

## Documentación de Casos de Uso:

- Se puede utilizar los diagramas de casos de uso con la documentación dada en puntos anteriores.
- Fijarse siempre en la abstracción del sistema, se pueden dejar de lados detalles triviales o que no aporten mucho a comprender la lógica del proceso que va a ser implementado.
- Al utilizar los diagramas de casos de uso y la descripción de los mismos, permite tener un mejor entendimiento de los requisitos del usuario. Estos posteriormente se pasaran a una etapa funcional haciendo uso de los diagramas de clases y paquetes respectivamente.

**Nota:** Los diagramas de casos de uso también pueden agruparse en paquetes, siendo estos distintos a los paquetes de programación conocidos comúnmente.

- Si consideramos, por ejemplo la metodología ágil ICONIX, la manera cómo se enlazan los diversos componentes gráficos sería de la siguiente manera: (tomado de [1])

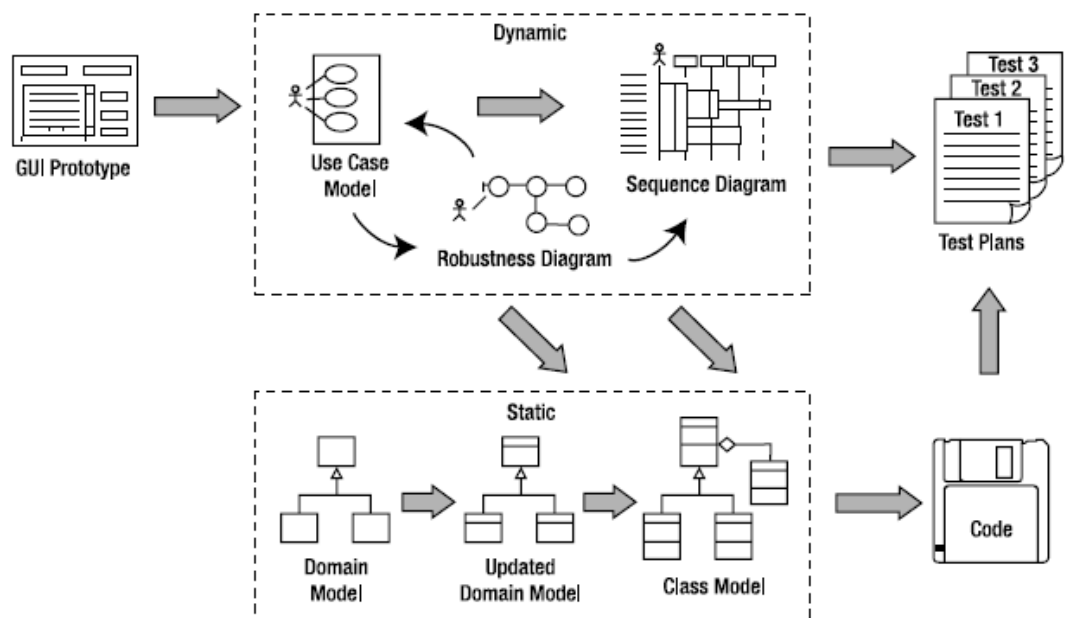


Figura 1: Enlace de las diversas herramientas de UML desde la fase del análisis a la implementación.

- Cómo se enlazan los casos de uso de forma gráfica usando UML con la descripción de los casos de uso? Tome en cuenta el siguiente ejemplo: (tomado de [2])

Use Case Name: Make Old Patient Appt	ID: 2	Importance Level: Low
Primary Actor: Old Patient	Use Case Type: Detail, Essential	
Stakeholders and Interests: Old Patient – wants to make, change, or cancel an appointment Doctor – wants to ensure patient's needs are met in a timely manner		
Brief Description: This use case describes how we make an appointment as well as changing or canceling an appointment for a previously seen patient.		
Trigger: Patient calls and asks for a new appointment or asks to cancel or change an existing appointment		
Type: External		
Relationships: Association: Old Patient Include: Extend: Update Patient Information Generalization: Manage Appointments		
Normal Flow of Events: 1. The Patient contacts the office regarding an appointment. 2. The Patient provides the Receptionist with his or her name and address. 3. If the Patient's information has changed Execute the Update Patient Information use case. 4. If the Patient's payment arrangements has changed Execute the Make Payments Arrangements use case. 5. The Receptionist asks Patient if he or she would like to make a new appointment, cancel an existing appointment, or change an existing appointment. If the patient wants to make a new appointment, the S-1: new appointment subflow is performed. If the patient wants to cancel an existing appointment, the S-2: cancel appointment subflow is performed. If the patient wants to change an existing appointment, the S-3: change appointment subflow is performed. 6. The Receptionist provides the results of the transaction to the Patient.		
SubFlows: S-1: New Appointment 1. The Receptionist asks the Patient for possible appointment times. 2. The Receptionist matches the Patient's desired appointment times with available dates and times and schedules the new appointment. S-2: Cancel Appointment 1. The Receptionist asks the Patient for the old appointment time. 2. The Receptionist finds the current appointment in the appointment file and cancels it. S-3: Change Appointment 1. The Receptionist performs the S-2: cancel appointment subflow. 2. The Receptionist performs the S-1: new appointment subflow.		
Alternate/Exceptional Flows: S-1, 2a1: The Receptionist proposes some alternative appointment times based on what is available in the appointment schedule. S-1, 2a2: The Patient chooses one of the proposed times or decides not to make an appointment.		

