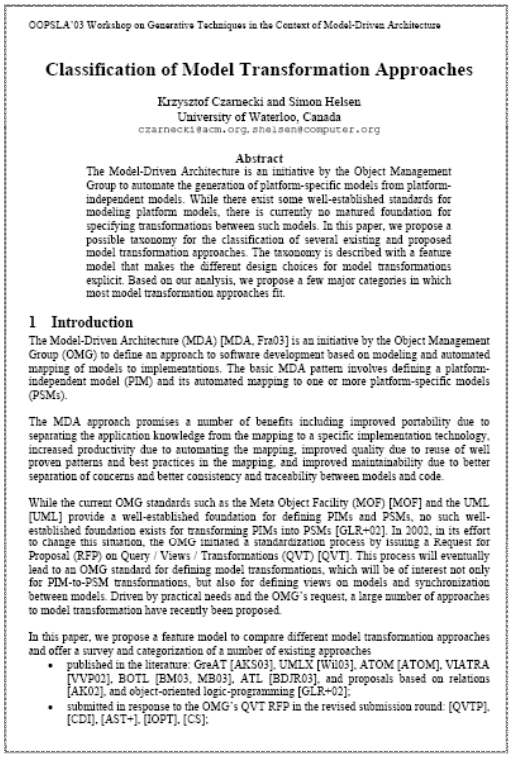
Design Space of Heterogeneous Synchronization REVISITED

Michal Antkiewicz
Krzysztof Czarnecki
Zinovy Diskin

University of Waterloo

K. Czarnecki, S. Helsen.
Classification of Model Transformation Approaches,
OOPSLA Workshop on Generative Techniques in the
Context of Model-Driven Architecture, 2003



Classification of Model Transformation Approaches

Krzysztof Czarnecki and Simon Helsen
University of Waterloo, Canada
czarnecki@acm.org, shelsen@computer.org

Abstract

The Model-Driven Architecture is an initiative by the Object Management Group to automate the generation of platform-specific models from platform-independent models. While there exist some well-established standards for modeling platform models, there is currently no matured foundation for specifying transformations between such models. In this paper, we propose a possible taxonomy for the classification of several existing and proposed model transformation approaches. The taxonomy is described with a feature model that makes the different design choices for model transformations explicit. Based on our analysis, we propose a few major categories in which most model transformation approaches fit.

1 Introduction

The Model-Driven Architecture (MDA) [MDA, Fra03] is an initiative by the Object Management Group (OMG) to define an approach to software development based on modeling and automated mapping of models to implementations. The basic MDA pattern involves defining a platform-independent model (PIM) and its automated mapping to one or more platform-specific models (PSMs).

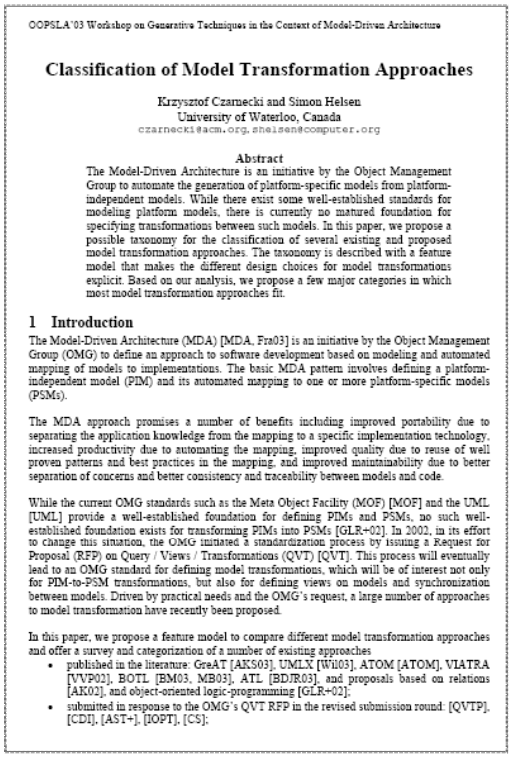
The MDA approach promises a number of benefits including improved portability due to separating the application knowledge from the mapping to a specific implementation technology, increased productivity due to automating the mapping, improved quality due to reuse of well proven patterns and best practices in the mapping, and improved maintainability due to better separation of concerns and better consistency and traceability between models and code.

While the current OMG standards such as the Meta Object Facility (MOF) [MOF] and the UML [UML] provide a well-established foundation for defining PIMs and PSMs, no such well-established foundation exists for transforming PIMs into PSMs [GLR+02]. In 2002, in its effort to change this situation, the OMG initiated a standardization process by issuing a Request for Proposal (RFP) on Query / Views / Transformations (QVT) [QVT]. This process will eventually lead to an OMG standard for defining model transformations, which will be of interest not only for PIM-to-PSM transformations, but also for defining views on models and synchronization between models. Driven by practical needs and the OMG's request, a large number of approaches to model transformation have recently been proposed.

