Apresentação do Capítulo 4 – MDA (Model-Driven Archtecture)

ALUNO: DOMENICO SCHETTINI FILHO

NÚMERO USP: 8429016

Definição de MDA

- OMG (Object Management Group) propôs uma aplicação abrangente das práticas do MDE para o desenvolvimento de sistemas. Essa proposição foi denominada MDA.
- MDA é um bom exemplo de framework MDE por duas razões principais: primeiro, MDA é um exemplo perfeito de introdução dos conceitos de MDE, como por exemplo fases padronizadas do processo de desenvolvimento de software como análise, design, e implementação; segundo, dada a importância da OMG na indústria de software, MDA é frequentemente utilizado como framework de referência utilizado por diversas organizações.

Princípios utilizados pela OMG para definir o framework de MDA

- Modelos devem ser expressos em uma notação bem definida, de modo a permitir uma comunicação efetiva e compreensão dos sistemas de descrição para soluções em escalas empresariais;
- Especificações de sistemas devem ser organizadas em torno de uma série de modelos e associados a transformações de implementações e de mapeamentos, e relações entre modelos. Tudo isto permite um design organizado baseado em um framework multicamadas e multi-perspectiva;
- Modelos devem ser construídos seguindo a tendência de uma série de metamodelos, facilitando desta maneira integrações e transformações significativas entre modelos e automações através de ferramentas;
- Este framework de modelos deve aumentar a aceitação e a adoção da abordagem MDE e alimentar a competição entre os vendedores de ferramentas.

MDA – Definições e Suposições

Toda a infraestrutura de MDA é baseada em núcleo de poucas definições e suposições. Os principais elementos de interesse para MDA são os seguintes:

- Sistema: O tópico de qualquer especificação MDA. Ele pode ser um programa, um sistema de computação simples, uma combinação de partes de diferentes sistemas, ou uma federação de sistemas;
- Espaço do Problema (ou Domínio): O contexto ou ambiente onde o sistema opera;
- Espaço de Solução: O espectro de possíveis soluções que satisfaçam os requerimentos do sistema;
- Modelo: Qualquer representação do sistema e/ou de seu ambiente;
- Arquitetura: A especificação de partes e de conectores do sistema, e de regras para interação de partes do sistema usando os conectores;
- Plataforma: Um conjunto de subsistemas e tecnologias que fornecem uma série de funcionalidades orientadas para atingir um objetivo específico.

MDA – Definições e Suposições

- Ponto de Vista: Uma descrição de um sistema que foca em um ou mais interesses particulares;
- Visão: Um modelo de sistema visto sob um ponto de vista específico;
- Transformação: A conversão de um modelo em outro modelo.

