Mengenal Model Driven Architecture (MDA) sebagai Alternatif Pendekatan pada Pengembangan Perangkat Lunak

Falahah
Information System Research Group, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung,
e-mail : andromeda1268@yahoo.com
Iwan Rijayana
Jurusan Teknik Informatika, Fakultas Teknik, Universitas Widyatama Bandung.
e-mail : rijayana@widyatama.ac.id

ABSTRAK

Model Driven Architecture (MDA) adalah sebuah pendekatan pengembangan perangkat lunak yang dikembangkan oleh Object Management Group untuk mendukung portabilitas dan fleksibilitas dalam pengembangan. Konsep utama MDA adalah proses pengembangan yang dikendalikan oleh model dan fleksibilitas transformasi antar model sehingga sebuah model dapat dirancang dan diterapkan untuk berbagai platform tanpa harus melakukan perubahan yang signifikan pada model utamanya. MDA menyajikan 3 komponen model utama yaitu Computational Independent Model (CIM) yang mewakili proses bisnis, kebutuhan sistem dan kondisi lingkungan dimana sistem akan diterapkan, Platform Independent Model (PIM) yang merupakan model spesifikasi sistem yang tidak bergantung pada platform tertentu, dan Platform Spesific Model (PSM) yang merupakan spesifikasi model berdasarkan spesifikasi platform tertentu. Perbedaan utama antara pendekatan MDA dengan pendekatan yang lain adalah bahwa satu PIM dapat saja ditransformasi menjadi beberapa PSM dan setiap CIM, PIM dan PSM harus dapat ditelusuri keterkaitan elemen-elemennya dan perubahan-perubahannya. Proses transformasi ini juga harus disertai dengan alat bantu (tools) yang dapat mengotomatisasi proses tersebut dan menjaga kesinambungan model.

Kata kunci : MDA, Model, Pengembangan Perangkat Lunak, transformasi model, otomatisasi.

I. Pendahuluan.

Model Driven Architecture (MDA) lahir dari sejarah panjang Object Management Group (OMG) dalam usahanya membuat standar di komputasi dalam dua dekade terakhir. Beberapa standar lain yang sudah dihasilkan oleh OMG meliputi spesifikasi sistem dan interoperasinya seperti Common Object Request Broker Architecture (CORBA), OMG Interface Definition Language (IDL), Internet Inter-ORB Protocol (IIOP), Unified Modeling Language (UML), Meta Object Facility (MOF), XML Metadata Interchange (XMI), Common Warehouse Model (CWM), and Object Management Architecture (OMA).

Tahun 2001, OMG mulai mengadopsi framework kedua yaitu Model Driven Architecture (MDA). MDA tidak seperti OMA dan CORBA yang merupakan framework untuk implementasi sistem terdistribusi. MDA adalah sebuah pendekatan dalam penggunaan model pada pengembangan software.

OMG mempromosikan MDA sebagai pendekatan yang fleksibel dalam proses pengembangan sistem dan menjadikannya sebagai kerangka kerja interoperabilitas yang komprehensif dalam mendefinisikan sistem yang saling terkait.

MDA menyediakan pendekatan, dan merupakan alat yang dapat digunakan untuk:

- Melakukan spesifikasi sistem yang tidak bergantung (independent) dari platform yang mendukungnya.
- Melakukan spesifikasi platform
- Memilih platform tertentu untuk sistem
- Mentransformasi spesifikasi sistem ke salah satu platform tertentu.

Tiga tujuan utama MDA adalah portabilitas, interoperabilitas, dan reusabilitas melalui pemisahan arsitektur berdasarkan sudut pandang masing-masing

II. Konsep-konsep dasar

Prinsip dasar yang mendasari MDA adalah :

- Model yang dinyatakan dalam notasi yang sudah didefinisikan dengan jelas merupakan hal penting dalam memahami sistem untuk solusi skala enterprise.
- Pembangunan sistem dapat diorganisasi berdasarkan sekumpulan model dengan melakukan serangkaian transformasi antar model, mengorganisasikannya dalam kerangka kerja arsitektural yang terdiri atas lapisan dan transformasinya.
- Landasan formal untuk menggambarkan model dalam sebuah kumpulan metamodel, yang dapat memfasilitasi integrasi dan transformasi di antara model, dan sebagai landasan otomatisasi melalui alat bantu.
- Penerimaan dan adopsi yang luas terhadap pendekatan berbasis model membutuhkan standar industri agar dapat diterima oleh konsumen dan membuka kompetisi di antara vendor.

