



# **UML DIAGRAMS INTERCHANGE ISSUES**

**DECEMBER 2, 2005**

---

**Information and Technology Management Branch**

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

## Document Purpose

It is desirable to provide a set of standards and guidelines for publishing the UML diagrams in a format that should be importable into Ministry's standard UML modeling tool(s) and environment. Ideally, a two-way model interchange mechanism is desirable so that the diagrams could be imported/exported from the Ministry's standard UML modeling tool(s) to the external UML tool(s) and vice versa. The purpose of this document is to highlight the issues and challenges in establishing standards for UML diagrams interchange between tools.

## XMI Issues

A typical Model Driven Development (MDD) environment advocates the use of models to represent all the relevant information in a software development project. Software development is then carried out as a sequence of model development and transformation. One of the challenges is how to represent models in a machine-independent format to allow model interchange between tools and systems. This exchange format should be well documented, stable and supported by different tools from different vendors.

The Object Management Group (OMG) proposed the use of XMI and XMI-DI standards to enable model/metadata and diagram interchange respectively. One of the strong points of XMI is that it is based on XML. XML has been successfully used to support many document and model representation standards. XML is well documented, machine-independent and there exist a number of tools supporting it. Thus, XMI documents should be portable and easy to parse. XMI has some limitations, however :

- XMI cannot be used to define the structure of a modeling language. [This is the role of the OMG's Meta Object Facility (MOF) standard.]
- *Two tools that use two different versions of the UML (say UML 1.5 and UML 2.0) will not be able to exchange models even if they both use XMI.*
- UML and MOF meta-models do not contain information about the diagrammatic representation of models. That is, a UML model may state that there is a class called "*Loan Applicant*" in a model, but it cannot state that this class is represented in the diagram by a rectangle in a certain position, size and color. [To remedy this situation OMG proposed the XMI-DI standard, which is described in later sections in this document].
- XMI is neither an application programming interface (API) (like Java JMI or Eclipse EMF) to retrieve information from models nor a communications protocol (like HTTP or WebDAV) to transport models between systems. This means that there are no standard software components to create or retrieve XMI documents from a file system or multi-user repository since the API and communication mechanism for such components has not been standardized.
- XMI standards are in constant evolution, the current version being XMI 2.0 and the older versions being 1.0, 1.1 and 1.2.

## XMI-DI Issues

Since XMI is not suitable for the interchange of diagrammatic representation of UML models, OMG proposed another standard called "XMI-DI" for diagram interchange. XMI-DI is not an extension to XMI or a different way of serializing models to streams. XMI-DI is just another standard explicitly designed for enabling to display models on a two-dimensional canvas such as a monitor. XMI-DI has three main concepts called *Diagram*, *GraphNode* and *GraphEdge*.

XMI-DI has the following limitations :

- The XMI-DI language is not rich enough to describe the graphical presentation of models.
- The information on how to render concrete XMI-DI nodes and edges is not present in the language. As a result, an XMI-DI document can state that there is a diagram with a UML use case in it, but it cannot state that it is rendered as an ellipse shape and not as a rectangle.
- XMI-DI does not contain information about how to create a diagram from an abstract model. For example, given a UML class model, XMI-DI cannot describe how we can create the corresponding GraphNodes and GraphEdges that represent all classes, associations, etc.
- XMI-DI cannot express layout constraints. Some UML diagrams, such as sequence diagrams, have a layout constrained by the semantic model.

## Tool specific issues (Ministry's existing UML tools)

- While importing diagrams from an external XMI file, most UML tools leave out any diagram elements/metadata that is unrecognizable by the target tool, which leads to only partial and sometimes inconsistently imported diagrams.
- JDeveloper 10g (9.0.5.2) does not have XMI export facility. It can only import diagrams from external XMI files.
- JDeveloper imports only UML 1.3 compliant diagrams.
- JDeveloper XMI import facility allows only class diagrams import from external XMI files. Other diagrams cannot be imported.
- JDeveloper imports diagrams only from "Rational Rose (Unisys) XMI 1.3.2 or 1.3.3" or "TogetherJ OMG XMI 1.1" files.
- Rational XDE Modeler exports and imports models in XMI format conformant to XMI 1.1 DTD.
- Rational XDE Modeler does not support UML 2.0. Only UML 1.4 diagrams can be interchanged in XMI format.
- The current version of Rational Rose Modeler does not support diagram import/export in XMI format.