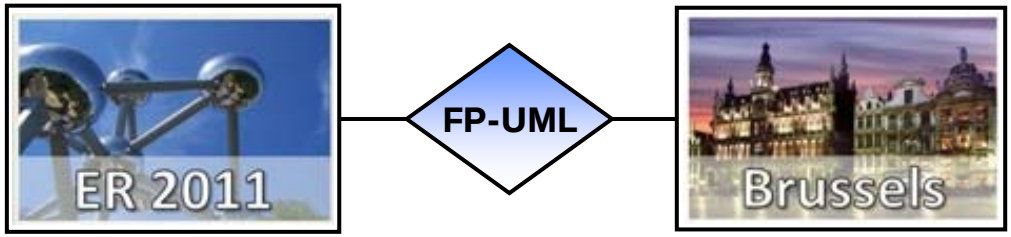


30th International Conference on Conceptual Modeling

●●●

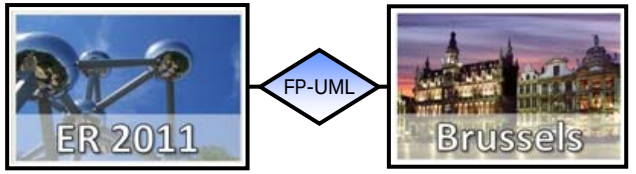
7th International Workshop on
Foundations and Practices of UML (FP-UML)

On Automated Generation of Associations in Conceptual Database Model

Drazen Brdjanin, Slavko Maric

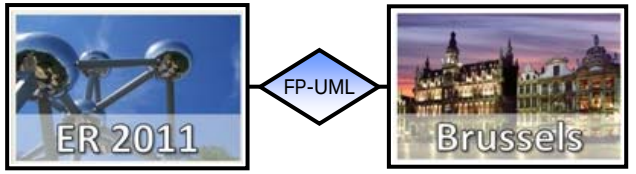
University of Banja Luka, Bosnia and Herzegovina

The paper & presentation



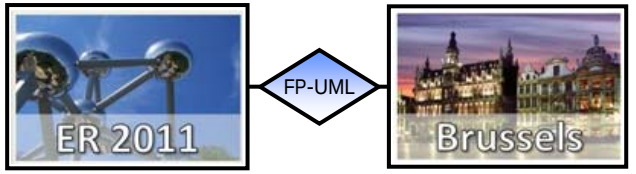
- **Context of the paper: Automated conceptual database model (CDM) design based on business process model (BPM)**
- **Analysis of the semantic capacity of object flows and action nodes in UML AD (BPM) for automated generation of object-object associations in UML CD (CDM)**
- **Formal rules (set-based mappings) for automated generation of object-object associations based on business process activities with input and output objects**
- **Implementation details**
- **Latest improvements**
- **Experimental results**
- **Conclusion**

Motivation

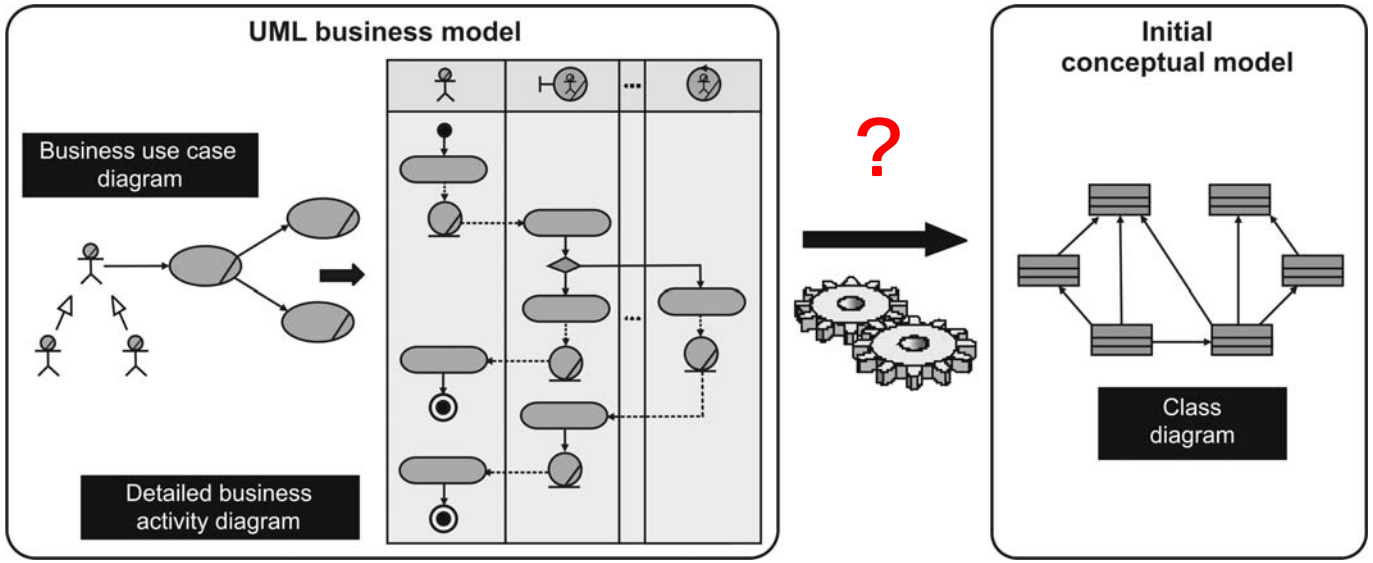


- **Business modeling = first phase of IS development**
  - BPM is used for the **identification of system requirements**, i.e. architecture of an IS that best supports given business system
  - **BPM can be used for automated generation of target software models** - very foundation of MDSD, a paradigm that sees the BPM (CIM) as the basis for automated PIM design
- **Business modeling and database design use different notations that usually don't conform to the same/common metamodel**
  - Business modeling – mainly characterized by process-oriented notations: IDEF0, EPC, Petri nets, BPMN, ...
  - Database design – ER (IE, IDEF1X, …) traditionally used for conceptual modeling + increasing use of the UML CD
- **Notational disharmony**
  - problem and challenge in automated CDM design based on BPM
  - one of the reasons for a fairly small number of papers in the field
- **Solution: use of UML for both BM and DB design**

Motivation (cont.)



Automated CDM (UML CD) design based on BPM (UML AD)

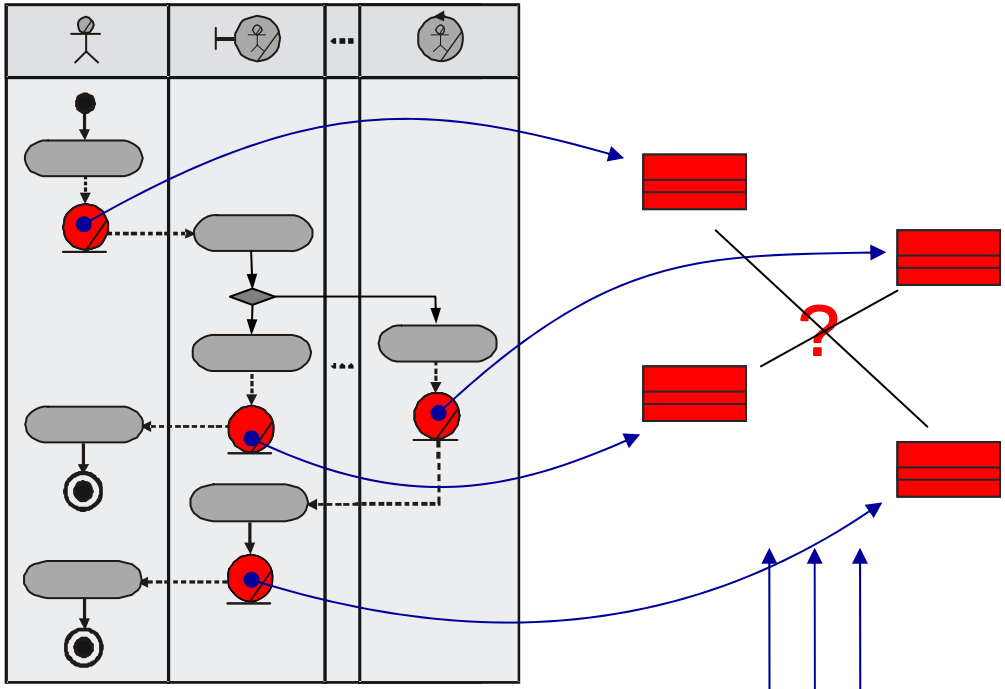


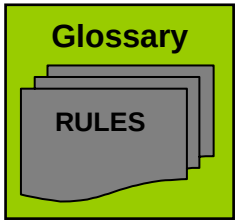
Semantic capacity of AD for automated CDM design is not completely identified yet!

