PIM->PSM 模型转换的途径

mdaSky　UML软件工程组织

由MDA 的PIM（平台独立模型）向PSM（平台特定模型）转换的方法目前尚未实现标准化。因此目前市售的工具不得不利用自主方法进行这部分的处理。由PIM 向PSM 的转换方法由于将在2004 年实现标准化，只有这个重要的步骤标准化了，才更加有利于MDA 这项技术的推广。

2004 年将是MDA 大发展的一年，为什么这样说，我们来看看业界一些重要的公司是如何应对MDA 这项技术的。最近，美国Compuware 的OptimalJ 等基于对象技术标准化团体美国OMG（Object Management Group ）倡导的模型驱动架构（MDA）的Java 开发工具业已亮相。那么Java 工具阵营的老大哥Borland 公司的JBuilder 是否会支持MDA 那？看看他们是怎么说：“我们也在关注MDA， 但是目前仍在观察其动向。比如说第一点，OptimalJ 等产品与JBuilder，包括价格在内，不属于同一类产品。要是支持MDA 的话，Together 更好一些。JBuilder X 在能够轻松构筑Web 应用的角度上，以比这些工具更低的成本实现了相同的功能。同样，即便1 行代码都不写，也能够自动生成可访问数据库的Web 应用架构，在开发过程中及开发完成后均可轻松变更Web 应用服务器等平台。由PIM 向PSM 的转换方法由于将在2004 年实现标准化，因此到时准备在Together 中配备基于MDA 的模型自动生成功能。”看来Borland 公司也不会轻视MDA 这项技术，准备在Together 产品中支持MDA。

