UsiXML, a User Interface Model and Language Engineering approach

Jean Vanderdonckt, ***Juan Manuel Gonzalez Calleros***

Université catholique de Louvain (UCL)
Louvain School of Management (LSM)
Information Systems Unit (ISYS)
Belgian Laboratory of Computer-Human Interaction (BCHI)
http://www.isys.ucl.ac.be/bchi

Who are we?

UCL - BCHI

- Belgian Lab of Human-Computer Interaction (BCHI) The BCHI Lab has **20 years of experience** in the domain of user interface engineering, which combines techniques from Human-Computer Interaction, Software Engineering, and Usability Engineering.
- Model based User Interface Development
  - Multi Modal
  - 3D UIS
  - 2D UIS (Web, desktop, …)
  - Migration
  - Context adaptation
  - ….

BCHI-Past Projects

- Cameleon (Context Aware Modelling for Enabling and Leveraging Effective interactiON)
- Envir3D (Automatic Generation of Virtual Reality Scenes)
- Kwaresmi (Knowledge-based Web Automatic REconfigurable evaluation with guidelineS optiMIzation)
- MetroWeb (METROlogy of WEB sites)
- Salamandre (User Interfaces for Mobile and Multi-platform Interactive Systems)
- Visme (VIsual Scene composition with multi-resolution and modulation for a Multi-sources Environment dedicated to neuro-navigation)
- Similar (The European taskforce creating human-computer interfaces SIMILAR to human-human communication)
- Destine (Design and Evaluation STudio for INtent-based Ergonomic web sites)

BCHI-Current Projects

- UsiXML (USer Interface eXtensible Markup Language)
- Vitality User Interface for Medical data visualization
- HUMAN Model-Based Analysis of Human Error During Aircraft Cockpit System Design

What are we doing?

UsiXML-the Problem

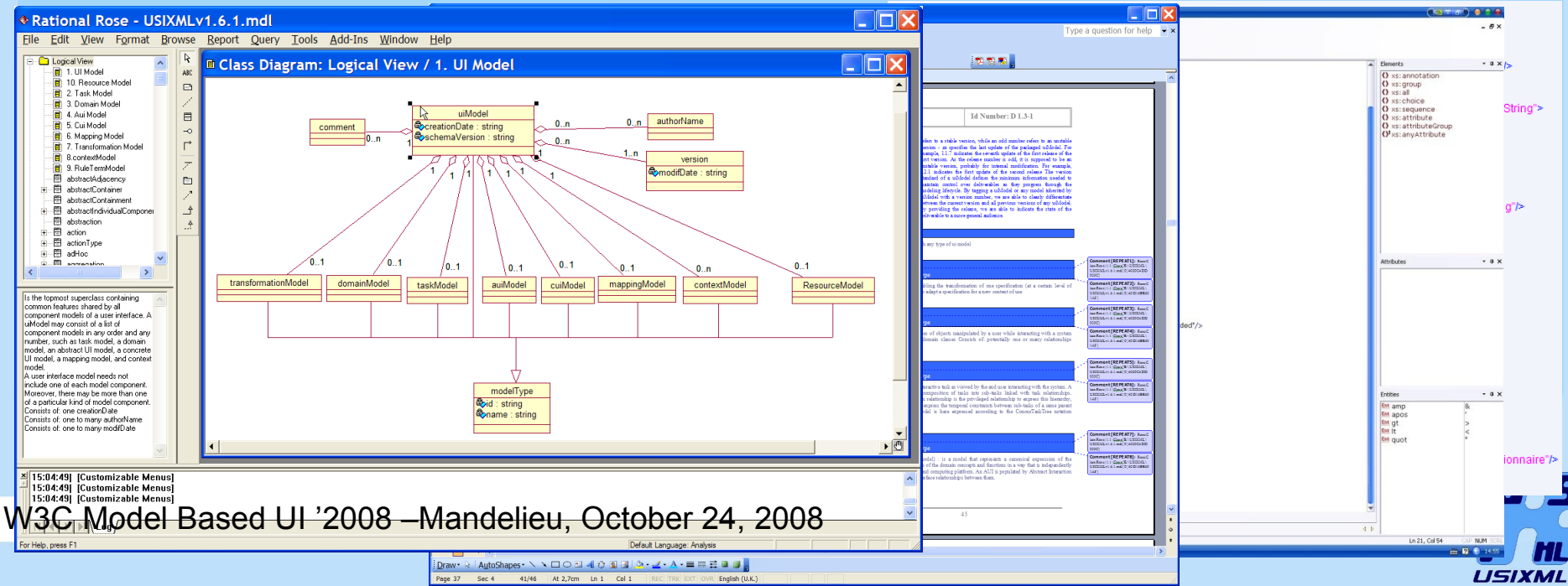
- To develop user interfaces (UIs) simultaneously for multiple contexts of use
- A context of use = triple
  - User
  - Computing platform
  - Surrounding environment
    - Organisation
    - Socio-psychological factors

UsiXML

- What is UsiXML?
  - It is a XML-compliant User Interface Description Language
  - Publicly available from http://www.usixml.org
  - Free to use, open for access, easy to expand
  - Definition of the language

UML Class Diagrams → **UsiXML Reference manual**

UML Class Diagrams → **XSD XML Schema Descriptions** → **UsiXML Models**



BCHI-UsiXML

- UsiXML = USer Interface exTensible Markup Language
  - http://www.usixml.org
  - Join the UsiXML Consortium by registering on line
  - Download the CD image from http://www.usixml.org/index.php?download=UsiXML RelOne.iso

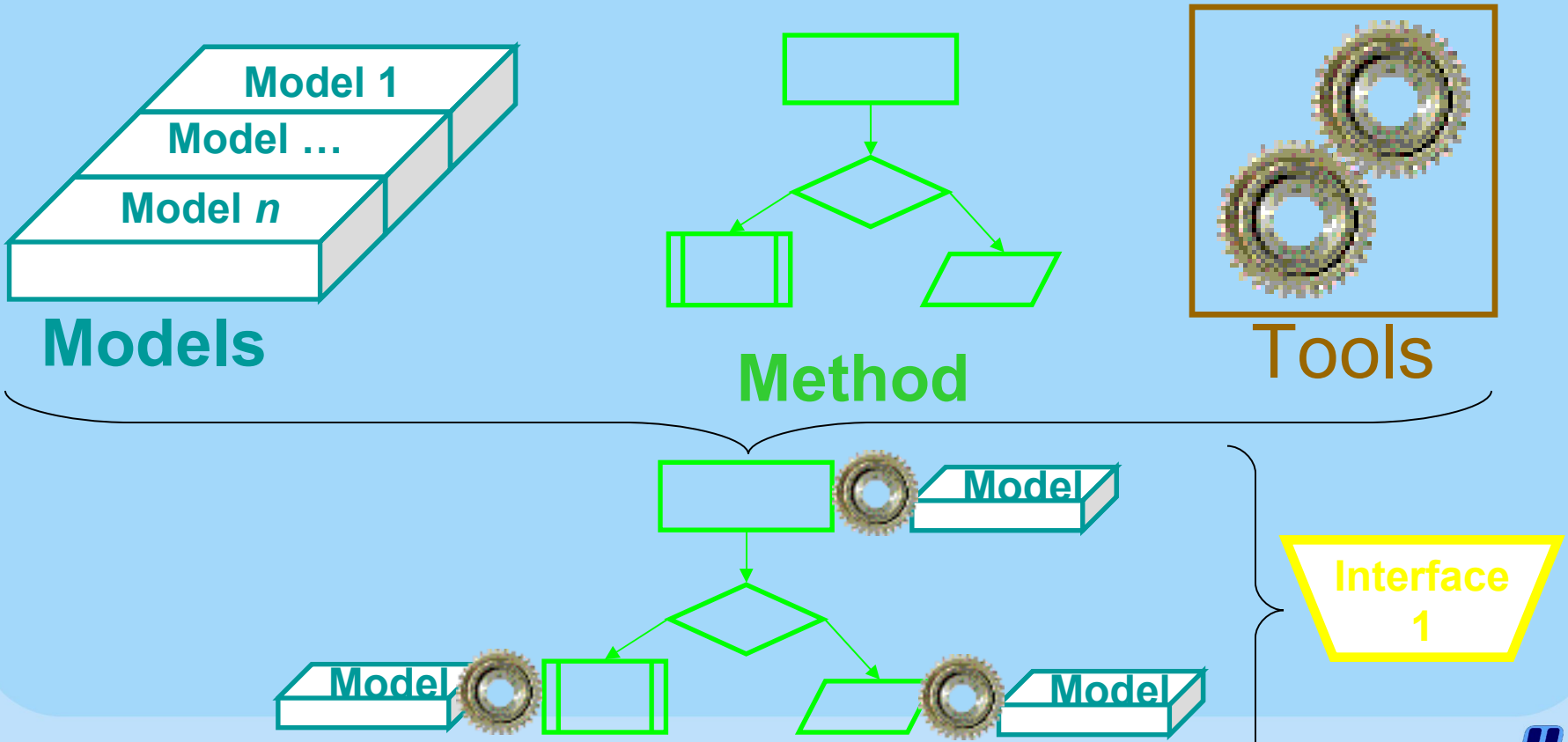
What do we have so far?

Model Based User Interface development method

- Any development method (or methodology) is decomposed into 4 axes:
  - Models: explicitly capture knowledge about UI and Interactive Applications with appropriate abstractions
  - Language: In order to specify different aspects and related models, a specification language is needed that allows designers and developers to exchange, communicate, and share fragments of specifications and that enables tools to operate on these specifications.
  - Method: structures the definition and use of underlying models in a stage-wise approach
  - Supporting tools: support the use of the method by providing tools for models and their related operations. Ideally, one model should be supported by at least one tool

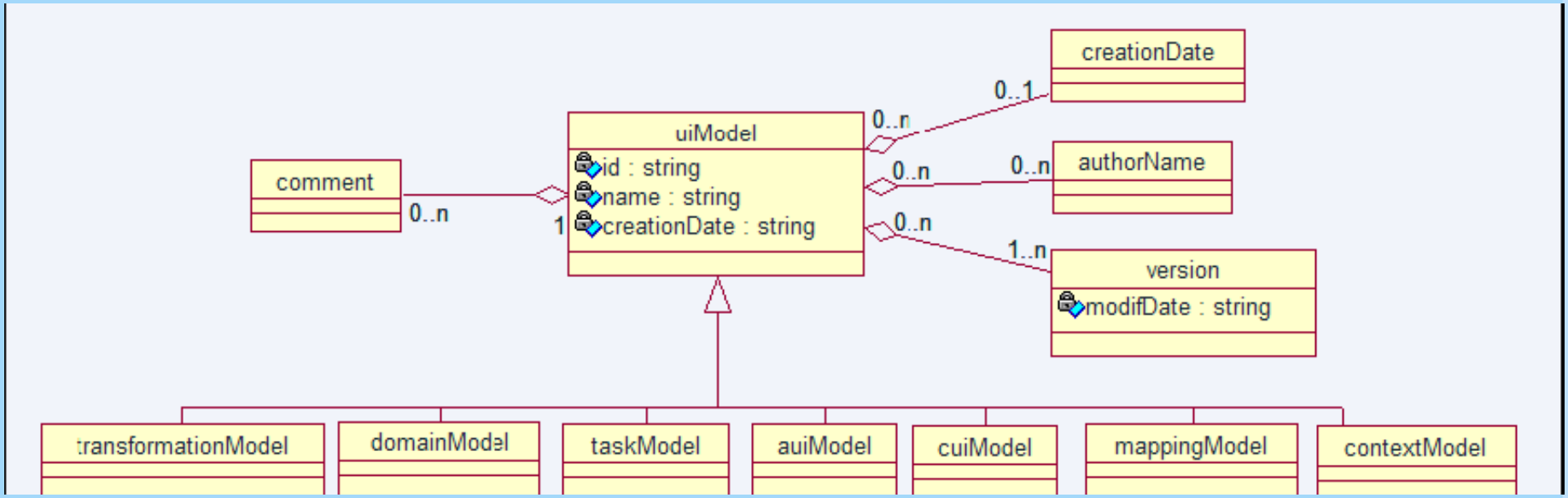
Mono-platform UI

- Goal: to integrate all three facets



Models

The collection of models for specifying a user interface

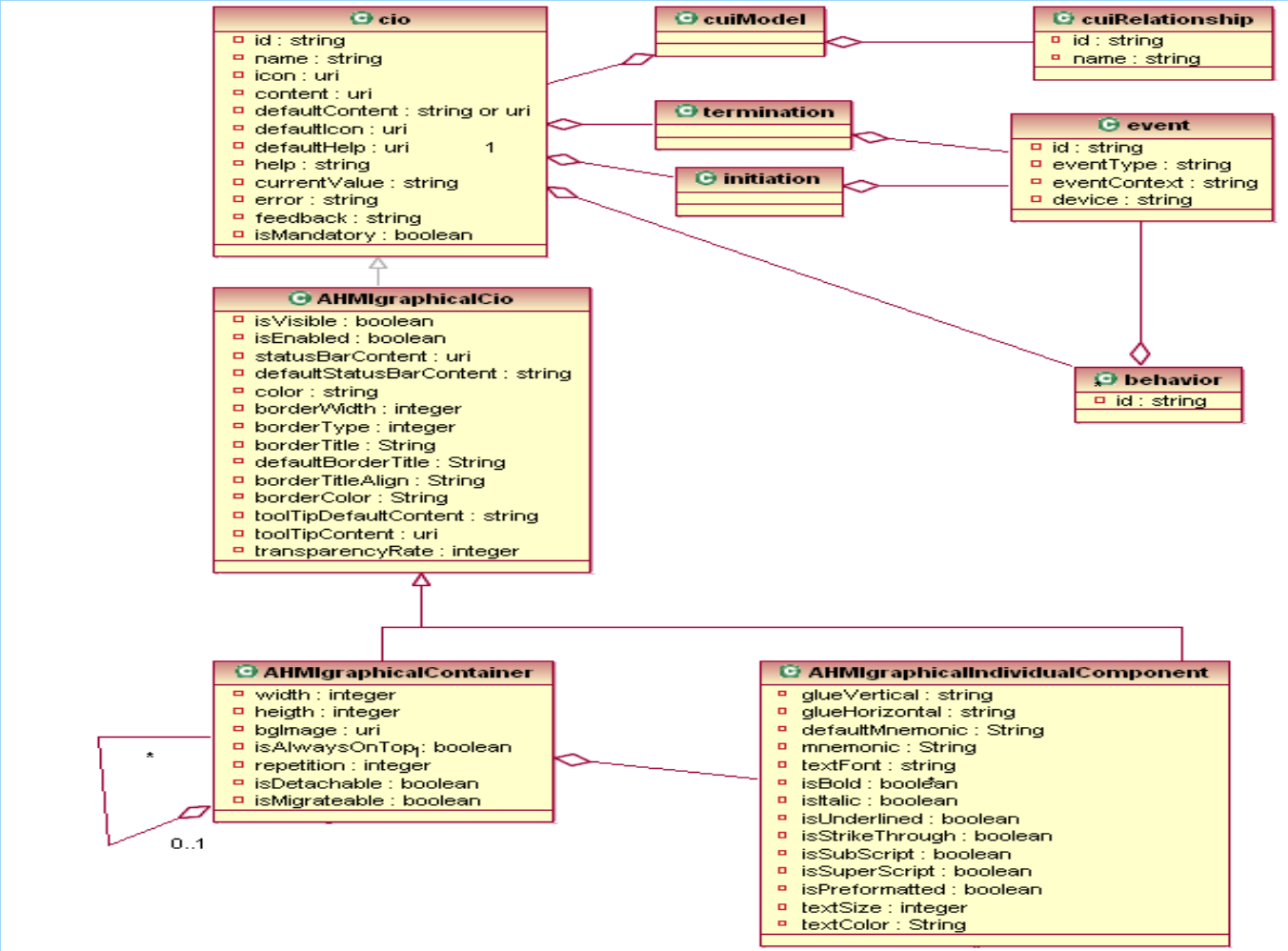


The language

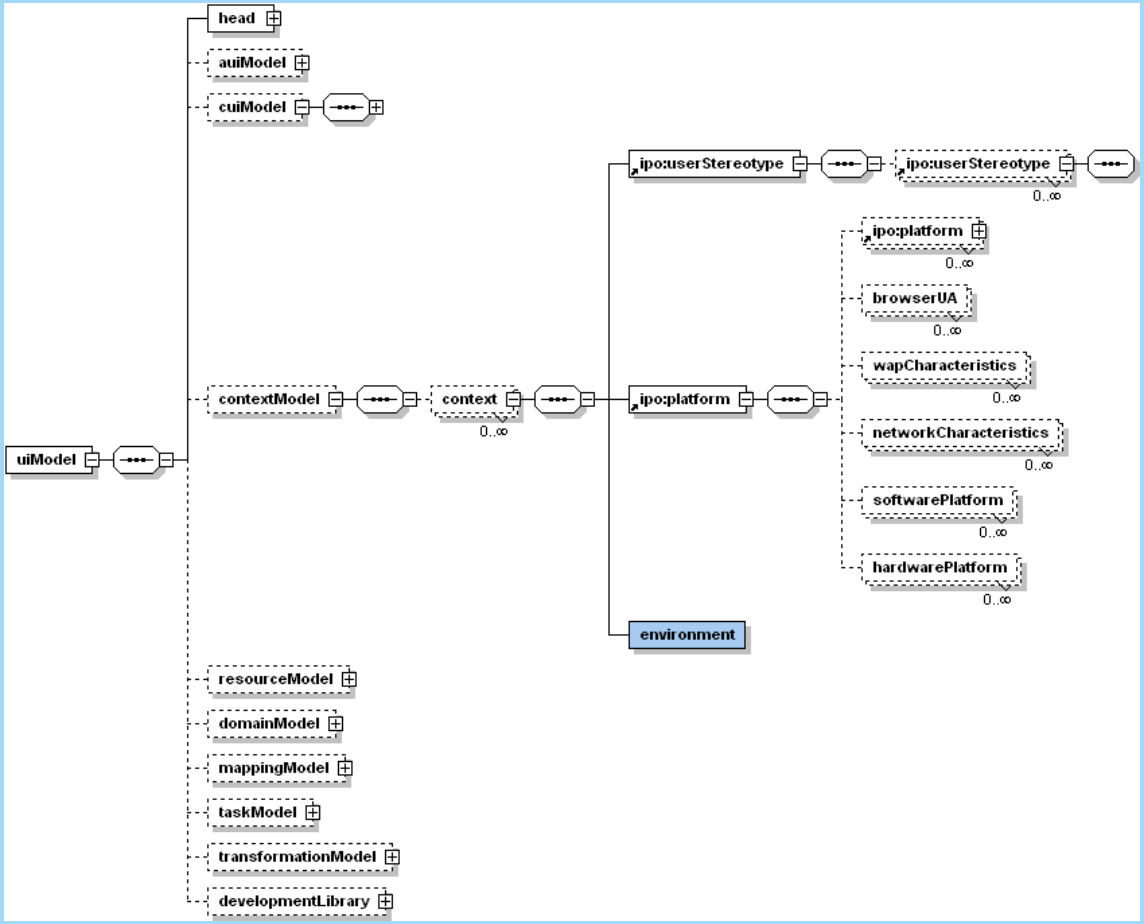
Language Engineering approach

- UsiXML is different from a pure UI authoring language in that it could also be used as a specification language.
- The ultimate goal is not only to generate code, but also to have the capability to reason about the UI specifications:
  - model checking
  - UI evaluation
  - model-driven engineering
  - maintenance of repository of UI cases or patterns
  - static and dynamic analysis
  - model testing

Semantics


cio
id : string
name : string
icon : uri
content : uri
defaultContent : string or uri
defaultIcon : uri
defaultHelp : uri
help : string
currentValue : string
error : string
feedback : string
isMandatory : boolean
1
cuiModel
cuiRelationship
id : string
name : string
termination
event
id : string
eventType : string
eventContext : string
device : string
initiation
behavior
id : string
AHMIgraphicalCio
isVisible : boolean
isEnabled : boolean
statusBarContent : uri
defaultStatusBarContent : string
color : string
borderWidth : integer
borderType : integer
borderTitle : String
defaultBorderTitle : String
borderTitleAlign : String
borderColor : String
toolTipDefaultContent : string
toolTipContent : uri
transparencyRate : integer
AHMIgraphicalContainer
width : integer
heigth : integer
bgImage : uri
isAlwaysOnTop : boolean
repetition : integer
isDetachable : boolean
isMigrateable : boolean
*
0..1
AHMIgraphicalIndividualComponent
glueVertical : string
glueHorizontal : string
defaultMnemonic : String
mnemonic : String
textFont : string
isBold : boolean
isItalic : boolean
isUnderlined : boolean
isStrikeThrough : boolean
isSubScript : boolean
isSuperScript : boolean
isPreformatted : boolean
textSize : integer
textColor : String

