Universidad de los Andes

Composing Graphical Languages

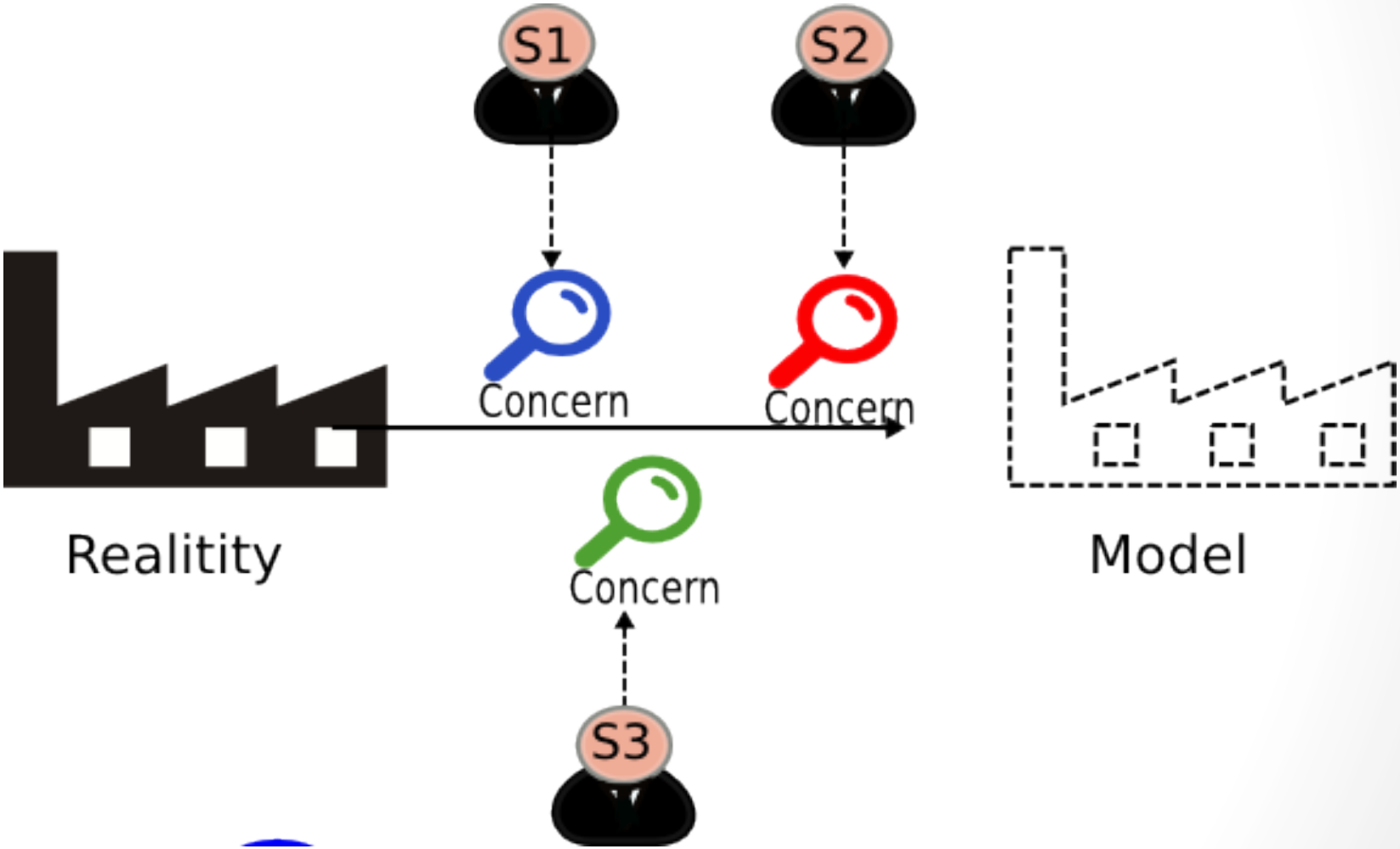
Iván Melo, Mario Sánchez, Jorge Villalobos

{im.melo33, mar-san1,jvillalo}@uniandes.edu.co

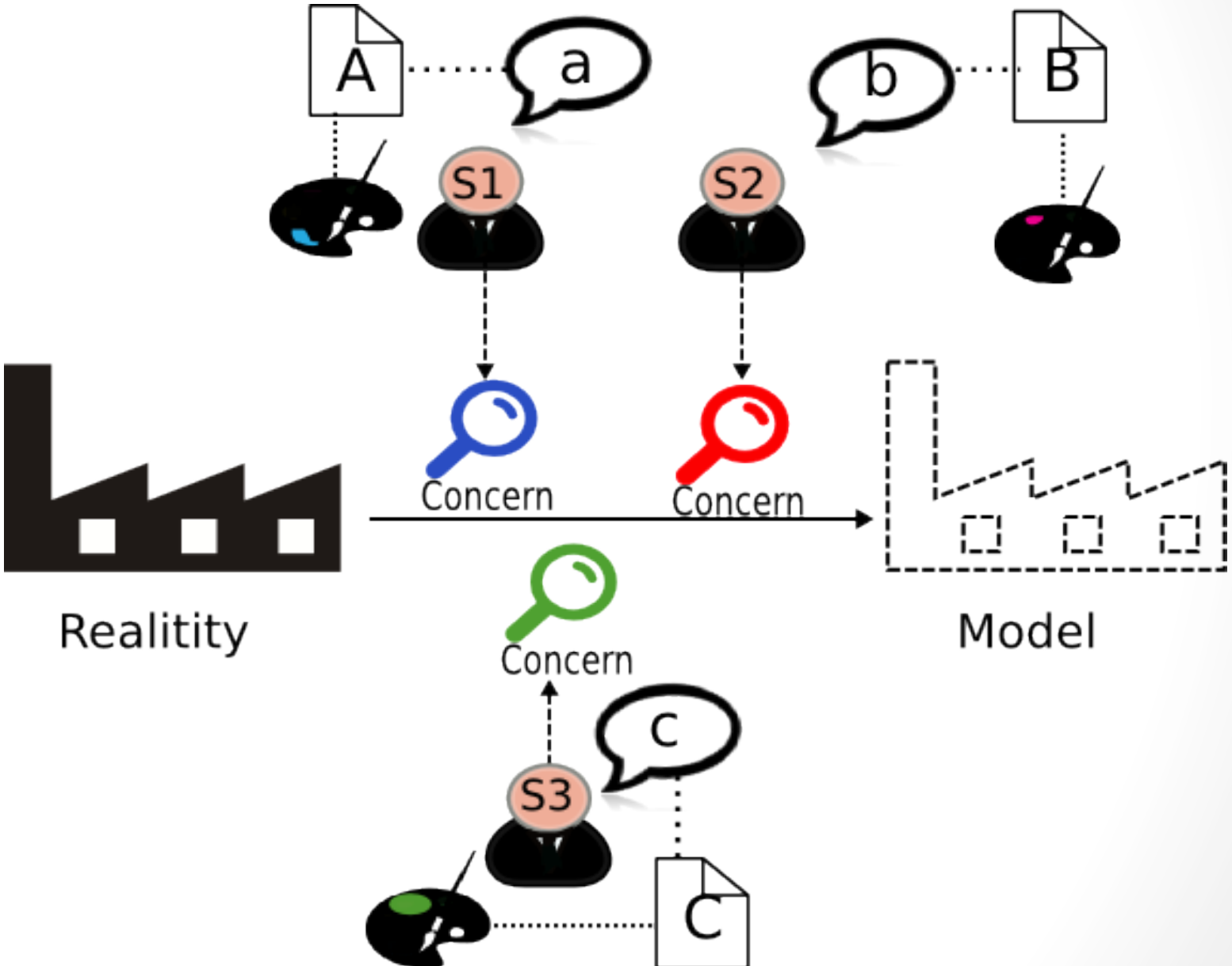
Outline

- Context
- Objectives
- Solution Strategy
  - Picture
  - Fusion
  - Gromp
- Example
- Conclusions