Níveis de Modelo

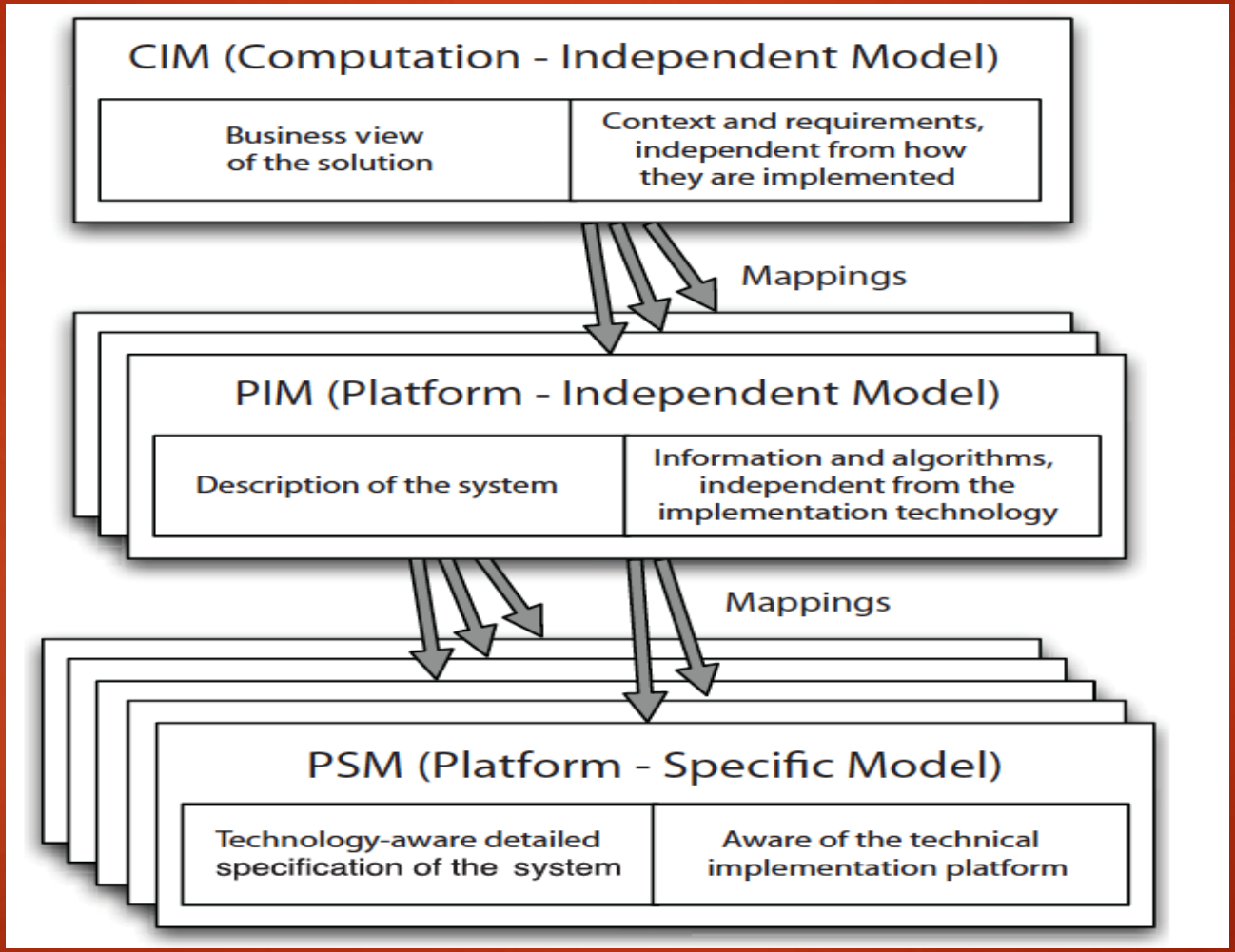
O nível de abstração dos modelos podem variar dependendo dos objetivos dos próprios modelos. Em MDA, três níveis de abstração foram definidos, como mostrados abaixo:

- Computation-Independent Model (CIM): Representa o maior nível de abstração dentre todos os modelos, define o contexto, requisitos, e o propósito da solução sem nenhuma ligação com implicações computacionais. Apresenta a solução do problema, porém esconde todas as especificações de TI, para permanecer independente de como o sistema será implementado. O CIM é frequentemente referenciado como um modelo de negócios ou modelo de domínio porque utiliza um vocabulário que é familiar aos Subject Matter Experts (SMEs);
- Plataform-Independent Model (PIM): O nível que descreve o comportamento e a estrutura da aplicação, independentemente de uma plataforma de implementação. Observa-se que o PIM é somente uma parte do CIM que é solucionada utilizando-se uma solução baseada em software e que refina os requerimentos para o sistema de software. O PIM possui um grau suficiente de independência, que o permite mapear um ou mais plataformas de implementação concretas;

Níveis de Modelo

- Plataform-Specific Model (PSM): Mesmo se este modelo não for executado, este modelo deve conter toda informação requisitada em relação ao comportamento e estrutura de uma aplicação sobre uma plataforma específica que os desenvolvedores poderiam usar para implementar o código executável.

