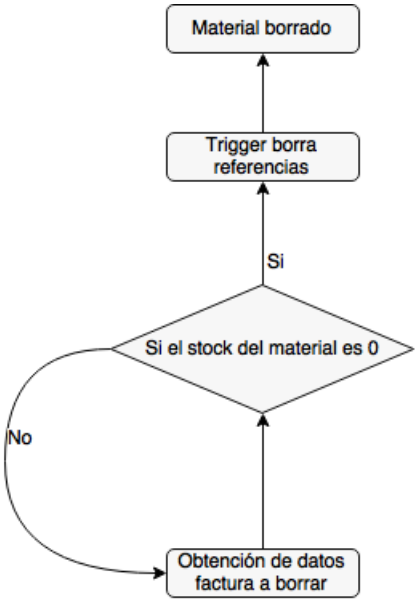
Las operaciones de borrado son algo más simples:



Borrado de una Empresa



Borrado de una Factura



Borrado de una Material

Ambas operaciones hacen uso de triggers para el borrado de referencias no necesarias cuando se procede a eliminar o bién para el borrado de una factura cuando esta se ha pagado, el código de los triggers usado es el siguiente:

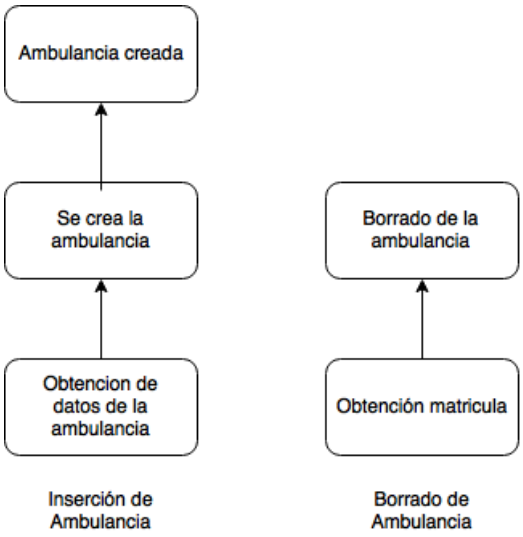
```
create or replace TRIGGER TR_BORRA_MATERIAL
```

```
  BEFORE DELETE ON MATERIAL
  for each row
BEGIN
    DELETE FROM TSUMINISTRAM WHERE IDMATERIAL = :old.ID;
    DELETE FROM MSEENCUENTRAS WHERE IDMATERIAL = :old.ID;
    dbms_output.put_line('Old id: ' || :OLD.ID);
END ;
```

```
create or replace TRIGGER TR_PAGA_FACTURA
  AFTER INSERT ON PAGO
  for each row
BEGIN
    DELETE FROM FACTURA WHERE ID = :new.IDFACTURA;
END ;
```

El trigger que borra material se encarga de eliminar las referencias de las tablas de relación entre la tabla de salas y la de empresas o terceros, y el trigger paga factura se encarga de que cada vez que se ha realizado un pago elimina la factura ya que esta no es necesaria, aunque se ha decidido eliminarla del sistema es mejor añadirla a una tabla auxiliar de facturas pagadas que no aparecerán como pendientes, es tan facil como cambiar este trigger para que además de eliminar la factura correspondiente antes la inserte con sus datos en esa otra tabla.

### 10.4. Logística y gestión de salas




## 11. Diseño Lógico

AMBULANCIAS(Matricula, Modelo, Marca, Potenciacv, Ultimarevision, FechaFabricacion, Disponible, Ruta)

CITA(Id,Hora, Fecha)

EMPLEADO(Id, Nombre, Apellidos, DNI, Direccion, Tipo, Telefono, Horas, Nomina, Turno, Cita1, Cita2, Cita3, Cita4, Especialidad)

FACTURA(Id, Descripción, Tipo, Cantidad, Nombre Proveedor, Fecha Límite, IdTercero)

MATERIAL(Id, Nombre, Descripción, Tipo, Stock)

MSEENCUENTRAM(IdMaterial, IdSala)

PACIENTE(Id, Nombre, Apellidos, Fecha nacimiento, DNI, Tarjeta Sanitaria, Medico asignado, Sala, cita asignada)

PAGO(Id, IBAN, Cantidad, Destinatario, Concepto, FechaPago, IdFactura)

RUTAS(Id, Fecha, Tipo, Distancia, Dirección)

SALA(Id, Tipo, Ocupación)

TERCERO(Id, Número Registro, Nombre Comercial, Descripción, Teléfono, Fax, IBAN)

TSUMINISTRAM(IdMaterial,IdTercero)

TURNO(Nombre, Inicio, Fin)