Several proposals for CDM based on AD, but:

- **There are no formal transformation rules!**
- Only few implemented automated CDM generators but with modest achievements
  - **extraction of participants and business objects**
  - **some participant-object associations**

Previous proposals (1/3)





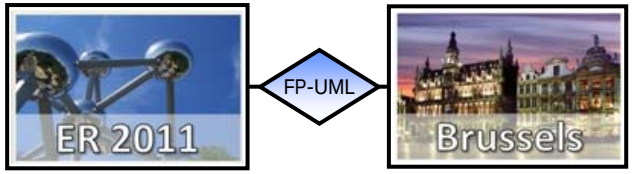
NO implementation

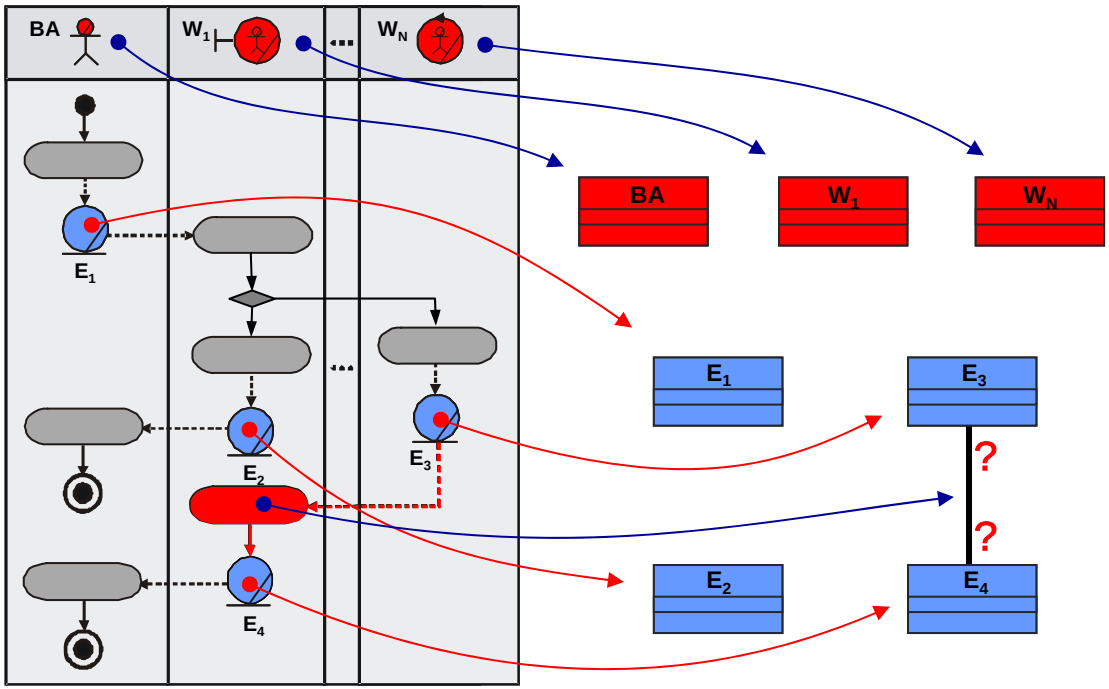
- **Garcia Molina et al.** (ER 2000) propose the initial conceptual model based on UML AD and supplementary glossary

- direct mapping of all **information objects** into the respective classes in the target CD

- creation of class associations based on business rules informally specified in the glossary - **NOT suitable basis for automatic generation**

Previous proposals (2/3)





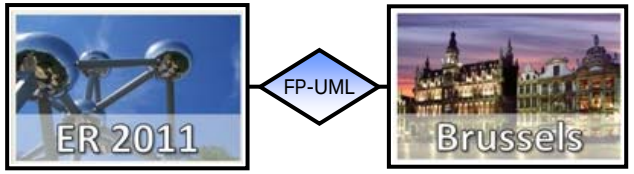
- **Rodriguez et al.** (FP-UML 2008) propose the direct mapping of all **information objects** and all **business process participants** into the respective classes in the target CD

QVT implementation

- **Suarez et al.** (WCE 2008) propose automatic creation of **associations for activities that have input and output objects**, as direct mapping of those activities to the respective associations between corresponding classes

- **Limitations related to automated generation of multiplicities**, since they don't propose any explicit rule

Previous proposals (3/3)

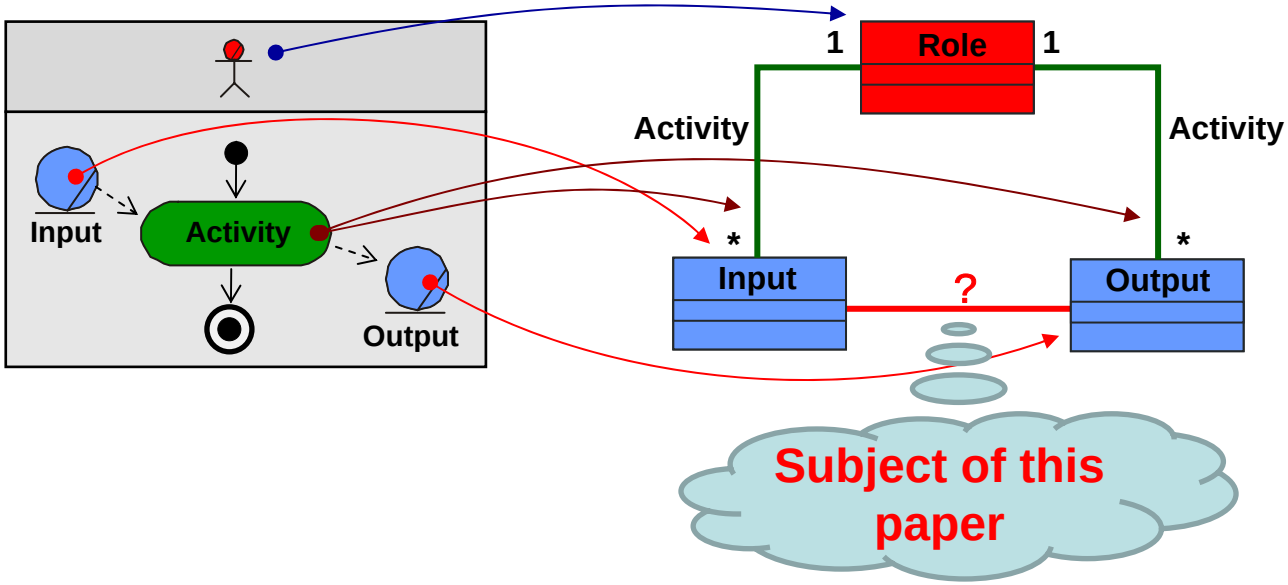


Brdjanin et al. (ADBIS 2010) **ADBdesign generator**

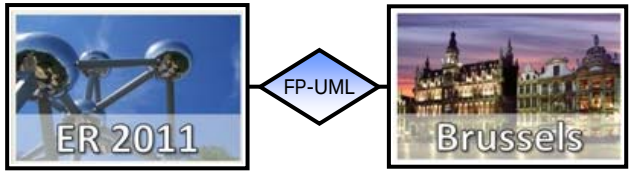
❶ **direct mapping of all business objects** to the respective classes

