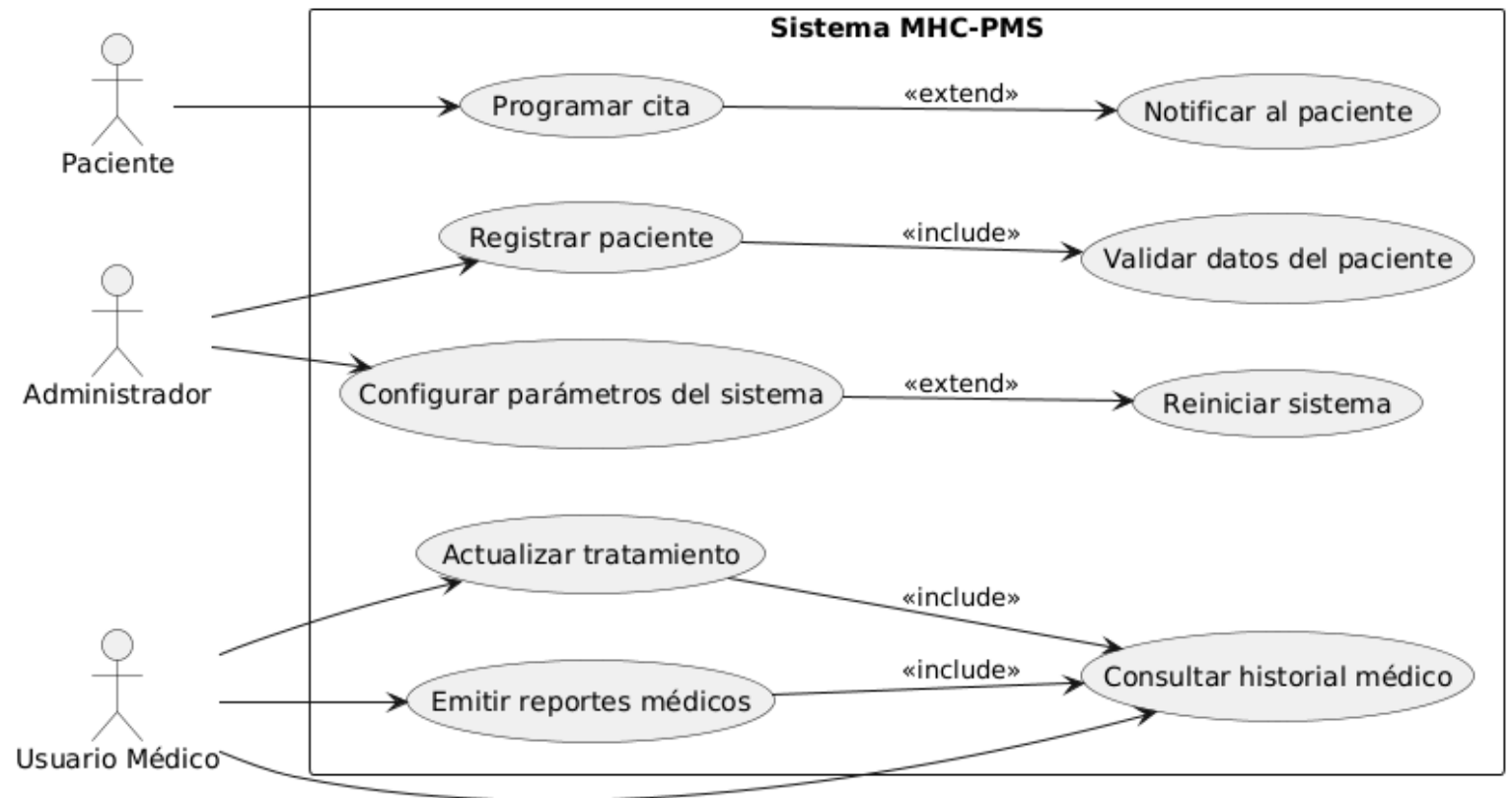


- Con la notación estructurada que se muestra en la figura Descripción del Cu del reporte del clima, especifique los casos de uso de la estación meteorológica para Report status (reporte de estatus) y Reconfigure (reconfigurar).