Abstract Syntax

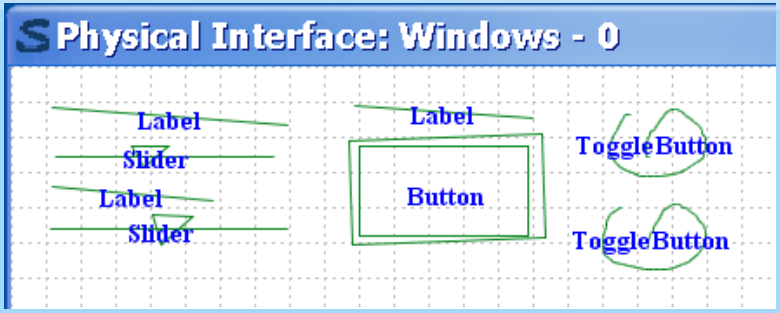


Excerpt for a UsiXML CUI specification

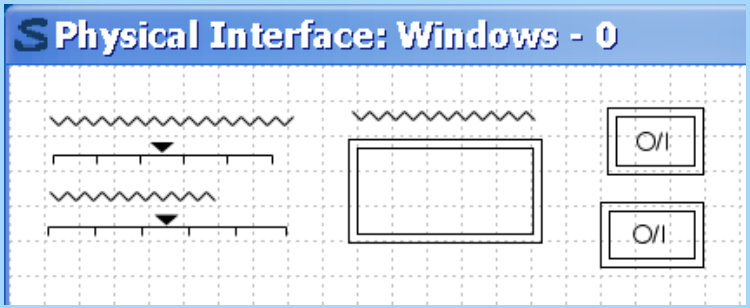
```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<cuiModel  name="MyModel">
   <version modifDate="2004-03-24T17:09:17.402+01:00" xmlns="">7</version>
   <authorName xmlns="">Youri</authorName>
   <window height="500" width="600" name="Formulaire (2/5)" id="window_1">
      <box relativeHeight="100" name="box1_0" id="box1_0">
         <box type="vert" name="boxTodo" id="boxTodo">
            ...
         <box type="horiz" name="box_2_2_2_1" id="box_2_2_2_1">
            <textComponent defaultContent="Sexe" isBold="true" id="label_2"/>
            <radioButton groupName="grupo01" defaultContent="Femme"
                         defaultState="false" id="radiobutton_0"/>
            <radioButton groupName="grupo01" defaultContent="Homme"
                         defaultState="true" id="radiobutton_1"/>
         </box>
            ...
         </box>
      </box>
   </window>
</cuiModel>
```

Stylistics

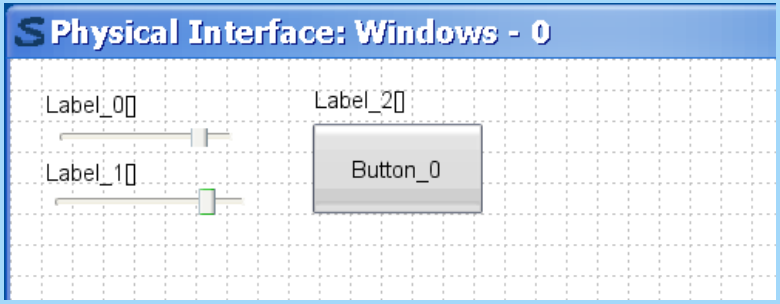
- Low



- Medium

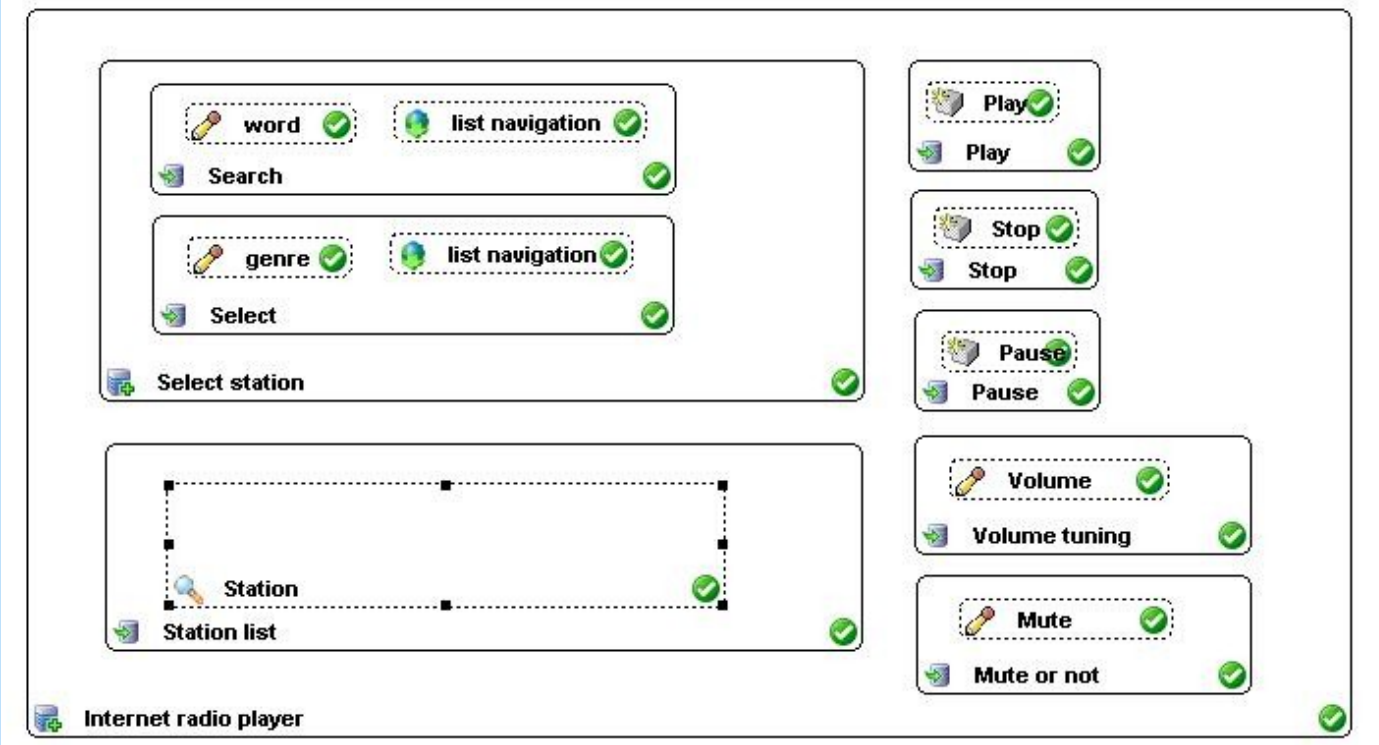


- High



Abstraction: the abstract UI

- Notation: based on L. Constantine's notation for canonical abstract prototypes [Constantine,2003]



- Abs.container
- Abs. component
- Input
- Output
- Navigation
- Control
- Select

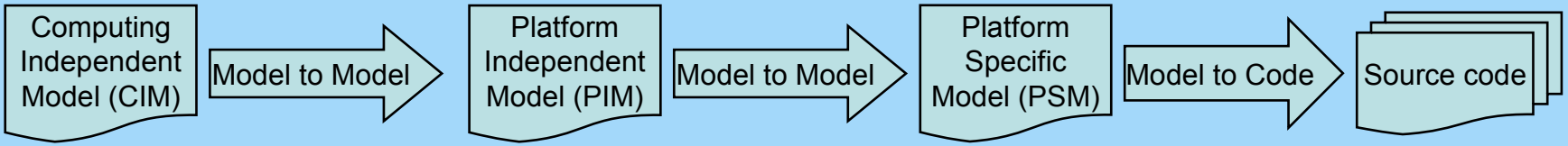
IdealXML

[Montero et al.,2005]

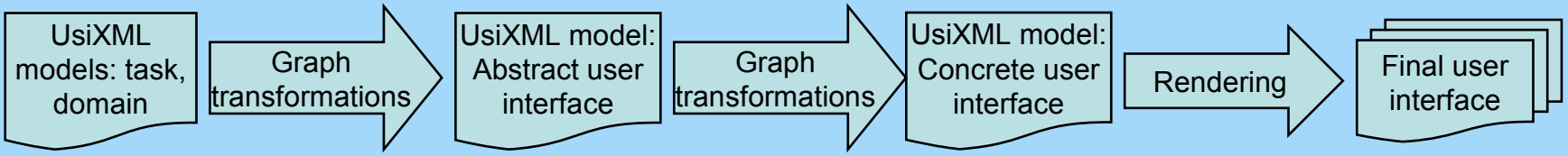
The method

MDE based on UsiXML

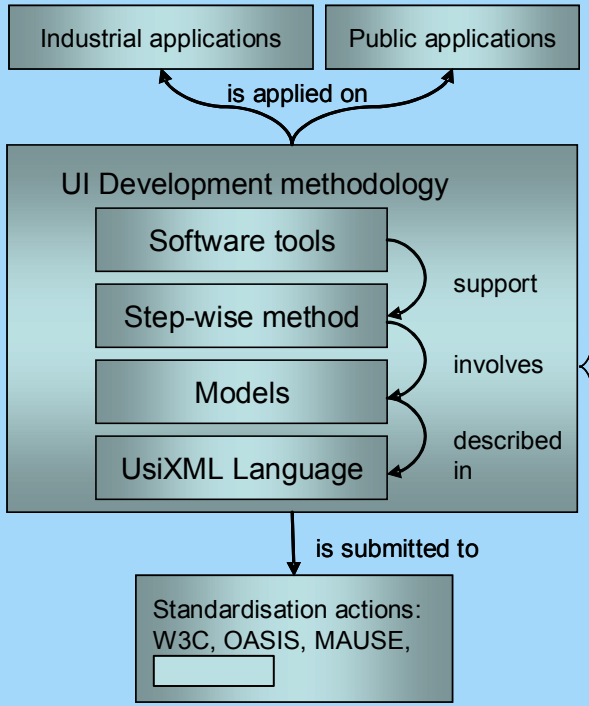
MDA Components

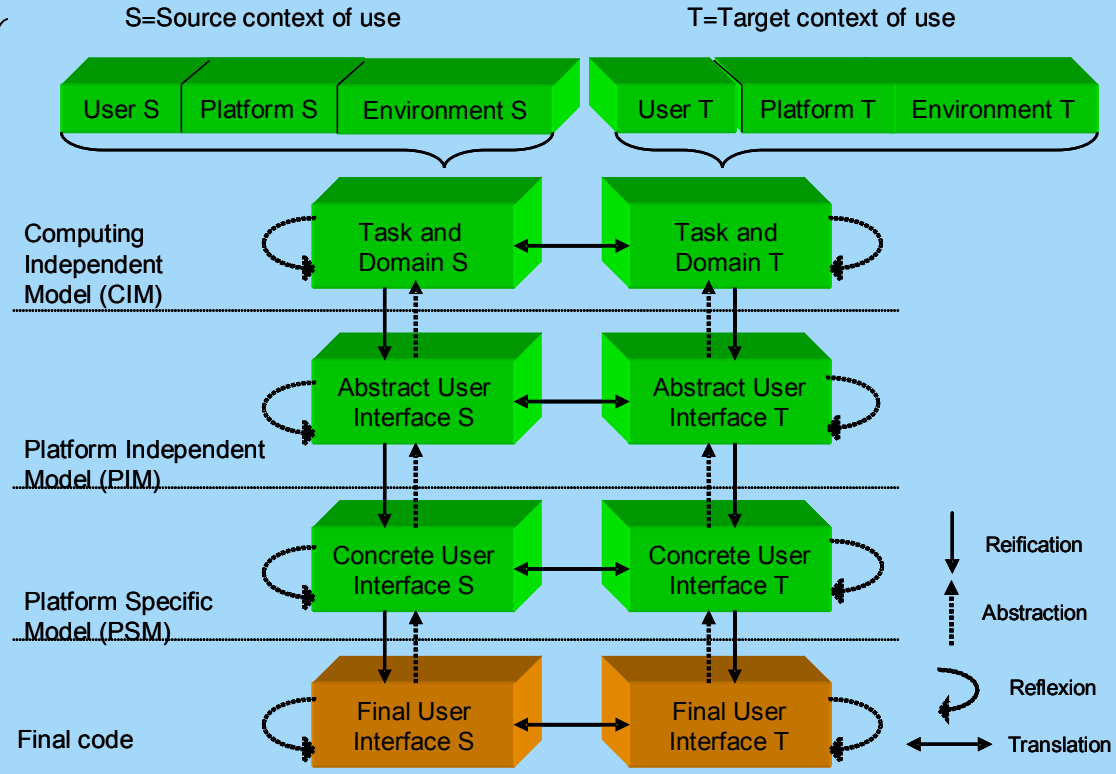


Techniques proposed based on UsiXML



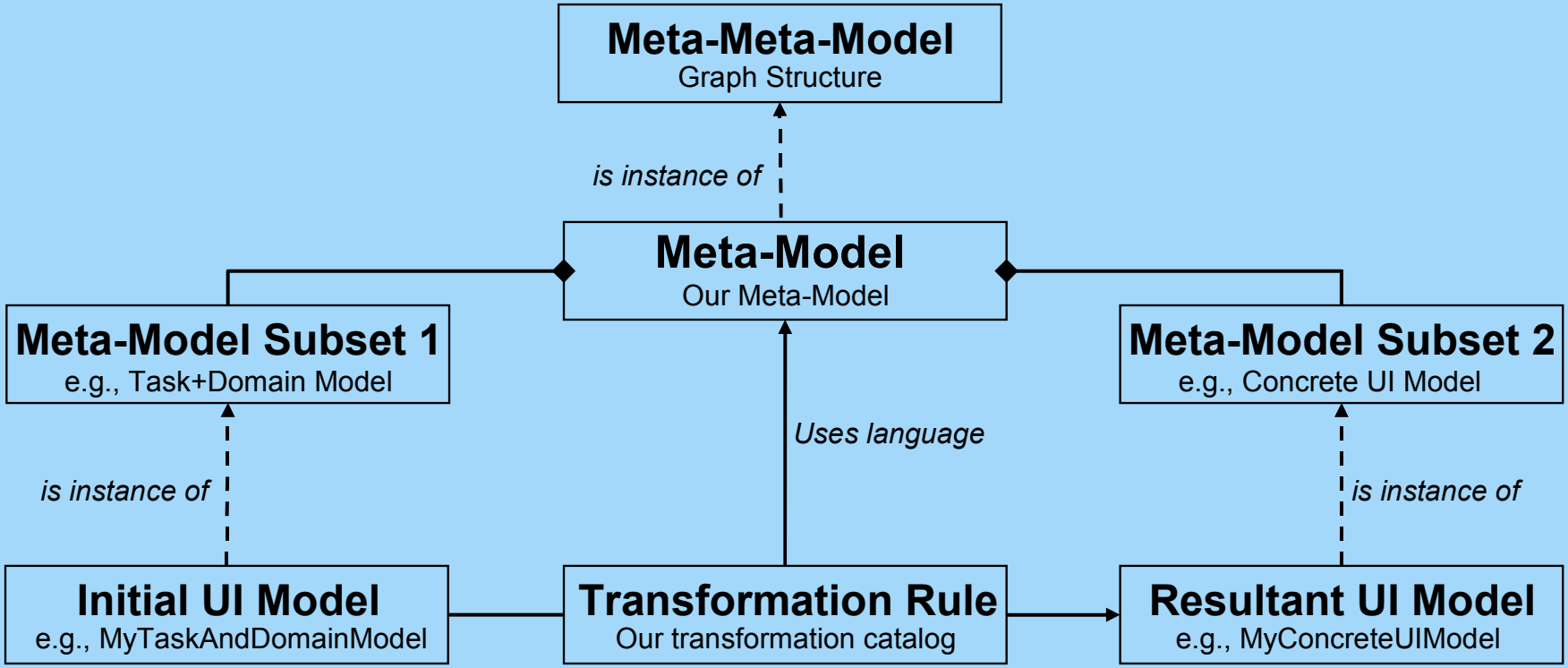
MDE based on UsiXML





Cameleon Reference Framework [Calv03]

Typed Model-to-Model Transformation



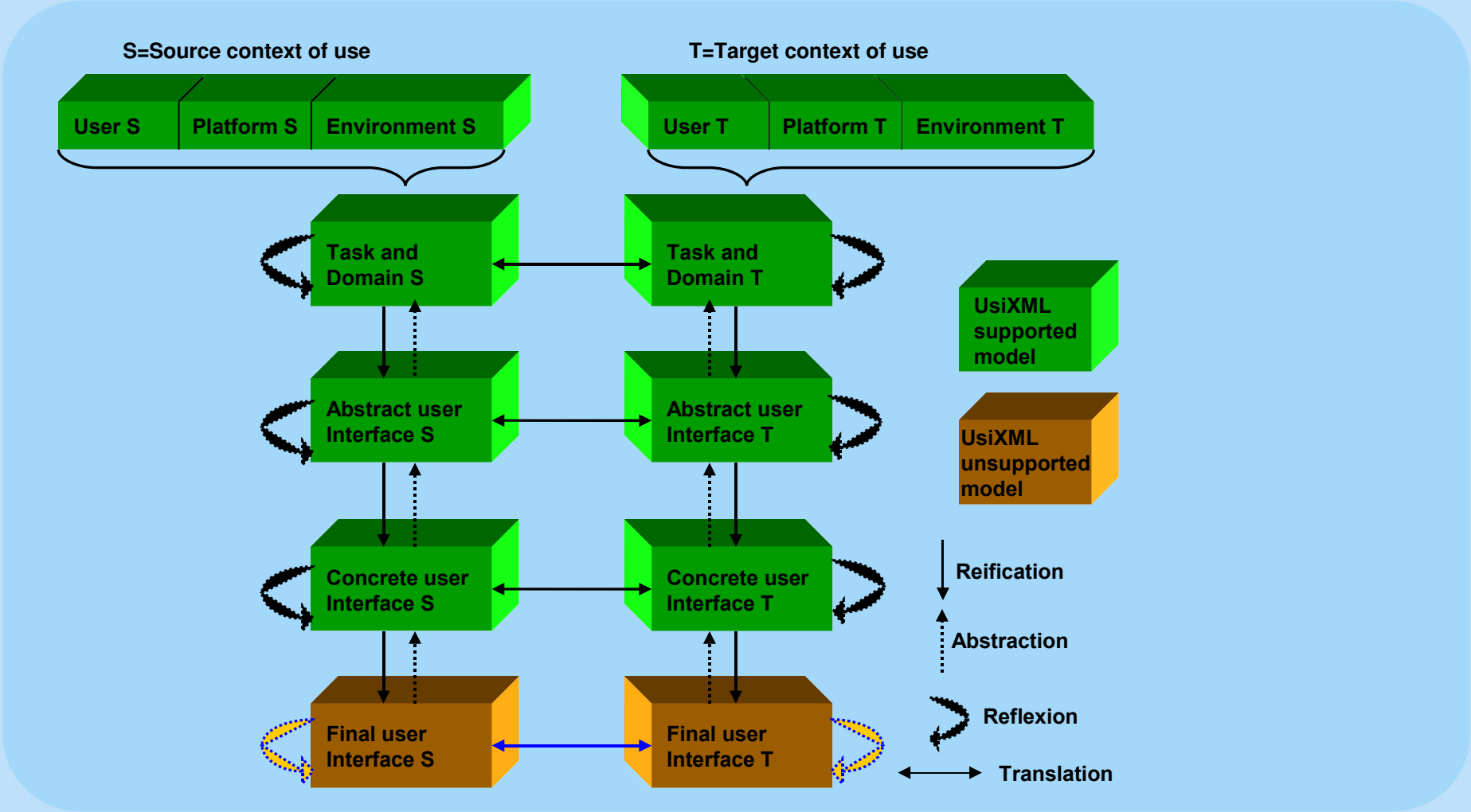
[Limbourg,2004]

Expression of models as graphs

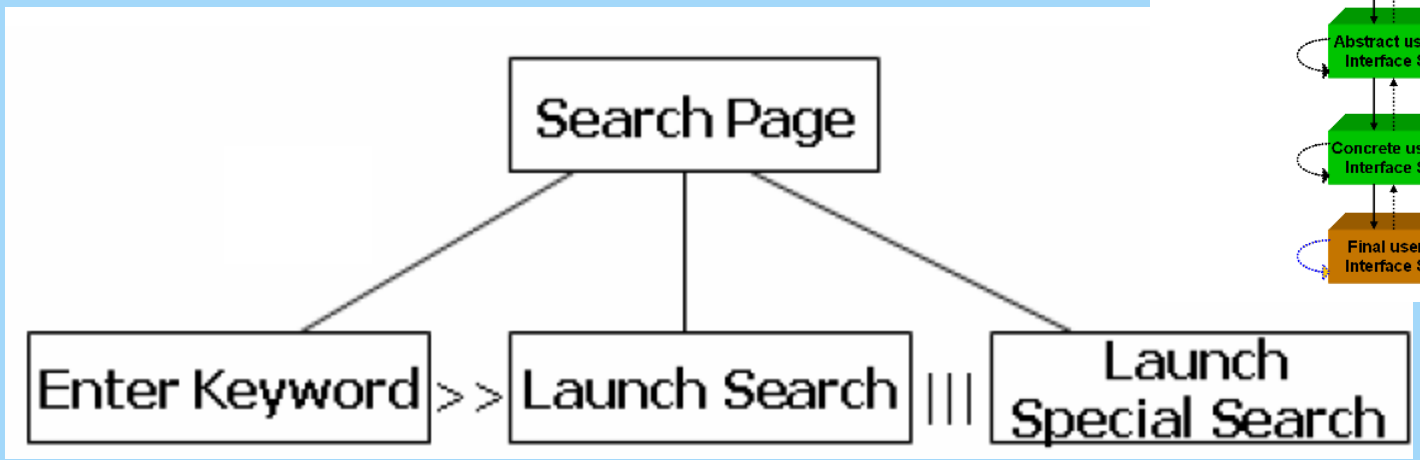
- All transformations are in UsiXML
  - Each model = instance of meta-model
  - Each model = graph as instance of graph type
  - Each model transformation =
    - graph transformation
    - Set of productions

