

# **Universidad Nacional de Mar del Plata**

**Carrera:** Ingeniería informática

**Materia:** Programación C

## **Trabajo Práctico Grupal - Primera etapa**

### **Profesores:**

- Ivonne Gellon
- Leonel Domingo Guccione
- Carlos Guillermo Lazurri

### **Integrantes:**

- Gribman, Ianai
- Gregorini, Dante
- Vazquez Baccaro, Mateo
- Acevedo, Valentina
- Milosevic, Natasha

## Índice

<b>1. Introducción</b>	3
<b>2. Funcionamiento general de los objetos utilizados</b>	4
<b>3. Descripción del diseño</b>	6
3.1. Módulo de registro	
3.2. Módulo de ingreso	
3.3. Módulo de resolución de conflictos de la sala de espera	
3.4. Módulo de atención	
3.5. Módulo de egreso y facturación	
3.6. Módulo de reporte de actividad de médicos	
<b>4. Diagrama de clases</b>	11

## 1. Introducción

Se presenta un sistema de gestión de una clínica privada, mediante el desarrollo de los siguientes módulos:

- Registro
- Ingreso
- Conflictos de la Sala de Espera
- Atención
- Egreso
- Facturaciones
- Reporte de actividad de médicos

A continuación, se describirán las formas de trabajo (resoluciones) de cada uno de los módulos mencionados, así como el diagrama de clases que acompaña la resolución.

## 2. Funcionamiento general de los objetos utilizados

### ➤ *Entidad*

Establece las características en común de la clínica y personas en ella (médicos y pacientes). Cada entidad tiene un nombre, teléfono y domicilio, teniendo una relación de composición con este último.

### ➤ *Clinica*

Guarda la información de toda la clínica mediante módulos de Registro, Espera, Reportes Médicos y Atención. A partir de estos módulos, la clínica posee hashmaps que identifican a cada paciente y médico con su respectivo DNI e información adicional de acuerdo a su rol, posee la lista de pacientes en espera y de aquellos que están siendo atendidos. Es, entonces, la responsable de registrar, ingresar, atender, internar y egresar pacientes, realizando las delegaciones a los distintos módulos según corresponda.

### ➤ *Persona*

Clase hija de Entidad, establece la característica en común de los pacientes y los médicos.

### ➤ *Pacientes - tipos*

Por cada paciente se almacena su correspondiente número de historia clínica, e información personal.

Posee métodos abstractos posteriormente implementados por sus clases hijas (Ninio, Joven, Mayor), a partir de los cuales se establece el orden de prioridad requerido para el manejo de la Sala de Espera.

### ➤ *Medicos - categorías*

Cada médico posee un número de matrícula y un honorario básico igual para todos los médicos, además de los atributos heredados de Persona.

Cada categoría, clases hijas de Médico, implementan su honorario concreto.

### ➤ *Médicos - decorators*

Implementan capas de diseño de acuerdo al respectivo Posgrado y Contrato de cada médico, realizando el cambio sobre los honorarios del mismo, según el tipo de cada uno.

### ➤ *ModuloRegistro*

Módulo de Registro manejado por la Clínica. Agrega al hashmap correspondiente aquellos Pacientes o Médicos indicados, siempre que no sean repetidos. También

cuenta con métodos booleanos para dar a conocer el estado de registro de los mismos (registrado / no registrado).

➤ *Sala de Espera Privada / Patio*

Poseen lista de los pacientes que se encuentran en esa sección (patio o sala, según corresponda). Cuentan con métodos para indicar si un paciente específico está en la sección, o ponerlo/sacarlo de la misma según se requiera.

➤ *ModuloEspera*

Módulo de Espera manejado por la Clínica. Cuenta con un atributo para el manejo del Patio y otro para el manejo de la Sala Privada. A partir de este módulo, la Clínica es capaz de conocer si un paciente en específico está en espera, y gestionar el ingreso o egreso del mismo a la zona que corresponda.

➤ *MedicoHonorario*

Clase que contiene una referencia a un médico y el honorario que le corresponde al momento de crear el objeto, según su categoría, Posgrado y Contrato.