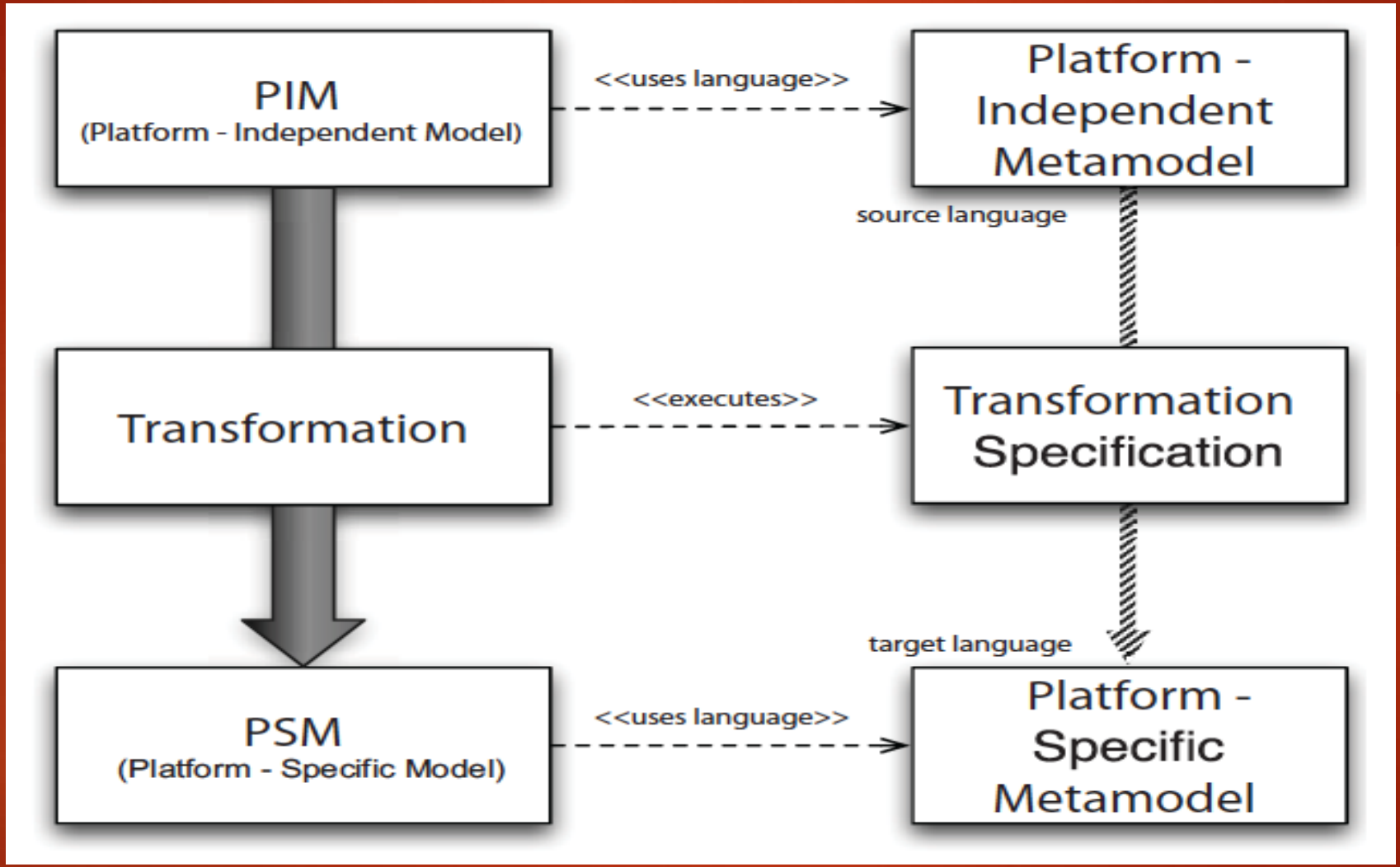
Níveis de Modelo



Mapeamento

- Geralmente, um mapeamento consiste em uma ponte entre diferentes metamodelos. Por exemplo, se o nível PIM é descrito por uma rede de Petri, enquanto que PSM é descrito por algum diagrama dinâmico UML, o mapeamento deve associar conceitos de redes Petri com os seus correspondentes nos diagramas UML. Mapeamento entre metamodelos permite a construção de mapeamentos entre modelos;
- O mapeamento fornece regras e/ou algoritmos para transformações de todas instâncias de tipos de metamodelos de linguagem PIM em instâncias de tipos no metamodelo de linguagem PSM;
- O mapeamento entre diferente modelos (ou níveis de modelo) podem ser aplicados através de transformações. Mapeamento pode prover uma especificação conceitual para implementar transformações entre os diferentes níveis de modelos. O objetivo é automatizar a implementação de mapeamento o tanto quanto for possível, deste modo evitando as dispendiosas transformações manuais.

