Runtime Adaptation of Architectural Models: an approach for adapting User Interfaces

Diego Rodríguez-Gracia, **Javier Criado**,
Luis Iribarne, Nicolás Padilla
Applied Computing Group
University of Almería, Spain

Cristina Vicente-Chicote
Dpt. of Information Communication Technologies
Technical University of Cartagena, Spain

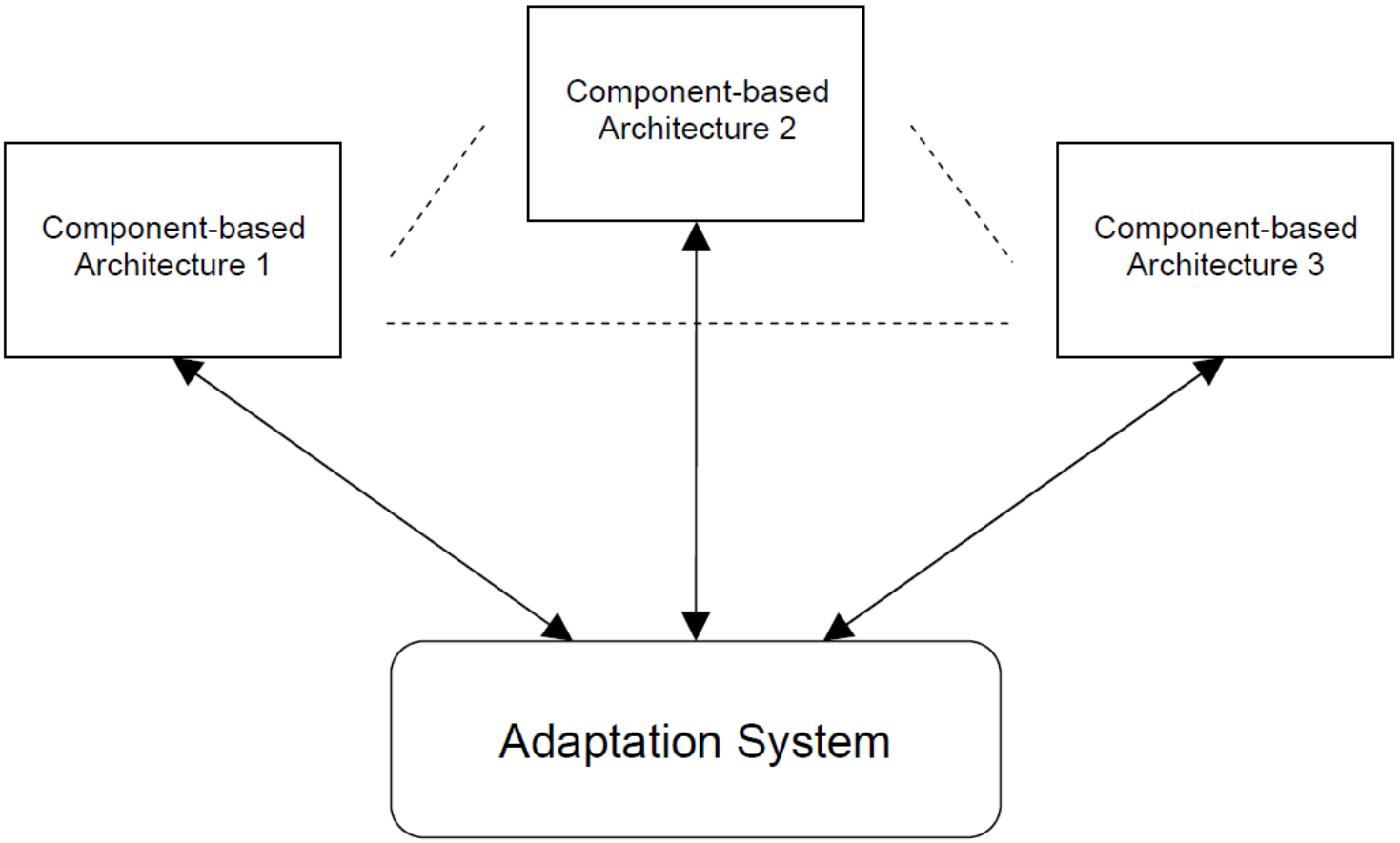
ACG
Applied Computing Group

2nd International Conference in Model & Data Engineering (MEDI'2012)

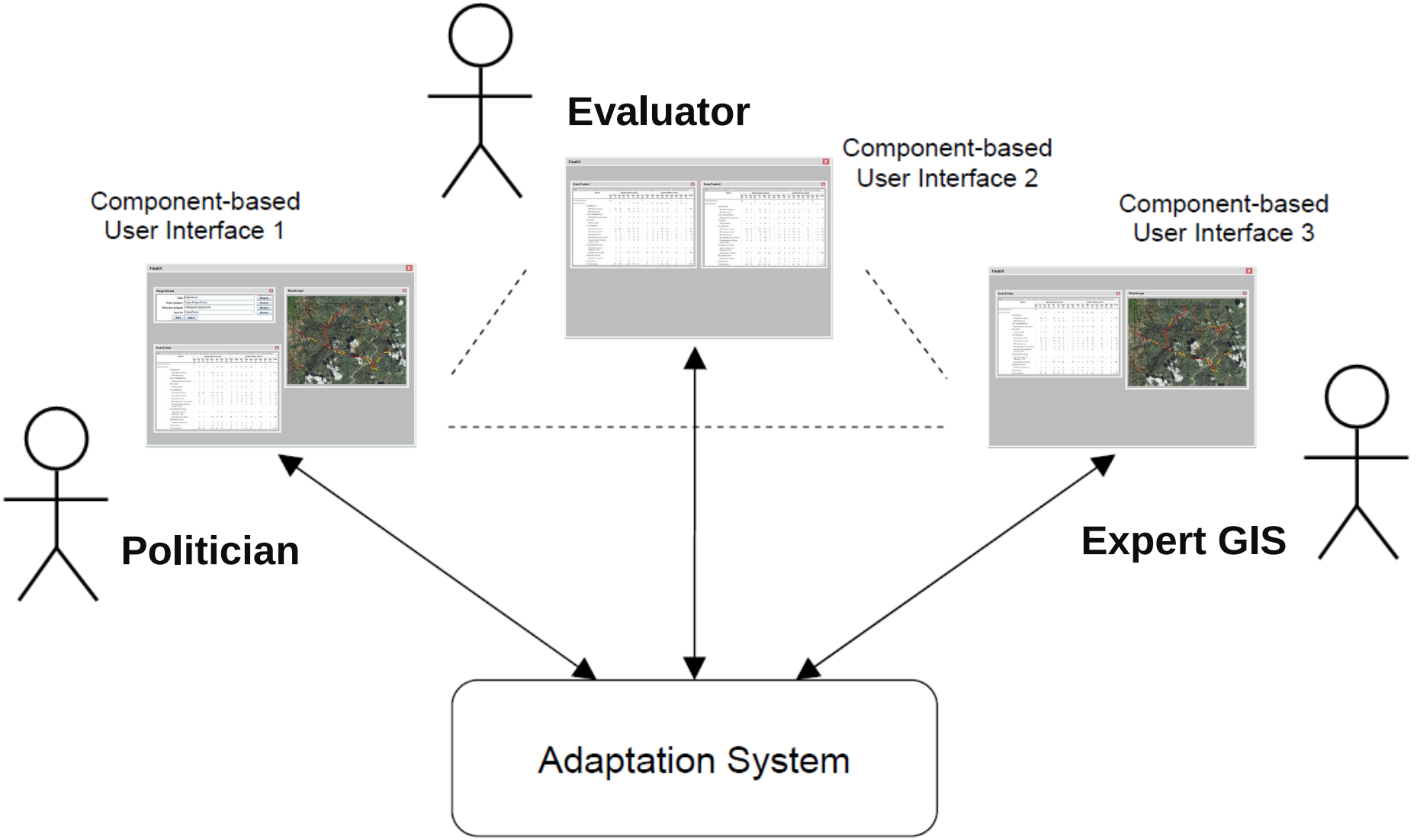
Outline

- Context
- Our goal
- User Interface Adaptation
  - Model Transformation Pattern
  - Adaptive Model Transformation
  - Transformation Rules
  - Rule Selection
  - Rule Selection Log
  - Rule Transformation
- Conclusions and future work

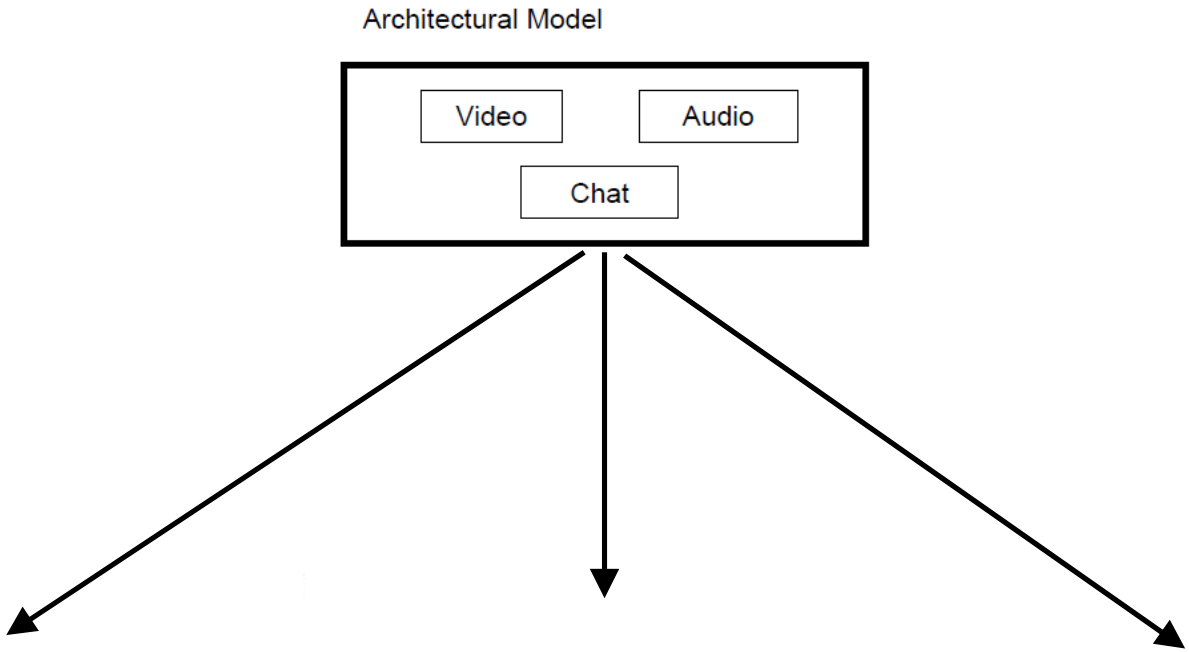
Context


Component-based Architecture 2
Component-based Architecture 1
Component-based Architecture 3
Adaptation System

Context


Evaluator
Component-based
User Interface 1
Component-based
User Interface 2
Component-based
User Interface 3
Politician
Expert GIS
Adaptation System

Context



Our goal

- **Adaptation of User Interfaces at runtime**

- UIs are **described by means of architectural models** that contain the specification of **user interface components**

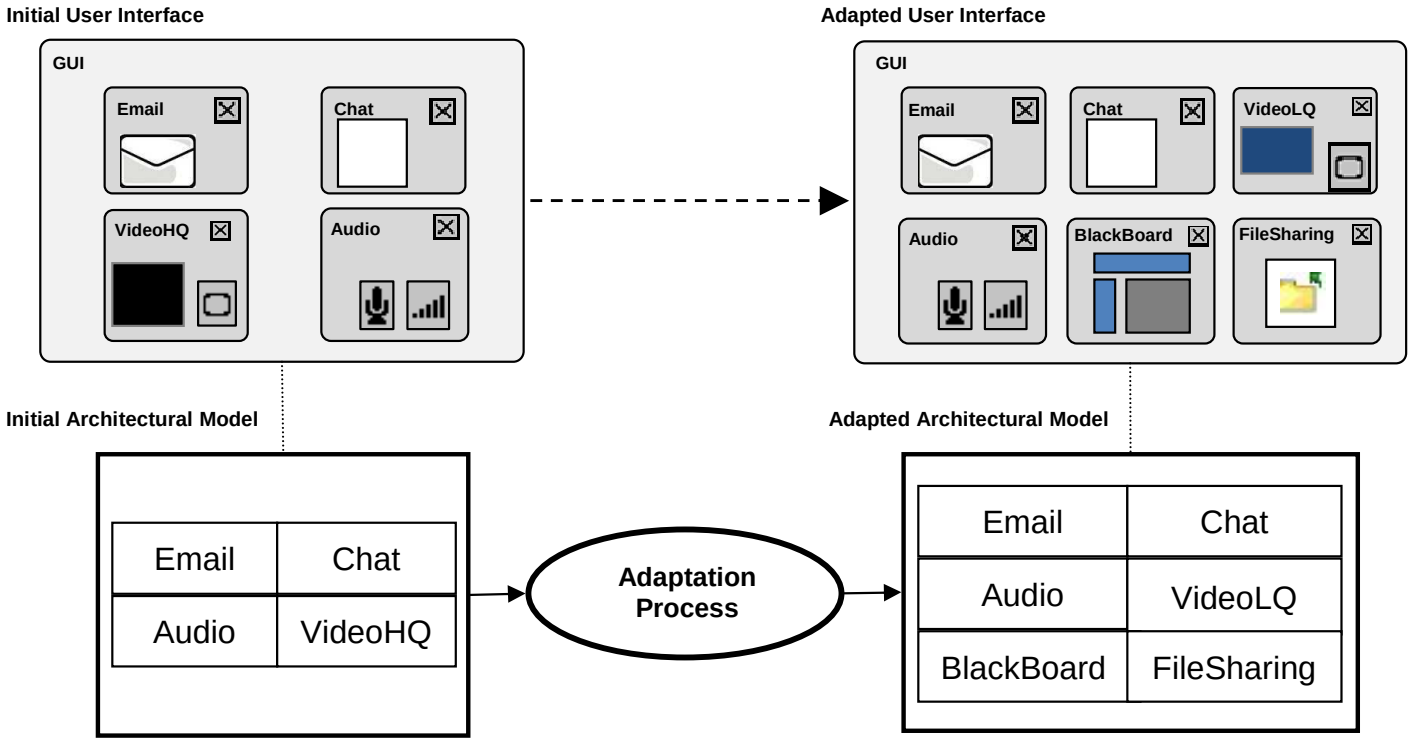
- Architectural Models **can vary at runtime due to changes in the context** (user interaction, a temporal event, etc)

