Actividad 3 - Diagramas Estáticos

Samuele Tonarini y Erik Pardillo

CIFP Francesc de Borja Moll

13 Marzo 2022

Tabla de Contenidos

Introducción

En esta practica se describen los diferentes diagramas UML de tipo estático, viendo primero los de clase, componentes y objetos, para después pasar por los de perfil, estructura compuesta, despliegue y, al final, de paquete.

Estos diagramas son el estándar para representar diferentes modelos de un determinado sistema; Principalmente se usan para que personas no familiarizadas con un programa en especifico y/o no familiarizadas con el desarrollo del software en si puedan entender las diferentes partes de un sistema, como funcionan, como se relacionan, y como se representan en la vida real.

Ademas, este tipo de diagramas nos permiten visualizar el sistema de manera abstracta, pudiendo analizarlo antes de la implementación y pudiendo detectar y solucionar problemas antes de introducirlos en el sistema.

Diagramas de Clases

Un diagrama de clases es una estructura estática que se utiliza en el desarrollo de software. Un diagrama de clases ensena las clases, los atributos, las operaciones y la relación entre ellos. Esto ayuda a los ingenieros de software a crear el código de una aplicación. También se usa para describir, visualizar y documentar todas las características de un sistema.

Los diagramas de clase son los únicos diagramas UML que se asocian directamente con los lenguajes orientados a objetos.

Los diagramas de clase son uno de los diagramas más usados en la programación, ya que constituyen la base de los diagramas de componentes y de despliegue, y describen las responsabilidades en un sistema. Aunque por su naturaleza, son unos diagramas muy cercanos al software, y, por eso, tienen menos valor para una persona no familiarizada al desarrollo.

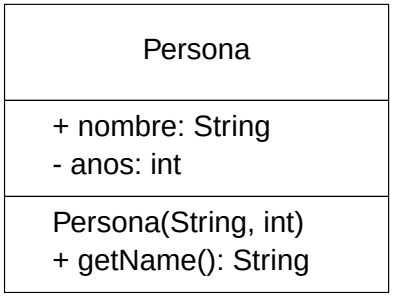
El diagrama de clases estándar está compuesto por tres partes:

Una sección superior que contiene el nombre de la clase. Una sección central que contiene los atributos de la clase. Se usa esta sección para describir las cualidades de la clase. Esto solo es necesario al describir una instancia específica de una clase. Y por ultimo una sección inferior que incluye los métodos de la clase. Cada operación requiere su propia línea y se necesita el tipo de parámetros y el tipo de dato que va a devolver ese método.

Todas los atributos y métodos de las clases poseen diferentes niveles de acceso dependiendo de la visibilidad que tienen, esto se representan usando diferentes símbolos antes del nombre del determinado método o atributo. Los símbolos son:

- **+** Público
- **-** Privado
- # Protegido
- ˜ Paquete

Aquí dejamos un ejemplo muy sencillo de un diagrama de una clase, con diferentes métodos y atributos:

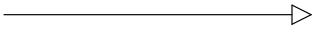


Relaciones de Clase

Tenemos tres tipos de relaciones entre las diferentes clases:

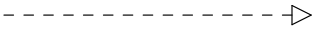
Generalizaciones

Las generalizaciones son normalmente conocidas como "Herencia" porque vincula una subclase a su superclase, pero no se puede utilizar para modelar la implementación de interfaces.



Asociaciones

Asociaciones muestra una relación estática entre dos entidades.



Dependencias

Dependencia muestra que una clase depende de otra.

Diagramas de Componentes

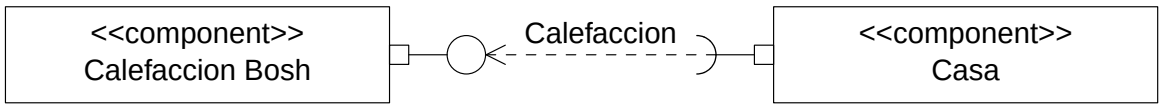
Los diagramas de componentes UML representan las relaciones entre los componentes individuales del sistema mediante una vista de diseño estática. Pueden demostrar aspectos de modelado lógico y físico.

