transformation languages

all about data processing

- processing formats
- representation
- design, keep logic

theoretical examples

NIMBIN
NEWSAGENCY
& MINI MART

inputs

- emailed stories
- uploaded stories
- stories from other news agencies
- etc.

outputs

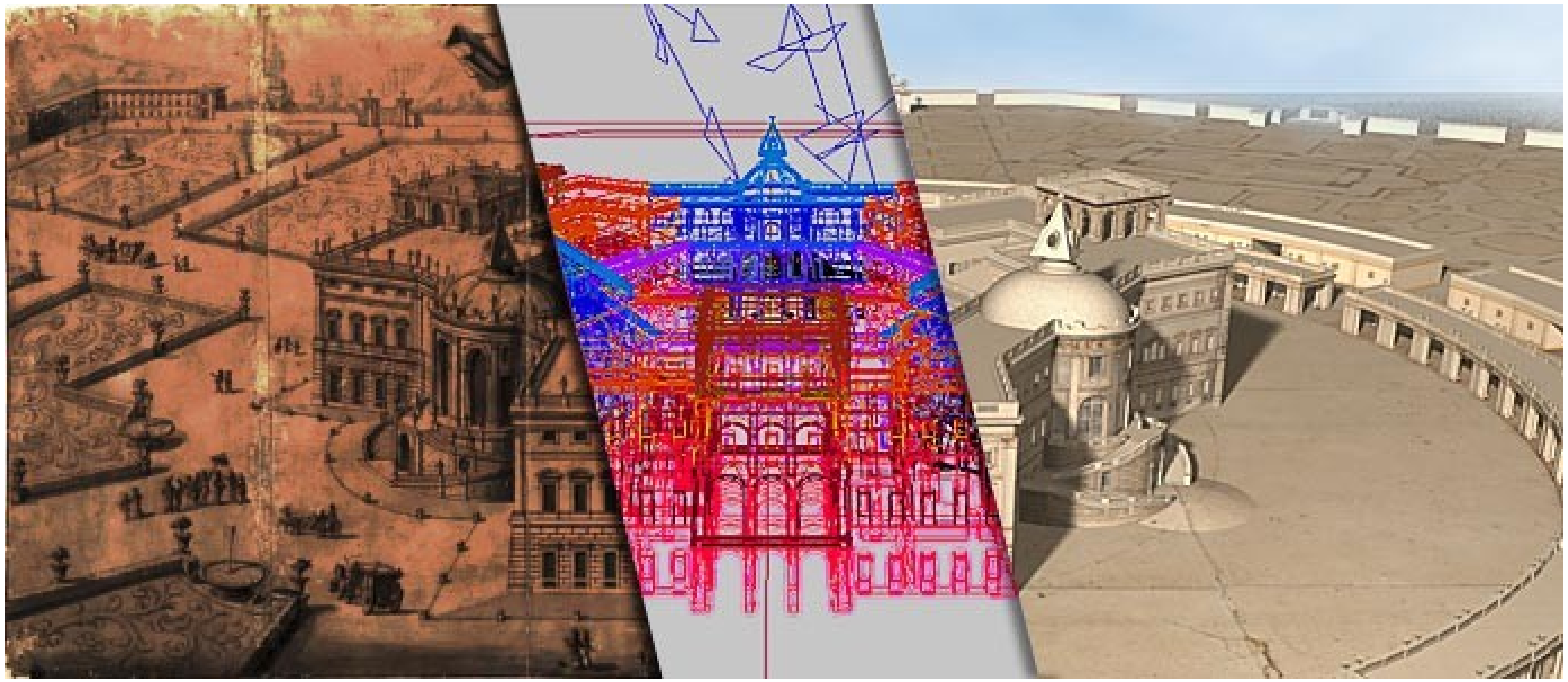
- newsletter
- website / feeds
- printed articles
- ticker
- etc.