Our goal

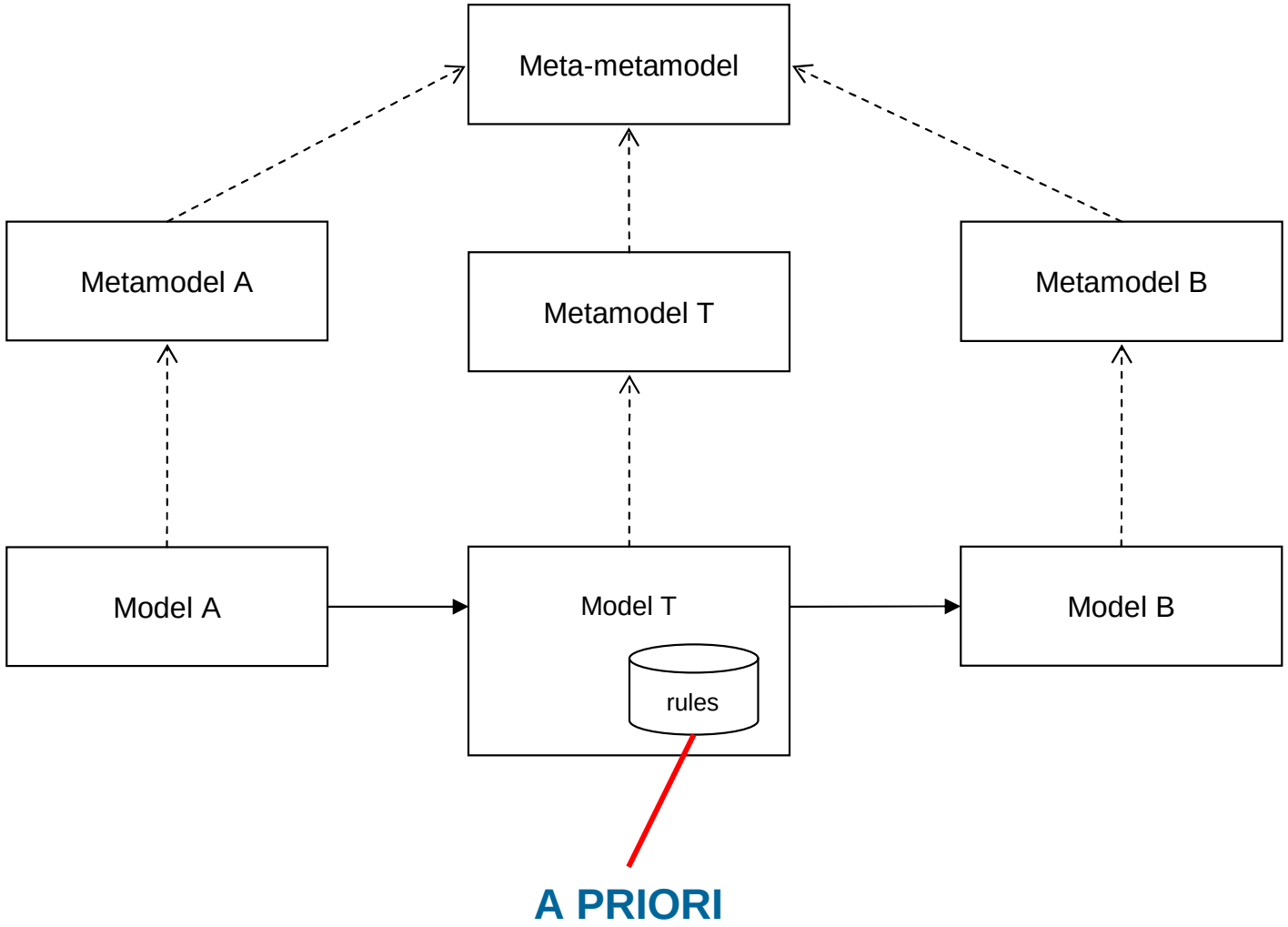
- **Adaptation of User Interfaces at runtime**

- UIs are **described by means of architectural models** that contain the specification of **user interface components**

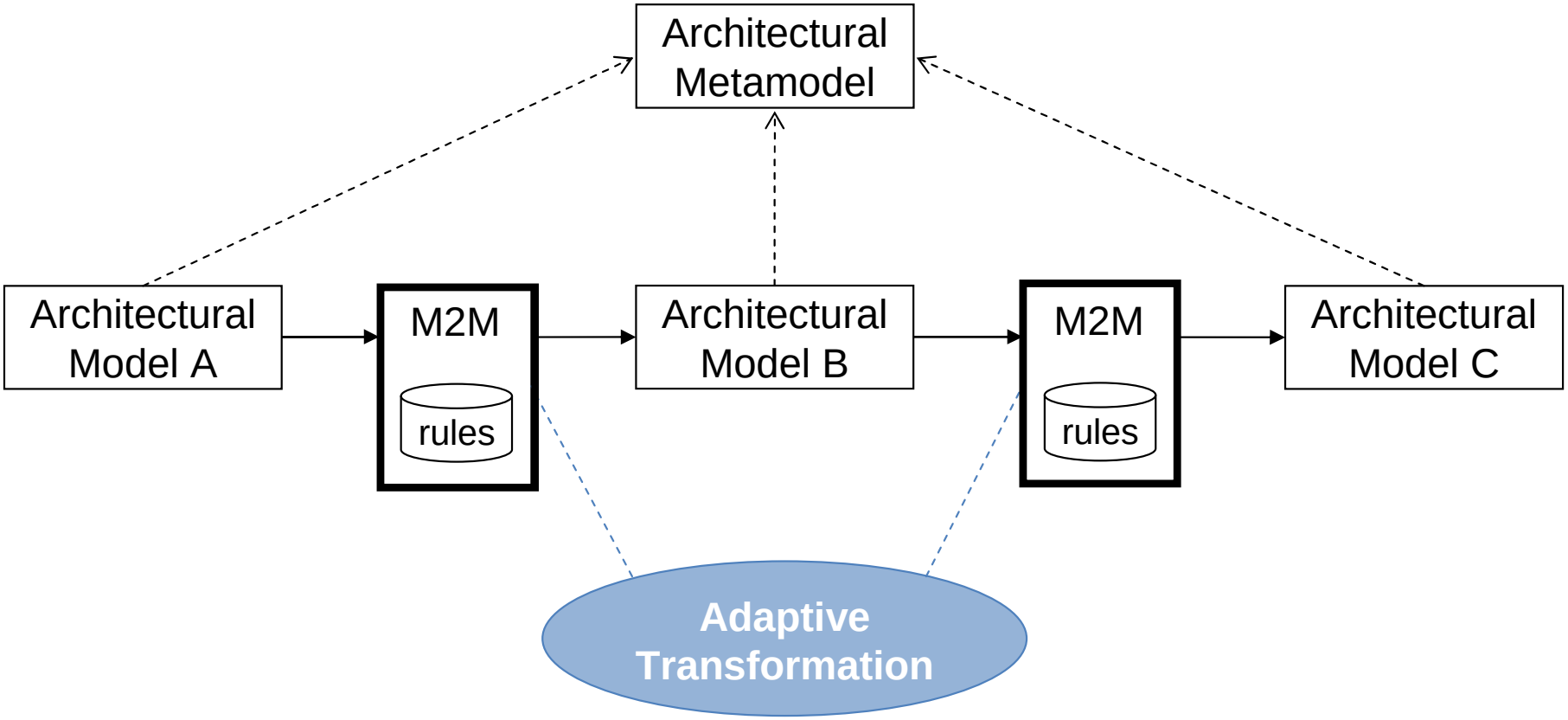
- Architectural Models **can vary at runtime due to changes in the context** (user interaction, a temporal event, etc.)



Our goal


Meta-metamodel
Metamodel A
Metamodel T
Metamodel B
Model A
Model T
rules
Model B
A PRIORI

Our goal

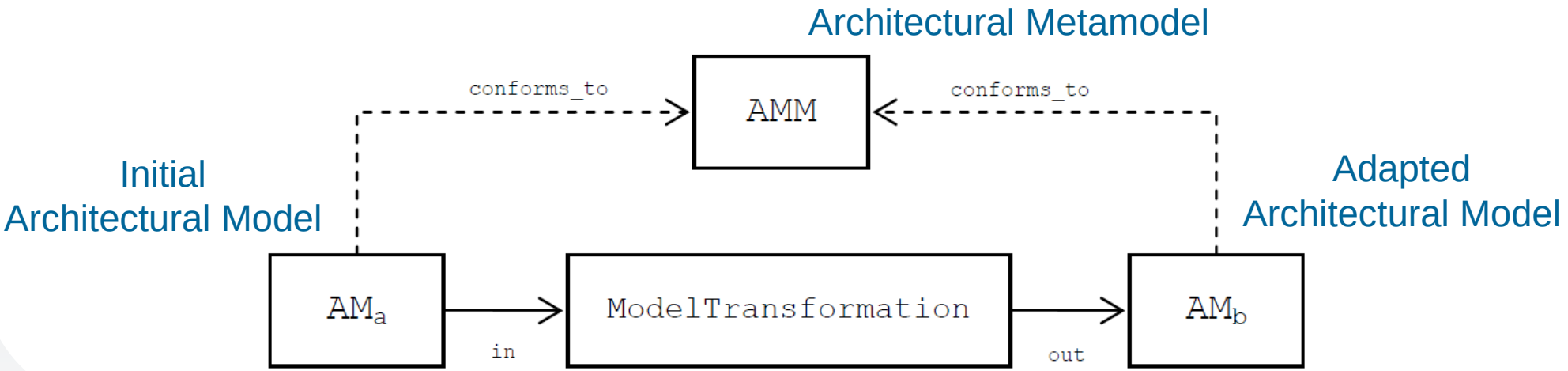

Architectural Metamodel
Architectural Model A
M2M
rules
Architectural Model B
M2M
rules
Architectural Model C
Adaptive Transformation

User Interface Adaptation

- **Adaptation** of **architectural models**
- **@Runtime**
- Using **M2M** transformations
- **Transformations** are also **adapted** at runtime.

  Model Transformations **not prepared a priori**

  M2M is **dynamically composed** from a rule model



User Interface Adaptation

- Build a **pattern/template** for modeling our adaptation schema

- Build a **Rule Repository**

- Design a **Rule Selection** process as an M2M
  - This selection process can generate different rule subsets

- Design a **Rule Selection Log** process as an M2M
  - This process updates the repository with selection information

- Develop a **Rule Transformation** process as an M2T
  - This process generates the model transformation

Model Transformation Pattern

- **Model the structure and composition** of our transformation schema elements

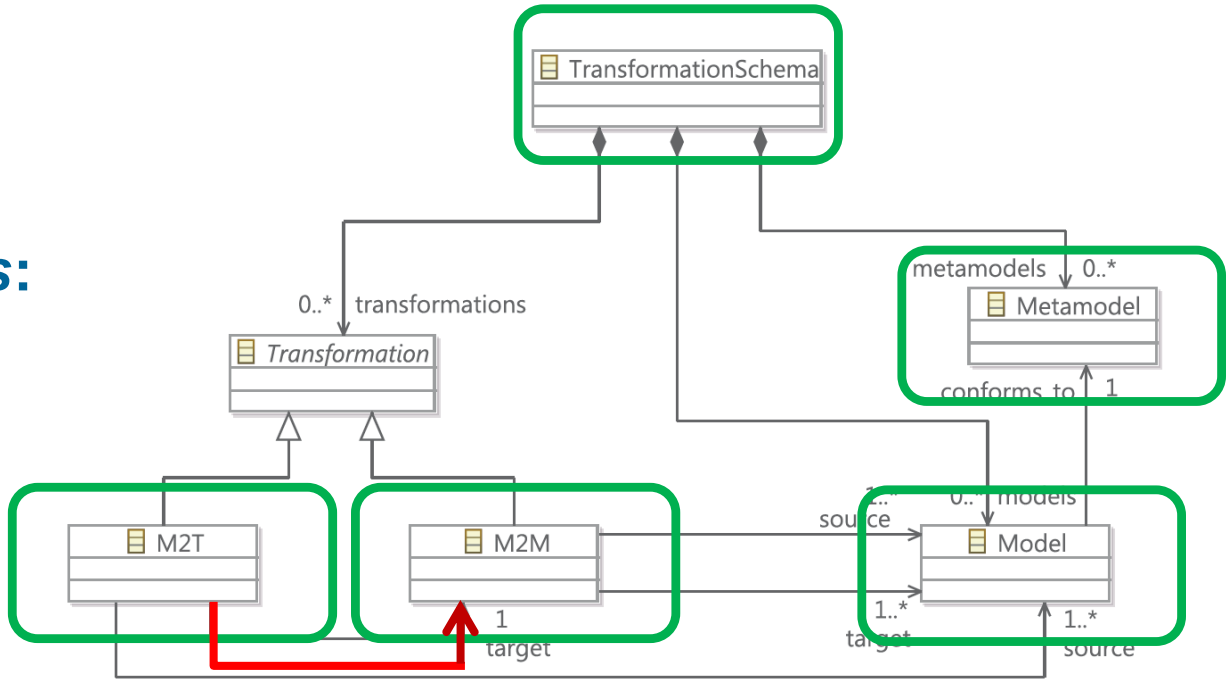
- **Possibility of changing** our adaptation schema