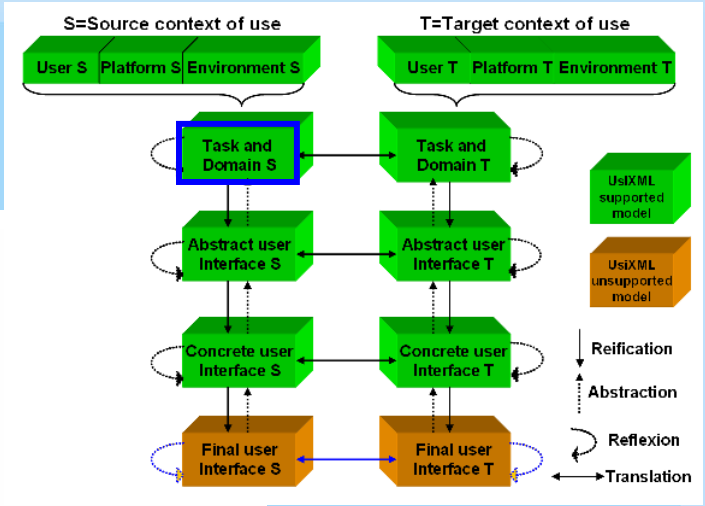
Example of the method

In practice

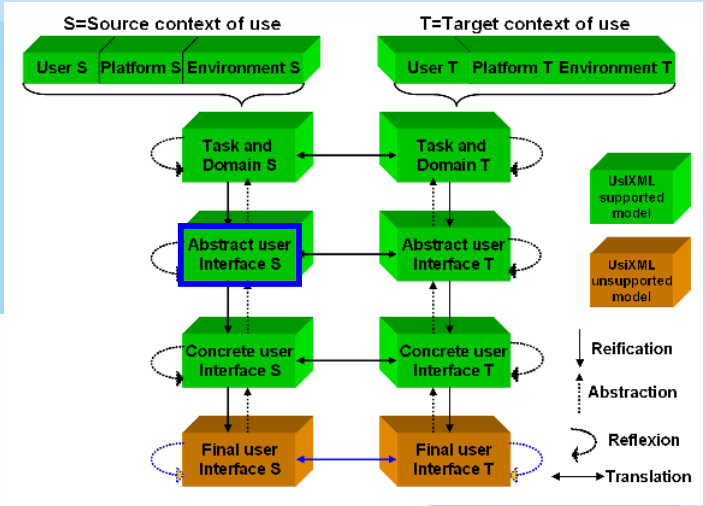

S=Source context of use
T=Target context of use
User S
Platform S
Environment S
User T
Platform T
Environment T
Task and Domain S
Task and Domain T
Abstract user Interface S
Abstract user Interface T
Concrete user Interface S
Concrete user Interface T
Final user Interface S
Final user Interface T
UsiXML supported model
UsiXML unsupported model
Reification
Abstraction
Reflexion
Translation

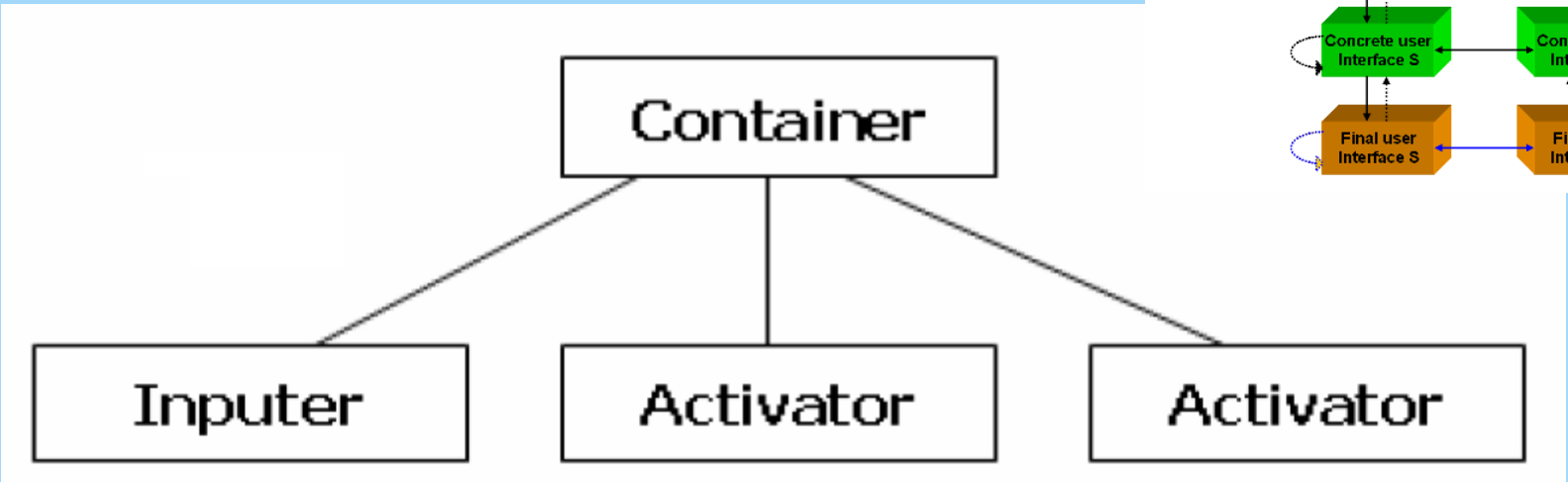
In practice


Search Page
Enter Keyword
>>
Launch Search
|||
Launch Special Search

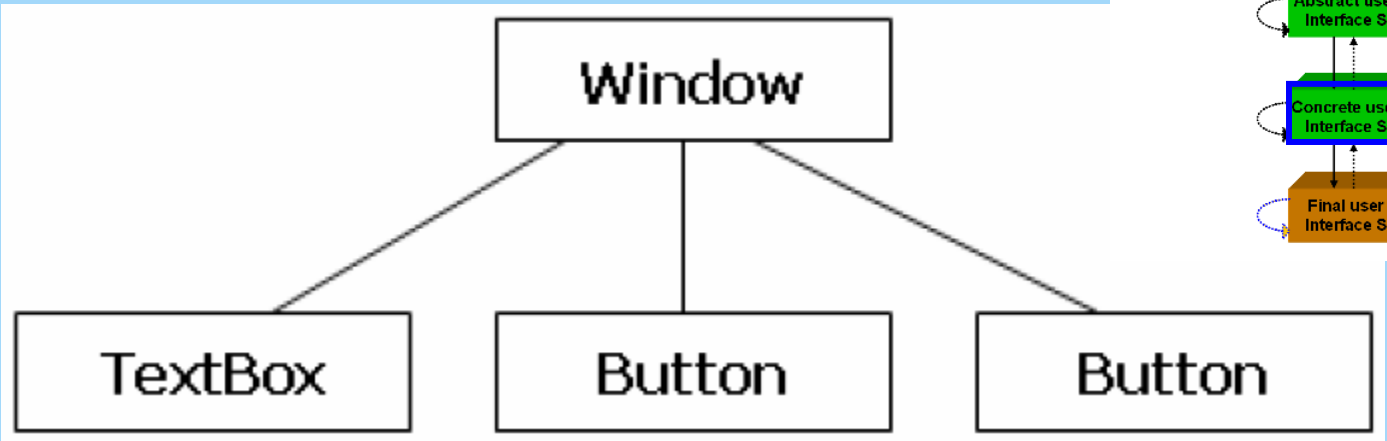

S=Source context of use
T=Target context of use
User S
Platform S
Environment S
User T
Platform T
Environment T
Task and Domain S
Task and Domain T
Abstract user Interface S
Abstract user Interface T
Concrete user Interface S
Concrete user Interface T
Final user Interface S
Final user Interface T
UsiXML supported model
UsiXML unsupported model
Reification
Abstraction
Reflexion
Translation

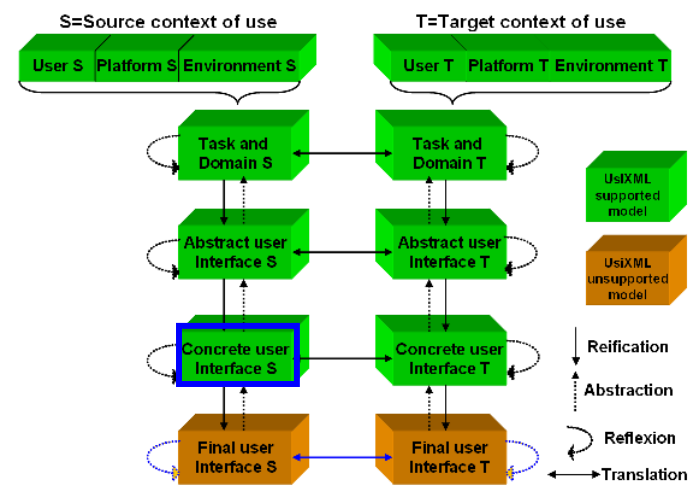
In practice


S=Source context of use
T=Target context of use
User S
Platform S
Environment S
User T
Platform T
Environment T
Task and Domain S
Task and Domain T
Abstract user Interface S
Abstract user Interface T
Concrete user Interface S
Concrete user Interface T
Final user Interface S
Final user Interface T
UsiXML supported model
UsiXML unsupported model
Reification
Abstraction
Reflexion
Translation

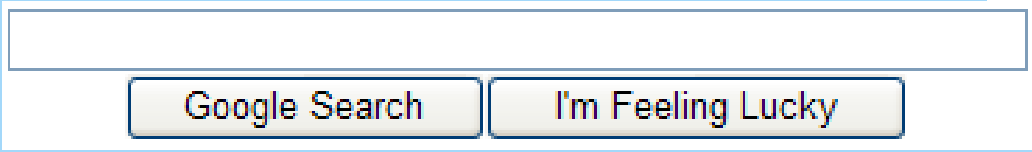

Container
Inputer
Activator
Activator

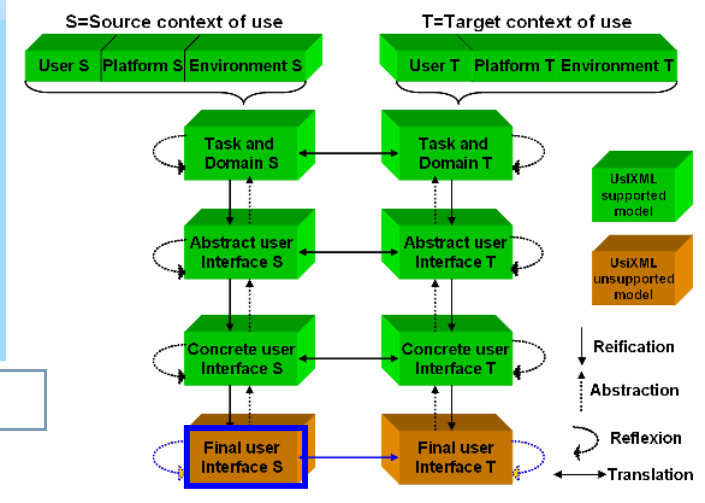
In practice


Window
TextBox
Button
Button

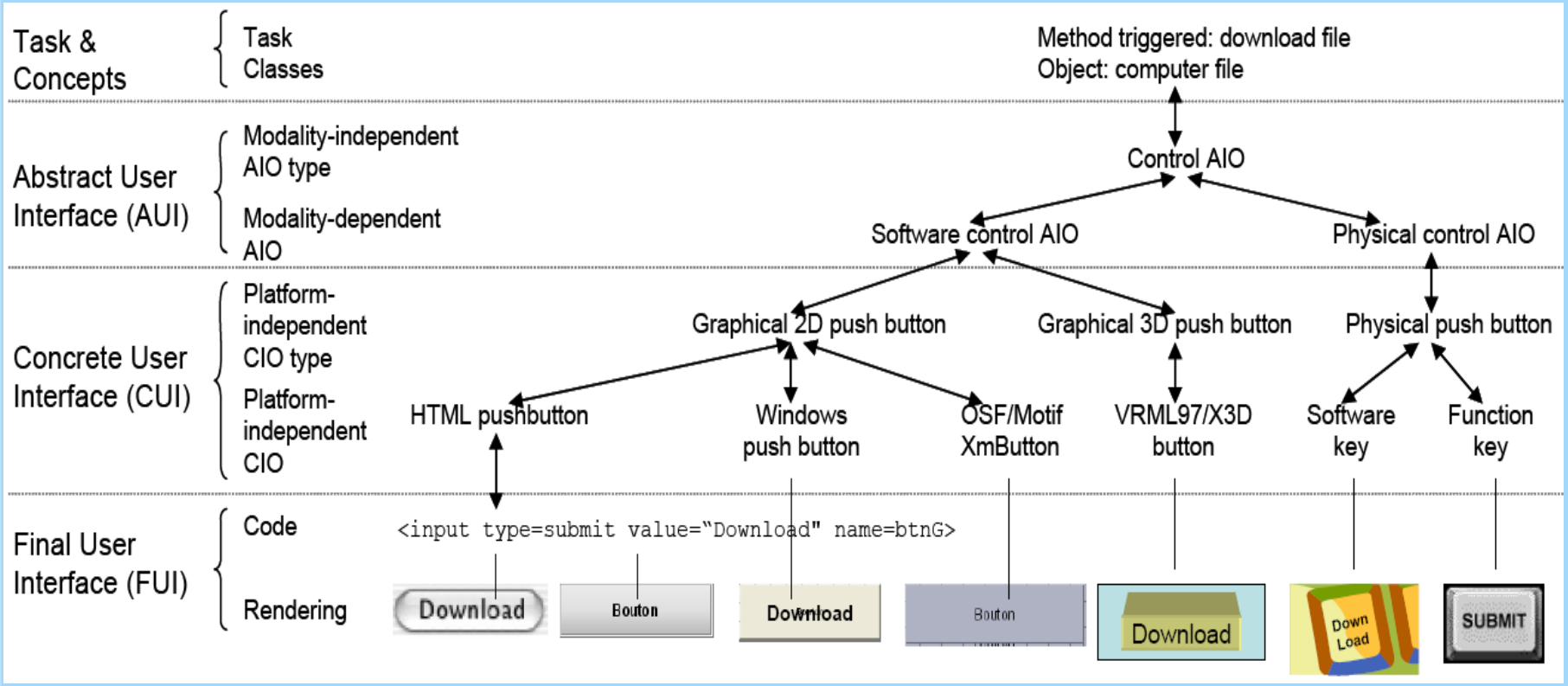

S=Source context of use
T=Target context of use
User S
Platform S
Environment S
User T
Platform T
Environment T
Task and Domain S
Task and Domain T
Abstract user Interface S
Abstract user Interface T
Concrete user Interface S
Concrete user Interface T
Final user Interface S
Final user Interface T
UsiXML supported model
UsiXML unsupported model
Reification
Abstraction
Reflexion
Translation

In practice


Google Search
I'm Feeling Lucky

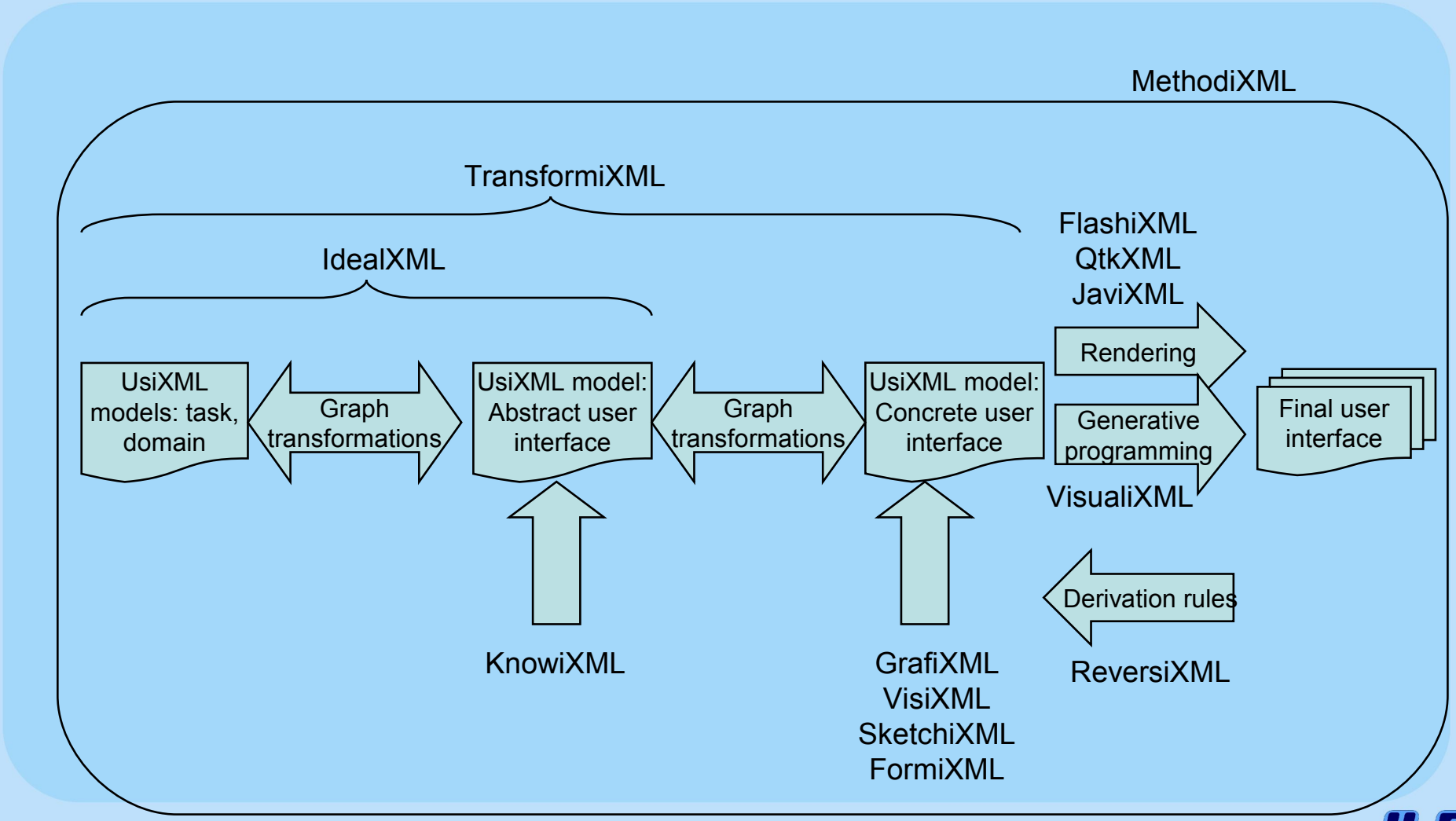

S=Source context of use
T=Target context of use
User S
Platform S
Environment S
User T
Platform T
Environment T
Task and Domain S
Task and Domain T
Abstract user Interface S
Abstract user Interface T
Concrete user Interface S
Concrete user Interface T
Final user Interface S
Final user Interface T
UsiXML supported model
UsiXML unsupported model
Reification
Abstraction
Reflexion
Translation

