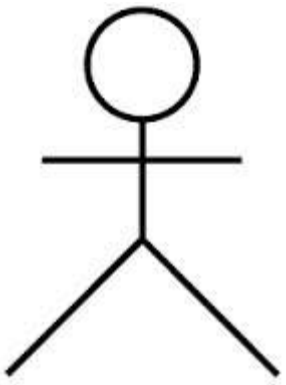
Introducción al UML

Las imágenes de pequeñas personas formadas por palillos representan la forma de comunicación más antigua registrada en la historia humana. Algo de este arte rupestre se remonta a épocas tan antiguas como hace 75,000 años. Lo que resulta bastante extraño es que nos encontramos al principio del moderno siglo XXI todavía estamos usando pequeñas figuras de línea para transmitir información. Eso es correcto; un pequeño hombre formado por palillos que llamamos Esaw es el carácter central en uno de los lenguajes más recientes desarrollado por los humanos.



Esaw

Figura 5. Esaw, a quien se menciona como actor en UML.

El lenguaje acerca del cual estoy hablando se llama Unified Modeling Language(Lenguaje unificado de modelado), o uml. El uml es un lenguaje tanto como Pascal, C# (C sharp), el alemán, el inglés y el latín; y el uml posiblemente es uno de los lenguajes más recientes inventados por la humanidad, alrededor de 1997.Como sucede con otros lenguajes, el uml fue inventado por necesidad. Es más, como con muchos lenguajes, en el uml se usan símbolos para transmitir significado. Sin embargo, a diferencia de los lenguajes orgánicos, como el inglés y el alemán, que evolucionan con el transcurso del tiempo a partir del uso común y la adaptación, el uml fue inventado por científicos, lo cual, por desgracia, es un problema. Los científicos son muy inteligentes pero con frecuencia no son muy buenos para explicar las cosas a aquellos menos científicos. Aquí es en donde intervengo.

En este capítulo, revisaremos el origen y la evolución del uml; también hablaremos acerca de cómo crear imágenes usando el uml, cuántas imágenes crear y qué tipos de ellas, qué deben transmitir esas imágenes y, lo más importante, cuándo suspender el dibujo de imágenes y empezar a escribir código.

Comprensión de los modelos

Un modelo es una colección de imágenes y texto que representa algo; para nuestros fines, software. (Los modelos no tienen que representar software, pero ahora reduciremos nuestro ámbito a los modelos de software.) Un modelo es para el software lo que un plano azul es para una casa. Los modelos son valiosos por muchas razones específicas; en gran parte, constan de imágenes e, incluso, las imágenes simples pueden transmitir más información que una gran cantidad de texto; por ejemplo, código. Esto resulta coherente con el viejo adagio un tanto modificado de que una imagen expresa un millar de líneas de código. Los modelos son valiosos porque es más fácil dibujar algunas imágenes sencillas que escribir código o incluso texto que describan lo mismo. Los modelos son valiosos porque es más barato, rápido y fácil cambiar modelos que cambiar código. La verdad simple es que barato, rápido, fácil y flexible es lo que usted quiere cuando está resolviendo problemas.

Desafortunadamente, si cada uno usa imágenes diferentes para dar a entender lo mismo, entonces las imágenes se agregan a la confusión, en lugar de mitigarla. Aquí es en donde entra el uml.

Comprensión del UML

El uml es una definición oficial de un lenguaje pictórico con símbolos y relaciones comunes que tienen un significado común. Si todos los participantes hablan uml, entonces las imágenes tienen el mismo significado para todos aquellos que las observen. Por lo tanto, aprender um les esencial para ser capaz de usar imágenes para experimentar barata, flexible y rápidamente con las soluciones. Es importante reiterar aquí que es más rápido, más barato y más fácil resolver problemas con imágenes que con código. La única barrera para obtener beneficios del modelado es aprender el lenguaje del mismo. El uml es un lenguaje precisamente como lo son el inglés o el afrikaans. El uml comprende símbolos y una gramática que define la manera en que se pueden usar estos símbolos. Aprenda los símbolos y la gramática, y sus imágenes serán comprensibles para todo aquel que reconozca estos símbolos y conozca la gramática.

