

UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

Metodologías para Análisis y Diseño Orientado a Objetos
y MDA (Model Driven Architecture)



Trabajo de investigación presentado por

Byron Orlando Morales Sequen

Josué Rendón Estrada

Asucena Sarazua

Guatemala, Febrero 2009



Metodologías para Análisis y Diseño Orientado a Objetos y MDA (Model Driven Architecture) by Byron Orlando Morales Sequen, Josué Rendón Estrada & Asucena Sarazua is licensed under a Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 Guatemala License.

Based on a work at uvg.edu.gt.

INDICE

INDICE

PÁGINA:

INDICE.....	3
<i>INDICE</i>	4
INTRODUCCIÓN.....	5
I. INTRODUCCIÓN.....	6
CONTENIDO	7
I. METODOLOGÍA DE BOOCH:.....	8
A. Definición y Origen	8
B. Funcionamiento	8
C. Herramientas:.....	8
II. METODOLOGIA DE JACOBSON	10
A. Historía	10
B. Descripción.....	10
C. Funcionalidad	11
D. Herramientas	13
III. MODEL DRIVEN ARCHITECTURE (MDA):.....	13
A. Origen:.....	13
B. Definición:.....	13
C. Funcionalidad:	15
D. Utilidad:.....	16
E. Herramientas:.....	17
CONCLUSIONES.....	18
I. CONCLUSIONES	19
BIBLIOGRAFÍA.....	20
I. BIBLIOGRAFÍA	21

INTRODUCCIÓN

I.INTRODUCCIÓN

Hoy en día cualquier organización, comercial o no comercial, necesita de alguna forma para tener un control y orden de la información que maneja. Para eso se usan sistemas de software. Para que ese sistema sea efectivo, es necesario que la información sea accesible por cualquier miembro del equipo en cualquier lugar y momento. Eso quiere decir que el sistema deberá depender de otros factores como una red, infraestructura o tendencias tecnológicas.

El hecho que un sistema o aplicación no necesite integrarse con nuevas tecnologías o implementarse en otra infraestructura es un mito. Esa idea murió en el siglo pasado, ya que en estos tiempos, con la gran velocidad de cambios tecnológicos, los sistemas deben adaptarse a nuevos ambientes. Por tal razón se necesario crear e implementar modelos estándares de arquitectura de software que le permitan a un sistema evolucionar junto a su infraestructura, plataforma o ambiente. El Model Driven Architecture (MDA) es una propuesta de este tipo de modelos, y en el siguiente escrito se detalla su metodología y funcionalidad.

Basándonos en la idea que estamos en tiempos de tecnologías cambiantes y de comunicación global, necesitamos optimizar el análisis y diseño de los sistemas y aplicaciones. Se necesita crear software que pueda ser implementado en cualquier infraestructura y que pueda ser modificado con facilidad por cualquier persona en el mundo.

Por tal razón, se han creado varias metodologías para análisis y diseño orientado a objetos, las cuales estandarizan la forma de trabajar una en el diseño de un software. Estas metodologías permiten que cualquier persona pueda trabajar en su diseño. Algunas que podemos mencionar son la metodología de Booch, la de Coad y Yourdon, la de Rumbaugh,, la de Jacobson y la de Schaller y Mellor. En esta investigación se expone las metodologías de Booch y Jacobson.

CONTENIDO

I.METODOLOGÍA DE BOOCH:

A. Definición y Origen

La Metodología de Booch es una técnica usada en ingeniería de software. Es un lenguaje de modelado de objetos. Es una metodología ampliamente usada en el diseño de software orientado a objetos. Fue desarrollada por Grady Booch mientras trabajaba para Rational Software, la cual fue absorbida por IBM (Figueroa, 1999).

Los aspectos notables de la metodología de Booch son sus elementos gráficos orientados a objetos. Los aspectos metodológicos de la metodología de Booch fueron incorporados en varias metodologías y procesos, siendo la principal de ellas el Proceso Racional Unificado (RUP en inglés) (Donadello, 2006).

B. Funcionamiento

El método de Booch funciona de la siguiente forma (Winblad, 1993):

- Hacer la descripción en prosa del problema
- Identificar los posibles objetos del párrafo escrito
- Asociar atributos a los objetos identificados
- Identificar los métodos correspondientes a cada objeto
- Hacer la definición de las interfaces entre objetos

Del funcionamiento anterior es fácil deducir cuál es la principal utilizad de la metodología de Booch: identificar y definir objetos. En algunas ocasiones utiliza lenguaje gráfico para expresar esta definición de objetos (Winblad, 1993).

