

Model transformations

<!-- -->

Table of contents

1 Model mapping and transformation.....	2
1.1 Mapping.....	2
1.2 Transformations.....	3

1. Model mapping and transformation

An important aspect of Model Driven Development (MDD) is model transformations, which allows automatically transformations of models. A model transformation is a transformation of one or more source models to one or more target models, based on the meta models of each of these models. In other words the instances of one meta model is transformed into instances of another meta model.

Such transformations are defined by mapping rules. Each mapping rule describes what one, or more elements in the source model should be transformed to in the target model. When all mapping rules are applied, the mapping describes the complete transformation from the source model to the target model.

Thus given a source model, and the metamodels of both the source and the target models, one can automatically generate the target model by applying the correct mapping to the source model.

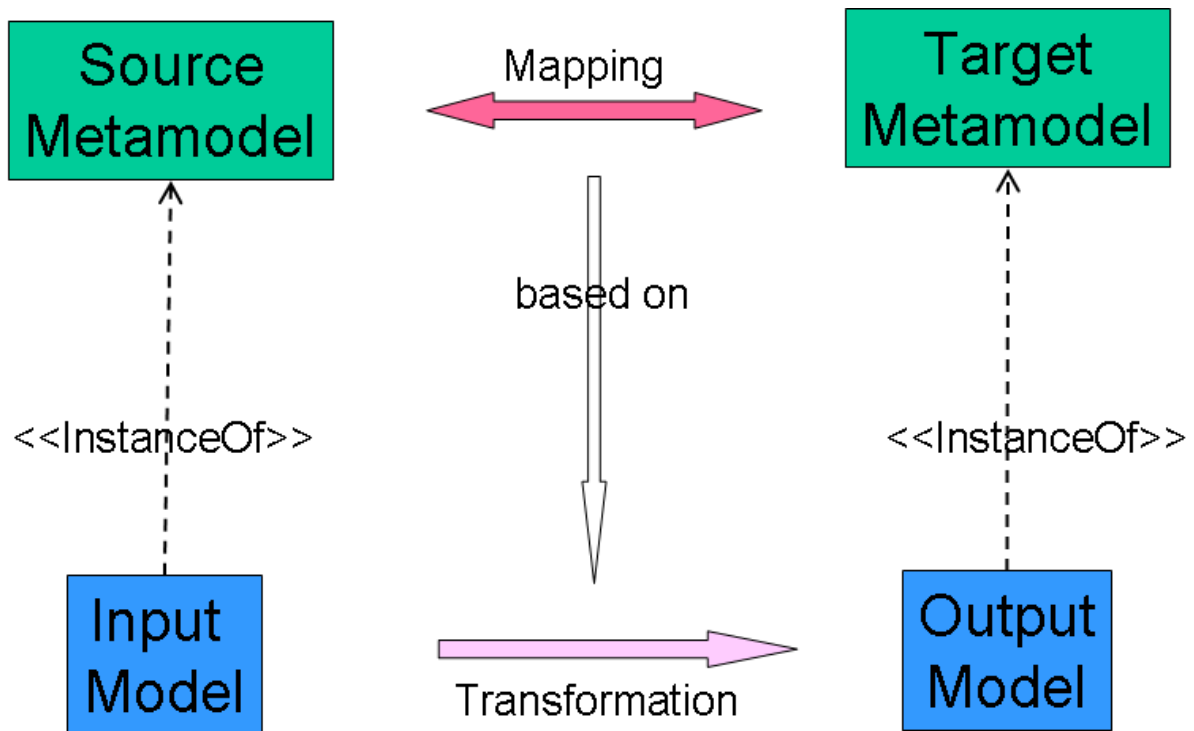
A common use of transformations is the transformations of Platform Independent Model (PIM) to Platform Specific Model(PSM), and PSM to code. The PIM should, as the name implies describe the system in a total platform independent way. Thus, whether your system stores data in a database or not, or is implemented in C, C++ or Java is in no interest here. The PIM simply captures what your system does, not how. After making a PIM of the system, one should then make PSM's for the different types of technologies used. The PSM should describe how the system is implemented, using a specific technology. In other words, if your system is implemented using Java, WSDL, and BPEL, you make one for each technology. When all the PSM's are made, one should create mapping rules describing the transformation from the PIM, to each of the PSM's.

As a last step, it's now possible to describe mappings from each of the PSM's to code. This code could be complete code, but most times it would be more of a skeleton of the code, where parts of the code would have to be manually entered.

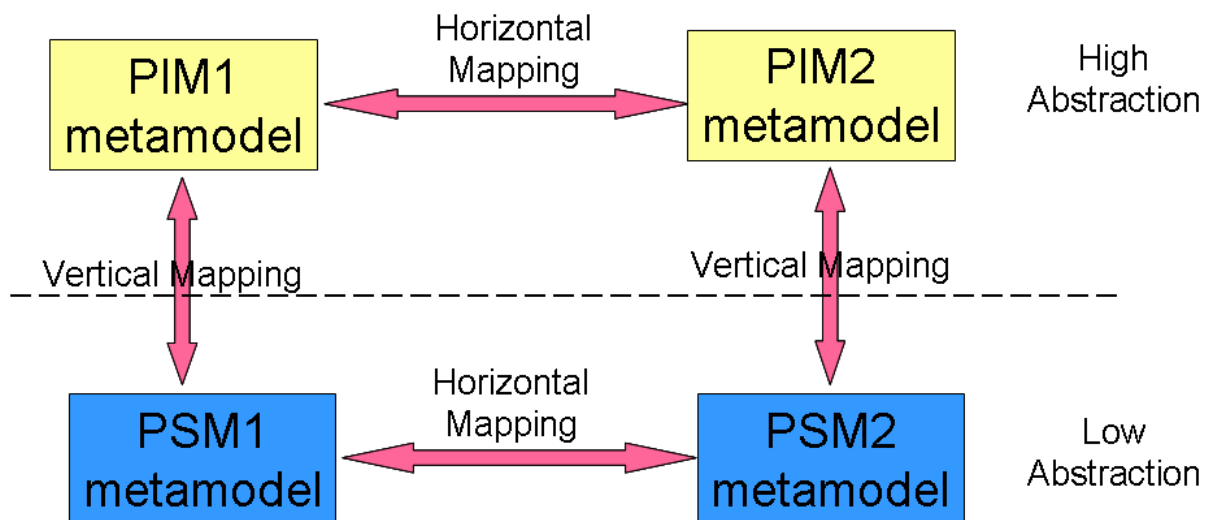
1.1. Mapping

Mapping is performed by defining relations between two models. The relations can be 1-to-1, n-to-1, 1-to-n or n-to-n. Mapping is performed in "design time".

The mapping is defined with models one meta-level higher than the input and output of the transformation. The mapping is used to perform transformation of instances of the mapped models. "The mapping describes the rules used for the transformation".



We can map between models that are on the “same” abstraction level illustrated with horizontal mapping, or we can map between abstraction levels illustrated with vertical mapping in the figure below.



1.2. Transformations

A transformations occur at "run-time" and takes input and produces output. A transformation is a one-way process which transforms according to a predefined mapping. Two main categories of transformation:

- Vertical transformation
- Horizontal transformation

In a vertical transformation the source model has the same level of abstraction as target model. Not to be confused with "meta-levels". Examples of horizontal transformation:

- Refactoring
- Merging

In a horizontal transformation the source model is at a different level of abstraction than the target model. Examples of vertical transformation:

- Refinement (specialization)
 - PIM->PSM transformations
- Abstraction (generalization)