**METODOLOGÍA ORIENTADA A OBJETOS (MOO)**

**DIAGRAMA DE SECUENCIA**

**PRESENTADO POR:**

**JOHANNA M. PINZON RIVERA**

**JHON ALEXANDER DIAZ**

**JOSE DAVID ORTEGA NORIEGA**

**DIRIGIDO A:**

**CARLOS ALFONSO PINEDA**

**TECNOLÓGICA FITEC**

**TECNOLOGÍA EN SISTEMAS**

**BUCARAMANGA**

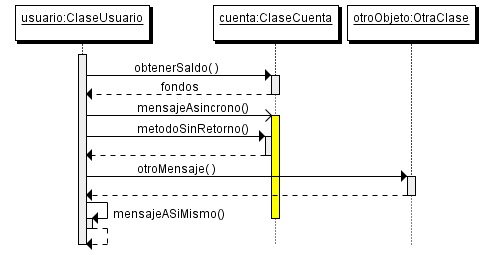
**2011-04-23**

**DIAGRAMA DE SECUENCIA**

El **diagrama de secuencia** es un tipo de diagrama usado para modelar interacción entre objetos en un sistema según [UML](http://es.wikipedia.org/wiki/UML). En inglés se pueden encontrar como "sequence diagram", "event-trace diagrams", "event scenarios" o "timing diagrams".

Un diagrama de secuencia es una forma de diagrama de interacción que muestra los objetos como líneas de vida a lo largo de la página y con sus interacciones en el tiempo representadas como mensajes dibujados como flechas desde la línea de vida origen hasta la línea de vida destino. Los diagramas de secuencia son buenos para mostrar qué objetos se comunican con qué otros objetos y qué mensajes disparan esas comunicaciones. Los diagramas de secuencia no están pensados para mostrar lógicas de procedimientos complejos.

Los Diagramas de Secuencias, también conocidos como de interacción o eventos, los cuales describen los diferentes casos de uso según la interacción o eventos enviados entre los objetos de la arquitectura del modelo de análisis.

[](http://es.wikipedia.org/wiki/Archivo:Sequencia.png)

En general, un diagrama de secuencia permite apreciar la fluidez de los eventos en la arquitectura y la correspondencia de la funcionalidad con la del caso de uso. Es muy importante que no existan interrupciones entre eventos y que exista una continuidad con los eventos externos del Sistema.

**ELEMENTOS**

**Los componentes de un diagrama de secuencia son:**

Un objeto (o actor) se representa como una línea vertical punteada con un rectángulo de encabezado y con rectángulos a través de la línea principal que denotan la ejecución de métodos (activación). El rectángulo de encabezado contiene el nombre del objeto y el de su clase.

**Activación**  
  
Muestra el periodo de tiempo en el cual el objeto se encuentra desarrollando alguna operación, bien sea por sí mismo o por medio de delegación a alguno de sus atributos. Se denota como un rectángulo delgado sobre la línea de vida del objeto.  
  
El envío de mensajes entre objetos se denota mediante una línea sólida dirigida, desde el objeto que emite el mensaje hacia el objeto que lo ejecuta. Representa la llamada de un método (operación) de un objeto en particular.  
  
También es posible visualizar llamadas a métodos desde el mismo objeto en estudio. 

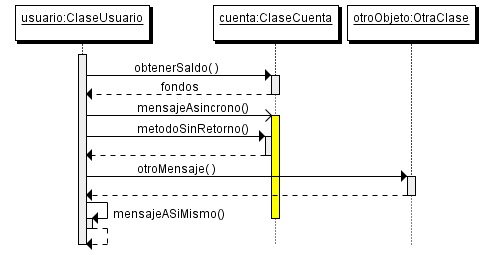
**REPRESENTACIÓN**

Un diagrama de secuencia se forma, colocando en primer lugar los objetos que participan en la interacción en la parte superior del diagrama, a lo largo del eje X, y normalmente en la parte izquierda se ubica el objeto que indica la interacción, y los objetos subordinados a la derecha.

Después se colocan los mensajes enviados y recibidos por estos, a lo largo del eje y, en orden de sucesión del tiempo de arriba hacia abajo.

**IMPORTANTE:**Si un objeto cambia el valor de sus atributos, su estado o sus roles, se puede colocar una copia del icono del objeto sobre su línea de vida en el punto en el que ocurre el cambio, mostrando la modificación.