En el contexto del UML, los componentes son partes de un sistema independientes entre sí, que pueden sustituirse con componentes equivalentes. Son autocontenidos y encapsulan estructuras de cualquier grado de complejidad. Los elementos encapsulados solo se comunican con los otros a través de interfaces.

Los componentes suelen encapsular clases y, por eso, también se los conoce como subformas o especializaciones de clases. Al igual que las clases, tienen una estructura compuesta y pueden definirse en más detalle con atributos y operaciones

¿Qué elementos tiene un diagrama de componentes?

El lenguaje de modelado UML utiliza una notación normalizada para crear los diagramas de componentes, que se basa en su propio conjunto de caracteres y símbolos. Los componentes pueden ser clases, agrupaciones de clases, paquetes o artefactos.

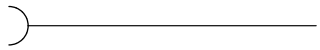


Relaciones de Componentes

Tenemos cinco tipos de relaciones entre los diferentes componentes:

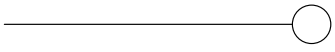
Interfaz Ofrecida

Símbolo para una o mas interfaces claramente definidas que proporcionan funciones, servicios o datos al mundo exterior.



Interfaz Requerida

Símbolo de una interfaz necesaria para recibir funciones, servicios o datos del exterior.



Puerto

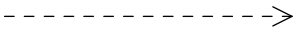
Este símbolo indica un punto de interacción independiente entre un componente y su entorno.

Relación

Las lineas actúan como conectores e indican las relaciones entre los componentes.

Relación de Dependencia

Conector especial para expresar una relación de dependencia entre los componentes del sistema.



Componentes

Hay tres maneras en que el símbolo del componente puede ser usado.

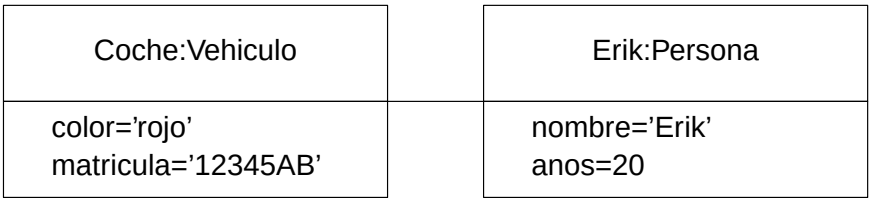
- Rectángulo con el estereotipo del componente (el texto <<componente>>). El estereotipo del componente se suele utilizar encima del nombre del componente para evitar confundir la forma con un icono de clase.
- Rectángulo con el icono del componente en la esquina arriba a la derecha y el nombre del componente.
- Rectángulo con el icono del componente y el estereotipo del componente.

Diagramas de Objetos

Un diagrama de objetos UML representa una instancia específica de un diagrama de clases en un momento determinado. Un diagrama de objetos se enfoca en los atributos de un conjunto de objetos y cómo esos objetos se relacionan entre sí. Los diagramas de objetos son fáciles de crear: están compuestos de objetos, representados por rectángulos, conectados mediante líneas.

Permiten una revisión de una iteración específica de un sistema general, dando una vista de nivel alto del sistema que desarrollarás. También sirven como ejemplo de un diagrama de clases que creaste para la estructura general del sistema, por medio de diagramas de objetos para casos de uso específicos.

Aquí podemos ver un ejemplo de objeto:



Títulos de clases

Los títulos de clases son los atributos específicos de una clase dada. Se pueden listar títulos de clases como elementos en el objeto o incluso en las propiedades del propio objeto

Atributos de clases

Los atributos de clases se representan por medio de un rectángulo con dos pestañas que indica un elemento de software.