One description- Many representations

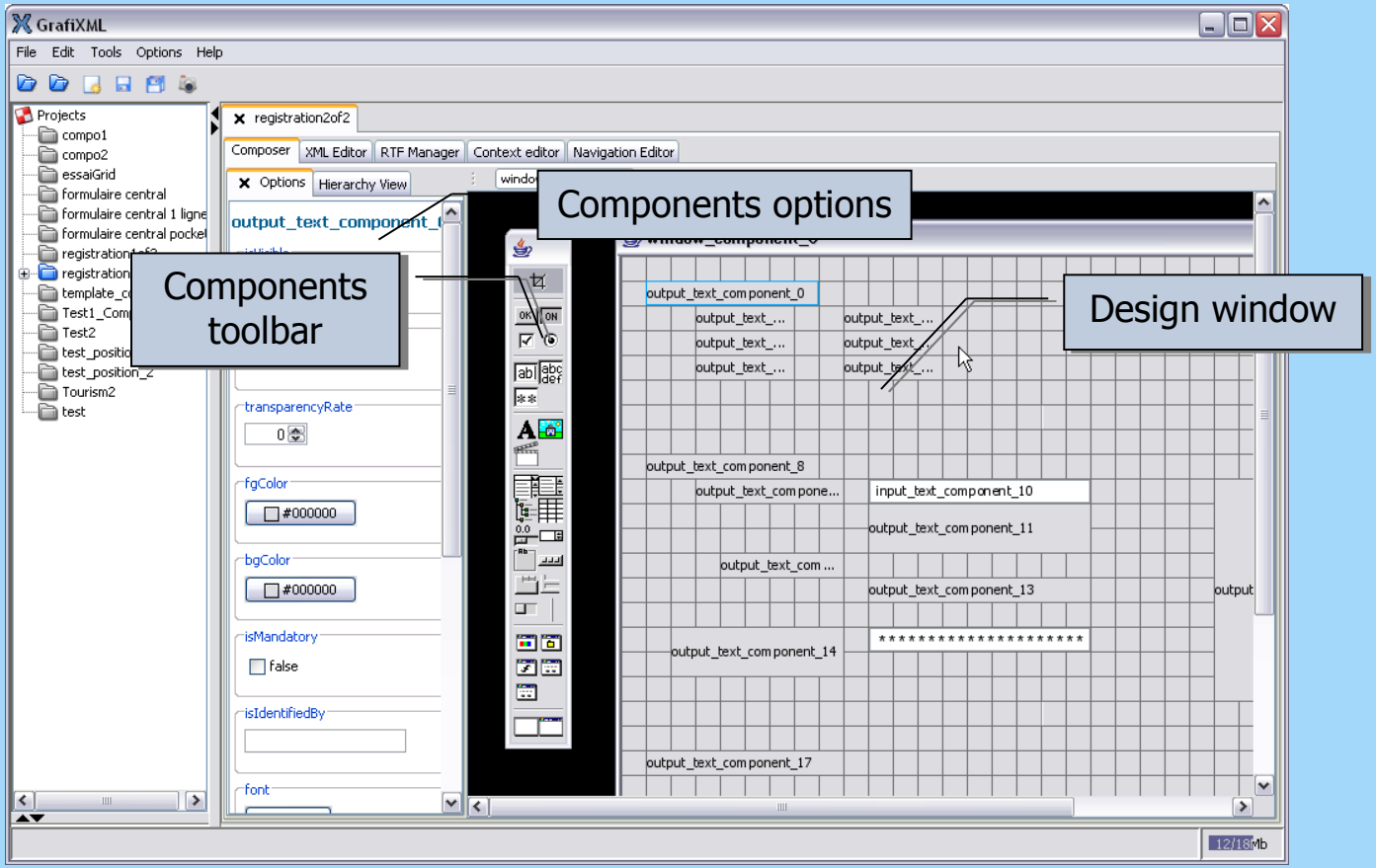


Tools

The big picture of MDA

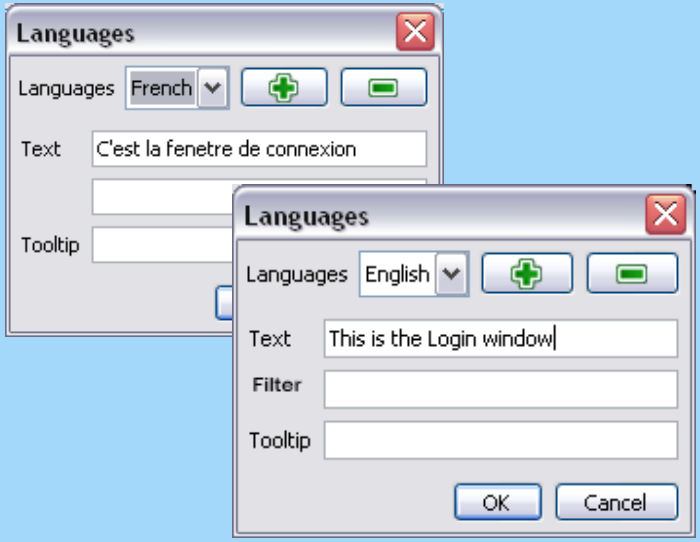


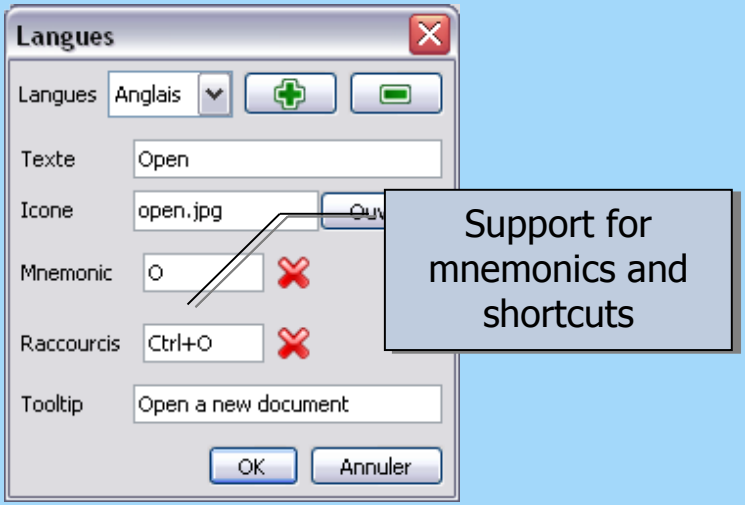
Design Tab


GrafiXML
File Edit Tools Options Help
Projects
compo1
compo2
essaiGrid
formulaire central
registration2of2
Composer
XML Editor
RTF Manager
Context editor
Navigation Editor
Options
Hierarchy View
Components options
Components toolbar
Design window
transparencyRate
fgColor
#000000
bgColor
#000000
isMandatory
false
isIdentifiedBy
font
output_text_component_0
output_text_component_8
input_text_component_10
output_text_component_11
output_text_component_13
output_text_component_14
output_text_component_17

Language

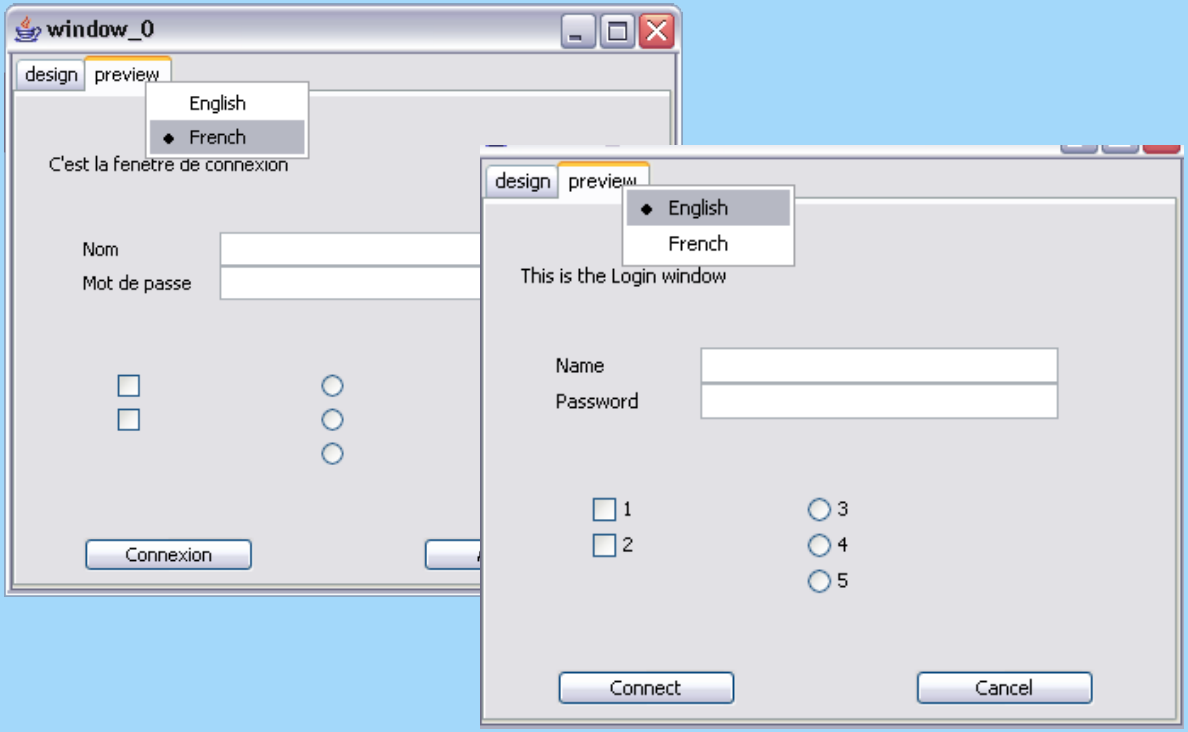
GrafiXML allows the user to create multi-language GUI





Preview

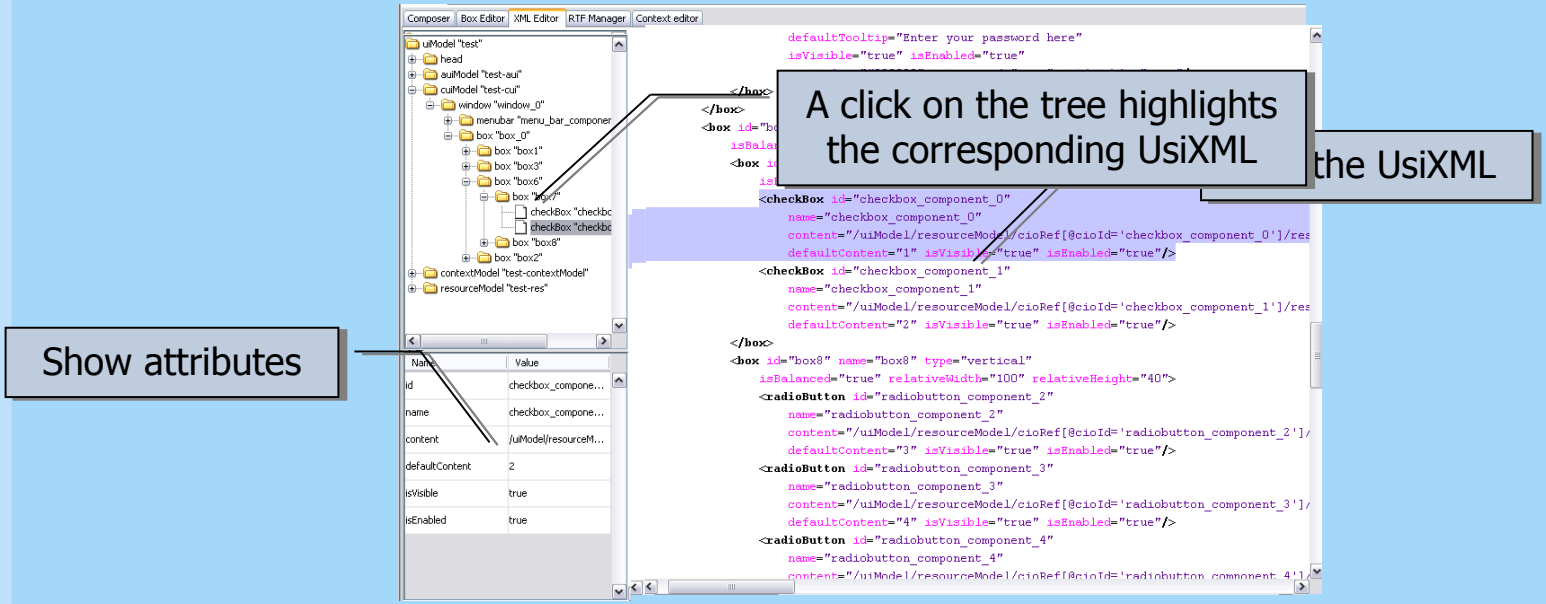
At any time, you can preview the ui in the language you want



XML Editor

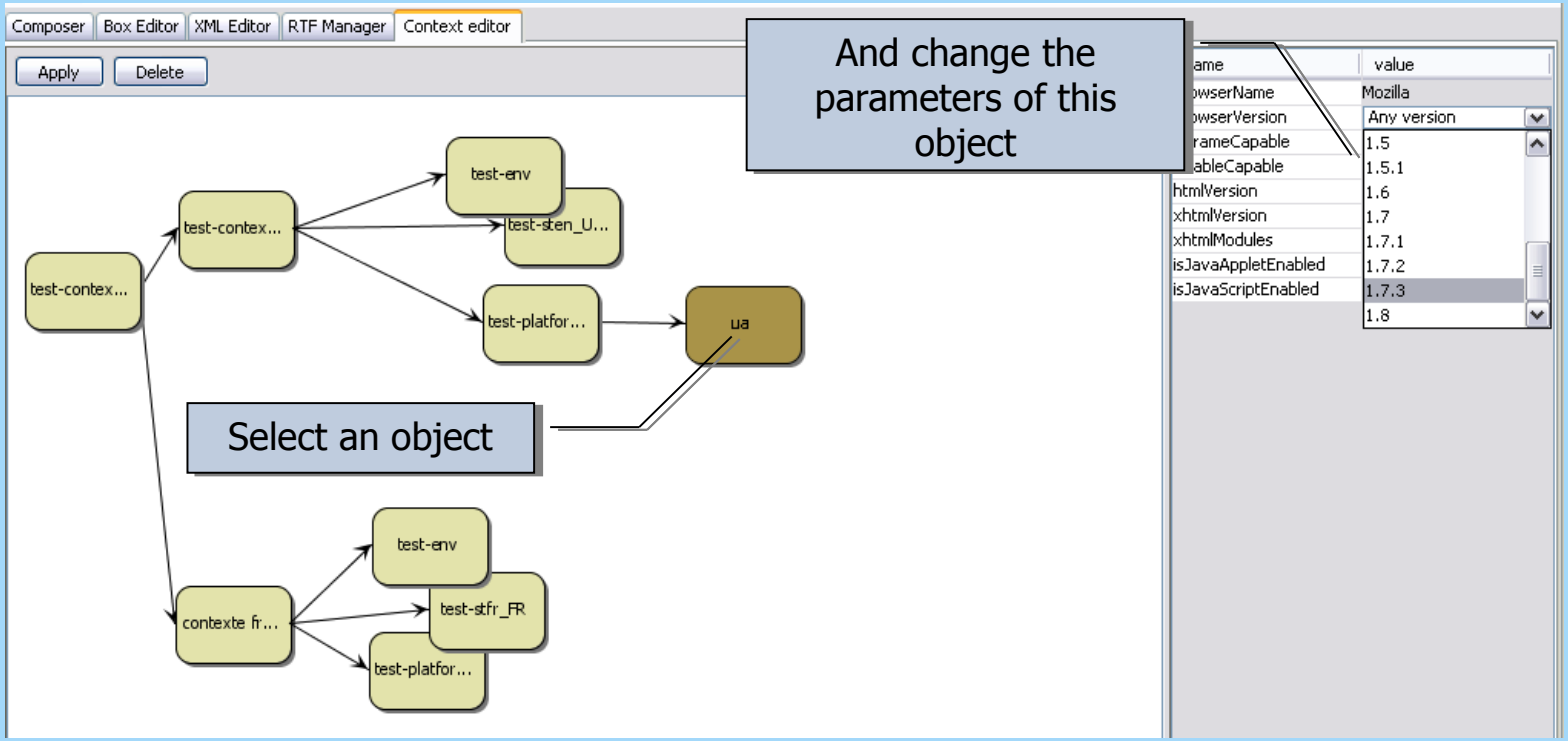
GrafiXML contains a XML editor which shows the UsiXML specification of your work

- You can edit yourself some part of the XML

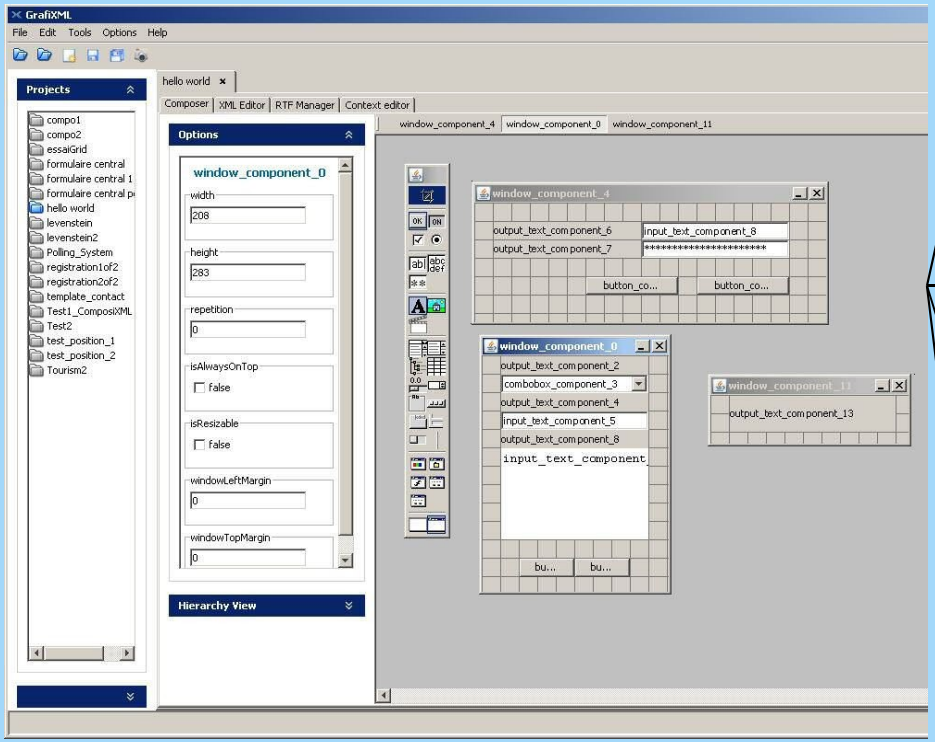


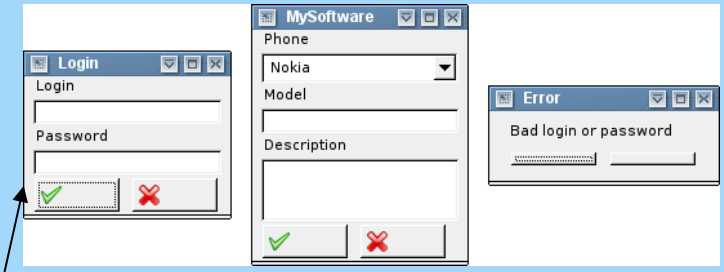
ContextModel Editor

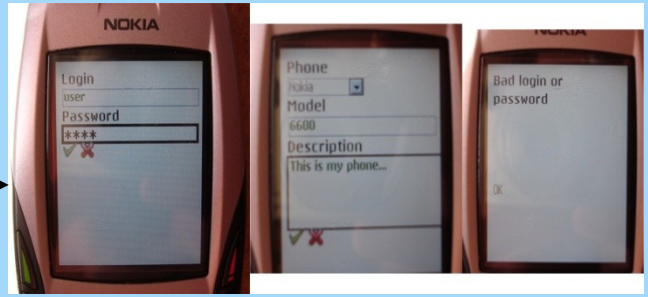
You can create a contextModel using Drag&Drop

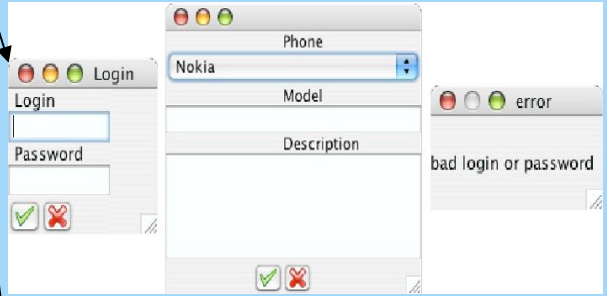


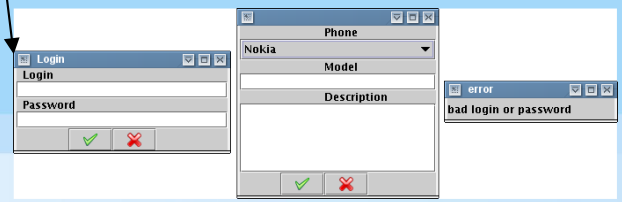
Example


GrafiXML
File Edit Tools Options Help
Projects
hello world
Composer XML Editor RTF Manager Context editor
Options
window_component_0
width
208
height
283
repetition
0
isAlwaysOnTop
false
isResizable
false
windowLeftMargin
0
windowTopMargin
0
Hierarchy View
window_component_4
window_component_0
window_component_11
output_text_component_6
input_text_component_8
output_text_component_7
button_co...
output_text_component_2
combobox_component_3
output_text_component_4
input_text_component_5
output_text_component_8
input_text_component
output_text_component_13


Login
Login
Password
MySoftware
Phone
Nokia
Model
Description
Error
Bad login or password


NOKIA
Login
user
Password
Phone
Nokia
Model
6600
Description
This is my phone...
Bad login or password


Login
Login
Password
Phone
Nokia
Model
Description
error
bad login or password


Login
Login
Password
Phone
Nokia
Model
Description
error
bad login or password

Thank you very much for your attention

http://www.usixml.org

User Interface eXtensible Markup Language

http://www.similar.cc

European network on Multimodal UIs

For more information and downloading,

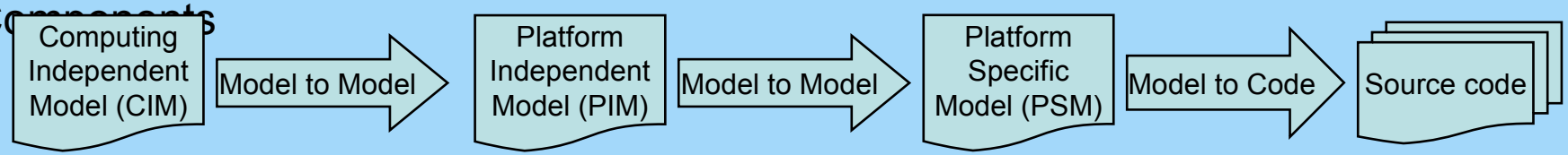
http://www.isys.ucl.ac.be/bchi

Special thanks to all members of the team!