Un **diagrama de secuencia** muestra los objetos que intervienen en el escenario con líneas discontinuas verticales, y los mensajes pasados entre los objetos como flechas horizontales.



**USO DE LOS DIAGRAMAS**

Se utiliza un diagrama para cada llamada a representar. Es imposible representar en un solo diagrama la secuencia de todas las llamadas posibles del sistema, es por ello que se escoge un punto de partida. El diagrama se compone con los objetos que forman parte de la secuencia, estos se sitúan en la parte superior de la pantalla, normalmente a la izquierda se sitúa el que inicia la acción. De estos objetos sale una línea que indica su vida en el sistema. Esta línea simple se convierte en una línea gruesa cuando representa que el objeto tiene el foco del sistema, es decir cuando él esta activo.

El objeto puede existir sólo durante la ejecución de la interacción, se puede crear o puede ser destruido durante la ejecución de la interacción.  
En este tipo de diagramas también intervienen los mensajes, que son la forma en que se comunican los objetos: el objeto origen solicita (llama a) una operación del objeto destino. El diagrama de secuencia puede ser obtenido de dos partes, desde el Diagrama Estático de Clases o desde el de Casos de Usos.   
  
Documentan el diseño no desde el punto de vista de los casos de uso. Observando qué mensajes se envían a los objetos, componentes o casos de uso y viendo a grosso modo cuanto tiempo consume el método invocado, los diagramas de secuencia nos ayudan a comprender los cuellos de botella potenciales, para así poder eliminarlos. A la hora de documentar un diagrama de secuencia resulta importante mantener los enlaces de los mensajes a los métodos apropiados del diagrama de clases.

## Utilidad

Un **diagrama de secuencia** muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada caso de uso. Mientras que el [diagrama de casos de uso](http://es.wikipedia.org/wiki/Diagrama_de_casos_de_uso) permite el modelado de una vista *business* del escenario, el diagrama de secuencia contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el escenario, y mensajes intercambiados entre los objetos.

Típicamente se examina la descripción de un [caso de uso](http://es.wikipedia.org/wiki/Caso_de_uso) para determinar qué objetos son necesarios para la implementación del escenario. Si se dispone de la descripción de cada [caso de uso](http://es.wikipedia.org/wiki/Caso_de_uso) como una secuencia de varios pasos, entonces se puede "caminar sobre" esos pasos para descubrir qué objetos son necesarios para que se puedan seguir los pasos. Un diagrama de secuencia muestra los objetos que intervienen en el escenario con líneas discontinuas verticales, y los mensajes pasados entre los objetos como flechas horizontales.