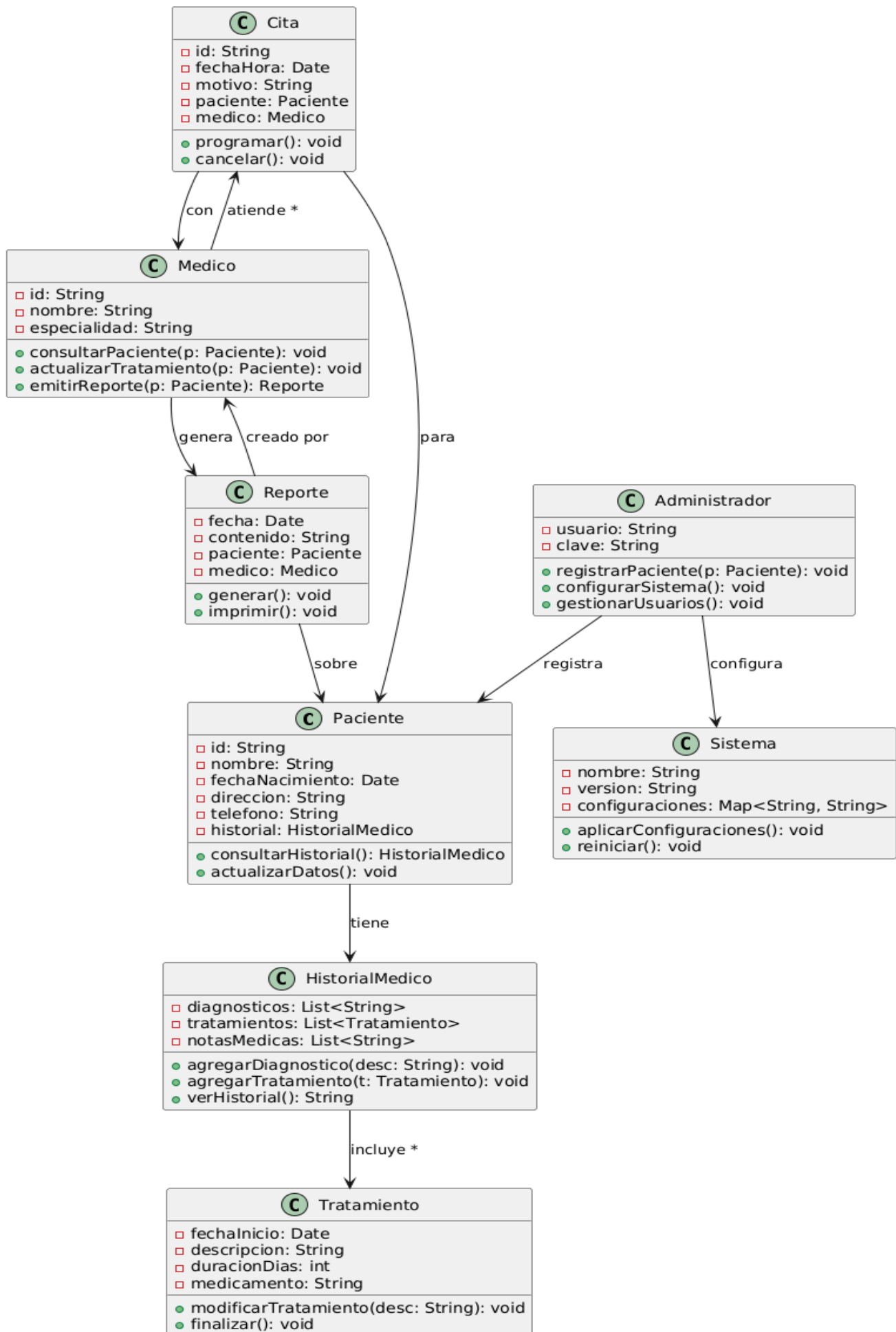
<b>Sistema</b>	Estación Metereológica
<b>Caso de Uso</b>	Reporte de Status
<b>Actores</b>	Sistema de Información Metereológica, Estación Metereológica
<b>Datos</b>	La estación metereológica manda información de estatus al sistema de información metereológica. Esta información puede incluir el estado de los sensores (activo/inactivo), nivel de batería, errores registrados.
<b>Estímulos</b>	El sistema de información metereológica establece un vínculo de comunicación con la estación y solicita un reporte de estado.
<b>Respuesta</b>	La estación metereológica transmite su reporte de estatus con los datos de funcionamiento al sistema de información metereológica.
<b>Comentarios</b>	Este reporte puede ser solicitado periódicamente (ej.: cada 24 horas) o bajo demanda cuando se sospecha de un mal funcionamiento.