MDA 技术是否会取得较大的成功，让我们拭目以待。

下面简单讲述一下从PIM 到PSM 转化的5 种途径：

1. Marking

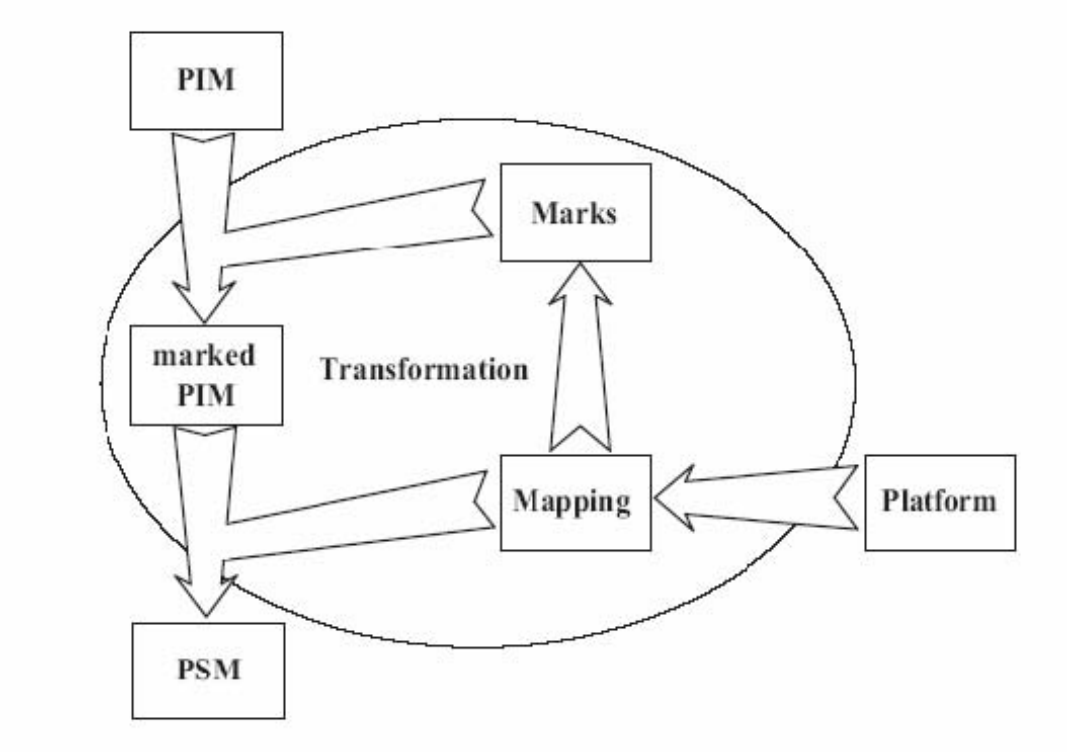

图1 Marking a Model

在转换之前选择了一个特定的平台，与这个平台对应有一个映射规则（mapping），这个映射规则包含一系列预先定义的标记（marks）。这些标记用于给模型中的元素增加标注，用于指导模型的转换过程。给模型元素增加完标记以后，将使用此平台对应的映射规则对加了标记的模型进行转换，生成对应的平台相关的PSM。

2. Metamodel Transform

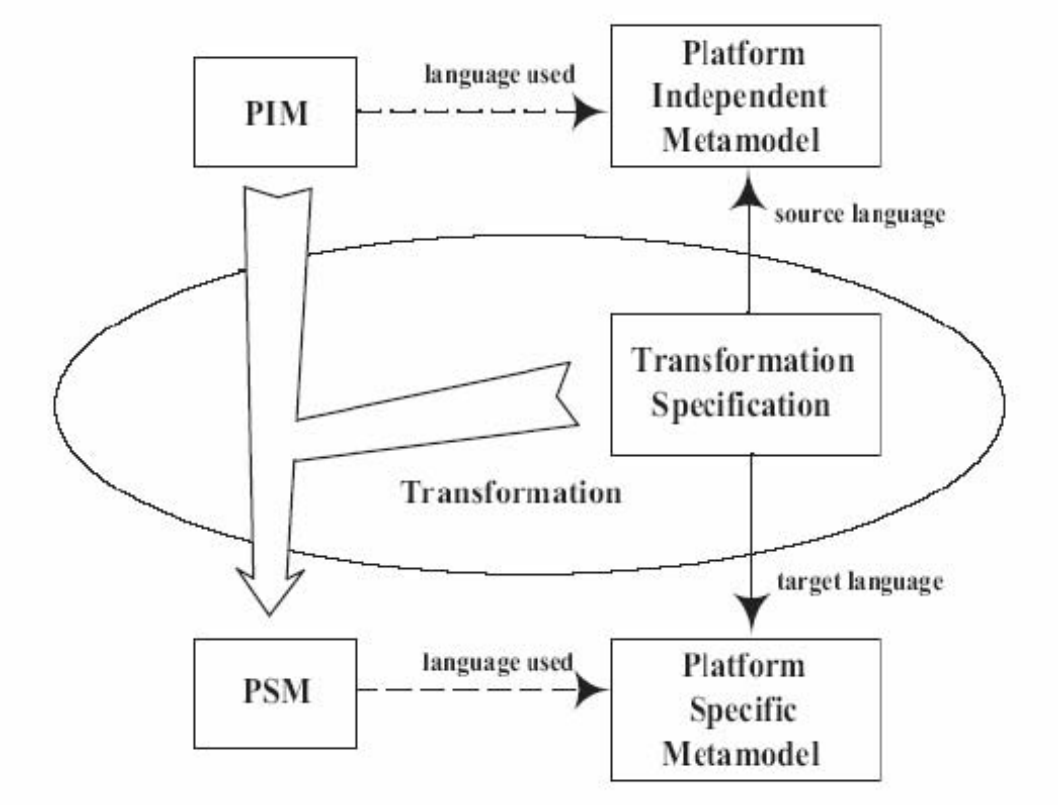

图2 MetaModel Transform PIM 模型使用平台无关的语言来说明，这种平台无关的语言使用平台无关的元模型（metamodel）来描述。

PSM 模型使用平台相关的语言来说明，这种平台相关的语言同样使用平台相关的元模型来描述。此选顶的特定平台存在一个转换规则，按照此规则将平台无关的元模型转换为平台相关的元模型，从而实现从PIM 生成PSM 的转换目的。

