PROGRAMACIÓN ORIENTADA A OBJETOS

Master de Computación

II MODELOS y HERRAMIENTAS UML

II.1 UML: Introducción

Técnica de modelado de objetos (I)

- El modelado orientado a objetos es una técnica de especificación semiformal para el paradigma orientado a objetos.
  - Ya que se trata de una técnica semiformal, una parte intrínseca es la notación gráfica asociada.
- El *Lenguaje de Modelado Unificado* (*UML*, Unified Modeling Language) se ha desarrollado en un intento de unificar las distintas notaciones existentes.
- El modelado orientado a objetos se ocupa de comprender y analizar la aplicación y el dominio en el que opera.
  - El punto de partida es la declaración del problema que hay que resolver.
  - Esta declaración, que proporciona una visión conceptual del sistema propuesto, puede ser textual o utilizar una técnica de descripción más formal, como la basada en casos de uso.
- El modelado orientado a objetos consta de tres pasos: modelado de casos de uso, modelado de clases y modelado dinámico.

Técnica de modelado de objetos (II)

- **Modelado de Casos de Uso.** La intención del modelado de casos de uso es identificar cómo se va a usar el sistema y lo que se espera que haga como respuesta a ese uso.
  - Determina cómo la aplicación genera los diversos resultados que se requieren (sin considerar el orden de generación).
  - La información se presenta en forma de *diagrama de caso de uso* y escenarios (scenarios: guiones) asociados.
  - Este paso, que a veces se denomina modelado funcional, está en gran parte orientado a las acciones.
- **Modelado de Clases.** Determina las clases, sus atributos y las relaciones entre las clases.
  - Esta información se presenta en forma de *diagramas de clases*.
  - Este paso está orientado a los datos.
- **Modelado Dinámico.** Determina las acciones realizadas por cada clase y sobre cada clase.
  - Esta información se presenta en forma de *diagramas de comportamiento*.
  - Este paso está orientado a las acciones.

Técnica de modelado de objetos (III)

- Los tres pasos del modelado se realizan realmente en paralelo, puesto que, en el paradigma orientado a objetos, ni los datos ni las acciones tienen prioridad los unos sobre los otros.
- Se usan varias técnicas de modelado para comprender los datos, las acciones y las interacciones entre los datos y las acciones.
  - Durante el proceso de modelado, el conocimiento adquirido sobre la aplicación se representa de distintas formas, cada una de las cuales refleja un aspecto diferente del producto buscado.
  - Los diagramas se actualizan continuamente a medida que se consigue una mejor percepción del sistema que se está modelando.
  - Finalmente, las perspectivas combinadas proporcionan una comprensión global del producto que sería difícil de alcanzar con una sola técnica de modelado.

Propósito del UML (I)

- El Lenguaje de Modelado Unificado (UML) es un lenguaje estándar para poner por escrito un proyecto de sistema y es parte del método de desarrollo del sistema.
  - Puede usarse para visualizar, especificar, construir y documentar un sistema complejo.
- Al tratarse de un lenguaje de modelado, su vocabulario y normas se enfocan a la representación conceptual y física del sistema.
  - El vocabulario y las normas del UML indican cómo crear y leer modelos bien formados gramaticalmente, pero no dicen qué modelos deben crearse ni cuándo hacerlo. Eso es el papel del proceso de desarrollo del sistema.
  - Un proceso bien definido guiará en la decisión de qué artefactos producir, qué actividades y qué trabajadores usar para crearlos y dirigirlos, y cómo usar estos artefactos para medir y controlar el proyecto en su totalidad.
- El UML es un lenguaje para visualizar: un modelo explícito facilita la comunicación.
  - Detrás de cada símbolo de la notación UML hay una semántica bien definida. De esta forma, un programador puede escribir un modelo en UML y otro programador, o incluso una herramienta, puede interpretar ese modelo inequívocamente.

Propósito del UML (II)

- El UML es un lenguaje para especificar: generar modelos precisos, inequívocos y completos.
  - En particular, el UML se aplica a la especificación de todas la decisiones importantes de análisis, diseño e implementación que deben hacerse al desarrollar y desplegar un sistema.
- El UML es un lenguaje para construir: no es un lenguaje de programación visual, pero sus modelos pueden conectarse directamente a diversos lenguajes de programación.
  - Es posible proyectar un modelo UML en un lenguaje de programación como Java, C++ o Visual Basic, o incluso en tablas de una base de datos relacional o en el archivo persistente de una base de datos orientada a objetos. Esta proyección permite generar código en un lenguaje de programación a partir de un modelo UML.
  - También es posible reconstruir un modelo en UML a partir de una implementación.
  - Además, UML es suficientemente expresivo e inequívoco para permitir ejecutar directamente los modelos, simular los sistemas e instrumentar sistemas en ejecución.
- El UML es un lenguaje para documentar: se aplica a la documentación de la arquitectura del sistema y todos sus detalles. También proporciona un lenguaje para expresar requerimientos y para tests.

Modelo conceptual del UML (I)

- El vocabulario del UML abarca tres clases de componentes básicos: cosas, relaciones y diagramas.
  - Las cosas son las abstracciones, que son la parte principal en un modelo; las relaciones conectan estas cosas; los diagramas agrupan colecciones significativas de cosas.
- Un diagrama se representa mediante un grafo conectado de vértices (cosas) y arcos (relaciones). El UML incluye trece diagramas y los divide en dos categorías: diagramas estructurales y diagramas de comportamiento.

- Los diagramas estructurales se usan para capturar la organización física de las cosas en el sistema, cómo se relacionan los objetos entre sí.
  - En UML 2.0, hay seis tipos: *Diagrama de clases*, *de objetos*, *de componentes*, *de estructuras compuestas*, *de despliegue* y *de paquetes*.
- *Diagrama de clases (Class diagrams)*: muestra un conjunto de clases, interfases y colaboraciones y sus relaciones. Se aplica a la vista estática de un sistema.