In this paper, we propose a feature model to compare different model transformation approaches and offer a survey and categorization of a number of existing approaches

- published in the literature: GreAT [AKS03], UMLX [Wil03], ATOM [ATOM], VIATRA [VVP02], BOTL [BM03, MB03], ATL [BDJR03], and proposals based on relations [AK02], and object-oriented logic-programming [GLR+02];
- submitted in response to the OMG's QVT RFP in the revised submission round: [QVTP], [CDI], [AST+], [IOPT], [CS];

K. Czarnecki, S. Helsen.
Feature-based survey of model transformation approaches,
IBM Systems Journal, 45(3), 2006, pp. 621-646



OOPSLA'03 Workshop on Generative Techniques in the Context of Model-Driven Architecture

Classification of Model Transformation Approaches

Krzysztof Czarnecki and Simon Helsen
University of Waterloo, Canada
czarnecki@acm.org, shelsen@computer.org

Abstract

The Model-Driven Architecture is an initiative by the Object Management Group to automate the generation of platform-specific models from platform-independent models. While there exist some well-established standards for modeling platform models, there is currently no matured foundation for specifying transformations between such models. In this paper, we propose a possible taxonomy for the classification of several existing and proposed model transformation approaches. The taxonomy is described with a feature model that makes the different design choices for model transformations explicit. Based on our analysis, we propose a few major categories in which most model transformation approaches fit.

1 Introduction

The Model-Driven Architecture (MDA) [MDA, Fra03] is an initiative by the Object Management Group (OMG) to define an approach to software development based on modeling and automated mapping of models to implementations. The basic MDA pattern involves defining a platform-independent model (PIM) and its automated mapping to one or more platform-specific models (PSMs).

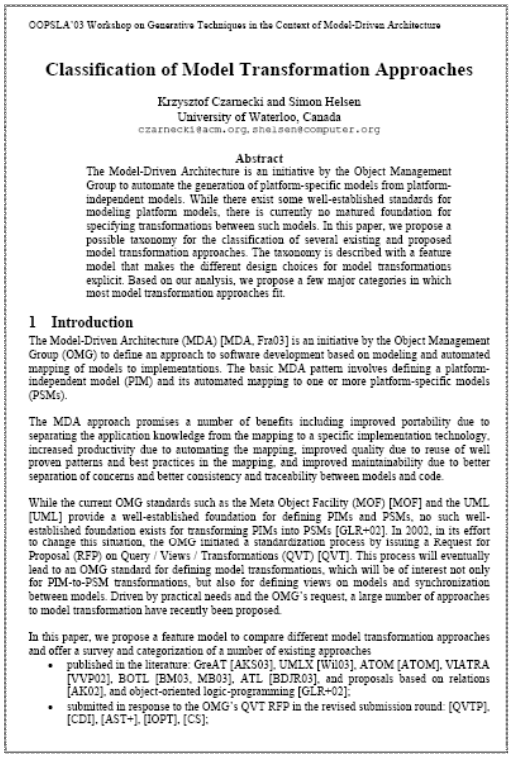
The MDA approach promises a number of benefits including improved portability due to separating the application knowledge from the mapping to a specific implementation technology, increased productivity due to automating the mapping, improved quality due to reuse of well proven patterns and best practices in the mapping, and improved maintainability due to better separation of concerns and better consistency and traceability between models and code.

While the current OMG standards such as the Meta Object Facility (MOF) [MOF] and the UML [UML] provide a well-established foundation for defining PIMs and PSMs, no such well-established foundation exists for transforming PIMs into PSMs [GLR+02]. In 2002, in its effort to change this situation, the OMG initiated a standardization process by issuing a Request for Proposal (RFP) on Query / Views / Transformations (QVT) [QVT]. This process will eventually lead to an OMG standard for defining model transformations, which will be of interest not only for PIM-to-PSM transformations, but also for defining views on models and synchronization between models. Driven by practical needs and the OMG's request, a large number of approaches to model transformation have recently been proposed.

In this paper, we propose a feature model to compare different model transformation approaches and offer a survey and categorization of a number of existing approaches

- published in the literature: GreAT [AKS03], UMLX [Wil03], ATOM [ATOM], VIATRA [VVP02], BOTL [BM03, MB03], ATL [BDJR03], and proposals based on relations [AK02], and object-oriented logic-programming [GLR+02];
- submitted in response to the OMG's QVT RFP in the revised submission round: [QVTP], [CDI], [AST+], [IOPT], [CS];

M. Antkiewicz, K. Czarnecki.
Design-Space of Heterogeneous Synchronization,
GTTSE'07 Summer School, Springer, 2008 (to appear),
pp. 40



OOPSLA'03 Workshop on Generative Techniques in the Context of Model-Driven Architecture

Classification of Model Transformation Approaches

Krzysztof Czarnecki and Simon Helsen
University of Waterloo, Canada
czarnecki@acm.org, shelsen@computer.org

Abstract

The Model-Driven Architecture is an initiative by the Object Management Group to automate the generation of platform-specific models from platform-independent models. While there exist some well-established standards for modeling platform models, there is currently no matured foundation for specifying transformations between such models. In this paper, we propose a possible taxonomy for the classification of several existing and proposed model transformation approaches. The taxonomy is described with a feature model that makes the different design choices for model transformations explicit. Based on our analysis, we propose a few major categories in which most model transformation approaches fit.

