



# UML: Lenguaje de Modelado Unificado

## Tema 6. Diagramas de secuencia



Licencia de Creative Commons.

UML: Lenguaje de Modelado Unificado

Tema 6. Diagramas de secuencia

por: Javier Martín Juan

Esta obra está publicada bajo una licencia [Creative Commons Reconocimiento-NoComercial-CompartirIgual 3.0 España](#) con las siguientes condiciones:



Reconocimiento - Debe reconocer los créditos de la obra de la manera especificada por el autor o el licenciador (pero no de una manera que sugiera que tiene su apoyo o apoyan el uso que hace de su obra).



No comercial - No puede utilizar esta obra para fines comerciales



Compartir bajo la misma licencia - Si altera o transforma esta obra, o genera una obra derivada, sólo puede distribuir la obra generada bajo una licencia idéntica a ésta.

Revisión: dd69bd1969f6

Última actualización: 28 de marzo de 2013

# Índice

<b>1. Diagramas de secuencia</b>	<b>4</b>
Elementos del diagrama de secuencia . . . . .	4
Tipos de mensajes . . . . .	4
Recomendaciones . . . . .	5
Ejemplo: llamada telefónica . . . . .	6
Ejemplo: lavadora . . . . .	6
Fragmentos combinados . . . . .	7
Ejemplo: fragmentos . . . . .	8
<b>2. Ejercicio guiado</b>	<b>10</b>
Solución . . . . .	10

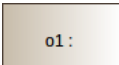


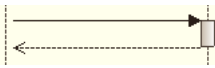
## 1. Diagramas de secuencia

Un **diagrama de secuencia** es un gráfico bidimensional donde el eje vertical representa el tiempo, el eje horizontal muestra objetos del sistema y la interacción entre los objetos se representa mediante flechas que van de unos objetos a otros, ordenadas cronológicamente de arriba abajo. Son un tipo de **diagramas de interacción**, que a su vez es una categoría de **diagramas de comportamiento**.

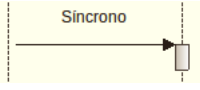
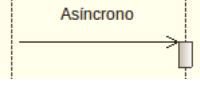



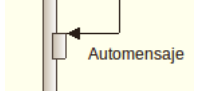
Los diagramas de secuencia se utilizan cuando queremos expresar qué objetos se relacionan con qué objetos, enfatizando el orden en que lo hacen y qué tipo de mensajes se envían entre sí.

También permiten expresar estructuras de control (condiciones y repeticiones), pero los diagramas de secuencia no están pensados para eso y debemos evitar incluir lógica procedimental compleja en ellos.