Modelo conceptual del UML (II)

- *Diagrama de objetos (Object diagrams)*: muestra un conjunto de objetos y sus relaciones. Representan instantáneas de instancias de las cosas que se encuentran en un diagrama de clases.

- *Diagrama de componentes (Component diagrams)*: muestra las organizaciones y dependencias entre un conjunto de componentes. Está dirigido a la vista de implementación estática de un sistema.

- *Diagrama de estructuras compuestas (Composite structure diagramas)*: conceptualmente, este diagrama enlaza diagramas de clase y de componentes para mostrar cómo se combinan los elementos del sistema para realizar comportamientos complejos.

- *Diagrama de despliegue (Deployment diagrams)*: muestra la configuración de los nodos de procesado en tiempo de ejecución y los componentes que están vivos sobre ellos. Se dirige a la vista de despliegue estático de una arquitectura.

Modelo conceptual del UML (III)

- *Diagrama de paquetes (Package diagrams)*: es un tipo especial de diagrama de clases enfocado en cómo se agrupan las clases e interfases.
- Los diagramas de comportamiento permiten capturar requerimientos, operaciones y cambios de estado interno de los elementos del sistema.
  - En UML 2.0 hay siete tipos: *Diagrama de estado, de actividad, de secuencia, de comunicación, de tiempo, de visión de conjunto de la interacción y de caso de uso.*
- *Diagrama de estado (State machine diagrams)*: muestra una máquina de estados, que consiste en estados, transiciones, eventos y actividades. Se aplica a la vista dinámica del sistema y subrayan el comportamiento ordenado por eventos de un objeto.
  - Son especialmente importantes para modelar el comportamiento de una interfase, clase o colaboración, y para modelar sistemas reactivos.
- *Diagrama de actividad (Activity diagrams)*: muestra el flujo de actividad en actividad dentro de un sistema. Se aplica a la vista dinámica de un sistema.
  - Son especialmente importantes para modelar la función de un sistema y subrayan el flujo de control entre objetos.

Modelo conceptual del UML (IV)

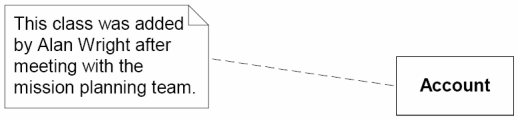
- *Diagramas de interacción (Interaction diagrams)*: muestran una interacción, que consiste en un conjunto de objetos y sus relaciones, incluyendo los mensajes que pueden enviarse entre sí. Se aplican a la vista dinámica del sistema. Son de cuatro tipos:
  - *Diagramas de secuencia (Sequence diagrams)*, que subrayan el tipo y orden de los mensajes intercambiados durante la interacción.
  - *Diagramas de comunicación (Communication diagrams)*, que subrayan la organización estructural de los objetos que envían y reciben mensajes.
  - *Diagramas de tiempo (Timing diagrams)*, en los que se detallan las especificaciones temporales de los mensajes (el tiempo disponible para procesar o responder a un mensaje, cómo tratar la interrupciones externas durante la ejecución) y se usan para modelar sistemas de tiempo real.
  - *Diagramas de visión de conjunto de la interacción (Interaction overview diagrams)*, que son una especialización de los diagramas de actividad centrados en el flujo de control, donde los nodos son interacciones o usos de interacciones.

Modelo conceptual del UML (V)

- *Diagrama de caso de uso (Use case diagrams)*: se centra en los requerimientos funcionales del sistema.
  - Proporciona una visión independiente de la implementación de lo que el sistema debe hacer, enfocada en las necesidades del usuario y no en los detalles de realización.

Conceptos comunes en UML

- Existen elementos de notación de UML que se utilizan a través de todos los diagramas, como son: los clasificadores (*classifiers*), las notas (*notes*) y los estereotipos (*stereotypes*).
- Clasificador (*classifier*): es el elemento básico de modelado en UML. Representa un grupo de cosas con propiedades comunes.
  - La notación genérica para un clasificador es un rectángulo, que se puede dividir en compartimentos para mostrar información específica del clasificador, como operaciones, atributos o actividades de su estado.
  - Sin embargo, muchos clasificadores tienen símbolos propios para distinguirlos visualmente, como los estados, las actividades, los objetos, etc.
- Nota (*note*): permite añadir información a los diagramas.
  - Suelen usarse para mostrar comentarios adicionales. que no tienen notación propia o cuya representación podría incrementar la complejidad del diagrama.
  - Su símbolo es un rectángulo con una esquina doblada, y pueden aparecer aisladas en el diagrama o asociadas a un elemento específico usando una línea punteada.



Conceptos comunes en UML

- Estereotipo (*stereotype*): sirve para modificar el significado de un elemento y describe el papel que representa el elemento dentro del modelo.
  - Se usa para dar al lector del modelo una idea general del uso de un clasificador determinado.
  - Los estereotipos se suelen asociar con conceptos de implementación.
  - En general, se representan con la notación <<stereotype_name>> dentro del símbolo del elemento que modifican, aunque hay estereotipos que tienen un icono propio asociado.

Bibliografía

- '*Learning UML 2.0*' [recurso electrónico]. Kim Hamilton, Russell Miles. O'Reilly, 2006.
- '*UML 2.0 in a Nutshell*' [recurso electrónico]. Dan Pilone, Neil Pitman. O'Reilly, 2005.
- '*Unified Modeling Language: Superstructure*' (version 2.1.1). *http://www.uml.org*. Febrero 2007.