<b>Sistema</b>	Estación Metereológica
<b>Caso de Uso</b>	Reconfigurar
<b>Actores</b>	Sistema de Control, Estación Metereológica
<b>Datos</b>	El sistema de Control envía nuevos parámetros de configuración a la estación metereológica. Estos pueden incluir cambios en la frecuencia de reportes, activación o desactivación de sensores.
<b>Estímulos</b>	El sistema de Control establece una comunicación con la estación metereológica para enviar la nueva configuración.
<b>Respuesta</b>	La estación metereológica actualiza su configuración según las instrucciones recibidas.
<b>Comentarios</b>	Algunas reconfiguraciones pueden requerir reinicio.

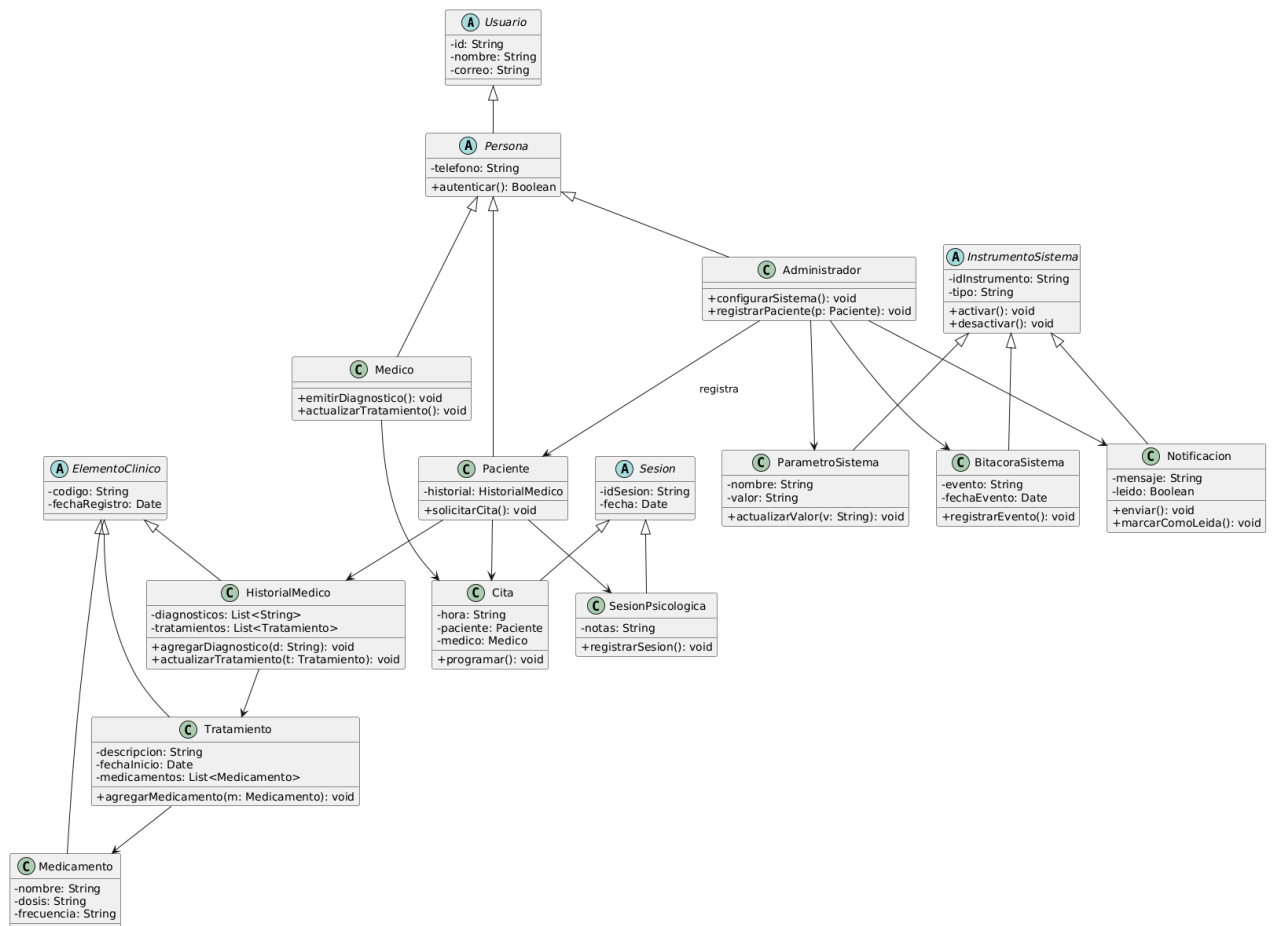
2. Suponga que el MHC-PMS Ver Cap5 Somerville9 los modelos se desarrollará usando un enfoque orientado a objetos. Dibuje un diagrama de caso de uso, que muestre al menos seis posibles casos de uso para este sistema.