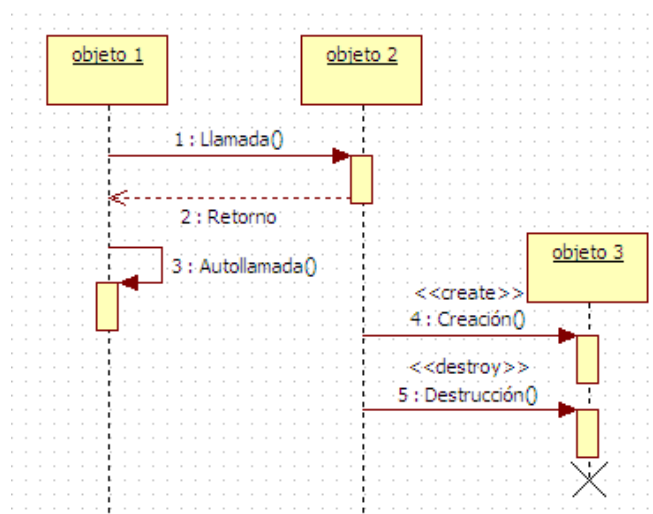
### Elementos del diagrama de secuencia

	<p><b>Objetos:</b> son elementos que participan en el diagrama. Y son instancias de una clase. Se representan por un rectángulo con el nombre del objeto escrito del siguiente modo: “Objeto : Clase”.</p>
	<p><b>Línea de vida:</b> indica la existencia de un objeto (desde que se crea hasta que se destruye) mediante una línea discontinua. El fin de la vida se expresa como un aspa.</p>
	<p><b>Activación:</b> indica cuándo el objeto está realizando una tarea concreta. Equivale al tiempo durante el cual se está ejecutando el método o función. Se expresa como un rectángulo a lo largo de la línea de vida.</p>
	<p><b>Mensaje:</b> la comunicación entre objetos y activaciones. Los mensajes pueden ser síncronos, asíncronos, de creación, de destrucción, de llamada o de retorno. Se representan mediante flechas y van <b>ordenados cronológicamente</b> de arriba abajo.</p>

## Tipos de mensajes

	<p><b>Mensajes síncronos</b> un mensaje es síncrono cuando el objeto llamador se detiene hasta que se obtiene respuesta del objeto llamado. Si estamos modelando una implementación de código (Java, C++, etc.), normalmente todas las llamadas a métodos se consideran síncronas. Se representa como una flecha con la punta triangular rellena.</p>
	<p><b>Mensajes asíncronos:</b> el objeto llamador continúa su ejecución sin esperar la respuesta del objeto llamado. Un ejemplo de mensaje asíncrono en la tecnología <i>AJAX</i> es el objeto <i>XMLHttpRequest</i> de <i>JavaScript</i>: permite enviar datos a un servidor y continuar la ejecución sin esperar a recibir respuesta. Se representa como una flecha con la punta en V.</p>
	<p><b>Respuestas:</b> puede ser la respuesta a un mensaje síncrono o asíncrono, aunque generalmente en los mensajes síncronos se omite la respuesta. Se representa con una línea discontinua acabada en V.</p>
	<p><b>Creación:</b> sirve para crear nuevos objetos. Equivale a la llamada a un constructor en lenguajes orientados a objetos. Se representa como un mensaje asíncrono que apunta al principio de la línea de vida del nuevo objeto.</p>
	<p><b>Destrucción:</b> destruye un objeto. Equivale a la llamada a un destructor en <i>LOO</i>. Se representa como un mensaje asíncrono que apunta al final de la línea de vida del objeto destruido, acabada con un aspa.</p>
	<p><b>Automensaje:</b> puede significar dos cosas: o una llamada recursiva de una operación, o una operación llamando a otra del mismo objeto. Se representa como un mensaje del objeto a sí mismo y con las activaciones anidadas.</p>

En la siguiente figura se pueden ver los elementos más habituales en un diagrama de secuencia:

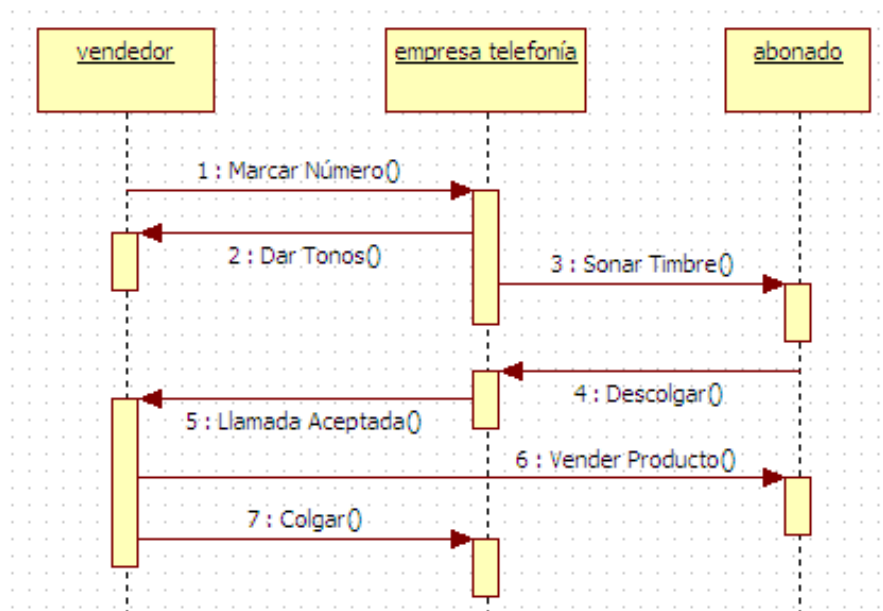


## Recomendaciones

- En general, un diagrama de secuencia sólo representa secuencias de mensajes y no intervalos de tiempo precisos. Si lo que queremos es representar tiempos, deberíamos pensar en usar un diagrama de temporización.
- Hay que **sintetizar**. Es un error intentar incluir todos los métodos de todos los participantes en el diagrama. Tampoco conviene llenar el diagrama de lógica procedimental. Cuanto más complicado sea el diagrama, menos probabilidad hay de que alguien lo lea.

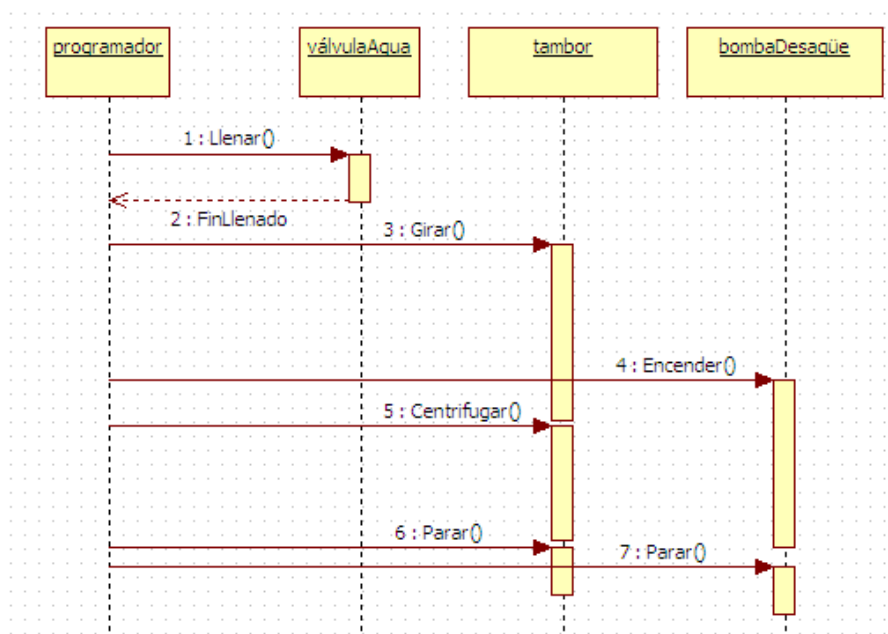
## Ejemplo: llamada telefónica

El siguiente ejemplo modela un sistema de propaganda telefónica. Consiste en una máquina que va llamando a números de teléfono. Cuando el abonado descuelga, una voz robotizada lee la propaganda y, después, cuelga.



## Ejemplo: lavadora

El siguiente diagrama de secuencia modela el comportamiento de una lavadora.



## Fragmentos combinados

Aunque hemos mencionado antes que los diagramas de secuencia no están pensados para expresar lógica procedimental compleja, en UML hay mecanismos para añadir estructuras de control llamados **fragmentos combinados**. Un fragmento combinado es un conjunto de uno o más mensajes encerrados en un marco que se etiqueta con una abreviatura y, opcionalmente, con una condición de guarda. Los fragmentos disponibles son:

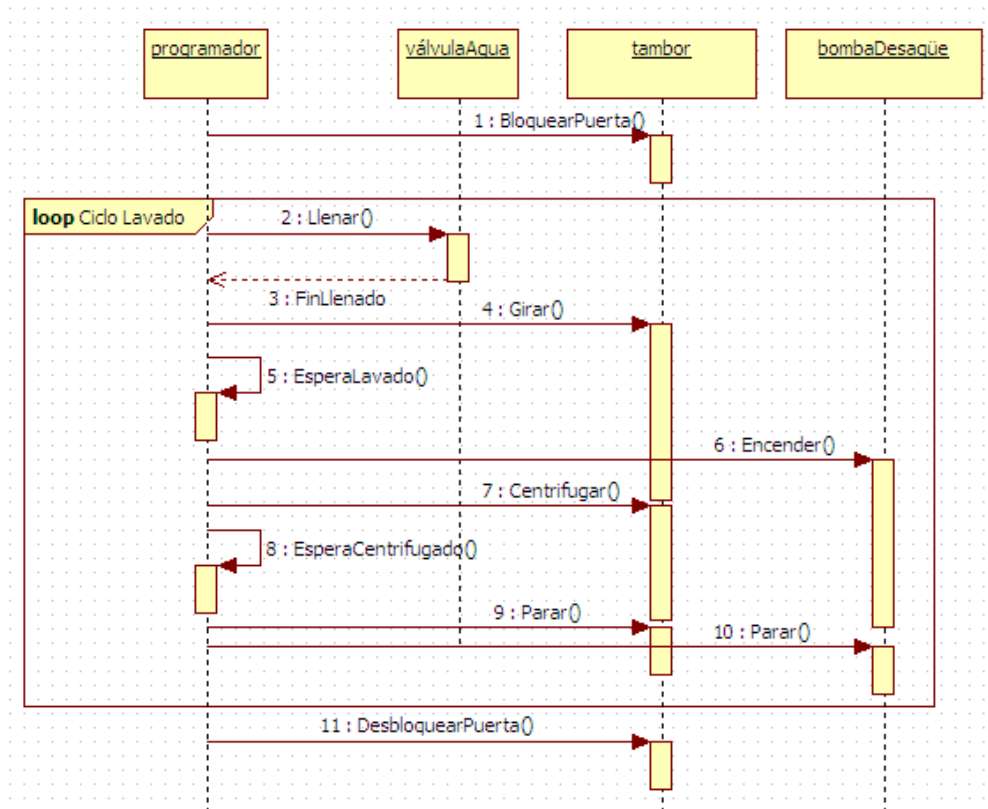
Fragmento	Abreviado	Significado
<b>Bucle</b>	<b>loop</b>	Encierra una serie de mensajes que se repite. La condición de guarda puede indicar el número de repeticiones o una condición de tipo <i>while / do ... while</i>
<b>Alternativa</b>	<b>alt</b>	Modela estructuras <i>if ... then ... else</i> .
<b>Opción</b>	<b>opt</b>	Modela estructuras <i>switch ... case</i> .
<b>Break</b>	<b>break</b>	Significa que después de terminar el fragmento, se interrumpe la ejecución y el resto de secuencia ya no se ejecuta. Modela la instrucción <i>break</i> de lenguajes como C o Java. Para que tenga algo de utilidad, el fragmento debe indicar una condición de guarda, o la interrupción sería incondicional.
<b>Paralelo</b>	<b>par</b>	Modela procesamiento en paralelo.
<b>Secuencia débil</b>	<b>seq</b>	Los mensajes que estén encerrados en este fragmento y no compartan una misma línea de vida pueden ejecutarse en cualquier orden.

Fragmento	Abreviado	Significado (... continuación)
<b>Secuencia estricta</b>	<b>strict</b>	Encierra una serie de mensajes que deben obligatoriamente procesarse en el orden dado.
<b>Crítico</b>	<b>critical</b>	Encierra una región crítica.
<b>Ignorar</b>	<b>ignore</b>	El mensaje o mensajes encerrados son irrelevantes en el contexto del diagrama.
<b>Considerar</b>	<b>consider</b>	Es lo contrario a <b>ignorar</b> : todo mensaje que no aparezca en este fragmento es irrelevante.
<b>Afirmativo</b>	<b>assert</b>	Indica que los únicos mensajes válidos son los encerrados en este fragmento y todos los demás no deberían ocurrir, o la interacción se consideraría inválida. Es útil para modelar casos de prueba.
<b>Negativo</b>	<b>neg</b>	Encierra una serie de mensajes inválidos. Es lo contrario de la afirmación.