❷ **direct mapping of all business process participants** to the respective classes

❸ **creation of associations between business objects and business process participants** based on activities performed on those objects



Object-Object associations



Suarez et al. (WCE 2008)

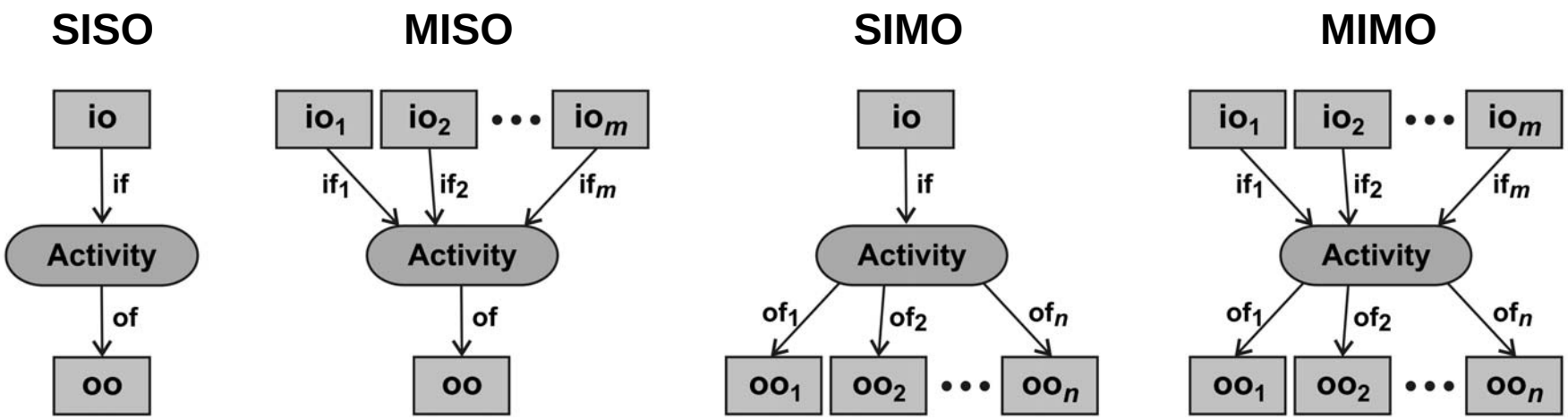
Proposal for automatic creation of **associations for activities that have input and output objects**, as direct mapping of those activities to the respective associations between corresponding classes, but **without any explicit rule for automated generation of association end multiplicities**

•••

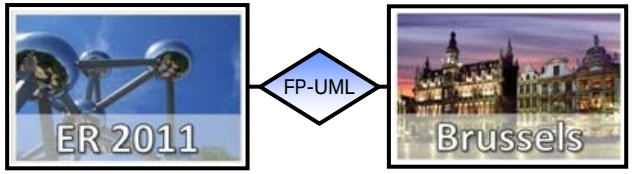
Our proposal

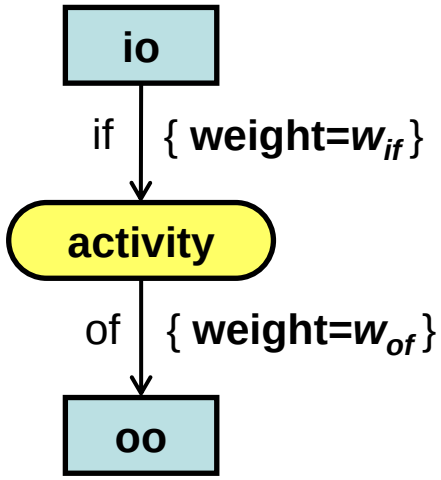
Direct mapping of such activity into the set of binary associations

Classification of activities regarding the **number of different type of IOs and OOs**



Prerequisite





Main prerequisite: Alignment of object flow weights

weight – minimum number of tokens that must traverse the edge at the same time (UML specification)

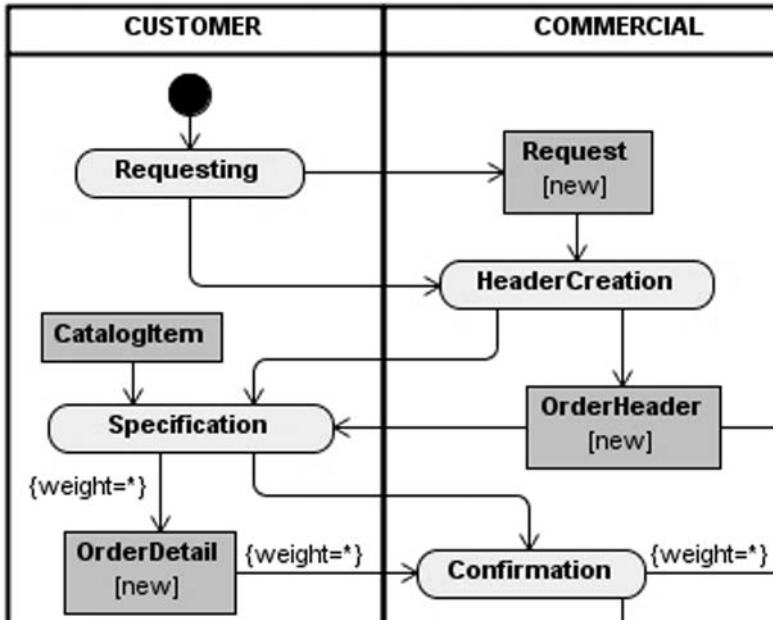
●●●

constant weight – not minimum, but exact number of objects

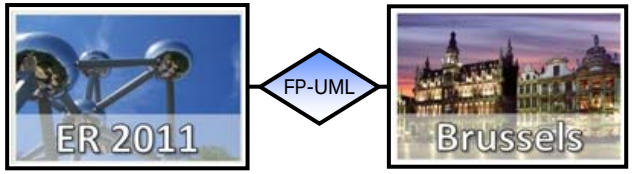
w_{if} – number of input objects required for activity

w_{of} – number of output objects created in activity

unlimited weight (*) – if number of objects is not constant



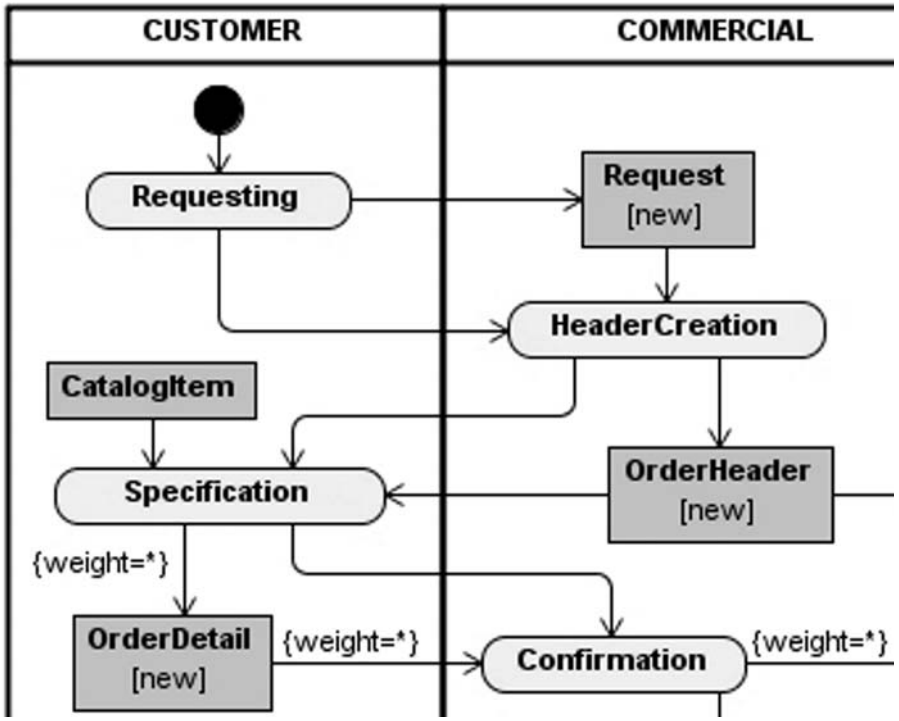
Existing/Generated IOs



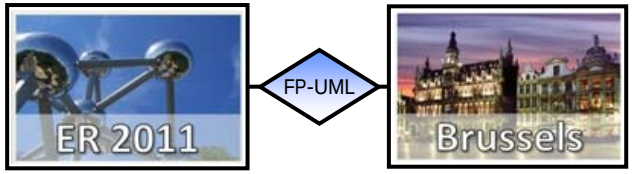
EXISTING/GENERATED input objects distinction