举个例子：

平台无关的元模型是EDOC ECA 业务处理模型（EDOC ECA Business Process Model）

，平台相关的元模型是一个工作流引擎（workflow engine）的MOF 模型。转换规则是一个MOF QVT 转换模型。转换过程由一个工具生成的转换引擎来完成， 此转换引擎使用一对MOF 模型来创建一个特定的转换引擎。

3. Model Transform

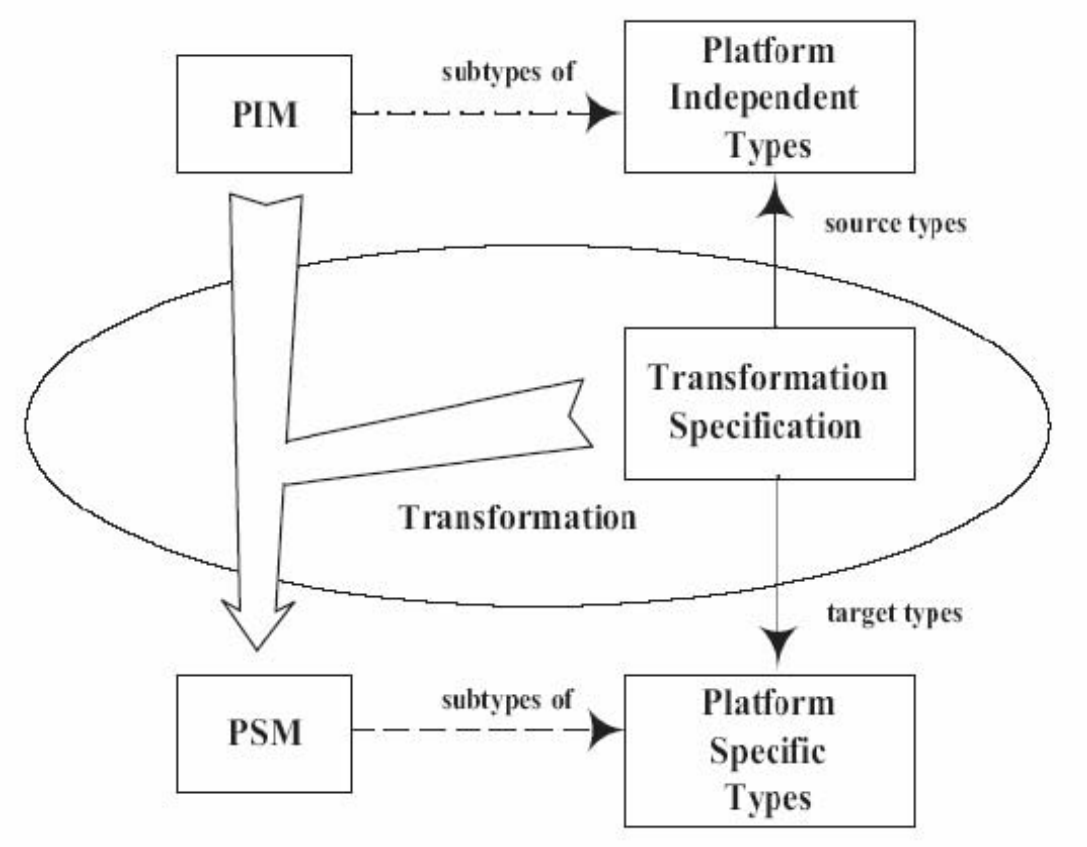

图3 Model Transform

PIM 模型使用平台无关的类型来说明，这些类型可能是一种软件框架（software framework） 的一部分。PIM 中的元素是这种平台无关的类型的子类型。

PSM 模型使用平台相关的类型来说明，PSM 中的元素是平台相关类型的子类型。此选定的特定平台存在对应的转换规则，按照此规则将平台无关的类型转换为平台相关的类型，从而实现从PIM 生成PSM 的转换目的。