processing

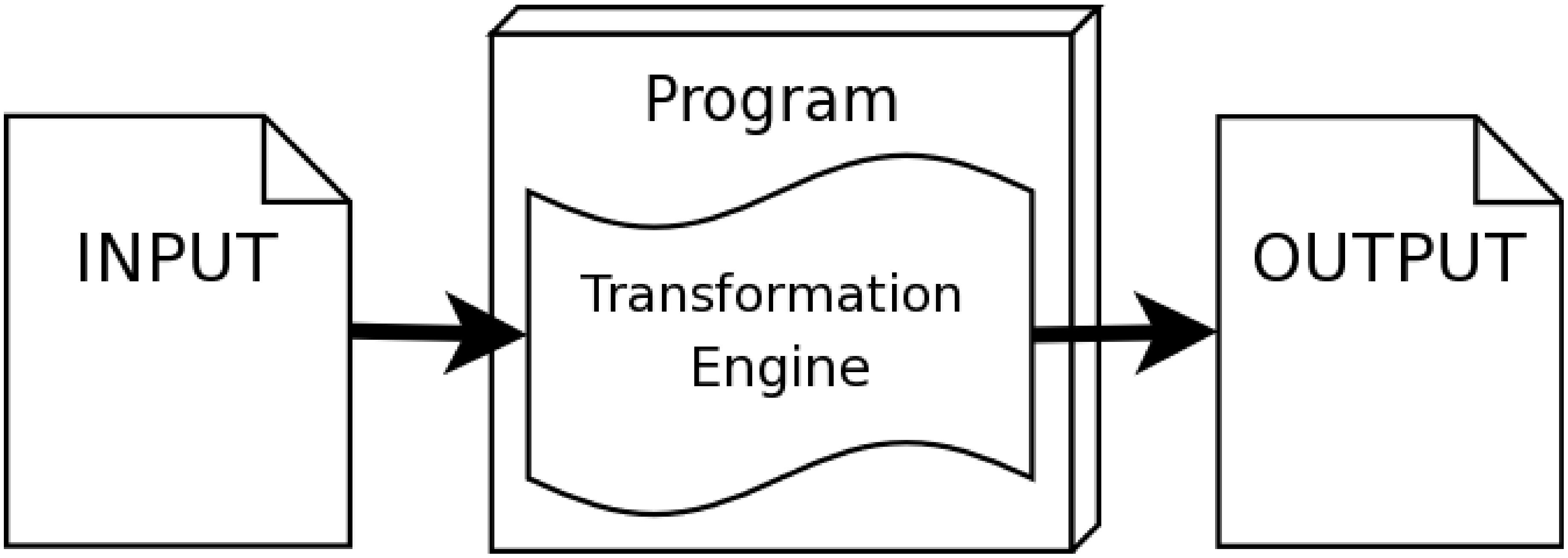
- formats
- ordering
- archiving
- output

NIMBIN
NEWSAGENCY
& MINI MART

other example: models



work flow



they are everywhere

practical examples

txl

The TXL Programming Language

Source Transformation by Example

- unqique programming language
- designed to support source transformation tasks
- quite old

txl: allrounder

- syntax checking / pretty printers
- automation in software maintenance
- supports many kind of transformations