Existing input object
Input object that is not created in the given business process
(CatalogItem)

Generated input object
Input object that is created in the given business process
(Request, OrderHeader, OrderDetail)

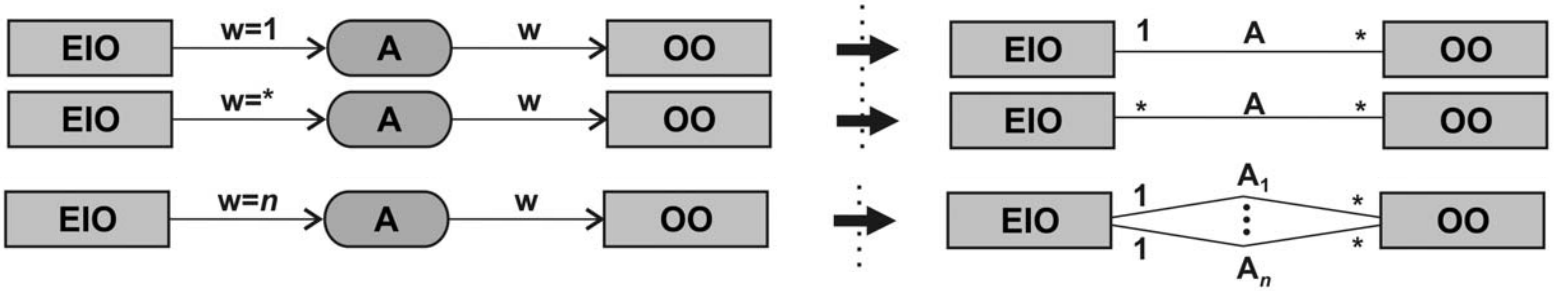


SISO activities (1/3)

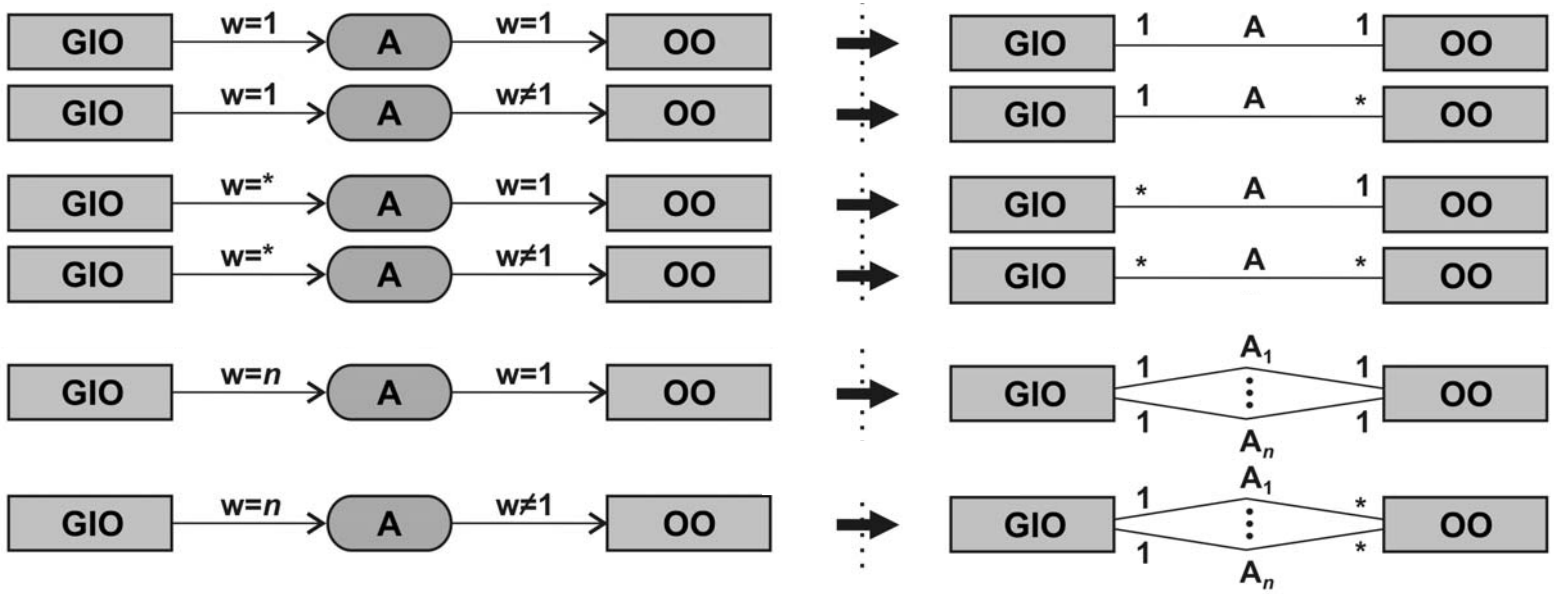


SISO = Single Input / Single Output

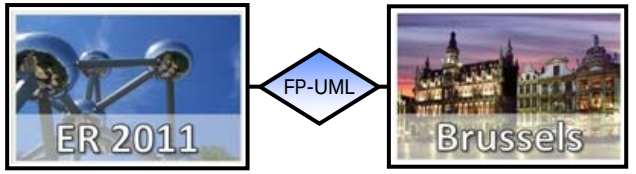
SISO cases for existing IOs and corresponding mappings



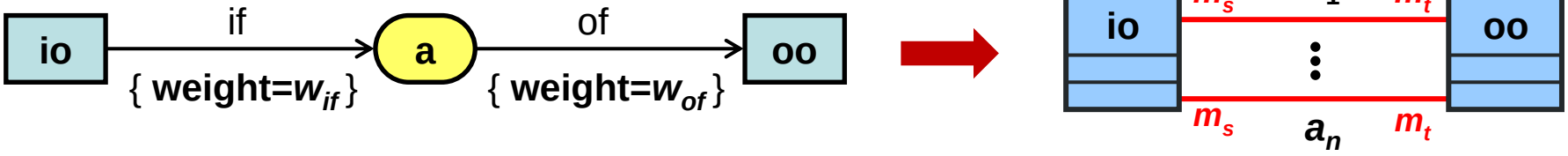
SISO cases for generated IOs and corresponding mappings



SISO activities (2/3)



Formal rule for mapping SISO tuple into the set of binary associations



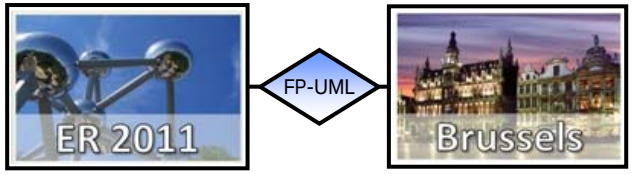
$$\mathcal{T}^{siso}_{\mathcal{OO}} : \mathcal{DAD}(\mathcal{P},\mathcal{A},\mathcal{O},\mathcal{F}) \mapsto \mathcal{CM}(\mathcal{E},\mathcal{R}) \overset{def}{\Longleftrightarrow} \mathcal{R}^{siso}_{\mathcal{OO}}(a) = \mathcal{T}^{siso}_{\mathcal{OO}}\big(\langle io, if, a, of, oo\rangle\big)$$

$$\mathcal{R}^{siso}_{\mathcal{OO}}(a) = \left\{ r^{(j)}_{\mathcal{OO}} \in \mathcal{R} \;\middle|\; r^{(j)}_{\mathcal{OO}} = \mathcal{T}^{*}_{\mathcal{OO}}\big(\langle io, if, a, of, oo\rangle\big), j = 1, ..., n \right\}$$