- Elements:

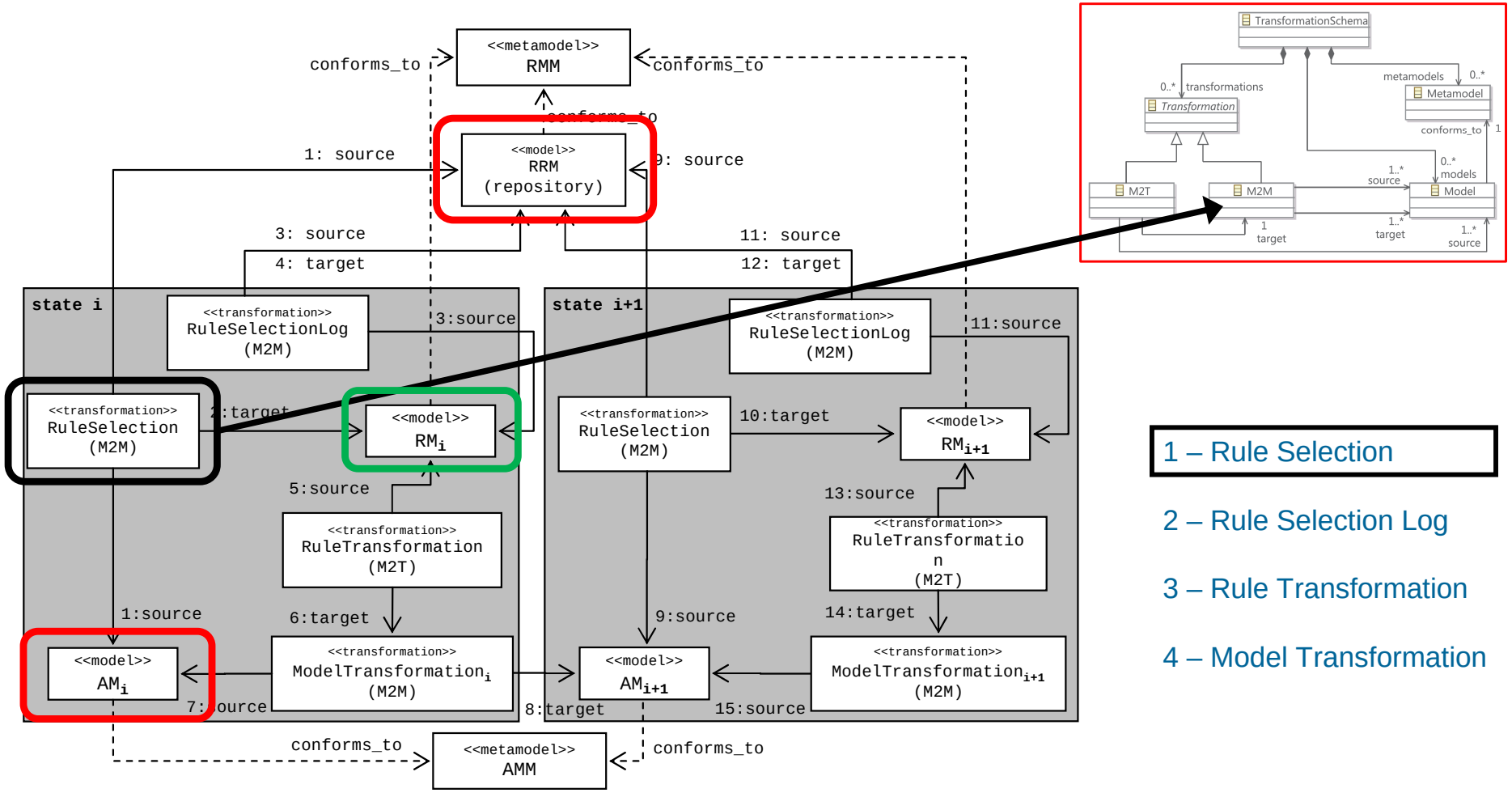
  - **TransformationSchema**
  - **Metamodel**
  - **Model**
  - ***Transformations*:**

    M2M

    M2T



Adaptive Model Transformation



1 – Rule Selection

2 – Rule Selection Log

3 – Rule Transformation

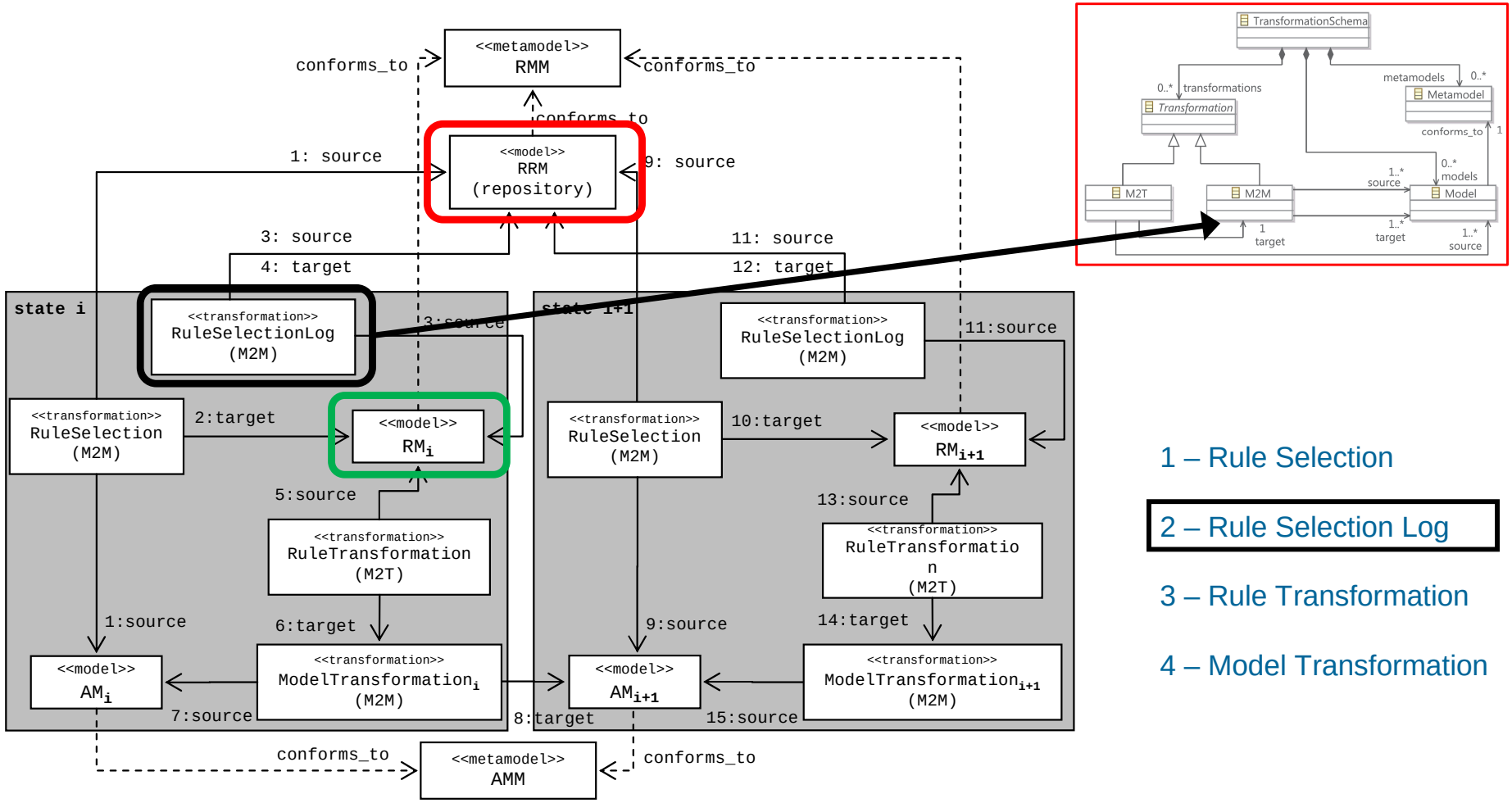
4 – Model Transformation

1º **Rule Selection**: is obtained as an instance of the M2M concept

Input: the **repository model (RRM)** and the initial **architectural model (AM_i)**

Output: the **selected rules model (RM_i)**

Adaptive Model Transformation



1 – Rule Selection

2 – Rule Selection Log

3 – Rule Transformation

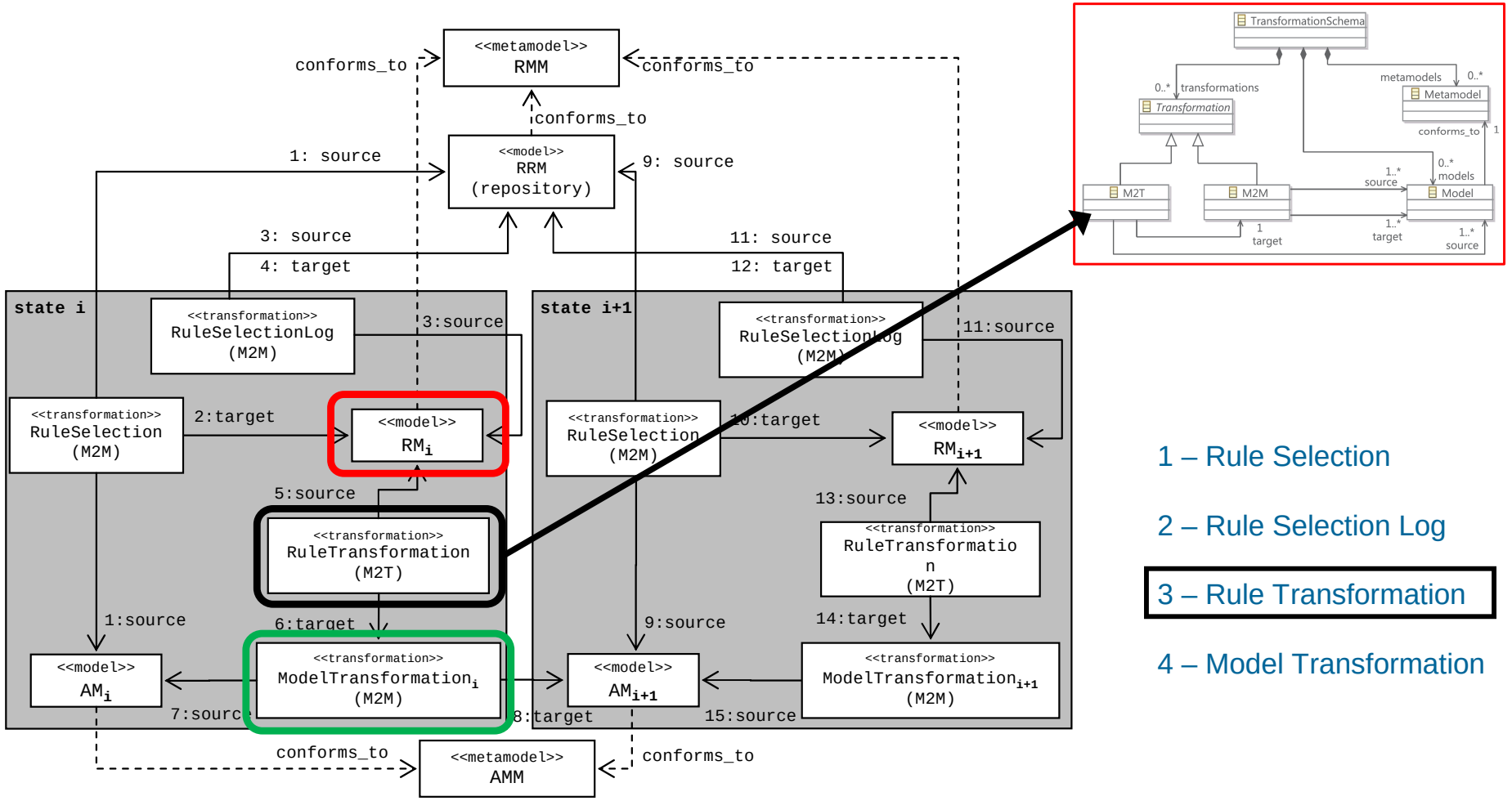
4 – Model Transformation

2º **Rule Selection Log**: is obtained as an instance of the M2M concept

Input: the **selected rules model (RM_i)** and the **rule repository model (RRM)**