Enlaces

Los enlaces son líneas que conectan dos figuras de un diagrama de objetos entre sí. El diagrama de objetos corporativo siguiente muestra cómo los departamentos están conectados al estilo del organigrama tradicional.

Diferencias entre el diagrama de clases y el diagrama de objetos

En UML, los diagramas de objetos muestran un instante en el sistema y las relaciones entre distintas instancias. Algunas lineas generales en comparación con el diagrama de clases son las siguientes: El diagrama de objetos utiliza notaciones similares a los usados en el diagrama de clases. Los diagramas de objetos se utilizan para modelar los elementos que están presentes en un diagrama de clases. El diagrama de objetos muestra los clasificadores reales del sistema y las relaciones entre ellos en un punto específico del tiempo. Los diagramas de objetos se pueden instanciar como diagrama de clases, despliegue, componentes e, incluso, casos de uso.

En ninguno de los dos diagramas se muestran los mensajes entre los elementos que colaboran, ya que se trata de diagramas estructurales.

Diagramas de Perfil

Un diagrama de perfiles permite extender la especificación UML para su uso con una plataforma de programación en particular o modelar sistemas destinados a ser usados en un dominio en específico.

Para comprender cómo funcionan los perfiles y el diagrama de perfiles es necesario entender los fundamentos de UML, ya que se trata de un metamodelo. La palabra modelo implica una abstracción de algún sistema del mundo real. La palabra meta implica una abstracción adicional de cualquier concepto al que lo apliquemos. Los nodos y elementos gráficos utilizados en los diagramas de perfil son: perfil, metaclase, estereotipo, extensión, referencia y aplicación de perfil.

Perfil

Un perfil es un paquete que extiende a un metamodelo permitiendo adaptar el metamodelo con directrices que son específicas de un dominio, plataforma o método de desarrollo de software en particular. En otras palabras, el perfil es una herramienta de extensión ligera del estándar UML. Un perfil utiliza la misma notación que un paquete del diagrama de paquetes, con la adición de que la palabra clave "perfil" se muestra antes o encima del nombre del paquete.

Metaclase

Una metaclase es una clase de perfiles y un elemento empaquetable que puede ser extendido a través de uno o más estereotipos. Se puede mostrar una metaclase con el estereotipo opcional "Metaclase" que se muestra arriba o antes de su nombre. La metaclase puede extenderse por uno o más estereotipos utilizando un tipo especial de asociación: extensión.

Estereotipo

Un estereotipo es una clase de perfil que define cómo una metaclase existente puede extenderse como parte de un perfil. Permite el uso de una plataforma o una terminología específica del dominio o una notación en lugar de, o además de, los utilizados para la metaclase extendida. Un estereotipo no puede usarse solo, sino que siempre debe usarse con una de las metaclases que extiende. El estereotipo no puede extenderse por otro estereotipo.

Un estereotipo usa la misma notación que una clase. Dado que el estereotipo es una clase, puede tener propiedades. Las propiedades de un estereotipo se entienden como definiciones de etiqueta. Cuando se aplica un estereotipo a un elemento del modelo, los valores de las propiedades se denominan valores etiquetados.

Extensión

Una extensión es la relación de asociación que se usa para indicar que las propiedades de una metaclase se extienden a través de un estereotipo, y brinda la posibilidad de agregar flexiblemente los estereotipos a las clases y eliminarlas más adelante, si es necesario.

Un extremo de la asociación de extensión es una propiedad ordinaria y el otro extremo es un extremo de extensión. La propiedad vincula la extensión a una metaclase, mientras que el extremo de la extensión vincula la extensión al estereotipo que extiende la metaclase. El extremo de la extensión es un extremo navegable, propiedad de la extensión. Esto permite que una instancia de estereotipo se adjunte a una instancia del clasificador extendido sin agregar una propiedad al clasificador.

