Universidad Autónoma del Estado de México

Unidad Académica Profesional Tianguistenco

Ingeniería en software

Unidad de aprendizaje:

Administración y Organización de Proyectos de Software

Profesor:

Jose Rafael Cruz Reyes

Alumno:

Andrés Alvir Guzmán

Fecha de entrega 5/03/23

En el capítulo número 5 del libro de Pressman podremos encontrar información de como es el modelado de un sistema que puedan desarrollarse mediante la ingeniería, así mismo los problemas que existen dentro del proceso del desarrollo de un proyecto y saber cuáles son los diferentes modelos de sistemas en los que podemos adaptar un proyecto, sin embargo no podemos iniciar a hablar del modelado de un sistema si no sabemos lo que es, por lo que Pressman primeramente comienza definiendo esta parte y nos dice que es es el proceso para desarrollar modelos abstractos de un sistema, donde cada modelo presenta una visión o perspectiva diferente de dicho sistema. En general, el modelado de sistemas se ha convertido en un medio para representar el sistema usando algún tipo de notación gráfica, que ahora casi siempre se basa en notaciones en el Lenguaje de Modelado Unificado (UML). Sin embargo, también es posible desarrollar modelos formales (matemáticos) de un sistema, generalmente como una especificación detallada del sistema. En este capítulo se estudia el modelado gráfico utilizando el UML, el modelado formal. Los modelos se usan durante el proceso de ingeniería de requerimientos para ayudar a derivar los requerimientos de un sistema, durante el proceso de diseño para describir el sistema a los ingenieros que implementan el sistema, y después de la implementación para documentar la estructura y la operación del sistema. El aspecto más importante de un modelo del sistema es que deja fuera los detalles. Un modelo es una abstracción del sistema a estudiar, y no una representación alternativa de dicho sistema. De manera ideal, una representación de un sistema debe mantener toda la información sobre la entidad a representar. Una abstracción simplifica y recoge deliberadamente las características más destacadas. Por ejemplo, en el muy improbable caso de que este libro se entregue por capítulos en un periódico, la presentación sería una abstracción de los puntos clave del libro. Si se tradujera del inglés al italiano, sería una representación alternativa. La intención del traductor sería mantener toda la información como se presenta en inglés. En una primera etapa en la especificación de un sistema, debe decidir sobre las fronteras del sistema. Esto implica trabajar con los participantes del sistema para determinar cuál funcionalidad se incluirá en el sistema y cuál la ofrece el entorno del sistema. Tal vez decida que el apoyo automatizado para algunos procesos empresariales deba implementarse, pero otros deben ser procesos manuales o soportados por sistemas diferentes. Debe buscar posibles traslapes en la funcionalidad con los sistemas existentes y determinar dónde tiene que implementarse nueva funcionalidad. Estas decisiones deben hacerse oportunamente durante el proceso, para limitar los costos del sistema, así como el tiempo necesario para comprender los requerimientos y el diseño del sistema. La definición de frontera de un sistema no es un juicio libre de valor. Las preocupaciones sociales y organizacionales pueden significar que la posición de la frontera de un sistema se determine considerando factores no técnicos. Por ejemplo, una frontera de sistema puede colocarse deliberadamente, de modo que todo el proceso de análisis se realice en un sitio; puede elegirse de forma que sea innecesario consultar a un administrador

particularmente difícil; puede situarse de manera que el costo del sistema aumente y la división de desarrollo del sistema deba, por lo tanto, expandirse al diseño y la implementación del sistema. Los modelos de caso de uso y los diagramas de secuencia presentan la interacción a diferentes niveles de detalle y, por lo tanto, es posible utilizarlos juntos. Los detalles de las interacciones que hay en un caso de uso de alto nivel se documentan en un diagrama de secuencia. El UML también incluye diagramas de comunicación usados para modelar interacciones. Aquí no se analiza esto, ya que se trata de representaciones alternativas de gráficos de secuencia. De hecho, algunas herramientas pueden generar un diagrama de comunicación a partir de un diagrama de secuencia. Los diagramas de caso de uso brindan un panorama bastante sencillo de una interacción, de modo que usted tiene que ofrecer más detalle para entender lo que está implicado. Este detalle puede ser una simple descripción textual, o una descripción estructurada en una tabla o un diagrama de secuencia, como se discute a continuación. Es posible elegir el formato más adecuado, dependiendo del caso de uso y del nivel de detalle que usted considere se requiera en el modelo. Para el autor, el formato más útil es un formato tabular estándar.