C.Herramientas:

Actualmente, se pueden utilizar diagramas:

- Diagrama de Clases. Consisten en un conjunto de clases y relaciones entre ellas. Los tipos de relaciones son asociaciones, contención, herencia, uso e instanciación.
- Especificación de Clases. Es usado para capturar toda la información importante acerca de una clase en formato texto.
- Diagrama de Categorías. Muestra clases agrupadas lógicamente bajo varias categorías
- Diagramas de transición de estados.
- Diagramas de Objetos. Muestra objetos en el sistema y su relación lógica. Pueden ser diagramas de escenario, donde se muestra cómo colaboran los objetos en cierta operación; o diagramas de instancia, que muestra la existencia de los objetos y las relaciones estructurales entre ellos.
- Diagramas de Tiempo. Aumenta un diagrama de objetos con información acerca de eventos externos y tiempo de llegada de los mensajes.
- Diagramas de módulos. Muestra la localización de objetos y clases en módulos del diseño físico de un sistema. Un diagrama de módulos representa parte o la totalidad de la arquitectura de módulos del sistema.
- Subsistemas. Un subsistema es una agrupación de módulos, útil en modelos de gran escala.
- Diagramas de procesos

En la actualidad para realizar estos diagramas se utilizan programas como DIA, VisualParadigm, Microsoft Visio entre otros.

II. METODOLOGIA DE JACOBSON

A.Historía

El desarrollo de UML comenzó a finales de 1994 cuando Grady Booch y Jim Rumbaugh de Rational Software Corporation empezaron a unificar sus métodos. A finales de 1995, Ivar Jacobson y su compañía Objectory se incorporaron a Rational en su unificación, aportando el método OOSE. La metodología de Jacobson es más centrada a usuario, ya que todo en su método se deriva de los escenarios de uso. UML se ha ido fomentando y aceptando como estándar desde el OMG que es también el origen de CORBA, el estándar líder en la industria para la programación de objetos distribuidos. En 1997 UML 1.1 fue aprobada por la OMG convirtiéndose en la notación estándar de facto para el análisis y el diseño orientado a objetos.

UML es el primer método en publicar una meta-modelo en su propia notación, incluyendo la notación para la mayoría de la información de requisitos, análisis y diseño. Se trata pues de una meta-modelo auto-referencial, cualquier lenguaje de modelado de propósito general debería ser capaz de modelarse a sí mismo (Morataya, 2007).

B.Descripción

La metodología de Jacobson se basa principalmente en diagramas de casos de uso y diagramas de interacción. Para encontrar una clase frontera, la cual modela la comunicación entre el o los alrededores del sistema y su parte interna, o simplemente sirve para ser usada como interfaz a otro sistema. A su vez busca también la clase entidad que es la que modela información y asocia comportamientos que generalmente son de larga duración. Y finalmente la clase de control la cual modela el comportamiento específico de uno o más casos de

uso; crea, inicializa y borra objetos controlados; controla la secuencia o coordina la ejecución de los objetos controlados (León, 2004).

