Entregable 10 – Diseñador.

Ruiz Avilés Fernanda Michelle

22140756

Ingeniería en Sistemas Computacionales, Instituto Tecnológico de México campus Querétaro.

Ingeniería de Software.

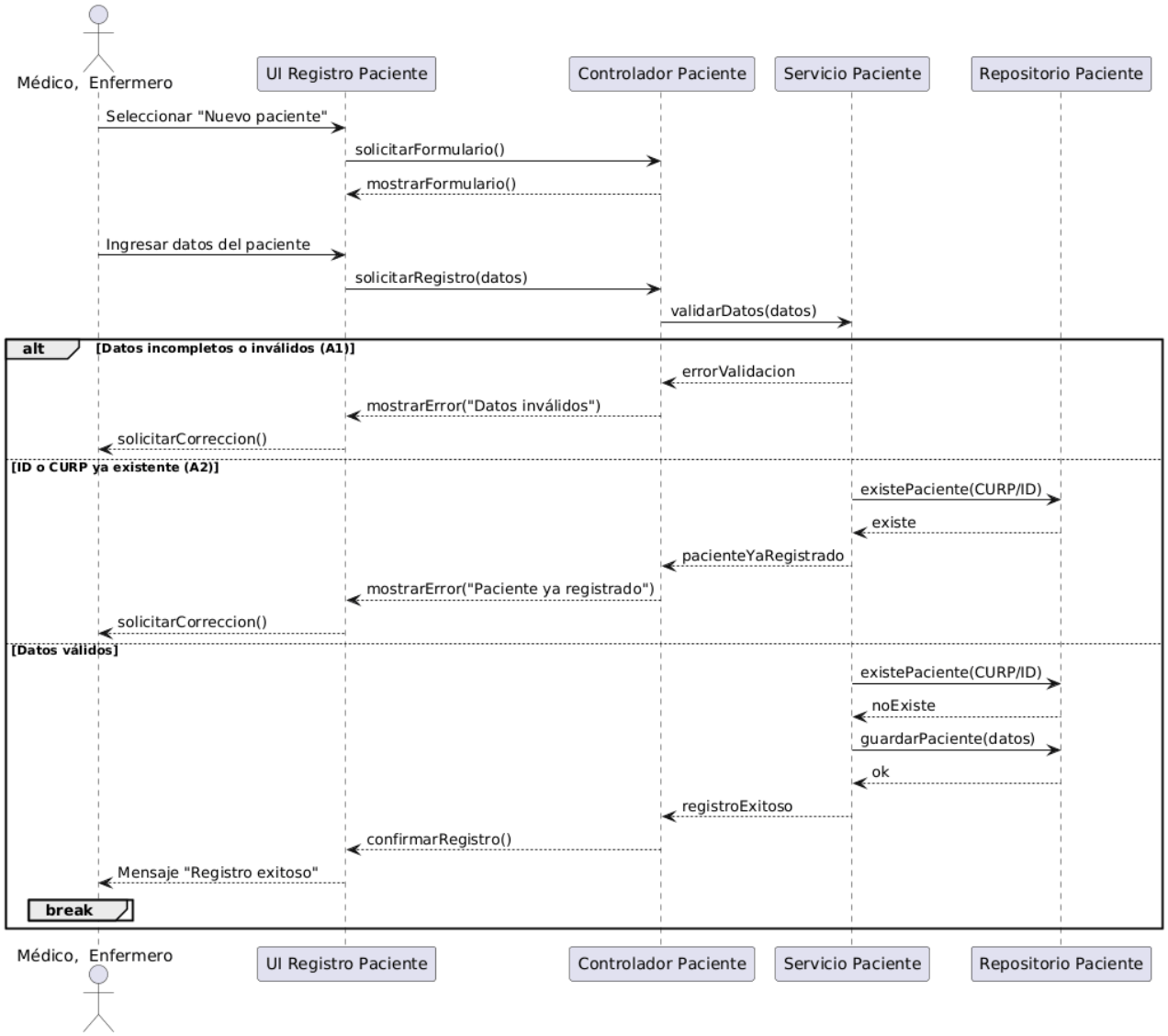
6A

SCC1007

Villeda Maldonado Julio Alejandro

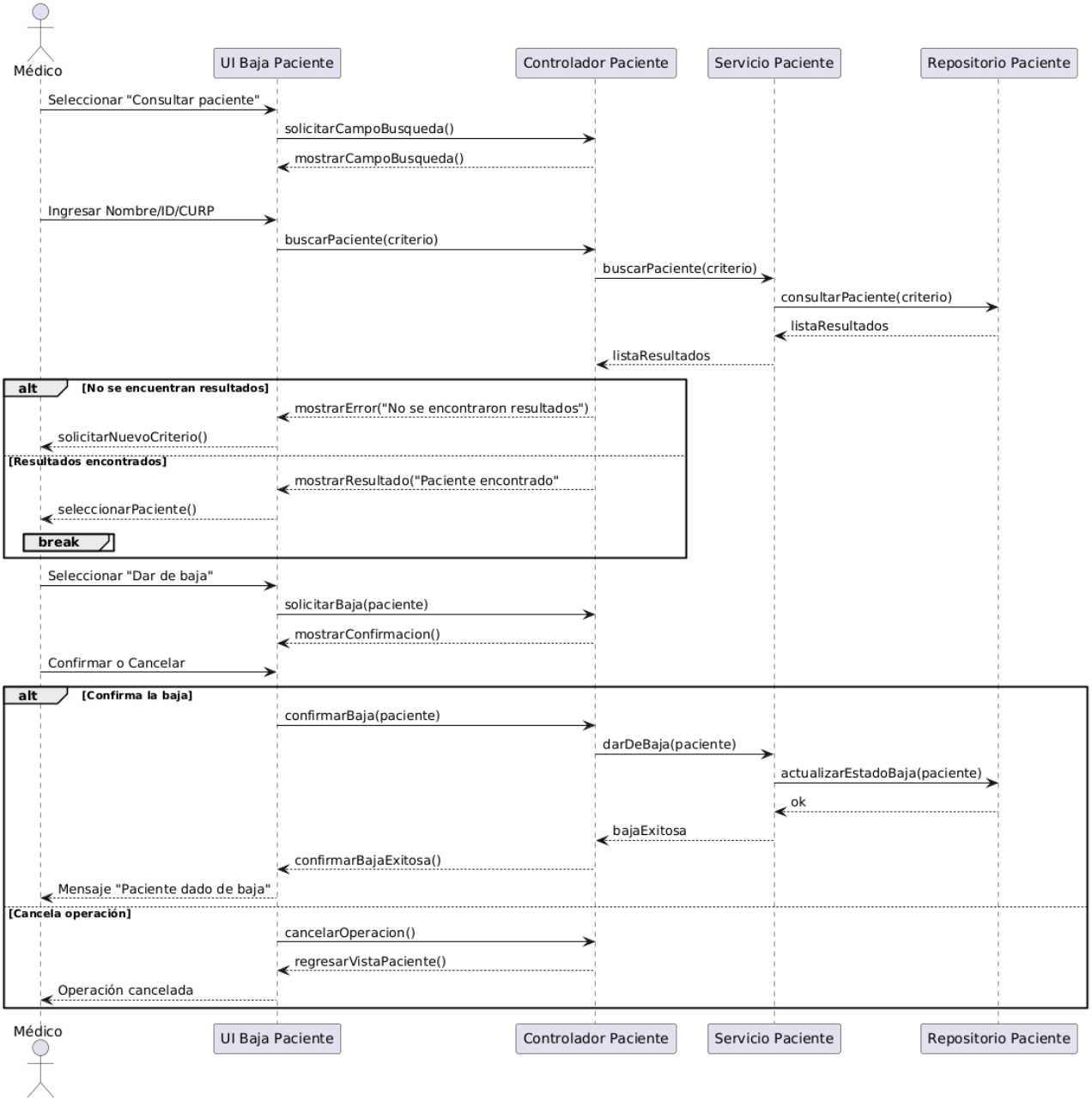
24 de febrero de 2026

Caso 1. Registrar paciente



```
@startuml
actor "Médico,  Enfermero" as Actor

participant "UI Registro Paciente" as UI
participant "Controlador Paciente" as CP
participant "Servicio Paciente" as SP
participant "Repositorio Paciente" as RP

Actor -> UI : Seleccionar "Nuevo paciente"
UI -> CP : solicitarFormulario()
CP --> UI : mostrarFormulario()

loop Reintentar captura mientras haya error (A1/A2)
Actor -> UI : Ingresar datos del paciente
UI -> CP : solicitarRegistro(datos)
CP -> SP : validarDatos(datos)

alt Datos incompletos o inválidos (A1)
SP --> CP : errorValidacion
CP --> UI : mostrarError("Datos inválidos")
UI --> Actor : solicitarCorreccion()

else ID o CURP ya existente (A2)
SP -> RP : existePaciente(CURP/ID)
RP --> SP : existe
SP --> CP : pacienteYaRegistrado
CP --> UI : mostrarError("Paciente ya registrado")
UI --> Actor : solicitarCorreccion()

else Datos válidos
SP -> RP : existePaciente(CURP/ID)
RP --> SP : noExiste
SP -> RP : guardarPaciente(datos)
RP --> SP : ok
SP --> CP : registroExitoso
CP --> UI : confirmarRegistro()
UI --> Actor : Mensaje "Registro exitoso"
  break
 end
end
@enduml
```

Caso 2. Dar de baja paciente



```
@startuml
actor Médico