➤ *Atención*

Contiene toda la información de una atención: paciente al que se refiere, fecha de ingreso y egreso a la Clinica, cantidad de días que permaneció en la misma, habitación asignada, y listado de médicos que lo atendieron durante una estadía. Esta clase es la responsable de agendar las consultas que recibe el paciente en cuestión, así como informar sobre las mismas (por ejemplo, si fue internado, o si fue atendido por algún médico en particular, entre otros).

➤ *ModuloAtenciones*

Módulo de Atenciones manejado por la Clínica. Consiste en una lista de todas las atenciones que realizó la misma, ordenadas por fecha de egreso. Cada elemento de la lista es un objeto Atención, y mediante este módulo es posible llevar un registro histórico de todas las consultas e internaciones realizadas a los pacientes.

Possee métodos para egresar a un paciente, registrando como histórica la última información agendada de él. También, permite obtener las atenciones que se registraron en un periodo de tiempo indicado por el usuario y obtener los pacientes que atendió un médico en específico así, a partir de ello, generar su reporte.

➤ *Habitaciones*

La clase Habitación guarda la información concreta de la misma; su costo de asignación, capacidad y ocupación. Se permite registrar si cada habitación ha sido sido completamente ocupada o aun hay espacio en la misma. Sus clases hijas implementan el costo total particular de ese tipo. La creación de la habitación utiliza el patrón Factory.

➤ *ConsultaMedicaFactura*

Obtiene la información del valor de la consulta de un médico en particular.

➤ *Factura*

De acuerdo a la Atención que recibió, organiza la información correspondiente de la misma para armar una factura. Obtiene la fecha de ingreso y egreso, la cantidad de días, los datos del paciente, si fue internado o no y sus consultas médicas. Agenda en una lista el valor de cada consulta que tuvo el paciente, para posteriormente poder informar de las mismas y obtener el costo total de su estadía en la Clínica.

➤ *ConsultaPacienteReporte*

Obtiene la información de un paciente, con la fecha en la que fue atendido y el honorario que le corresponde.

➤ *ReporteMedico*

Possee listado de consultas para un médico en particular. Genera el Reporte Médico correspondiente, a partir de los honorarios que recibe de cada paciente que atendió. Calcula la ganancia total.

➤ *ModuloGestionCostos*

Permite modificar los costos de cada habitación de la Clínica, así como el honorario básico de los médicos.

➤ *Excepciones*

Se manejan excepciones específicas:

- DNIException → Referido a posibles errores con el DNI (repetición, invalidez)
- DNIRepetidoException → Hija de DNIException. Querer registrar a un paciente que ya se encuentra en el hashmap de la Clínica.
- EgresoSinMedicoException → Al querer egresar a un paciente que no ha sido atendido por ningún médico.
- HabitacionLlenaException → Intento de acceso a una habitación llena.
- HabitacionInvalidaException → Acceso a habitación que no existe.
- MedicoInvalidoException → Crear médico de un tipo no reconocido.
- MedicoNoRegistradoException → Intento de llamar a un médico que no se encuentra registrado en la Clinica.
- PacienteNoIngresadoException → Al querer atender a un paciente que no fue previamente ingresado.
- PacienteNoRegistradoException → Al querer ingresar o atender un paciente que no fue previamente registrado.
- PacienteYaIngresadoException → Intento de ingreso para pacientes que ya se encuentran en la lista de espera.
- PacienteInvalidoException → Al intentar crear un paciente cuyo rango Etario no es reconocido

- PacienteYaInternadoException → Al intentar internar un paciente que ya ha sido asignado a una habitacion (internado).

### **3. Descripción del diseño**

La clase *Clinica* hace de intermediario entre el cliente y los módulos que implementan las funcionalidades de la clínica (*ModuloRegistro*, *ModuloEspera*, *ModuloAtenciones*, *ModuloGestionCostos*). Provee a cada módulo la información que necesita, reduciendo las interacciones y dependencias entre ellos. Así, se hace uso del patrón Facade.