举例如下：

平台无关的类型声明了各种平台通用的能力和特性，平台相关的类型可以是某个特定平台的合成类，声明了这个特定平台类型提供的能力和特性。

这种方式和元模型映射主要的不同就是使用模型中指明的类型来进行映射，而不是使用元模型指明的概念来进行映射。

4. Pattern Application

扩展自模型映射和元模型映射方式，在他们的基础上增加了模式（patterns），和类型或者模型语言概念配合使用。

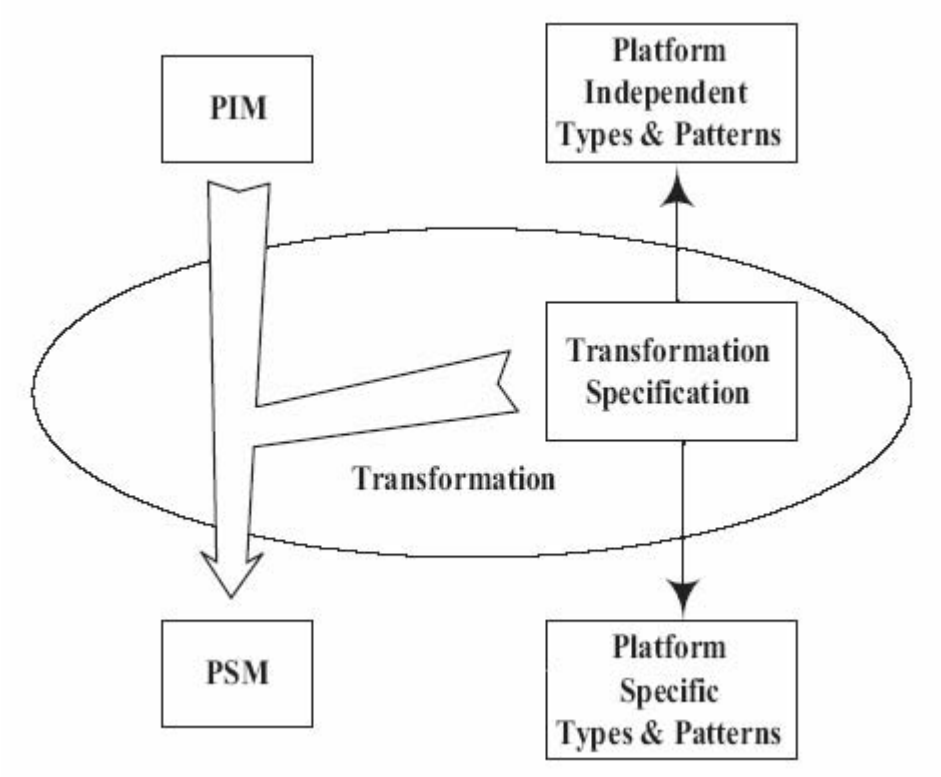

图4 Model Transform

除了平台无关的类型之外，一个通用的模型还可以补充一些模式。平台无关的类型和模式都可以被映射为特定平台相关的类型和模式。

举例如下：

我们为一种概念性的RM-ODP 工程语言（RM-ODP Engineering Language） 定义了一个平台无关的通用模型。此模型根据此语言的概念定义了相应的对象类型，根据此工程语言的结构规则定义了相应的模式。我们选择使用CORBA 来实现它，使用CORBA 的映射规则将概念语言的对象类型映射为相应的CORBA 对象类型，将其模式映射为相应的通用ORB 架构。ODP 存根模块（stubs）变为CORBA 存根（stubs）和骨架（skeletons）。ODP binders 的功能将被映射为ORB 和object adapter 的功能。ODP interceptors 变为CORBA interceptors 等等。

