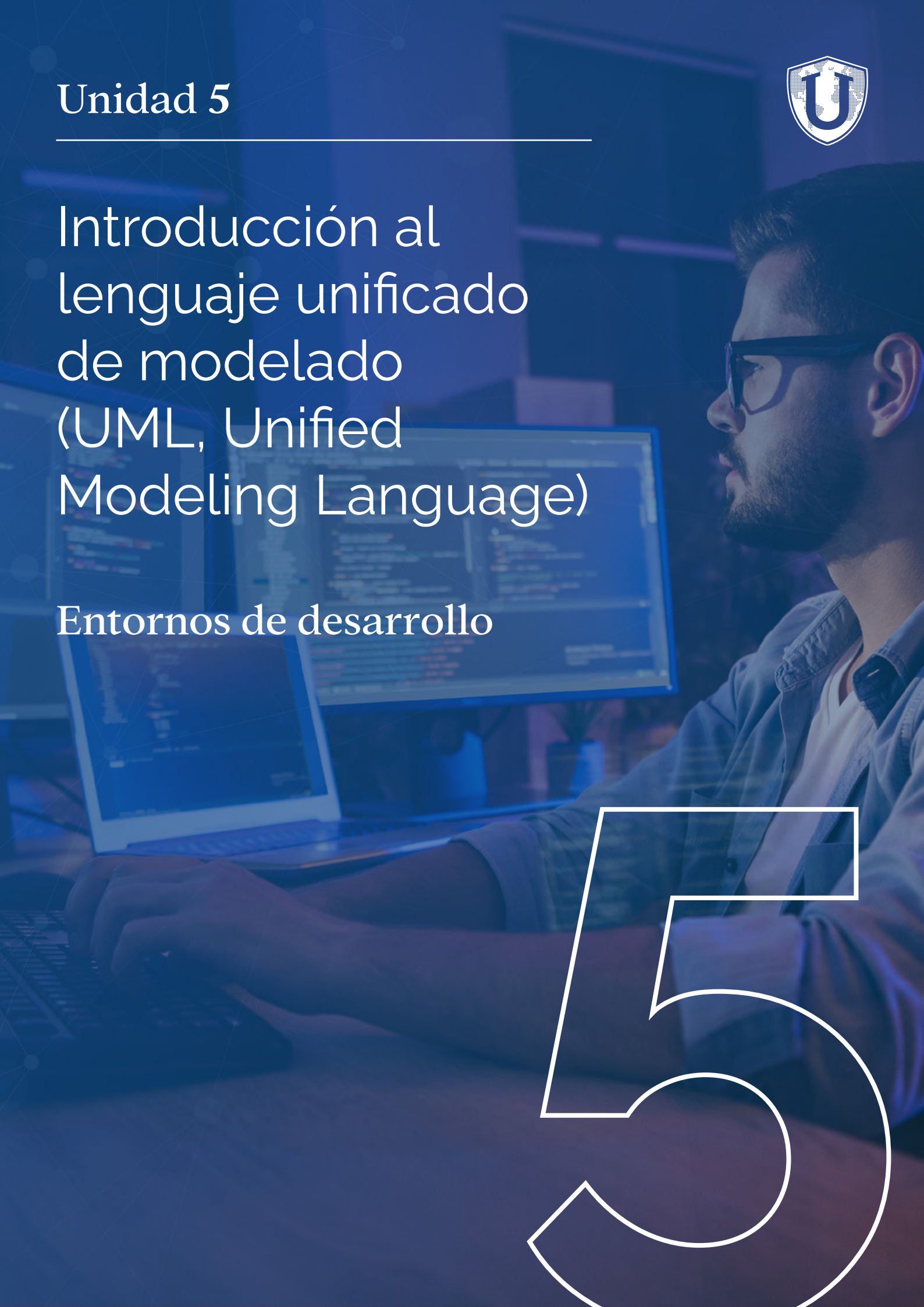
Unidad 5

Introducción al lenguaje unificado de modelado (UML, Unified Modeling Language)

Entornos de desarrollo



5

Índice

Introducción

La llegada de los elementos informáticos ha llevado a la necesidad de buscar nuevas soluciones a los problemas que estos nos han planteado. En muchos casos dichas soluciones requieren de la construcción de teorías y metodologías completas, es el caso ocurrido con los OO.