III. Beberapa Istilah pada MDA

Sistem: Pada MDA, sistem dapat terdiri dari berbagai komponen, misalnya berupa program, sistem komputer tunggal, kombinasi yang merupakan bagian dari sistem lain, kombinasi dari beberapa sistem, dan masing-masing dalam kontrol yang terpisah. Sebagian besar pembahasan mengacu pada software di dalam sistem.

Model sebuah sistem adalah deskripsi atau spesifikasi suatu sistem dan lingkungannya untuk tujuan tertentu. Model sering dinyatakan sebagai kombinasi gambar dan tulisan. Tulisan dapat dalam bentuk bahasa pemodelan maupun bahasa biasa.

Model-Driven : MDA adalah pendekatan dalam pengembangan sistem, yang meningkatkan kekuatan model dalam prosesnya. Disebut model-driven karena pendekatan ini menyediakan fasilitas penggunaan model untuk memahami, merancang, menerapkan, mengoperasikan, memelihara dan memodifikasi sistem.

Viewpoint sebuah sistem adalah teknik abstraksi dengan menggunakan sekumpulan konsep arsitektur dan aturan penstrukturan, agar dapat lebih memfokuskan ke tujuan tertentu.

MDA menggunakan 3 viewpoint terhadap sistem yaitu *viewpoint computation independent, platform independent,* dan *platform spesific.*

Platform : adalah sekumpulan sub sistem dan teknologi yang dapat menyediakan sekumpulan fungsionalitas melalui antarmuka dan pola penggunaan tertentu, sehingga semua aplikasi yang didukung oleh platform tersebut dapat digunakan tanpa harus memperhatikan bagaimana fungsionalitas tersebut disediakan oleh platformnya.

Aplikasi : istilah aplikasi pada MDA mengacu pada fungsionalitas yang akan dibangun. Sistem dinyatakan dalam satu atau lebih aplikasi yang didukung oleh satu atau lebih platform.

Platform Independence : adalah sebuah kualitas yang ingin dihasilkan oleh model. Yaitu kualitas bahwa model yang dihasilkan tidak bergantung pada platform tipe tertentu.

Model platform : menyediakan sekumpulan konsep teknikal, yang menyatakan bagian-bagian yang berbeda yang membangun platform dan layanan yang disediakan oleh platform tersebut. Model platform juga menspesifikasikan kebutuhan koneksi dan penggunaan bagian-bagianplatform tersebut dan koneksi antara aplikasi dengan platform.

Transformasi model : adalah proses untuk mengkonversikan model dari satu model ke model lain pada sistem yang sama

IV. ViewPoints dan Model utama MDA

- *Computation Independent Viewpoint (CIV)*
  CIV berfokus pada lingkungan sistem dan kebutuhan sistem yang meliputi rincian struktur dan proses yang dilakukan oleh sistem yang tersembunyi atau belum diketahui.
- *Platform Independent Viewpoint (PIV)*
  PIV berfokus pada operasi sebuah sistem dan menunukkan bagaimana bagian dari spesifikasi lengkap tidak berubah akibat perubahan platform.
- *Platform Specific Viewpoint(PSV)*
  Merupakan kombinasi antara PIV dengan fokus tambahan agar dapat menggambarkan penggunaan detil platform tertentu pada sebuah sistem.

MDA terdiri atas 3 model utama yaitu :

Computation Independent Model (CIM)

CIM adalah sebuah view dari sistem berdasarkan CIV. CIM tidak menunjukkan struktur detil sistem dan sering disebut dengan model domain. CIM berperan sebagai penghubung antara pihak yang paham dengan domain dan kebutuhannya dan dengan pihak yang paham dengan proses desain dan konstruksi software.

CIM menggambarkan kebutuhan sistem, situasi dimana sistem akan digunakan dan menggambarkan secara persis apa yang diharapkan dapat dilakukan oleh sistem. Sebagai model, CIM sering juga disebut dengan model domain atau model bisnis dan dapat saja tidak menampilkan semua informasi mengenai penggunaan sistem pemrosesan data otomatis. Biasanya model ini dibuat tidak bergantung pada bagaimana sistem ini diimplementasikan.