Aunque, ¿por qué el uml? Usted podría usar cualesquiera símbolos y reglas con el fin de crear su propio lenguaje de modelado, pero el truco estaría en hacer que otros también lo usaran. Si sus aspiraciones son inventar un mejor lenguaje de modelado, entonces no me corresponde detenerlo. Debe saber que el uml se considera un estándar y que lo que este lenguaje es o no es lo define un consorcio de empresas que constituyen el Object Management Group (omg, Grupo de Administración de Objetos). La especificación del uml está definida y ha sido publicada por el omg en www.omg.org.

La Concepción del UML

El UML es la creación de Grady Booch, James Rumbaugh e Ivar Jacobson. Estos caballeros, apodados recientemente “Los tres amigos”, trabajaban en empresas distintas durante la década de los años ochenta y principios de los noventa y cada uno diseñó su propia metodología para el análisis y diseño orientado a objeto. Sus metodologías predominaron sobre las de sus competidores. A mediados de los años noventa empezaron a intercambiar ideas entre si y decidieron desarrollar su trabajo en conjunto.

En 1994 Rumbaugh ingresó a Rational Software Corporation, donde ya trabajaba Booch. Jacobson ingreso a Rational un año después; el resto, como dicen es historia.

Los anteproyectos del UML empezaron a circular en la industria del software y la reacciones resultantes trajeron consigo considerables modificaciones. Conforme a diversos corporativos vieron que el UML era útil a sus propósitos, se conformo un consorcio del UML. Entre los miembros se encuentran DEC, Hewlett-Packard, Intellicorp, Microsift, Oracle, Texas Instrument y Rational. En 1997 el consorcio produjo la versión 1.0 del UML y lo puso a consideración del OMG (Grupo de administración de objetos) como respuesta a su propuesta para un lenguaje de modelado estándar.

El consorcio aumento y generó la versión 1.1 misma que se puso nuevamente a consideración del OMG. El grupo adopto esta versión a finales de 1997. El OMG se encargó de la conservación del UML y produjo otras dos versiones en 1998- El UML ha llegado a ser el estándar de facto en la industria del software, y su evolución.

