



## **MDD/MDA ISSUES**

**DECEMBER 2, 2005**

---

**Information and Technology Management Branch**

**❖ IM / IT Standards & Guidelines ❖**

*“The entire history of software engineering is that of the rise in levels of abstraction.”*

- Grady Booch in "The limits of Software, September 2002"

## **Document Purpose**

It is desirable to provide a set of standards and guidelines for Model Driven Development (MDD) approach using UML tools. The purpose of this document is to highlight the issues and challenges in establishing standards for the MDD.

## **Model Driven Development (MDD)**

A typical Model Driven Development (MDD) environment advocates the use of models to represent all the relevant information in a software development project. It is a style of software development that revolves around models and metadata definition of the Application. Often, the MDD entails diagrams that in turn form the basis of the application model and metadata. Metadata and code could be generated from the models typically in an MDD environment. MDD was practiced by different Organizations using different architectures and tools. However, Object Management Group (OMG) has since standardized this practice by publishing Industry standard architecture specifications for MDD and named the specifications as Model Driven Architecture (MDA).

## **Model Driven Architecture (MDA)**

Model Driven Architecture (MDA) is touted as the biggest shift in software development since the move from the assembler. MDA includes the MDD. The vision of the MDA is to decouple the Applications from the technology they run on. The purpose of this decoupling is to ensure that the investments made in building the systems can be preserved even when the underlying technologies change. The key buzz-words of MDA are Platform Independent Model (PIM) and Platform Specific Model (PSM).

The PIM is a representation of Application's business functionality and behavior, undistorted by technology details. The PSM is a design-level model more specific to a target technology platform. In order to implement a PIM on a specific platform, a tool is needed to generate the PSM from the PIM. The tool should understand the target technology and should know how to translate the logical constructs of the PIM into a suitable form for the chosen platform. A given PIM element may be mapped to multiple elements in the PSM. For example, a single business object of the PIM may be mapped to an Oracle database table definition, an EJB Entity Bean and a Remote Interface. The code and other runtime technical artifacts are finally generated from the PSM.

Theoretically, automation is used in the MDA approach to produce the PSM and Code from the higher-level PIM artifacts. This automation not only speeds the development process but also ensures that the Code is a true representation of the higher-level models, and the models are a true representation of the Code.

In the MDA paradigm, the Application development process is typically comprised of the following high-level tasks :

- Capture and freeze Business requirements of the Application (functional and non-functional)
- Develop Application's Domain model independent of technology (this is actually the PIM). The Domain model is usually developed in UML. This domain model represents core business services and components.
- Develop other UML diagrams for the Application specific to a particular technology (this is actually the PSM). Typically the PSM is generated from the PIM using an MDA tool and then some required customization is done manually.
- Generate majority of the code from the PSM using an MDA tool.

MDA is different from traditional development in the sense that

- MDA starts from a higher level of abstraction than other design processes. The top-level model PIM is very abstract and just contains conceptual entities and services.
- The PSM is a complete description of the application in the form of metadata. At that level enhancements can be done to the design with technology specific features (e.g. custom finders for EJB entity beans) without touching the code.
- The code generated from the PSM is close to a complete application. The algorithms that generate PSM from PIM, and code from PSM, are intended to be configurable by the architect.

Some of the anticipated benefits of MDA include Faster Application development, Architectural advantages, improved code consistency and maintainability (since code is generated rather than hand-coded) and increased portability across different platforms.

## **Challenges in implementing MDD/MDA**

- MDD/MDA requires increased rigor and training in UML modeling.
- MDA can only do so much. It is unrealistic to expect 100% code generation for every computing problem.
- There is hardly any tool available in the market that realistically offers a complete MDA solution.
- To realize the full benefits of MDA, organizations must support the full software lifecycle development process, from analysis and requirements management through design, development, implementation, deployment, and maintenance.
- The lack of maturity of some MDA standards and specifications are prohibitive to organizations interested in taking full advantage of all that MDA offers. Until these standards and specifications evolve, silver-bullet solutions for MDA do not exist.
- In order to wrap standards and guidelines around MDD/MDA, the development tools and deployment (runtime) environment needs to be precisely determined.