Pada MDA, spesifikasi CIM kebutuhan sebuah sistem harus dapat ditelusuri menjadi PIM dan PSM yang menerapkan CIM tersebut dan sebaliknya. CIM dapat terdiri atas dua model UML dari sudut pandang informasi dan enterprise. CIM juga dapat terdiri atas beberapa model yang didetilkan atau befokus pada sudut pandang tertentu.

Platform Independent Model (PIM)

PIM adalah sebuah view berdasarkan PIV. PIM menyatakan model yang dalam derajat tertentu tidak bergantung pada sebuah platform sehingga dapat diterapkan pada berbagai platform yang sejenis.

PIM menggambarkan sistem tetapi tidak menunjukkan secara detail penggunaannya pada platform tertentu. PIM dapat terdiri atas spesifikasi informasi, enterprises dan komputasi dari sudut ODP (open distributed processing). Struktur model informasinya dapat saja berbeda dengan struktur model informasi pada CIM untuk sistem yang sama.

Platform Specific Model (PSM)

PSM adalah sebuah view dari sistem berdasarkan PSV. PSM merupakan kombinasi PIM dengan informasi tambahan mengenai bagaimana sistem digunakan pada platform tertentu.

V. Transformasi Model

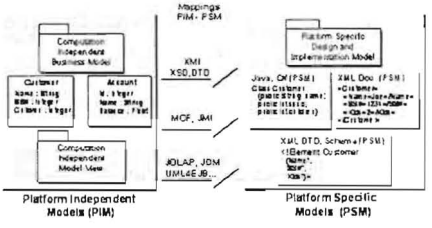
Sebuah tool MDA dapat mendukung transformasi model dalam beberapa tahap, mulai dari model analisis sampai menjadi kode yang siap dijalankan (executable).

Pembedaan antara berbagai jenis model memungkinkan kita untuk berfikir bahwa pengembangan sistem dan softwere sebagai sederetan penyempurnaan antara berbagai representasi model. Model ini dan penyempurnaannya merupakan bagian kritis bagi metodologi pengembangan pada situasi yang meliputi penyempurnaan antar model merepresentasikan aspek yang berbeda dari sebuah sistem, menambah rincian terhadap model atau konversi antara model yang berbeda.

Dengan menganggap bahwa pengembangan software dan system sebagai sekumpulan penyempurnaan model, transformasi antar model menjadi elemen utama pada proses pengembangan. Hal ini penting karena pekerjaan besar terjadi dalam proses definisi transformasi ini dan memerlukan pengetahuan baik dari segi domain bisnis maupun teknologi nya sendiri.

Pada MDA, 'platform' hanya berarti berdasarkan sudut pandang tertentu, dengan kata lain, PIM bagi seseorang dapat saja menjadi PSM bagi orang lain. Misalnya, sebuah model dapat saja menjadi PIM bagi pemilihan peralatan komunikasi antara jika model ini tidak menjelaskan pilihan teknologi tertentu.

Tetapi, jika kemudian diambil keputusan untuk menggunakan middleware tertentu misalnya CORBA, model ini ditransformasi menjadi PSM yang spesifik terhadap CORBA. Model ini tetap dapat menjadi PIM bagi pemilihan ORB – dan dari sudut pandang pemilihan sistem operasi dan hardware. Proses ini dapat dilihat pada gambar berikut :



Gambar 1. Proses Transformasi PIM menjadi PSM

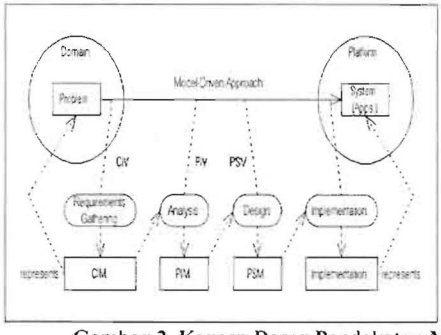
Contoh PIM sederhana pada gambar 1 menyatakan dua obyek yaitu customer dan account. Pada tingkatan ini, model menggambarkan karakteristik penting domain dalam bentuk kelas dan atributnya, tetapi tidak menggambarkan pilihan platform spesifik mengenai teknologi yang akan digunakan untuk mewujudkan kelas tersebut.