# 12. Diseño Físico

Las instrucciones DDL que se han usado para crear la base de datos son las siguientes:

```
--------------------------------------------------------
--  DDL for Table FACTURA
--------------------------------------------------------

  CREATE TABLE "X4742492"."FACTURA"
   ( "ID" NUMBER GENERATED BY DEFAULT AS IDENTITY NOT NULL ,
"DESCRIPCION" VARCHAR2(255 BYTE),
"NOMBREPROVEEDOR" VARCHAR2(50 BYTE),
"TIPO" VARCHAR2(20 BYTE),
"CANTIDAD" FLOAT(126),
"FECHALIMITE" DATE,
"IDTERCERO" NUMBER REFERENCES "X4742492"."TERCERO" ("ID")
   ) PRIMARY KEY (ID) ;
--------------------------------------------------------
--  DDL for Table MATERIAL
--------------------------------------------------------

  CREATE TABLE "X4742492"."MATERIAL"
   ( "ID" NUMBER GENERATED BY DEFAULT AS IDENTITY NOT NULL ,
"NOMBRE" VARCHAR2(50 BYTE),
"DESCRIPCION" VARCHAR2(255 BYTE),
"TIPO" VARCHAR2(50 BYTE),
"STOCK" NUMBER(*,0) DEFAULT 0
   ) PRIMARY KEY (ID) ;
--------------------------------------------------------
--  DDL for Table MSEENCUENTRAS
--------------------------------------------------------

  CREATE TABLE "X4742492"."MSEENCUENTRAS"
   ( "CANTIDAD" NUMBER DEFAULT 0,
"IDMATERIAL" NUMBER REFERENCES "X4742492"."MATERIAL" ("ID"),
```

```
"IDSALA" NUMBER REFERENCES "X4742492"."SALA" ("ID")
   ) PRIMARY KEY (IDMATERIAL) ;
--------------------------------------------------------
--  DDL for Table PAGO
--------------------------------------------------------

  CREATE TABLE "X4742492"."PAGO"
   ( "ID" NUMBER GENERATED BY DEFAULT AS IDENTITY NOT NULL ,
"IBAN" VARCHAR2(34 BYTE),
"CANTIDAD" FLOAT(126),
"DESTINATARIO" VARCHAR2(50 BYTE),
"CONCEPTO" VARCHAR2(255 BYTE),
"FECHAPAGO" DATE,
"IDFACTURA" NUMBER
   ) PRIMARY KEY (ID) ;
--------------------------------------------------------
--  DDL for Table TERCERO
--------------------------------------------------------

  CREATE TABLE "X4742492"."TERCERO"
   ( "ID" NUMBER GENERATED BY DEFAULT AS IDENTITY NOT NULL ,
"NREGISTRO" NUMBER,
"NOMBRECOMERCIAL" VARCHAR2(50 BYTE),
"DESCRIPCION" VARCHAR2(255 BYTE),
"DIRECCION" VARCHAR2(100 BYTE),
"TELEFONO" NUMBER,
"FAX" NUMBER,
"IBAN" VARCHAR2(34 BYTE)
   ) PRIMARY KEY (ID) ;
--------------------------------------------------------
--  DDL for Table TSUMINISTRAM
--------------------------------------------------------

  CREATE TABLE "X4742492"."TSUMINISTRAM"
   ( "IDMATERIAL" NUMBER REFERENCES "X4742492"."MATERIAL" ("ID"),
"IDTERCERO" NUMBER REFERENCES "X4742492"."TERCERO" ("ID"),
   ) PRIMARY KEY (IDMATERIAL) ;
--------------------------------------------------------
--  DDL for Table SALA
--------------------------------------------------------

  CREATE TABLE "X4742492"."SALA"
   ( "ID" NUMBER GENERATED BY DEFAULT AS IDENTITY NOT NULL ,
"TIPO" VARCHAR2(255 BYTE),
"OCUPACION" NUMBER(1,0)
   ) PRIMARY KEY(ID) ;
--------------------------------------------------------
--  DDL for Table EMPLEADO
--------------------------------------------------------

CREATE TABLE "X4742492"."EMPLEADO" (
"ID" NUMBER DEFAULT "X4742492"."EMPLEADO_SEQ".nextval,
"NOMBRE" VARCHAR2(20 BYTE),
"APELLIDOS" VARCHAR2(32 BYTE),
"DNI" VARCHAR2(20 BYTE),
"DIRECCION" VARCHAR2(40 BYTE),
"TIPO" VARCHAR2(20 BYTE),
"TELEFONO" NUMBER,
"HORAS" NUMBER,
"NOMINA" NUMBER,
"TURNO" VARCHAR2(20 BYTE),
"CITA1" NUMBER(3,0) DEFAULT 1,
"CITA2" NUMBER(3,0) DEFAULT 1,
"CITA3" NUMBER(3,0) DEFAULT 1,
"CITA4" NUMBER(3,0) DEFAULT 1,
"ESPECIALIDAD" VARCHAR2(20 BYTE),
CONSTRAINT "EMPLEADO_PK" PRIMARY KEY ("ID"),
CONSTRAINT "EMPLEADO_FK1" FOREIGN KEY ("TURNO") REFERENCES "X4742492"."TURNO" ("NOMBRE")
);
--------------------------------------------------------
--  DDL for Table TURNO
--------------------------------------------------------
CREATE TABLE "X4742492"."TURNO" (
"NOMBRE" VARCHAR2(20 BYTE),
```

```
"INICIO" VARCHAR2(20 BYTE),
"FIN" VARCHAR2(20 BYTE)
);
--------------------------------------------------------
--  DDL for Table MEDICO
--------------------------------------------------------
CREATE TABLE Medico(
id NUMBER GENERATED BY DEFAULT AS IDENTITY NOT NULL,
dni CHAR(9),
nombre VARCHAR(100),
apellido1 VARCHAR(100),
apellido2 VARCHAR(100),
especialidad VARCHAR(50) REFERENCES Especialidad(tipo),
PRIMARY KEY (id)
);
--------------------------------------------------------
--  DDL for Table CITA
--------------------------------------------------------
CREATE TABLE Cita(
id NUMBER GENERATED BY DEFAULT AS IDENTITY NOT NULL,
hora CHAR(5),
fecha date,
PRIMARY KEY (id)
);
--------------------------------------------------------
--  DDL for Table PACIENTE
--------------------------------------------------------
CREATE TABLE Paciente(
id NUMBER GENERATED BY DEFAULT AS IDENTITY NOT NULL,
nombre VARCHAR(100),
apellido1 VARCHAR(100),
apellido2 VARCHAR(100),
fecha_nacimiento date,
dni CHAR(9),
tarjeta_sanitaria CHAR(12),
medico_asignado NUMBER REFERENCES Médico(id),
sala NUMBER REFERENCES Sala(id),
cita_asignada NUMBER REFERENCES Cita(id),
PRIMARY KEY (id)
);
--------------------------------------------------------
--  DDL for Table AMBULANCIAS
--------------------------------------------------------

  CREATE TABLE "X4742492"."AMBULANCIAS"
   ( "MATRICULA" VARCHAR2(20 BYTE),
"MODELO" VARCHAR2(20 BYTE),
"MARCA" VARCHAR2(20 BYTE),
"POTENCIACV" NUMBER(*,0),
"ULTIMAREVISION" DATE,
"FECHAFABRICACION" DATE,
"DISPONIBLE" NUMBER(*,0) DEFAULT 0,
"RUTA" NUMBER
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 NOCOMPRESS LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "BDCCIA" ;

   COMMENT ON COLUMN "X4742492"."AMBULANCIAS"."DISPONIBLE" IS '0=No, 1=Si';
--------------------------------------------------------
--  DDL for Index AMBULANCIAS_PK
--------------------------------------------------------

  CREATE UNIQUE INDEX "X4742492"."AMBULANCIAS_PK" ON "X4742492"."AMBULANCIAS" ("MATRICULA")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "BDCCIA" ;
--------------------------------------------------------
--  Constraints for Table AMBULANCIAS
--------------------------------------------------------
```