Output: the **updated rule repository model (RRM)**

Adaptive Model Transformation



1 – Rule Selection

2 – Rule Selection Log

3 – Rule Transformation

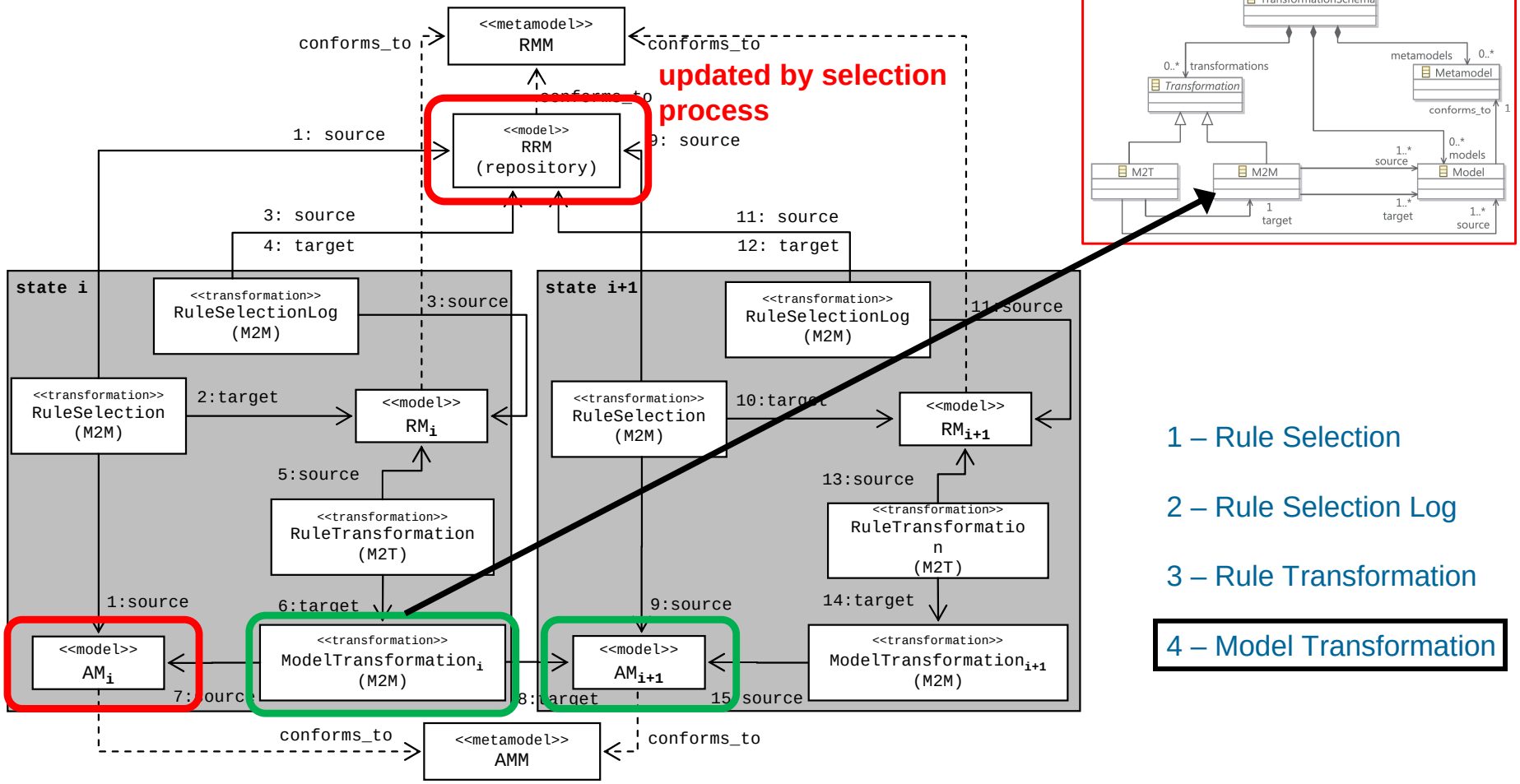
4 – Model Transformation

3º **Rule Transformation**: is obtained as an instance of the M2T concept

Input: the **selected rules model (RM_i)**

Output: the transformation for architectural model **($ModelTransformation_i$)**

Adaptive Model Transformation



1 – Rule Selection

2 – Rule Selection Log

3 – Rule Transformation

4 – Model Transformation

4º **Model Transformation**: is an instance of the M2M concept

Input: the **initial architectural model (AM_i)**

Output: the **new architectural model (AM_{i+1})**

Transformation Rules

- **Metamodel** for transformation rules

- Rule Repository Model **(RRM)**
- Selected rules model **(RM_i)**

- Rule attributes:

Transformation Rules

- **Metamodel** for transformation rules

- Rule Re

- Selected

- Rule attr

rule_name	purpose	is_priority	weight	ratio	run_c	selec_c
Insert_Chat	InsertChat	true	2.0	1.0	3	3
Insert_Audio	InsertAudio	false	4.0	1.0	2	2
Insert_Video1	InsertVideoLowQ	false	11.0	0.5	2	1
Insert_Video2	InsertVideoLowQ	true	6.0	0.5	2	1
Insert_Video3	InsertVideoHighQ	false	9.0	1.0	3	3
Insert_BlackBoard1	InsertBlackBoard	false	6.0	0.5	2	1
Insert_BlackBoard2	InsertBlackBoard	false	4.0	0.5	2	1
Insert_FileSharing	InsertFileSharing	false	3.0	0.0	0	0
Delete_Chat	DeleteChat	true	3.0	0.0	0	0
Delete_Audio	DeleteAudio	true	3.0	0.0	0	0
Delete_Video	DeleteVideo	true	3.0	0.0	0	0
Delete_BlackBoard	DeleteBlackBoard	true	3.0	0.0	0	0
Delete_FileSharing	DeleteFileSharing	true	3.0	0.0	0	0

rule_name: unique; identifies the rule

purpose: indicates the purpose of the rule

is_priority: boolean; if its value is true, the rule must be selected.

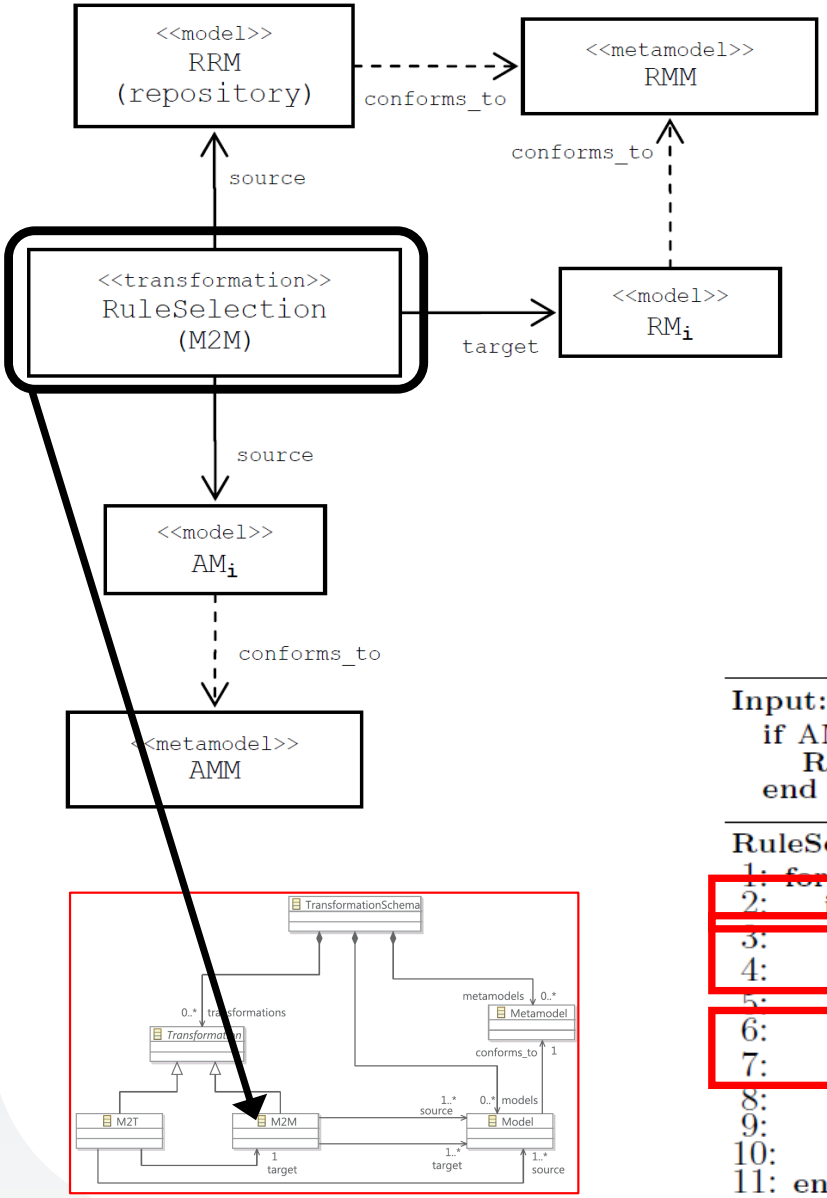
weight: the selection process uses this attribute to select the rules

run_counter: number of times the *purpose* of the rule matches adaptation *purposes*

selection_counter: number of times the rule has been selected

ratio: frequency of using a transformation rule

Rule Selection



Instance of the **M2M concept**

The **process starts** when the system detects that it is necessary an adaptation

Input: Architectural Model (**AMi**), Rule Repository Model (**RRM**)

Output: Selected Rules Model (**RMi**)

Input: AM_i and RRM **Output:** RM_i

```
if AM_i!Launcher.running = true then
    RuleSelection
end if
```

RuleSelection

```
1: for n = 1 .. RRM.size do
2:     if AM_i!Launcher.purposes[EString_1..EString_n] contains RRM!Rule_n.purpose then
3:         if RRM!Rule_n.is_priority = true then
4:             RM_i.add(RRM!Rule_n)
5:         else
6:             if ∃! j, j ∈ 1..RRM.size/ RRM!Rule_j.weight > RRM!Rule_n.weight then
7:                 RM_i.add(RRM!Rule_n)
8:             end if
9:         end if
10:    end if
11: end for
```

Rule Selection

Example:

We have an Architectural Model (AM_i) where **Launcher.purposes = ['DeleteVideo', 'InsertVideoLowQ', 'InsertBlackBoard', 'InsertFileSharing'].**

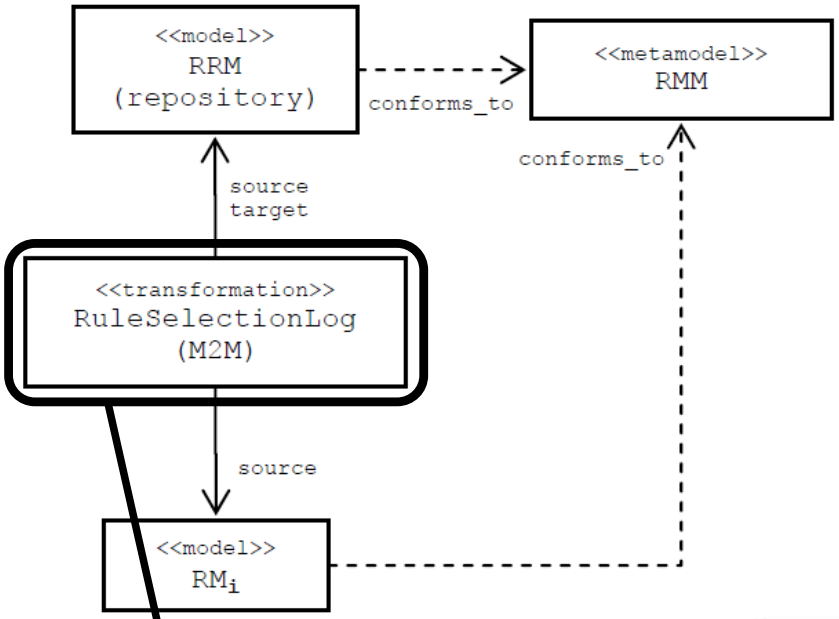
Rule Repository Model (RRM):

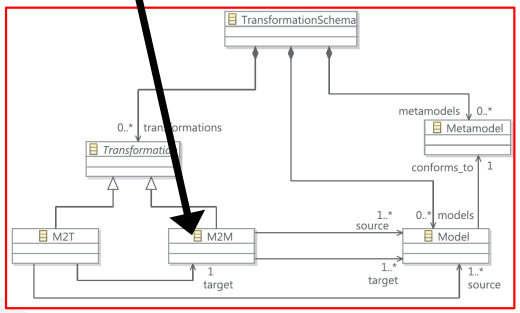
rule_name	purpose	is_priority	weight	ratio	run_c	selec_c
Insert_Chat	InsertChat	true	2.0	1.0	3	3
Insert_Audio	InsertAudio	false	4.0	1.0	2	2
Insert_Video1	InsertVideoLowQ	false	11.0	0.5	2	1
Insert_Video2	InsertVideoLowQ	true	6.0	0.5	2	1
Insert_Video3	InsertVideoHighQ	false	9.0	1.0	3	3
Insert_BlackBoard1	InsertBlackBoard	false	6.0	0.5	2	1
Insert_BlackBoard2	InsertBlackBoard	false	4.0	0.5	2	1
Insert_FileSharing	InsertFileSharing	false	3.0	0.0	0	0
Delete_Chat	DeleteChat	true	3.0	0.0	0	0
Delete_Audio	DeleteAudio	true	3.0	0.0	0	0
Delete_Video	DeleteVideo	true	3.0	0.0	0	0
Delete_BlackBoard	DeleteBlackBoard	true	3.0	0.0	0	0
Delete_FileSharing	DeleteFileSharing	true	3.0	0.0	0	0

Selected Rules Model **(RM_i) is generated:**

rule_name	purpose	is_priority	weight
Insert_Video2	InsertVideoLowQ	true	6.0
Insert_BlackBoard1	InsertBlackBoard	false	6.0
Insert_FileSharing	InsertFileSharing	false	3.0
Delete_Video	DeleteVideo	true	3.0

Rule Selection Log





Instance of the **M2M concept**

The **process starts** after ***RuleSelection***

Input: Selected Rules Model (**RMi**), Rule Repository Model (**RRM**)

Output: Updated Rule Repository Model (**RRM**)

process *RSL*

1: for $n = 1 \rightarrow RRM.size$ do
2: if RM_i contains $RRM[n]$ then
3: $RRM[n].run\_counter \leftarrow RRM[n].run\_counter + 1$
4: $RRM[n].selection\_counter \leftarrow RRM[n].selection\_counter + 1$
5: $RRM[n].ratio \leftarrow RRM[n].selection\_counter / RRM[n].run\_counter$
6: $RRM[n].weight \leftarrow RRM[n].weight + RRM[n].ratio * RRM.bonus$
7: else
8: if $RM_i.action = RRM[n].action$ then
9: $RRM[n].run\_counter \leftarrow RRM[n].run\_counter + 1$
10: $RRM[n].ratio \leftarrow RRM[n].selection\_counter / RRM[n].run\_counter$
11: $RRM[n].weight \leftarrow RRM[n].weight - RRM[n].ratio * RRM.penalty$
12: end if
13: end if
14: end for

Rule Selection Log

Example:

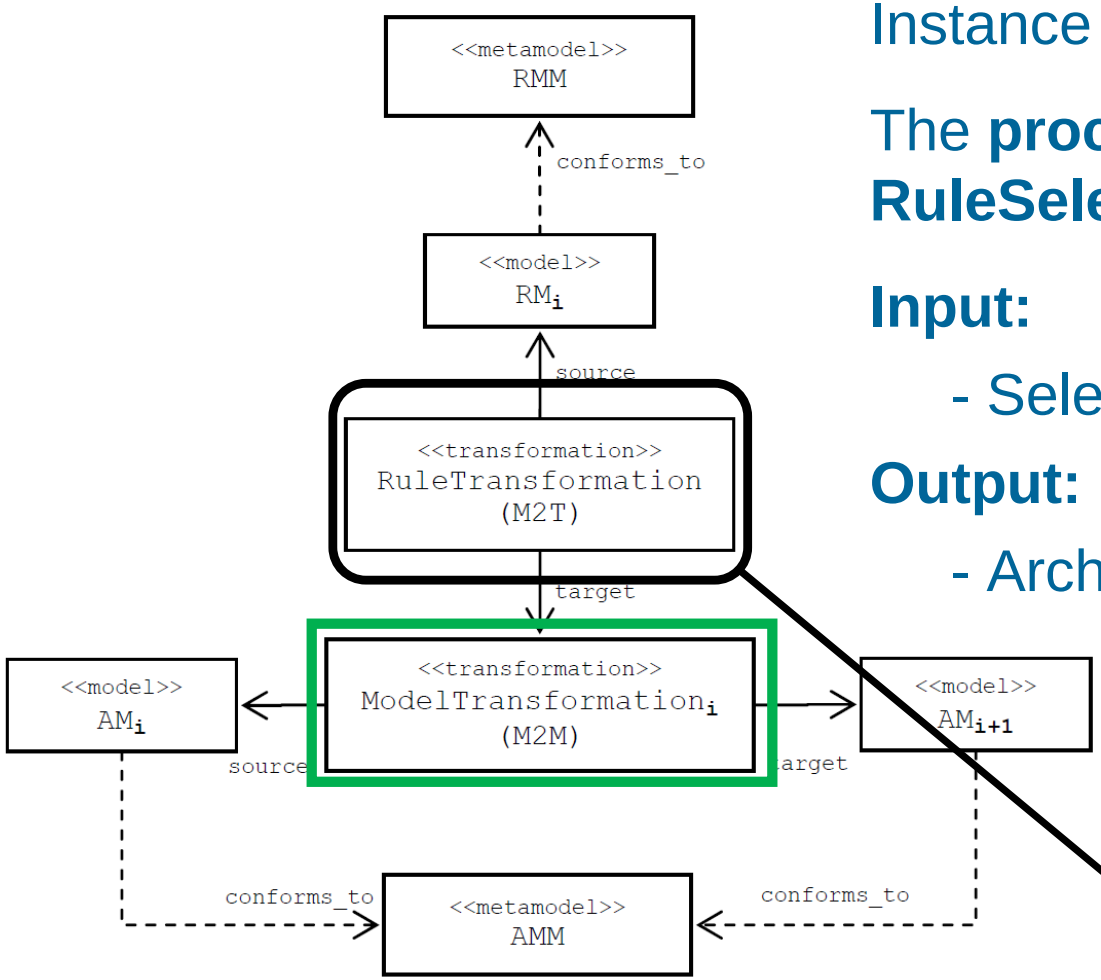
Selected Rules Model **(RM_i):**

rule_name	purpose	is_priority	weight
Insert_Video2	InsertVideoLowQ	true	6.0
Insert_BlackBoard1	InsertBlackBoard	false	6.0
Insert_FileSharing	InsertFileSharing	false	3.0
Delete_Video	DeleteVideo	true	3.0

Updated Rule Repository Model (RRM):

rule_name	purpose	weight	ratio	run_c	selec_c
Insert_Chat	InsertChat	2.0	1.0	3	3
Insert_Audio	InsertAudio	4.0	1.0	2	2
Insert_Video1	InsertVideoLowQ	11.0→10.33	0.5→0.33	2→3	1
Insert_Video2	InsertVideoLowQ	6.0→7.65	0.5→0.66	2→3	1→2
Insert_Video3	InsertVideoHighQ	9.0	1.0	3	3
Insert_BlackBoard1	InsertBlackBoard	6.0→7.65	0.5→0.66	2→3	1→2
Insert_BlackBoard2	InsertBlackBoard	4.0→3.33	0.5→0.33	2→3	1
Insert_FileSharing	InsertFileSharing	3.0→5.5	0→1.0	0→1	0→1
Delete_Chat	DeleteChat	3.0	0.0	0	0
Delete_Audio	DeleteAudio	3.0	0.0	0	0
Delete_Video	DeleteVideo	3.0→5.5	0.0→1.0	0→1	0→1
Delete_BlackBoard	DeleteBlackBoard	3.0	0.0	0	0
Delete_FileSharing	DeleteFileSharing	3.0	0.0	0	0

Rule Transformation



Instance of the **M2T concept**

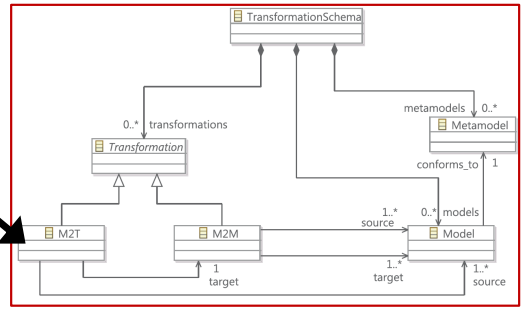
The **process starts** after **RuleSelectionLog**

Input:

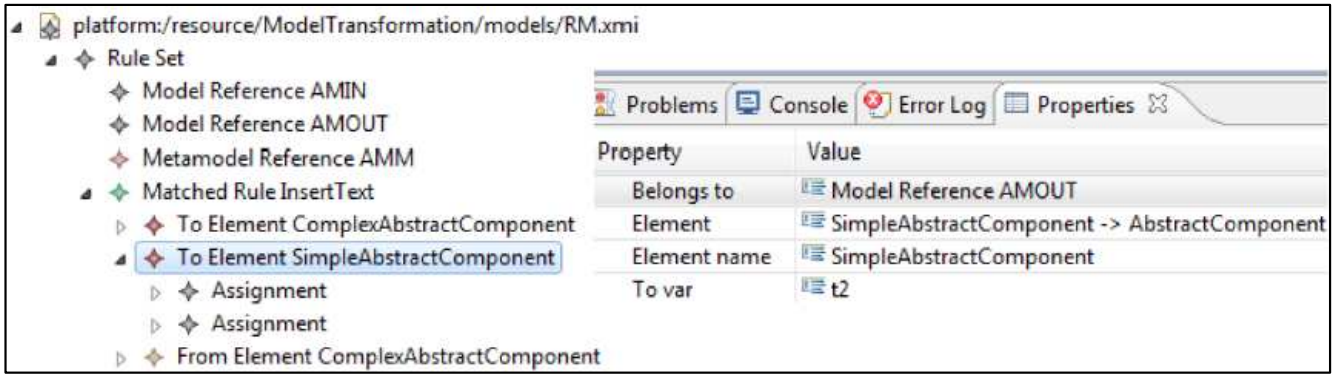
- Selected Rules Model (**RM_i**)

Output:

- Architectural model transformation (**$ModelTransformation_i$**)



Rule Transformation



Selected Rules Model (RMi)

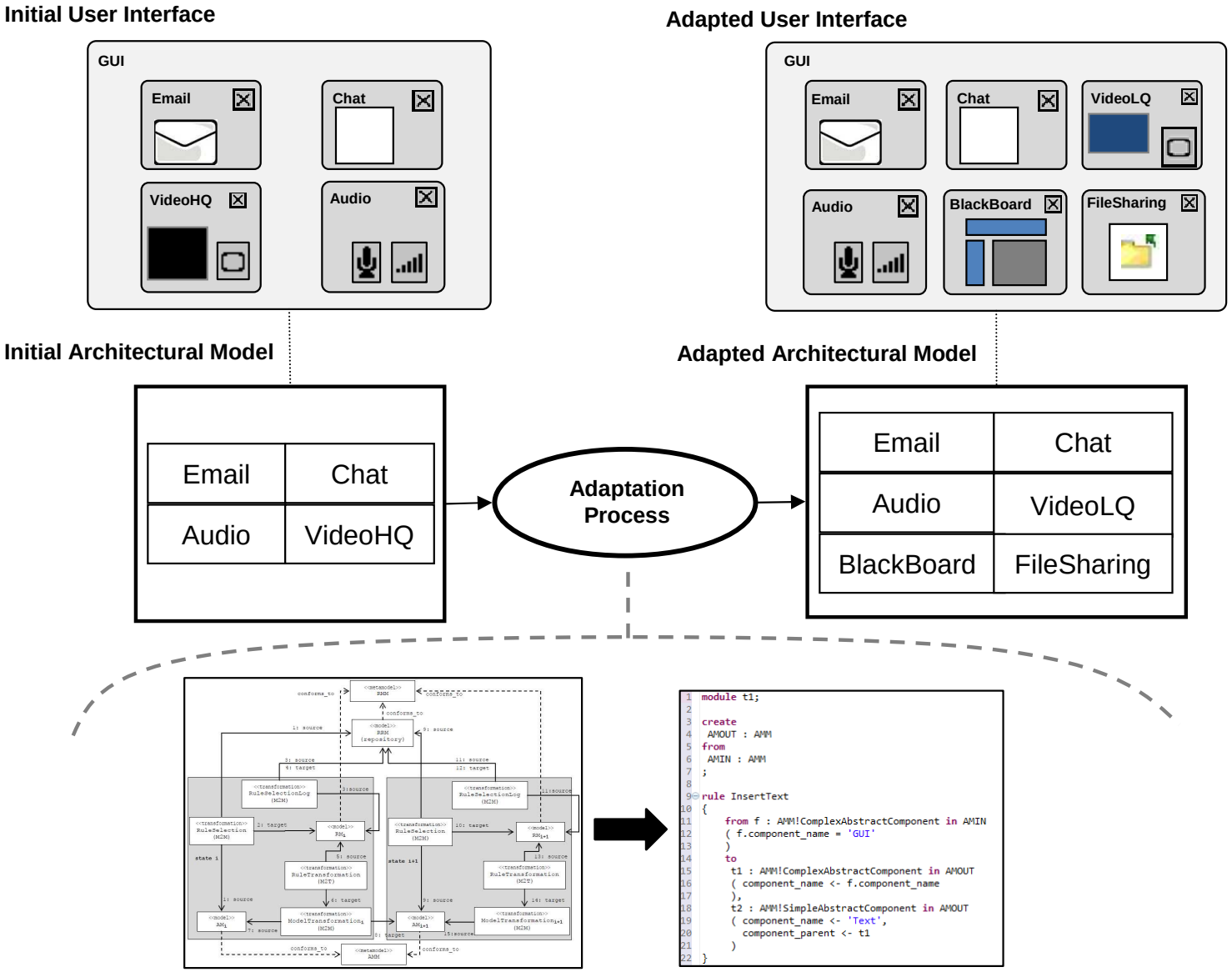
RuleTransformation (JET)

```
module t1;
create
<c:iterate var="model_ref" select="/RuleSet/model_ref[@model_type = 'OUT']" delimiter=",">
 <c:get select="$model_ref/@model_name"/> :
 <c:get select="$model_ref/conforms_to/@metamodel_name"/></c:iterate>
from
<c:iterate var="model_ref" select="/RuleSet/model_ref[@model_type = 'IN']" delimiter=",">
 <c:get select="$model_ref/@model_name"/> :
 <c:get select="$model_ref/conforms_to/@metamodel_name"/></c:iterate>;
<c:iterate var="rules" select="/RuleSet/rules">
 <c:if test="$rules[self :: MatchedRule or self :: LazyRule]">
<c:if test="$rules[self :: MatchedRule]">rule</c:if>
<c:if test="$rules[self :: LazyRule]">lazy rule</c:if>
<c:get select="$rules/@rule_name"/>
{
 <c:include template="templates/fromElements.jet" passVariables="rules"/>
 <c:include template="templates/toElements.jet" passVariables="rules"/>
 <c:include template="templates/doBlock.jet" passVariables="rules"/>
}
 </c:if>
</c:iterate>
```

ModelTransformation (ATL)

```
1  module t1;
2
3  create
4   AMOUT : AMM
5  from
6   AMIN : AMM
7  ;
8
9  rule InsertText
10 {
11     from f : AMM!ComplexAbstractComponent in AMIN
12     ( f.component_name = 'GUI'
13     )
14     to
15      t1 : AMM!ComplexAbstractComponent in AMOUT
16      ( component_name <- f.component_name
17      ),
18      t2 : AMM!SimpleAbstractComponent in AMOUT
19      ( component_name <- 'Text',
20        component_parent <- t1
21      )
22 }
```

User Interface Adaptation



Conclusions

- **Adaptive transformation for architectural models at runtime**
- Transformation **pattern/template** for adaptation schema
- Adaptation schema is also **changeable** and **adaptable**
- High degree of **adaptability**
- All adaptation elements are based on **MDE**
  - **Models** (architectures, rule repository, selected rules)
  - **M2M** (RuleSelection, RuleSelectionLog, ModelTransformation)
  - **M2T** (RuleTransformation)
- Case study:

  Adaptation of component-based user interfaces at runtime

Future work

- **Higher degree of adaptability**

- Add to selection logic: **use frequency of rules**, **rule weight management policies**, etc.