Referencia

La referencia es una relación de importación representada por la importación del elemento "metaclassReference" y la importación del paquete "metamodelReference". Las importaciones de elementos "metaclassReference" y las importaciones de paquetes "metamodelReference" sirven para dos propósitos:

Identificar los elementos del metamodelo de referencia que importa el perfil. Especificar las reglas de filtrado del perfil. Las reglas de filtrado determinan qué elementos del metamodelo son visibles cuando se aplica el perfil y cuáles están ocultos.

Aplicación de perfil

La aplicación de perfil es una relación dirigida que se utiliza para mostrar qué perfiles se han aplicado a un paquete. Se pueden aplicar uno o más perfiles a un paquete que se crea a partir del mismo metamodelo que se extiende por el perfil. Aplicar un perfil significa que está permitido, pero no necesariamente requerido, aplicar los estereotipos que se definen como parte del perfil.

Es posible aplicar múltiples perfiles a un paquete siempre que no tengan restricciones en conflicto. Si un perfil que se está aplicando depende de otros perfiles, entonces esos perfiles deben aplicarse primero.

Cuando se aplica un perfil, se deben crear instancias de los estereotipos apropiados para aquellos elementos que son instancias de metaclases con extensiones requeridas. El modelo no está bien formado sin estos casos.

¿Cuántos tipos de diagramas de perfiles existen?

Existen distintos tipos de diagramas de perfiles que son programados dependiendo el tema que el UML está tratando. Si bien, estos pueden extender a, prácticamente, cualquier tema, los más destacados son:

- Diagramas de perfil personal.
- Diagramas de perfiles de herrería.
- Diagramas de perfiles multivariados.
- Diagramas de perfil de riesgo.
- Diagramas de perfil de arquitectura.

Diagramas de Estructura Compuesta

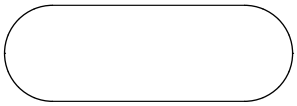
Un diagrama de estructura compuesta es un diagrama de estructura UML que brinda una vista general lógica de todo o parte de un sistema de software. Actúa como una mirada al interior de un clasificador estructurado determinado a fin de definir sus clases de configuración, interfaces, paquetes y las relaciones entre ellos a un micronivel.

Componentes básicos de un diagrama de estructura compuesta

Un diagrama de estructura compuesta está formado por distintos símbolos UML que representan cada parte de un sistema, además de las relaciones entre ellas.

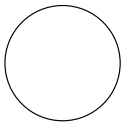
Terminador

Indica puntos de inicio y finalización.



Nodo

Representa eventos o hitos y contiene números.

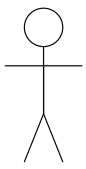


Nodo Rectangular

Representa eventos o hitos y contiene números.

Actor

Interactúa con el sistema desde afuera del sistema.

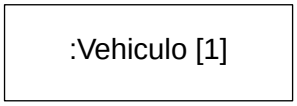


Conector

Ilustra la comunicación entre las partes.

Parte

Actúa como una instancia de ejecución de clases o interfaces.



Puerto

Actúa como un punto de interacción entre una instancia de clasificador y su entorno.

Clase

Agrupa los objetos con propriedades y/o comportamientos comunes.

Interfaz

Especifica el comportamiento que el implementados acepta cumplir.

Diagramas de estructura compuesta versus diagrama de clases

Como diagramas UML, tanto los diagramas de estructura compuesta como los diagramas de clases se emplean para visualizar y organizar a los actores, las interacciones y los artefactos de un sistema. No obstante, aunque los diagramas de estructura compuesta y los diagramas de clases tienen significados similares, son esencialmente distintos en la manera en la que expresan esos significados. En términos simples, los diagramas de estructura compuesta son más específicos y menos ambiguos que los diagramas de clases.

Un diagrama de estructura compuesta permite que los usuarios modelen con mayor claridad las implementaciones de la actividad de un artefacto dentro de una ejecución. También son más adecuados para representar la descomposición en contexto, ya que describen la estructura interna de varias clases y las relaciones establecidas entre ellas. En términos simples, si quieres transmitir información concreta y explícita acerca de los comportamientos y las relaciones dentro de tu sistema, un diagrama de estructura compuesta es la mejor opción.