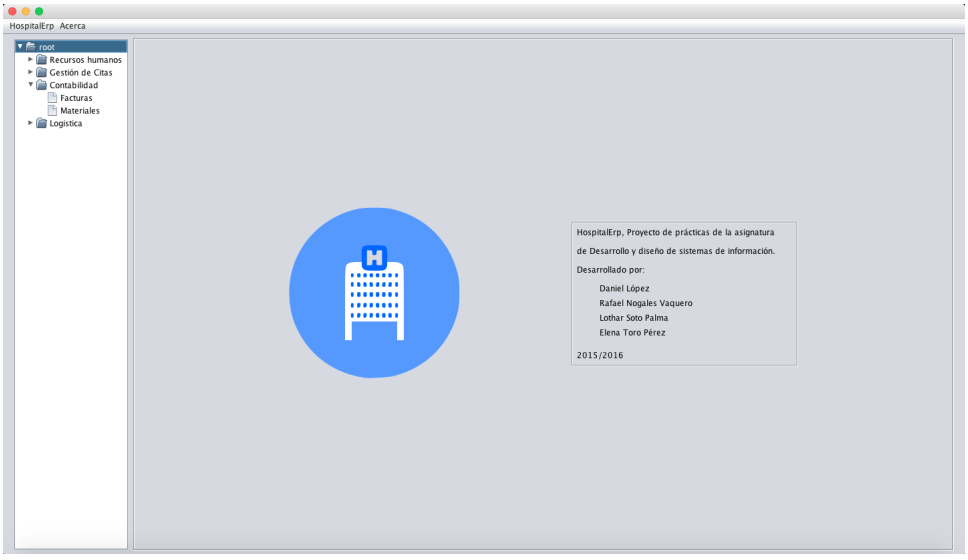
```
  ALTER TABLE "X4742492"."AMBULANCIAS" MODIFY ("MATRICULA" NOT NULL ENABLE);
  ALTER TABLE "X4742492"."AMBULANCIAS" MODIFY ("DISPONIBLE" NOT NULL ENABLE);
  ALTER TABLE "X4742492"."AMBULANCIAS" ADD CONSTRAINT "AMBULANCIAS_PK" PRIMARY KEY ("MATRICULA")
  USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "BDCCIA"  ENABLE;
--------------------------------------------------------
--  Ref Constraints for Table AMBULANCIAS
--------------------------------------------------------

  ALTER TABLE "X4742492"."AMBULANCIAS" ADD CONSTRAINT "AMBULANCIAS_FK1" FOREIGN KEY ("RUTA")
  REFERENCES "X4742492"."RUTAS" ("ID") ENABLE;
```