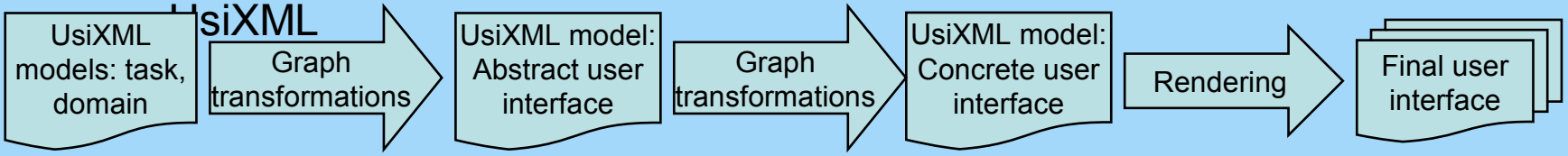
More Details

MDE based on UsiXML

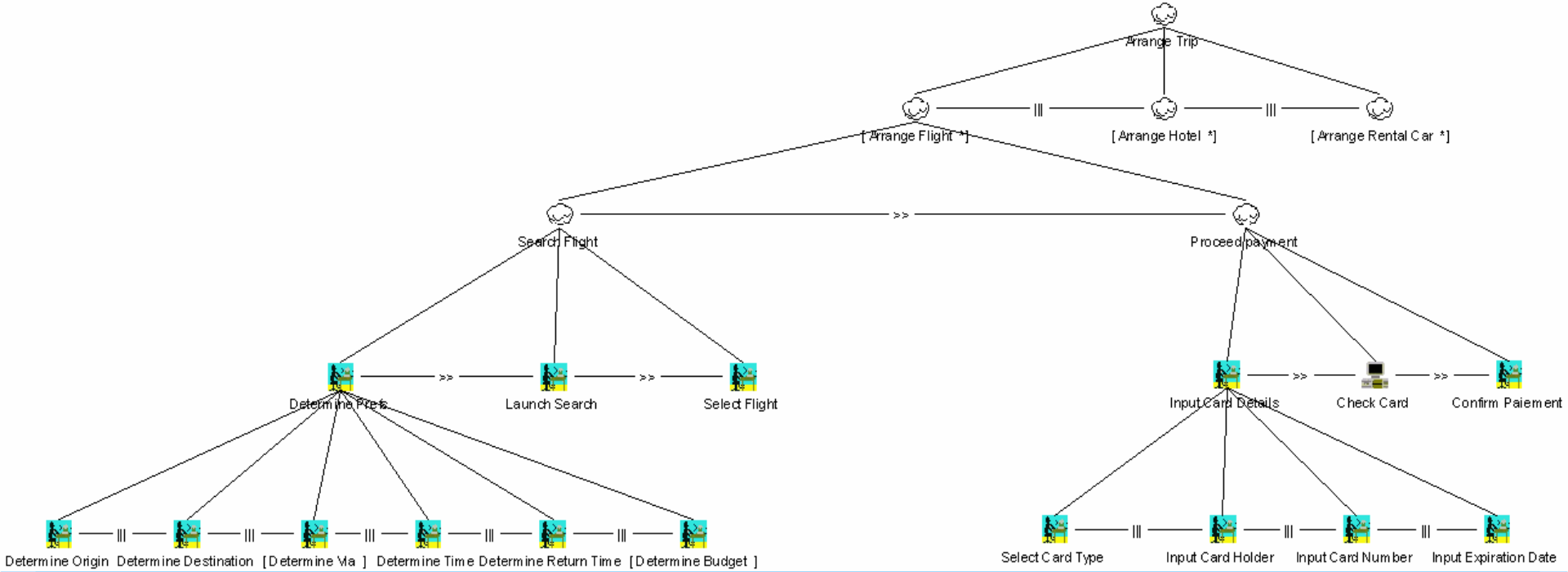
MDA Components



Techniques proposed based on UsiXML



CIM Step 1: Task model



New Abstraction: the user's task

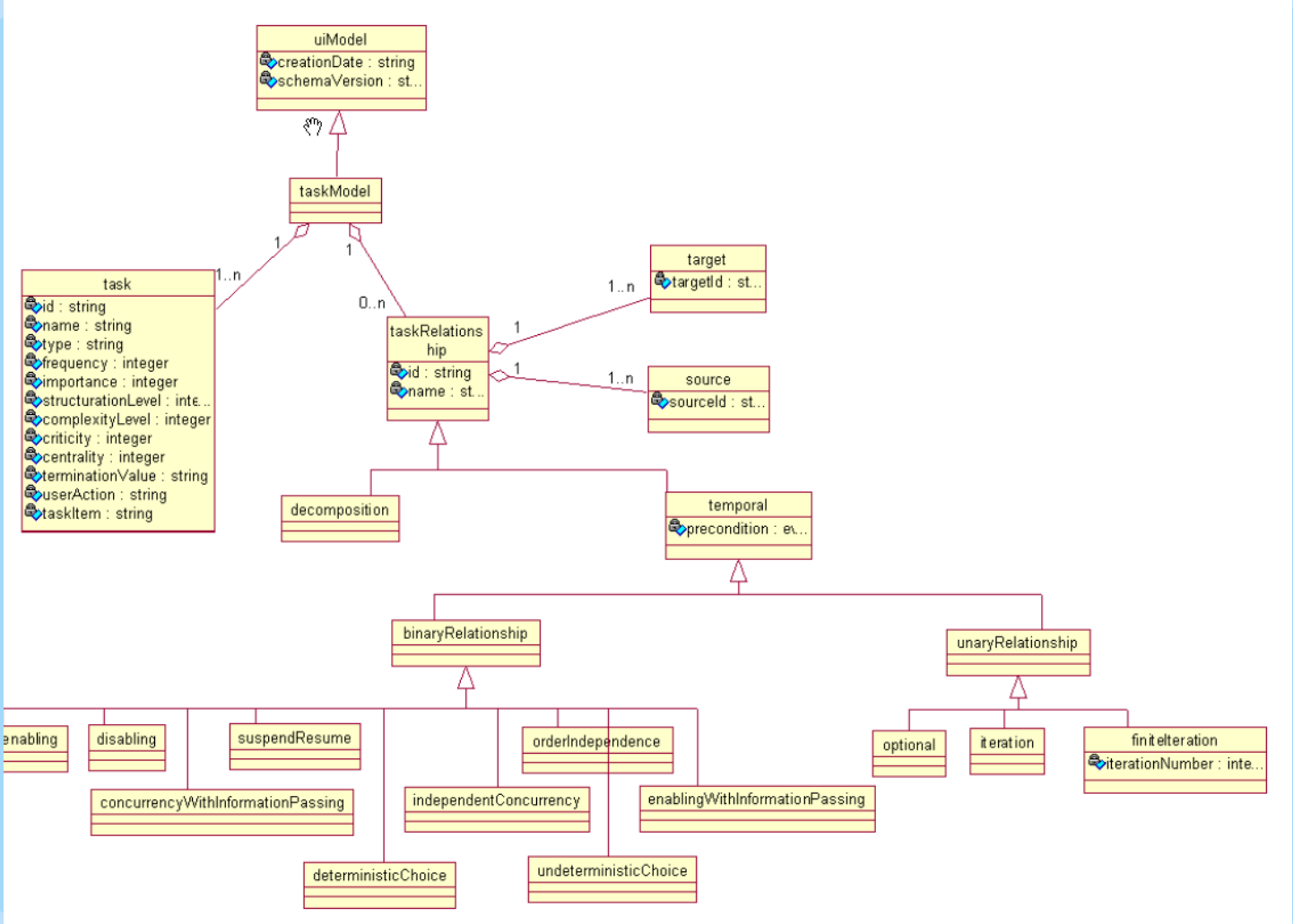
- Task = set of actions carried out by a user in a given context to reach a goal
- Logical decomposition of task into sub-tasks
- Temporal ordering: LOTOS operators (in CTTE)
  - T1 >> T2 Enabling
  - T1[ ]>>T2 Enabling + information passing
  - T1 |> T2 Suspend/resume
  - T1 [ ] T2 Non-deterministic choice
  - T1 π T2 Deterministic choice
  - T1 [> T2 Disabling (e.g. Form submit)
  - T1 |=| T2 Independence (any order, but finished)
  - T1* Iteration
  - T1{n} Finite iteration
  - T1 ||| T2 Concurrency
  - T1 |[x]| T2 Concurrency + information passing
  - [T] Optional
  - T Recursion

[Markopoulos,1992]

New Abstraction: the user's task

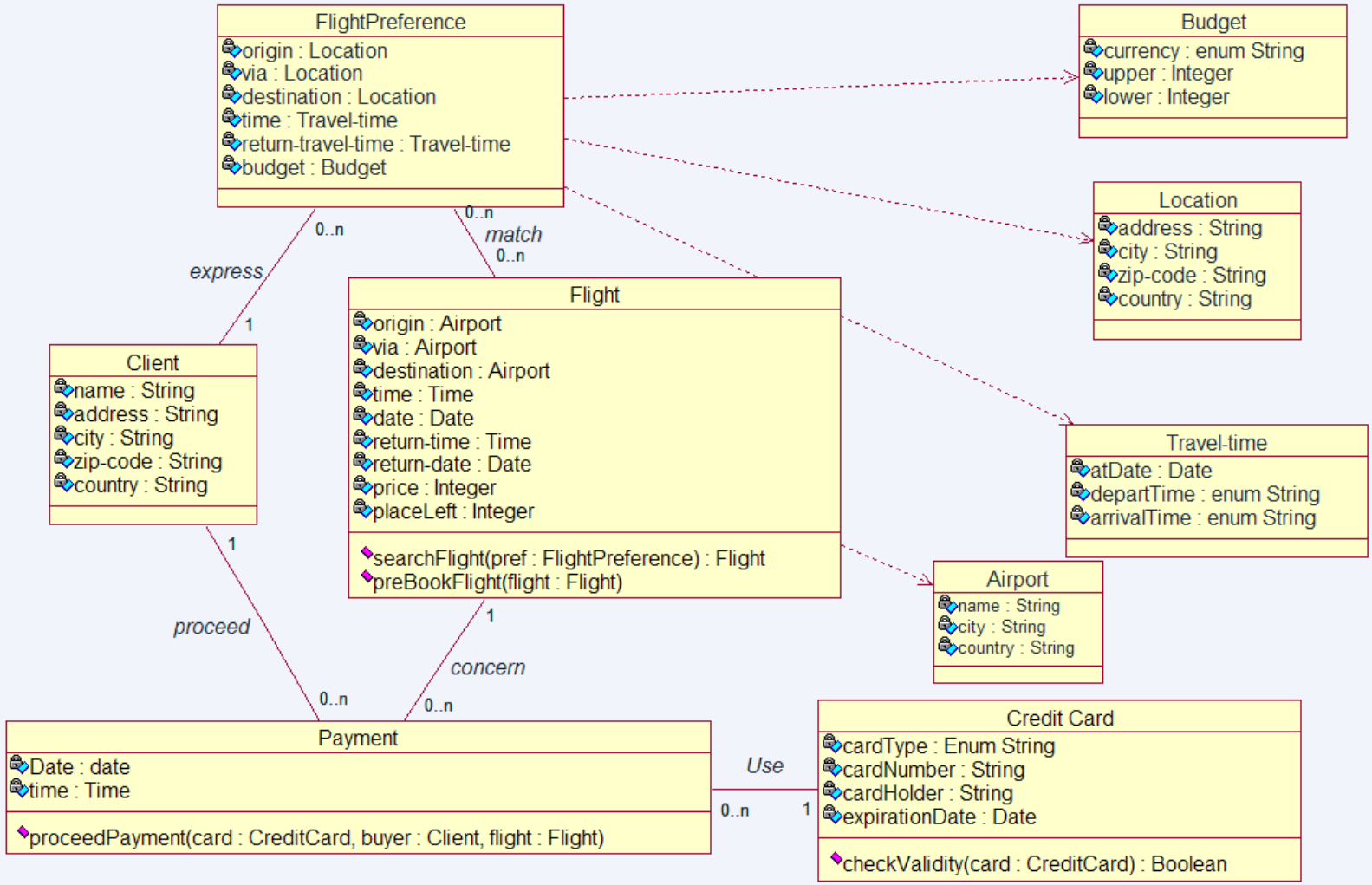
- Task definition = action + object
  - Action types
    - CRUD pattern: create, read, update, delete
    - Select, control,…
    - Acquire, render, modify, publish, compute, derive,…
  - Object types:
    - Element, list, table, collection, compound,…

New Abstraction: the task meta-model



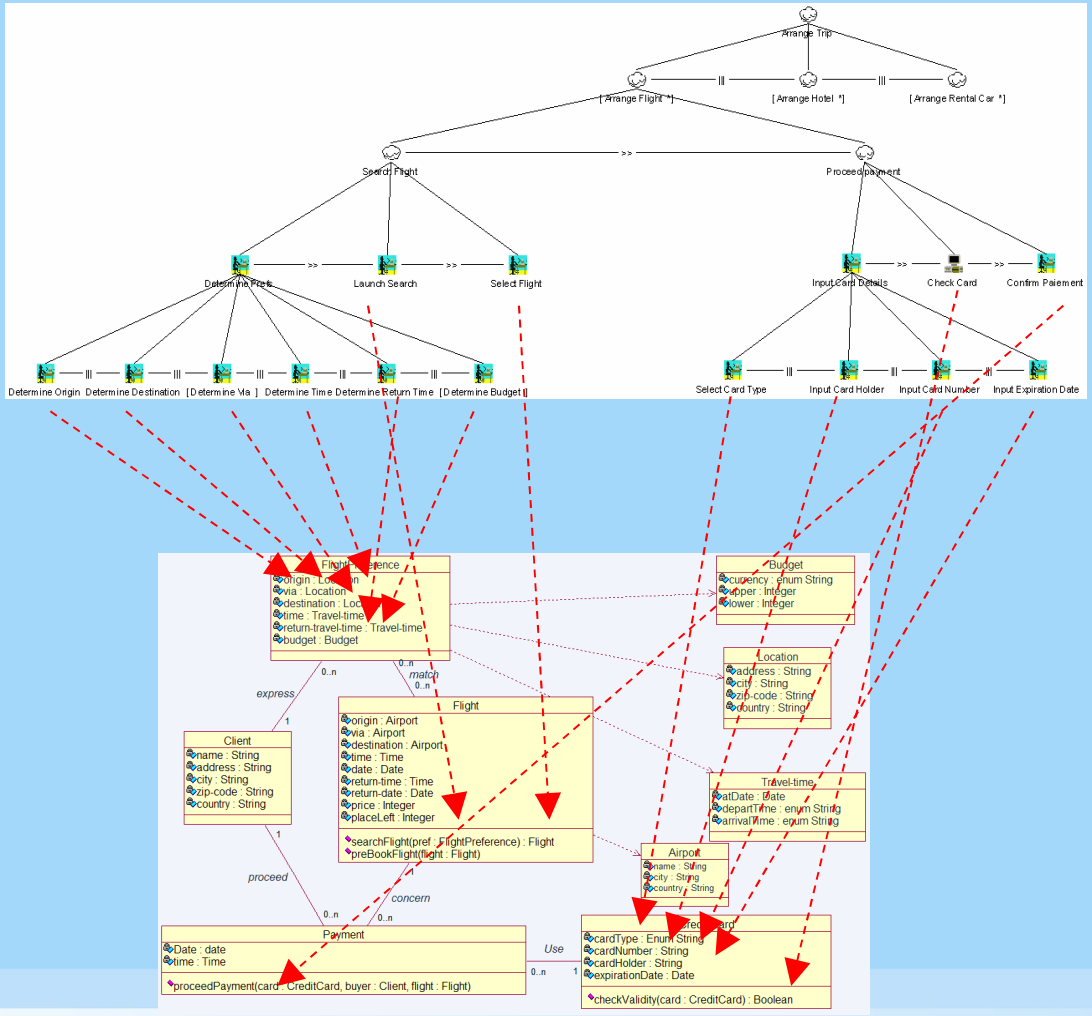
[Limbourg,2004]

CIM Step 2: domain model



[Limbourg,2004]

CIM Step 3: Task-domain mappings



IdealXML

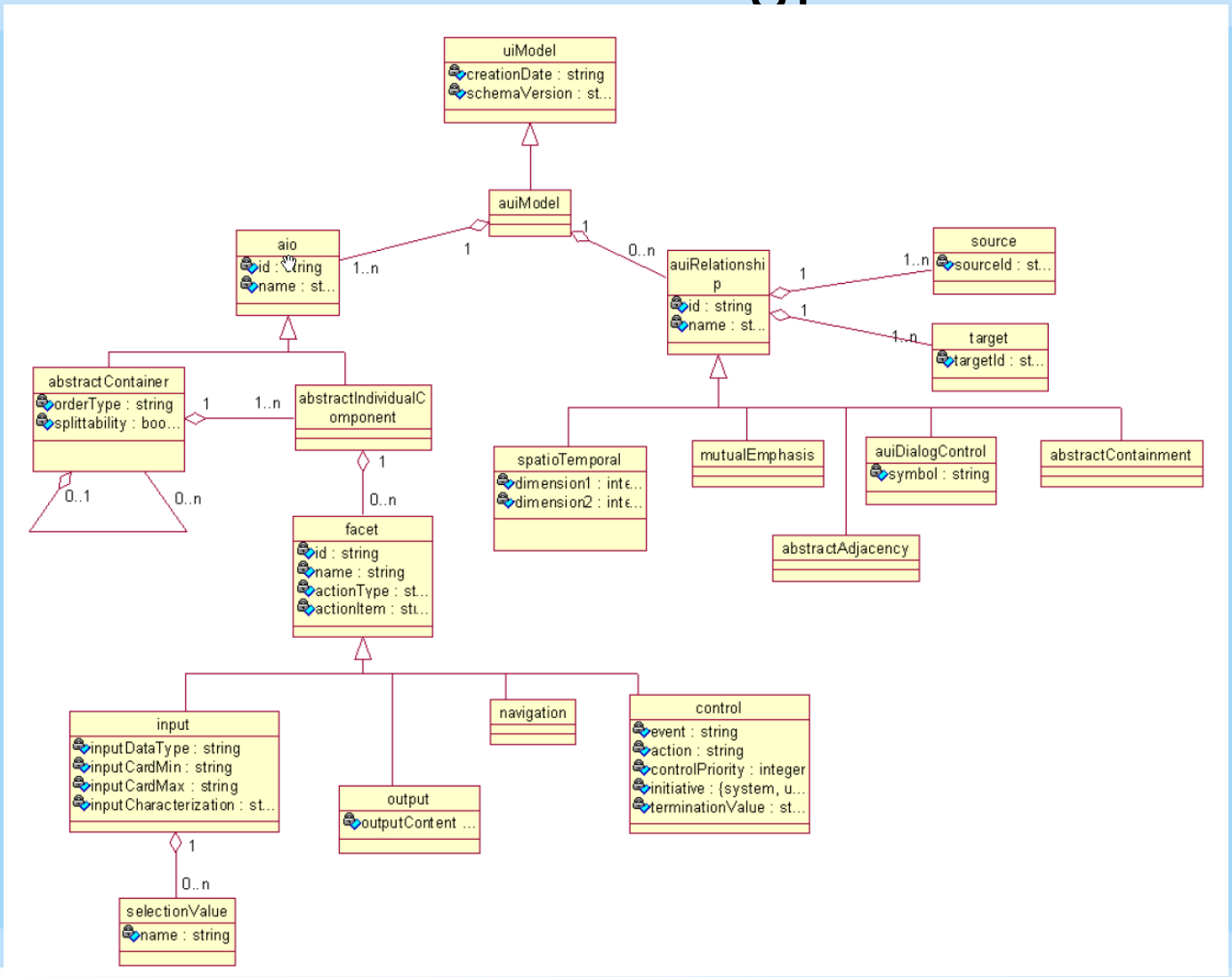
[Limbourg,2004]

New Abstraction: the abstract UI

- Different CIOs can be used for the same purpose, but with different interaction modalities
- Definition
  - Abstract Container = set of Abstract Individual Component
  - AIC = abstraction of CIOs of the same type, but independently of any interaction modality
  - Abstract User Interface (AUI) = decomposition into AC+AIC

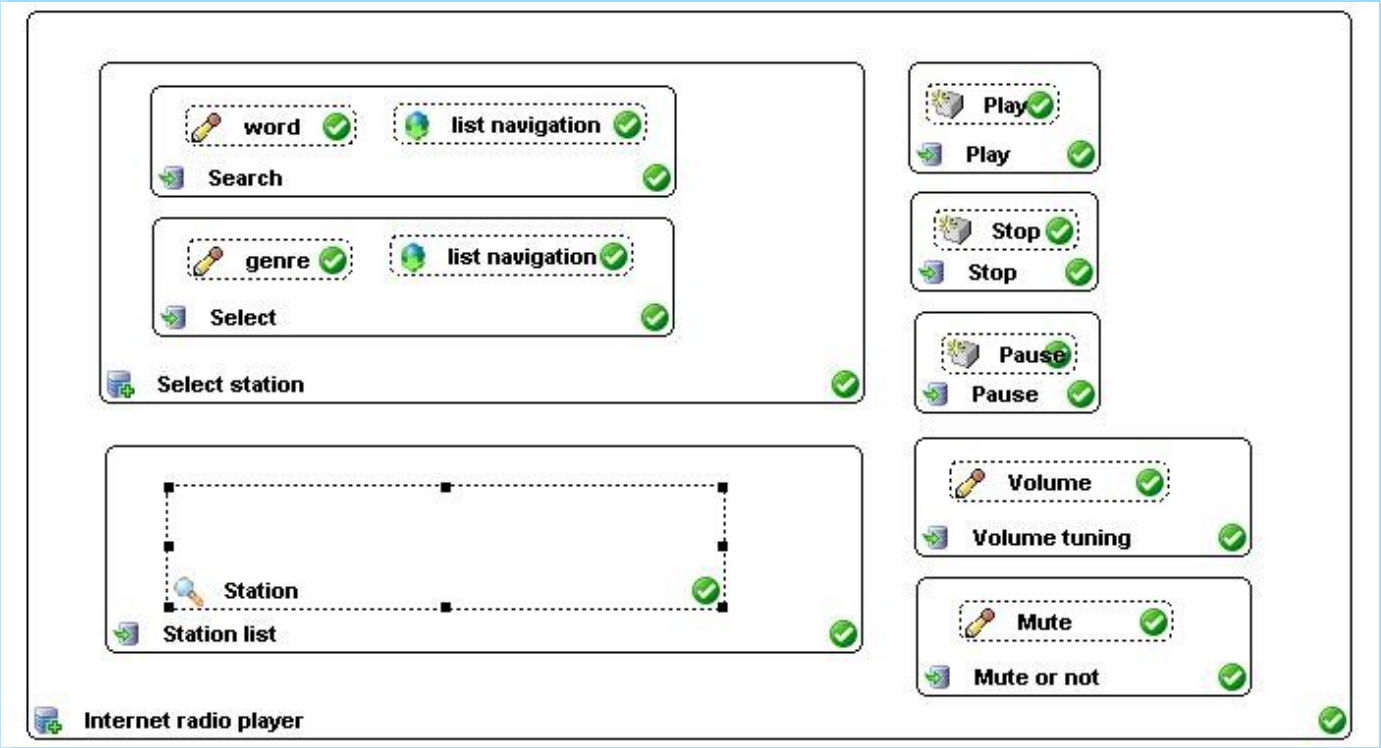
[Vanderdonckt & Bodart,1993]