#### ***1. Módulo de registro***

Cuando se quiere registrar un paciente o un médico a la clínica, la clase *Clinica* delega esta tarea a la clase *ModuloRegistro*. Esta clase guarda la referencia al paciente o médico si ya no existe un paciente o médico con el mismo DNI.

#### ***2. Módulo de ingreso***

Al ingresar un paciente a la clínica, *Clinica* verifica que el paciente esté registrado. De ser así, delega la tarea de instalar al paciente en la sala de espera privada o en el patio a la clase *ModuloEspera*. Esta última, antes de alojar a un paciente, comprueba que no se encuentre ya en alguno de los espacios. Si el paciente fue instalado exitosamente, *Clinica* encomienda a la clase *ModuloAtenciones* la tarea de crear una *Atencion*, y agregarla a la lista de atenciones.

#### ***3. Módulo de resolución de conflictos de la sala de espera***

*ModuloEspera* gestiona a partir de dos objetos: *Patio* y *SalaEsperaPrivada*.

Luego de ingresar un paciente, se presentan dos casos:

- Sala de Espera Privada vacía: se agenda al paciente en el atributo “huésped” de la misma, como único ocupador.
- Sala de Espera Privada ocupada: se le delega al huésped de la sala la tarea de decidir si deberá moverse al patio o permanecer donde está, a partir de métodos booleanos donde se comparara con el nuevo paciente.

Las comparaciones se realizan en la clase específica del tipo del paciente (clasificado según rango etario), a partir de la utilización del Double Dispatch. El paciente que ocupa la sala recibe el llamado para la verificación. Este, entonces, llama a la prioridad del paciente que pide ingresar, el cual compara y devuelve el resultado al ocupante. Este último devuelve al método que lo invocó si deberá permanecer o no.

#### ***4. Módulo de atención***

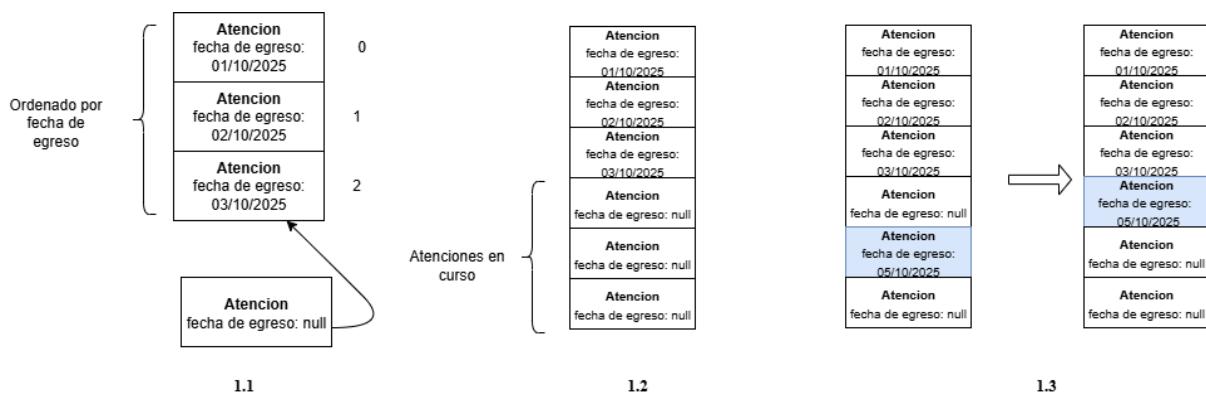
Tanto para la consulta con un médico como para la internación de un paciente, *Clinica* revisa si el paciente se encuentra en espera. Si fuera así, *ModuloEspera* lo quita del patio o sala de

espera privada. Para el caso de la consulta de un paciente con un médico, *Clinica* comprueba que el médico esté registrado. Para la internación, verifica que haya espacio en la habitación donde se hospedará el paciente. Luego, delega la tarea de registrar el respectivo evento a *ModuloAtenciones*.