Diagramas de Despliegue

El diagrama de despliegue es otro de los diagramas de estructura del conjunto de los diagramas de UML 2.0. Es utilizado para representar la distribución física (estática) de los componentes software en los distintos nodos físicos de la red. Suele ser utilizado junto con el diagrama de componentes (incluso a veces con el diagrama de paquetes) de forma que, juntos, dan una visión general de como estará desplegado el sistema de información.

Sus principales características son las siguientes:

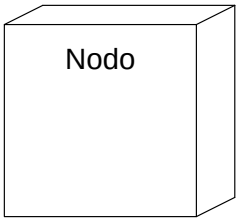
- Permite identificar los nodos en los que trabajará o utilizarán el sistema de información, identificando a su vez agentes externos e internos que interactuen con el sistema.
- Permite representar de forma clara la arquitectura física de la red, así como la distribución del componente software. UML no tiene un tipo de diagramas específico para mostrar la arquitectura de la red, así que se utiliza este tipo de diagrama que cumple efectivamente este cometido, aunque se le suele hacer alguna modificación gráfica. Lo más normal es utilizarlo para dar una visión global, pero es posible utilizarlo para representar partes específicas de la implementación.

Notación

El diagrama de componentes utiliza, principalmente, dos tipos de elementos: Nodos y conexiones.

Nodos

Los nodos se definen como elementos utilizados para representar un elemento físico que interactúa de alguna manera con el sistema o bien forma parte del mismo. Se representa utilizando un cubo tridimensional:



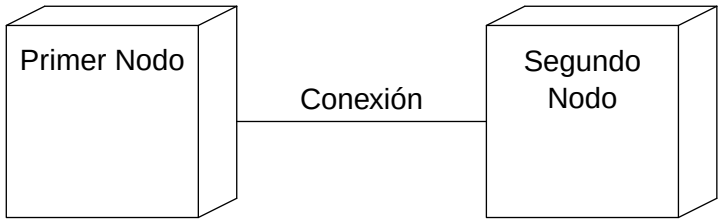
Los nodos también pueden ser representados utilizando iconos personalizados con la finalidad de clarificar el contenido del diagrama. Algunos de estos iconos de uso extendido son:

- Un muro para representar un Firewall.
- Un icono de un PC para representar el equipo de un usuario.
- Un circulo con flechas para identificar a un router.
- Una nube para representar una WAN (aunque no es propiamente un nodo)
- Un cilindro para representar una base de datos.

Un nodo a su vez puede tener nodos incluidos en su interior, dando a conocer que son sistemas separados incluidos dentro del mismo nodo físico. De esta forma se compondrían los nodos compuestos.

Conexión

La conexión representa una asociación entre dos nodos, a través de la cual estos nodos son capaces de transmitir información en forma de mensajes o señales. Se representa utilizando una línea continua que une los dos nodos que se asocian.



Diagramas de Paquetes

El diagrama de paquetes es uno de los diagramas estructurales comprendidos en UML 2.0, por lo que, como tal, representa de forma estática los componentes del sistema de información que está siendo modelado. Es utilizado para definir los distintos paquetes a nivel lógico que forman parte de la aplicación y la dependencia entre ellos. Es principalmente utilizado por desarrolladores y analistas. Es importante destacar que este diagrama es utilizado en los sistemas de información con programación orientada a objetos. El objetivo principal del diseño debe maximizar la cohesión y minimizar el acoplamiento.

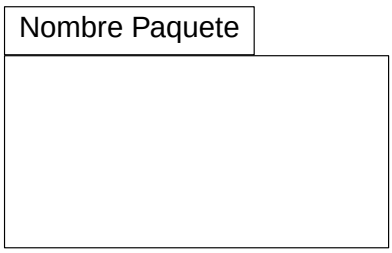
En sí el diagrama es muy sencillo, dependiendo su complicación del detalle con el que se quieran tratar los elementos que mostrará, que puede llegar a ser muy específico.