1 Introduction

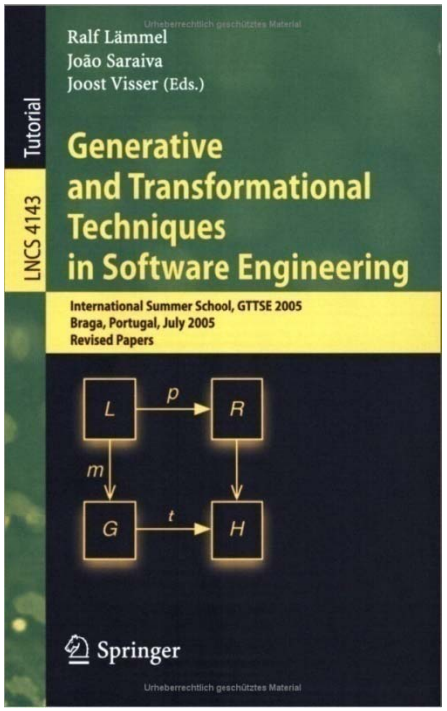
The Model-Driven Architecture (MDA) [MDA, Fra03] is an initiative by the Object Management Group (OMG) to define an approach to software development based on modeling and automated mapping of models to implementations. The basic MDA pattern involves defining a platform-independent model (PIM) and its automated mapping to one or more platform-specific models (PSMs).

The MDA approach promises a number of benefits including improved portability due to separating the application knowledge from the mapping to a specific implementation technology, increased productivity due to automating the mapping, improved quality due to reuse of well proven patterns and best practices in the mapping, and improved maintainability due to better separation of concerns and better consistency and traceability between models and code.

While the current OMG standards such as the Meta Object Facility (MOF) [MOF] and the UML [UML] provide a well-established foundation for defining PIMs and PSMs, no such well-established foundation exists for transforming PIMs into PSMs [GLR+02]. In 2002, in its effort to change this situation, the OMG initiated a standardization process by issuing a Request for Proposal (RFP) on Query / Views / Transformations (QVT) [QVT]. This process will eventually lead to an OMG standard for defining model transformations, which will be of interest not only for PIM-to-PSM transformations, but also for defining views on models and synchronization between models. Driven by practical needs and the OMG's request, a large number of approaches to model transformation have recently been proposed.

In this paper, we propose a feature model to compare different model transformation approaches and offer a survey and categorization of a number of existing approaches

- published in the literature: GreAT [AKS03], UMLX [Wil03], ATOM [ATOM], VIATRA [VVP02], BOTL [BM03, MB03], ATL [BDJR03], and proposals based on relations [AK02], and object-oriented logic-programming [GLR+02];
- submitted in response to the OMG's QVT RFP in the revised submission round: [QVTP], [CDI], [AST+], [IOPT], [CS];