元模型映射方式也可以采用同样的方法来使用模式：

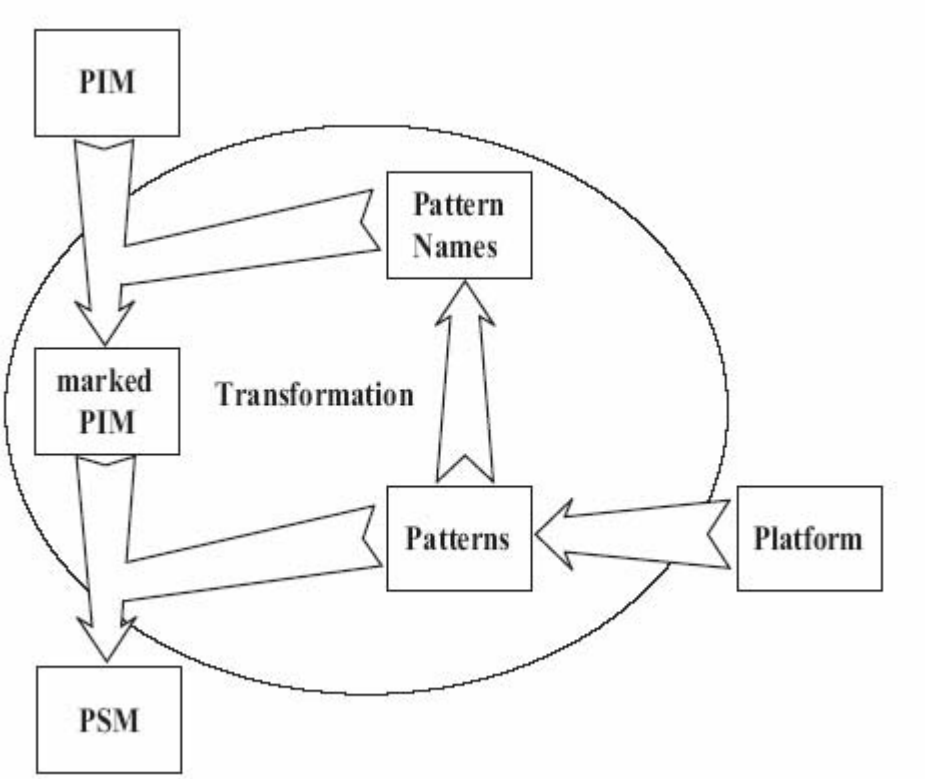

图5 Another way to use Patterns 图5 展示了另一种使用模式的方法－将模式作为平台相关的标记的名称，也就是说，模式的名称对于某个平台是特定的。