# 13. Descripción de la solución implementada

Se decidio llevar a cabo la implementación en Java junto a una base de datos Oracle, usamos las tecnologías Netbeans y SqlDeveloper, y para las vistas hemos usado las librerías de Swing. Cada integrante del grupo ha desarrollado su propia implementación basandose en la estructura Módelo-Vista-Controlador de manera que se crean clases que controlan objetos creados en el modelo como pueden ser facturas, materiales, empleados entre otros, y las vistan recogen la información obtenida por estos controladores y lo reflejan al usuario.

En primer lugar había muchas ideas planteadas muchas de ellas se han llevado a cabo pero al final no se ha abarcado tanto como se pretendía por temas de tiempo. Por otro lado nos hemos percatado de algunos fallos a la hora de implementar y desarrollar el sistema como puede ser el uso de funciones PL/SQL en lugar de procedimientos debido a que al principio se pretendía devolver una variable que informara del estado de la operación en PL/SQL, pero nos dimos cuenta de que eso se hacia automáticamente pero eliminar en ese momento las funciones parecía engorroso.



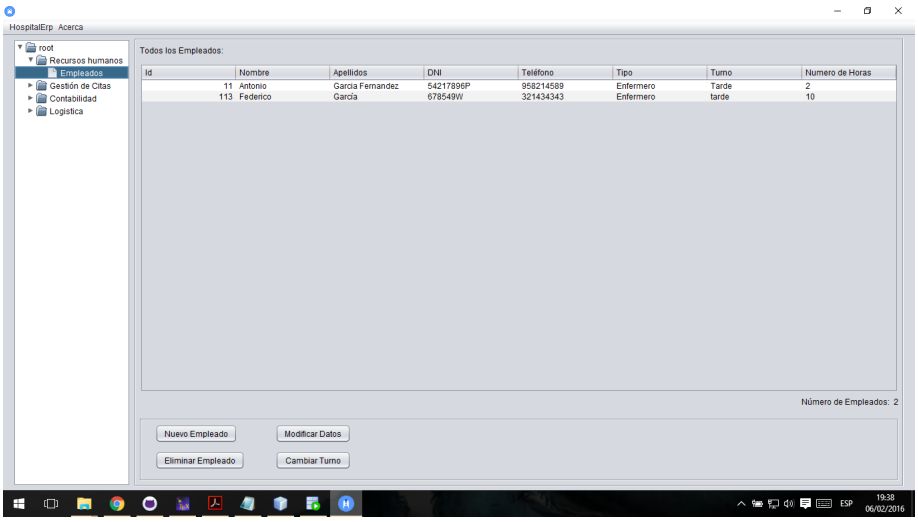

En lo que a vistas respecta el programa se divide en dos partes un panel a la izquierda que indica las funcionalidades que hay y pueden abrirse para añadir subfuncionalidades dependientes de la anterior, de esta manera se favorece a la escalabilidad del programa en cuanto a desarrollo se refiere, esto quiere decir que se pueden desarrollar módulos de forma independiente y añadir

vistas y funcionalidades sin ningún tipo de dependencia de las otras partes (cada una de las funcionalidades propuestas funciona con independencia de las demás), y a la derecha se encuentra la vista principal que se está usando, esta vista irá cambiando en función de la funcionalidad seleccionada.