Abstraction: the abstract UI



Abstraction: the abstract UI

- Notation: based on L. Constantine's notation for canonical abstract prototypes [Constantine,2003]

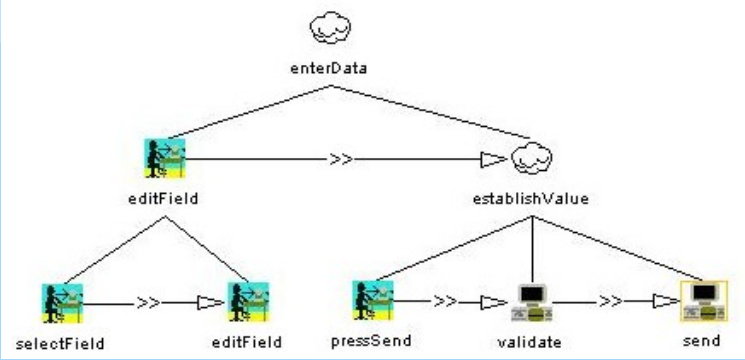


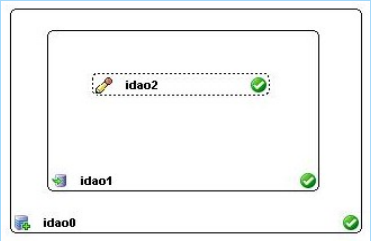
- Abs.container
- Abs. component
- Input
- Output
- Navigation
- Control
- Select

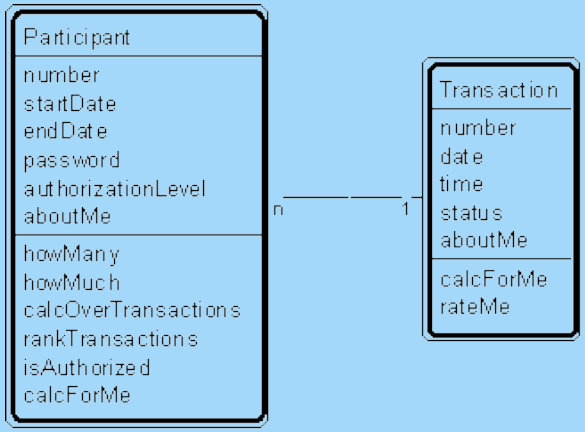
IdealXML

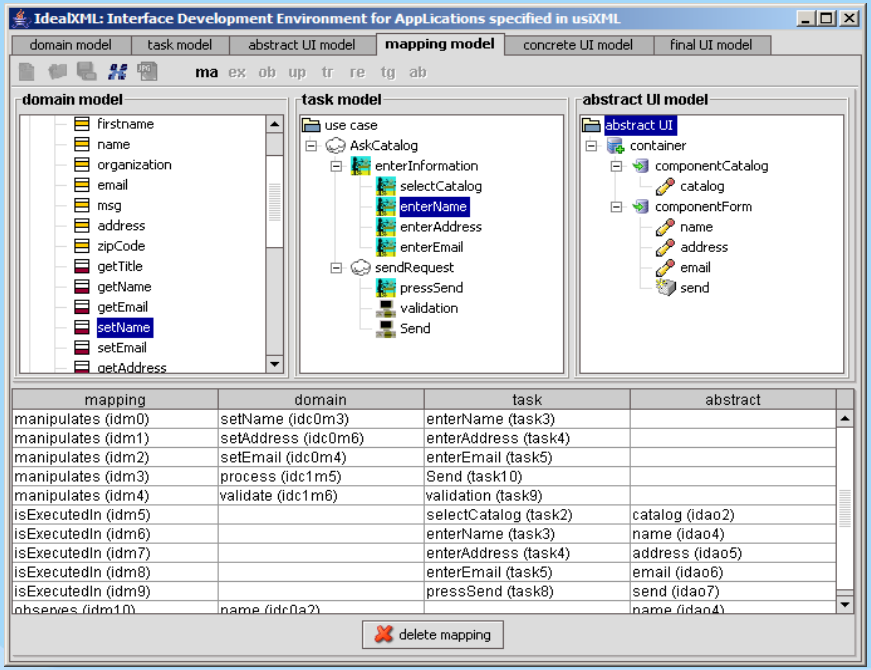
[Montero et al.,2005]

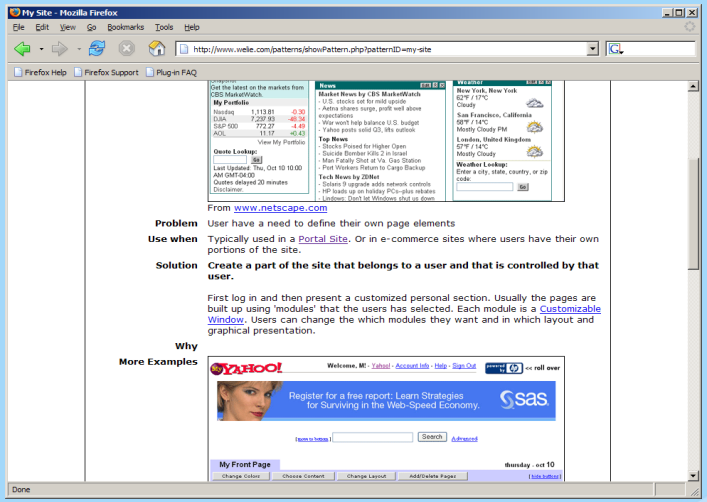
IdealXML





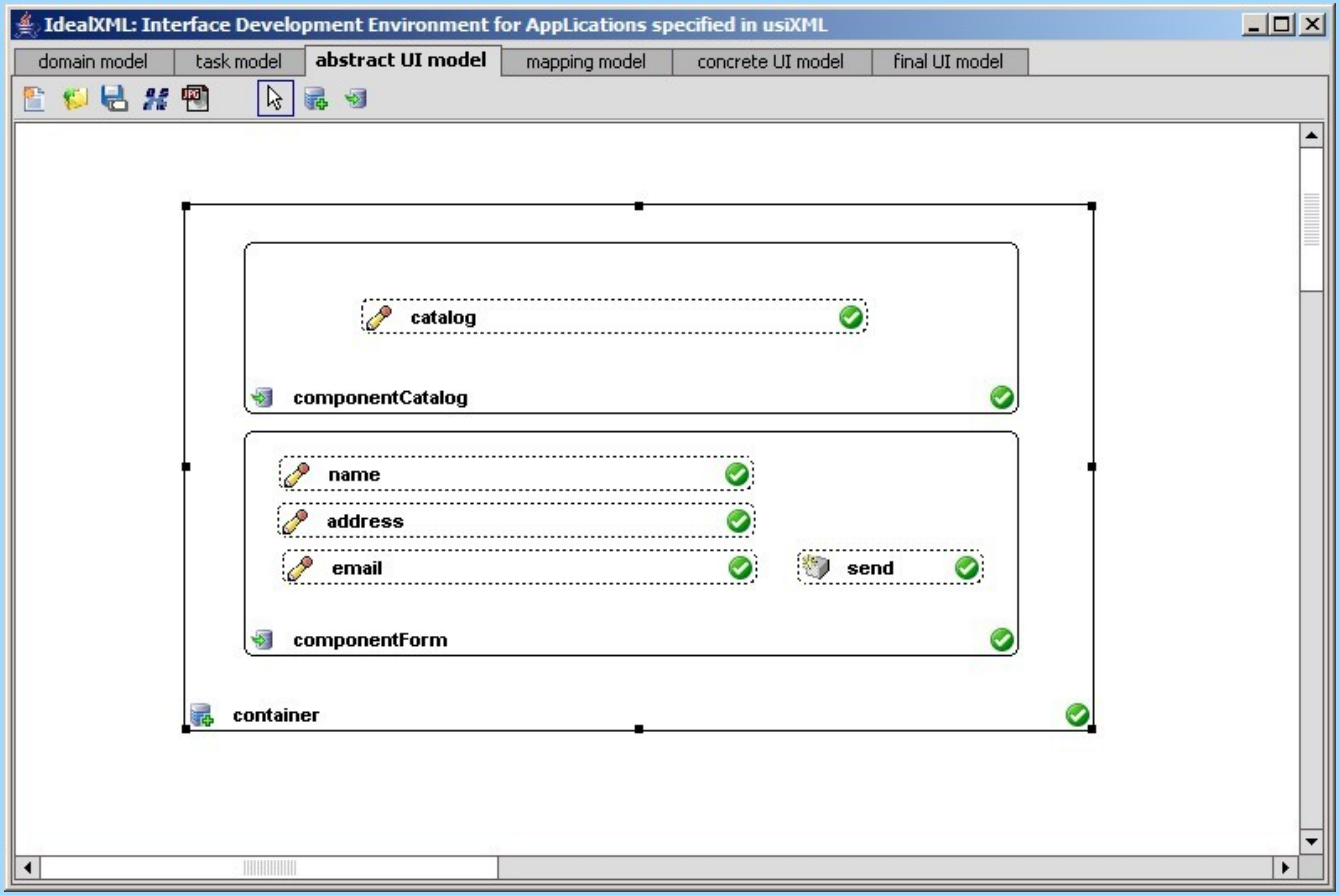




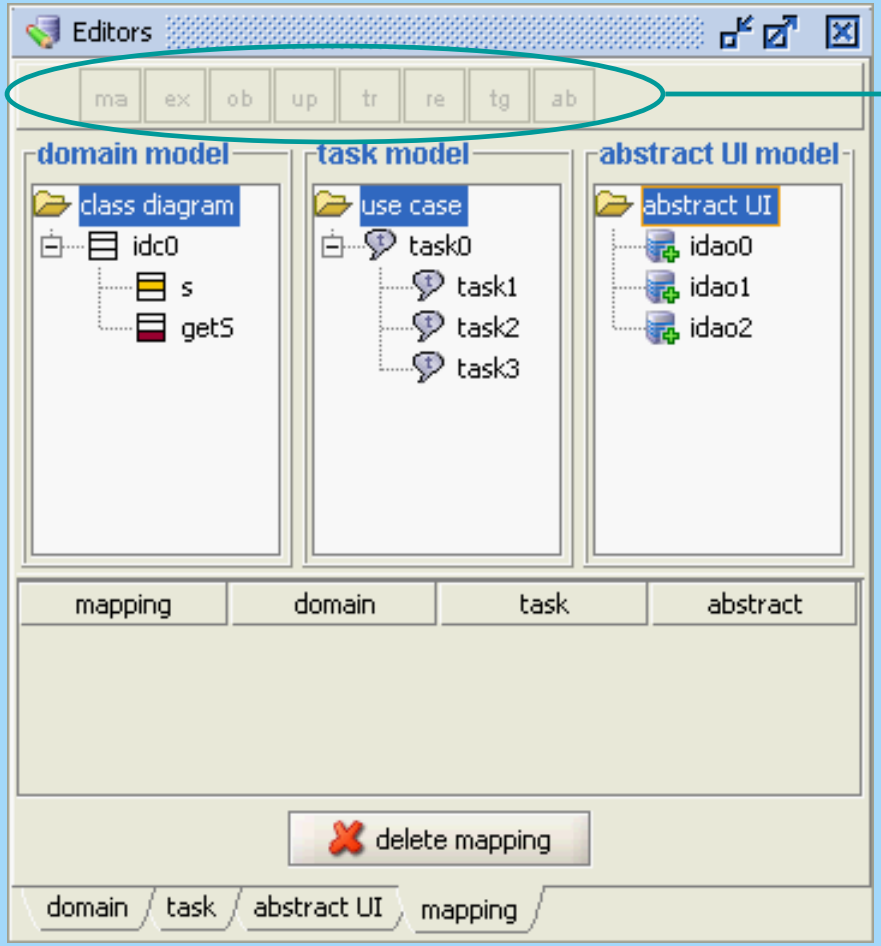


[Montero,2005]

Example of AUI produced



Mapping the models



These mappings can be established:

triggers (tg): { , } x

updates (up): x

observes (ob): x

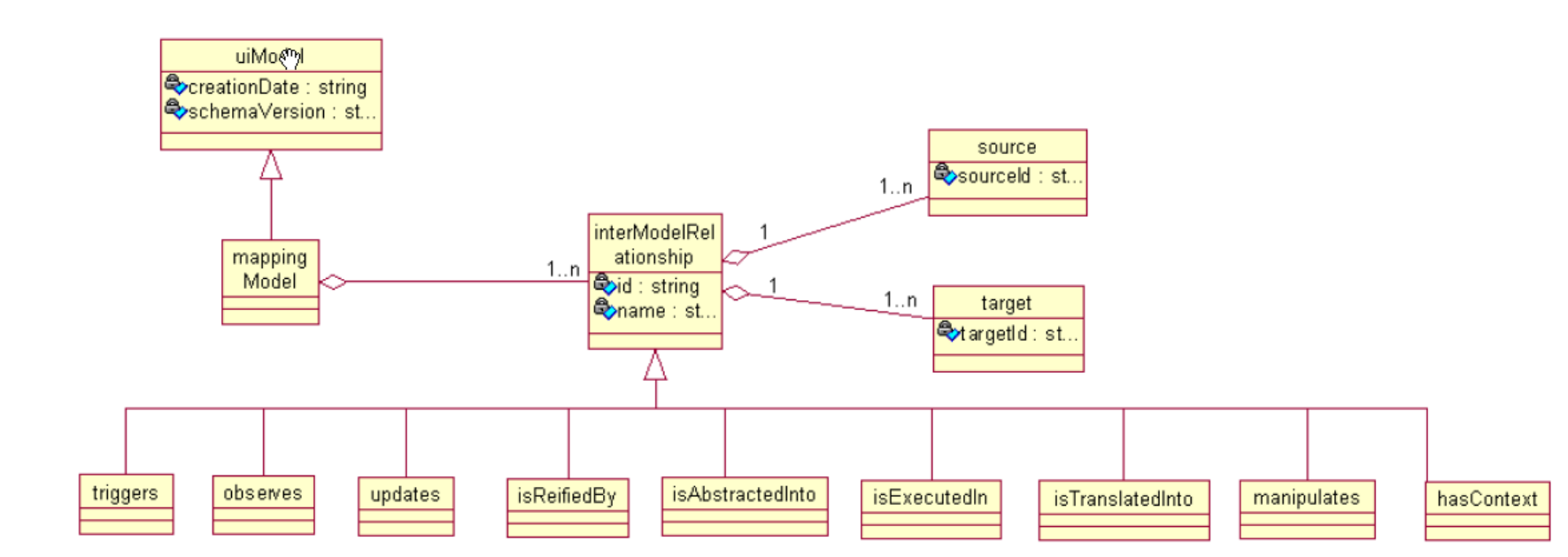
isExecutedIn (ex): x

manipulates (ma): { , } x

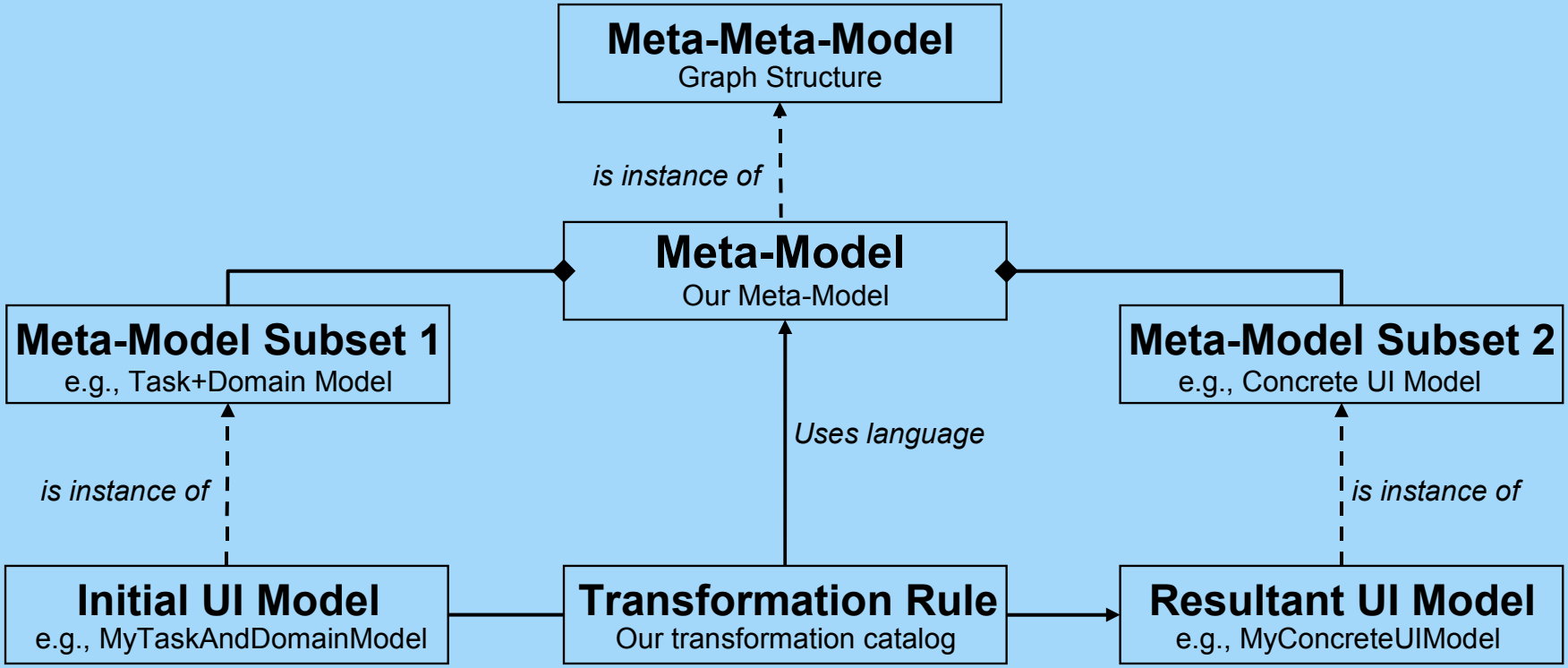
[Montero,2005]

Mapping the models

- Mapping the models with a mapping model (!!)