En las dos situaciones, *ModuloAtenciones* itera secuencialmente comenzando por el final la lista de atenciones. Recorre la lista siempre y cuando la fecha de egreso de la atención sea null. Es decir, aquellas atenciones que siguen en curso. Si no encuentra una atención en curso que tenga la referencia al paciente que se quiere internar o atender, implica que el paciente no fue ingresado y se lanza su correspondiente excepción. Para una internación, verifica que el paciente no haya sido internado previamente. Si no lo fue, registra la habitación en la atención e incrementa la ocupación de la habitación. Un paciente no puede ser internado más de una vez en cada permanencia en la clínica. Para una consulta con un médico, la actualización de la lista de médicos consultados es llevada a cabo por la propia atención. La atención almacena la referencia al médico al igual que su honorario en ese instante de tiempo.

## 5. Modulo de egreso y facturación

Cuando egresa un paciente, *Clinica* invoca el método de egreso en *ModuloAtencion*. Este recorre la lista de la misma manera que en la sección anterior. Si no se encuentra la atención, entonces el paciente nunca ingresó. Si se encuentra la atención pero no fue atendido por ningún médico, no se le permite al paciente egresar y se lanza una excepción. Si fue en efecto atendido, se modifica la fecha de egreso de la atención, se reduce la ocupación de la habitación (si es que el paciente se había internado), se ordena nuevamente la lista de atenciones y se devuelve una referencia al objeto de atención. *Clinica* recibe la atención y construye una factura a partir de ella. Finalmente, *Clinica* devuelve una factura referente al periodo desde que el paciente ingresó por última vez hasta su egreso.