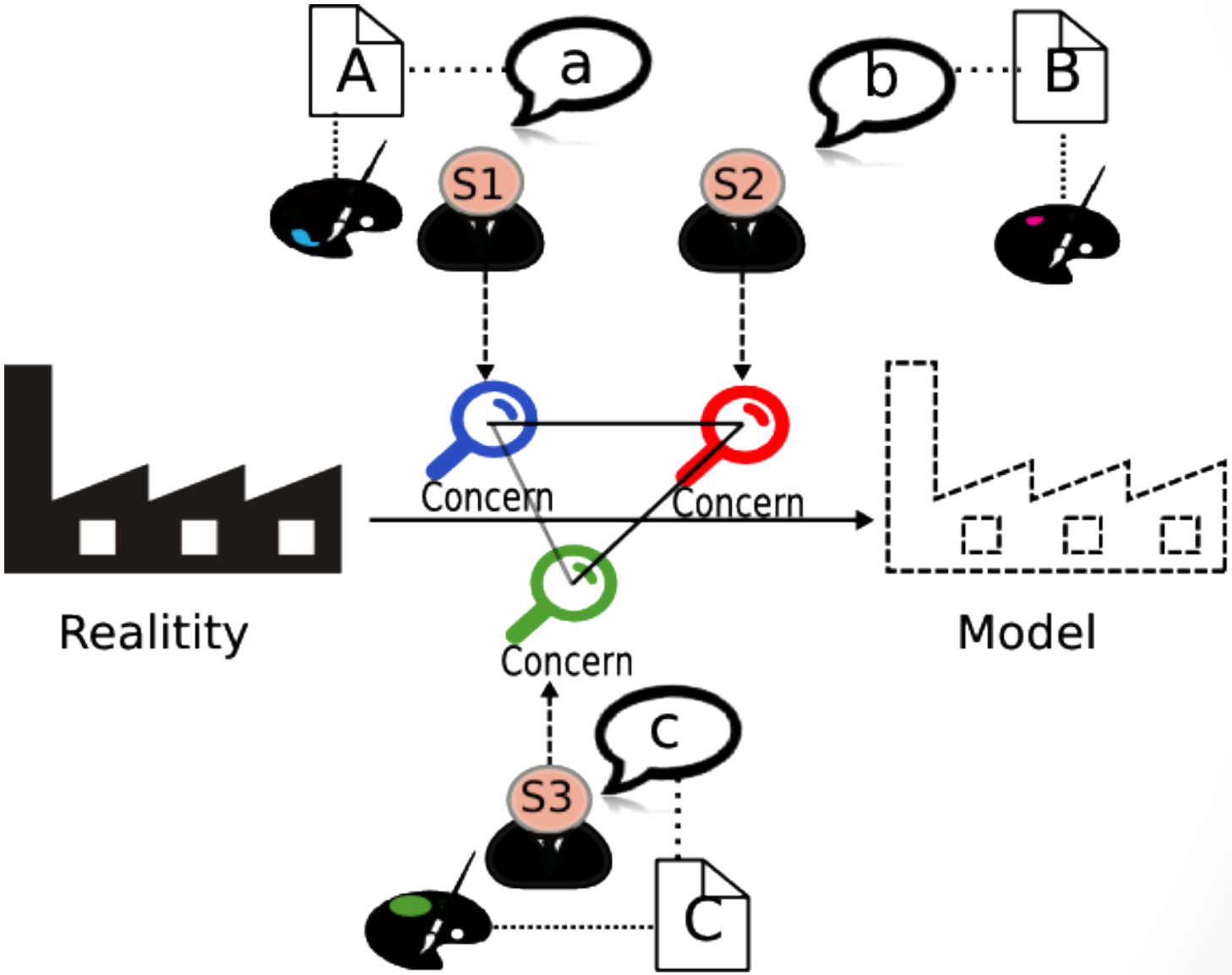
Context



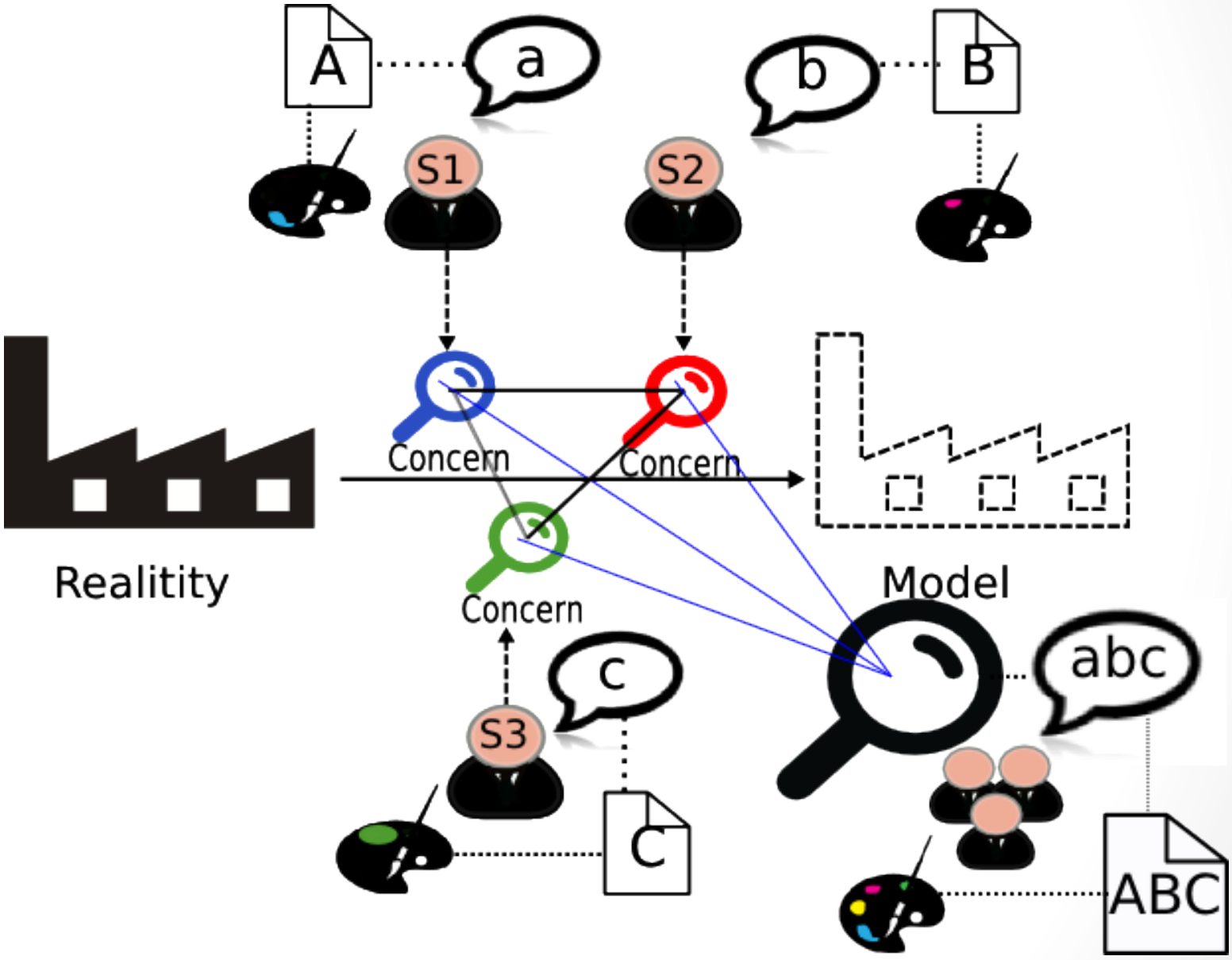
Context



Context



Context


A
a
b
B
S1
S2
Concern
Concern
Concern
Realitity
Model
C
S3
C
abc
ABC

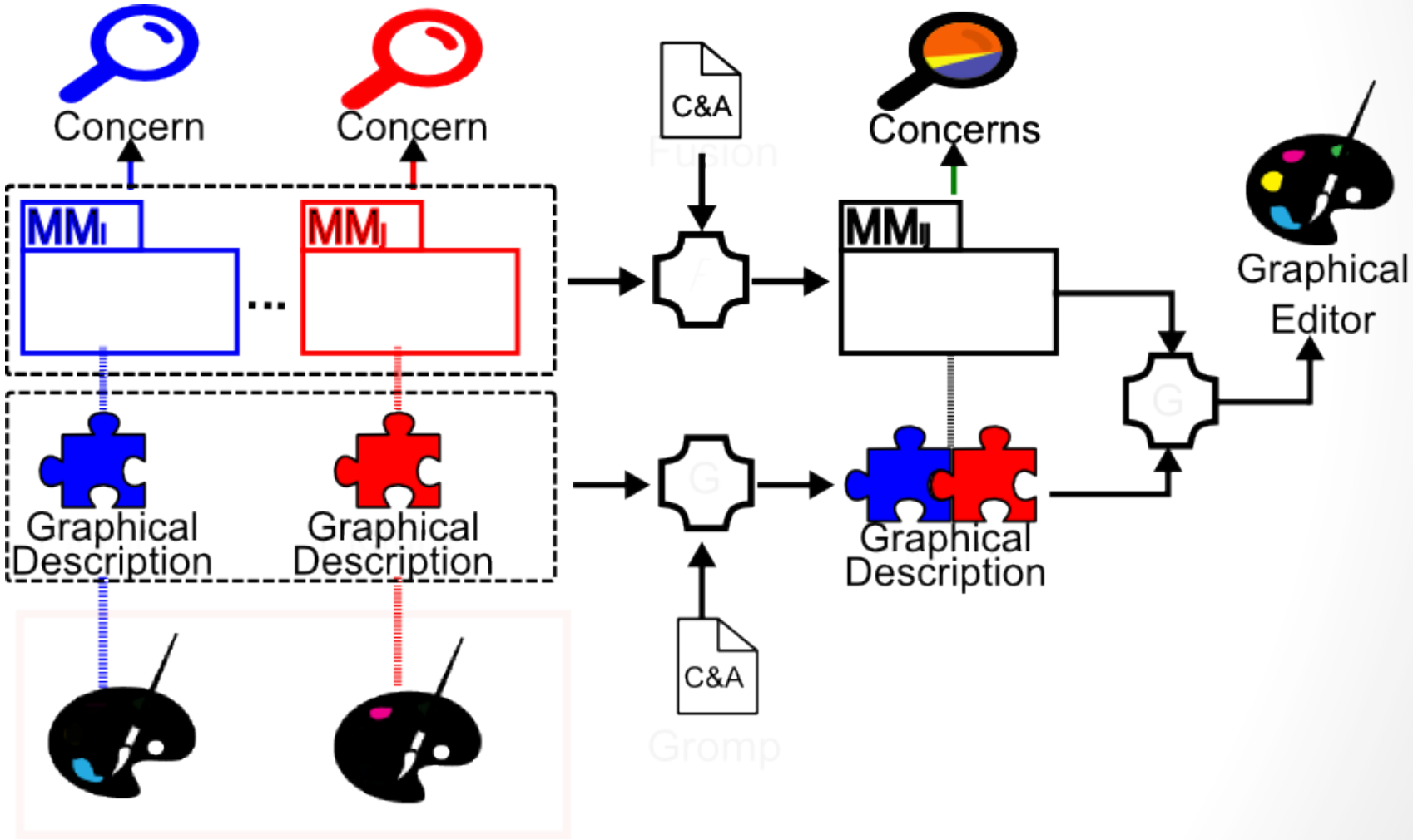
Solution Objectives

- Reduce the time and complexity required to create a graphical editor
- Compose graphical languages and editors in order to facilitate the reuse of existing languages

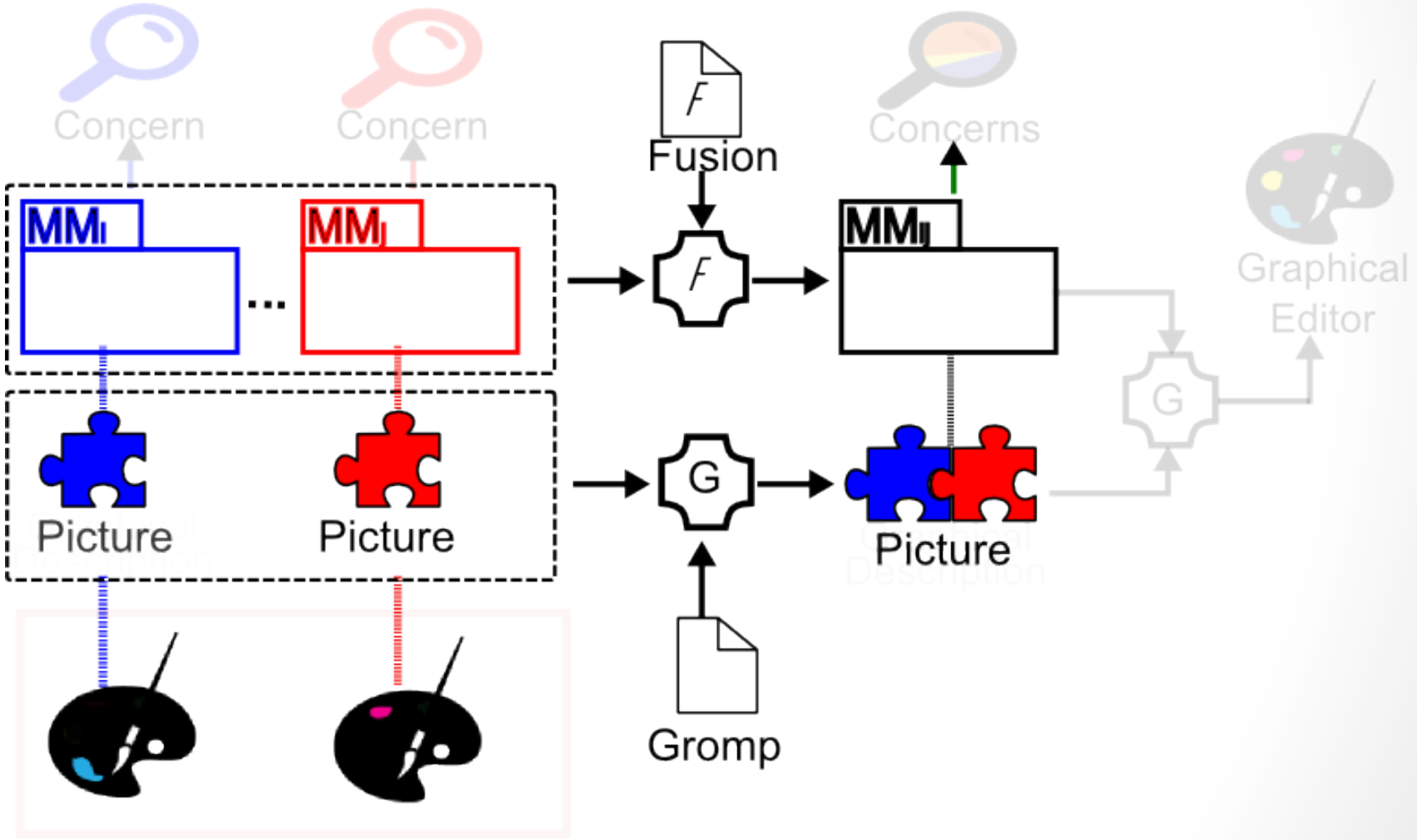
Specific Requirements

1. Use metamodels to describe the abstract syntax of a language
2. Separate the graphical and abstract syntax of a language
3. It should be possible to have several graphical syntaxes for a single language
4. Describe the composition of languages abstract syntaxes
5. Describe the composition of languages graphical syntaxes
6. The composition of the graphical representation should follows the composition of the abstract syntax