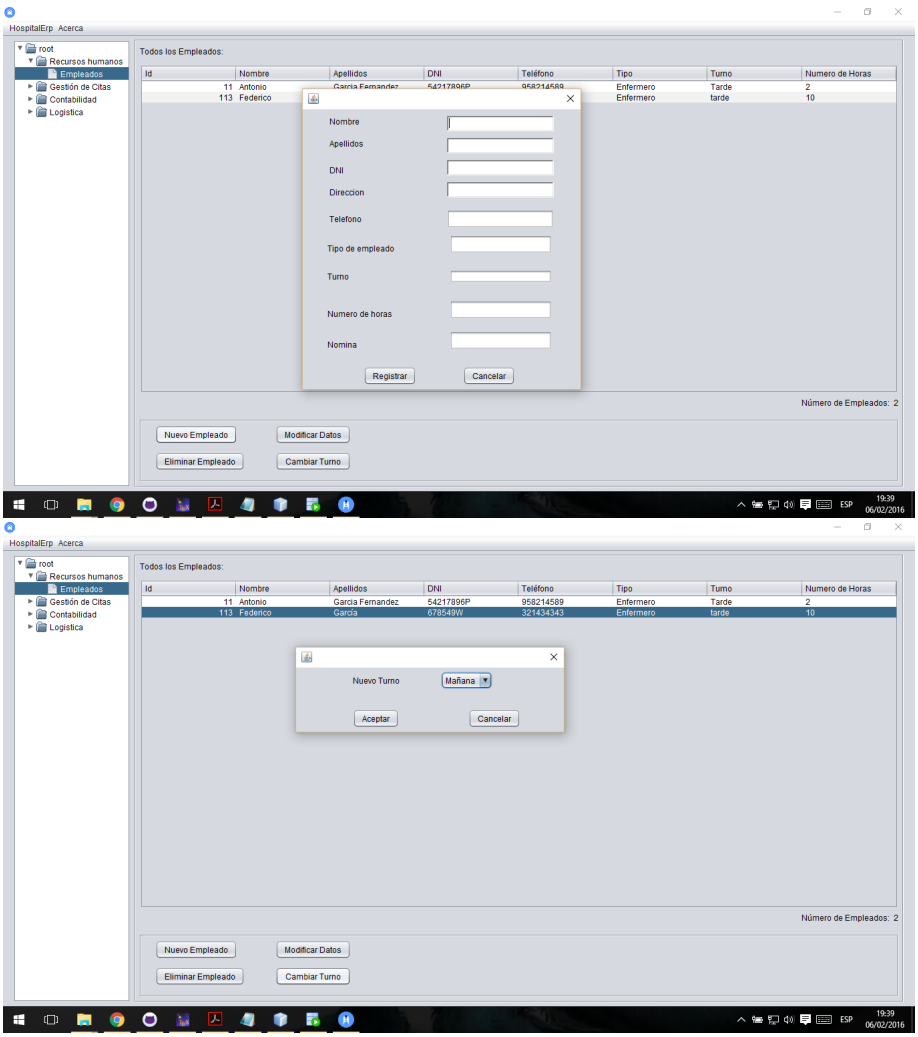
Se procede a explicar el desarrollo de cada una de las partes del sistema:

## 13.1. Recursos humanos

Vista que muestra la lista de empleados registrados en la base de datos. Se complementa con 4 botones con las siguientes funciones: registrar un empleado, borrar un empleado existente, cambiar el turno y modificar los datos. El funcionamiento de dichos botones es el siguiente :



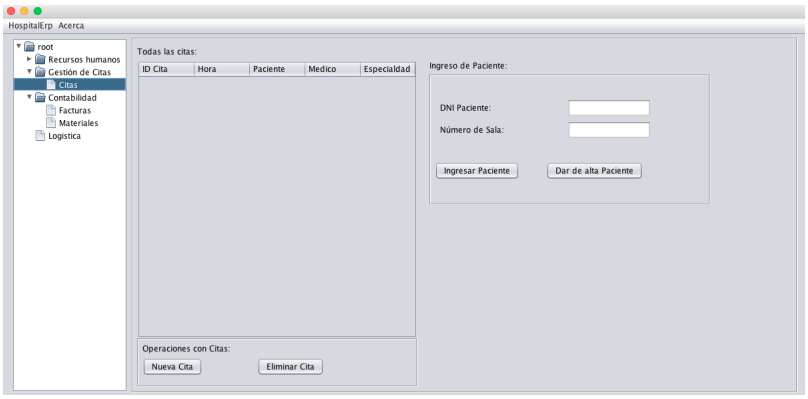

1. Nuevo Empleado: Se despliega un formulario donde se deben introducir los datos referentes al nuevo empleado.
2. Eliminar Empleado: Se borra de la base de datos el empleado seleccionado de la lista.
3. Cambiar Turno: Para el empleado seleccionado en la lista, se despliega un formulario donde podemos elegir el nuevo turno asignado.
4. Modificar Datos: Para el empleado seleccionado en la lista, se despliega un formulario donde introducimos los nuevos datos(se deben introducir todos aunque no hayan sido modificados).



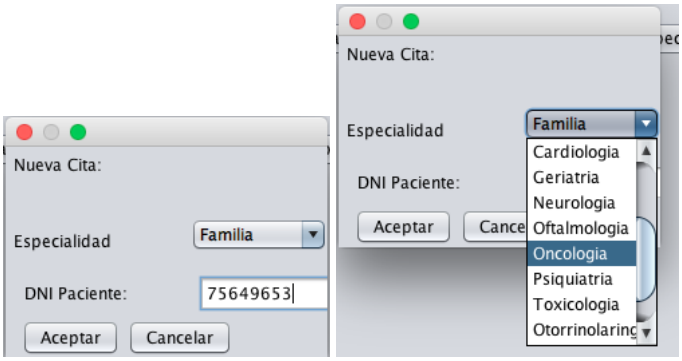

Se han omitido las funciones buscar por un tipo de empleado, ya que carecía de sentido al tener una vista en la base de datos que contiene a los médicos.