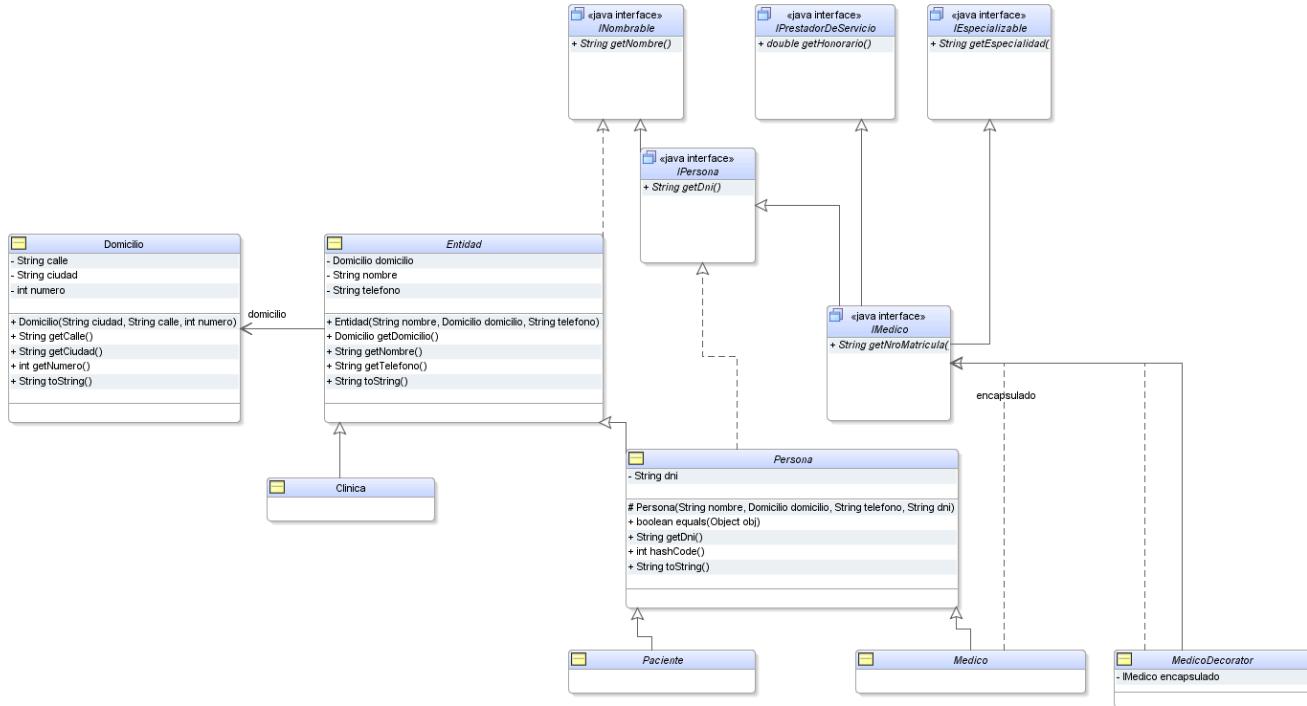


- 1.1** Cuando ingresa un paciente, se crea una nueva atención con fecha de egreso null que se inserta al final de la lista de atenciones. **1.2** Todas las atenciones en curso quedan al final de la lista. **1.3** Cuando egresa un paciente, se modifica su fecha de egreso y se ordena la lista de atenciones, de manera que esté ordenada por fecha de egreso y las atenciones en curso queden al final.

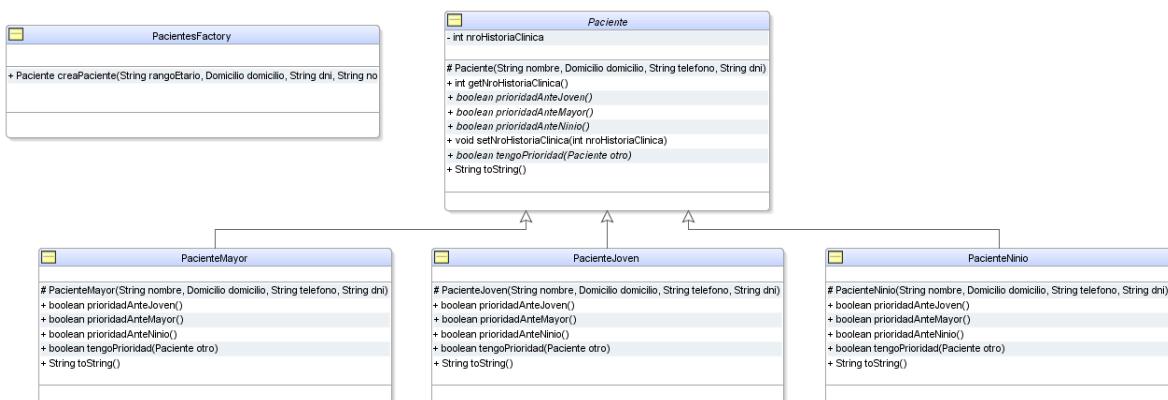
## **6. Módulo de reporte de actividad de médicos**

*ModuloAtenciones* es quien tiene la responsabilidad de generar el reporte de actividad de médicos. Primero crea una sublista de las atenciones cuyo egreso ocurrió en un periodo de tiempo ingresado por el usuario. La creación de la sublista aprovecha el orden de la lista de atención. Luego, se extraen los datos necesarios de la sublista para armar el reporte y se guardan en un objeto de tipo *ReporteMedico*.