En la búsqueda de mejores sistemas se teorizaron los OO, orientados a objetos, los cuales pretendían trabajar a partir de los elementos en el mundo real. Con el fin de lograr esto se desarrolló, tras mucho esfuerzo, el lenguaje UML, con el cual crear POO, programas orientados a objetos.

En este tema veremos qué es este lenguaje UML y que características posee para hacerlo tan relevante para este tipo de programación.

Al finalizar esta unidad

- Sabremos qué es UML y sus principales características.
- Aprenderemos la evolución del UML
- Conoceremos los diferentes diagramas que se pueden representar.
- Descubriremos las principales herramientas para UML

5.1. Características

El UML es un lenguaje de modelado de sistemas, como tal cuenta con todas las características propias d ellos lenguajes de programación y algunas particularidades, entre estas las que destacan son:

- **Documentación:** El empleo de UML permite una documentación sencilla, ya que los diagramas que puede generar sirven como parte de esta documentación, permitiendo el empleo de un lenguaje técnico y preciso a la vez que el diagrama da una imagen clara que los no capacitados pueden seguir fácilmente.
- **Estandarización:** El UML se considera el lenguaje estándar para los sistemas orientados a objetos.
- **Implantación en sistemas OO, orientados a objetos:** El UML es un lenguaje creado y destinado a la creación de sistemas orientados a objetos, por lo que su radio de uso se reducirá a este tipo de sistemas.

  Esto indica que el lenguaje UML esta preparado para representar objetos reales dentro del sistema, así como procesos empleando estos como piezas para ello.
- **Flexibilidad:** El UML permite una gran flexibilidad a la hora de la representación, y de la modificación de elementos dentro de programas completamente formados.

  La inclusión, eliminación o modificación de los elementos dentro de un POO son sencillos de hacer u en muchos casos no requiere de grandes cambios para su adaptación.
- **Capacidad de diagramación:** La mayor característica sin duda de los elementos conformados con UML, es su capacidad para una rápida y sencilla transformación en diagramas.

  Dentro de esta capacidad es necesario destacar el hecho de que estos diagramas no son de un solo tipo y estructura, sino que, adaptándose a las necesidades, el UML permite el empleo de múltiples tipos de diagramas con los que representar un mismo objeto, permitiendo dar distintas visualizaciones de un mismo programa en función de los requisitos de la información que necesitemos, ya sea una simple visualización de los elementos a una explicación de los procesos con su temporalización y secuencia.

5.2. Versiones

En la década de los ochenta, un creciente número de empresas comenzó a utilizar la POO para crear sus aplicaciones, lo cual generó la necesidad de un proceso estándar de análisis y diseño orientado a objetos. Muchos metodologistas (incluyendo a Booch, Rumbaugh y Jacobson) produjeron y promocionaron, por su cuenta, procesos separados para satisfacer esta necesidad. Cada uno de estos procesos tenía su propia notación, o "lenguaje" (en forma de diagramas gráficos), para transmitir los resultados del análisis y el diseño.

A principios de la década de los noventa, diversas compañías (e inclusive diferentes divisiones dentro de la misma compañía) utilizaban sus propios procesos y notaciones únicos. Al mismo tiempo, estas compañías querían utilizar herramientas de software que tuvieran soporte para sus procesos particulares. Con tantos procesos, se les dificultó a los distribuidores de software proporcionar dichas herramientas. Evidentemente era necesario contar con una notación y un proceso estándar.

En 1994, James Rumbaugh se unió con Grady Booch en Rational Software Corporation (ahora una división de IBM), y comenzaron a trabajar para unificar sus populares procesos, Pronto se unió a ellos Ivar Jacobson. En 1996, el grupo liberó las primeras versiones de UML para la comunidad de ingeniería de software, solicitando retroalimentación. Casi al mismo tiempo, una organización conocida como Object Management Group hizo una invitación para participar en la creación de un lenguaje común de modelado. El OMG (www.omg.org) es una organización sin fines de lucro que promueve la estandarización de las tecnologías orientadas a objetos, emitiendo lineamientos y especificaciones como UML. Varias empresas (entre ellas HP, IBM, Microsoft, Oracle y Rational Software) habían reconocido ya la necesidad de un lenguaje común de modelado. Estas compañías formaron el consorcio UML Partners (Socios de UML) en respuesta a la solicitud de proposiciones por parte del OMG (el consorcio que desarrolló la versión 1.1 de UML y la envió al OMG). La propuesta fue aceptada y, en 1997, el OMG asumió la responsabilidad del mantenimiento y revisión de UML en forma continua. La versión que está ahora disponible marca la primera modificación importante al UML desde el estándar de la versión 1.1 de 1997, presentada como la versión UML 2.