- **RuleTransformation** process

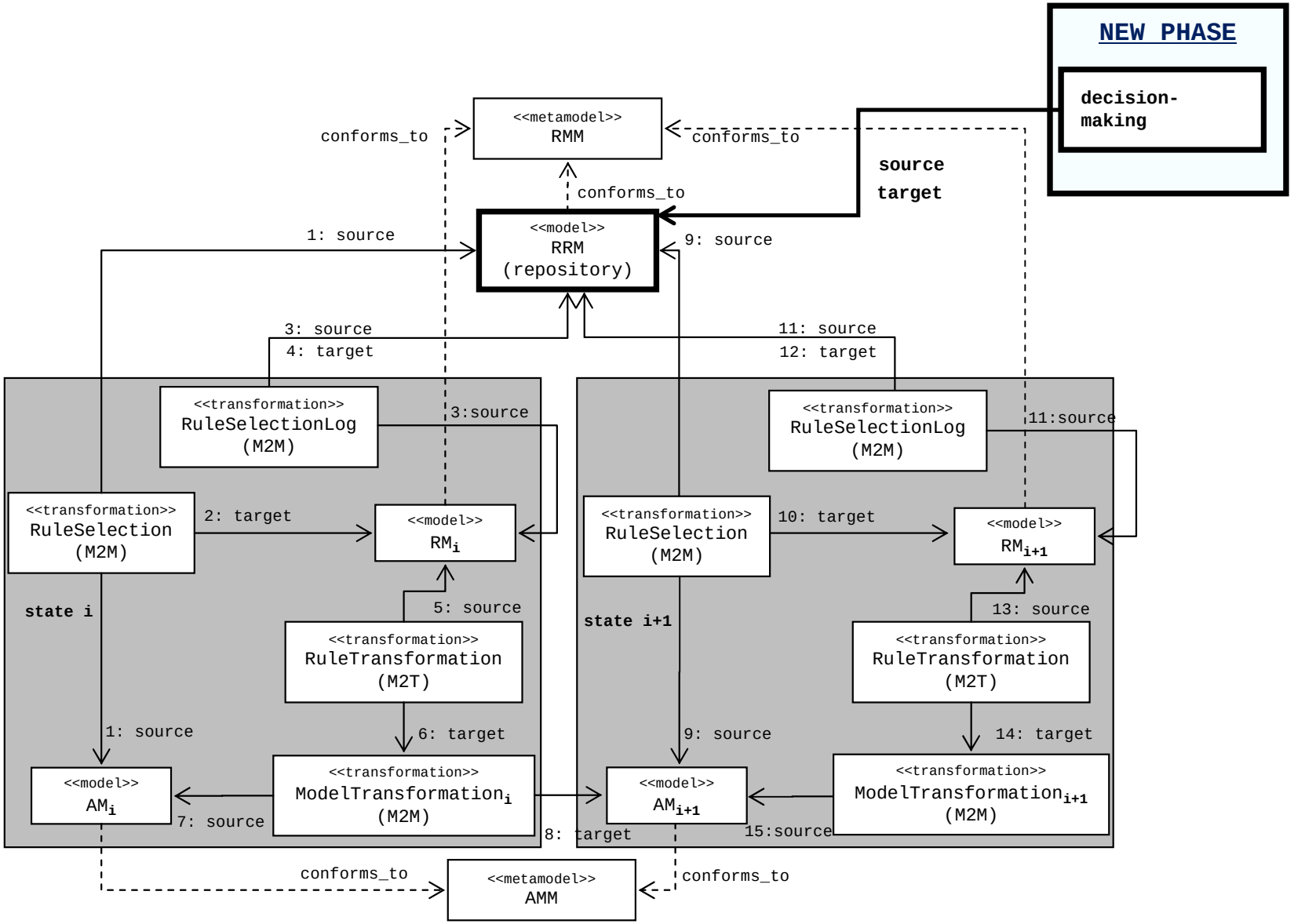
  Replace M2T transformation (JET)
  by **HOT (Higher Order Transformation)**

  - **Editing tool** for building:

  Adaptation rules
  Architectural models

  - Automate the **context processing** by using **M2M** transformations

Future work


NEW PHASE
decision-making
source
target
<<metamodel>>
RMM
conforms_to
conforms_to
conforms_to
<<model>>
RRM
(repository)
1: source
9: source
3: source
4: target
11: source
12: target
<<transformation>>
RuleSelectionLog
(M2M)
3:source
<<transformation>>
RuleSelectionLog
(M2M)
11:source
<<transformation>>
RuleSelection
(M2M)
2: target
<<model>>
RMi
<<transformation>>
RuleSelection
(M2M)
10: target
<<model>>
RMi+1
state i
5: source
state i+1
13: source
<<transformation>>
RuleTransformation
(M2T)
<<transformation>>
RuleTransformation
(M2T)
1: source
6: target
9: source
14: target
<<model>>
AMi
7: source
<<transformation>>
ModelTransformationi
(M2M)
8: target
<<model>>
AMi+1
15:source
<<transformation>>
ModelTransformationi+1
(M2M)
conforms_to
<<metamodel>>
AMM
conforms_to

Thank you very much for your attention!!

Questions??

Runtime Adaptation of Architectural Models: an approach for adapting User Interfaces

Diego Rodríguez-Gracia, Javier Criado,
Luis Iribarne, Nicolás Padilla
Applied Computing Group, University of Almería, Spain

Cristina Vicente-Chicote
Department of Information Communication Technologies
Technical University of Cartagena, Spain

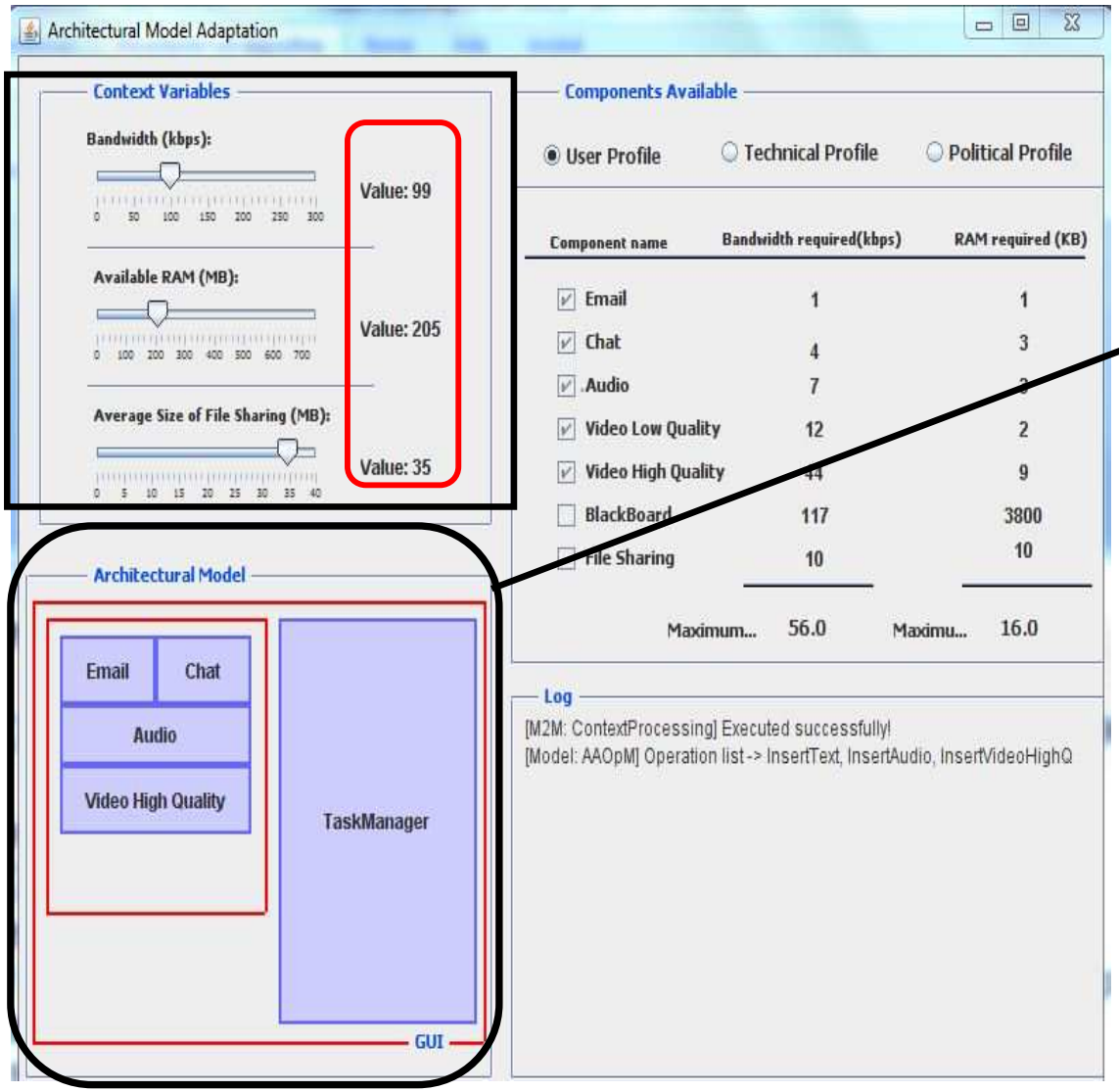
GOBIERNO DE ESPAÑA MINISTERIO DE CIENCIA E INNOVACIÓN

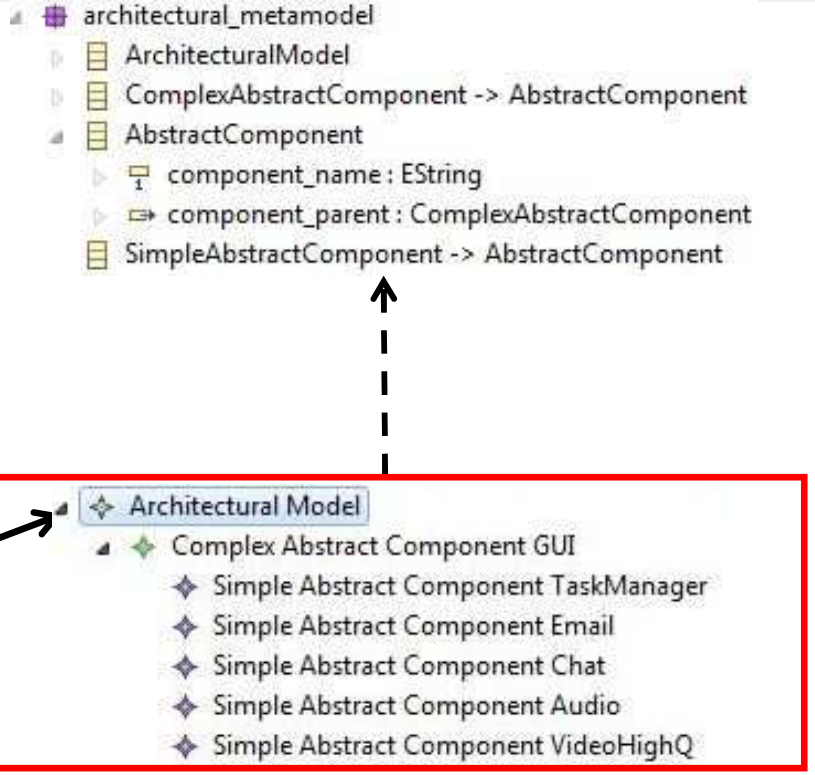
Una Metodología para la Recuperación y Explotación de Información Medioambiental **(TIN2010-15588)**

JUNTA DE ANDALUCIA

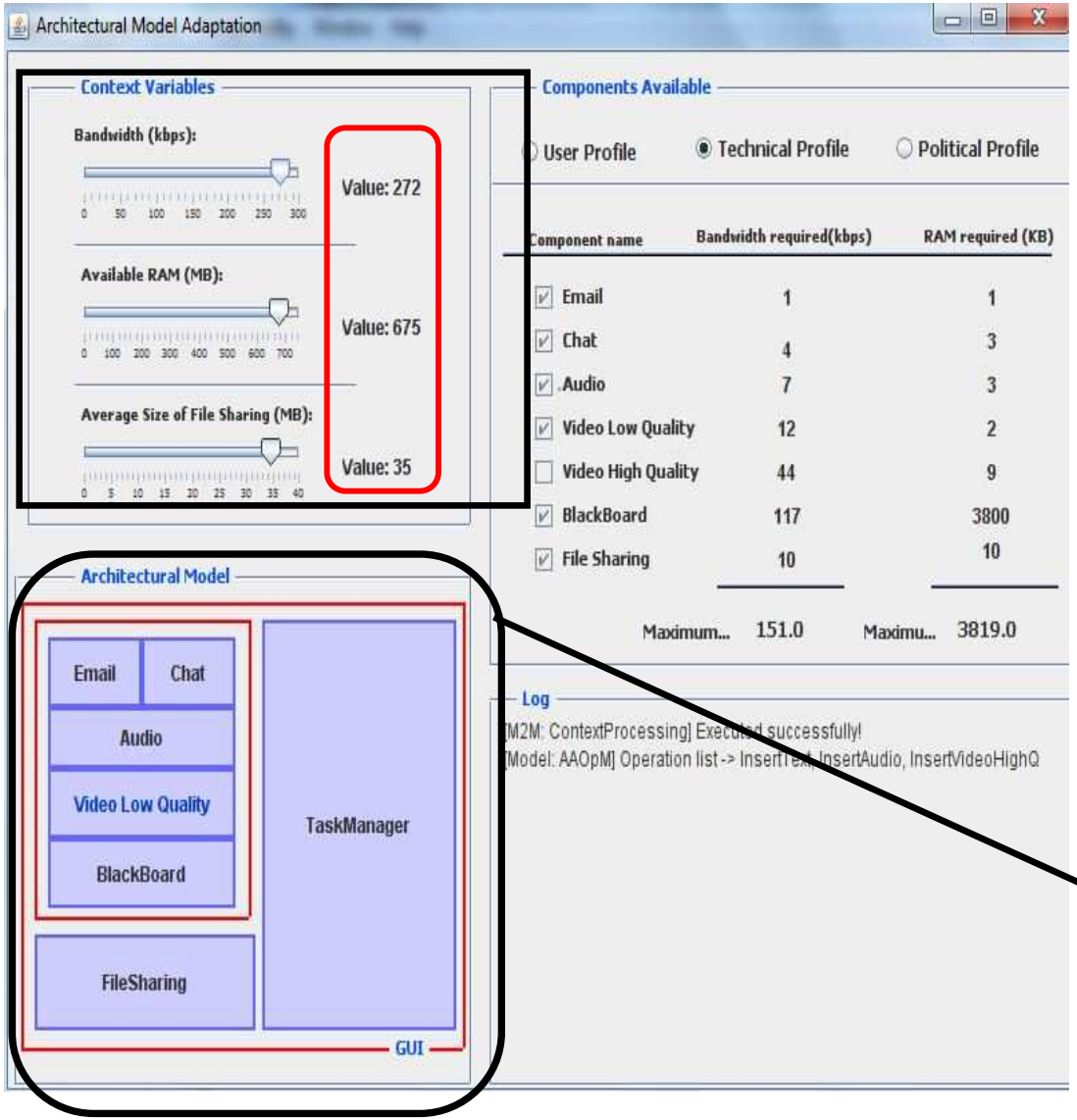
Desarrollo de un Agente Web Inteligente de Información Medioambiental **(TIC-6114)**