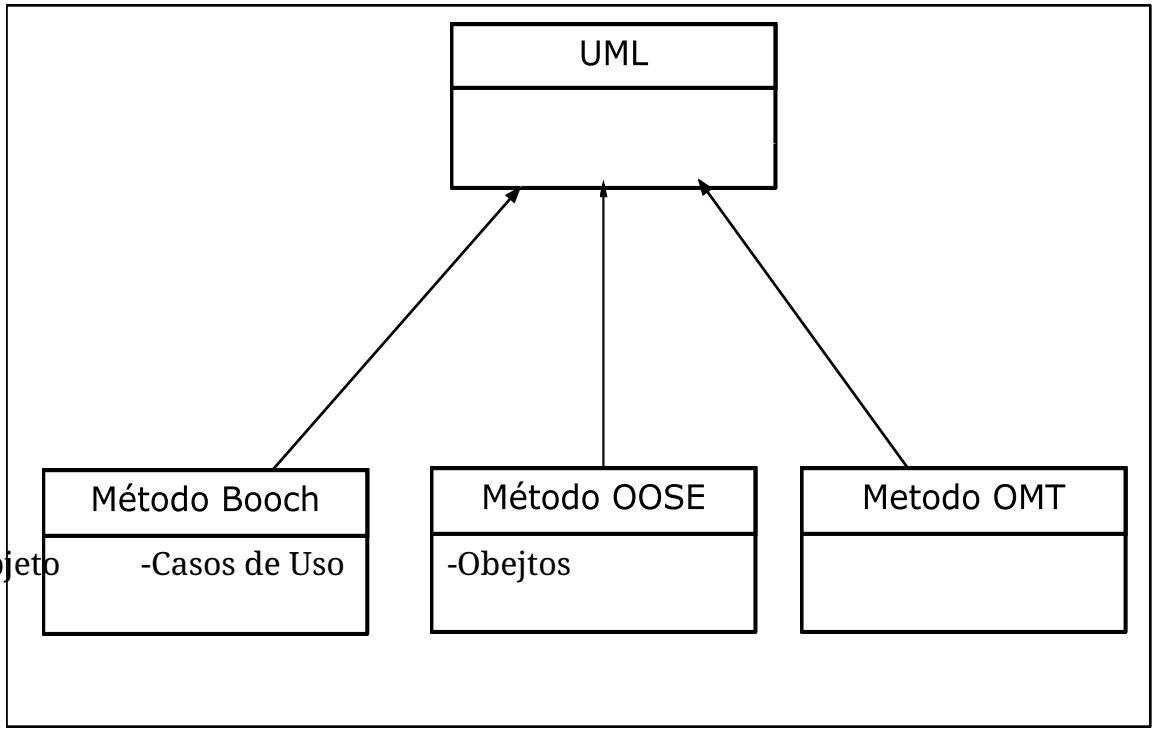


Figura 6. Modelo UML esquematizando el origen de UML

Si nadie está modelando, ¿por qué debe hacerlo usted?

Una persona racional podría preguntar: ¿por qué entonces, si Bill Gates está ganando miles de millones escribiendo software sin hacer un hincapié significativo en el modelado formal, debo preocuparme acerca del UML? La respuesta es que casi el 80% de todos los proyectos de software fallan. Estos proyectos sobrepasan sus presupuestos, no proporcionan
las características que los clientes necesitan o desean o, lo que es peor, nunca se entregan. La tendencia actual es llevar al exterior el desarrollo del software, hacia las naciones en desarrollo o del tercer mundo. La idea básica es que si los ingenieros estadounidenses especializados en software están fallando, entonces si se paga una quinta parte a un
desarrollador euroasiático de software esto permitirá a las empresas intentar tener éxito con una frecuencia cinco veces mayor. ¿Qué están hallando estas empresas que están llevando el desarrollo hacia el exterior? Están descubriendo que Estados Unidos tiene algunos de los mejores talentos y recursos disponibles, y que la mano de obra barata en
lugares alejados sólo introduce problemas adicionales y tampoco es garantía de

éxito. La respuesta real que se necesita consumir más tiempo en el análisis y el diseño del software, y esto significa modelos.

Modelado y el futuro del desarrollo de software

Un énfasis creciente en el análisis y diseño formales no significa el fin del crecimiento de la industria del software; significa que los días del salvaje, salvaje oeste de las décadas de 1980 y 1990 llegarán al momento en que terminen; pero todavía está el salvaje, salvaje oeste de los hackers, allí en la tierra del software, y estará por algún tiempo. Lo que un énfasis creciente en el análisis y diseño del software significa precisamente ahora es que los profesionales capacitados en UML tienen una oportunidad única para capitalizar este interés creciente en este lenguaje; también significa que, de manera gradual, menos proyectos fallarán, la calidad del software mejorará y se esperará que más ingenieros en software aprendan el UML.

Primeros Pasos

UML es un **lenguaje de modelado**. Los modelos UML representan tanto la **estructura estática**, con el comportamiento dinámico de nuestro sistema. Es un **lenguaje visual** que, como desarrolladores, nos sirve para comunicarnos con el usuario, obteniendo y validando los requerimientos y también con colegas desarrolladores, para discutir las funcionalidades del sistema. Además de modelar nuestro sistema, obtenemos una documentación precisa de él. Los modelos guiarán el desarrollo del sistema, por lo que podemos considerarlos como los **cimientos** sobre los cuales se edificará.