Michal Antkiewicz
Herman Lee
Matthew Stephan

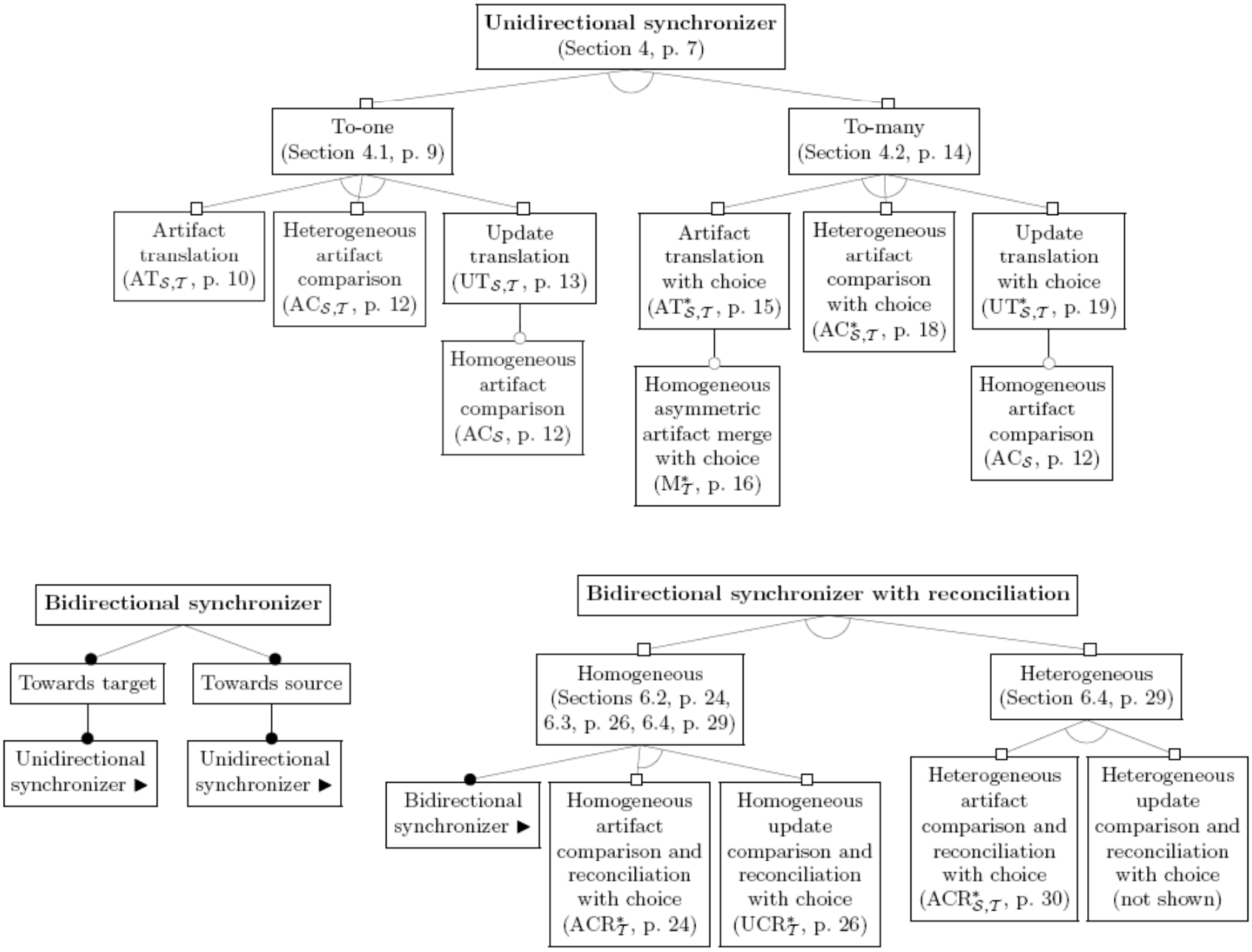
1-1, 1-N, M-N relations

with or w/o reconciliation

different implementation strategies

13 scenarios

12 primitives


Unidirectional synchronizer
(Section 4, p. 7)
To-one
(Section 4.1, p. 9)
To-many
(Section 4.2, p. 14)
Artifact translation ($AT_{\mathcal{S},\mathcal{T}}$, p. 10)
Heterogeneous artifact comparison ($AC_{\mathcal{S},\mathcal{T}}$, p. 12)
Update translation ($UT_{\mathcal{S},\mathcal{T}}$, p. 13)
Homogeneous artifact comparison ($AC_{\mathcal{S}}$, p. 12)
Artifact translation with choice ($AT^{*}_{\mathcal{S},\mathcal{T}}$, p. 15)
Heterogeneous artifact comparison with choice ($AC^{*}_{\mathcal{S},\mathcal{T}}$, p. 18)
Update translation with choice ($UT^{*}_{\mathcal{S},\mathcal{T}}$, p. 19)
Homogeneous asymmetric artifact merge with choice ($M^{*}_{\mathcal{T}}$, p. 16)
Homogeneous artifact comparison ($AC_{\mathcal{S}}$, p. 12)
Bidirectional synchronizer
Towards target
Towards source
Unidirectional synchronizer ▶
Unidirectional synchronizer ▶
Bidirectional synchronizer with reconciliation
Homogeneous
(Sections 6.2, p. 24, 6.3, p. 26, 6.4, p. 29)
Heterogeneous
(Section 6.4, p. 29)
Bidirectional synchronizer ▶
Homogeneous artifact comparison and reconciliation with choice ($ACR^{*}_{\mathcal{T}}$, p. 24)
Homogeneous update comparison and reconciliation with choice ($UCR^{*}_{\mathcal{T}}$, p. 26)
Heterogeneous artifact comparison and reconciliation with choice ($ACR^{*}_{\mathcal{S},\mathcal{T}}$, p. 30)
Heterogeneous update comparison and reconciliation with choice (not shown)