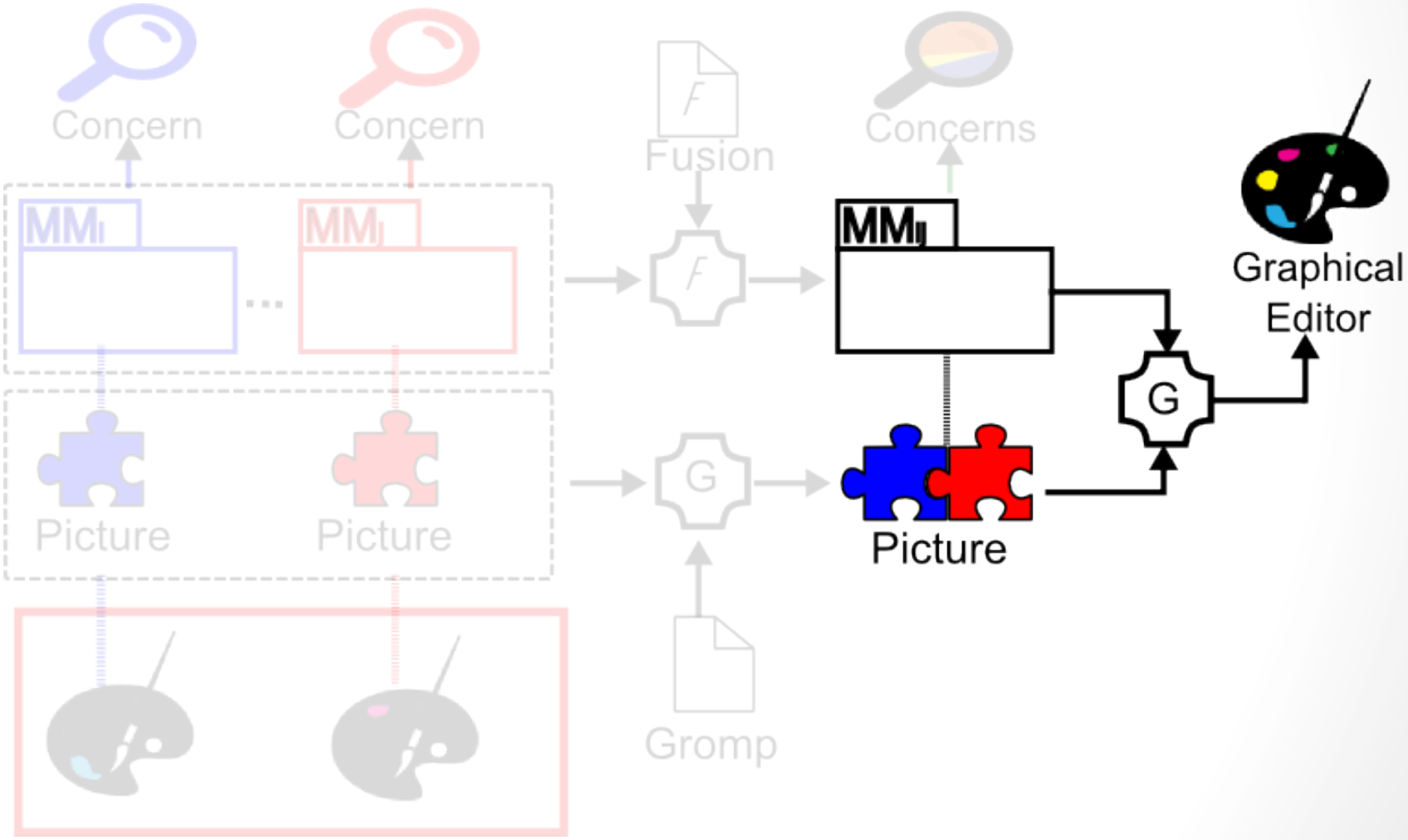
Solution Overview



Solution Overview

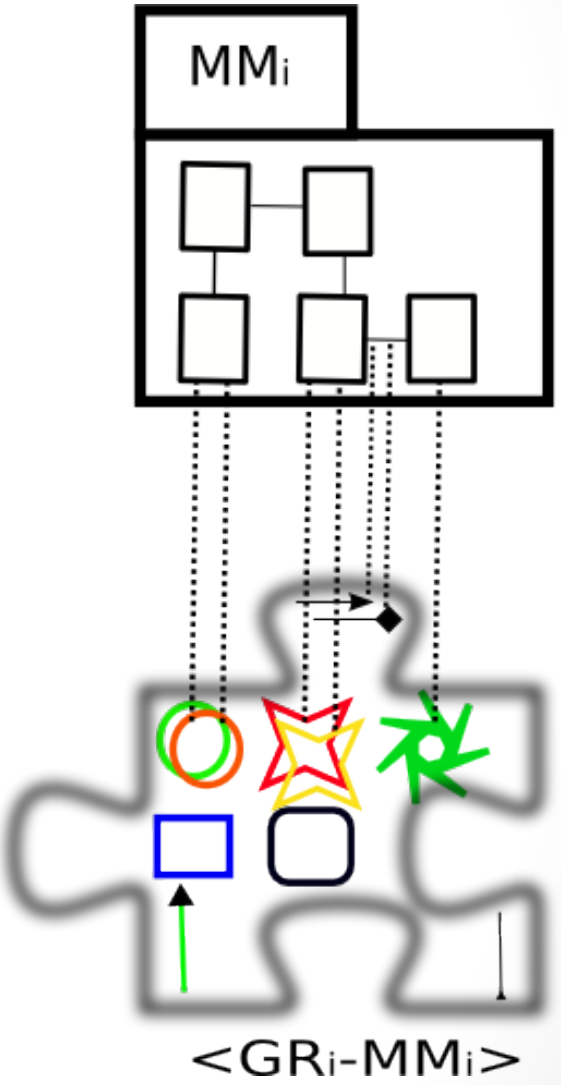


Solution Overview



Picture

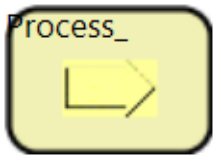
- A Language for describing the graphical syntax of a metamodel
- Describe one or several graphical representations for each element of a metamodel
- 5 different concerns:
  1. Graphical elements
  2. Rules
  3. Tools
  4. Interaction
  5. Style

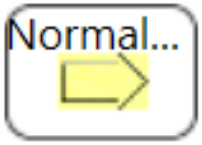


1.Graphical Definition

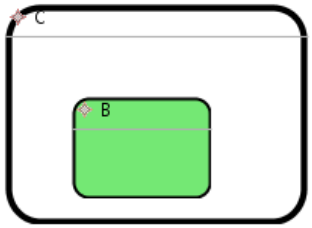
- At least one graphical representation for each meta entity and relation of the language

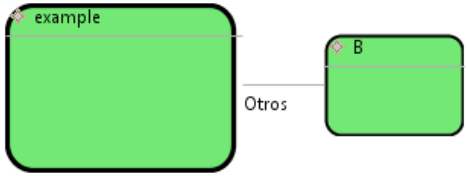
```
Node_element BusinessProcessDef for class BusinessProcess {
    label elementName
    label icon false
    label placement internal
    size (70, 50)
    Regular figure extends rectRounded10 {
        background color behaviorElementsColor
        border BorderBlack2
        icon path "Process.gif"
        icon size ( 33 , 28)
        icon position Point (20, 15)
    }
    phantom true
}
Node_element BusinessProcessWarning extends BusinessProcessDef{
    Regular figure extends rectRounded10 {
        background color yellow
        border BorderBlack2
        icon path "Process.gif"
        icon size ( 33 , 28)
        icon position Point (20, 15)
    }
}
```






```
Internal_node componesDef
for reference A.compone {
    layout default
    layout
}
Node_line otrosDef
for reference A.otros {
    label "Otros"
    style BorderBlack2
    source decoration none
    target decoration none
}
```





2.Rules:

Rules for using alternative graphical representations based on values of attributes or number of relations:

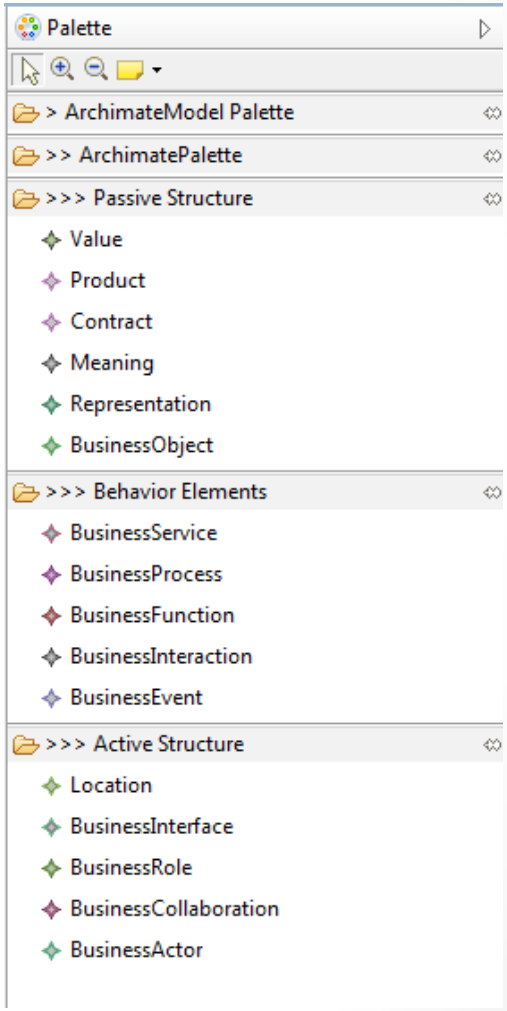
Example:

- Importance >= 5
- 20 >= importance < 60
- Importance > 60

```
Rules for class BusinessProcess{
    case "element.getImportance()<20 &&
          element.getImportance()>=5"
        use BusinessProcessBigWhite
    case "element.getImportance()<60 &&
          element.getImportance()>=20"
        use BusinessProcessWarning
    case "element.getImportance()>60"
        use BusinessProcessBigwarning
}
```