UML es un lenguaje de modelado de **propósito general**. Esto significa que podemos modelar cualquier tipo de aplicaciones, desde servicios web hasta sistemas de telefonía celular. Su variedad de modelos y expresividad cubren todos los pasos del análisis y diseño, desde la obtención de requerimientos hasta la especificación formal de los componentes del sistema. Con respecto a la capacidad de abstracción, UML cuenta con una amplia gama de **sabores** para visualizar un determinado componente desde diferentes perspectivas. También cuenta con una rica **expresividad** para agrupar los componentes según diferentes ángulos, adecuándose tanto a la estructura estática como a las características dinámicas o de comportamiento.
Como cualquier otro lenguaje, UML tiene reglas que guían la construcción de sus modelos. Por ejemplo, el lenguaje castellano tiene reglas ortográficas y modelos verbales que rigen la estructura del lenguaje. Estas reglas nos dicen que *Antonia y Charoltte son dos gatitas preciosas* es una oración valida, mientras

que *casa llueve ordenadamente verde no* es una oración bien formada. Esta ultima oración nos permite introducir dos conceptos importantes: sintaxis y semántica.

La **sintaxis** se refiere, en este casa, a que todas las palabras están formadas correctamente y separadas con los mecanismos establecidos, como espacios en blanco, comas o puntos y comas. La RAE define la sintaxis de la siguiente manera: *parte de la gramática que enseña a coordinar y unir las palabras para formar las oraciones y expresar conceptos.*

En el caso de la última oración, podemos ver que es sintácticamente correcta. Cada palabra está bien escrita. El problema es que no tiene sentido, y es aquí donde entra en juego el concepto de **semántica**, es decir, el **significado**.

Nuevamente, nos ayudamos con la RAE para definir la palabra semítica: *perteneciente o relativo a la significación de las palabras.*

Las sintaxis de UML nos dice como construir nuestros modelos. La semántica nos dice cómo interpretarlos. La semántica juega entonces un papel crucial, ya que nos garantiza, a través de una modelo, que todas las personas involucradas comprendan exactamente la información contenida en el. Si un mismo modelo es entregado a dos programadores para su codificación, se espera que el código resultante sea equivalente.

UML como documentación.

Claramente, UML es mucho más que un lenguaje para generar **documentación** de nuestro sistema. Sin embargo, es un beneficio extra que viene sin costo, ya que al modelar nuestro sistema con modelos UML obtenemos una documentación valiosa. El propósito de la documentación de un sistema de software consiste en acompañar el código resultante con todo aquello que pueda ser útil para su comprensión. Una buena documentación es fundamental a la hora de introducir cambios en un sistema de software, ya que nos permite conocer rápidamente la **estructura interna** de un modulo o su interacción con el resto del sistema. Entonces, los modelos beneficiaran enormemente la calidad de la documentación del sistema.

Filosofía del modelo UML.

Otra característica importante del UML es que nos permite abstraernos de los detalles físicos de implementación cuando estamos modelando. Un error común en la etapa de modelado es adelantarse e introducir detalles de implementación de forma anticipada. Esto trae como consecuencia decisiones prematuras que pueden afectar la calidad de nuestro sistema. Es fundamental evitar incluir en nuestros modelos detalles de implantación, de manera de poder razonar con mayor solidez sobre el sistema, considerando todas las alternativas y no **atarse** a una implementación en particular antes de tiempo. Un modelo que se adelante en decisiones de implementación en particular antes de tiempo. Un modelo que se adelante en decisiones de implementación resultara en un modelo de bajo nivel, poco flexible, y obtendremos un sistema demasiado rígido para enfrentar cambios. Modelar es concentrarse en las propiedades del sistema. Las cuestiones de implementación deberán esperar su lugar en el proceso de desarrollo.

Constructores UML

La expresividad de los modelos UML cubre todos los aspectos del análisis y del diseño, incluyendo el análisis arquitectónico, sin ser un lenguaje sumamente complejo. Esta ausencia de complejidad se debe a que básicamente tenemos en cualquier modelo UML solamente tres cosas:

- Entidades.
- Relaciones.
- Grupos de entidades que se relacionan.