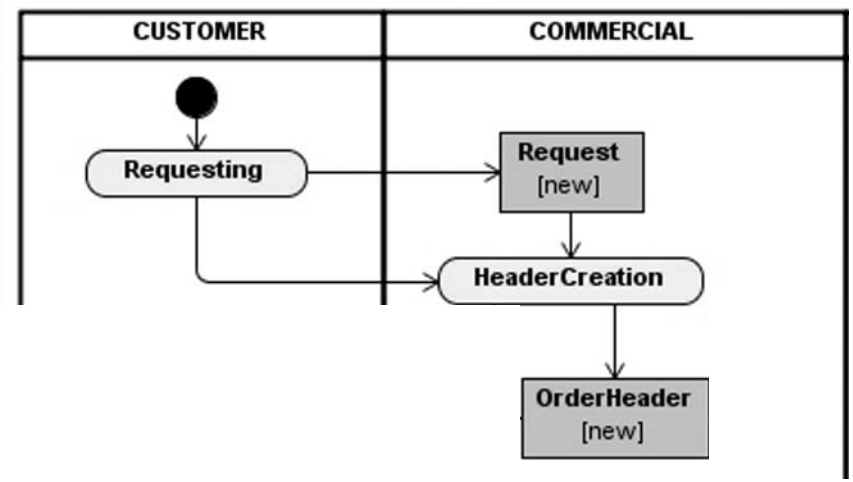
$$\mathcal{T}^{*}_{\mathcal{OO}}\big(\langle io, if, a, of, oo\rangle\big) \overset{def}{=} r_{\mathcal{OO}} \;\middle|\; \Big(name(r_{\mathcal{OO}}) = name(a) \wedge \big(memberEnd(r_{\mathcal{OO}}) = \{source, target\} \;\middle|\; type(source) = e_{\mathcal{IO}} \wedge multiplicity(source) = m_s \wedge type(target) = e_{\mathcal{OO}} \wedge multiplicity(target) = m_t\big)\Big),$$

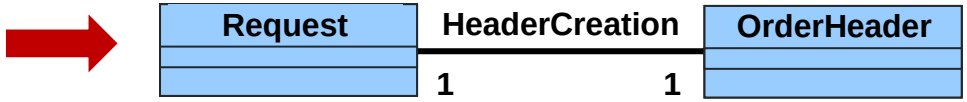
$$m_s = \begin{cases} *, w_{if} = * \\ 1, otherwise \end{cases} \quad m_t = \begin{cases} *, w_{of} \neq 1 \vee io \in \mathcal{O}_{\mathcal{X}} \\ 1, otherwise \end{cases} \quad n = \begin{cases} 1, w_{if} \in \{1, *\} \\ w_{if}, otherwise \end{cases}$$

SISO activities (3/3)

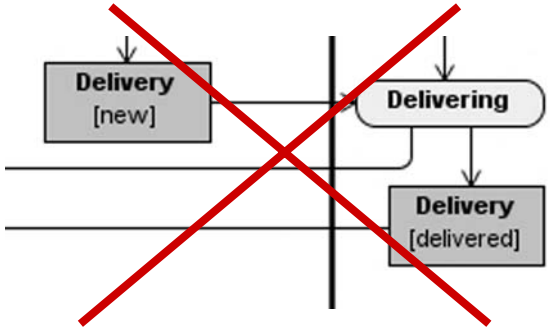


Example 1: *Application of SISO mapping rule*



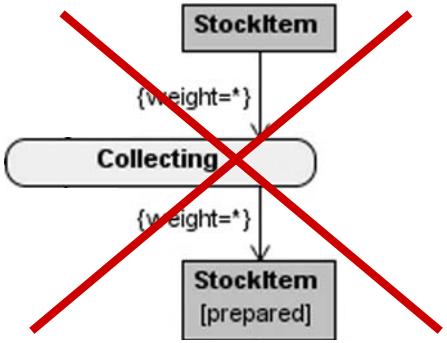


Example 2:



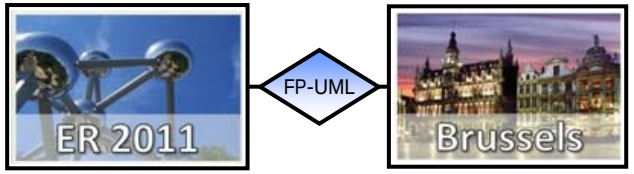
Activity changing GIO state

Example 3:



Activity changing EIO state (activation)