Figura 2: Descripción de un caso de uso de atención al paciente.  
El cual se enlaza con el siguiente caso de uso en UML:

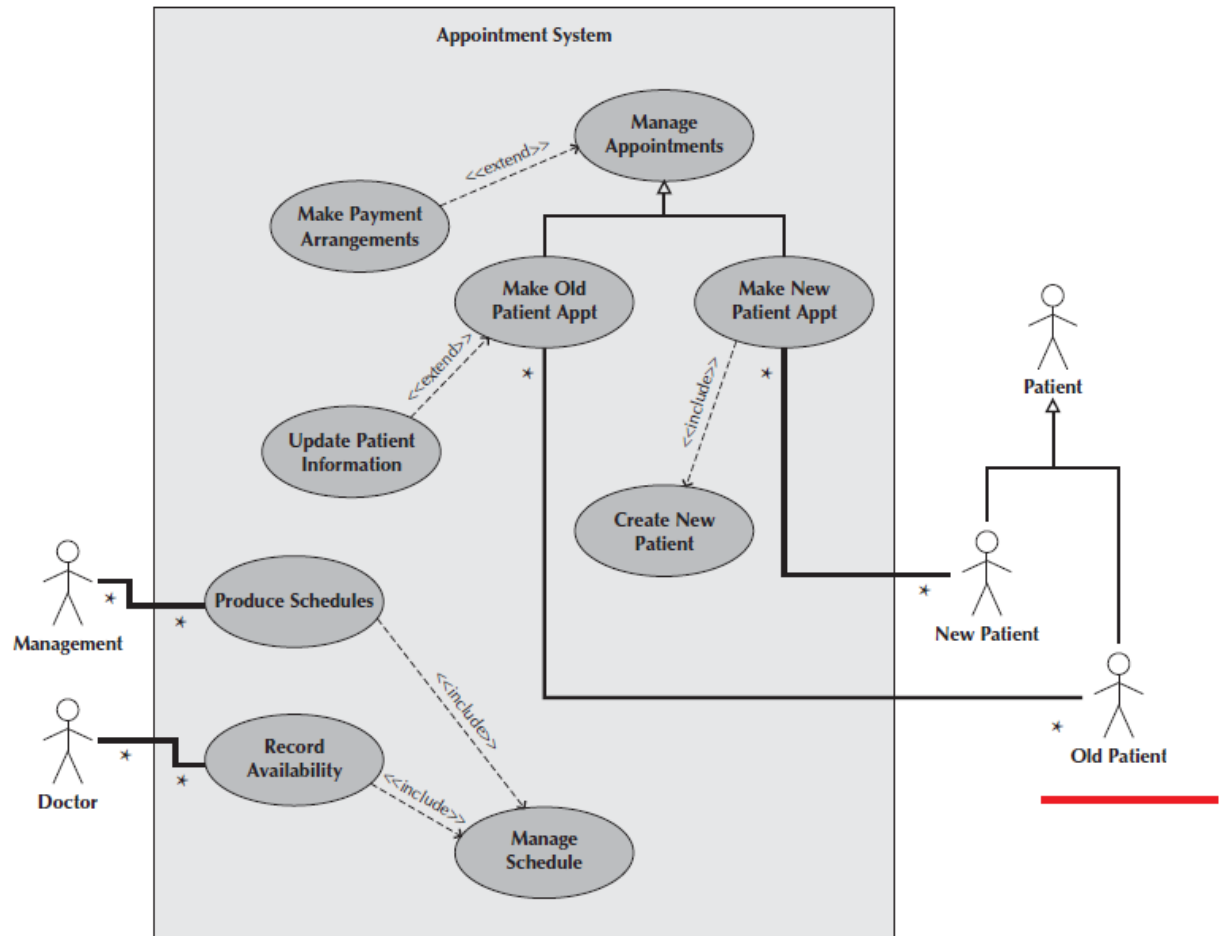


Figura 3: Representación de casos de uso para el problema de atención de pacientes.