problems

update as a partial function

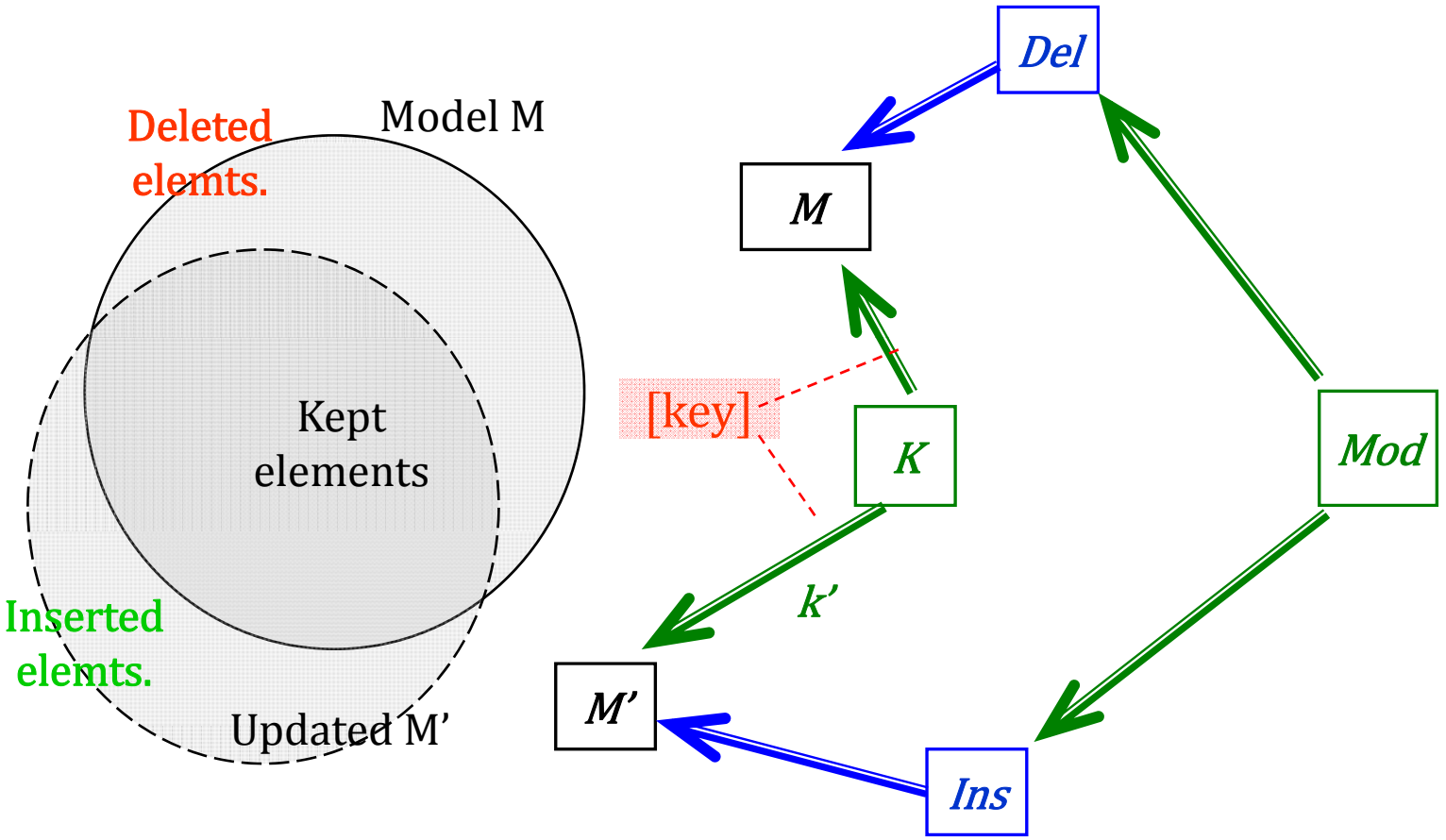
operational rather than declarative