|  |
| --- |
| **Línea de Vida**  Una línea de vida representa un participante individual en un diagrama de secuencia. Una línea de vida usualmente tiene un rectángulo que contiene el nombre del objeto. Si el nombre es self entonces eso indica que la línea de vida representa el clasificador que posee el diagrama de secuencia.  **http://www.sparxsystems.com/images/screenshots/uml2_tutorial/seq01.GIF**  Algunas veces un diagrama de secuencia tendrá una línea de vida con un símbolo del elemento actor en la parte superior. Este usualmente sería el caso si un diagrama de secuencia es contenido por un caso de uso. Los elementos entidad, control y límite de los diagramas de robustez también pueden contener líneas de vida.  **http://www.sparxsystems.com/images/screenshots/uml2_tutorial/seq02.GIF**  **Mensajes**  Los mensajes se muestran como flechas. Los mensajes pueden ser completos, perdidos o encontrados; síncronos o asíncronos: llamadas o señales. En el siguiente diagrama, el primer mensaje es un mensaje síncrono (denotado por una punta de flecha oscura), completo con un mensaje de retorno implícito; el segundo mensaje es asíncrono (denotado por una punta de flecha en línea) y el tercero es un mensaje de retorno asíncrono (denotado por una línea punteada).  **http://www.sparxsystems.com/images/screenshots/uml2_tutorial/seq03.GIF**  **Ocurrencia de ejecución**  Un rectángulo fino a lo largo de la línea de vida denota la ocurrencia de ejecución o activación de un foco de control. En el diagrama anterior hay tres ocurrencias de ejecución. El primero es el objeto origen que envía dos mensajes y recibe dos respuestas, el segundo es el objeto destino que recibe un mensaje asíncrono y retorna una respuesta, y el tercero es el objeto destino que recibe un mensaje asíncrono y retorna una respuesta. Tipos de mensajes Existen dos tipos de mensajes: sincrónicos y asincrónicos. Los mensajes sincrónicos se corresponden con llamadas a métodos del objeto que recibe el mensaje. El objeto que envía el mensaje queda bloqueado hasta que termina la llamada. Este tipo de mensajes se representan con flechas con la cabeza llena. Los mensajes asincrónicos terminan inmediatamente, y crean un nuevo hilo de ejecución dentro de la secuencia. Se representan con flechas con la cabeza abierta.  **Los mensajes síncronos:** se corresponden con llamadas a métodos del objeto que recibe el mensaje. El objeto que envía el mensaje queda bloqueado hasta que termina la llamada. Este tipo de mensajes se representan con flechas con la cabeza llena.   **Los mensajes asíncronos:** terminan inmediatamente, y crean un nuevo hilo de ejecución dentro de la secuencia. Se representan con flechas.  seg7.JPG  También se representa la respuesta a un mensaje con una flecha discontinua. Pueden ser usados en dos formas  * De instancia: describe un escenario especifico (un escenario es una instancia de la ejecución de un caso de uso). * Genérico: describe la interacción para un caso de uso; Utiliza ramificaciones ("Branches"), condiciones y bucles.  Estructura Los mensajes se dibujan cronológicamente desde la parte superior del diagrama a la parte inferior; la distribución horizontal de los objetos es arbitraria. Durante el análisis inicial, el modelador típicamente coloca el nombre 'business' de un mensaje en la línea del mensaje. Más tarde, durante el diseño, el nombre 'business' es reemplazado con el nombre del método que está siendo llamado por un objeto en el otro. El método llamado, o invocado, pertenece a la definición de la clase instanciada por el objeto en la recepción final del mensaje.  **Operadores de interacción:**   1. 1. Alt: divide la interacción mediante una condición (modela estructuras if…then…else) 2. 2. Opt: alternativa simple, se hace solo si se cumple la condición (modela estructuras switch) 3. 3. Par: indica que cada fragmento trabaja en paralelo, modela procesos concurrentes 4. 4. Loop: indica varias veces la operación según indique la condición (actúa como un bucle) 5. 5. Critical: indica una secuencia que no puede ser interrumpida por otro proceso 6. 6. Sd : representa un diagrama de secuencia completo 7. 7. Break: modela una secuencia alternativa de eventos que se procesa en lugar de todo del resto del diagrama 8. 8. Weak (denotado “seq”) incluye un número de secuencias para las cuales todos los mensajes se deben procesar en un segmento anterior, antes de que el siguiente segmento pueda comenzar, pero que no impone ningún secuenciado en los mensajes que no comparten una línea de vida. 9. 9. Strict : incluye una serie de mensajes que se deben procesar en el orden proporcionado. 10. 10. Negative (denotado “neg”): incluye una serie de mensajes inválidos. 11. 11. Ignore: declara un mensaje o mensajes que no son de ningún interés si este aparece en el contexto actual. 12. 12. Consider: es el opuesto del fragmento Ignore cualquier mensaje que no se incluya en el fragmento Consider se debería ignorar 13. 13. Assertion (denotado “assert”) : designa que cualquier secuencia que no se muestra como un operando de la aserción es inválida.   **Mensaje Self**  Un mensaje self puede representar una llamada recursiva de una operación, o un método llamando a otro método perteneciente al mismo objeto. Este se muestra como cuando crea un foco de control anidado en la ocurrencia de ejecución de la línea de vida.  http://www.sparxsystems.com/images/screenshots/uml2_tutorial/seq04.GIF  **Mensajes perdidos y encontrados**  Los mensajes perdidos son aquellos que han sido enviados pero que no han llegado al destino esperado, o que han llegado a un destino que no se muestra en el diagrama actual. Los mensajes encontrados son aquellos que llegan de un remitente no conocido, o de un remitente no conocido en el diagrama actual. Ellos se denotan yendo o llegando desde un elemento de punto final.  http://www.sparxsystems.com/images/screenshots/uml2_tutorial/seq10.GIF  **Inicio y final de línea de vida**  Una línea de vida se puede crear o destruir durante la escala de tiempo representada por un diagrama de secuencia. En el último caso, la línea de vida se termina por un símbolo de detención, representado como una cruz. En el primer caso, el símbolo al inicio de la línea de vida se muestra en un nivel más bajo de la página que el símbolo del objeto que causó la creación. El siguiente diagrama muestra un objeto que fue creado y destruido.  http://www.sparxsystems.com/images/screenshots/uml2_tutorial/seq05.GIF  **Restricciones de tiempo y duración**  En forma predeterminada, un mensaje se muestra como una línea horizontal. Ya que la línea de vida representa el pasaje de tiempo hacia abajo, cuando se modela un sistema en tiempo real, o incluso un proceso de negocios en tiempo límite, puede ser importante considerar el tiempo que toma realizar las acciones. Al configurar una restricción de duración para un mensaje, el mensaje se mostrará como una línea inclinada.  http://www.sparxsystems.com/images/screenshots/uml2_tutorial/seq06.GIF  **Fragmentos combinados**  Se estableció anteriormente que no se espera que los diagramas de secuencia muestren lógicas de procedimientos complejos. Siendo este el caso, hay un número de mecanismos que permiten agregar un grado de lógicas de procedimientos a los diagramas y que a la vez vienen bajo el encabezado de fragmentos combinados. Un fragmento combinado es una o más secuencias de procesos incluidas en un marco y ejecutadas bajo circunstancias nombradas específicas. Los fragmentos disponibles son:   * El fragmento Alternative (denotedo “alt”) modela estructuras if…then…else. * El fragmento Option (denotado “opt”) modela estructuras switch. * El fragmento Break modela una secuencia alternativa de eventos que se procesa en lugar de todo del resto del diagrama. * El fragmento Parallel (denotado “par”) modela procesos concurrentes. * El fragmento de secuenciado Weak (denotado “seq”) incluye un número de secuencias para las cuales todos los mensajes se deben procesar en un segmento anterior, antes de que el siguiente segmento pueda comenzar, pero que no impone ningún secuenciado en los mensajes que no comparten una línea de vida. * El fragmento de secuenciado Strict (denotado “strict”) incluye una serie de mensajes que se deben procesar en el orden proporcionado. * El fragmento Negative (denotado “neg”) incluye una serie de mensajes inválidos. * El fragmento Critical incluye una sección crítica. * El fragmento Ignore declara un mensaje o mensajes que no son de ningún interés si este aparece en el contexto actual. * El fragmento Considera es el opuesto del fragmento Ignore: cualquier mensaje que no se incluya en el fragmento Consider se debería ignorar. * El fragmento Assertion (denotado “assert”) designa que cualquier secuencia que no se muestra como un operando de la aserción es inválida. * El fragmento Loop incluye una serie de mensajes que están repetidos.   El siguiente diagrama muestra un fragmento loop.  http://www.sparxsystems.com/images/screenshots/uml2_tutorial/seq07.GIF  También hay una ocurrencia de interacción, que es similar a un fragmento combinado. Una ocurrencia de interacción es una referencia a otro diagrama que tiene la palabra “ref” en la esquina izquierda arriba del marco, y tiene el nombre del diagrama referenciado que se muestra en el medio del marco  **Puerta**  Una puerta es un punto de conexión para conectar un mensaje dentro de un fragmento con un mensaje fuera del fragmento. EA muestra una puerta como un cuadro pequeño en un marco del fragmento.  http://www.sparxsystems.com/images/screenshots/uml2_tutorial/seq08.GIF  **Descomposición en parte**  Un objeto puede tener más de una línea de vida que viene de ésta. Esto permite mensajes de entre e intra objetos para que se muestren en el mismo diagrama.  http://www.sparxsystems.com/images/screenshots/uml2_tutorial/seq11.GIF  **Continuaciones / Invariantes de Estado**  Una invariante de estado es una restricción ubicada en una línea de vida que debe ser verdadera en el tiempo de ejecución. Esta se muestra como un rectángulo con los extremos en semicírculos.  http://www.sparxsystems.com/images/screenshots/uml2_tutorial/seq09.GIF  Una continuación tiene la misma notación que una invariante de estado pero se usa en fragmentos combinados y puede extenderse a través de más de una línea de vida. |