Algunos detalles de la gráfica anterior:

- Existe una relación de generalización entre Old Patient y Patient, esta relación pareciese a la que se hace en un sistema de clases con la denominada Herencia.
- Sobre el **Flujo de Eventos** dado en la descripción se encuentran los siguientes:

a) Flujo normal (normal flow of events): Son aquellos que se dan cuando el funcionamiento del caso de uso se da sin inconvenientes.

**Nota:** Recuerde que la funcionalidad se da de forma general, para mayor detalle existen otros tipos de diagramas como el Diagrama de Actividad.

b) Subflujos (subflow): Estos se dan cuando se requiere describir, con mayor detalle, el flujo de información dentro de un caso de uso. Estos guardan relación con los **Diagramas de Actividad**, por ejemplo:

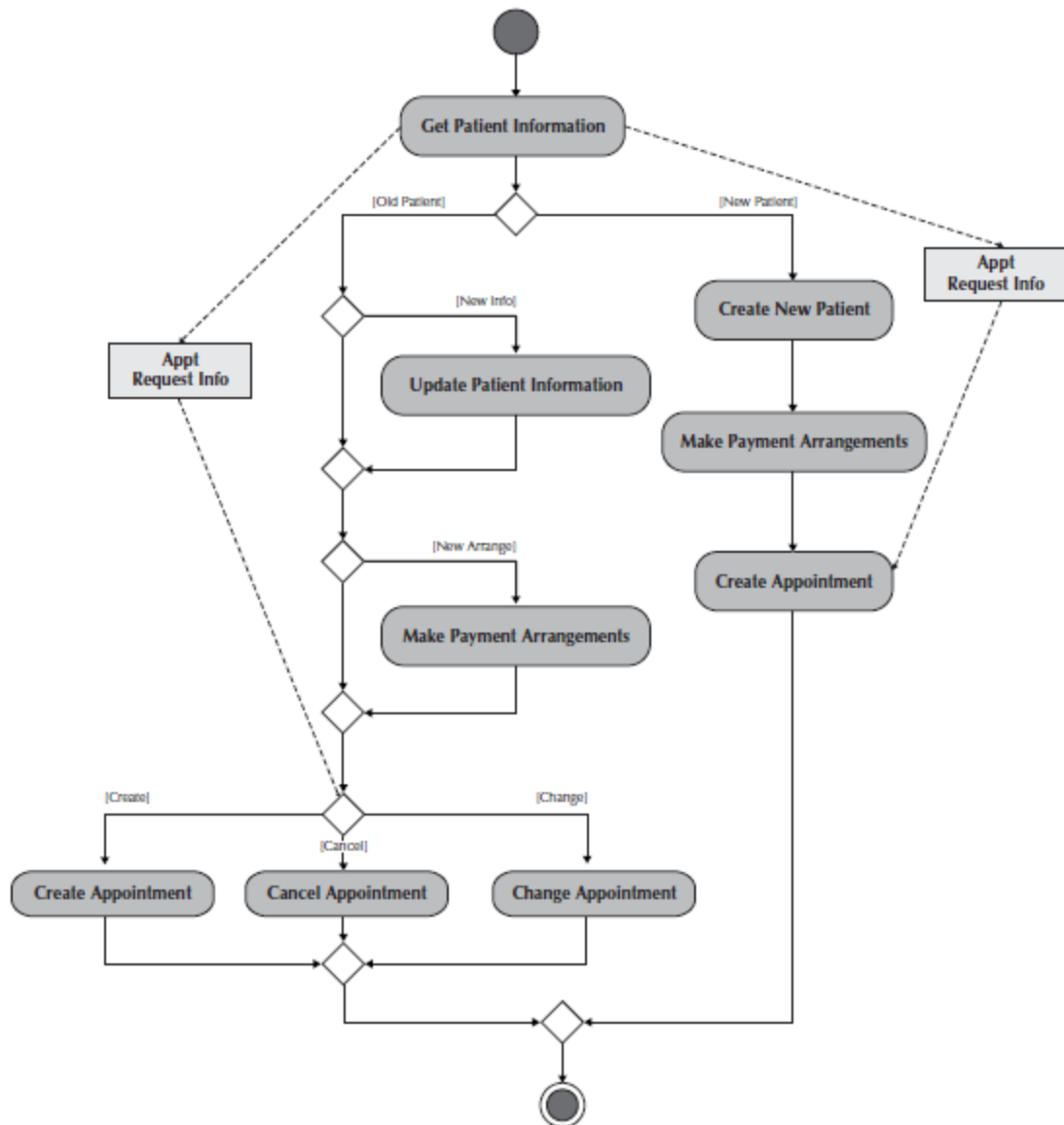


Figura 4: Diagrama de actividades para el caso de uso de Manage Appointments.