## 13.2. Asistencia

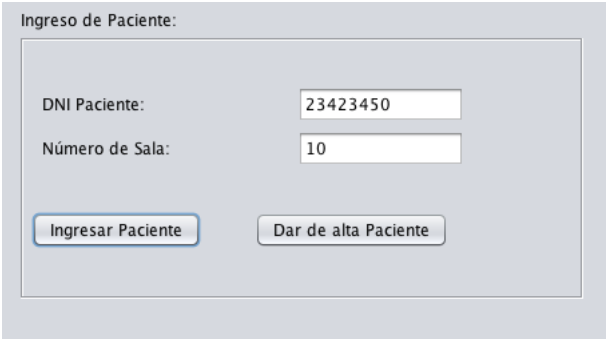
La sección de asistencia se utiliza para gestionar las citas y para ingresar o dar de alta a los pacientes.




Detalles de Nueva cita:




Detalles de ingresar pacientes:

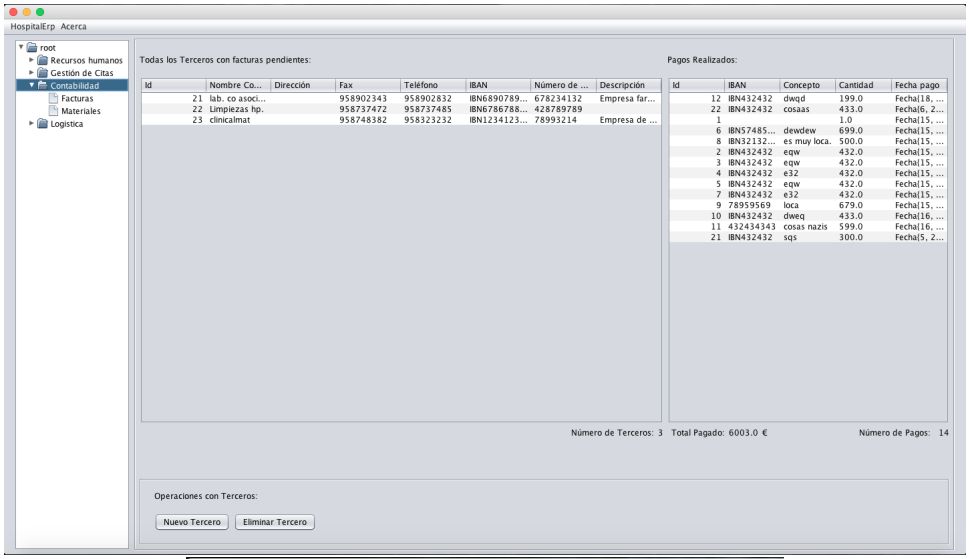


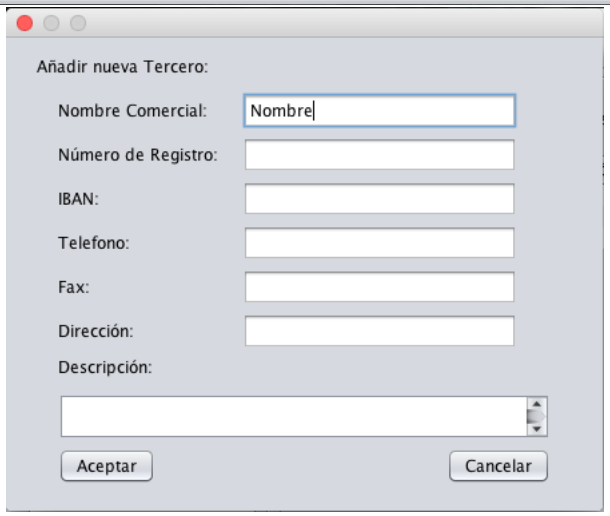

## 13.3. Contabilidad y gestión de material

Se han creado para esta funcionalidad dos módulos independientes: Contabilidad.java y Gestion-Material.java con sus respectivas vistas asociadas, en el caso de contabilidad alguna más porque se decidió separar la representación de los terceros y las facturas y relacionar los pagos y los terceros en una única vista.