Las primeras versiones de UML, desde la 1.1 a la 1.5, se emplearon hasta 2005, en ellas se eliminaron la mayoría de los errores que se pudieron encontrar y se incluyó una gran cantidad de información de otras metodologías orientadas a objetos, MOO, con el fin de que estas fueran compatibles con UML.

En 2005 se lanza la versión 2.1, en su versión 2.1.1, y 2.1.2 en 2007, el 2.1 fue empleado como avance y modelo para las siguientes versiones, pero no como un modelo real. Las sucesivas versiones se siguieron lanzando en 2009, la 2.2; en 2010, la 2.3; en 2011, la 2.4.1; y en 2015, la 2.5.1, aunque no se lanzó oficialmente hasta 2017.

La 2.5.1. es actualmente la última versión, aunque ya se está trabajando en las terceras versiones, la 3.1.

Debido a las complejidades y extensión del UML, es muy difícil conocerlo completamente, por lo que se suele dividir en función de las necesidades y trabajarse mediante el empleo de programas de creación de diagramas. Esto implica que generalmente no se trabaja directamente con el lenguaje, sino que se hace uso de este a través de terceros como es el programa, por lo que las versiones usadas serán las de dichos programas, aunque siempre que sea posible es preferible elegir un programa con una versión actualizada.

5.3. Diagramas UML

El UML permite la creación de múltiples diagramas para los POO, los cuales nos darán una visión clara y con distintos matices del código, a la vez que facilitan su estructuración. Los diagramas son proyecciones gráficas de un sistema destinadas a su clara visualización.

Existen un total de 14 tipos de diagramas que pueden ser desarrollados por UML, divididos en dos grandes categorías:

- Estructura.
- Comportamiento.
  - Dentro de comportamiento encontramos también los diagramas de interacción.

Categoría	Tipo de diagrama	Aplicación
Estructura	Diagrama de clases	Visualización de las clases del programa
	Diagrama de objetos	Es un diagrama de clase en una estancia específica en lugar de una visión general
	Diagrama de componentes	Visualización de los componentes del sistema y de sus relaciones
	Diagrama de estructura compositiva	Clasifica las clases y especifica las relaciones entre ellas.
	Diagrama de paquete	Agrupa clases en paquetes y los jerarquiza
	Diagrama de distribución	Distribución física de los elementos en nodos informáticos
	Gráfica de perfil	Visualiza elementos del metamodelo mediante estereotipos, condiciones, etc.
Comportamiento	Diagrama de casos de uso	Representa varias aplicaciones de un sistema y los elementos que lo llevan a cabo
	Diagrama de actividades	Muestra el proceso y secuenciación de este, con hincapié en las desviaciones paralelas
	Diagrama de estados	Visualización de los cambios de estado sufridos por un objeto
Comportamiento: interacción	Diagrama secuencial	Secuenciación temporal del programa con las interacciones entre objetos
	Diagrama de colaboración	Distribución de los roles y la interacción entre ellos
	Diagrama de tiempos	Visualización temporal de los procesos en relación con los cambios de estado
	Diagrama de interacción	Visualización de sistemas complejos mediante elementos interactivos

Imagen 1. Tipos de diagramas

Los elementos necesarios para la fabricación de diagramas, así como el estudio en profundidad de algunos de ellos, seleccionados por su uso e importancia, se verán en los temas siguientes.

5.4. Utilización en metodologías de desarrollo orientadas a objetos

UML es un lenguaje grafico que permite visualizar, especificar, construir y documentar todos los objetos/elementos de un sistema.

UML nos proporciona un lenguaje estándar para construir el mapa/plano de un sistema, cubriendo cosas conceptuales como los procesos de los negocios y las funciones del sistema, así como cosas concretas como clases escritas en lenguaje especifico de programación, esquemas de base de datos y componentes de software reutilizable.