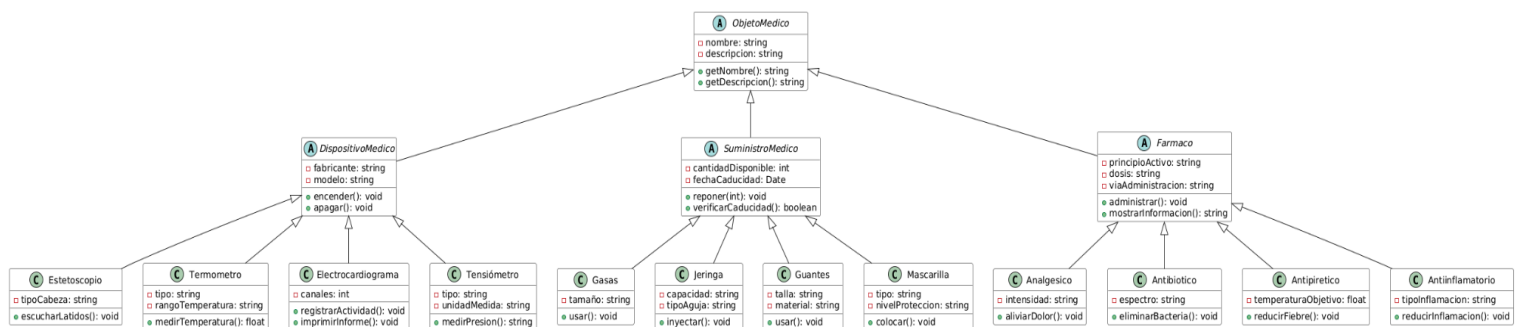
3. Con la notación gráfica UML para clases de objetos, diseñe las siguientes clases de objetos, e identifique los atributos y las operaciones. Use su experiencia para decidir sobre los atributos y las operaciones que deban asociarse con estos objetos:



4. Con los objetos de la estación meteorológica, identificados en la figura Objetos de la estación meteorológica como punto de partida, identifique más objetos que puedan usarse en este sistema. Diseñe una jerarquía de herencia para los objetos que haya identificado.



Jerarquía de herencia para los objetos identificados.



## Código Herramienta planUML **Caso de uso** Sistema MHC-PMS:

```
@startuml
left to right direction

actor "Administrador" as Admin
actor "Usuario Médico" as Medico
actor "Paciente" as Paciente

rectangle "Sistema MHC-PMS" {

    usecase "Registrar paciente" as UC1
    usecase "Consultar historial médico" as UC2
    usecase "Actualizar tratamiento" as UC3
    usecase "Programar cita" as UC4
    usecase "Emitir reportes médicos" as UC5
    usecase "Configurar parámetros del sistema" as UC6

    usecase "Validar datos del paciente" as UC7
    usecase "Notificar al paciente" as UC8
    usecase "Reiniciar sistema" as UC9

    ' Relaciones <<include>>
    UC1 --> UC7 : <<include>>
    UC3 --> UC2 : <<include>>
    UC5 --> UC2 : <<include>>

    ' Relaciones <<extend>>
    UC4 --> UC8 : <<extend>>
    UC6 --> UC9 : <<extend>>
}

' Relación de actores con casos de uso principales
Admin --> UC6
Admin --> UC1
Medico --> UC2
Medico --> UC3
Medico --> UC5
Paciente --> UC4

@enduml
```