Mapeamento


PIM
(Platform - Independent Model)
<<uses language>>
Platform - Independent Metamodel
source language
Transformation
<<executes>>
Transformation Specification
target language
PSM
(Platform - Specific Model)
<<uses language>>
Platform - Specific Metamodel

Linguagens de Propósito Geral e de Domínio Específico

- Uma das principais contribuições da OMG é a padronização de um grande número de linguagens de modelagem, abrangendo desde propósito geral até domínio específico. O relacionamento entre os níveis de modelagem e as linguagens de modelagem possui cardinalidade M:N, no sentido que mais de uma linguagem modela diferente aspectos de um nível, e vice-versa a linguagem pode ser utilizada em mais de um nível de modelagem;
- Em MDA a atividade principal (ou central) é a fase de conceituação, que é onde a análise é conduzida: primeiro, requisitos são codificados como PIM (ou mesmo CIM), e então o design atual da aplicação pode ser executado. Isto tipicamente produz o nível PSM. Por sua vez, PSM é transformado em um código executável, e que possivelmente irá requerer programação adicional;
- A linguagem de uso geral mais utilizada é a UML (Unified Modeling Language), que é frequentemente notada como um conjunto de linguagens, pois permitem que os designers especifiquem suas aplicações através de uma série de diferentes tipos de diagramas. Em torno e sobre a UML, um conjunto de linguagens de modelagem de domínio específico (DSMLs – Domaisn-Specific Modeling Language) tem sido definidas para endereçar domínios verticais (por exemplo, financeiro, utilidades, e-commerce) ou domínios horizontais (como aplicações WEB, aplicações mobile e arquiteturas orientadas a serviços), pela exploração tanto de técnicas de meta-modelagem ou características extensíveis fornecidas dentro da linguagem UML;

Linguagens de Propósito Geral e de Domínio Específico

- MDA é neutra quanto a decisão de qual tipo de linguagem utilizar, não favorecendo as de propósito geral (UML) e nem as de domínio específico (DSML). Como conclusão temos que MDA pode utilizar nos níveis CIM, PIM e PSM: a UML pura, uma versão customizada ou ainda a DSML.

Modernização Dirigida por Arquitetura

- ADM (Architecture-Driven Modernization) inclui o problema de engenharia reversa de sistema e também diversos padrões que ajudam este assunto. Em particular a força-tarefa de ADM da OMG ajuda na criação de especificações e na promoção de um consenso para a indústria de que a modernização de aplicações existentes, definidas por qualquer software de produção existente, independentemente da plataforma que rodam, linguagem que são escritas, ou quantidade de tempo que tem estado em produção. O objetivo da força-tarefa de ADM é aperfeiçoar o processo de entendimento e evolução das aplicações de software existentes que tem entrado em manutenção e em modo de evolução, para garantir assim: aperfeiçoamento de software, modificações, interoperabilidade, refatoramento e reestruturação , reuso, migração, integração orientada a serviços e assim por diante.

Modernização Dirigida por Arquitetura

A força-tarefa de ADM definiu uma série de meta-modelos que permitem a descrição de vários aspectos do problema de modernização. Esses meta-modelos são:

- Meta-Modelo de Descoberta do Conhecimento (KDM, Knowledge Discovery Metamodel): Uma representação intermediária para os sistemas de software existentes que define meta-dados comuns requeridos para uma profunda integração do ciclo de vida das ferramentas de gerenciamento. KDM usa o MOF da OMG para definir um formato intercambiável XMI entre ferramentas que trabalham com o software existente e com interfaces abstratas para ferramenta de modernização. KDM suporta descoberta de conhecimento em artefatos de engenharia de software;
- Meta-Modelo de Gerenciamento de Software (SMM, Software Measurement Metamodel): meta-modelo que representa a medida de informação relacionada com o software, sua operação e seu design. A especificação é uma meta-modelo estendido para troca de informações de medição relacionada com software sobre os ativos de software existentes (designs, implementações ou operações);

Modernização Dirigida por Arquitetura

- Meta-Modelo de Ávore de Sintaxe Abstrata (ASTM, Abstract Syntax Tree Model): É uma especificação de modelo complementar ao KDM, onde o KDM estabelece uma especificação para modelos de grafo semânticos, enquanto ASTM estabelece uma especificação para um modelo abstrato de árvore de sintaxe. Portanto, ASTM suporta um mapeamento direto de todos os comandos em nível de código de uma linguagem para modelos de software de baixo nível. Este mapeamento fornece um framework para representação invertida do código escrito em qualquer linguagem de programação.

Modernização Dirigida por Arquitetura