El UML es una técnica de modelado de objetos y como tal supone una abstracción de un sistema para llegar a construirlo en términos concretos. El modelado no es más que la construcción de un modelo a partir de una especificación.

Un modelo es una abstracción de algo, que se elabora para comprender ese algo antes de construirlo. El modelo omite detalles que no resultan esenciales para la comprensión del original y por lo tanto facilita dicha comprensión.

Los modelos, además, al no ser una representación que incluya todos los detalles de los originales, permiten probar más fácilmente los sistemas que modelan y determinar los errores. Los modelos permiten una mejor comunicación con el cliente por distintas razones:

- Es posible enseñar al cliente una posible aproximación de lo que será el producto final.
- Proporcionan una primera aproximación al problema que permite visualizar cómo quedará el resultado.
- Reducen la complejidad del original en subconjuntos que son fácilmente tratables por separado.

El uso de UML facilita ampliamente el entendimiento de las relaciones que hay entre los objetos a programar y saber cómo interactúan.

UML no define un proceso concreto que determine las fases de desarrollo de un sistema, las empresas pueden utilizar UML como el lenguaje para definir sus propios procesos y lo único que tendrán en común con otras organizaciones que utilicen UML serán los tipos de diagramas.

UML es un método independiente del proceso. Los procesos de desarrollo deben ser definidos dentro del contexto donde se van a implementar los sistemas.

5.5. Herramientas CASE con soporte UML

Se puede definir a las herramientas CASE como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software.

La razón para la creación de estas herramientas fue el incremento en la velocidad de desarrollo de los sistemas. Por esto, las compañías pudieron desarrollar sistemas sin encarar el problema de tener cambios en las necesidades del negocio, antes de finalizar el proceso de desarrollo.

La tecnología CASE supone la automatización del desarrollo del software, contribuyendo a mejorar la calidad y la productividad en el desarrollo de sistemas de información y se plantean los siguientes objetivos:

- Permitir la aplicación práctica de metodologías estructuradas, las cuales al ser realizadas con una herramienta se consigue agilizar el trabajo.
- Facilitar la realización de prototipos y el desarrollo conjunto de aplicaciones.
- Simplificar el mantenimiento de los programas.
- Mejorar y estandarizar la documentación.
- Aumentar la portabilidad de las aplicaciones.
- Facilitar la reutilización de componentes software.
- Permitir un desarrollo y un refinamiento visual de las aplicaciones, mediante la utilización de gráficos

Automatizar:

- El desarrollo del software.
- La documentación.
- La generación del código.
- El chequeo de errores.
- La gestión del proyecto.

Permitir:

- La reutilización del software.
- La portabilidad del software.
- La estandarización de la documentación.

De una forma esquemática podemos decir que una herramienta CASE se compone de los siguientes elementos:

- **Repositorio (diccionario)** donde se almacenan los elementos definidos o creados por la herramienta, y cuya gestión se realiza mediante el apoyo de un Sistema de Gestión de Base de Datos (SGBD) o de un sistema de gestión de ficheros.
- **Metamodelo (no siempre visible)**, que constituye el marco para la definición de las técnicas y metodologías soportadas por la herramienta.
- **Carga o descarga de datos**, son facilidades que permiten cargar el repertorio de la herramienta CASE con datos provenientes de otros sistemas, o bien generar a partir de la propia herramienta esquemas de base de datos, programas, etc. que pueden, a su vez, alimentar otros sistemas. Este elemento proporciona así un medio de comunicación con otras herramientas.
- **Comprobación de errores**, facilidades que permiten llevar a cabo un análisis de la exactitud, integridad y consistencia de los esquemas generados por la herramienta.
- **Interfaz de usuario**, que constará de editores de texto y herramientas de diseño gráfico que permitan, mediante la utilización de un sistema de ventanas, iconos y menús, con la ayuda del ratón, definir los diagramas, matrices, etc. que incluyen las distintas metodologías.

Algunos ejemplos de las principales herramientas para el diseño de UML:

- Microsoft Visio
- yUML
- DIA
- Poseidon
- MagicDraw
- Papyrus UML
- Modelio
- StarUML
- WhiteStarUML

www.universae.com