Typed Model-to-Model Transformation



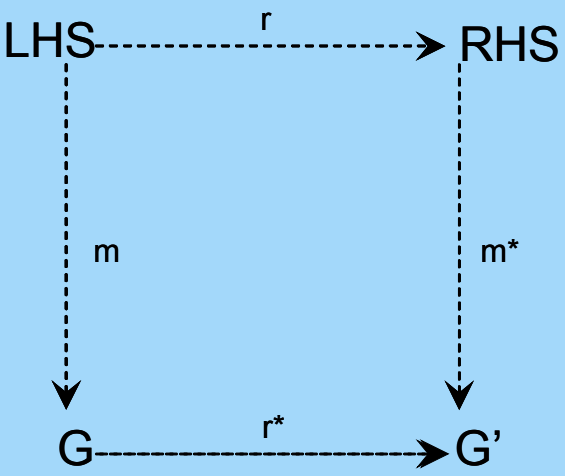
[Limbourg,2004]

Expression of models as graphs

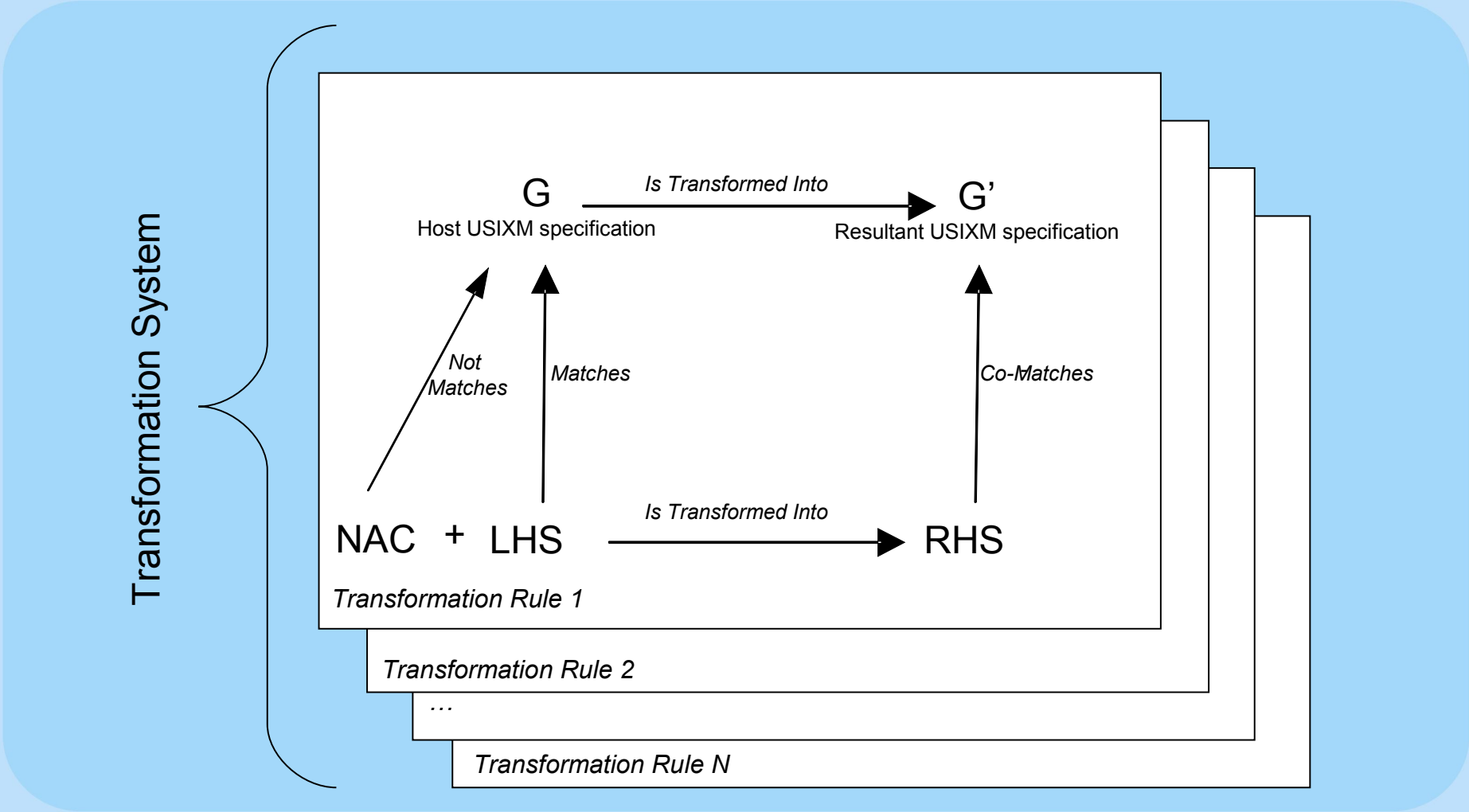
- All transformations are in UsiXML
  - Each model = instance of meta-model
  - Each model = graph as instance of graph type
  - Each model transformation =
    - graph transformation
    - Set of productions

Definition of a production

- Find an occurrence of LHS in G (this occurrence is called a match). If several occurrences exist, choose one non-deterministically.
- Check preconditions of both type PAC and NAC. If not verified, then skip.
- Remove the part of G which corresponds to (LHS – K), where K is the morphism specified between LHS and RHS.
- Add RHS – K into G – (LHS – K) as it is given by the corresponding relation between RHS – K and K
- Check postconditions of both type PAC (and notably that the resulting graph is properly typed) and NAC. If not verified, then undo the transformation rule.



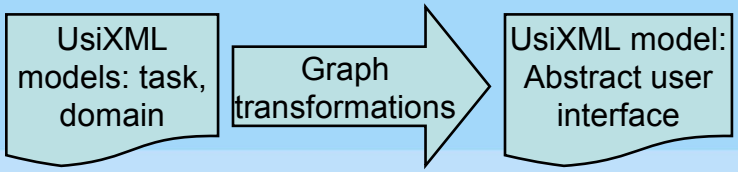
Transformation system



[Limbourg,2004]

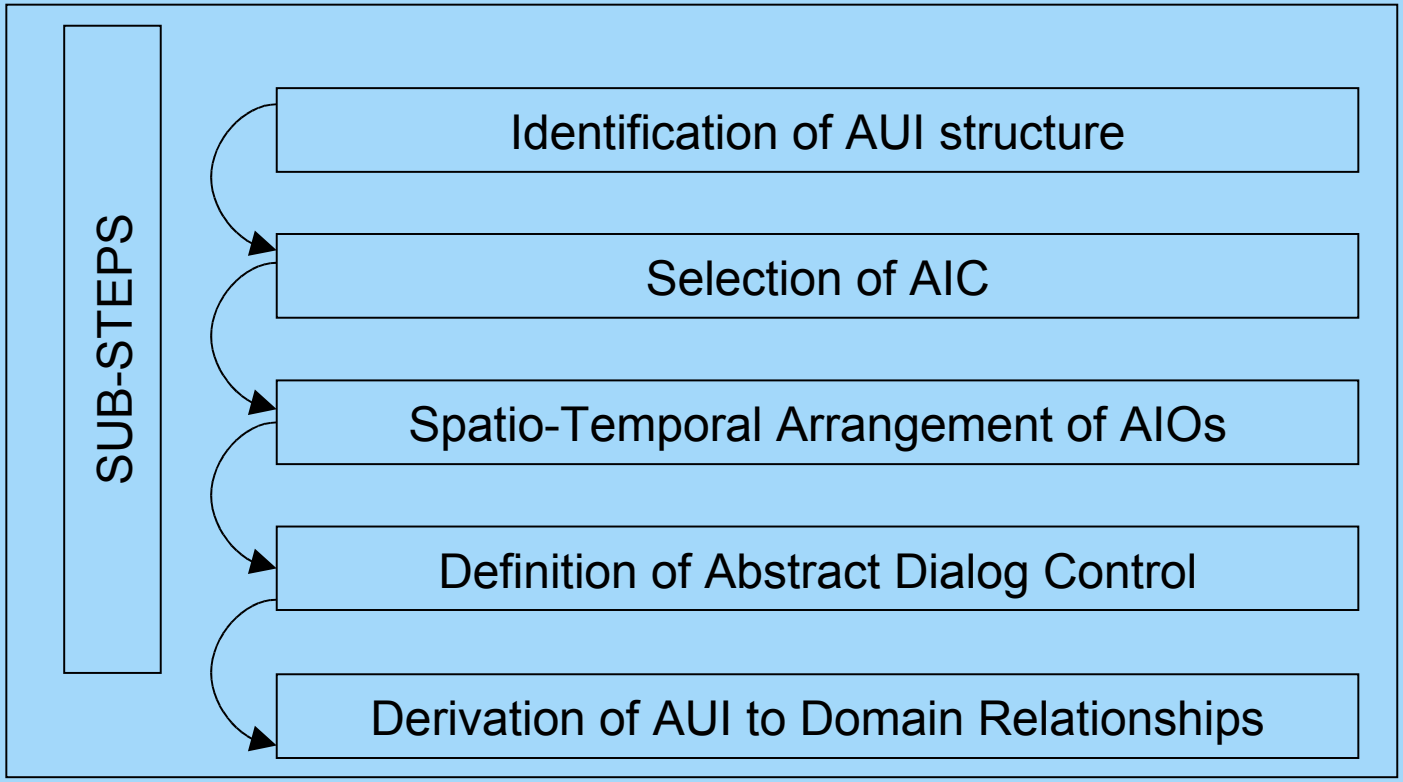
PIM step: task+domain to AUI

- Abstract UI (AUI) = UI independent of any interaction modality
- Definition of AUI structure in terms of Abtract Containers (AC)
  - Which tasks should be logically grouped?
- Definition of Abstract Individual Components (AIC) types
  - Which « functionnality » should assume AICs and what data do they manipulate ?
- Definition of spatio-temporal arrangement
  - How should AIC be arranged in space and time ?
- Definition of dialog control
  - What is the valid flow of action on AICs ?



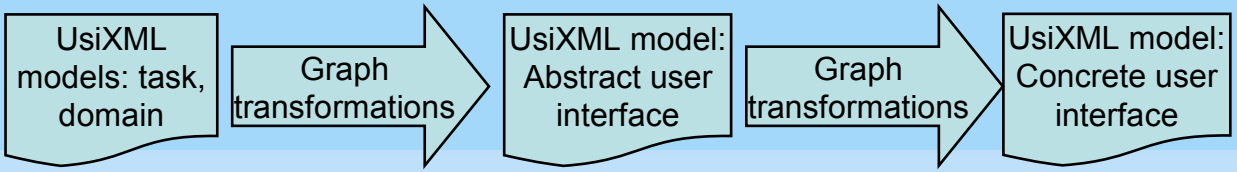
PIM step: task+domain to AUI

STEP : From Task & Domain to AUI

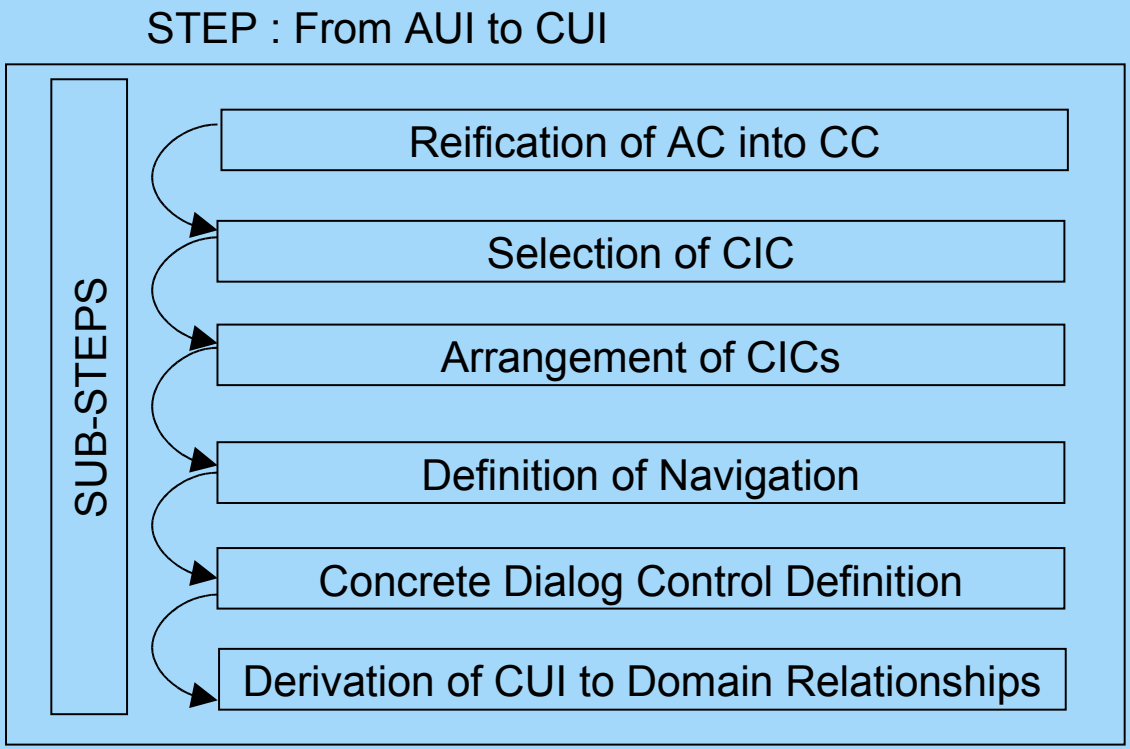


PSM Step: AUI to CUI

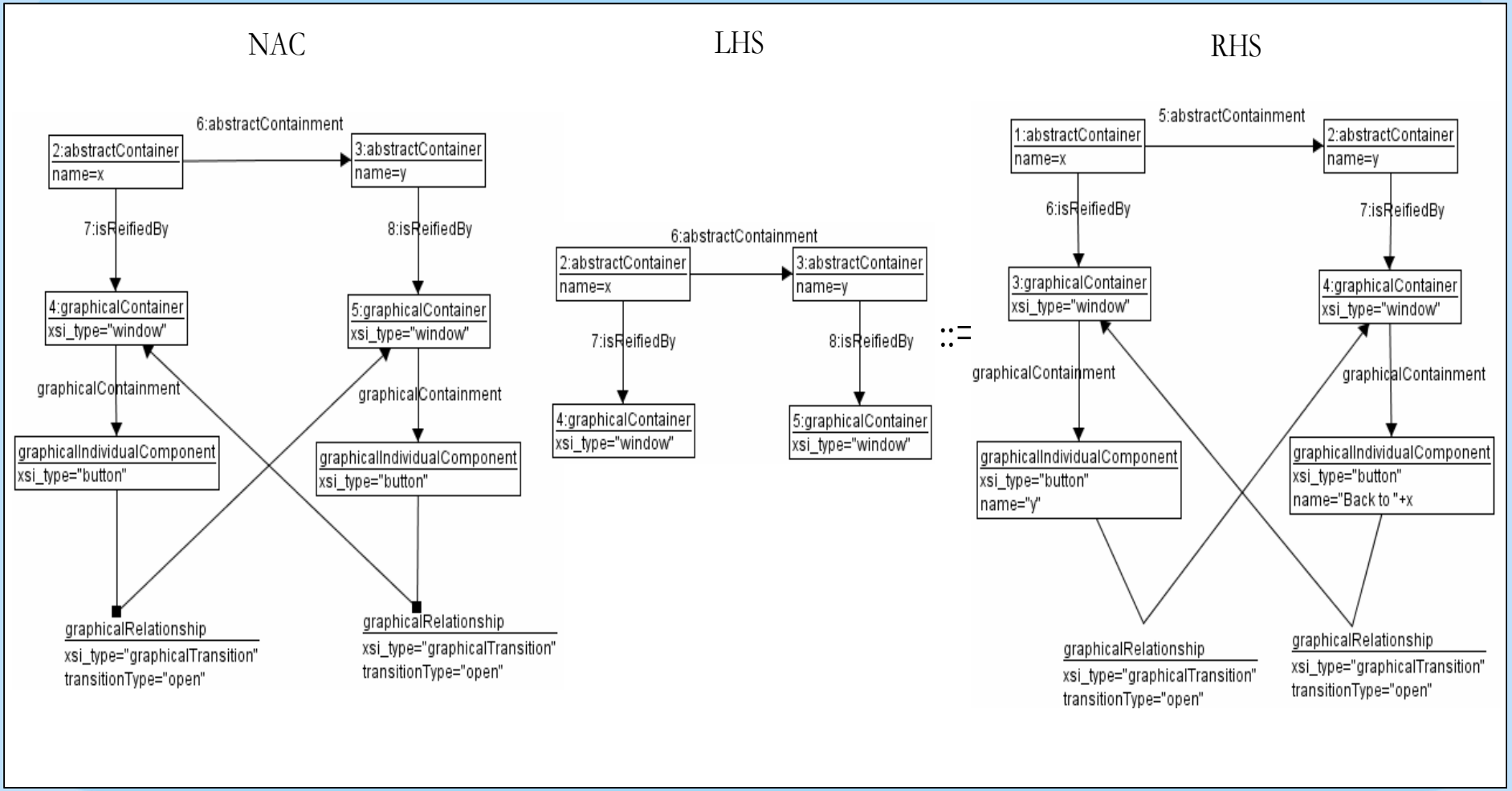
- Concrete UI (CUI) = UI independent of toolkit
- Definition of CUI structure
  - Which AIC is a window?
- Definition of Concrete Interaction Component (CIC) type
  - Which « widget » should represent which AIC ?
- Definition of placement
  - What layout can be specified between CICs,…
- Definition of navigation
  - Which container can be started or closed from which container?
- Definition of dialog control
  - What is the valid flow of action on AIOs