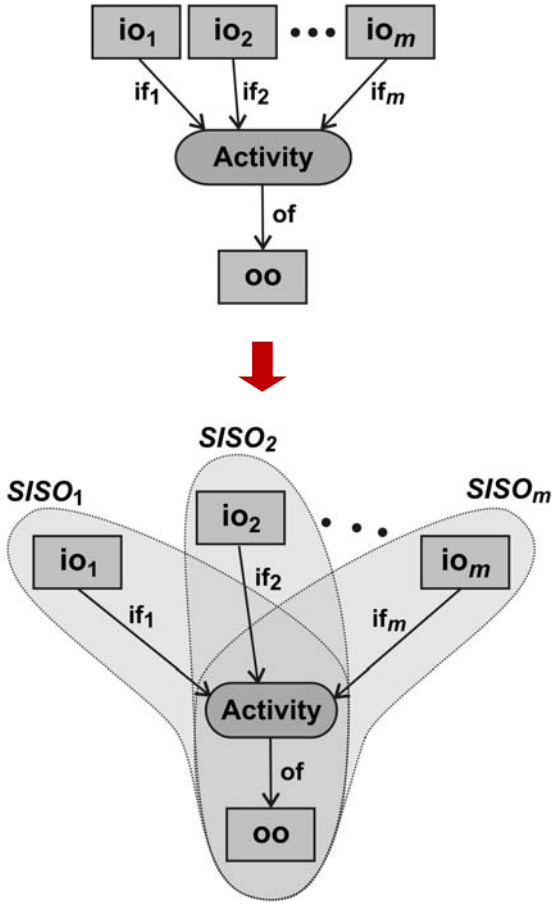
MISO and SIMO activities



MISO = Multi Input / Single Output

Input objects of $m \in N$ different types

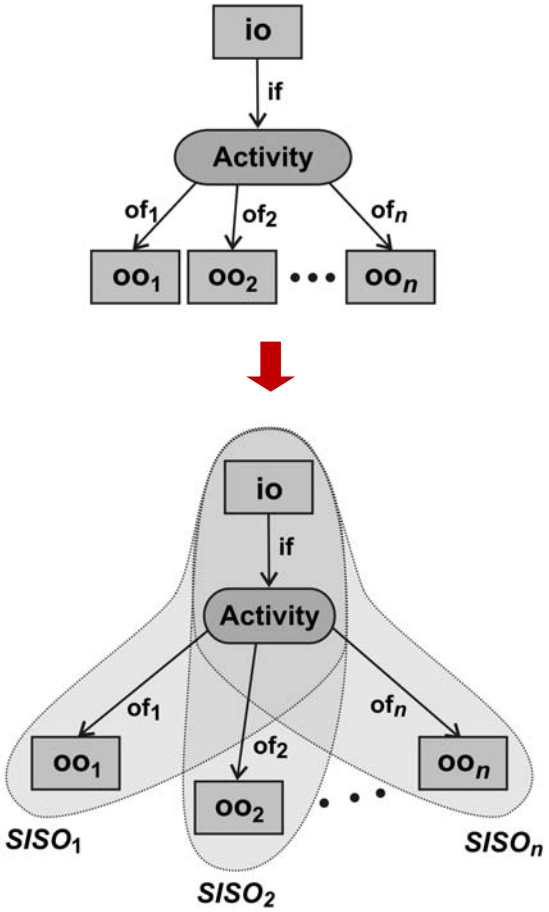
Output objects of exactly one type



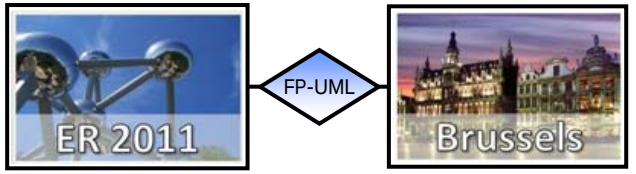
SIMO = Single Input / Multi Output

Input objects of one type

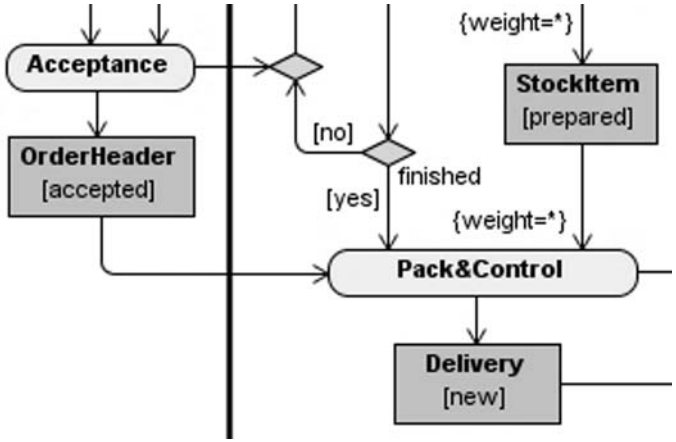
Output objects of $n \in N$ different types

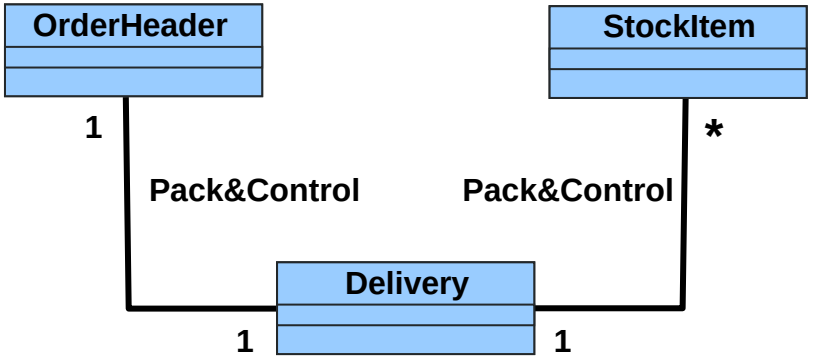


MISO and SIMO activities

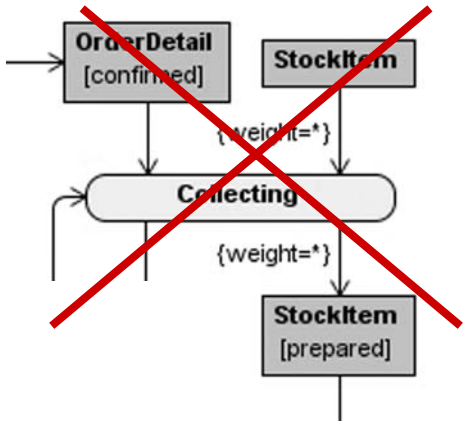


Example 1: *Application of MISO mapping rule*



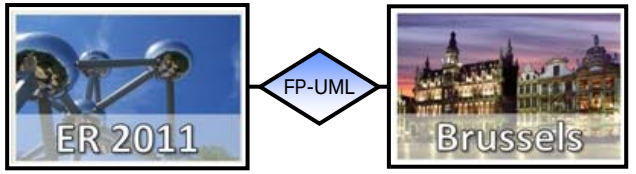


Example 2:



Not MISO activity

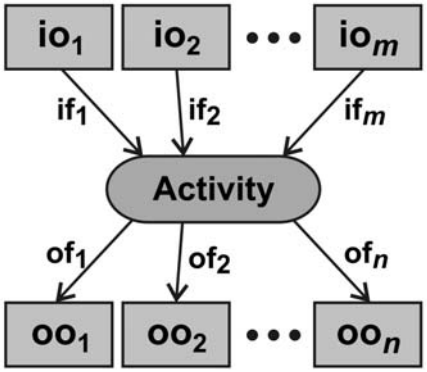
MIMO activities



MIMO = Multi Input / Multi Output

Input objects of $m \in N$ different types

Output objects of $n \in N$ different types



$$\mathcal{R}_{\mathcal{OO}}^{mimo}(a) \stackrel{def}{=} \bigcup_{\substack{1 \le j \le m \\ 1 \le k \le n}} \mathcal{R}_{\mathcal{OO}}^{(j,k)}(a),$$

$$\mathcal{R}_{\mathcal{OO}}^{(j,k)}(a) = \mathcal{T}_{\mathcal{OO}}^{siso}\left(\langle io_j, if_j, a, of_k, oo_k \rangle\right).$$

Total number of associations

m x n

Special cases

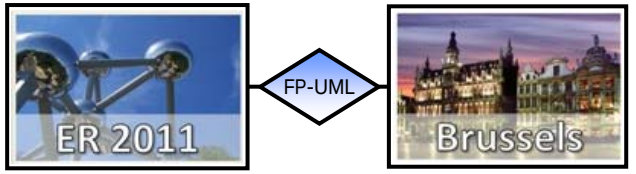
$m=1, n \neq 1 \Rightarrow$ SIMO

$m \neq 1, n=1 \Rightarrow$ MISO

$m=1, n=1 \Rightarrow$ SISO

Each activity can be considered as MIMO activity. Each MIMO activity can be considered as a set of concurrent SISO activities i.e. Basic SISO rule is to be applied to each SISO tuple.