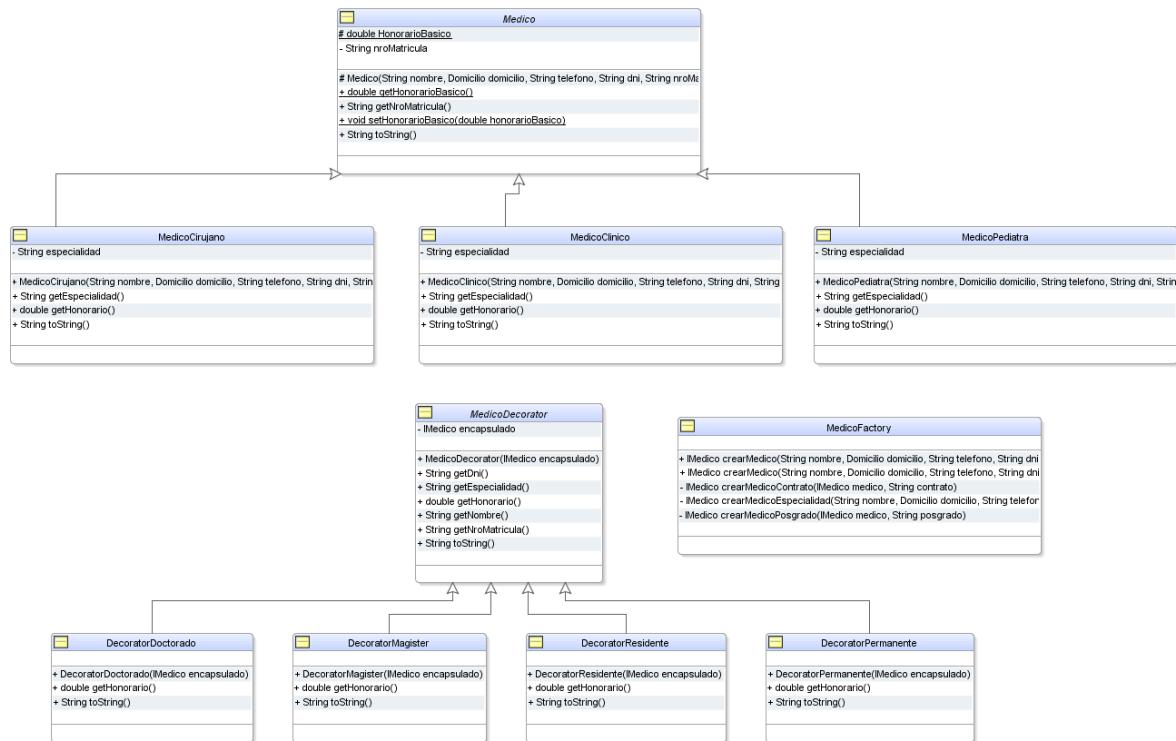
## Diagrama de Clases (UML)



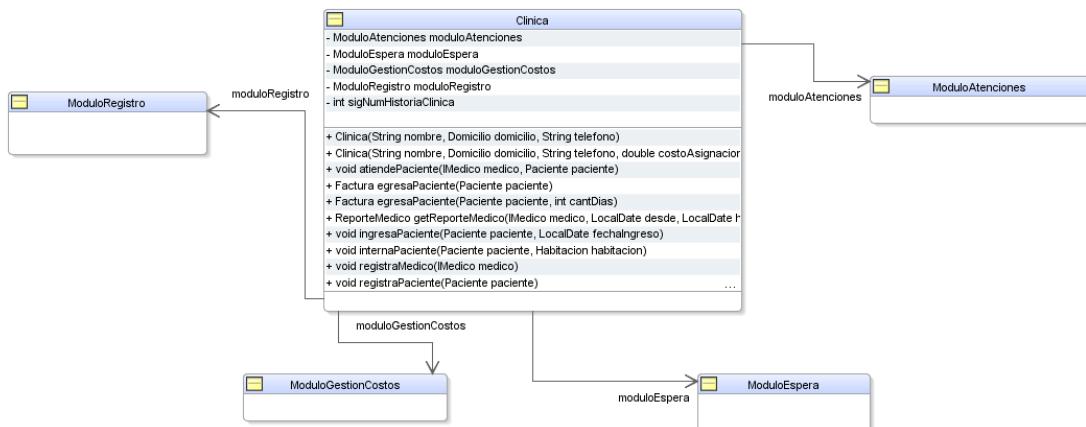
### 2.1 Relaciones entre las clases Clinica, Paciente y Medico.



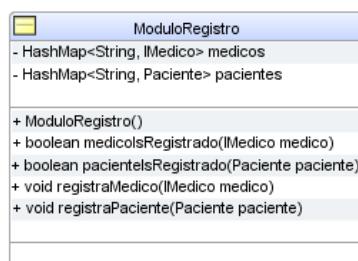
### 2.2 Clase Paciente, sus clases hijas y PacienteFactory.