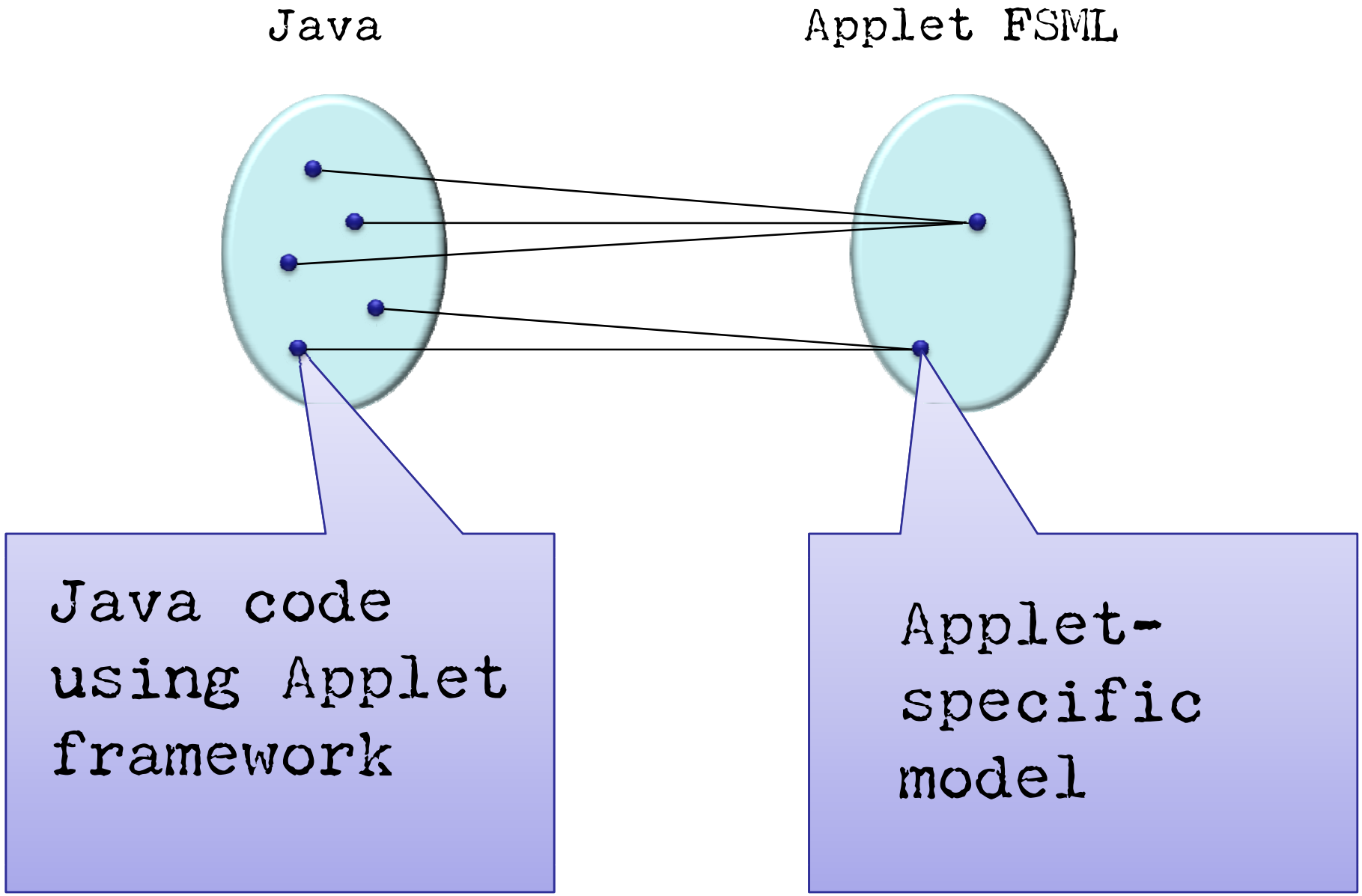
missing structure for reconciliation

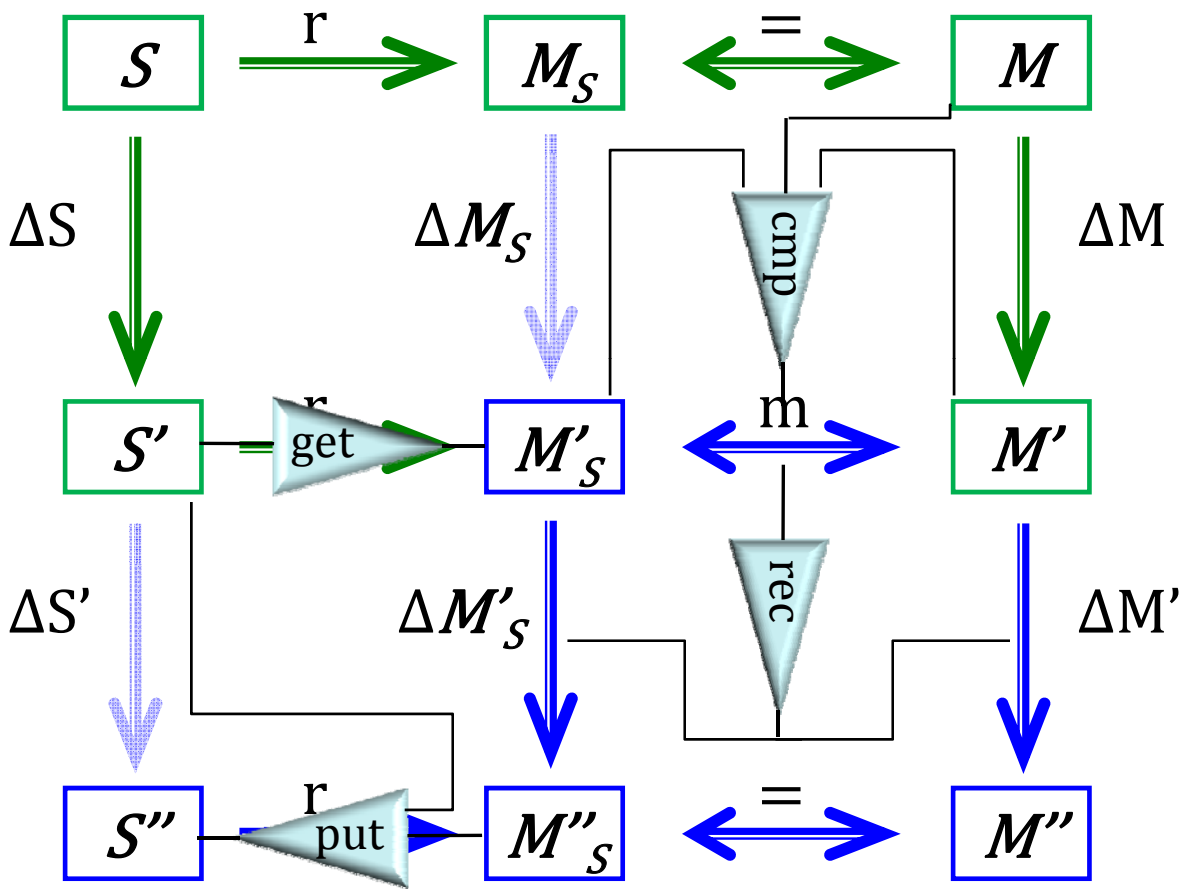
too many primitives (12)