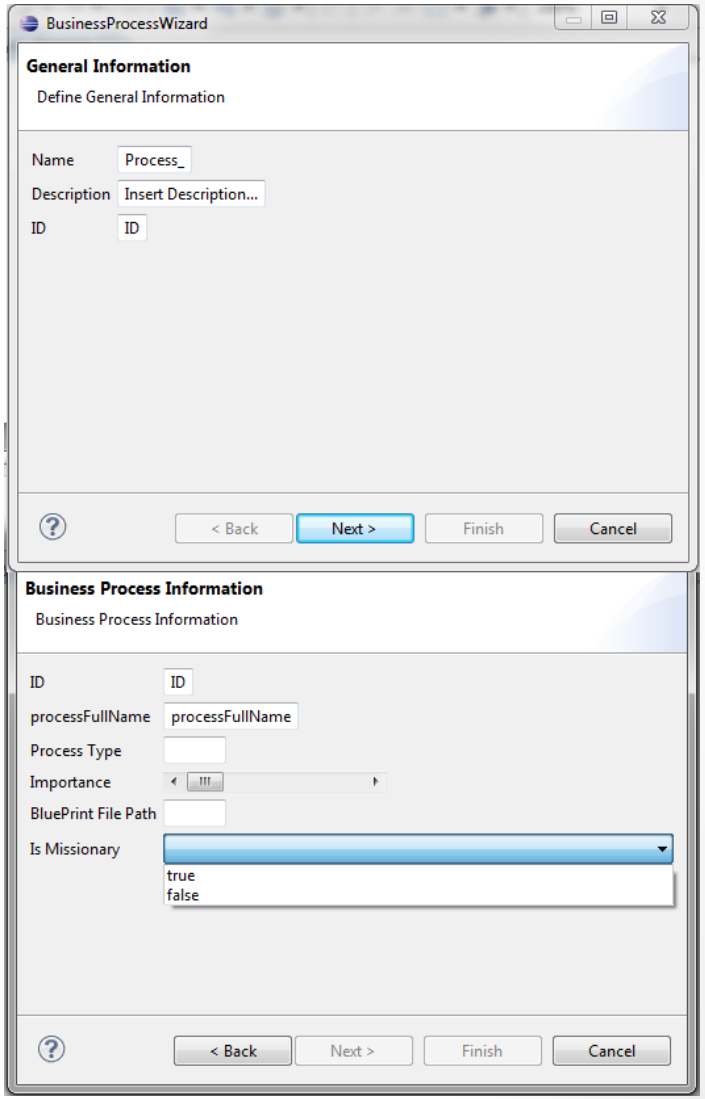
3.Tool Definition

```
Palette for ArchimateModel {

    Tool group ArchimatePalette {
        description "ArchimatePalette"

        Tool group PassiveStructures {
            description "Passive Structure"

            Creation button for class Value {
            name "Value"
            description "Value"
            icon "/imagenes/ValueIcon.gif"
            }
            Creation button for class Product {
                name "Product"
                description "Product"
                icon "/imagenes/ProductIcon.gif"
            }
            Creation button for class Contract {
                name "Contract"
                description "Contract"
                icon "/imagenes/ContractIcon.gif"
            }
                name "Meaning"
            Creation button for class Representation {
                name "Representation"
                description "Representation"
                icon "/imagenes/RepresentationIcon.gif"
            }
            Creation button for class BusinessObject {
                name "BusinessObject"
                description "BusinessObject"
                icon "/imagenes/BusinessObjectIcon.gif"
            }

        }
        Tool group BehaviorElements {

        Tool group ActiveStructures {
    }
}
```



4.User Interaction

```
Wizard WizardForCreateBasicBusinessProcess for context class BusinessProcess {
    title "BusinessProcessWizard"
    description "Create BusinessProcess Wizard"
    type create
    pages {
        Page {
            title "General Information"
            description "Define General Information"
            Attributes to show {
                ("elementName", "Name", textField, "Process_")
                ("description", "Description", textArea, "Insert Description...")
                ("processID", "ID", textField, "ID")
            }
            References to show {

            }
            Additionals buttons {

            }
        }
        Page {
            title "Business Process Information"
            description "Business Process Information"
            Attributes to show {
                ("processID", "ID", textField, "ID")
                ("processFullName", "processFullName", textField, "processFullName")
                ("processType", "Process Type", textField, "            ")
                ("importance", "Importance", Slider, "1,1,50")
                ("processDesign", "BluePrint File Path", textField, "            ")
                ("missionary", "Is Missionary", comboBox, "true,false")
            }
            References to show {

            }
            Additionals buttons {

            }
        }

    }
    default buttons true
}
```



Specific Requirements

1.Use metamodels to describe the abstract syntax of a language

2.Separate the graphical and abstract syntax of a language

3. It should be possible to have several graphical syntaxes for a single language

4. Describe the composition of languages abstract syntaxes

5.Describe the composition of languages graphical syntaxes

6. The composition of the graphical representation should follows the composition of the abstract syntax

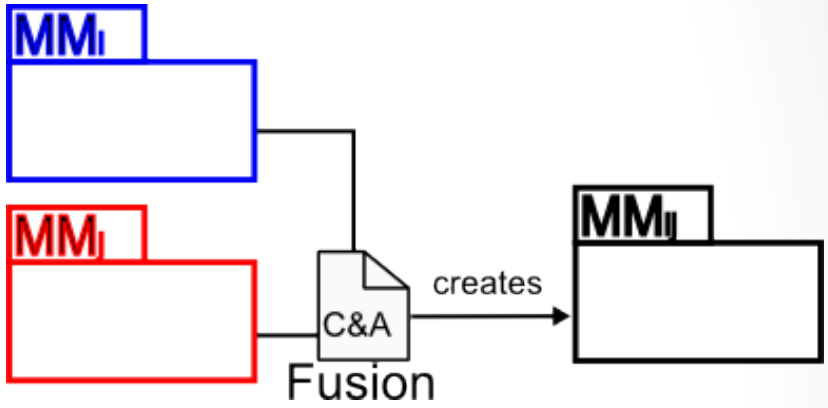
Fusion

Describe the composition of the abstract syntax

Language to adapt and compose existing metamodels

Imperative language with 16 actions for metaentitites and relations:

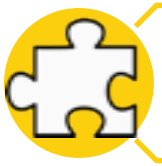
- Modifying
- Deleting
- Merging
- Spliting




```
import "./models/MMi.ecore“
import "./models/MMj.ecore“

export “MMij"
"http://uniandes.edu.co/exampleMMij"
```

Specific Requirements

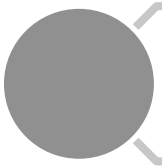
1.Use metamodels to describe the abstract syntax of a language



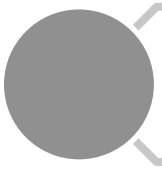
2.Separate the graphical and abstract syntax of a language

3. It should be possible to have several graphical syntaxes for a single language

4.Describe the composition of languages abstract syntaxes



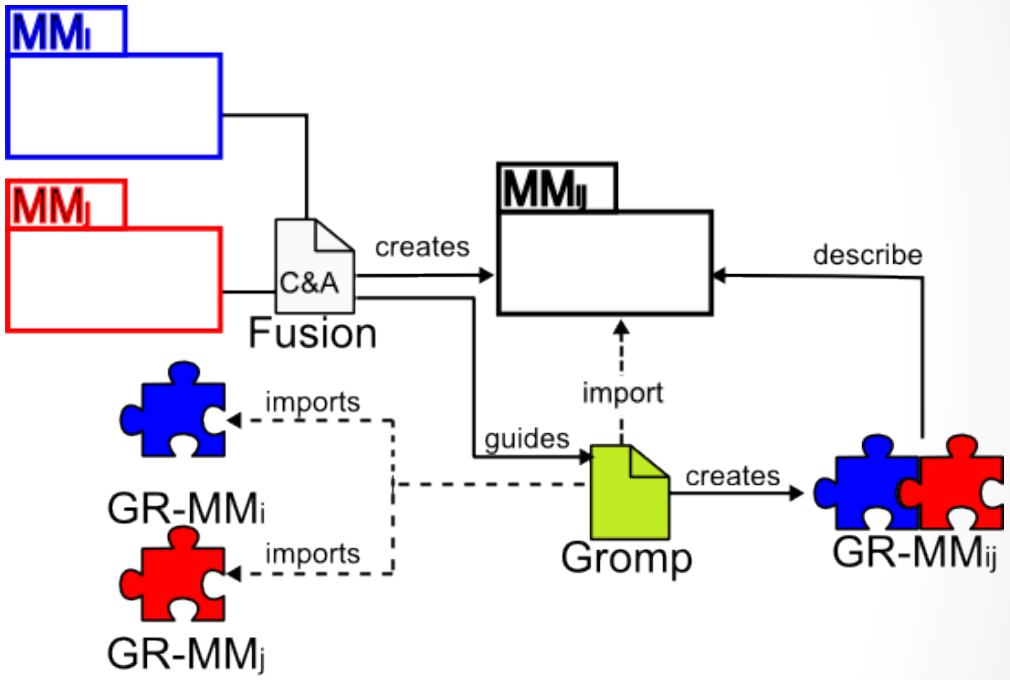
5.Describe the composition of languages graphical syntaxes



6. The composition of the graphical representation should follows the composition of the abstract syntax

Gromp

- A language designed to adapt and compose graphical representations.
- Aligned with the composition of the abstract syntax
- Imperative Language



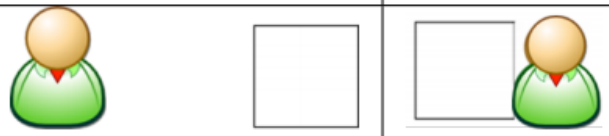
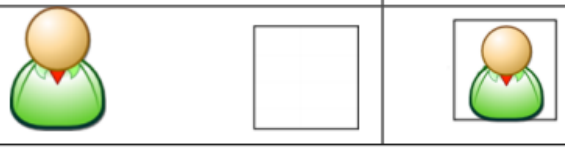
Gromp Instructions