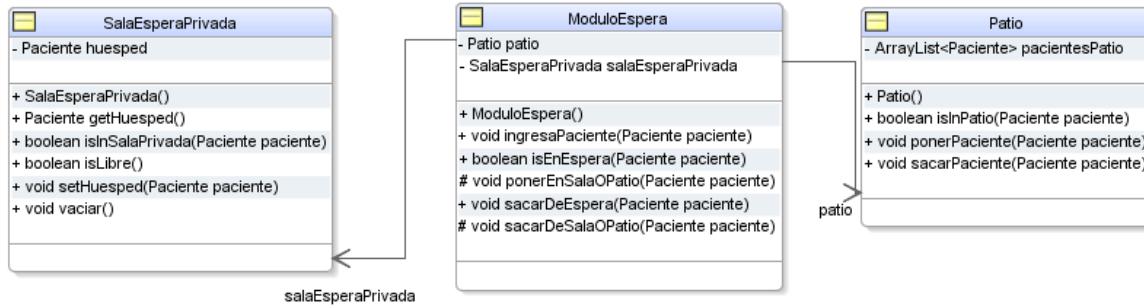
## 2.3 Clase Medico, sus decoradores y MedicoFactory



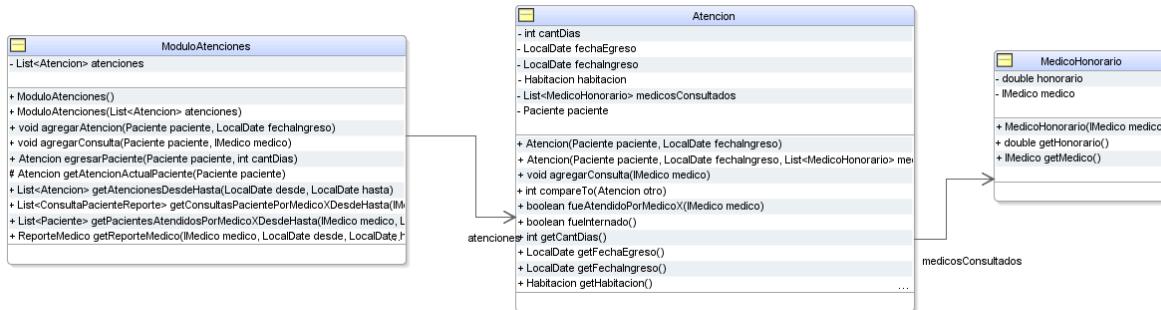
## 2.4 Clase Clinica y los módulos que la componen.



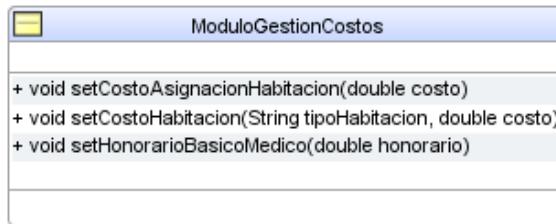
## 2.5 Clase que conforma el módulo de registro.



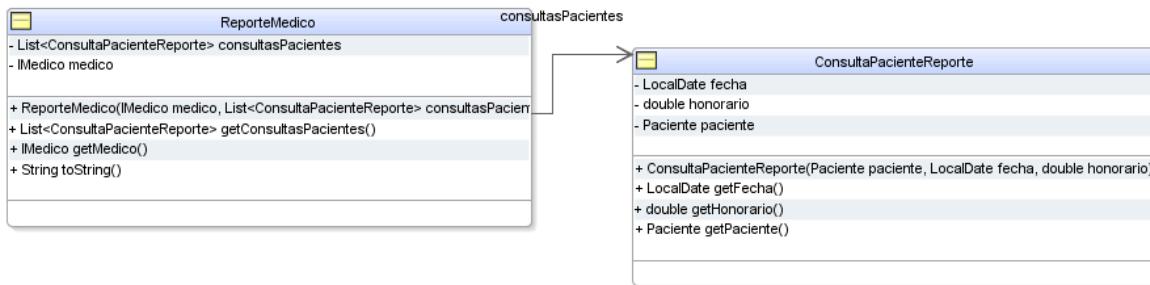
## 2.6 Clases que conforman el módulo de espera.