## Código Herramienta planUML **Diagrama de clases** Sistema MHC-PMS:

```
@startuml

' Clases principales
class Paciente {
    - id: String
    - nombre: String
    - fechaNacimiento: Date
    - direccion: String
    - telefono: String
    - historial: HistorialMedico
}
```

```
+ consultarHistorial(): HistorialMedico
+ actualizarDatos(): void
}
```

```
class HistorialMedico {
- diagnosticos: List<String>
- tratamientos: List<Tratamiento>
- notasMedicas: List<String>

+ agregarDiagnostico(desc: String): void
+ agregarTratamiento(t: Tratamiento): void
+ verHistorial(): String
}
```

```
class Tratamiento {
- fechaInicio: Date
- descripcion: String
- duracionDias: int
- medicamento: String

+ modificarTratamiento(desc: String): void
+ finalizar(): void
}
```

```
class Medico {
- id: String
- nombre: String
- especialidad: String

+ consultarPaciente(p: Paciente): void
+ actualizarTratamiento(p: Paciente): void
+ emitirReporte(p: Paciente): Reporte
}
```

```
class Cita {
- id: String
- fechaHora: Date
- motivo: String
- paciente: Paciente
- medico: Medico

+ programar(): void
+ cancelar(): void
}
```

```
class Reporte {
- fecha: Date
- contenido: String
- paciente: Paciente
- medico: Medico

+ generar(): void
+ imprimir(): void
}
```

```

class Administrador {
    - usuario: String
    - clave: String

    + registrarPaciente(p: Paciente): void
    + configurarSistema(): void
    + gestionarUsuarios(): void
}

class Sistema {
    - nombre: String
    - version: String
    - configuraciones: Map<String, String>

    + aplicarConfiguraciones(): void
    + reiniciar(): void
}

' Relaciones
Paciente --> HistorialMedico : tiene
HistorialMedico --> Tratamiento : incluye *
Medico --> Reporte : genera
Medico --> Cita : atiende *
Cita --> Paciente : para
Cita --> Medico : con
Administrador --> Paciente : registra
Administrador --> Sistema : configura
Reporte --> Paciente : sobre
Reporte --> Medico : creado por

@enduml

```

Código Herramienta planUML **jerarquía de herencia** Sistema MHC-PMS:

```

@startuml

skinparam classAttributeIconSize 0

' ABSTRACT CLASSES
abstract class Usuario {
    - id: String
    - nombre: String
    - correo: String
}

abstract class Persona extends Usuario {
    - telefono: String
    + autenticar(): Boolean
}

```

```
}
```

```
abstract class ElementoClinico {  
    - codigo: String  
    - fechaRegistro: Date  
}
```

```
abstract class Sesion {  
    - idSesion: String  
    - fecha: Date  
}
```

```
abstract class InstrumentoSistema {  
    - idInstrumento: String  
    - tipo: String  
    + activar(): void  
    + desactivar(): void  
}
```

