N-way Model Merging

Presenter: Mohammad-Sajad Kasaei

smskasaei@gmail.com

TELECOM Paris

INSTITUT POLYTECHNIQUE DE PARIS

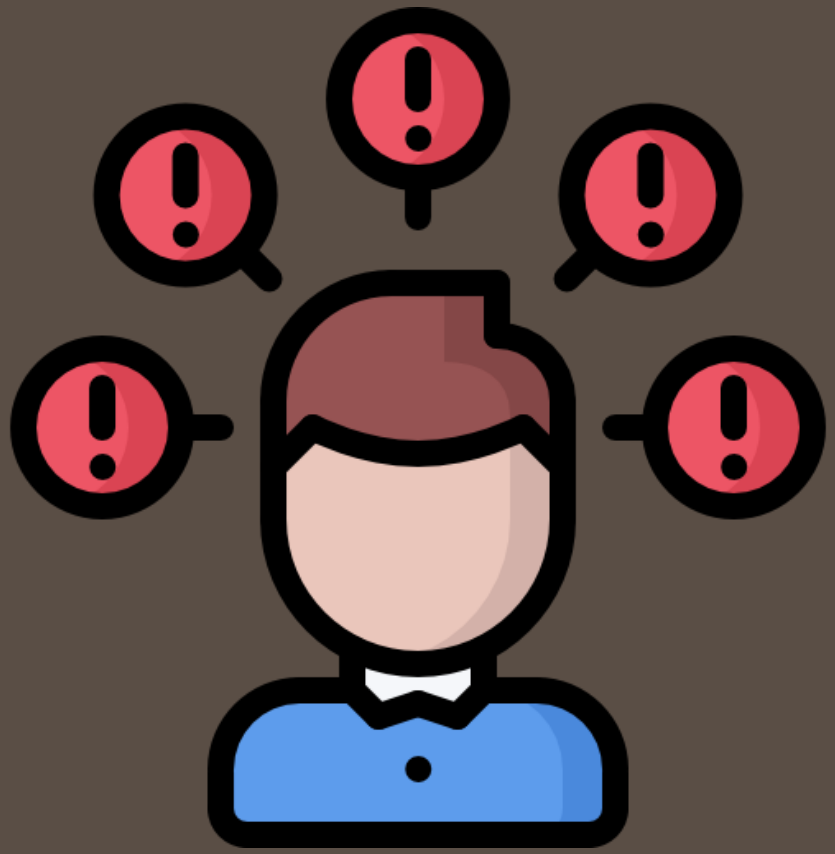
University of Isfahan

MDSE Research Group

TABLE OF CONTENTS

01 Problem Statement

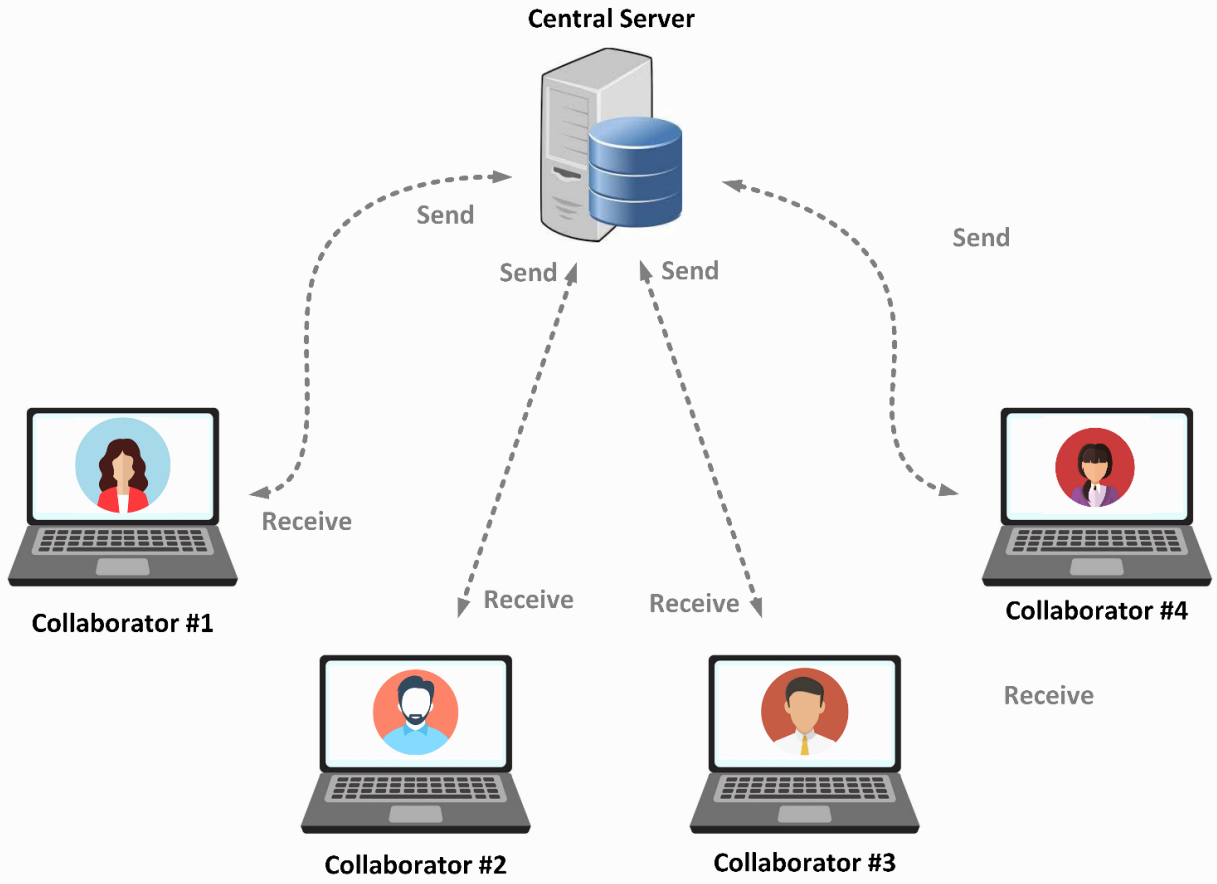
