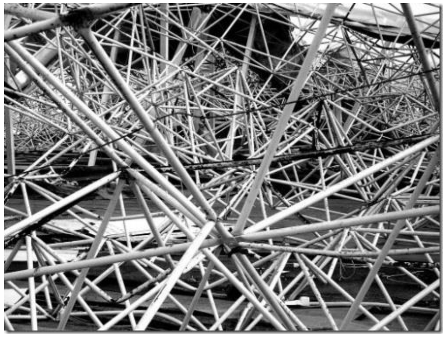
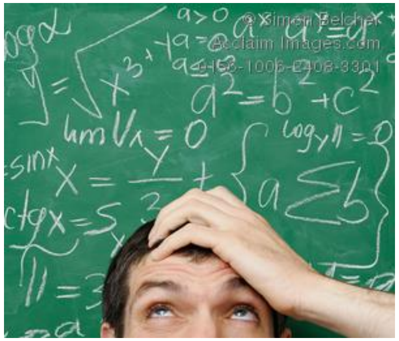
Formal Specification and Testing of Model Transformations



Javier Troya
Manuel Wimmer
Antonio Vallecillo
TU Wien & Univ. Málaga



UNIVERSIDAD DE MÁLAGA

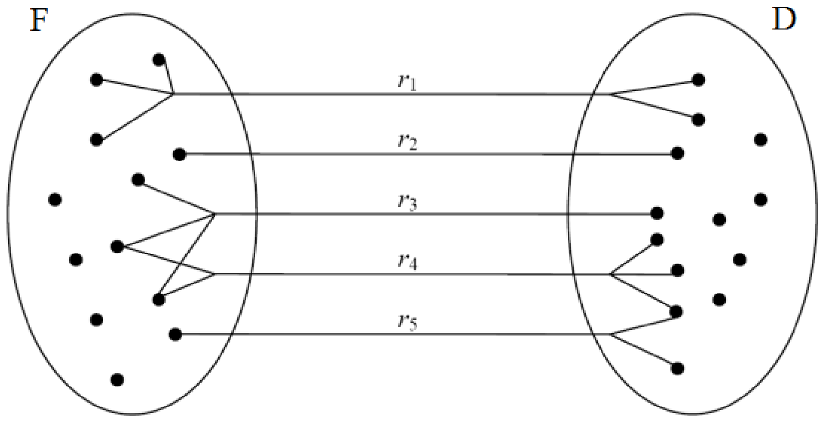
TU WIEN

Model Transformations

A **model transformation** is

(1) *The specification of the relationship between one set of source models and one set of output models*

(2) *The process that generates such relationship*

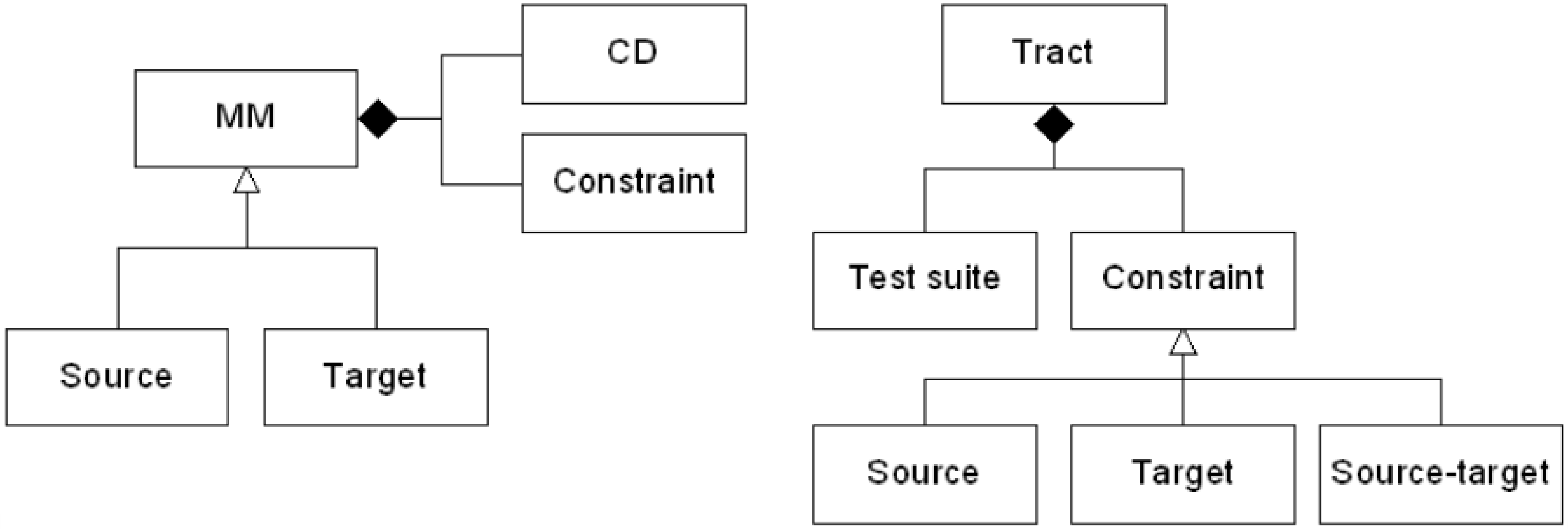


- Model Transformations describe the relationships between views of a system, or provide bridges between views of two systems
  - Refinement relations, development relations, abstraction relations, mapping relations, …