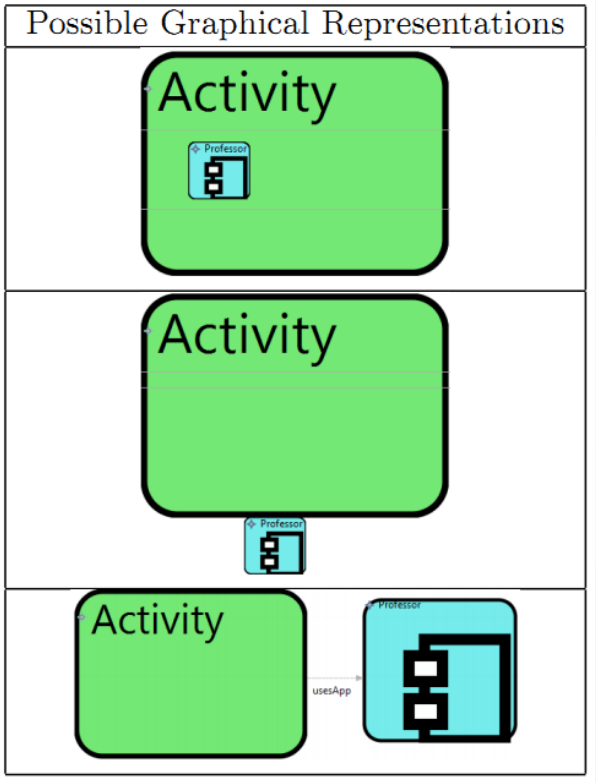
- 35 instructions in 5 categories

```
create new NodeElementDef "ProcessGraphDef" for MacroProcess {
    using
    Merge ( bpmn.ActivityDef ,buLayer.BusinessProcess)->(30,90,1,1)
    //(merge: graphicalDef1, graphicalDef 2, with2, height, coordx, coordy

    {
        label processName
        label icon false
        label placement external
        size  (500, 400)
        phantom false


    }
     create new button "MacroProcessButton" in group bpmn.ElementsPalette{
        description "MacroProcess boton"
        icon "bau.svg"
    }
```

Category	Instruction Name
Main Rules	import Metamodel import Picture
Rename	renameEntity renameAttribute renameRelation
Deletion	delete node_element delete Node_link delete Creation button for delete reference rule delete reference case delete entity rule delete entity case delete wizard delete wizard page delete wizard page attribute delete wizard page relation delete wizard page button delete statement delete statement event
Extension	create new NodeElementDef create new NodeElement create new ReferenceElementDef create new SubReferenceElementDef add entity rule add entity case add reference rule add reference case add attribute to add reference to add page to add new wizard by add new event add new statement
Other Instructions	use reference representation set Root

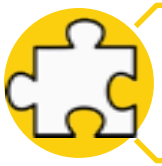
Gromp Operators

Operator	Before	After
Join Side Order		
Join Top Order		
Merge		



Specific Requirements

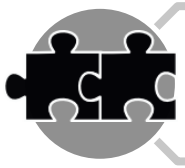
1.Use metamodels to describe the abstract syntax of a language



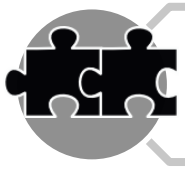
2.Separate the graphical and abstract syntax of a language

3. It should be possible to have several graphical syntaxes for a single language

Describe the composition of languages abstract syntaxes



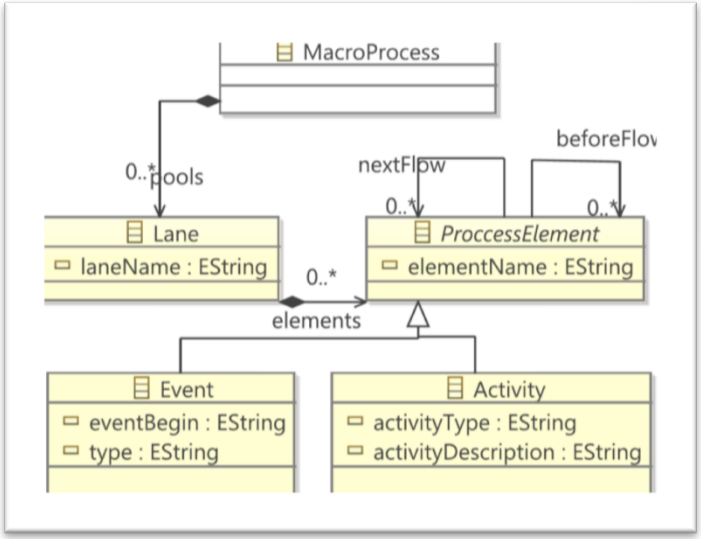
5.Describe the composition of languages graphical syntaxes

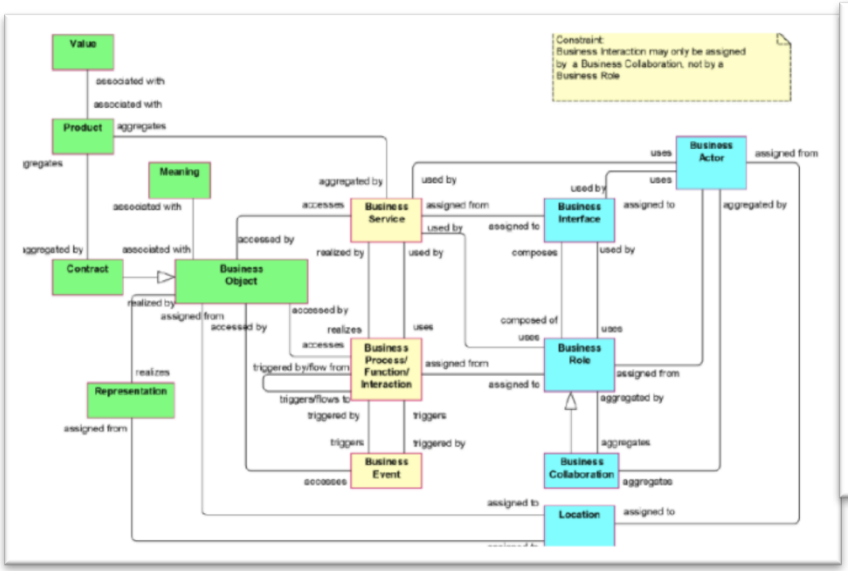


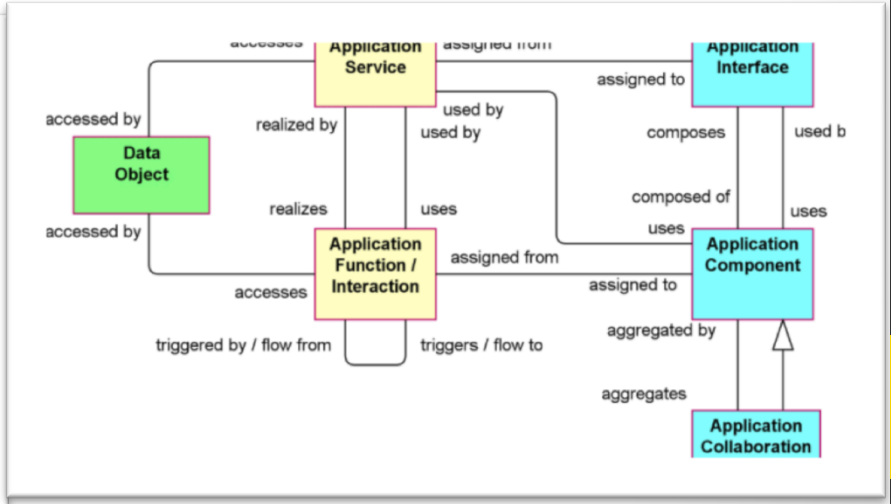
6. The composition of the graphical representation should follows the composition of the abstract syntax