5. Model Merging

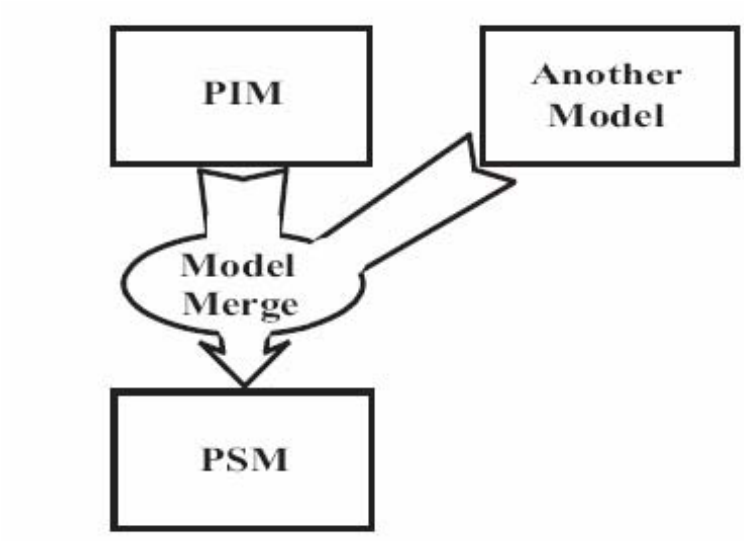

图6 Model Merging 这里有多种转换方法是基于模式合并的，例如上面的使用

patterns 和pattern application 的例子。pattern application 就是模式合并的一种。

6. Additional Information

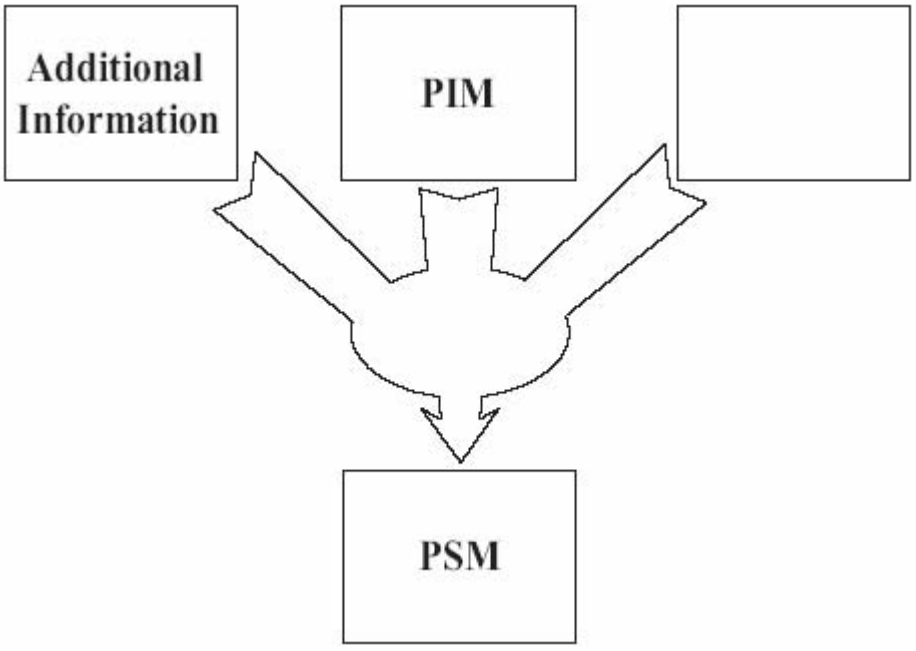

图7 Inclusion of Additional Information 除了PIM 和平台相关的标记以外，还可以提供附加信息来指导转换的过程。例如：可以指定一个特定的框架风格，也可以给Connectors 增加特定的服务质量的附加信息。通常附加的信息吸收了设计者的特定认识，这些认识是对应用程序领域和平台知识共有的认识。下图扩展了简单的MDA 模式用来展现如何使用附加信息：

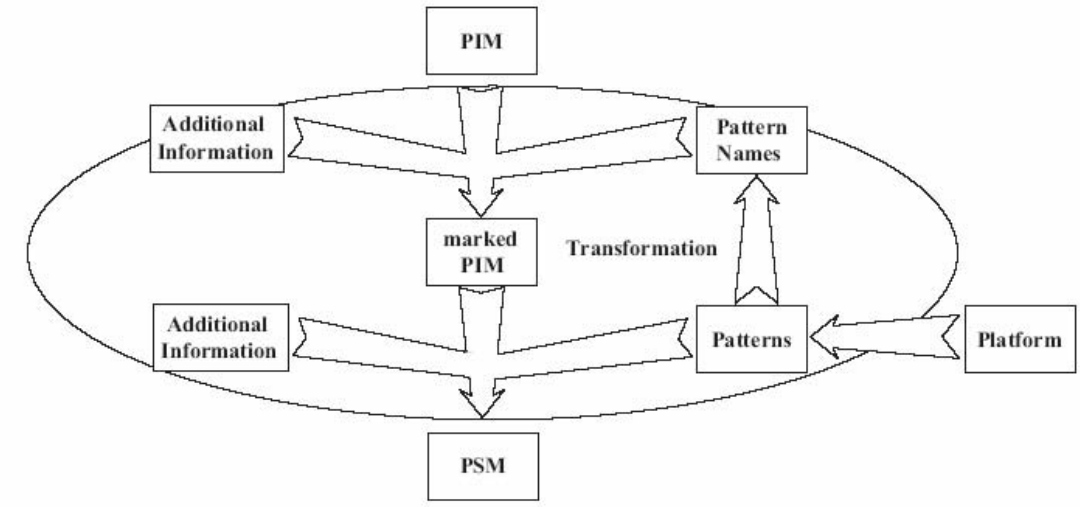

图8 Use of Additional Information in a Particular Transformation Technique 图8 深入展开MDA 模式用来展示在某种特定转换技术上使用附加信息的方法。在准备PIM

的过程中，模式和附加信息都被加入用来生成加了标记的PIM。然后，模式和附加信息都被使用，用来对加了标记的PIM 进行转换来生产PSM。