



APRENDERAPROGRAMAR.COM

¿QUÉ ES Y PARA QUÉ SIRVE  
UML? VERSIONES DEL  
LENGUAJE UNIFICADO DE  
MODELADO. TIPOS DE  
DIAGRAMAS. INGENIERÍA  
DEL SOFTWARE  
(DV00205D)

Sección: Divulgación

Categoría: Lenguajes y entornos

Fecha revisión: 2029

**Resumen:** UML ó Lenguaje Unificado de Modelado es un estándar para la representación de procesos, abstracciones o esquemas de software (programas informáticos).

Autor: César Krall

## UML

UML son las siglas de “Unified Modeling Language” o “Lenguaje Unificado de Modelado”. Se trata de un estándar que se ha adoptado a nivel internacional por numerosos organismos y empresas para crear esquemas, diagramas y documentación relativa a los desarrollos de software (programas informáticos).



### ¿QUÉ ES Y PARA QUÉ SIRVE UML?

El término “lenguaje” ha generado bastante confusión respecto a lo que es UML. En realidad el término lenguaje quizás no es el más apropiado, ya que no es un lenguaje propiamente dicho, sino una serie de normas y estándares gráficos respecto a cómo se deben representar los esquemas relativos al software. Mucha gente piensa por confusión que UML es un lenguaje de programación y esta idea es errónea: UML no es un lenguaje de programación. Como decimos, UML son una serie de normas y estándares que dicen cómo se debe representar algo.

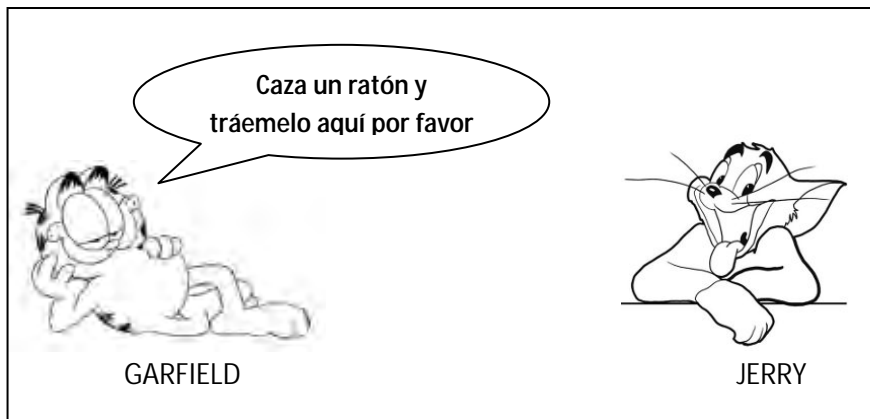
UML es una herramienta propia de personas que tienen conocimientos relativamente avanzados de programación y es frecuentemente usada por analistas funcionales (aquellos que definen qué debe hacer un programa sin entrar a escribir el código) y analistas-programadores (aquellos que dado un problema, lo estudian y escriben el código informático para resolverlo en un lenguaje como Java, C#, Python o cualquier otro). Por tanto si estás dando tus primeros pasos en programación, te recomendaríamos que te olvides de UML hasta que tengas unos conocimientos mínimos como uso de condicionales, bucles, y conocimiento de la programación orientada a objetos. Esto es solo una recomendación, en realidad prácticamente cualquier persona puede usar UML, incluso podría usarse para realizar esquemas o documentación de procesos que no tengan que ver con la informática.

Hemos dicho que UML es un estándar. Vamos a aclarar primero qué es un estándar. Supongamos que vamos a definir un estándar llamado “LMAPR” o lenguaje de modelado de aprenderaprogramar.com. Ahora definimos dentro de nuestro estándar estas normas:

Un animal debe representarse con su nombre escrito enteramente en minúsculas enmarcado dentro de un rectángulo doble. Encima del nombre debe etiquetarse el tipo de animal así: <<Tipo de Animal>>. Por ejemplo, <<Gato>>.