example: tree transformations

- bad (old) html
- want to have strict xhtml
- let txl translate it for us

demo

program transformation

- changing one program into another.
- a big taxonomy

program

structured object

with semantics

stratego / xt

language and toolset for program transformations

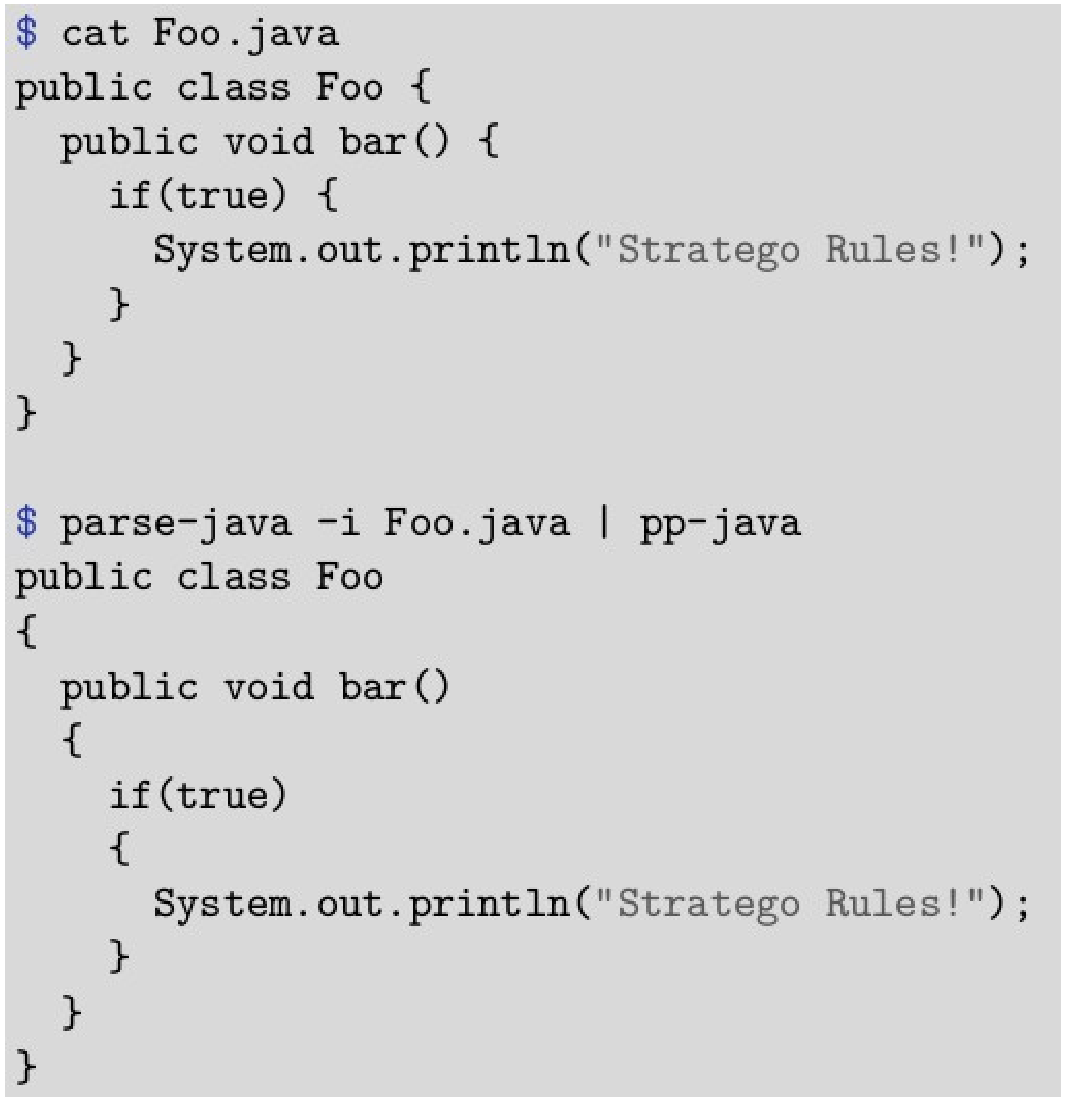
demo

unfortunately no demo

use cases

- java
- php