Example

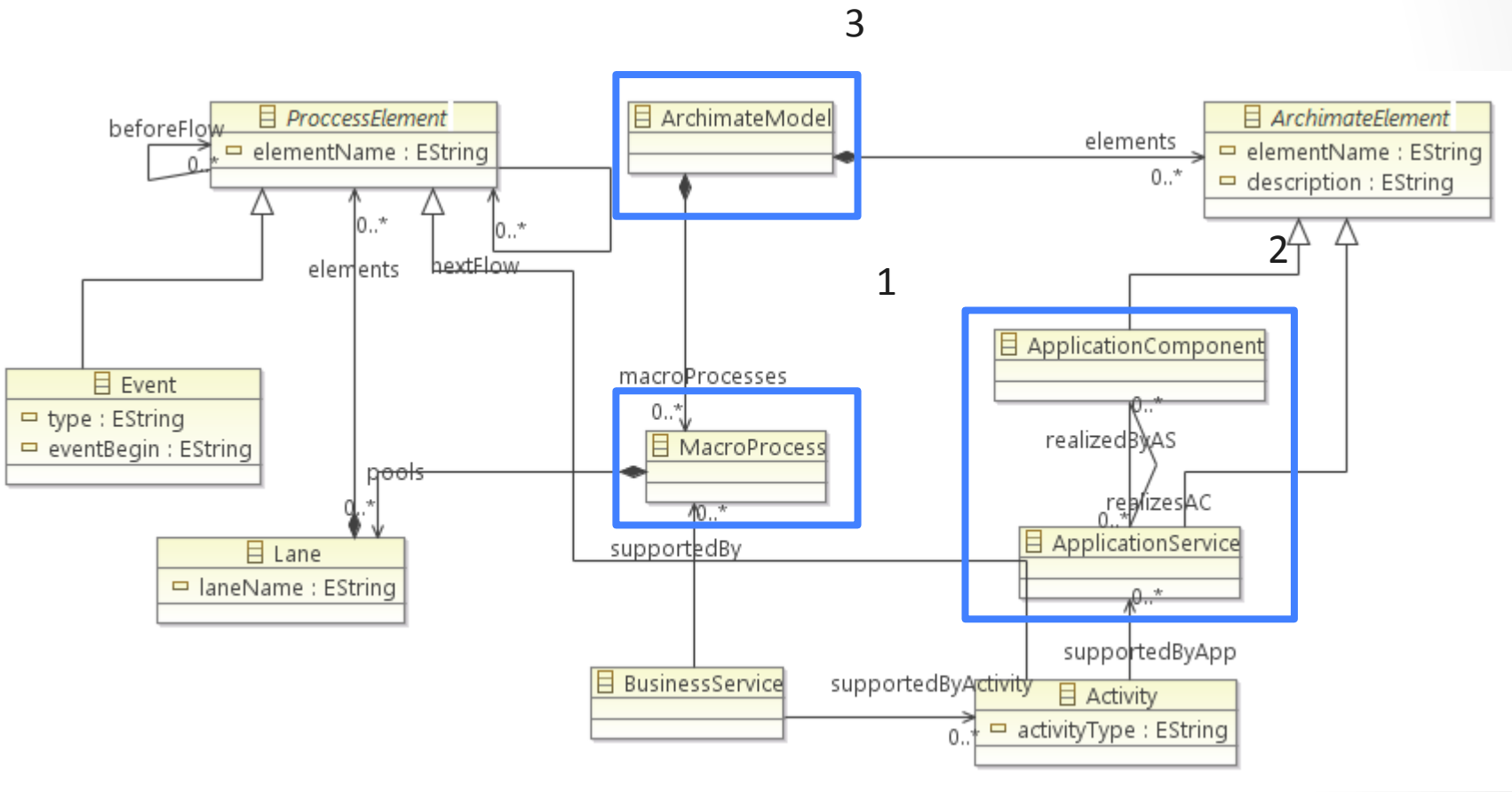
- Archimate Business Layer
- Archimate Application Layer
- “Bpmn”







Metamodel Composition & Adaptation



Example – Adaptation & Composition

```
import Metamodel "/model/composedMetamodel.ecore" as MM
import Picture "E:/runtime-EOLTest/AppLayer/model/APPLayerGromp.picture" as appLayer
import Picture "E:/runtime-EOLTest/BpmnOriginal/model/BpmnGRep.picture" as bpmn
import Picture "E:/runtime-EOLTest/BusinessLayer/model/BusinessLayerArchi.picture" as buLayer

set Root ArchimateModel

//Merge: Process & MacroPorcess -> into MacroProcess using Process Graphical Definition
create new NodeElementDef "ProcessGraphDef" for MacroProcess {
    using Node_element ( buLayer.BusinessProcessDef ){
        label processName
        size (500, 400)
    }
    create new button "MacroProcessButton" in group bpmn.ElementsPalette{
        description "MacroProcess boton"
        icon "bau.svg"
    }

}
```

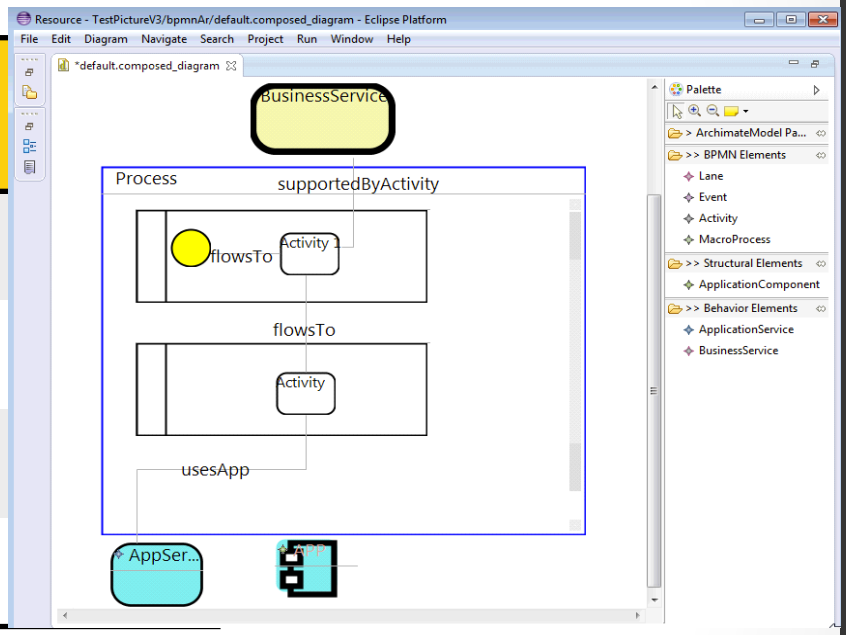
Example – Adaptation & Composition

```
//creates two new graphical definition reusing an existing one
use reference representation ProccessElement.nextFlow  for {
    reference ApplicationComponent.supportedByApp changing{
        label "usesApp"
    }
    reference BusinessService.supportedByActivity changing{
        label "supportedByActivity"
    }
    reference BusinessService.supportedBy changing{
        label "supportedByMacroProcess"
    }
    reference ApplicationComponent.realizesAC changing{
        label "realized"
        source decoration  none
        target decoration closed arrow
    }
}

create new ReferenceElementDef "GraphRepresentationForPools"
for reference MacroProcess.pools  using Internal_node {
    default layout
}
```

Example - results

Picture & MM:

Metamodel	MetaEntitiies	Picture	Source Files	LOC
Business Layer	16	1285 LOC	795	217596
Application Layer	8	748LOC	437	86952
BPMN	5	117 LOC	157	29415
ComposedMM	10	377 LOC	266	56196



composition

File	LOC
Fusion	48
Gromp	42

Conclusions

- Picture & Gromp make possible to reduce the time and complexity needed to create a new graphical editor by reusing existing graphical syntaxes and using concepts of adaptation and composition of graphical representations.
- The separation of graphical and abstract syntax created with Picture makes possible to have several graphical syntaxes for a single language.
- It is possible have metamodels and picture representations repositories to reduce the time during following projects

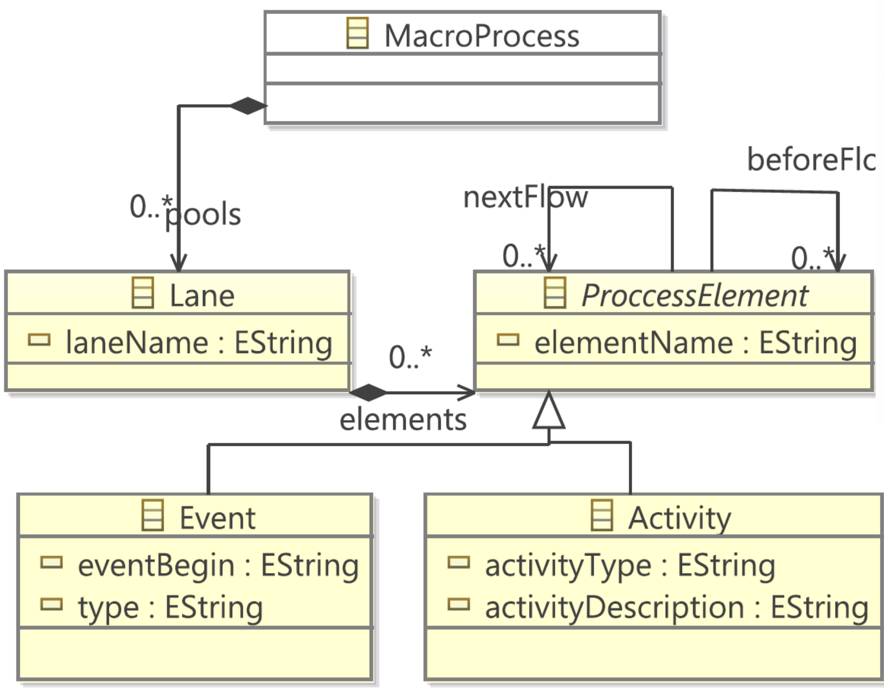
Questions?