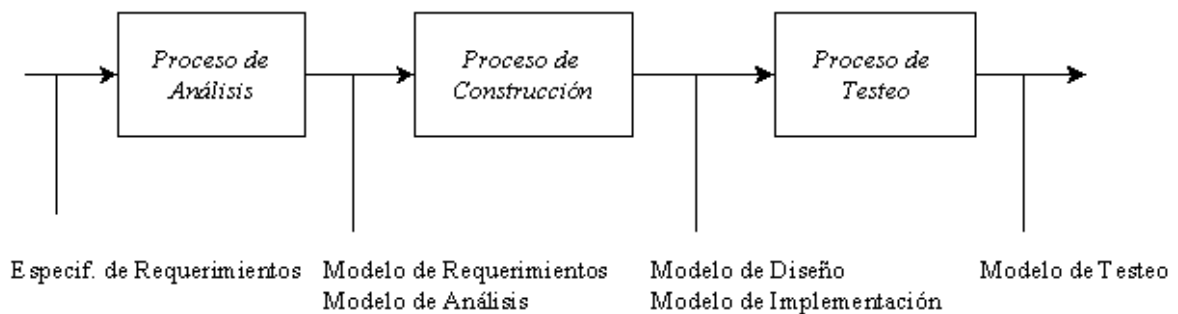
C.Funcionalidad

Esta metodología utiliza principalmente tres técnicas diferentes

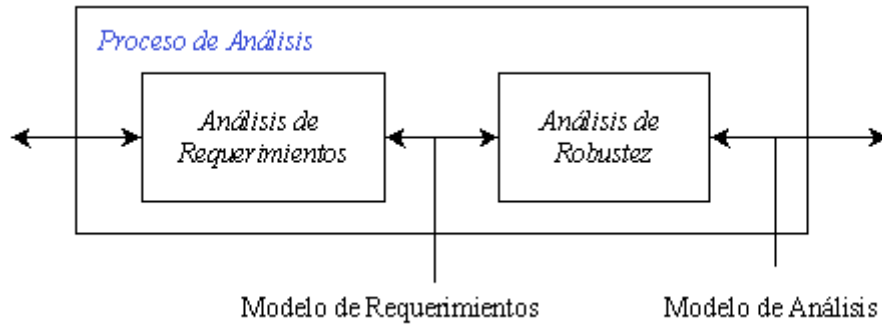
1.La programación orientada al objeto: De esta técnica utiliza los conceptos de encapsulación, herencia y relaciones principalmente entre las clases y casos.

2.El trazado conceptual: El cual se usa para crear los diferentes modelos del sistema u organización a ser analizado. Extendiéndolos con los conceptos orientados a objetos y con la posibilidad de modelar la conducta dinámica. Los mismos sirven para entender el sistema y obtener una arquitectura del sistema definida.

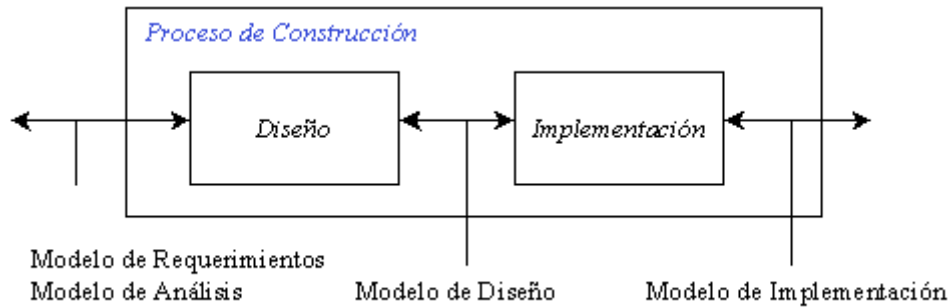
3. El plan de bloque: Modela los módulos con funcionalidades propias, que se conectan con las interfaces bien definidas. Este plan implica la mutabilidad mayor y mantención de software (Conallen, 1998).



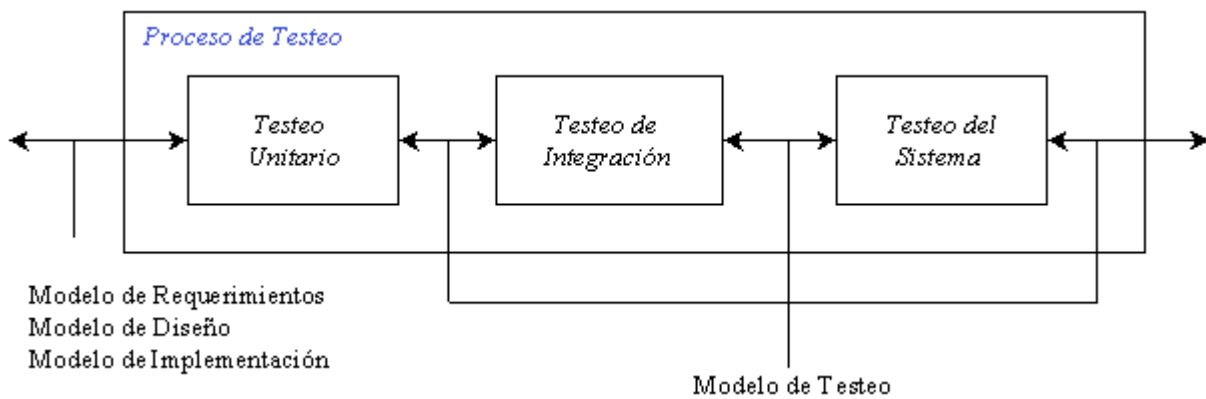
Gráfica 1. Funcionamiento de metodología de Jacobson parte 1
(Donadello, 2004)



Gráfica 2. Funcionamiento de metodología de Jacobson parte 2 (Donadello, 2004)



Gráfica 3. Funcionamiento de metodología de Jacobson parte 3 (Donadello, 2004)



Gráfica 4. Funcionamiento de metodología de Jacobson parte 4 (Donadello, 2004)

D.Herramientas

La metodología no está condicionada por la herramienta. Pero algunas de las herramientas que lo soportan son Dia, ArgoUml, Netbean, Visual Paradigm, PacestarUML , también Umbrello para Linux.