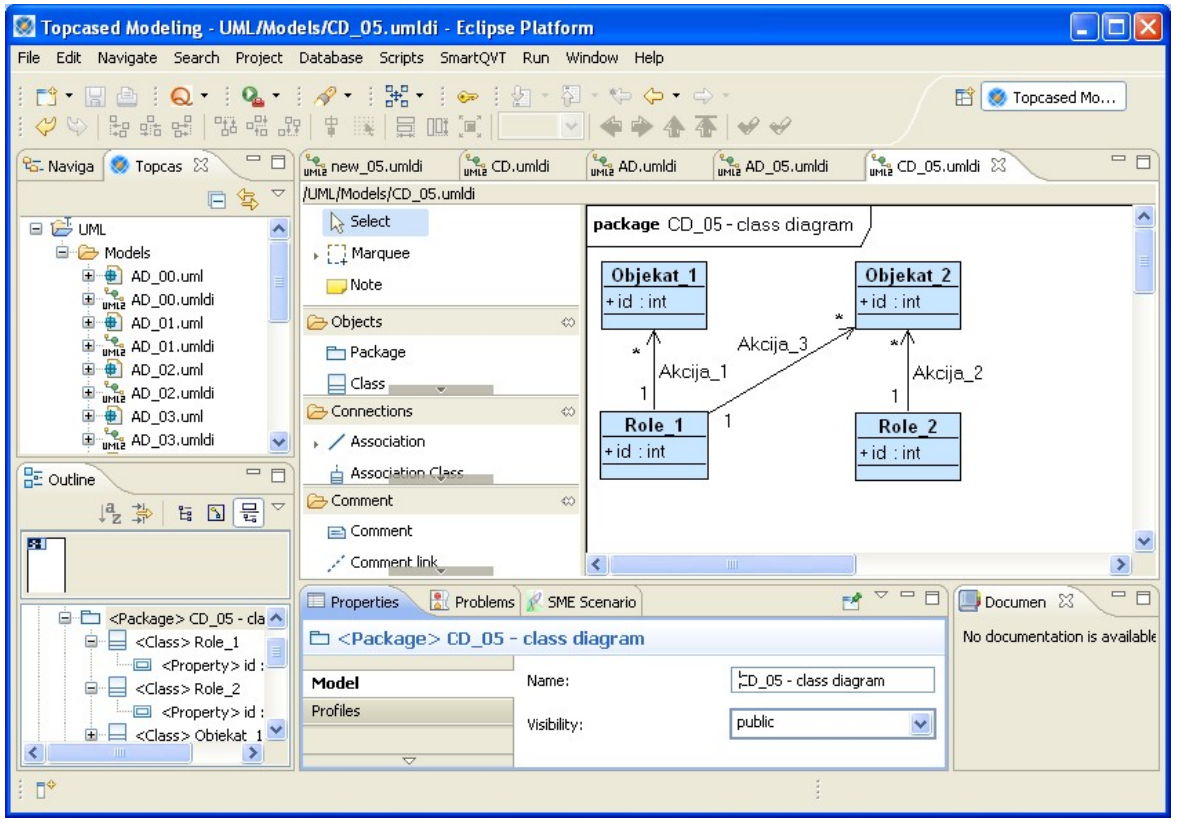
Implementation



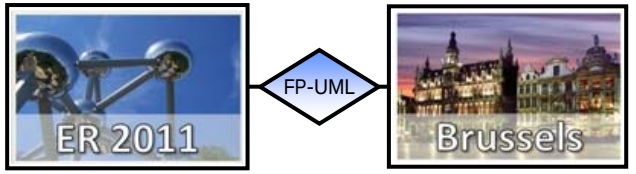
Topcased development platform

(Toolkit in OPen-source for Critical Application & SystEms Development)

- **Topcased v3.2.0** is a system/software engineering toolset based on **Galileo (Eclipse 3.5)**
- It includes **several graphical model editors** (**UML**, SysML, SAM, AADL), QVT and **ATL processor**s, some code generators (UML2Java, UML2Python, etc), and some other experimental features
- Topcased UML toolkit enables the manipulation of most UML diagrams and **satisfactory visualization of automatically generated diagrams**
- **ATL-based implementation of automated CDM generator**



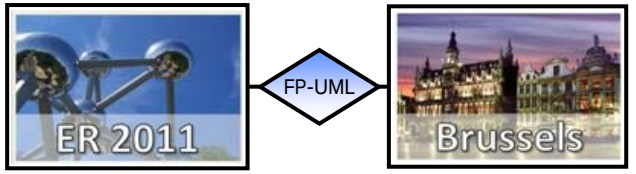
ATL-based CDM generator



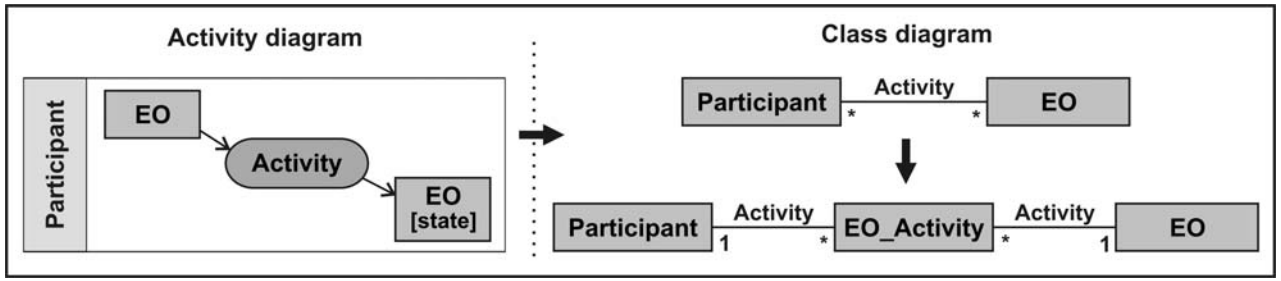
Listing 1. Implementation of transformation rule $\mathcal{T}_{OO}^{siso}$

```
helper def : gC (s:Sequence(Integer)) : Sequence(Integer) =
   if (s.last()>1) then thisModule.gC((s.append(s.last()-1))) else s endif;
rule SISO (io:uml!Element, iof:uml!Element, ac:uml!Element,
                           oof:uml!Element, oo:uml!Element)
{
  using { ct:Sequence(Integer) = thisModule.gC(Sequence {iof.weight.value}); }
  do { for (c in ct) { thisModule.TOO(io,iof,ac,oof,oo); }}
}
rule TOO (io:uml!Element, iof:uml!Element, ac:uml!Element,
                          oof:uml!Element, oo:uml!Element)
{ to
   d : uml!Association ( name<-ac.name, ownedEnd<-Sequence{os,od} ),
   os : uml!Property ( name<-'source', type<-thisModule.resolveTemp(io,'d'),
          upperValue<-us, lowerValue<-ls ),
   od : uml!Property ( name<-'target', type<-thisModule.resolveTemp(oo,'d'),
          upperValue<-ut, lowerValue<-lt ),
   us : uml!LiteralUnlimitedNatural
          (value<-if iof.weight.value=-1 then '-1'.toInteger() else 1 endif),
   ls : uml!LiteralInteger (value<-if iof.weight.value=-1 then 0 else 1 endif),
   ut : uml!LiteralUnlimitedNatural
          (value<- if (oof.weight.value<>1 or io.incoming.isEmpty())
                   then '-1'.toInteger() else 1 endif),
   lt : uml!LiteralInteger
          (value<- if (oof.weight.value<>1 or io.incoming.isEmpty())
                   then 0 else 1 endif)
 do { thisModule.cd.packagedElement <- d; }
}
```