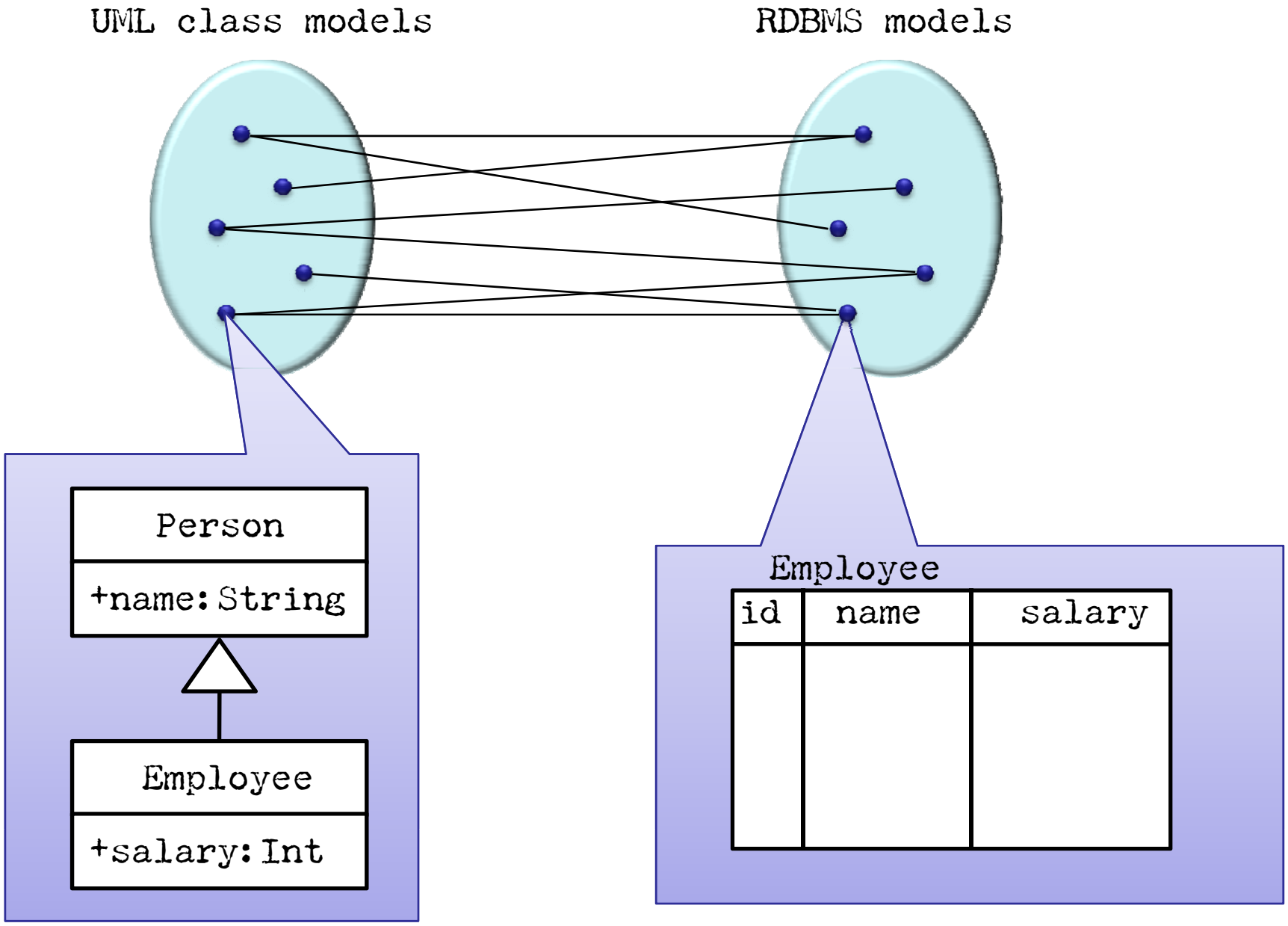
cathegorical redesign

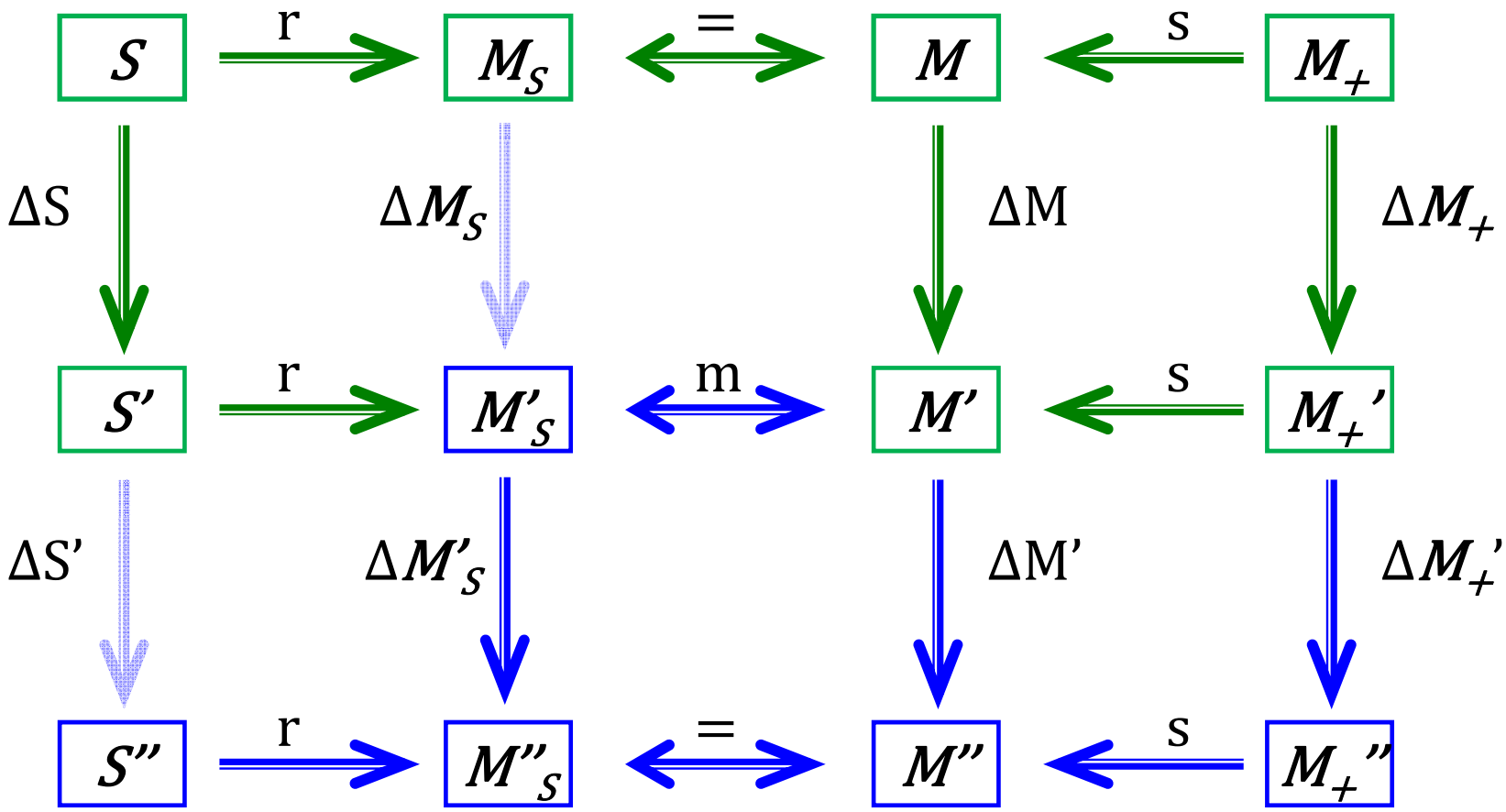
update and matches


Deleted
elemts.
Model M
Kept
elements
Inserted
elemts.
Updated M'
Del
M
[key]
K
Mod
k'
M'
Ins


Java
Applet FSML
Java code using Applet framework
Applet-specific model


S
r
M_S
=
M
ΔS
ΔM_S
cmp
ΔM
S'
r
get
M'_S
m
M'
ΔS'
$\Delta M'_S$
rec
ΔM'
S''
r
put
M''_S
=
M''


UML class models
RDBMS models
Person
+name: String
Employee
+salary: Int
Employee
id
name
salary


S
r
M_S
=
M
s
M_+
ΔS
ΔM_S
ΔM
ΔM_+
S'
r
M'_S
m
M'
s
M_+'
ΔS'
$\Delta M'_S$
ΔM'
$\Delta M_+'$
S''
r
M''_S
=
M''
s
M_+''

declarative formulation

more intuitive and richer
update and match definition

richer structure to formulate
algorithms

handling of M-to-N case

discussion topic:

Properties of trafo design space

Tech talk:

Bidirectional trafo for Framework Specific Modeling Languages (FSMLs)

Looking for PhD students...