III.MODEL DRIVEN ARCHITECTURE (MDA):

A.Origen:

El *Model Driven Architecture* (MDA) fue propuesto y patrocinado en el 2001 por el *Object Management Group* (OMG), un grupo que se dedica al establecimiento de estándares en tecnologías orientadas a objetos. Es una organización no lucrativa conformada por más de 800 compañías, organizaciones e individuales, entre las cuales podemos mencionar a Hewlett-Packard, IBM y Apple. Tiene un largo historial en el establecimiento de estándares de tecnología e informática, tales como CORBA, UML o XMI (OMG, 2007).

B.Definición:

El MDA es un conjunto emergente de estándares y tecnologías enfocadas en un estilo particular de desarrollo de software

Surgió por la necesidad de crear un modelo de diseño de software que facilitara la separación del diseño y la arquitectura para que pudieran ser alterados independientemente.

Se basa en la idea de separar las especificaciones de la operación de un sistema de los detalles de la manera que el sistema usa las capacidades de su plataforma. Provee herramientas para:

- Independizar el sistema de la plataforma que lo soporta
- Especificar plataformas
- Escoger una plataforma específica para un sistema
- Transformar las especificaciones del sistema para que sean compatibles con cualquier plataforma en particular.

Los objetivos principales de este modelo son: portabilidad, interoperabilidad y reutilización a través de separar los asuntos arquitectónicos (Miller, 2003).

1. Conceptos Básicos:

El MDA se basa en los siguiente conceptos:

- Sistema: Puede incluir lo que sea, un programa, un sistema computacional o una combinación de varios sistemas.
- Modelo: Es la descripción de un sistema y su ambiente para una razón específica.
- Arquitectura: Es la especificación de las partes y conectores del sistema y las reglas de interacción de las partes que usan conectores.
- Punto de vista: Es una técnica de abstracción que usa un seleccionado grupo de conceptos arquitectónicos y reglas estructurales, con el fin de enfocar determinado asunto en un sistema.
- Plataforma: Grupo de sub-sistemas y tecnologías que proveen un conjunto de funcionalidades a través de interfaces y patrones específicos. Cualquier aplicación soportada por una plataforma puede ser usada sin importar los detalles de su funcionalidad (The ATHENA Consortium, 2008)

C.Funcionalidad:

Para lograr estos objetivos, la OMG a definido una serie específica de puntos de vista y modelos que proveen una construcción conceptual de la MDA.

1.Puntos de Vista:

- Computation Independent Viewpoint*: Se enfoca en el ambiente y requerimientos del sistema. Los detalles de la estructura y procesos del sistema estan ocultos o indeterminados.
- Platform Independent Viewpoint*: Se enfoca en la operación de un sistema pero oculta los detalles necesarios para una plataforma en particular. Este punto de vista muestra que parte de las especificaciones completas no cambian cuando son usadas en diferentes plataformas.
- Platform Specific Viewpoint*: Combina el Platform Independent Viewpoint con un enfoque en los detalles del uso de una plataforma especifica por un sistema (Miller, 2003).

2.Módelos:

- Computation Independent Model (CIM)*: Es la observación de un sistema desde el punto de vista de computación independiente (Computation Independent Viewpoint). Se enfoca en el ambiente y requerimientos del sistema. Oculta los detalles de su estructura y plataforma.
- Platform Independent Model (PIM)*: Es la observación de un sistema desde el punto de vista de plataforma independiente (Platform Independent Viewpoint). Se enfoca en las operaciones del sistema y oculta los detalles de la plataforma. Muestra independencia de

plataforma y puede ser utilizada en cualquiera de ellas. Almacena toda la información para describir el comportamiento de un sistema en una plataforma.

- *Platform Specific Model (PSM)*: Es la observación de un sistema desde el punto de vista de plataforma específica (Platform Specific Viewpoint). Combina las especificaciones de PIM con los detalles que especifican como un sistema se comporta y usa una determinada plataforma (The ATHENA Consortium, 2008).