**Nota:** En caso algunas actividades puedan ser reagrupadas en un caso de uso, esto es factible haciendo uso del modificador **include**.

#### Diagrama de Actividades:


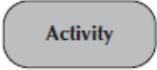
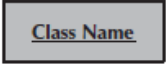





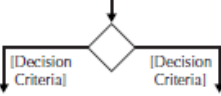
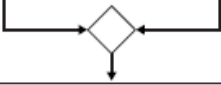


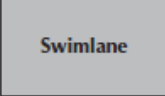
- Este tipo de diagramas es utilizado para modelar el flujo de la información o el comportamiento de los procesos; esto se hace de forma independiente a los objetos.
- Se puede utilizar para modelar procesos extremadamente abstractos como los casos de uso e incluso para el modelamiento de métodos o de funciones que tengan que ver con el código.
- Entre sus elementos tenemos:
  - a) Acciones y Actividades: Pueden representar partes de un proceso que requieran un tratamiento manual o computarizado. Mayormente empiezan con un verbo y terminan con un

sustantivo. La única diferencia es que una Actividad puede ser descompuesta en más Actividades o Acciones.

b) Nodos Objeto: Las actividades pueden modificar o transformar objetos. Son representados mediante un rectángulo y representan el flujo de la información de una actividad hacia la otra.

c) Flujos de control y flujos de objeto: El primero se refiere a cómo se va ejecutando un proceso de negocios. El flujo de objetos indica como los objetos, al estar estos siendo modificados por las actividades, van pasando a través de las acciones y actividades.

En resumen los símbolos a usar serían los siguientes:

<b>An action:</b> <ul style="list-style-type: none"> <li>■ Is a simple, nondecomposable piece of behavior.</li> <li>■ Is labeled by its name.</li> </ul>	
<b>An activity:</b> <ul style="list-style-type: none"> <li>■ Is used to represent a set of actions.</li> <li>■ Is labeled by its name.</li> </ul>	
<b>An object node:</b> <ul style="list-style-type: none"> <li>■ Is used to represent an object that is connected to a set of object flows.</li> <li>■ Is labeled by its class name.</li> </ul>	
<b>A control flow:</b> <ul style="list-style-type: none"> <li>■ Shows the sequence of execution.</li> </ul>	
<b>An object flow:</b> <ul style="list-style-type: none"> <li>■ Shows the flow of an object from one activity (or action) to another activity (or action).</li> </ul>	
<b>An initial node:</b> <ul style="list-style-type: none"> <li>■ Portrays the beginning of a set of actions or activities.</li> </ul>	
<b>A final-activity node:</b> <ul style="list-style-type: none"> <li>■ Is used to stop all control flows and object flows in an activity (or action).</li> </ul>	
<b>A final-flow node:</b> <ul style="list-style-type: none"> <li>■ Is used to stop a specific control flow or object flow.</li> </ul>	
<b>A decision node:</b> <ul style="list-style-type: none"> <li>■ Is used to represent a test condition to ensure that the control flow or object flow only goes down one path.</li> <li>■ Is labeled with the decision criteria to continue down the specific path.</li> </ul>	
<b>A merge node:</b> <ul style="list-style-type: none"> <li>■ Is used to bring back together different decision paths that were created using a decision node.</li> </ul>	
<b>A fork node:</b> Is used to split behavior into a set of parallel or concurrent flows of activities (or actions)	
<b>A join node:</b> Is used to bring back together a set of parallel or concurrent flows of activities (or actions)	
<b>A swimlane:</b> Is used to break up an activity diagram into rows and columns to assign the individual activities (or actions) to the individuals or objects that are responsible for executing the activity (or action)  Is labeled with the name of the individual or object responsible	

**Referencias: (sin un formato en particular)**

[1] Agile Development with ICONIX process, Doug Rosenberg.

[2] System Analysis and Design: An Object Oriented approach with UML, Alan Dennis.