Pada gambar tersebut dilakukan 3 pemetaan spesifik atau transformasi, yang mendefinisikan PSM dan standar yang digunakan untuk melakukan pemetaan tersebut. Misalnya, satu pendekatan yaitu mentransformasika PSM yang dinyatakan dalam UML menjadi format XMI, mengguankan definisi standar baik yang dinyatakan dalam bentuk XML Schema Definition (XSD) atau document type definition (DTD). Hal ini dapat dijadikan input untuk menghasilkan kode melalui alat bantu yang mampu menghasilkan interface dalam bentuk Java untuk setiap class yang didefinisikan pada UML.

Biasanya, sekumpulan aturan dibangun bersama alat bantu penghasil kode untuk menjalankan transformasi. Tentunya alat bantu kode ini memungkinkan aturan-aturan tersebut didefinisikan sebagai template dalam bahasa script tertentu.

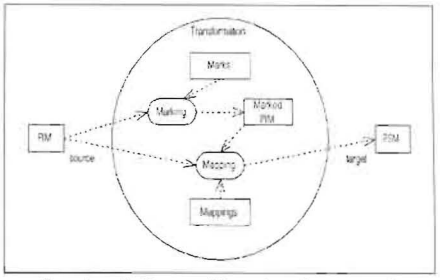
VI. Implementasi Pendekatan MDA pada SDLC.

Pendekatan Model Driven berfokus pada pemodelan sistem yang meliputi pemahaman, perancangan, pengembangan, implementasi, pengoperasian, pemeliharaan dan modifikasi sistem. Jika dikaitkan dengan siklus hidup pengembangan sistem maka pendekatan MDA adalah seperti pada gambar berikut :



Gambar 2. Konsep Dasar Pendekatan MDA

Mapping dan Transformasi



Gambar 3. Proses Mapping dan transformasi

Mapping

Sebuah pemetaan (mapping) adalah sebuah spesifikasi (atau spesifikasi transformasi), yang meliputi aturan dan informasi lain, untuk mentrasnformasi sebuah PIM menjadi sebuah PSM untuk platform tertentu. MDA mendefinisikan jenis-jenis mapping sebagai berikut :

- Model mapping mendefinisikan mapping berdasarkan jenis elemen model. Mapping ini mendefinisikan aturan bagiaman elemen-elemen yang berbeda pada PIM ditransformasi ke jenis elemen yang berbeda pada PSM. Misalnya, jika elemen PIM adalah sebuah entitas, maka akan ditransformasi menjadi entitas Enterprise Java Bean (EJB) atau sebuah class Java pada platform Java.
- Model instance mapping mendefinisikan bagaimana elemen model tertentu ditransformasikan dengan cara tertentu menggunakan mark (tanda). Elemen PIM diberi tanda untuk menunjukkan bagaimana elemen tersebut akan ditransformasikan. Sebuah elemen PIM dapat diberi tanda berkali-kali untuk mapping yang berbeda-beda.

Template mendefinisikan satu jenis transformasi tertentu menggunakan model yang diberi parameter atau pola elemen model yang digunakan sebagai aturan dalam pemetaan tipe model dan mapping instance model.

Proses mapping ini dapat dilihat pada gambar 3 di atas, yang menunjukkan tanda yang digunakan untuk menandai sebuah PIM dan menggunakan PIM yang sudah bertanda ini untuk menghasilkan sebuah PSM.

Transformasi

Transformasi model adala proses untuk mengkonversi sebuah PIM, dikombinasikan dengan informasi lain, untuk menghasilkan PSM. MDA mendefinisikan jenis-jenis transformasi berikut berdasarkan jenis mappingnya:

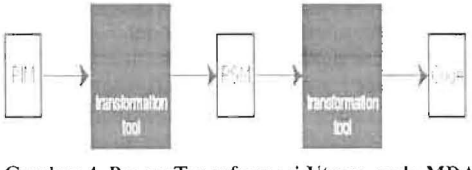
- Transformasi untuk model mapping adalah proses mengkonversi sebuah PIM untuk menghasilkan PSM dengan mengikuti mapping yang telah ditetapkan.
- Transformasi untuk model instance mapping adalah proses konversi sebuah PIM yang telah diberi tanda (marked) untuk menghasilkan PSM dengan mengikuti mapping yang telah ditetapkan.