' CLASES DERIVADAS

```
class Paciente extends Persona {  
    - historial: HistorialMedico  
    + solicitarCita(): void  
}
```

```
class Medico extends Persona {  
    + emitirDiagnostico(): void  
    + actualizarTratamiento(): void  
}
```

```
class Administrador extends Persona {  
    + configurarSistema(): void  
    + registrarPaciente(p: Paciente): void  
}
```

```
class HistorialMedico extends ElementoClinico {  
    - diagnosticos: List<String>  
    - tratamientos: List<Tratamiento>  
    + agregarDiagnostico(d: String): void  
    + actualizarTratamiento(t: Tratamiento): void  
}
```

```
class Tratamiento extends ElementoClinico {  
    - descripcion: String  
    - fechaInicio: Date  
    - medicamentos: List<Medicamento>  
    + agregarMedicamento(m: Medicamento): void  
}
```

```
class Medicamento extends ElementoClinico {  
    - nombre: String  
    - dosis: String  
    - frecuencia: String  
}
```



```

class Cita extends Sesion {
    - hora: String
    - paciente: Paciente
    - medico: Medico
    + programar(): void
}

class SesionPsicologica extends Sesion {
    - notas: String
    + registrarSesion(): void
}

class Notificacion extends InstrumentoSistema {
    - mensaje: String
    - leido: Boolean
    + enviar(): void
    + marcarComoLeida(): void
}

class BitacoraSistema extends InstrumentoSistema {
    - evento: String
    - fechaEvento: Date
    + registrarEvento(): void
}

class ParametroSistema extends InstrumentoSistema {
    - nombre: String
    - valor: String
    + actualizarValor(v: String): void
}

' RELACIONES
Paciente --> HistorialMedico
HistorialMedico --> Tratamiento
Tratamiento --> Medicamento
Paciente --> Cita
Medico --> Cita
Paciente --> SesionPsicologica
Administrador --> ParametroSistema
Administrador --> BitacoraSistema
Administrador --> Notificacion
Administrador --> Paciente : registra

@enduml

```

Código Herramienta planUML **jerarquía de herencia solo objetos** Sistema MHC-PMS:

```

@startuml
skinparam class {
    BackgroundColor White
    BorderColor Black
}

' Clase base general

```

```
abstract class ObjetoMedico {  
    - nombre: string  
    - descripcion: string  
    + getNombre(): string  
    + getDescripcion(): string  
}
```

' Categorías principales

```
abstract class DispositivoMedico {  
    - fabricante: string  
    - modelo: string  
    + encender(): void  
    + apagar(): void  
}
```

```
abstract class SuministroMedico {  
    - cantidadDisponible: int  
    - fechaCaducidad: Date  
    + reponer(int): void  
    + verificarCaducidad(): boolean  
}
```

```
abstract class Farmaco {  
    - principioActivo: string  
    - dosis: string  
    - viaAdministracion: string  
    + administrar(): void  
    + mostrarInformacion(): string  
}
```

' Dispositivos específicos

```
class Estetoscopio {  
    - tipoCabeza: string  
    + escucharLatidos(): void  
}
```

```
class Termometro {  
    - tipo: string  
    - rangoTemperatura: string  
    + medirTemperatura(): float  
}
```

```
class Electrocardiograma {  
    - canales: int  
    + registrarActividad(): void  
    + imprimirInforme(): void  
}
```

```
class Tensiómetro {  
    - tipo: string  
    - unidadMedida: string  
    + medirPresion(): string  
}
```

' Suministros específicos

```
class Gasas {  
    - tamaño: string  
    + usar(): void  
}
```

```
class Jeringa {  
    - capacidad: string  
    - tipoAguja: string  
    + inyectar(): void  
}
```

```
class Guantes {  
    - talla: string  
    - material: string  
    + usar(): void  
}
```

```
class Mascarilla {  
    - tipo: string  
    - nivelProteccion: string  
    + colocar(): void  
}
```

' Fármacos específicos

```
class Analgesico {  
    - intensidad: string  
    + aliviarDolor(): void  
}
```

```
class Antibiotico {  
    - espectro: string  
    + eliminarBacteria(): void  
}
```

```
class Antipiretico {  
    - temperaturaObjetivo: float  
    + reducirFiebre(): void  
}
```

```
class Antiinflamatorio {  
    - tipoInflamacion: string  
    + reducirInflamacion(): void  
}
```

' Relaciones de herencia

ObjetoMedico <|-- DispositivoMedico

ObjetoMedico <|-- SuministroMedico

ObjetoMedico <|-- Farmaco

DispositivoMedico <|-- Estetoscopio

DispositivoMedico <|-- Termometro

DispositivoMedico <|-- Electrocardiograma

DispositivoMedico <|-- Tensiómetro

SuministroMedico <|-- Gasas

SuministroMedico <|-- Jeringa  
SuministroMedico <|-- Guantes  
SuministroMedico <|-- Mascarilla

Farmaco <|-- Analgesico  
Farmaco <|-- Antibiotico  
Farmaco <|-- Antipiretico  
Farmaco <|-- Antiinflamatorio

@enduml