02 The Proposed Approach

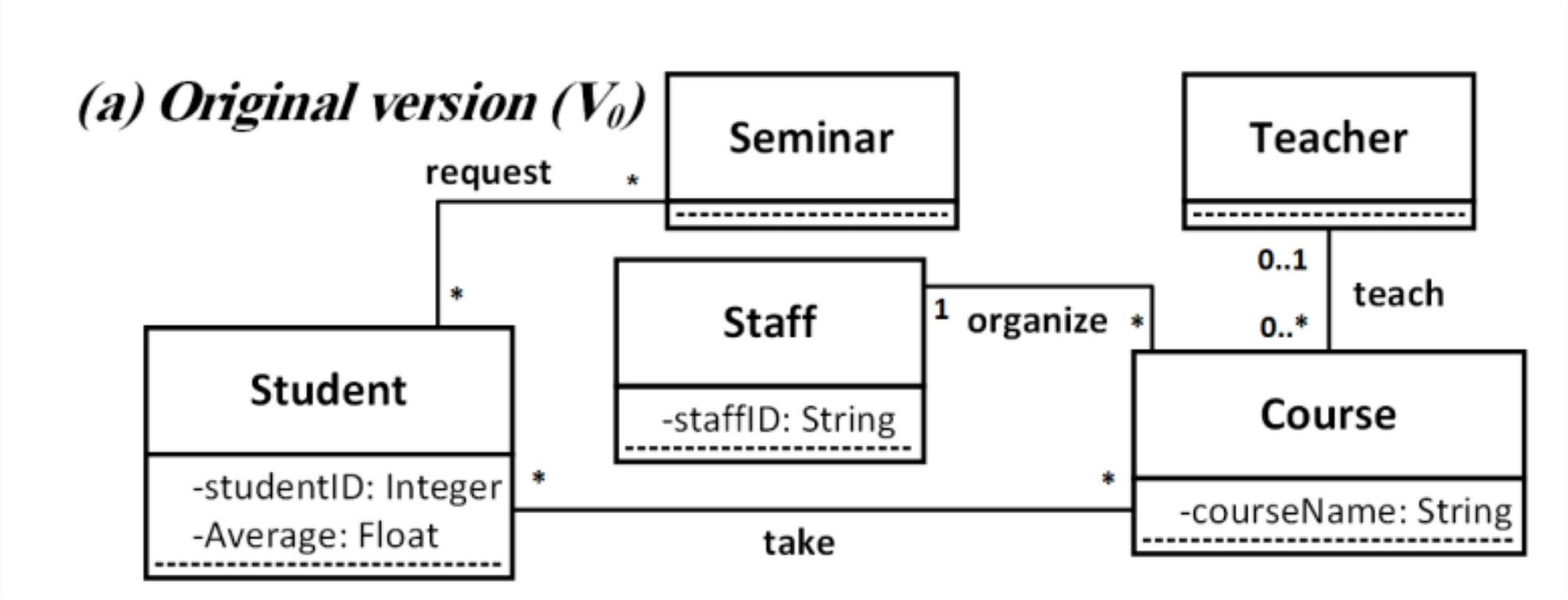


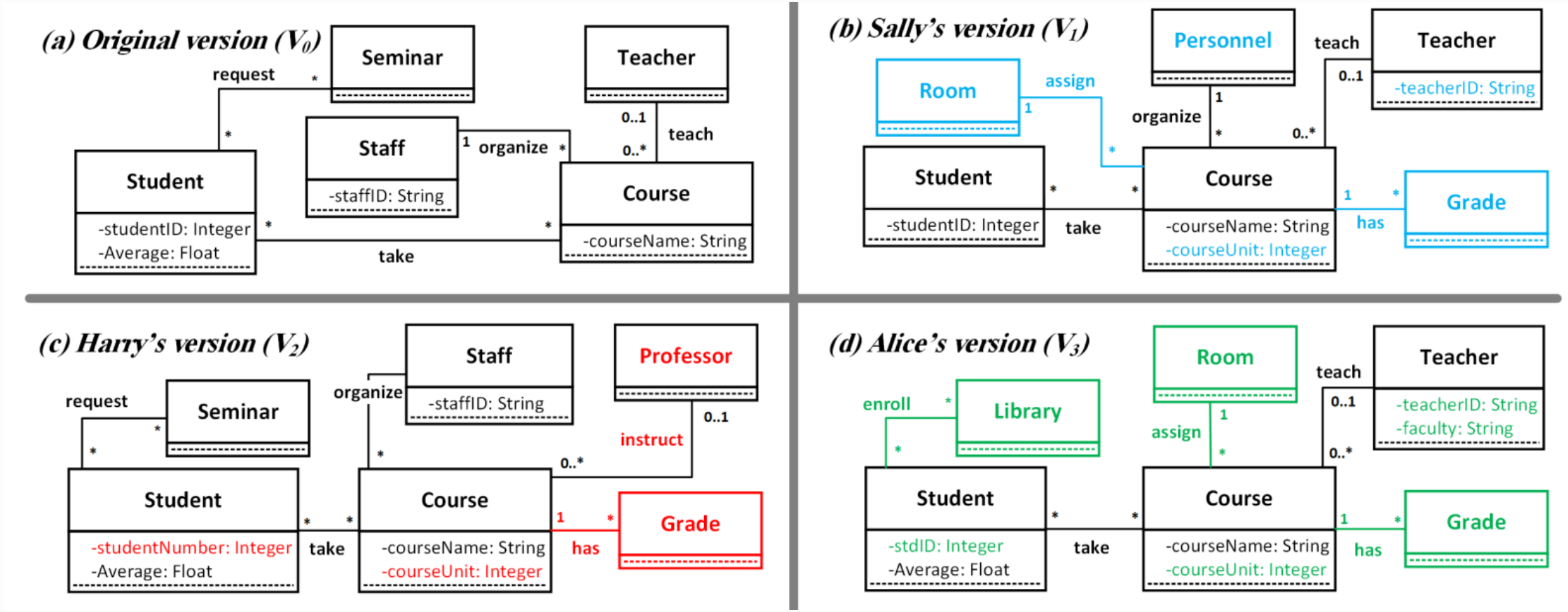
01 Problem Statement

We aim to design a model for a complex system

- The collaboration of different experts
  - A system to simplify this collaboration




(a) Original version (V_0)
Seminar
request
*