Los diagramas de secuencia en el UML se usan principalmente para modelar las interacciones entre los actores y los objetos en un sistema, así como las interacciones entre los objetos en sí. El UML tiene una amplia sintaxis para diagramas de secuencia, lo cual permite muchos tipos diferentes de interacción a modelar. Como aquí no hay espacio para cubrir todas las posibilidades, sólo nos enfocaremos en lo básico de este tipo de diagrama. Los objetos y actores que intervienen se mencionan a lo largo de la parte superior del diagrama, con una línea punteada que se dibuja verticalmente a partir de éstos. Las interacciones entre los objetos se indican con flechas dirigidas. El rectángulo sobre las líneas punteadas indica la línea de vida del objeto tratado (es decir, el tiempo que la instancia del objeto está involucrada en la computación). La secuencia de interacciones se lee de arriba abajo. Las anotaciones sobre las flechas señalan las llamadas a los objetos, sus parámetros y los valores que regresan. **Los modelos estructurales** de software muestran la organización de un sistema, en términos de los componentes que constituyen dicho sistema y sus relaciones. Los modelos estructurales son modelos estáticos, que muestran la estructura del diseño del sistema, o modelos dinámicos, que revelan la organización del sistema cuando se ejecuta. No son lo mismo: la organización dinámica de un sistema como un conjunto de hilos en interacción tiende a ser muy diferente de un modelo estático de componentes del sistema. Los modelos estructurales de un sistema se crean cuando se discute y diseña la arquitectura del sistema. El diseño arquitectónico es un tema particularmente importante en la ingeniería de software, y los diagramas UML de componente, de paquete y de implementación se utilizan cuando se presentan modelos arquitectónicos. En los capítulos 6, 18 y 19 se cubren diferentes aspectos de la arquitectura de software y del modelado arquitectónico. Esta sección se enfoca en el uso de diagramas de clase para modelar la estructura estática de las clases de objetos, en un sistema de software. Los **diagramas de clase** pueden

usarse cuando se desarrolla un modelo de sistema orientado a objetos para mostrar las clases en un sistema y las asociaciones entre dichas clases. De manera holgada, una clase de objeto se considera como una definición general de un tipo de objeto del sistema. Una asociación es un vínculo entre clases, que indica que hay una relación entre dicha clases. En consecuencia, cada clase puede tener algún conocimiento de esta clase asociada. Cuando se desarrollan modelos durante las primeras etapas del proceso de ingeniería de software, los objetos representan algo en el mundo real, como un paciente, una receta, un médico, etcétera. Conforme se desarrolla una implementación, por lo general se necesitan definir los objetos de implementación adicionales que se usan para dar la funcionalidad requerida del sistema. Aquí, el enfoque está sobre el modelado de objetos del mundo real, como parte de los requerimientos o los primeros procesos de diseño del software. En el modelado de sistemas, con frecuencia es útil examinar las clases en un sistema, con la finalidad de ver si hay ámbito para la generalización. Esto significa que la información común se mantendrá solamente en un lugar. Ésta es una buena práctica de diseño, pues significa que, si se proponen cambios, entonces no se tiene que buscar en todas las clases en el sistema, para observar si se ven afectadas por el cambio. En los lenguajes orientados a objetos, como Java, la generalización se implementa usando los mecanismos de herencia de clase construidos en el lenguaje. a ingeniería dirigida por modelo tiene sus raíces en la arquitectura dirigida por modelo (MDA, por las siglas de Model-Driven Architecture), que fue propuesta por el Object Management Group (OMG) en 2001 como un nuevo paradigma de desarrollo de software. La ingeniería dirigida por modelo y la arquitectura dirigida por modelo se ven normalmente iguales. Sin embargo, se considera que la MDE tiene un ámbito más amplio que la MDA. Como se estudia más adelante en esta sección, la MDA se enfoca en las etapas de diseño e implementación del desarrollo de software, mientras que la MDE se interesa por todos los aspectos del proceso de ingeniería de software. Por lo tanto, los temas como ingeniería de requerimientos basada en un modelo, procesos de software para desarrollo basado en un modelo, y pruebas basadas en un modelo son parte de MDE, pero no, en este momento, de la MDA.E xiste una relación difícil entre métodos ágiles y arquitectura dirigida por modelo. La noción de modelado frontal extenso contradice las ideas fundamentales del manifiesto ágil y se conjetura que pocos desarrolladores ágiles se sienten cómodos con la ingeniería dirigida por modelo. Los desarrolladores de MAD afirman que se tiene la intención de apoyar un enfoque iterativo para el desarrollo y, por lo tanto, puede usarse dentro de los métodos ágiles (Mellor et al., 2004). Si las transformaciones pueden automatizarse completamente y a partir de un PIM se genera un programa completo, entonces, en principio, MDA podría usarse en un proceso de desarrollo ágil, ya que no se requeriría codificación separada. Sin embargo, hasta donde se sabe, no hay herramientas de MDA que soporten prácticas como las pruebas de regresión y el desarrollo dirigido por pruebas.