### Ejemplo: fragmentos

El siguiente diagrama muestra el uso del fragmento **bucle** en el modelo de la lavadora anterior para expresar la repetición del ciclo de lavado. También se puede observar el uso de **automensajes**:



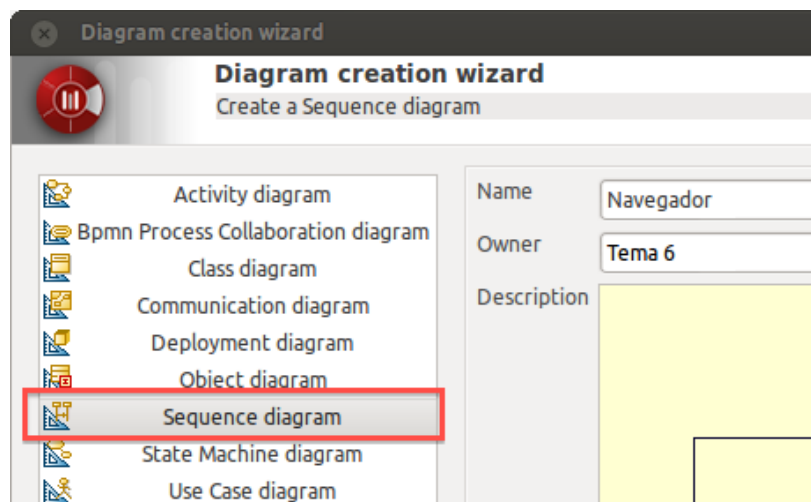


## 2. Ejercicio guiado

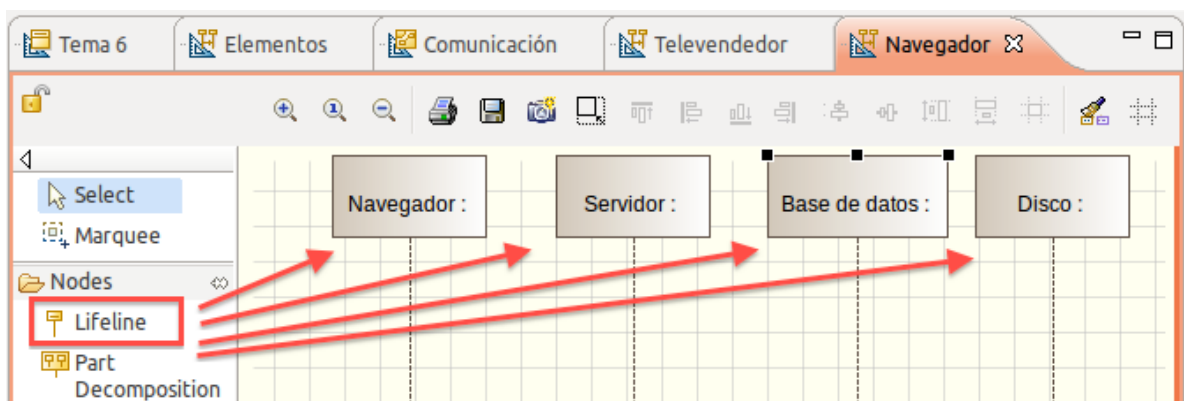
Un usuario entra a una página por Internet tecleando la web en el navegador. El navegador hace la petición al servidor. El servidor accede a la base de datos, y finalmente la base de datos accede al disco. El disco devuelve a la base de datos los ficheros solicitados. Ésta, a su vez, devuelve al servidor las tablas con los datos. El servidor entrega al navegador código HTML, y finalmente el navegador muestra la página al usuario.

### Solución

Empezamos creando un nuevo proyecto “Tema 6” y, dentro de él, un diagrama de secuencia:



Hay 4 objetos involucrados en la secuencia: navegador, servidor, base de datos y disco:



Añadimos los mensajes. Asumiremos que todos los mensajes son síncronos y que, por tanto, la ejecución de cada objeto se detiene hasta que se recibe respuesta del objeto llamado. Los mensajes de respuesta se añaden automáticamente:

