

# *El Lenguaje Unificado de Modelado (UML)*

*Enrique Hernández Orallo (ehernandez@disca.upv.es)*

Cualquier rama de ingeniería o arquitectura ha encontrado útil desde hace mucho tiempo la representación de los diseños de forma gráfica. Desde los inicios de la informática se han estado utilizando distintas formas de representar los diseños de una forma más bien personal o con algún modelo gráfico. La falta de estandarización en la manera de representar gráficamente un modelo impedía que los diseños gráficos realizados se pudieran compartir fácilmente entre distintos diseñadores.



*Figura 1: Logo de UML*

Se necesitaba por tanto un lenguaje no sólo para comunicar las ideas a otros desarrolladores sino también para servir de apoyo en los procesos de análisis de un problema. Con este objetivo se creó el Lenguaje Unificado de Modelado (UML: *Unified Modeling Language*). UML se ha convertido en ese estándar tan ansiado para representar y modelar la información con la que se trabaja en las fases de análisis y, especialmente, de diseño.

El lenguaje UML tiene una notación gráfica muy expresiva que permite representar en mayor o menor medida todas las fases de un proyecto informático: desde el análisis con los casos de uso, el diseño con los diagramas de clases, objetos, etc., hasta la implementación y configuración con los diagramas de despliegue.

---

## HISTORIA DE UML

El lenguaje UML comenzó a gestarse en octubre de 1994 [1], cuando Rumbaugh se unió a la compañía *Rational* fundada por Booch (dos reputados investigadores en el área de metodología del software). El objetivo de ambos era unificar dos métodos que habían desarrollado: el método Booch y el OMT (*Object Modeling Tool*). El primer borrador apareció en octubre de 1995. En esa misma época otro reputado investigador, Jacobson, se unió a *Rational* y se incluyeron ideas suyas. Estas tres personas son conocidas como los “tres amigos”. Además, este lenguaje se abrió a la colaboración de otras empresas para que aportaran sus ideas. Todas estas colaboraciones condujeron a la definición de la primera versión de UML.

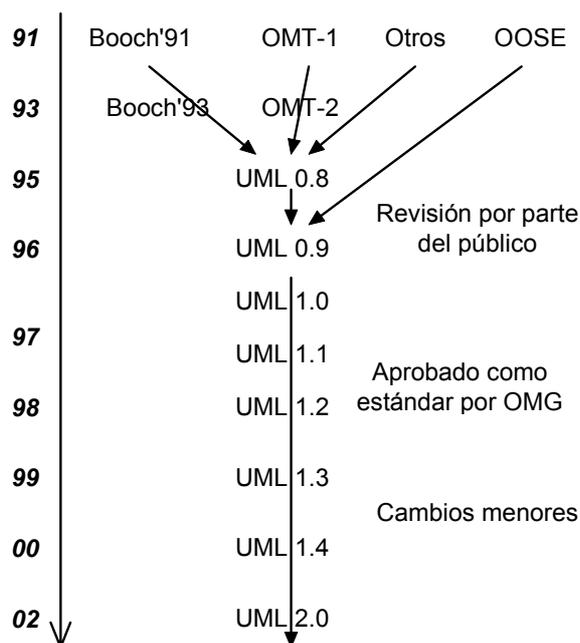


Figura 2: Evolución de UML

Esta primera versión se ofreció a un grupo de trabajo para convertirlo en 1997 en un estándar del OMG (*Object Management Group* <http://www.omg.org>). Este grupo, que gestiona estándares relacionados con la tecnología orientada a objetos (metodologías, bases de datos objetuales, CORBA, etc.), propuso una serie de modificaciones y una nueva versión de UML (la 1.1), que fue adoptada por el OMG como estándar en noviembre de 1997. Desde aquella versión han habido varias revisiones que gestiona la *OMG Revision Task Force*. La última versión aprobada es la 1.4. En estos momentos se está desarrollando una nueva versión en la que se incluirán cambios importantes (principalmente añadir nuevos diagramas) que conducirán a la versión 2.0 planificada para fines del 2002.