participant "UI Baja Paciente" as UI
participant "Controlador Paciente" as CP
participant "Servicio Paciente" as SP
participant "Repositorio Paciente" as RP

Médico -> UI : Seleccionar "Consultar paciente"
UI -> CP : solicitarCampoBusqueda()
CP --> UI : mostrarCampoBusqueda()

loop Reintentar búsqueda mientras no haya resultados
Médico -> UI : Ingresar Nombre/ID/CURP
UI -> CP : buscarPaciente(criterio)
CP -> SP : buscarPaciente(criterio)
SP -> RP : consultarPaciente(criterio)
RP --> SP : listaResultados
SP --> CP : listaResultados

alt [No se encuentran resultados]
CP --> UI : mostrarError("No se encontraron resultados")
UI --> Médico : solicitarNuevoCriterio()

else [Resultados encontrados]
CP --> UI : mostrarResultado(listaResultados)
UI --> Médico :seleccionarPaciente(pacienteID)
break
end
end

Médico -> UI : Seleccionar "Dar de baja"
UI -> CP : solicitarBaja(paciente)
CP --> UI : mostrarConfirmacion()
Médico -> UI : Confirmar o Cancelar

alt [Confirma la baja]
  UI -> CP : confirmarBaja(paciente)
  CP -> SP : darDeBaja(paciente)
  SP -> RP : actualizarEstadoBaja(paciente)
  RP --> SP : ok
  SP --> CP : bajaExitosa
  CP --> UI : confirmarBajaExitosa()
  UI --> Médico : Mensaje "Paciente dado de baja"
```

```
else [Cancela operación]
  UI -> CP : cancelarOperacion()
  CP --> UI : regresarVistaPaciente()
  UI --> Médico : Operación cancelada
end
@enduml
```

Justificación.

En la arquitectura se modelo la interacción del sistema usando capas de UI → Controlador → Servicio → Repositorio, para separar las responsabilidades.