D.Utilidad:

La utilidad del MDA es permitir a sistemas, aplicaciones o modelos de datos tener gran flexibilidad en:

- Implementación: nueva implementación en infraestructura. Las nuevas tendencias tecnológicas pueden ser implementadas en los diseños existentes.
- Integración: Se puede automatizar los puentes de integración de producción de datos y las la conexiones de nuevas integraciones de infraestructura.
- Mantenimiento: Da acceso directo los desarrolladoras a que cambien cualquier especificación del sistema.
- Chequeo y simulación: Cualquier sistema puede ser probado con cualquier infraestructura y puede simular el comportamiento del sistema.

A raíz de esto, la vida útil de un sistema podría ascender a 20 años (Miller, 2003).

E.Herramientas:

Actualmente, se está desarrollando un amplio conjunto de herramientas que empleen MDA. Estas permiten la especificación rudimentaria de arquitecturas. Algunos ejemplos son:

- seleccionar una de las arquitecturas de referencia tales como Java EE o Microsoft .NET.
- Especificar la arquitectura a un nivel de mayor detalle

CONCLUSIONES

I.CONCLUSIONES

Optar por una metodología a seguir para el desarrollo de un proyecto es de gran beneficio pues esto nos brinda un mayor control de que es lo que se va a hacer y a su vez nos hace que realicemos todo paso a paso, y analicemos mejor que es lo que necesitamos, con qué recursos contamos y que esperamos obtener.

La metodología de Booch es muy útil para modelar los objetos que son necesarios en un sistema, por lo tanto, se puede utilizar para hacer los diseños básicos de sistemas que no sean tan grandes y en los cuales la identificación de todos los objetos es deducible de por la simple observación de lo que el pequeño sistema requiera. Esta metodología puede ser de mucho provecho para los laboratorios y la definición de los objetos que utilizará el proyecto para J2ME porque no son sistemas muy complejos

El Model Driven Architecture (MDA) es una excelente modelo para volver una sistema mas eficiente y duradero. Es muy importante implementar este tipo de modelos porque hoy en día los cambios de infraestructura y tecnología son evidentes. El MDA un trabajo que todavía esta en desarrollo y su definición todavía esta evolucionado. Actualmente son pocas las compañías, organizaciones o individuos que implementan este modelo en el diseño de sus sistemas,

BIBLIOGRAFÍA

I.BIBLIOGRAFÍA

Conallen, J. (1998). Modeling Web Application Design with UML. Recuperado el 20 de febrero de 2009. URL: www.conallen.com

Donadello, D. (2006, 26 de Marzo). Unified Modeling Language UML. Universidad de Belgrano, Argentina. Recuperado el 19 de febrero de 2009. URL: <http://tinyurl.com/jacobson2>

Figuroa, P. (1999). Object Oriented Design, por Grady Booch. Recuperado el 20 de febrero de 2009. URL: <http://www.cs.ualberta.ca/~pfiguero/soo/metod/ood.html>

León, R. (2004, 12 de septiembre). Resumen Introducción al Lenguaje UML. Departamento de Ciencias de la Computación, Escuela de Ingeniería, Pontificia Universidad Católica de Chile. Recuperado el el 20 de febrero de 2009. URL: <http://tinyurl.com/jacobson1>

Morataya, M. Citado el (2007, 28 de febrero). UML (Lenguaje Unificado Modelado. Escuela de Ciencias y Sistemas, Facultad de Ingeniería, Universidad de San Carlos. Guatemala. Recuperado el 20 de febrero de 2009. URL: www.itforcegt.org/200714355/tarea2_200714355.doc

Miller, J. Mukerji, J. (2003, 12 de junio). MDA Guide Version 1.0.1. Object Management Group. Recuperado el 19 de febrero de 2009. URL: <http://www.omg.org/docs/omg/03-06-01.pdf>

OMG (2007, 9 de septiembre). About the Object Management Group™ (OMG™). Object Management Group, Inc. Recuperado el 21 de febrero de 2009. URL: <http://www.omg.org/gettingstarted/gettingstartedindex.htm>

The ATHENA Consortium. (2008, 22 de febrero). What is MDA? Recuperado el 19 de febrero de 2009. URL: <http://www.modelbased.net/mdi/mda/mda.html>

Winblad, A. (1993). Software orientado a objetos. Recuperado el 20 de febrero de 2009. URL: <http://tinyurl.com/c8799f>