- java flavours

example: pretty printing java



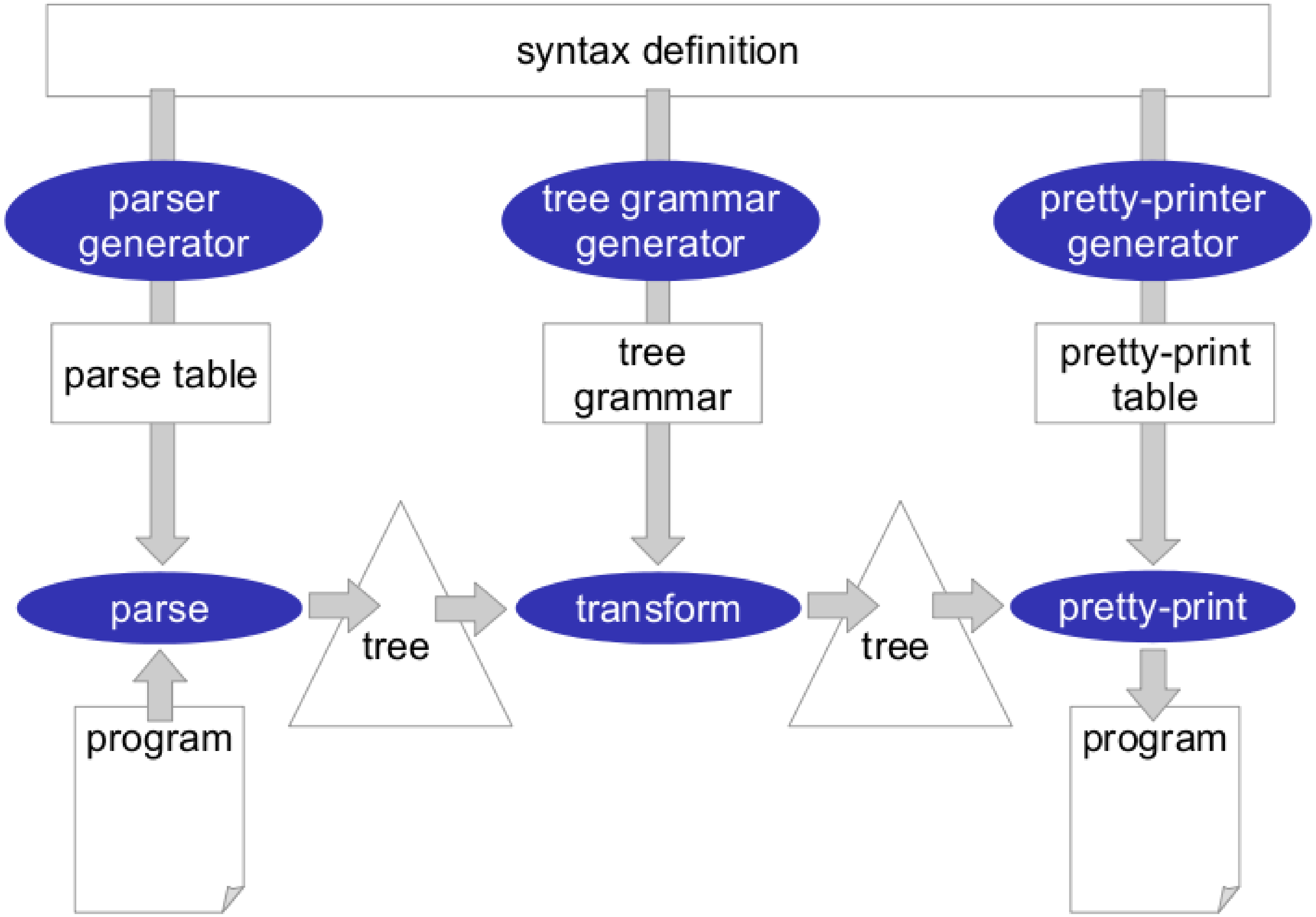
```
$ cat Foo.java
public class Foo {
  public void bar() {
    if(true) {
      System.out.println("Stratego Rules!");
    }
  }
}

$ parse-java -i Foo.java | pp-java
public class Foo
{
  public void bar()
  {
    if(true)
    {
      System.out.println("Stratego Rules!");
    }
  }
}
```

is this everything?