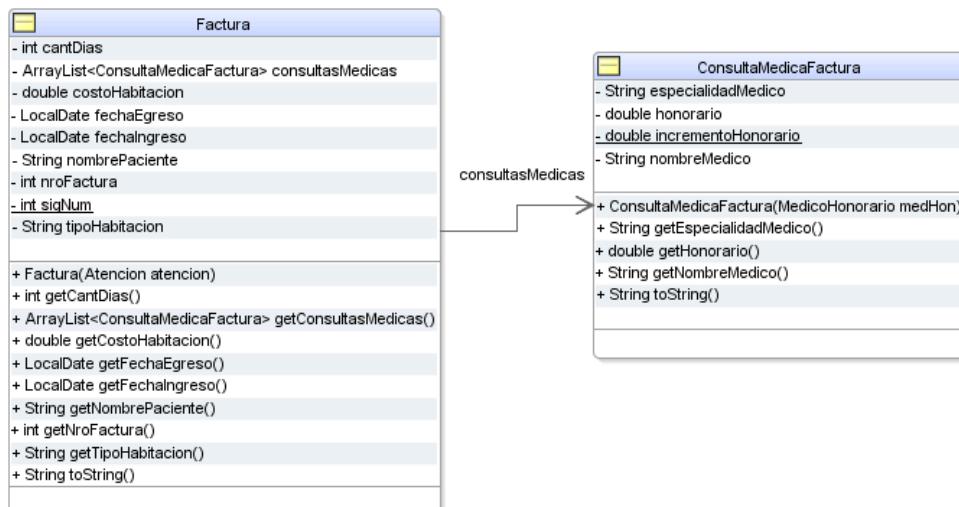
## 2.7 Clases que conforman el módulo de atención.



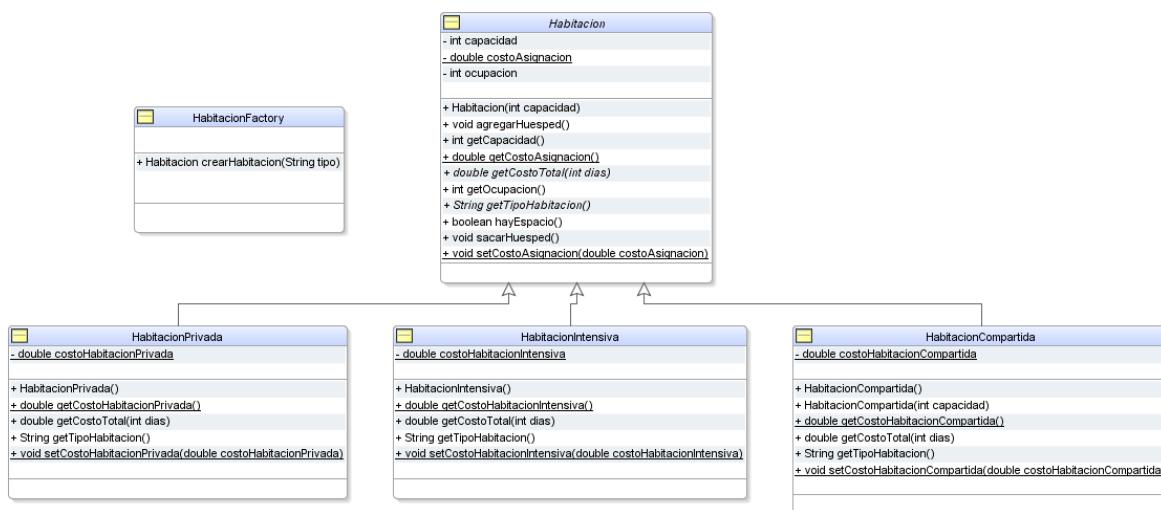
## 2.8 Clase que conforma el módulo de gestión de costos.



## 2.9 Clase ReporteMedico.



## 2.10 Clase Factura.



## 2.11 Clase Habitacion, sus clases hijas y HabitacionFactory.