Elementos de un diagrama de paquetes

El diagrama de paquetes está constituido por dos elementos: Los paquetes y las dependencias.

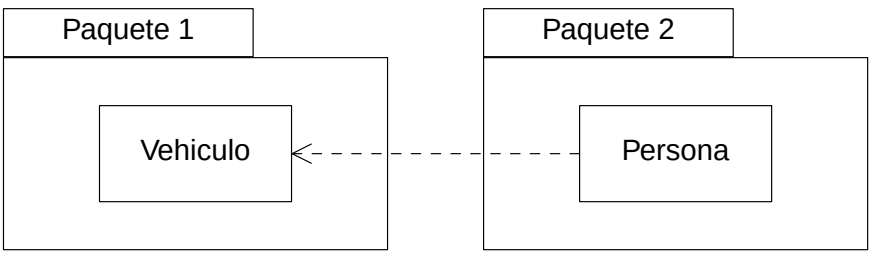
Paquete

Es el elemento clave del diagrama y que da el nombre al mismo. Un paquete es un conjunto de elementos. Puede ser un conjunto de clases, casos de uso, componentes u otros paquetes. No obstante, lo más común es que incluya otros paquetes. Lo ideal es que este conjunto de elementos tenga una función diferenciada del resto de elementos. De esta forma, además de maximizar la cohesión, se dará la máxima claridad al diagrama y, por tanto, al Sistema de Información. Es también importante identificar con nombres representativos de estas funciones a los distintos paquetes. Por supuesto, hay que tener en cuenta el nivel de granularidad del diagrama.



Dependencia entre paquetes

Una dependencia entre paquetes representan que un paquete necesita de los elementos de otro paquete para poder funcionar con normalidad. Se representa con una flecha discontinua que va desde el paquete que requiere la función hasta el paquete que ofrece esa función.



Conclusión

Hemos visto diferentes diagramas UML para crear modelos estaticos de un determinado sistema. Para cada caso en concreto usaremos un mix de estos diagramas para ayudarnos a tener una vision mas concreta de un sistema, ademas de poder compartir el funcionamiento del sistemas con personas externas. Un buen analista de sistemas podra usar cada diagrama a su ventaja y dependiendo de la implementacion en concreto se podra profundizar el conocimiento de un determinado tipo de diagrama.

Para conocer todos los detalles de los modelos UML se recomienda profundizar con la especificacion original y/o un libro recopilatorio como *UML 2.0 in a Nutshell* (O'Reilly).

Bibliografía

Diagramas de Clase

https://www.lucidchart.com/pages/es/tutorial-de-diagrama-de-clases-uml

https://www.edrawsoft.com/es/example-uml-class-diagram.html

Diagramas de Componentes

https://www.ionos.mx/digitalguide/paginas-web/desarrollo-web/diagrama-de-componentes/

https://es.wikipedia.org/wiki/Diagrama_de_componentes

https://diagramasuml.com/componentes/

Diagramas de Objetos

https://es.wikipedia.org/wiki/Diagrama_de_estructura_compuesta

https://www.lucidchart.com/pages/es/diagrama-de-objetos-uml

https://diagramasuml.com/objetos/

Diagramas de Perfil

https://dediagramas.com/perfiles-uml/

https://diagramasuml.com/perfiles/

Diagramas de Estructura Compuesta

https://www.lucidchart.com/pages/es/diagrama-de-estructura-compuesta-uml

Diagramas de Despliegue

https://diagramasuml.com/despliegue/

https://creately.com/blog/es/diagramas/tutorial-de-diagrama-de-despliegue/

Diagramas de Paquete

https://diagramasuml.com/paquetes/

https://es.wikipedia.org/wiki/Diagrama_de_paquetes