-> strategic rewritting

stratego / xt



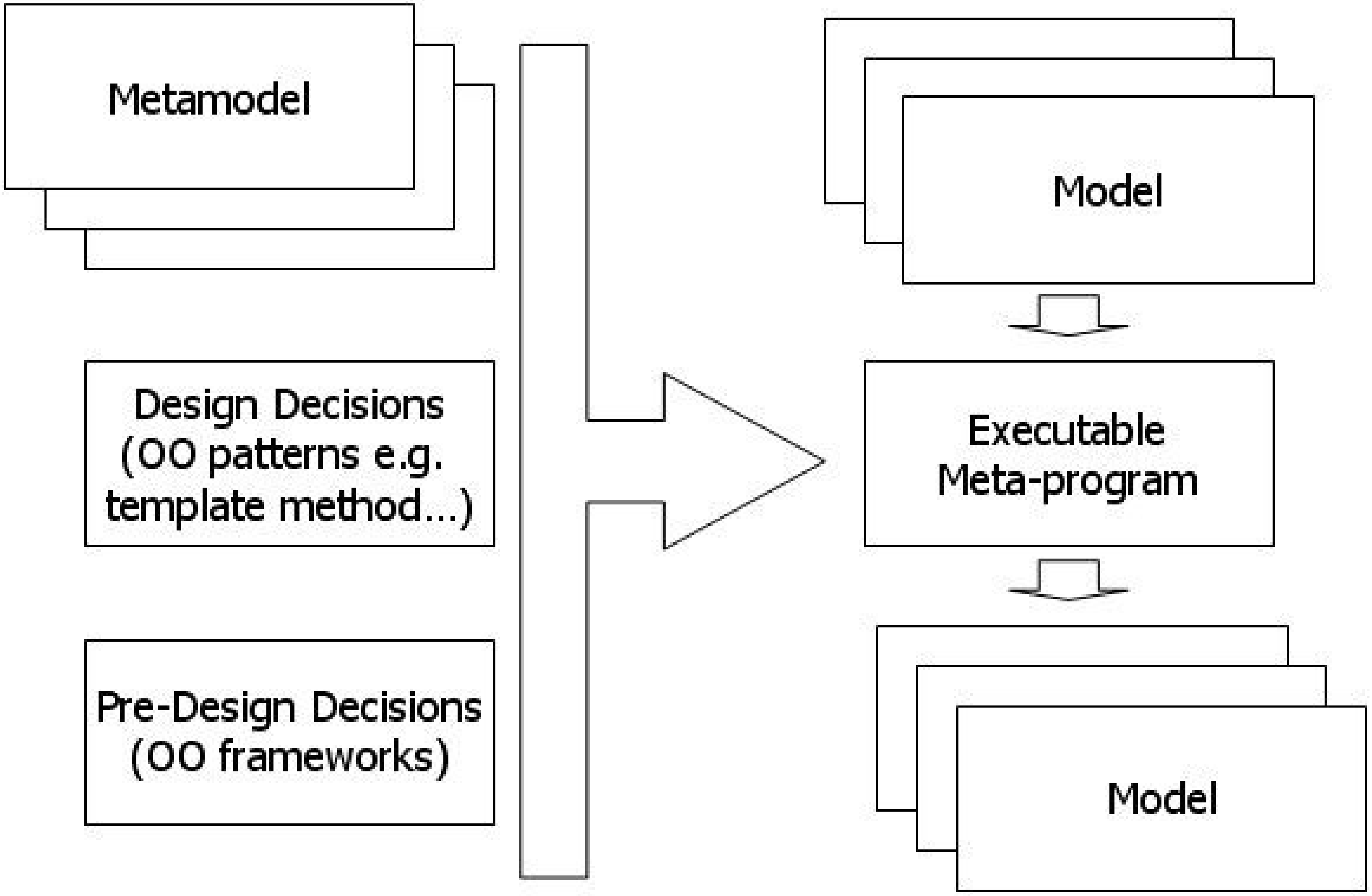
syntax definition
parser generator
tree grammar generator
pretty-printer generator
parse table
tree grammar
pretty-print table
parse
tree
transform
tree
pretty-print
program
program