Sebuah PSM dapat berupa implementasi, menyediakan informasi untuk membangun sistem secara manual, atau dapat berfungsi sebagai PIM untuk transformasi ke PSM lainnya. Misalnya, sebuah PIM untuk sistem penjualan dapat diberi tanda (marked) dan ditransformasi untuk menghasilkan sebuah PSM pada platform Java dan PSM untuk platform .Net.

Otomatisasi Tahapan Transformasi

Pada metoda yang tradisional, transformasi dari satu model ke model lain, atau dari sebuah model ke kode, dilakukan secara manual. Banyak alat bantu yang dapat membuat kode dari sebuah model, tetapi biasanya hanya dapat menghasilkan beberapa template kode yang masih harus dilengkapi secara manual.

Transformasi pada MDA dilakukan dengan alat bantu khusus dan transformasi dari PIM menjadi PSM dan dari PSM menjadi kode dilakukan secara otomatis. Proses otomatis ini menghemat waktu dalam mewujudkan sebuah model, misalnya membangun sebuah database untuk sebuah desain PIM tingkat atas, membangun komponen model COM, dan lain-lain. Proses transformasi otomatis ini digambarkan pada diagram berikut :



Gambar 4. Proses Transformasi Utama pada MDA

VII. Keuntungan MDA

Produktivitas

Pada MDA, focus pengembang bergeser menjadi bagaimana membuat PIM. PSM yang diperlukan dapat dihasilkan dengan mentransformasi PIM menjadi PSM, meskipun proses transformasi ini harus didefinisikan dengan jelas. Tetapi transformasi cukup didefinisikan satu kali dan kemudian diaplikasikan pada beberapa pengembangan sistem lainnya. Dengan berfokus pada PIM maka proses perancangan sistem tidak terlalu terganggu pada aspek teknikal yang biasanya sangat bergantung pada platform yang dipilih. Aspek teknikal ini baru ditambahkan kemudian ketika melakukan transformasi. Hal ini dapat meningkatkan produktivitas dengan dua cara yaitu :

Pengembangan PIM memerlukan waktu yang lebih singkat karena rincian platform yang spesifik tidak perlu dirancang atau dituliskan, tetapi akan diselesaikan pada proses pendefinisian transformasi. Pada tingkatan PSM dan kode, kode yang harus ditambahkan relatif sedikit karena sebagian besar kode sudah dibuat ketika merealisasikan PIM.

Fakta bahwa focus pada PIM membuat pengembang lebih berfokus pada penyelesaian masalah bisnis, sehingga diharapkan dapat menghasilkan sistem yang sesuai dengan kebutuhan user.

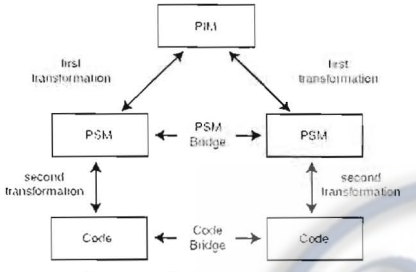
Produktivitas tersebut hanya dapat diperoleh dengan penggunaan alat bantu yang berkemampuan penuh untuk secara otomatis membuat PSM dari PIM, sehingga banyak informasi yang harus ditambahkan ketika membuat PIM.

Portabilitas

Pada MDA, portabilitas dicapai dengan memfokuskan pengembangan PIM, yang bersifat tidak bergantung pada platform. PIM yang sama dapat ditransformasi secara otomatis menjadi beberapa PSM untuk platform yang berbeda. Semua yang dispesifikasikan pada tingkatan PIM kemudian menjadi bersifat portable. Portabilitas yang dapat dicapai tergantung pada alat transformasi otomatis yang tersedia.

Interoperabilitas

Seperti terlihat pada gambar berikut, beberapa PSM dapat dibuat dari satu PIM dan PSM tersebut dapat saling terkait. Pada MDA hubungan ini disebut dengan bridges. Ketika PSM dibuat untuk satu platform tertentu maka PSM ini tidak dapat berkomunikasi dengan PSM untuk platform yang lain, sehingga kita harus mentransformasikan konsep dari satu platform ke platform lain. MDA menyelesaikan masalah ini dengan menambahkan 'bridges' antar platform tersebut.