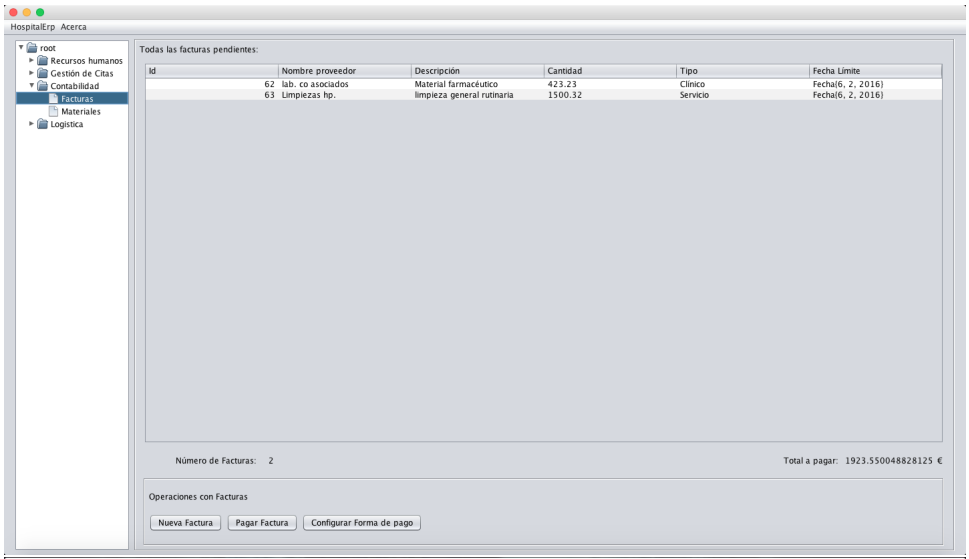
Si se selecciona a la izquierda Contabilidad se subdivide en otras dos material y facturas, y aparece en la vista principal dos tablas que permiten visualizar los pagos que se han realizado y

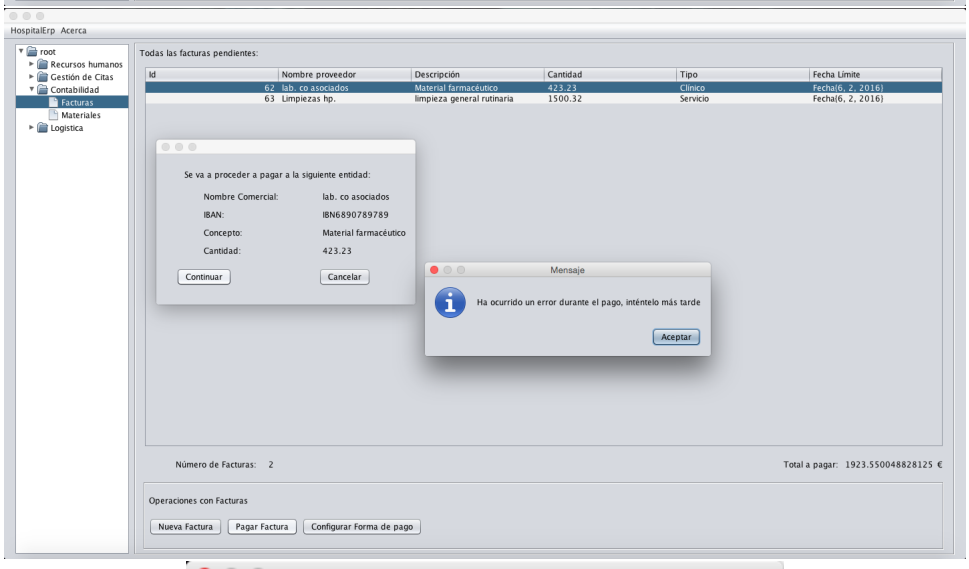
los terceros registrados en tiempo real, además aparecen dos botones para agregar y eliminar terceros, puesto que los pagos se generan al pagar una factura se podría implementar también un boton para ir eliminando aquellos que no interese mantener.




En lo que a facturas respecta se ha procedido de la misma forma se crea una vista con una tabla en la que se muestran las facturas activas y dos botones uno de ellos sirve para agregar nuevas facturas pero tiene una particularidad y es que si no existe el tercero del cual es la factura se obliga a ingresar el tercero, si este no se agrega la factura tampoco lo hace, el boton de pagar factura es similar al de borrar pero en vez de unicamente borrar se simula si la factura se ha pagado con exito o no, se ha puesto un porcentaje de exito y fallo del 50 % de forma que se pagara solo algunas veces, si se paga con exito la factura se elimina del sistema y se crea un pago asociado, por último el botón configuración de pago simplemente sirve para dar la información de la entidad que permite los pagos.