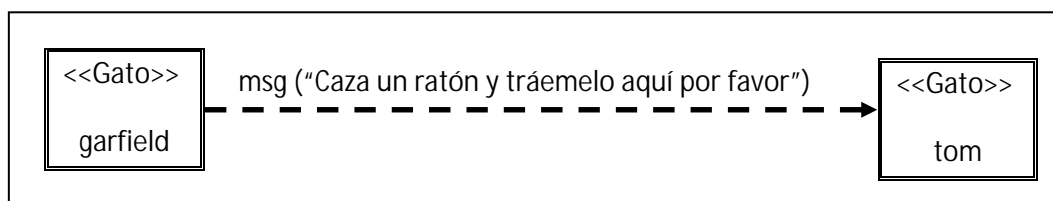
Si un animal envía un mensaje a otro animal deben conectarse los dos animales con una línea punteada terminada en flecha encima de la cual debe figurar el texto msg(“Contenido del mensaje”).

Ahora supongamos que tenemos dos gatos, uno de los cuales le dice al otro "Caza un ratón y tráemelo aquí por favor". Veamos formas de representar esto:

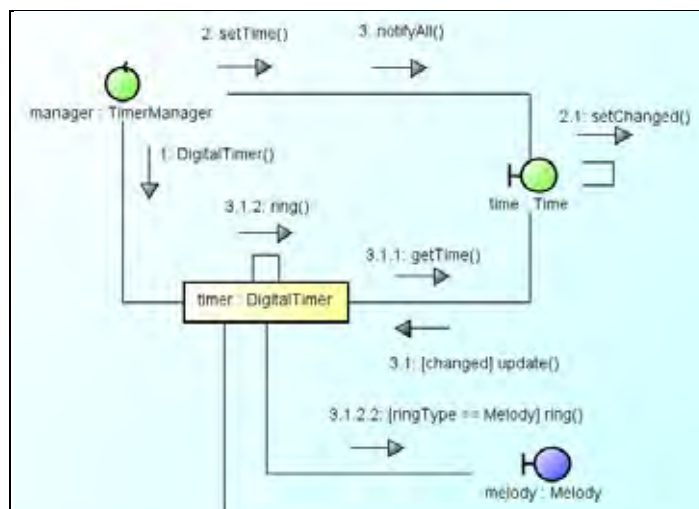


Esta es una forma de representación. Sin embargo, no se adapta al estándar que hemos definido por varios motivos: no indica <<Gato>> encima de los nombres de los animales, no escribe los nombres en minúsculas, no representa los animales con un rectángulo, etc.

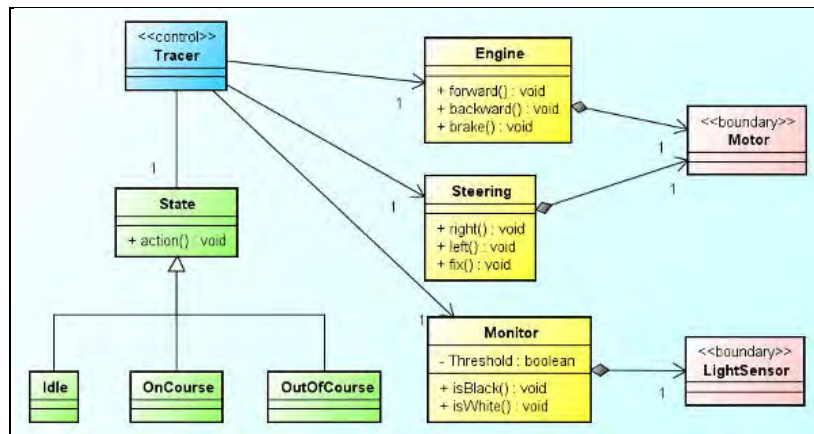
Veamos la representación que sí se adaptaría al estándar definido:



Con este ejemplo sencillo hemos tratado de hacer explícito qué es y para qué sirve UML: un conjunto de normas que nos dicen cómo hay que representar esquemas de software. En el caso del software orientado a objetos, en vez de gatos tendremos clases u objetos instanciados, y dispondremos de numerosos tipos de esquemas y diagramas para representar distintas cosas. Un esquema que cumple las normas UML podría tener este aspecto:



O también este otro:



¿Por qué si ambos esquemas cumplen con UML tienen un aspecto tan distinto? Porque UML define normas para construir muchos tipos de esquemas, no esquemas de un solo tipo.

¿Quién usa UML? UML lo suelen usar las empresas o medianos o grandes equipos de desarrollo software con el objetivo de planificar y documentar cómo se construyen los programas informáticos complejos. Los usuarios individuales o pequeños equipos de desarrollo de 2 ó 3 personas no suelen usar herramientas UML. UML es un término que se relaciona mucho con "Ingeniería del software". Al igual que un proyecto de edificio requiere la participación de un arquitecto y unos planos, un proyecto software requiere la participación de ingenieros informáticos y una planificación y documentación.

### ¿CUÁLES SON LAS VERSIONES DE UML?

Los antecedentes de UML se sitúan en la década de los 90 con distintos estándares para modelado de software, no obstante podemos hablar de dos grandes versiones:

UML 1.X (comprende UML 1.1, 1.2, 1.3, 1.4, 1.5): desde finales de los 90 se empezó a trabajar con el estándar UML. En los años sucesivos fueron apareciendo nuevas versiones que introducían mejoras o ampliaban a las anteriores.

UML 2.X (comprende UML 2.1 hasta UML 2.5, 2.6, etc.): en torno a 2005 se difundió una nueva versión de UML a la que podemos denominar UML 2.X. Comprende varias revisiones.

UML 3.X: evolución que se espera para UML 2.X.

Hay que tener en cuenta que UML es un conjunto muy amplio de normas. Prácticamente nadie las conoce todas. Según la empresa o universidad, institución o centro de trabajo se usan determinados programas para crear diagramas y se conocen ciertas partes de UML, pero no el conjunto de UML.

¿Qué versión usar? Para generar diagramas UML se usan programas informáticos. Usa un programa actualizado pero no te preocupes en exceso por qué versión de UML usar, lo importante es que en tu grupo de trabajo o personas a las que se les vaya a enviar documentación sobre un proyecto software sepan interpretar lo que se les envía. A nivel profesional no se le presta demasiada atención a que se cumpla estrictamente con las normas de una determinada versión de UML, sino a que los esquemas estén bien contruidos y razonados.

## TIPOS DE DIAGRAMAS EN UML

Usando UML se pueden construir numerosos tipos de diagramas. Vamos a citar algunos:

**Diagramas de casos de uso:** representan a los actores y casos de uso (procesos principales) que intervienen en un desarrollo de software.

**Diagramas de clases:** para UML una clase es una entidad, no una clase software. Un diagrama de clases UML puede ser un diagrama del dominio o representación de conceptos que intervienen en un problema, o también un diagrama de clases software. El sentido de un diagrama UML se lo da la persona que lo construye.

**Diagramas de secuencia:** suelen usarse para representar objetos software y el intercambio de mensajes entre ellos, representando la aparición de nuevos objetos de izquierda a derecha.

**Diagramas de colaboración:** suelen usarse para representar objetos o clases y la forma en que se transmiten mensajes y colaboran entre ellos para cumplir un objetivo.

**Diagramas de estados:** suelen usarse para representar cómo evoluciona un sistema (cómo va cambiando de estado) a medida que se producen determinados eventos.

**Otros diagramas:** diagramas de actividad, diagramas de paquetes, diagramas de arquitectura software, etc.

## HERRAMIENTAS O PROGRAMAS PARA TRABAJAR CON UML

Hay muchísimos programas que permiten trabajar con UML, aunque aprender a usarlos requiere tiempo.

Astah community: herramienta sencilla, adecuada para aprender. Se puede descargar una versión gratuita en <http://astah.net/editions/community>. Astah (antes conocido como Jude) también tiene una versión profesional.

Rational Rose: conjunto de herramientas IBM usado por muchas empresas.

Otros: Erwin, Oracle Designer, EasyCASE, Power Designer, etc. son herramientas que incorporan muchas utilidades, entre ellas UML.

## CRÍTICAS A UML

UML recibe numerosas críticas por parte de los miembros de la comunidad de desarrolladores software, entre ellas el ser demasiado extenso, carecer de significados precisos para los elementos representados, dificultad para representar algunos tipos de sistemas software o elementos, etc.

A pesar de ello y de no ser "perfecto", es un estándar de amplio uso hoy día y una herramienta fundamental en desarrollos software de gran envergadura.