En el caso uno de “Registrar pacientes”.En la parte de UIRegistroPaciente se concentra la interacción con el autor (Enfermero/ Médico) y la interfaz despliega el formulario donde se toma la captura de datos del paciente, los mensajes y las confirmaciones. El ControladorPaciente coordina el flujo del caso sin contener reglas de negocio. El ServicioPaciente se concentra en la lógica de la validación de datos y verificación de existencia. El RepositorioPaciente se encarga del acceso de datos.

Se utilizó alt ya que el caso tiene decisiones alternativas para la validación de Datos inválidos, Paciente ya registrado y, Datos válidos y no existentes. Cada condición tiene una respuesta distinta dentro del sistema. Igual se empleó loop en el caso de Datos inválidos, donde el autor (Enfermero/ Médico) debe corregir la información y volver a intentar el registro.

En el caso dos de “Dar de baja paciente”. La UI Baja Paciente realiza la búsqueda y confirmación. El ControladorPaciente coordina la solicitud de baja. El ServidorPaciente aplica las reglas de negocios para cambiar el estado del paciente. El RepositorioPaciente realiza la actualización del paciente.

Se utilizó loop en la búsqueda ya que si no se encuentra resultados, el sistema debe permitir reintentar hasta que exista una coincidencia. También se empleó alt en No se encontraron resultados, Resultados encontrados, Confirma la baja y Cancela operación, ya que cada una tiene un comportamiento distinto dentro del sistema y se deben modelar las alternativas.