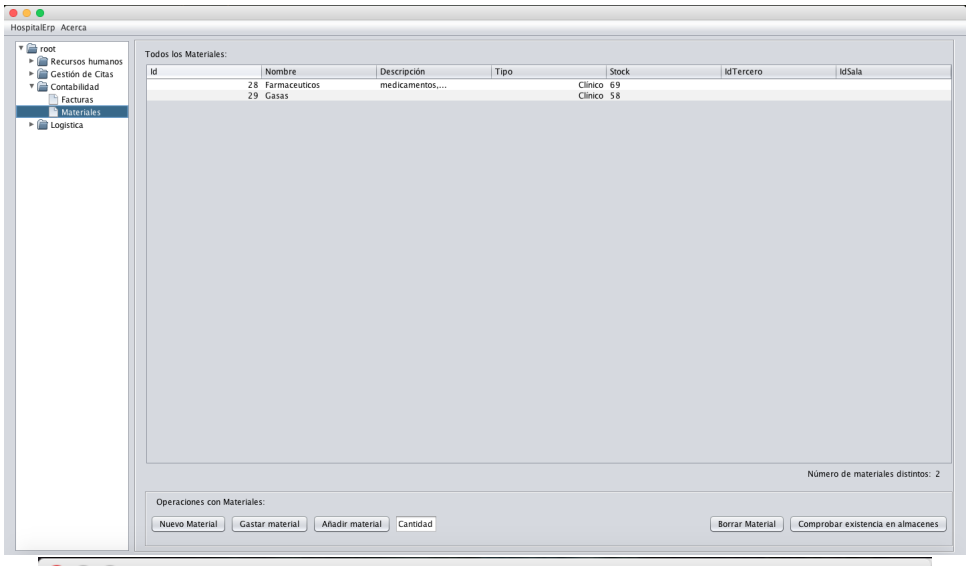


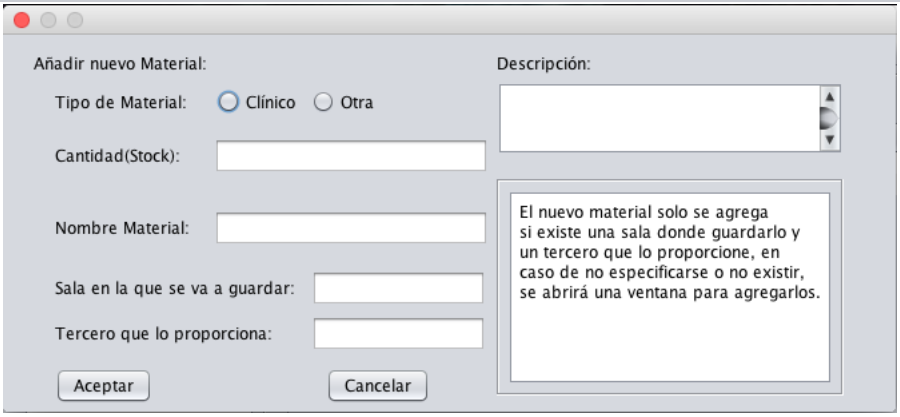





En esta última parte se tratan los materiales de forma similar a las facturas sin embargo estas

pueden crearse con un stock o cantidad, pero solo pueden eliminarse de la base de datos cuando no hay materiales de ese tipo, tambien hay un botón que permite agregar stock a un determinado material.



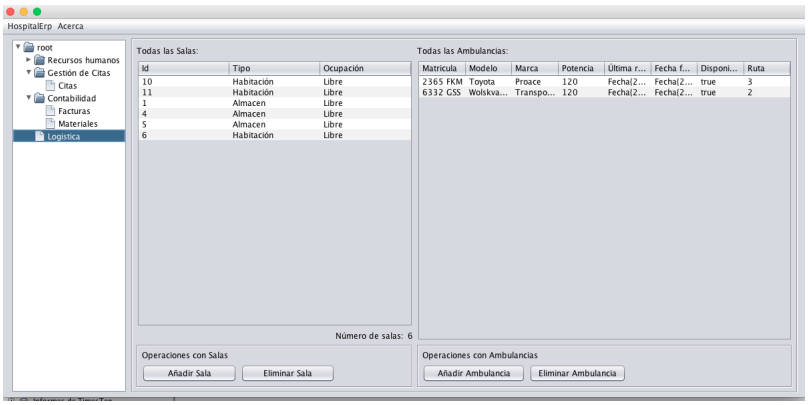




Para acabar el botón comprobar existencia en almacenes se encarga de a partir de un identificador de sala que sea un almacén obtener una lista de todos los materiales que se pueden encontrar en dicho almacén.

No se ha implementado consultas a nivel de usuario por falta de tiempo pero estoy seguro de pode implementarlo para antes de la defensa, tan solo se añadiría a las operaciones de consulta una del mismo estilo que las demás pero parametrizada en la tabla en la que se busca y el 'where' dependiendo que se quiera obtener.

## 13.4. Logística y gestión de salas

La gestion de ambulancias y salas corresponde al apartado de Logística. Las ambulancias son un módulo extra en el ERP, las salas se utilizan para almacenar materiales, como habitaciones para los pacientes y para cualquier otro uso que se le quiera dar.




Utilizando los botones agregar sala y agregar ambulancia se crean nuevas salas y ambulancias en la base de datos y pueden utilizarse en el resto de módulos del ERP