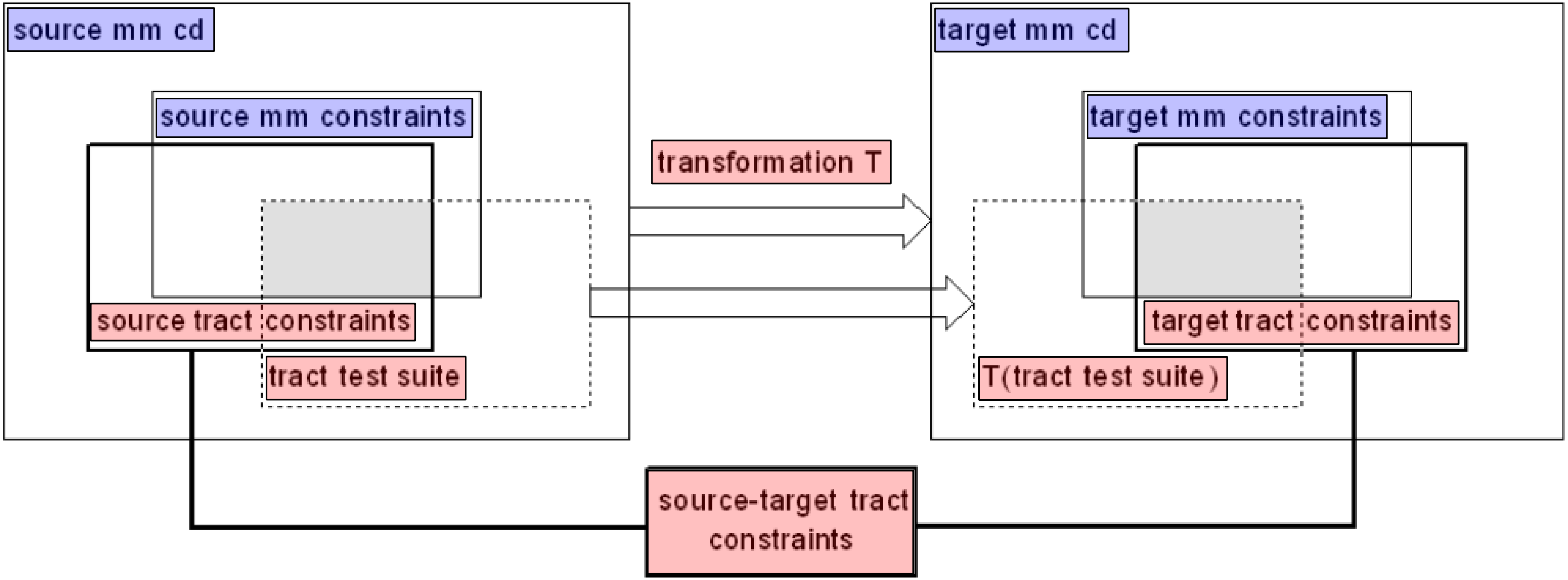
(Joint work with M. Gogolla)

Tracts

- Blackbox Con***Tracts*** for Model transformation
- A Tract defines
  - a set of **constraints on the source and target metamodels**,
  - a set of **source-target constraints**, and
  - a **tract test suite** (a collection of source models satisfying the source constraints)



The elements of a *Tract*



Lightweight & Black-box Testing of MTs

- For each tract
  - Input test suite models are automatically generated using ASSL
  - Input models are transformed into output models by the transformation under test
  - The results are checked with the USE tool against the constraints defined for the transformation
- Different tracts are defined for every transformation
  - Each one defines either a *use case* or a *special condition* or a *negative test case*
  - Test suites are key to Tracts (coverage, repeatability, etc.)

Mutation Testing: Discussion Points

- *How to **derive** meaningful mutation operators?*
  - Generate change metamodel from transformation metamodel?
  - Explore mutations from transformation evolutions?
  - Extract mutations from transformation corrections?
- *How to **use** mutations?*
  - Test suite evaluation?
  - Tracts evaluation?
  - Fault localization evaluation?
  - Repair?
- *How **language-specific** should mutation operators be?*
  - In-place languages
  - Out-place languages
  - Domain-specific transformation languages
- After mutations are identified, how should the artifacts **evolve?**
  - Propagate changes to model transformation
  - Propagate changes to output model