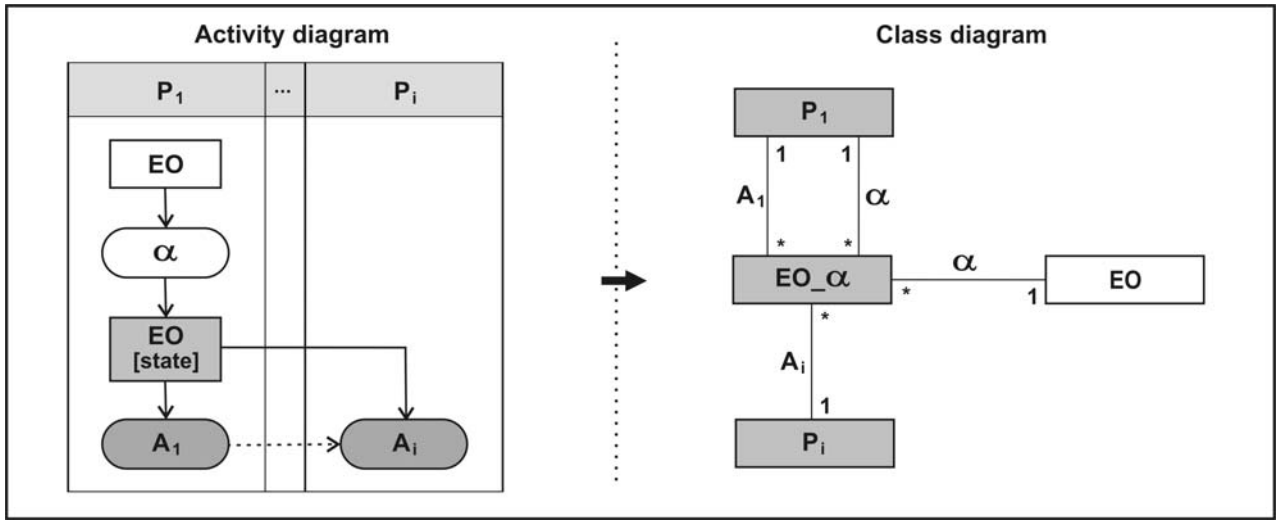
Latest improvements



Activation of existing objects



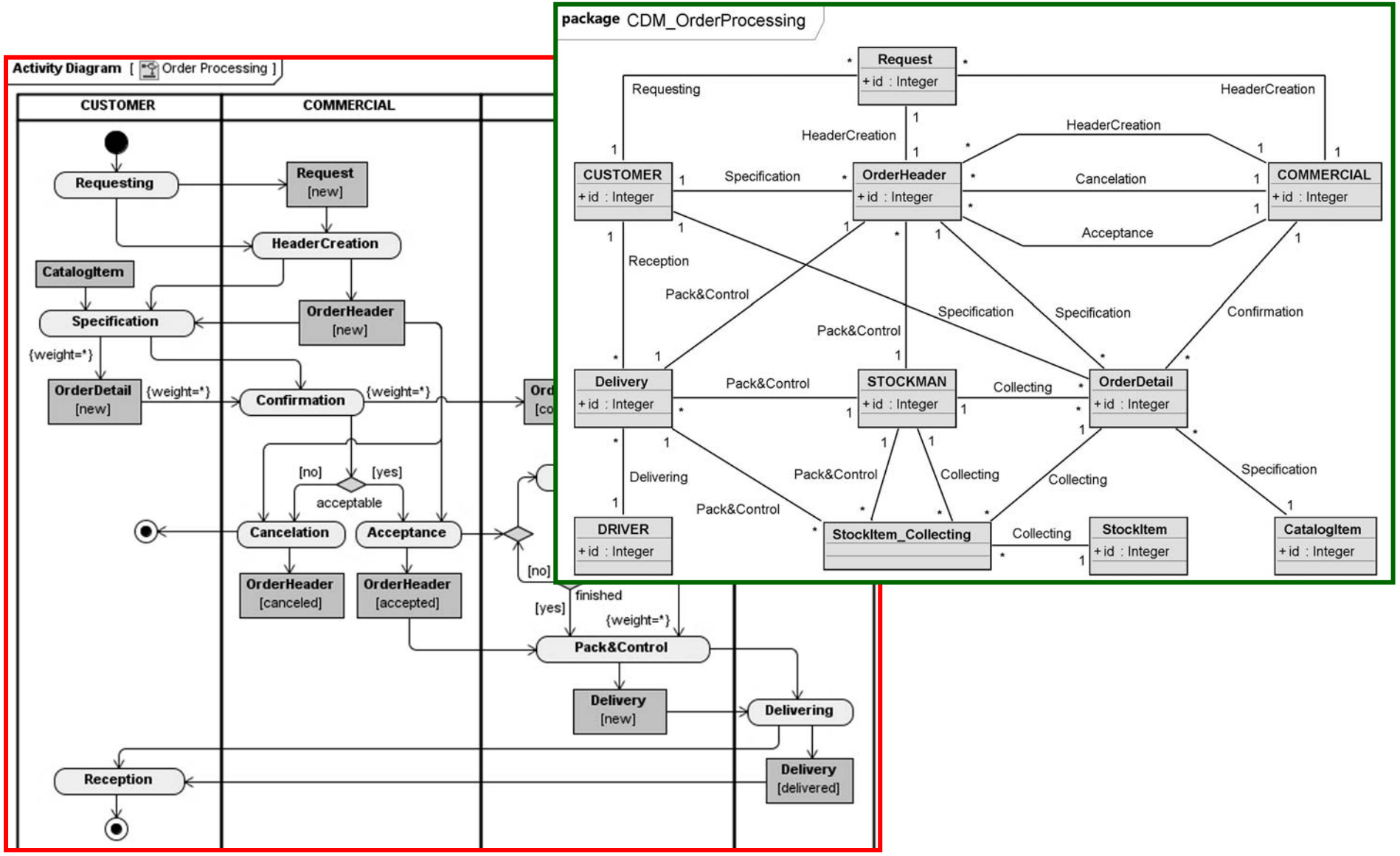
Usage of activated existing objects



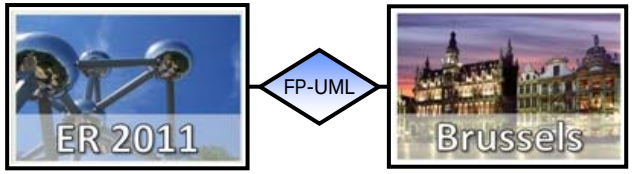
EIO_α = GIO

Improvement of ADBdesign generator

Experimental results (1/2)



Experimental results (2/2)



Quantitative evaluation

Recall

percentage of all concepts in the target CDM that is automatically generated
(percentage of all classes that is automatically generated and percentage of all associations that is automatically generated)

$$Recall = \frac{N_{correct}}{N_{correct} + N_{missing}} \cdot 100\%$$

Precision

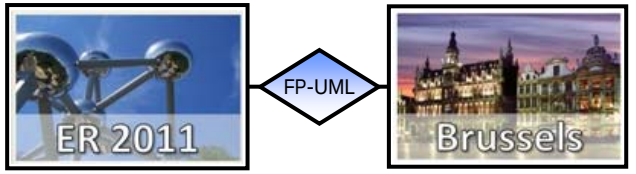
percentage of correctly generated concepts in the automatically generated CDM
(percentage of correctly generated classes and percentage of correctly generated associations)

$$Precision = \frac{N_{correct}}{N_{correct} + N_{incorrect}} \cdot 100\%$$

Results of quantitative analysis based on sample CDM

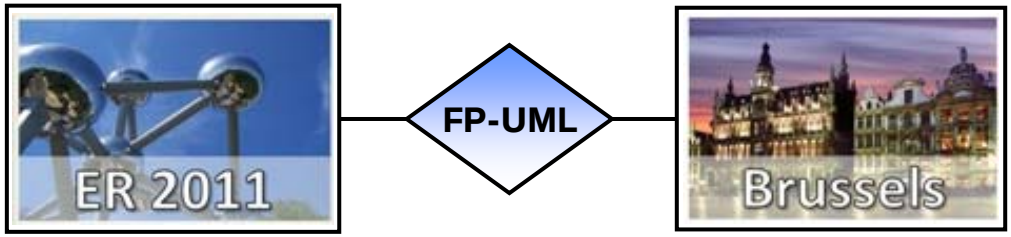
Concepts	Metrics & Measures					
	$N_{generated}$	$N_{correct}$	$N_{incorrect}$	$N_{missing}$	$Recall\ [\%]$	$Precision\ [\%]$
Classes	11	11	0	0	100	100
Associations	22	17+4	1	1	95	95

Conclusion



- In this paper we presented **an approach to automated generation of object-object associations** in the initial CDM (UML CD) based on BPM (UML AD).
- By:
  1) assuming that the **weights of the input and output object flows represent** not minimum, but **exact number of objects required for some activity and exact number of objects created in some activity**, and
  2) introducing the **difference between existing and generated objects,**

  we defined **formal rule for mapping activity having input and output objects into the set of binary associations** of corresponding classes (**basic SISO rule**).
- **Each MIMO activity** (activity having input objects of more than one different type and output objects of more than one different type) can be considered as a **set of concurrent SISO activities** and the basic **SISO rule can be successfully applied to MIMO activities**, as well.
- The **results of the qualitative and quantitative analysis** of the generator's application to the sample model, as well as the preliminary results of application to some other BPMs in different business domains, show that the generator is able to generate a very high percentage of the target CDM (**recall usually exceeds 90%**) and has a very high precision (**over 90% of all automatically generated concepts are usually correct**).



Drazen Brdjanin, Slavko Maric
University of Banja Luka, Bosnia and Herzegovina

On Automated Generation of Associations in Conceptual Database Model

Thank You!