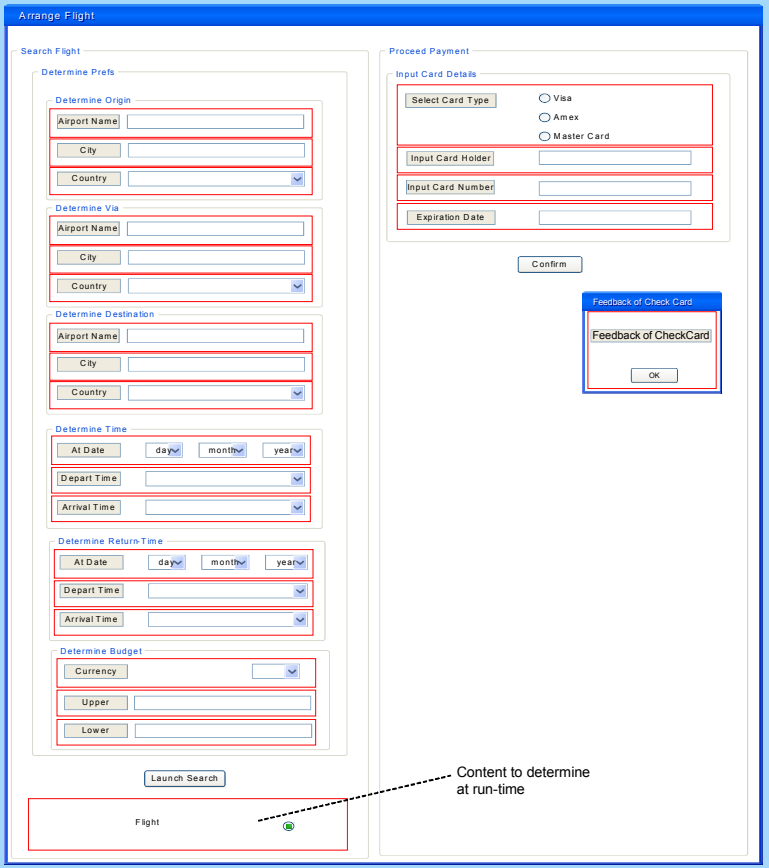
PSM Step: AUI to CUI

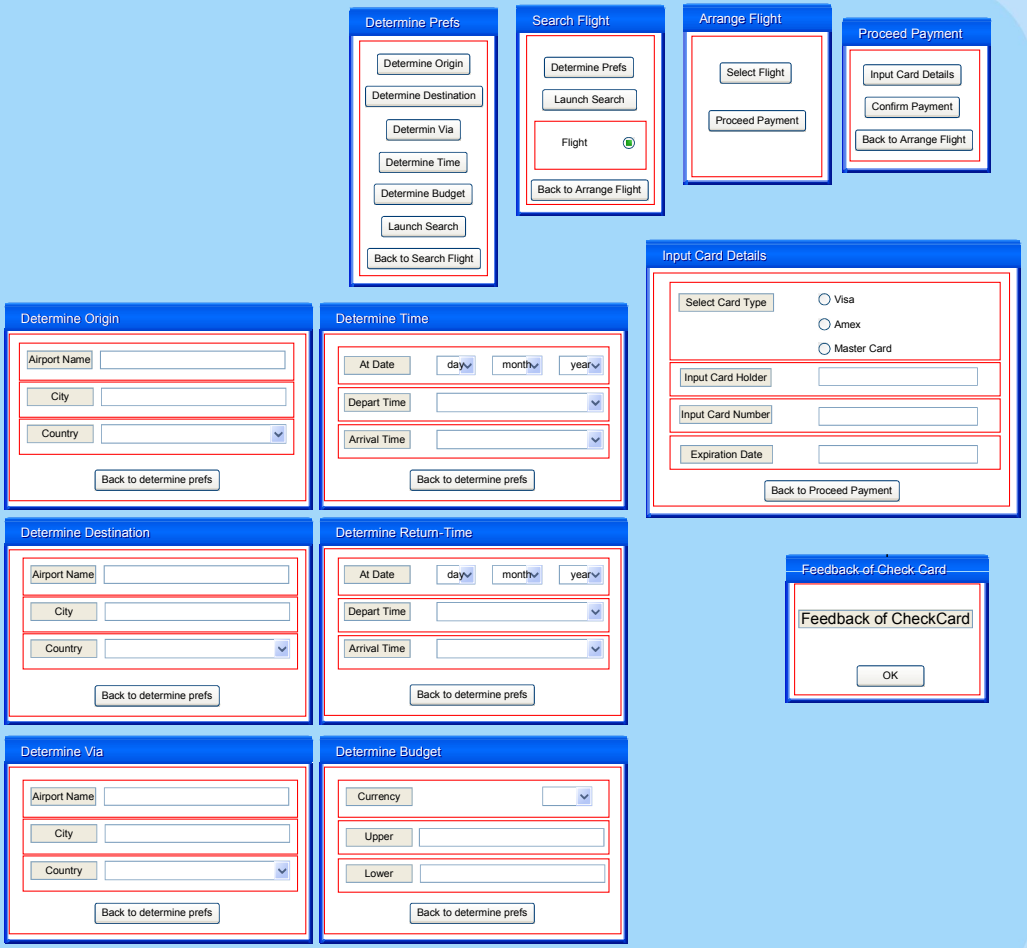


PSM sub-step 3: definition of navigation
An example of a complex rule

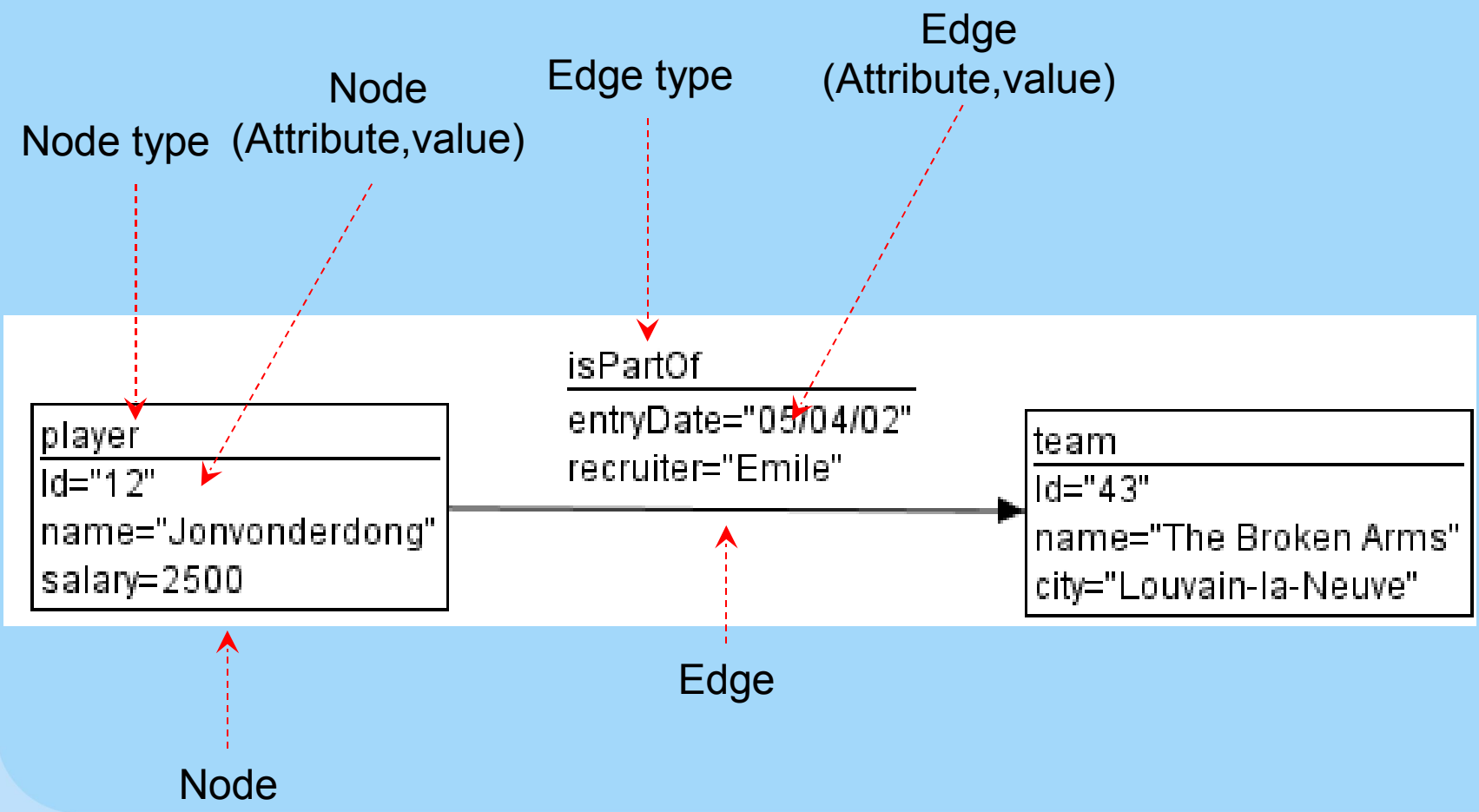


PSM: Concrete User Interface

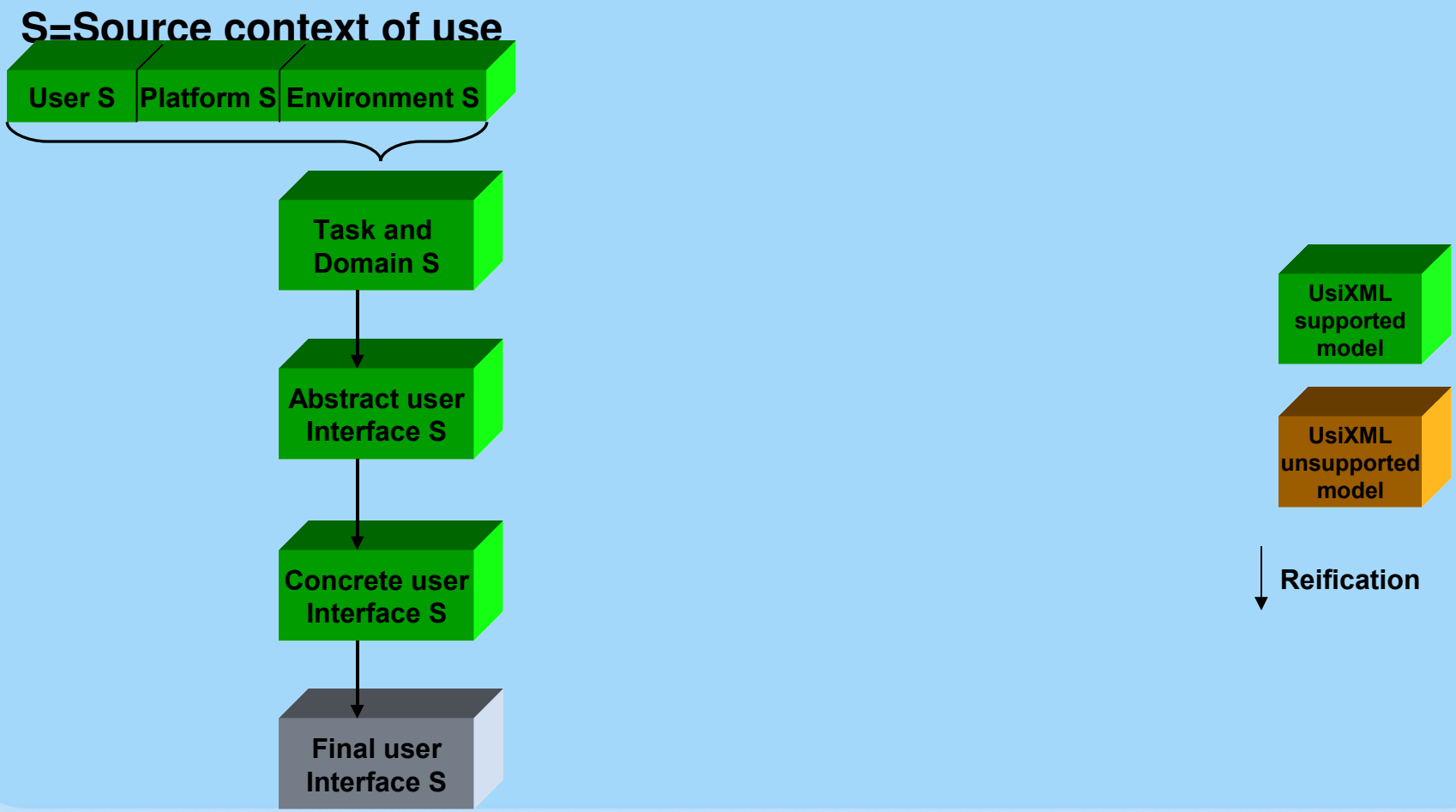




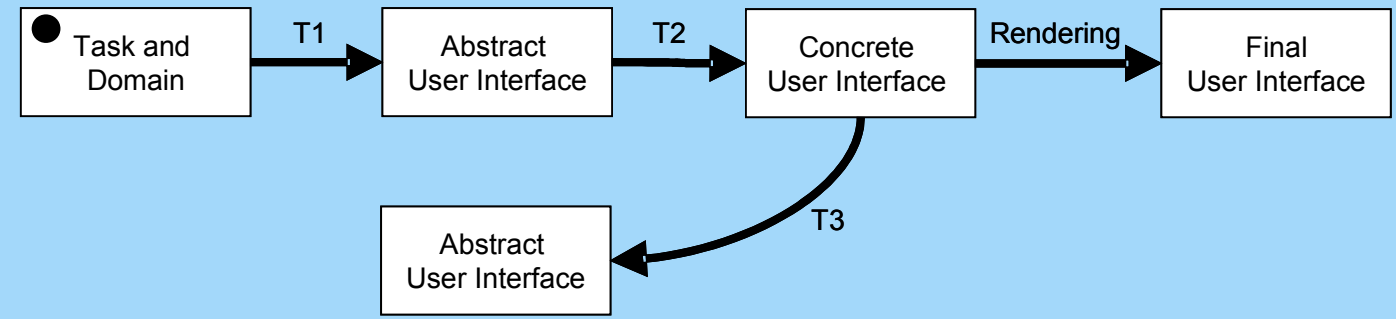
How to read a graph transformation?

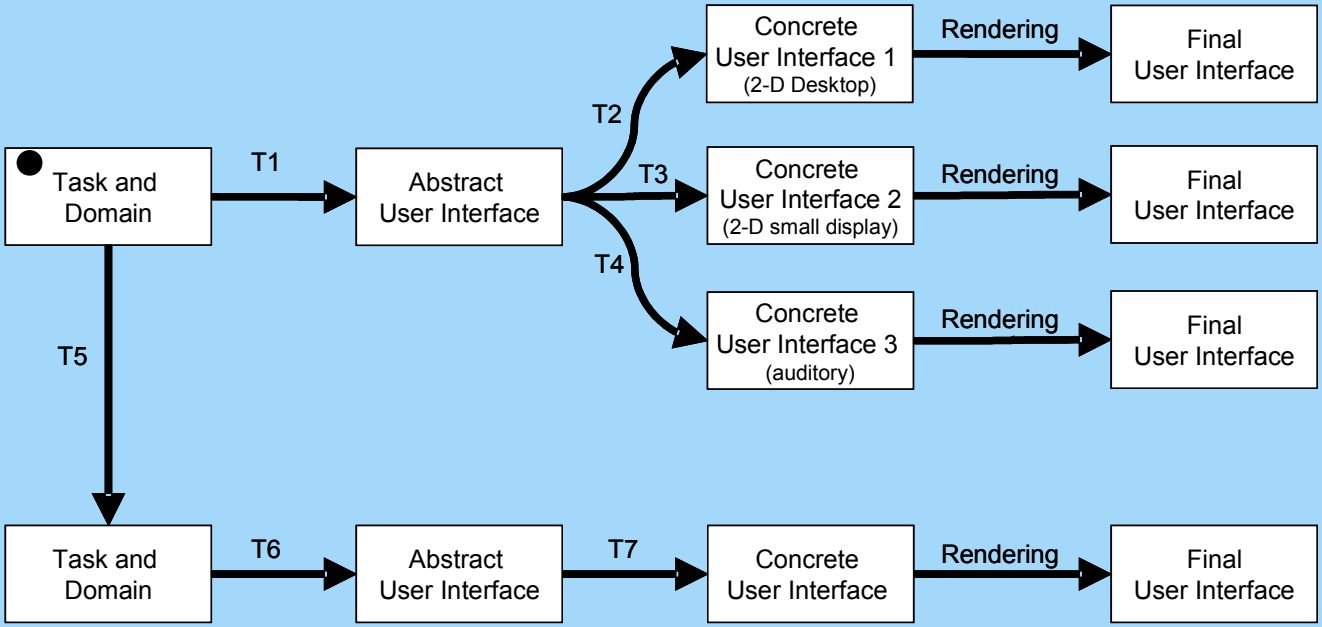


What do we have so far?



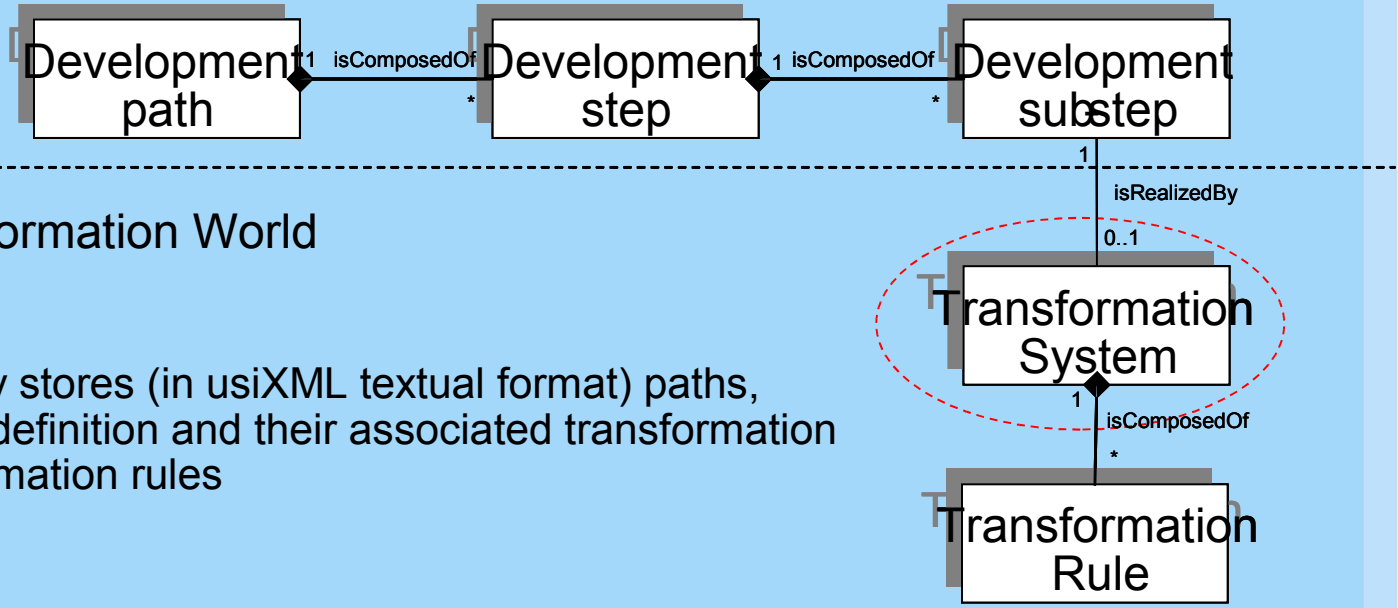
Multiple development paths





Mapping the Methodological World onto the Transformation World

Methodological World

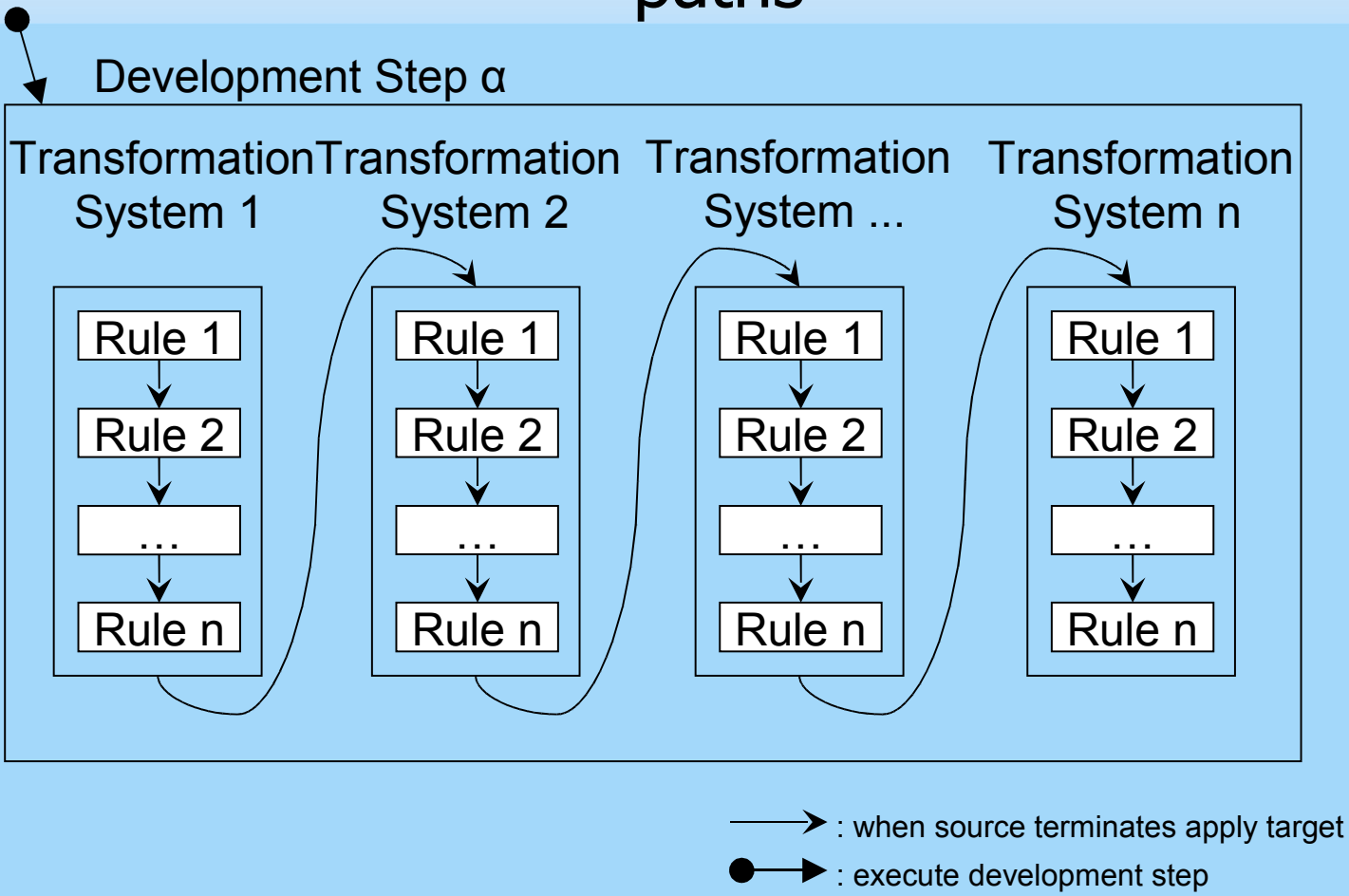


Transformation World

A development library stores (in usiXML textual format) paths, steps and sub-steps definition and their associated transformation systems and transformation rules

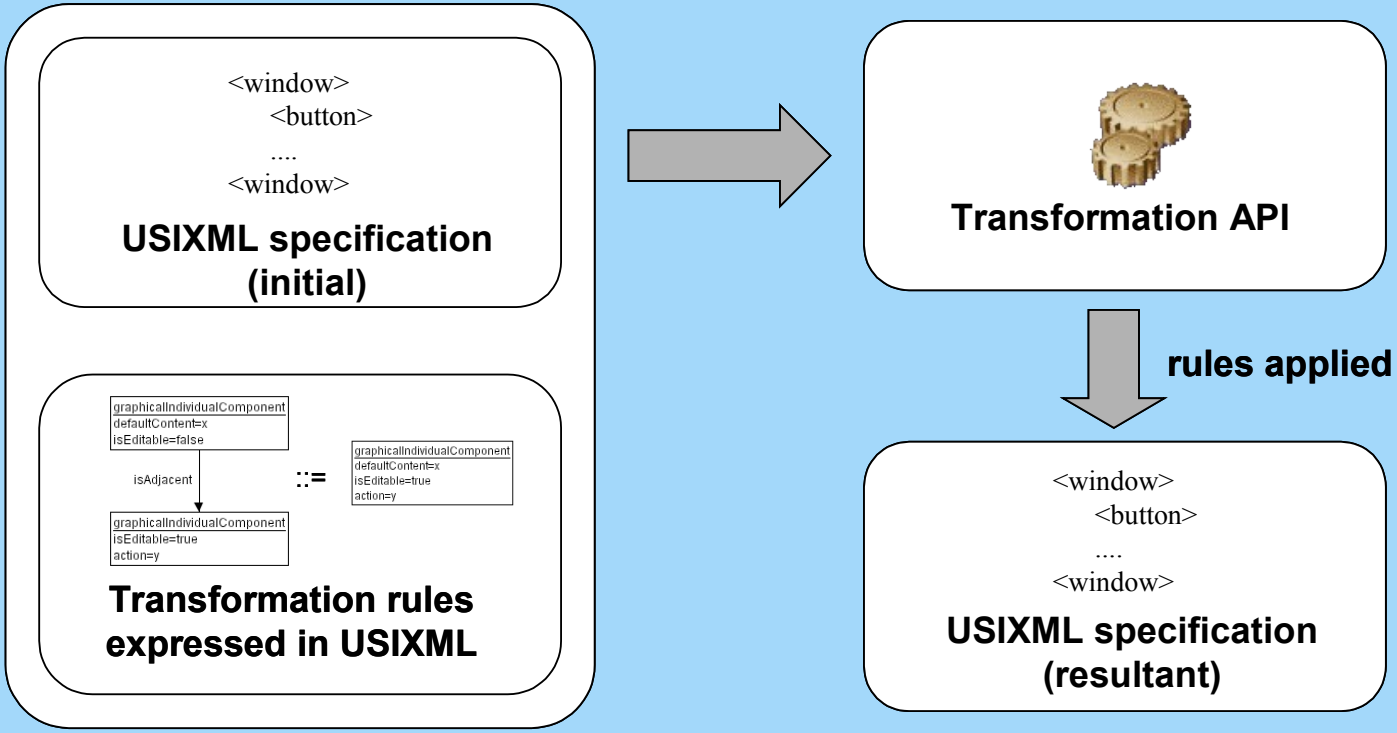
[Limbourg,2004]

Multiple development paths



TransformiXML API/GUI

- API = set of transformations



From T&D to AUI

- TransformiXML

TransformiXML

TransformiXML

Choose File

gs\Adrian Stanciulescu\Desktop\icmi.usi | Add file

Select development path

Starting point: aui | Destination point: cui | Shortcut

Development path explorer

DP_1
Step From AUI to CUI
SubStep Reification of AC i
SubStep Selection of CIC
SubStep Definition of Navig
SubStep Arrangement of CI
SubStep Concrete Dialog C
SubStep Derivation of CUI t

Attach System | Detach System

New Sub-Step | Transform Step by Step | Transform

Transformation system explorer

Transformation system 1

New System | Edit in AGG | Edit in GrafiXML

Transformation rule explorer

Rule 1: Creation of a text component
Rule 2: Creation of a group of radio buttons
Rule 3: Creation of a label
Rule 4: Creation of a button

Save | Back to GrafiXML

[Bouillon et al.,2005]

Final user interface

- Two forms of UI rendering
  - Interpretation
    - By run-time static analysis and direct rendering (InterpiXML & FormiXML)
  - Code generation
    - By program synthesis (GrafiXML)
    - By generative programming (Angie)
      - Feature model
      - Components assembling