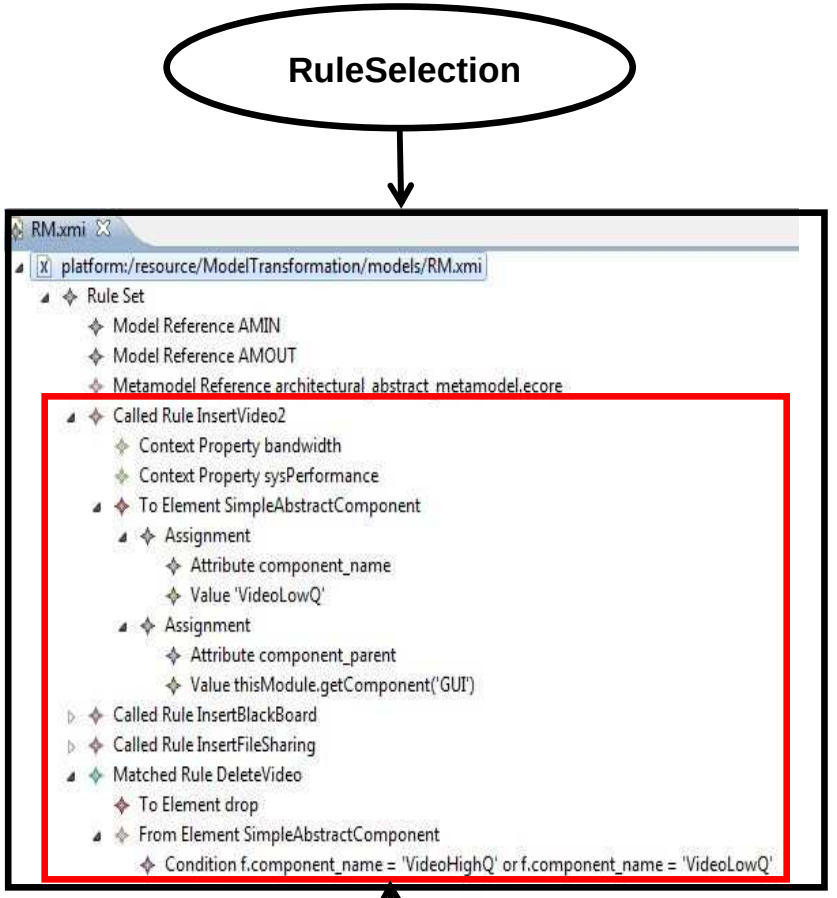
A validation tool of our proposal

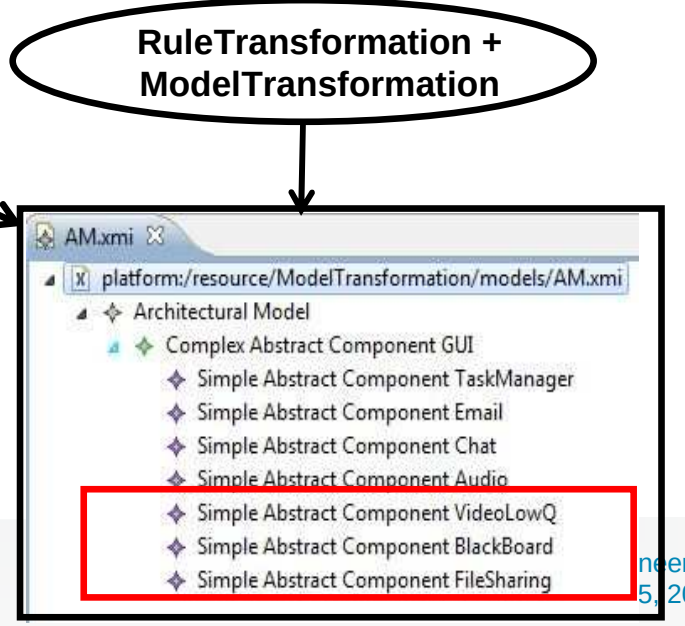




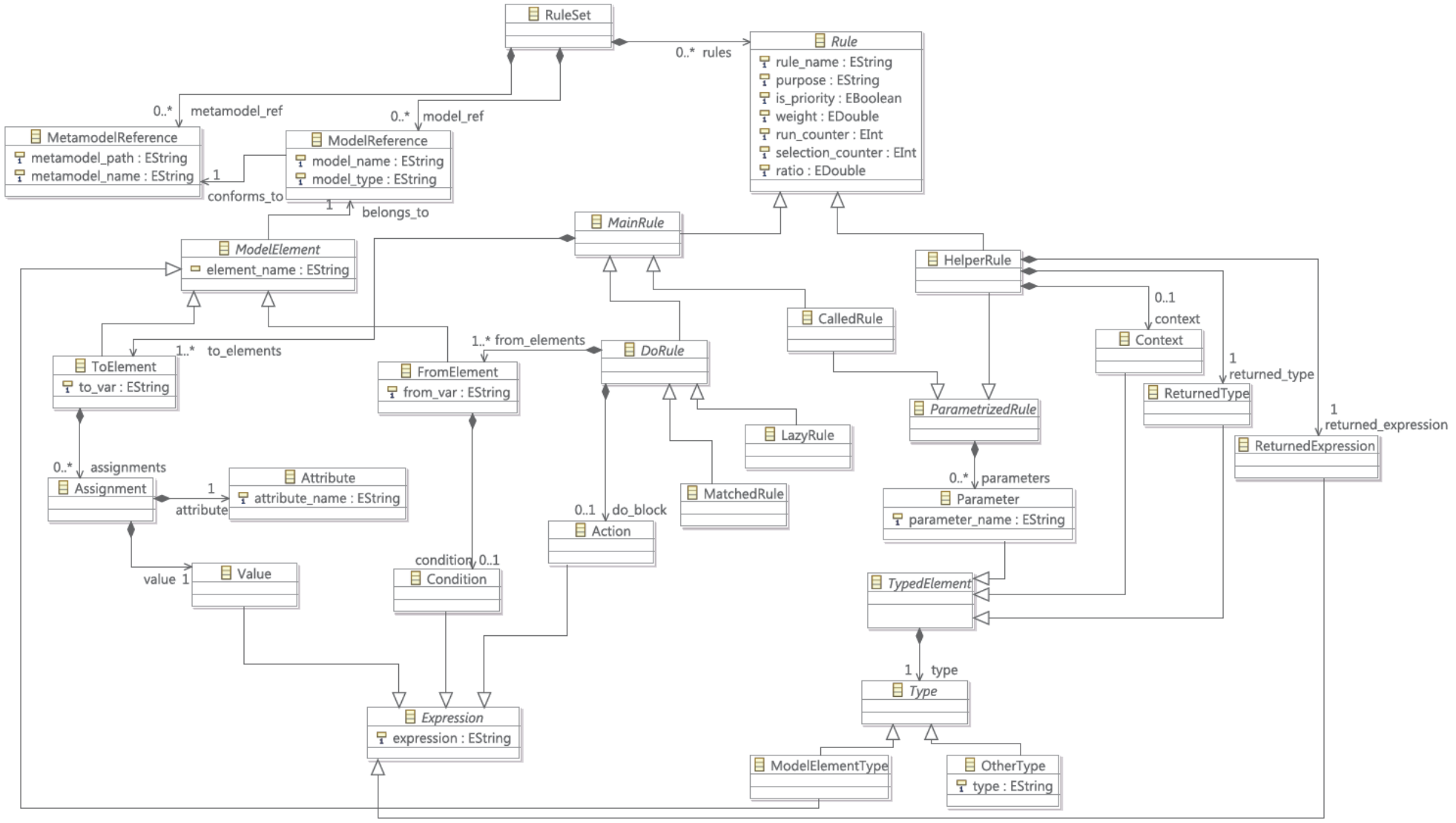
A validation tool of our proposal



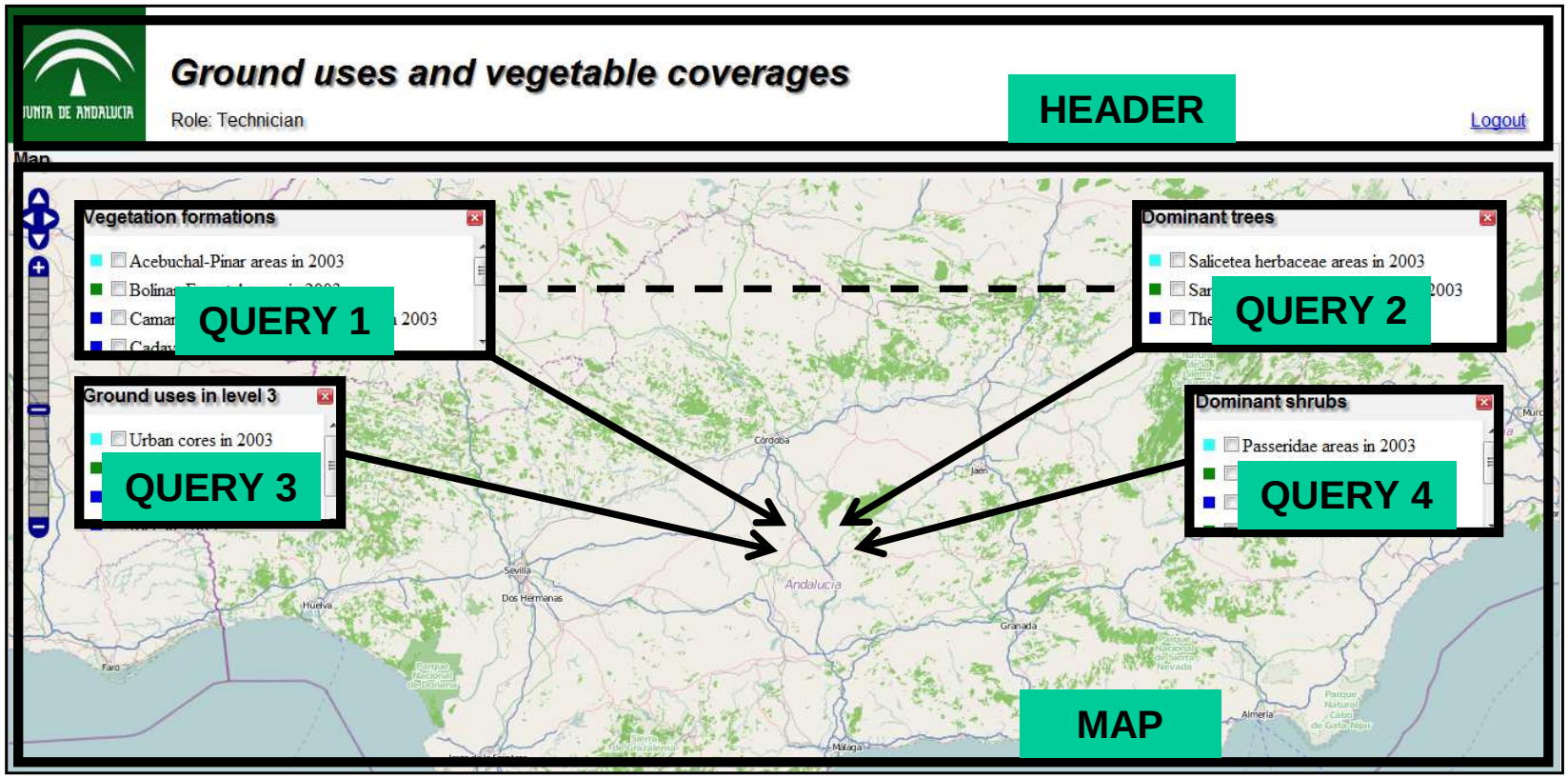




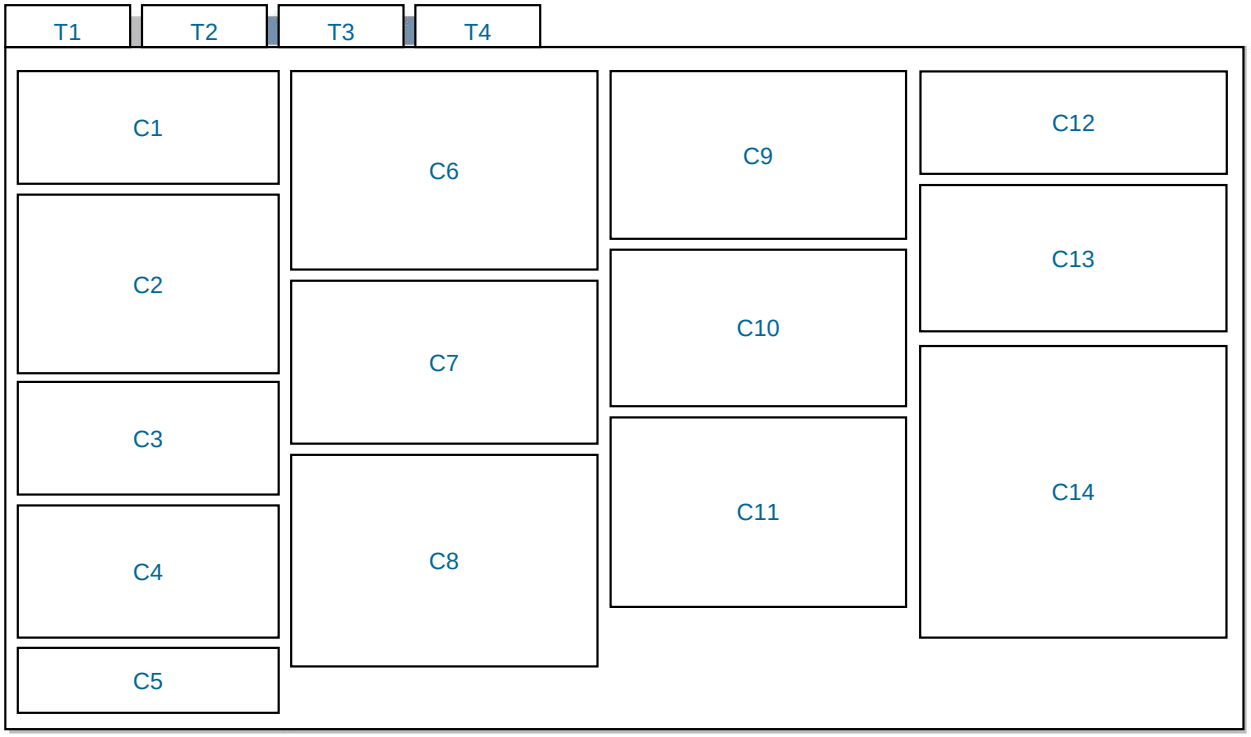
Transformation Rules



Our goal


Ground uses and vegetable coverages
Role: Technician
HEADER
Logout
Map
Vegetation formations
Acebuchal-Pinar areas in 2003
QUERY 1
Dominant trees
Salicetea herbaceae areas in 2003
QUERY 2
Ground uses in level 3
Urban cores in 2003
QUERY 3
Dominant shrubs
Passeridae areas in 2003
QUERY 4
MAP

Our goal

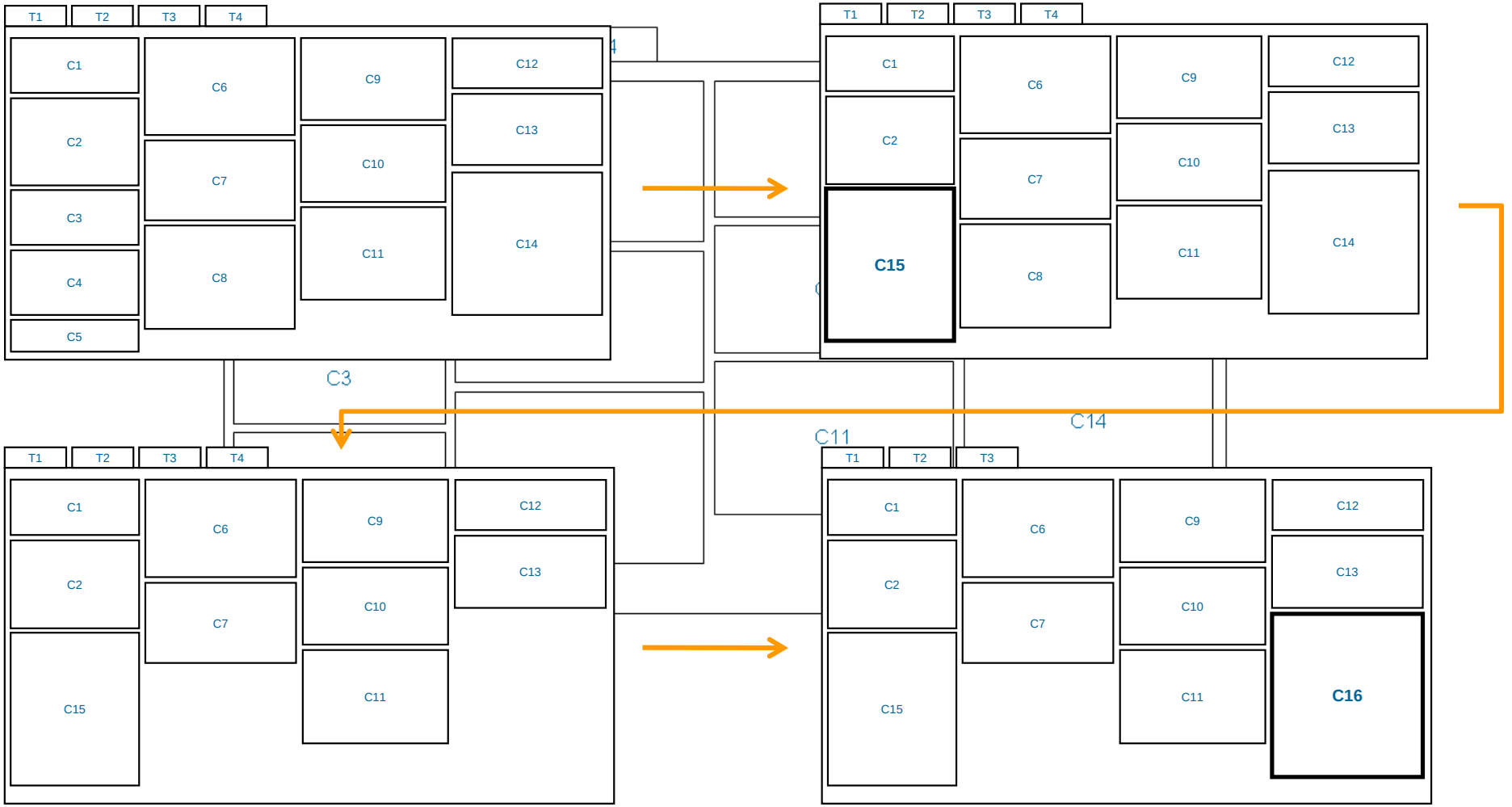


* Extensible to other component-based software models

Our goal