## MODELADO VISUAL

Tal como indica su nombre, UML es un lenguaje de modelado. Un modelo es una simplificación de la realidad. El objetivo del modelado de un sistema es capturar las partes esenciales del sistema. Para facilitar este modelado, se realiza una abstracción y se plasma en una notación gráfica. Esto se conoce como modelado visual.

El modelado visual permite manejar la complejidad de los sistemas a analizar o diseñar. De la misma

forma que para construir una choza no hace falta un modelo, cuando se intenta construir un sistema complejo como un rascacielos, es necesario abstraer la complejidad en modelos que el ser humano pueda entender.

UML sirve para el modelado completo de sistemas complejos, tanto en el diseño de los sistemas software como para la arquitectura hardware donde se ejecuten.

Otro objetivo de este modelado visual es que sea independiente del lenguaje de implementación, de tal forma que los diseños realizados usando UML se pueda implementar en cualquier lenguaje que soporte las posibilidades de UML (principalmente lenguajes orientados a objetos).

UML es además un método formal de modelado. Esto aporta las siguientes ventajas:

- Mayor rigor en la especificación.
- Permite realizar una verificación y validación del modelo realizado.
- Se pueden automatizar determinados procesos y permite generar código a partir de los modelos y a la inversa (a partir del código fuente generar los modelos). Esto permite que el modelo y el código estén actualizados, con lo que siempre se puede mantener la visión en el diseño, de más alto nivel, de la estructura de un proyecto.

## ¿QUÉ ES UML?

UML es ante todo un lenguaje. Un lenguaje proporciona un vocabulario y una reglas para permitir una comunicación. En este caso, este lenguaje se centra en la representación gráfica de un sistema.

Este lenguaje nos indica cómo crear y leer los modelos, pero no dice cómo crearlos. Esto último es el objetivo de las metodologías de desarrollo.

Las objetivos de UML son muchos, pero se pueden sintetizar sus funciones:

- Visualizar: UML permite expresar de una forma gráfica un sistema de forma que otro lo puede entender.
- Especificar: UML permite especificar cuáles son las características de un sistema antes de su construcción.

- Construir: A partir de los modelos especificados se pueden construir los sistemas diseñados.
- Documentar: Los propios elementos gráficos sirven como documentación del sistema desarrollado que pueden servir para su futura revisión.

Aunque UML está pensado para modelar sistemas complejos con gran cantidad de software, el lenguaje es lo suficientemente expresivo como para modelar sistemas que no son informáticos, como flujos de trabajo (*workflow*) en una empresa, diseño de la estructura de una organización y por supuesto, en el diseño de hardware.

Un modelo UML esta compuesto por tres clases de bloques de construcción:

- Elementos: Los elementos son abstracciones de cosas reales o ficticias (objetos, acciones, etc.)
- Relaciones: relacionan los elementos entre sí.
- Diagramas: Son colecciones de elementos con sus relaciones.

Veamos con mayor detalle los diagramas de UML.

---

## DIAGRAMAS UML

Un diagrama es la representación gráfica de un conjunto de elementos con sus relaciones. En concreto, un diagrama ofrece una vista del sistema a modelar. Para poder representar correctamente un sistema, UML ofrece una amplia variedad de diagramas para visualizar el sistema desde varias perspectivas. UML incluye los siguientes diagramas:

- Diagrama de casos de uso.
- Diagrama de clases.
- Diagrama de objetos.
- Diagrama de secuencia.

- Diagrama de colaboración.
- Diagrama de estados.
- Diagrama de actividades.
- Diagrama de componentes.
- Diagrama de despliegue.

Los diagramas más interesantes (y los más usados) son los de casos de uso, clases y secuencia, por lo que nos centraremos en éstos. Para ello, se utilizará ejemplos de un sistema de venta de entradas de cine por Internet.

El *diagrama de casos de usos* representa gráficamente los casos de uso que tiene un sistema. Se define un caso de uso como cada interacción supuesta con el sistema a desarrollar, donde se representan los requisitos funcionales. Es decir, se está diciendo lo que tiene que hacer un sistema y cómo. En la figura 3 se muestra un ejemplo de casos de uso, donde se muestran tres actores (los clientes, los taquilleros y los jefes de taquilla) y las operaciones que pueden realizar (sus roles).

El *diagrama de clases* muestra un conjunto de clases, interfaces y sus relaciones. Éste es el diagrama más común a la hora de describir el diseño de los sistemas orientados a objetos. En la figura 4 se muestran las clases globales, sus atributos y las relaciones de una posible solución al problema de la venta de entradas.

En el *diagrama de secuencia* se muestra la interacción de los objetos que componen un sistema de forma temporal. Siguiendo el ejemplo de venta de entradas, la figura 5 muestra la interacción de crear una nueva sala para un espectáculo.

El resto de diagramas muestran distintos aspectos del sistema a modelar. Para modelar el comportamiento dinámico del sistema están los de *interacción*, *colaboración*, *estados* y *actividades*. Los *diagramas de componentes* y *despliegue* están enfocados a la implementación del sistema.

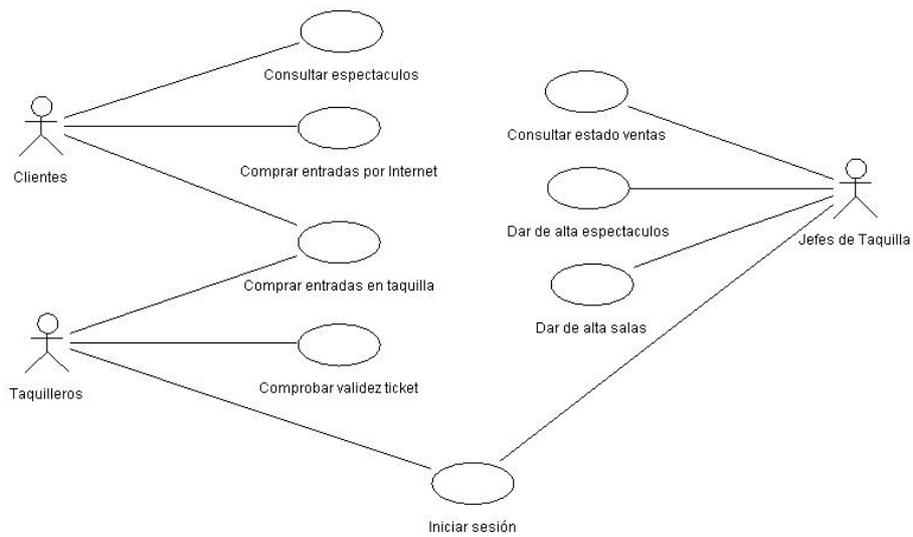


Figura 3: Diagrama de casos de uso

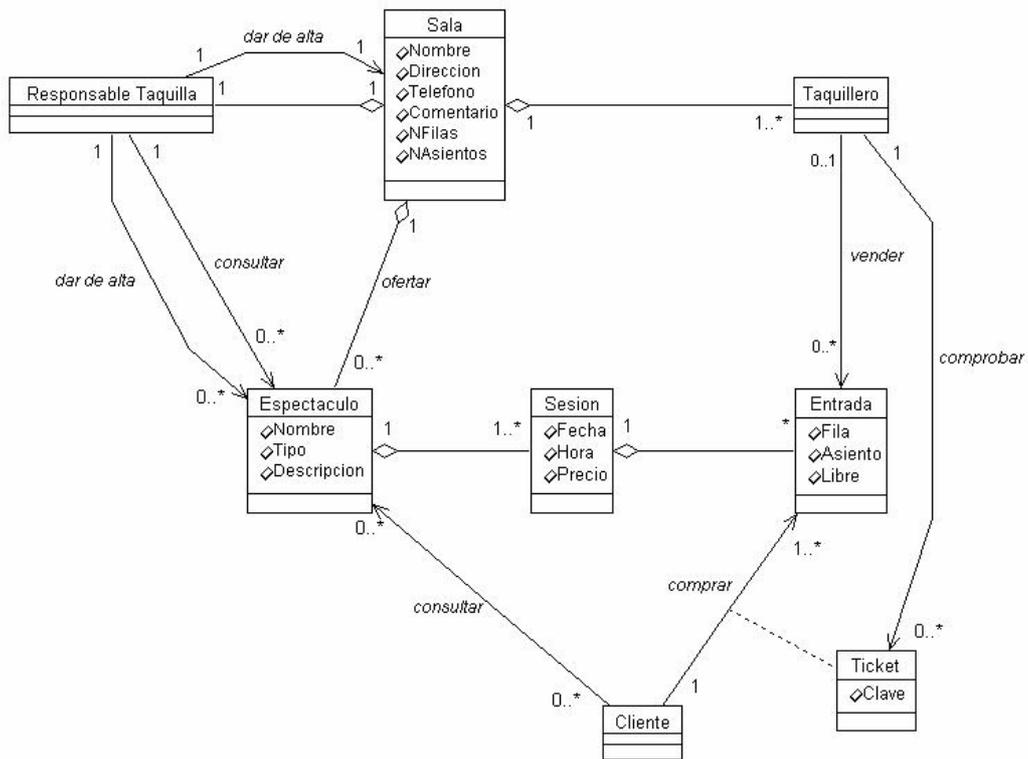


Figura 4: Diagrama de clases.

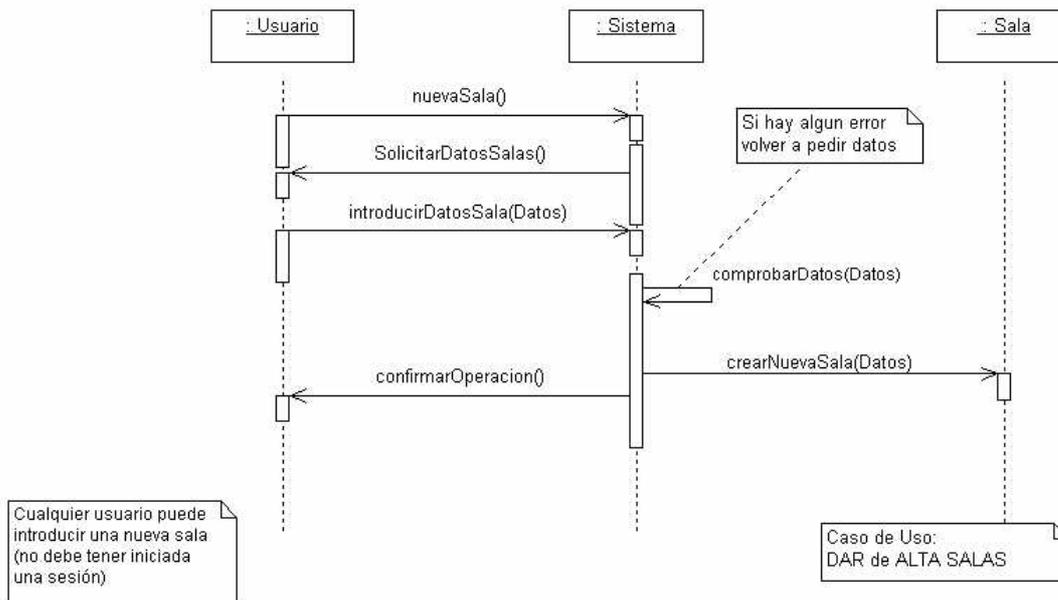


Figura 5: Diagrama de secuencia.

## PROCESO DE DESARROLLO

Aunque UML es bastante independiente del proceso de desarrollo que se siga, los mismos creadores de UML han propuesto su propia metodología de desarrollo, denominada el *Proceso Unificado de Desarrollo* [2].

El Proceso Unificado está basado en componentes, lo cual quiere decir que el sistema software en construcción está formado por componentes software interconectados a través de interfaces bien definidos. Además, el Proceso Unificado utiliza el UML para expresar gráficamente todos los esquemas de un sistema software. Pero, realmente, los aspectos que definen este Proceso Unificado son tres: es iterativo e incremental, dirigido por casos de uso y centrado en la arquitectura [3]:

- *Dirigido por casos de uso:* Basándose en los casos de uso, los desarrolladores crean una serie de modelos de diseño e implementación que los llevan a cabo. Además, estos modelos se validan para que sean conformes a los casos de uso. Finalmente, los casos de uso también sirven para realizar las pruebas sobre los compo-

nentes desarrollados.

- *Centrado en la arquitectura:* En la arquitectura de la construcción, antes de construir un edificio éste se contempla desde varios puntos de vista: estructura, conducciones eléctricas, fontanería, etc. Cada uno de estos aspectos está representado por un gráfico con su notación correspondiente. Siguiendo este ejemplo, el concepto de arquitectura software incluye los aspectos estáticos y dinámicos más significativos del sistema.
- *Iterativo e incremental:* Todo sistema informático complejo supone un gran esfuerzo que puede durar desde varios meses hasta años. Por lo tanto, lo más práctico es dividir un proyecto en varias fases. Actualmente se suele hablar de ciclos de vida en los que se realizan varios recorridos por todas las fases. Cada recorrido por las fases se denomina iteración en el proyecto en la que se realizan varios tipos de trabajo (denominados flujos). Además, cada iteración parte de la anterior incrementado o revisando la funcionalidad implementada. Se suele denominar proceso. Ver figura 6.

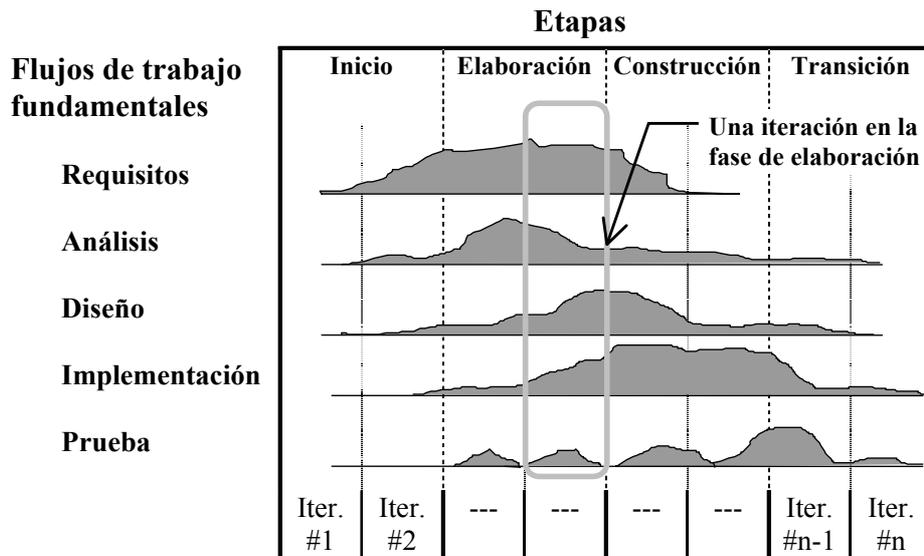


Figura 6: Proceso iterativo e incremental

Resumiendo, el Proceso Unificado es un modelo complejo con mucha terminología propia, pensado principalmente para el desarrollo de grandes proyectos. Es un proceso que puede adaptarse y extenderse en función de las necesidades de cada empresa.

## CONCLUSIONES

Es fácil predecir que UML será el lenguaje de modelado de software de uso universal. Las principales razones para ello son:

- En el desarrollo han participado investigadores de reconocido prestigio.
- Ha sido apoyado por prácticamente todas las empresas importantes de informática.
- Se ha aceptado como un estándar por la OMG.
- Prácticamente todas las herramientas CASE y de desarrollo la han adaptado como lenguaje de modelado.

En resumen, UML resuelve de forma bastante satisfactoria un viejo problema del desarrollo de software como es su modelado gráfico. Además, se ha llegado a una solución unificada basada en lo mejor que había hasta el momento, lo cual lo hace todavía más excepcional.

## REFERENCIAS

1. G. Booch, J. Rumbaugh y I. Jacobson, "El Lenguaje Unificado de Modelado", Addison Wesley, 1999
2. I. Jacobson, G. Booch, J. Rumbaugh, "El Proceso Unificado de Desarrollo", Addison Wesley, 2000
3. E. Hernández, J. Hernández, C. Lizandra, "C++ Estándar", ITP Paraninfo 2001.