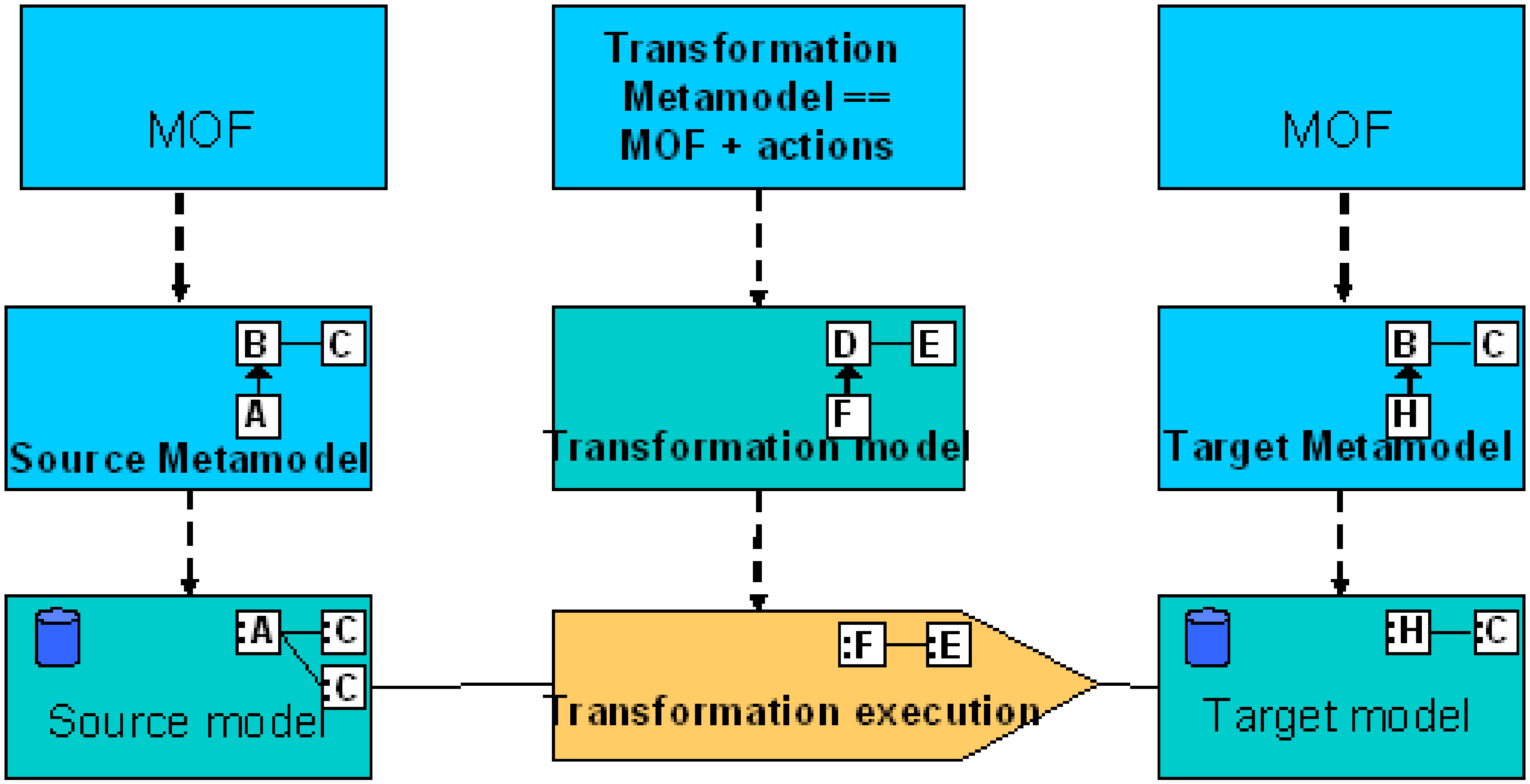
model transformations

model 2 model

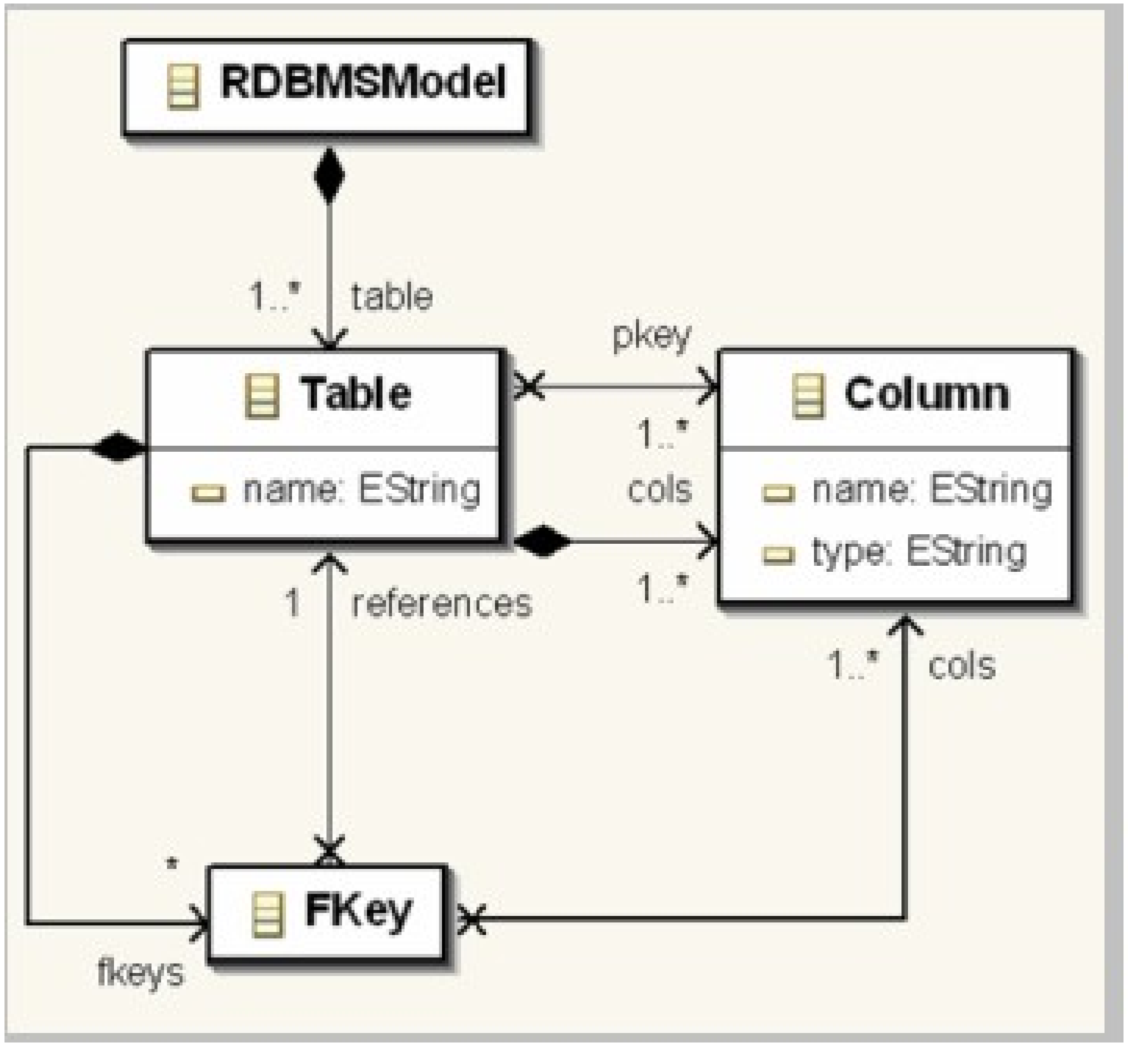
transformations

metamodeling transformations


Metamodel
Design Decisions (OO patterns e.g. template method...)
Pre-Design Decisions (OO frameworks)
Model
Executable Meta-program
Model


MOF
Transformation Metamodel == MOF + actions
MOF
B
C
A
Source Metamodel
D
E
F
Transformation model
B
C
H
Target Metamodel
:A
:C
:C
Source model
:F
:E
Transformation execution
:H
:C
Target model

example: class2rdbms



```
package RDBMSMM;
require kermeta
using kermeta::standard
class Table{
        attribute name : String
        attribute cols : Column[1..*]
        reference pkey : Column[1..*]
        attribute fkeys : FKey[0..*]
}
class FKey{
        reference references : Table
        reference cols : Column[1..*]
}
class Column{
        attribute name : String
        attribute type : String
}
class RDBMSModel{
        attribute table : Table[1..*]
}
```

demo

conclusions

- helps to deal with different formats / models
- helps to transform different formats / models

conclusions

- (meta-)modeling!
- abstract or concrete transformations

usage

- many data formats
- different representations or usages

fin

questions?