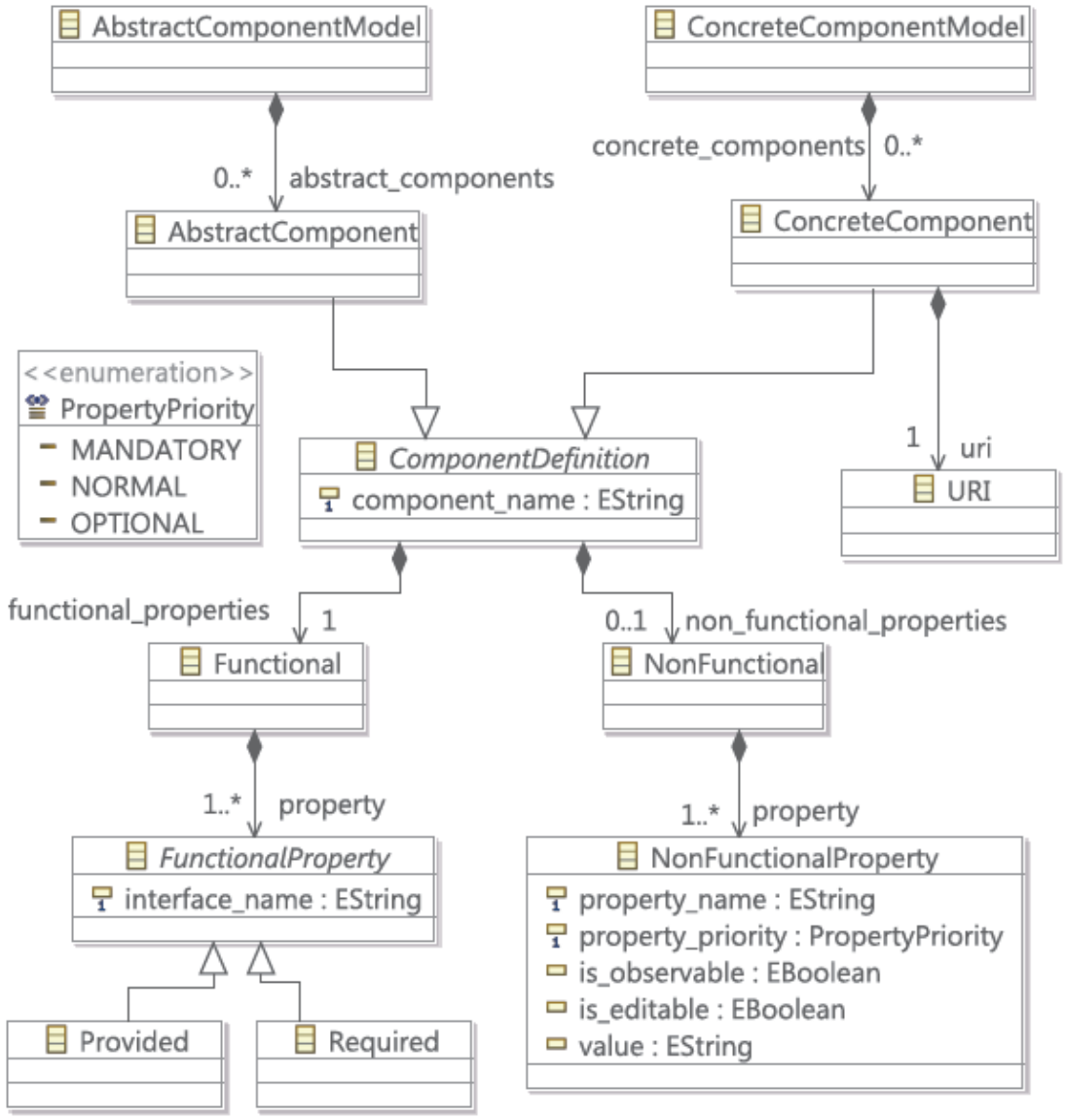
(Starting model)

Replace C3, C4 and C5 by C15

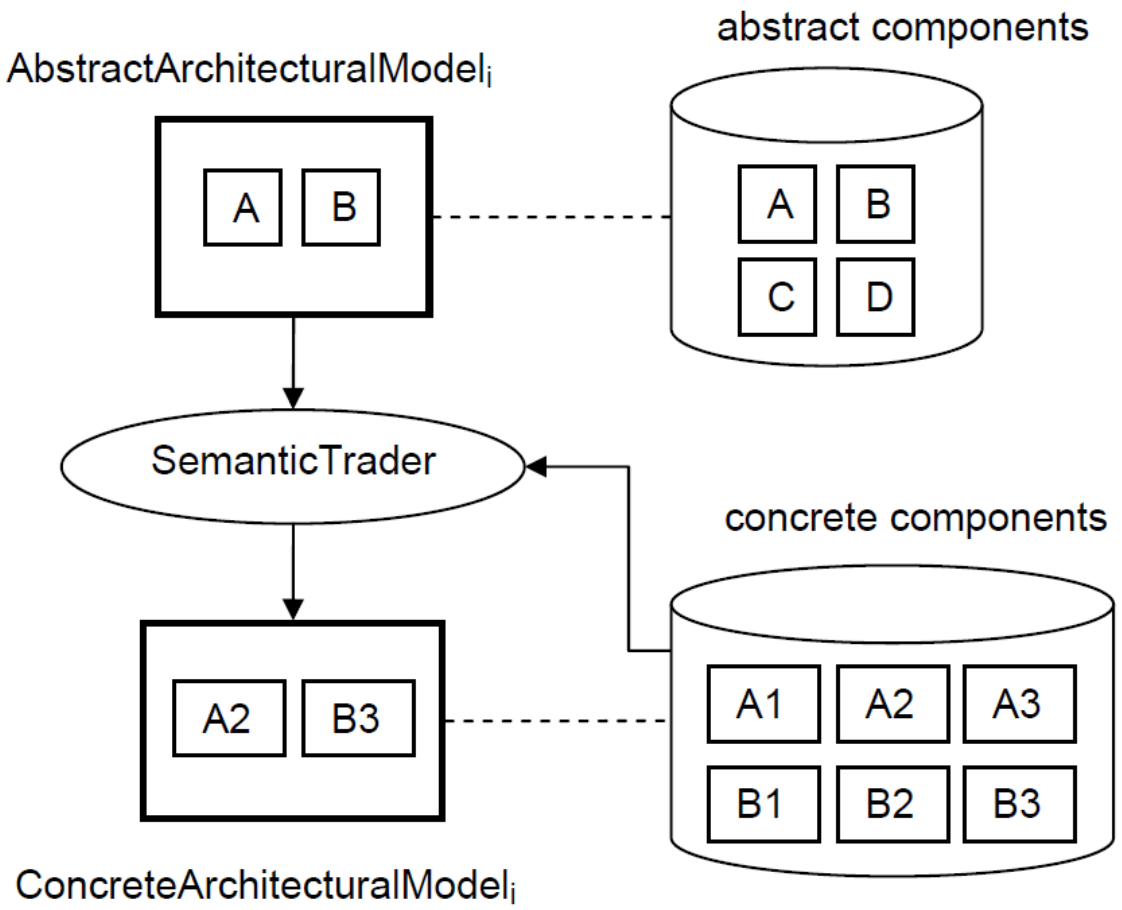


Delete C8 and C14

Insert C16 and Delete T4


AbstractComponentModel
ConcreteComponentModel
0..*
abstract_components
concrete_components
0..*
AbstractComponent
ConcreteComponent
<<enumeration>>
PropertyPriority
MANDATORY
NORMAL
OPTIONAL
ComponentDefinition
component_name : EString
1
uri
URI
functional_properties
1
0..1
non_functional_properties
Functional
NonFunctional
1..*
property
1..*
property
FunctionalProperty
interface_name : EString
NonFunctionalProperty
property_name : EString
property_priority : PropertyPriority
is_observable : EBoolean
is_editable : EBoolean
value : EString
Provided
Required

Regeneration



Performance