**Cómo hacer un Diagrama de Secuencia del Sistema**

1. Dibuje una línea vertical representando el sistema como una caja negra.

2. Identifique cada actor que opera directamente sobre el sistema y dibuje una línea vertical para cada actor.

3. De la descripción del caso de uso, curso típico de eventos, identifique los eventos (externos) que cada actor genera sobre el sistema. Ilústrelos en el diagrama con una línea horizontal desde el actor al sistema.

4. Opcionalmente, incluya el texto de la descripción del caso de uso a la izquierda del diagrama

**Registro de Operaciones del Sistema:**

El UML incluye una notación para registrar operaciones de un

tipo como la siguiente:

|  |
| --- |
| Tipo X  Operaciones del tipo |
| Operación 1() |
| Operación 2() |

Con esta notación, las operaciones del sistema pueden agruparse como operaciones de un tipo llamado sistema así:

|  |
| --- |
| Sistema |
| Entrar Artículo (código, cantidad)  terminar Venta()  pagar (Valor) |

(Los parámetros pueden, opcionalmente, ignorarse)

Nombre de Eventos y Operaciones.

Los eventos del sistema (y sus correspondientes operaciones) deben ser expresados al nivel de la intención y no en términos del medio de entrada físico del nivel de los elementos de la interfaz.

El nombre debe empezar con un verbo así:

“terminar Venta” es mejor que “entrar tecla presionada” Para las operaciones, expresarlas en términos del objetivo último. Con respecto a la operación que captura el pago:

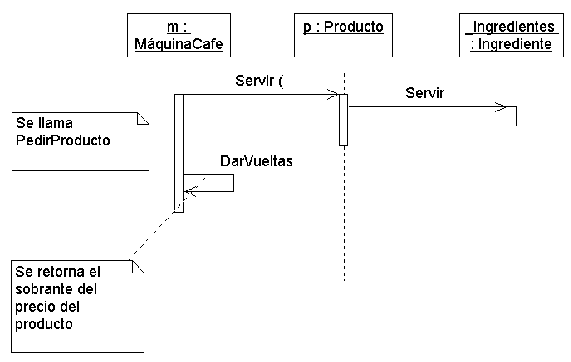
Entrar Cantidad Entregada (valor)...............pobre

Entrar Pago (valor)................................mejor

Pagar (valor)......................................aún mejor

**EJEMPLO PEDIDO MAQUINA DE CAFE**

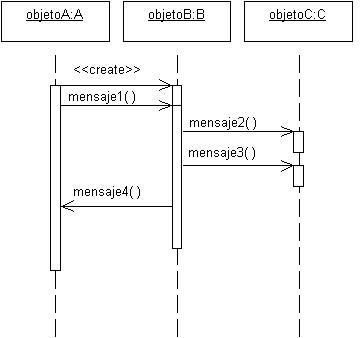
|  |
| --- |
|  |
|  |



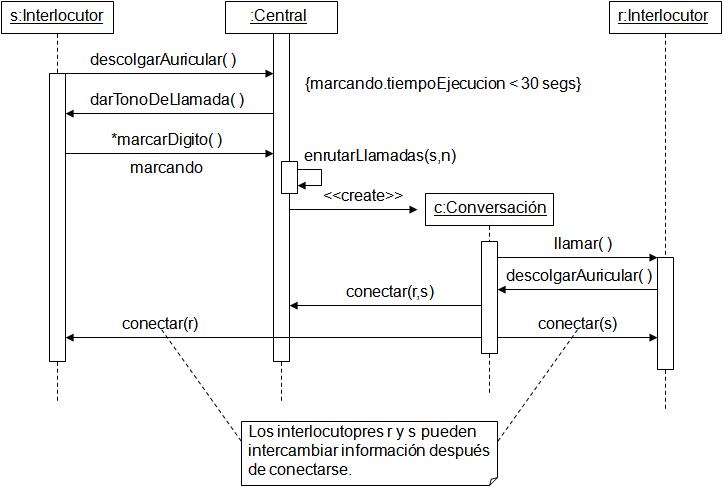
En la figura se muestra un ejemplo de diagrama de secuencia, que da detalle al caso de uso *Pedir Producto* del ejemplo de la máquina de café. Los conceptos más importantes relacionados con los diagramas de secuencia son:

* **Línea de vida de un objeto** (*lifeline*): La línea de vida de un objeto representa la vida del objeto durante la interacción. En un diagrama de secuencia un objeto se representa como una línea vertical punteada con un rectángulo de encabezado y con rectángulos a través de la línea principal que denotan la ejecución de métodos (activación). El rectángulo de encabezado contiene el nombre del objeto y el de su clase, en un formato *nombre Objeto: nombre Clase*. Por ejemplo, el objeto *m*, instancia de la clase *Maquina Café* envía dos mensajes seguidos para dar respuesta a la operación *Pedir Producto*: *Servir* al objeto *p* de la clase *Producto* y *Dar Vueltas* a sí mismo (*self-delegation*).
* **Activación**: Muestra el período de tiempo en el cual el objeto se encuentra desarrollando alguna operación, bien sea por sí mismo o por medio de delegación a alguno de sus atributos. Se denota como un rectángulo delgado sobre la línea de vida del objeto. En el ejemplo anterior el objeto *ingredientes* se encuentra activo mientras ejecuta el método correspondiente al mensaje *Servir*, el objeto *p* se encuentra activo mientras se ejecuta su método *Servir*, que ejecuta *ingredientes. Servir*, y el objeto *m* se encuentra activo mientras se ejecuta *p. Servir* y *Dar Vueltas*.
* **Mensaje**: El envío de mensajes entre objetos se denota mediante una línea sólida dirigida, desde el objeto que emite el mensaje hacia el objeto que lo ejecuta. En el ejemplo anterior el objeto *m* envía el mensaje *Servir* al objeto *p* y un poco más adelante en el tiempo el objeto *m* se envía a sí mismo el mensaje *Dar Vueltas*.
* **Tiempos de transición**: En un entorno de objetos concurrentes o de demoras en la recepción de mensajes, es útil agregar nombres a los tiempos de salida y llegada de mensajes.

Destaca el orden temporal de los mensajes.



Conceptos:  
  
Ambos diagramas (secuencia y colaboración) son semántica- mente equivalente. Se puede pasar de uno a otro sin pérdida de información.  
En los diagramas de secuencia, la línea de vida de un objeto es la línea discontinua vertical, que representa la existencia de un objeto a lo largo de un periodo de tiempo. El foco de control es un rectángulo delgado que representa el periodo de tiempo durante el cual un objeto ejecuta una acción.  
  
Ejemplo:  
  
Se quiere modelar una llamada a través de una central telefónica.  
  
Para esto se tienen cuatro objetos involucrados: dos interlocutores (s y r), una central y una conversación. La secuencia empieza cuando un interlocutor envía un mensaje a la central al descolgar el auricular. La central da el tono de llamada, y el interlocutor marca el número al que desea llamar. El tiempo de marcado debe ser menor que 30 segundos.



**CONCLUSIONES**

* El Diagrama de Secuencia es una forma de diagrama de interacción que nos muestra los objetos como líneas de vida a lo largo de la página y con sus interacciones en el tiempo simbolizadas como mensajes dibujados como flechas desde la línea de vida origen hasta la línea de vida Destino.
* Son uno de los diagramas más efectivos para modelar interacción entre objetos en un sistema. Este diagrama (también llamado diagrama de interacción) muestra las interacciones entre un conjunto de objetos (clases, actores).
* Además un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada método de la clase. Mientras que el [diagrama de casos de uso](http://es.wikipedia.org/wiki/Diagrama_de_casos_de_uso) permite el modelado de una vista *business* del escenario.
* El contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el escenario, y mensajes pasados entre los objetos.
* Cada objeto en el diagrama es representado con una línea vertical, correspondiente al eje del tiempo, donde el tiempo avanza hacia abajo. El diagrama de secuencia muestra los eventos que ocurren en el tiempo, los cuales son enviados de un objeto a otro.

**BIBLIOGRAFIA**

* <http://www.sparxsystems.com.ar/resources/tutorial/uml2_sequencediagram.html>
* <http://es.wikipedia.org/wiki/Diagrama_de_secuencia>
* [*Ingeniería de Software Orientada a Objetos con UML, Java e Internet*](http://go.galegroup.com/ps/aboutEbook.do?pubDate=120050000&actionString=DO_DISPLAY_ABOUT_PAGE&inPS=true&prodId=GVRL&userGroupName=fitec&searchType=BasicSearchForm&docId=GALE%7C2VGZ). Alfredo Weitzenfeld. Mexico City: Cengage Learning, 2005. p275-310.