Gambar 5. Interoperabilitas pada MDA menggunakan bridges

Jika kita dapat mentransformasi PIM menjadi dua PSM pada dua platform yang berbeda, semua informasi yang diperlukan untuk menjembatani dua PSM tersebut akan tersedia. Untuk setiap elemen pada satu PSM akan diketahui elemen asalnya pada PIM, dan dari elemen PIM ini dapat diketahui elemen pada PSM yang lainnya, sehingga keterkaitan masing-masing elemen pada dua PSM tersebut dapat diketahui. Misalnya, salah satu PSM yang dihasilkan berupa kode Java dan PSM yang lainnya adalah model database relasional. Untuk satu elemen customer pada PIM, kita mengetahui kelasnya pada Java, juga mengetahui tabel untuk meyimpan elemen customer tersebut. Membangun jembatan antara obyek Java dengan tabel pada database relatif mudah. Untuk menyimpan sebuah obyek, mula-mula akan dicari data pada obyek Java dan kemudian menyimpannya pada tabel customer. Interoperabilitas hanya dapat diwujudkan melaui alat yang tidak hanya membuat PSM tetapi juga jembatan antara PSM tersebut dan kemungkinkan jembatan terhadap platform lain.

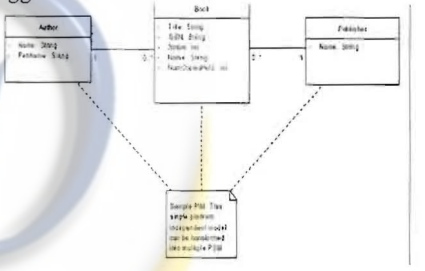
Maintenance and Documentation

PIM yang dihasilkan dapat berfungsi sebagai sebuah dokumentasi tingkat atas yang diperlukan pada pengembangan semua sistem software. Perbedaan khusus pada pendekatan MDA adalah bahwa PIM tidak ditinggalkan begitu saja setelah dibuat. Perubahan pada sistem biasanya dilakukan dengan cara mengubah PIM dan kemudian membuat kembali PSM dan kodenya. Secara praktis saat ini, banyak perubahan dibuat pada PSM dan kode dibuat ulang dari PSM ini. Alat bantu yang baik harus dapat memelihara hubungan antara PIM dan PSM dan mendokumentasikan perubahan pada PSM. Perubahan pada PSM kemudian harus dapat tercermina dan PIM sehingga dokumentasi tingkat atas tetap dijaga actual.

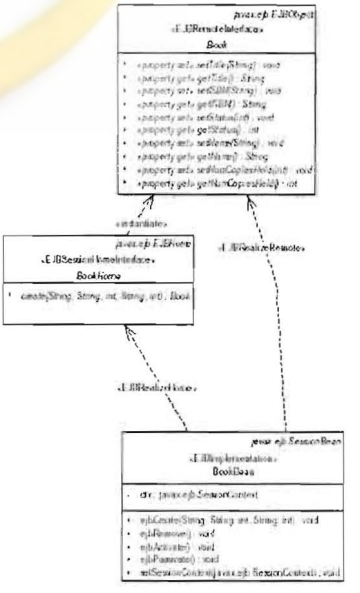
VIII. Alat Bantu

Peranan Alat Bantu pada pendekatan MDA dapat digambarkan pada ilustrasi berikut yang menggunakan "Enterprise Architect" dari www.sparxsystem.com.au.

Penggambaran PIM



Hasil Transformasi menjadi EJB Class (bean)



Selanjutnya Bean tadi dapat dieksekusi menjadi sebuah kode yang tetap terintegrasi dengan Bean yang dihasilkan.

Daftar Pustaka

[1] www.omg.org/mda

[2] Alhir, Sinan Si: "***Understanding the Model Driven Architecture***", www.methodsandtools.com/archive

[3] Brown, Alan, "***An Introduction to Model Driven Architecture"*** www.ibm.com/developerworks/rational/3100.html, 2004

[4] OMG " ***MDA Guide version1.0.1***" , www.omg.org , 2003

[5] Sparks, Geoffrey, "***MDA Overview***", http://www.sparxsystems.com