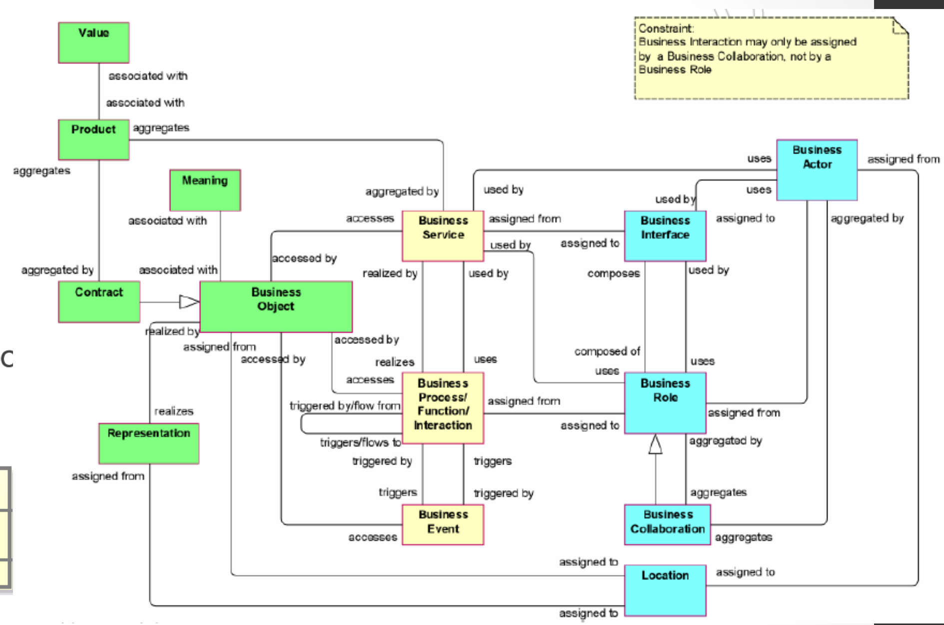
im.melo33@uniandes.edu.co

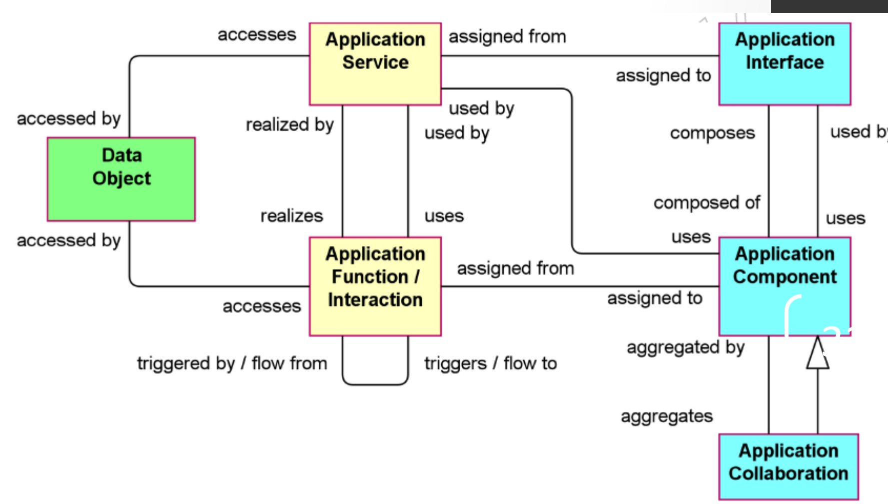
Conclusions

- Advantages with respect to the definition of languages and the generation of their editors.
- Definition of languages in a modular way thanks to the tools to compose and adapt languages.
- Cost effective
- Increasing reusability among projects

Example

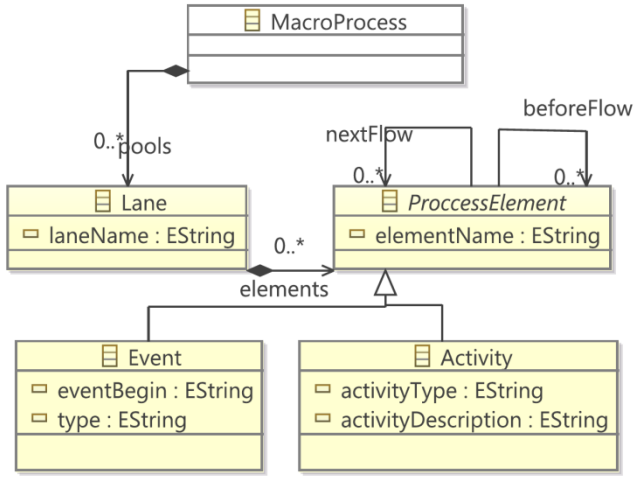

MacroProcess
0..*
pools
Lane
laneName : EString
0..*
elements
nextFlow
0..*
beforeFlc
0..*
ProccessElement
elementName : EString
Event
eventBegin : EString
type : EString
Activity
activityType : EString
activityDescription : EString

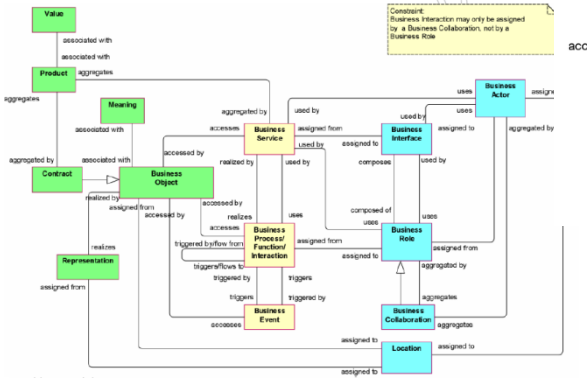

Value
Product
Meaning
Contract
Business Object
Representation
Business Service
Business Process/ Function/ Interaction
Business Event
Business Interface
Business Actor
Business Role
Business Collaboration
Location
Constraint: Business Interaction may only be assigned by a Business Colaboration, not by a Business Role
associated with
aggregates
aggregated by
accesses
accessed by
realized by
realizes
used by
uses
assigned from
assigned to
composes
composed of
triggers
triggered by
triggered by/flow from
triggers/flows to

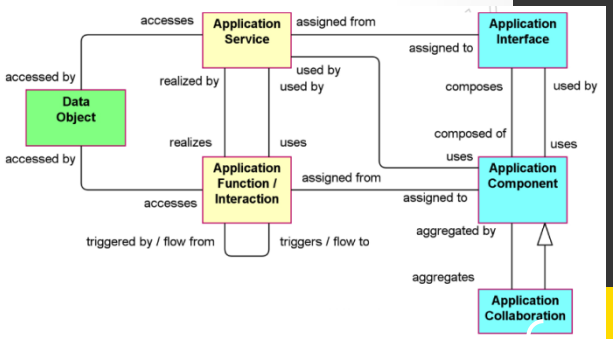

Application Service
Application Interface
Data Object
Application Function / Interaction
Application Component
Application Collaboration
accesses
assigned from
assigned to
accessed by
realized by
used by
composes
realizes
uses
composed of
aggregated by
aggregates
triggered by / flow from
triggers / flow to

Example

- 3 metamodels
- 3 graphical representations:
  - Business Layer = 1285LOC
  - Application Layer = 748 LOC
  - BPMN = 117LOC







Universidad de los Andes

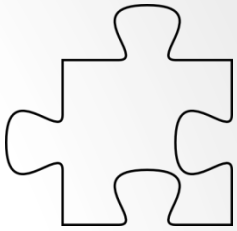
Construcción de modelos de simulación

ISIS – 3302 Modelaje, Simulación y Optimización

Mario Sánchez
2013

Departamento de Ingeniería de Sistemas y Computación

Style Definition



Principles

- Defines:
  - Colors
  - Shapes
  - Lines
- Referenced by the graphical representations.

Syntax

```
Style definition {
    Color black (0, 0, 0)
    Color white (255, 255, 255)
    Color yellow (255, 255, 0)
    Color blue (0, 0, 255)
    Color red (255, 0, 0)
    Color green (0, 255, 0)
    Line style BorderBlack1 {
        width 1
        type dot
        color black
    }
    Rounded rectRounded10
    {
        radiox 10
        radioy 10
    }
    Ellipse ellip {
        proportion ( 22, 17)
    }

    Line style BorderBlack3 {

    Line style BorderBlack4 {

    Line style BorderRed2 {

    Regular polygon deca {
        vertex quantity 8
        start angle 90
    }

    Color Withe (255, 255, 255)
    Color activeElementsColor (117, 235, 235)
    Color pasiveElementsColor (116, 232, 116)
    Color behaviorElementsColor (241, 241, 181)
    Color greenTestXX (44, 255, 45)
```