Teacher
0..1
0..*
teach
*
Staff
1
organize
*
Student
-staffID: String
-studentID: Integer
-Average: Float
*
*
Course
-courseName: String
take

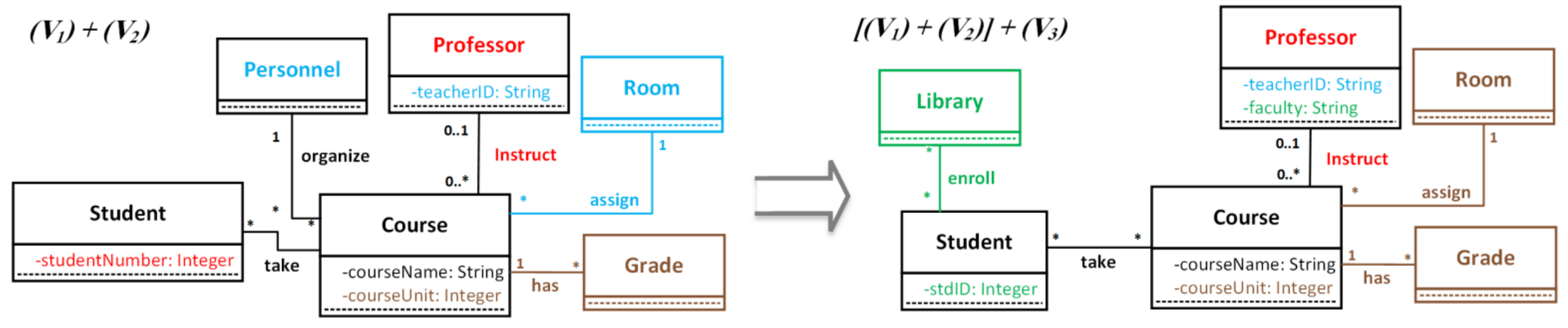

(a) Original version (V_0)
Seminar
request
Teacher
0..1
teach
0..*
Staff
-staffID: String
1
organize
*
Student
-studentID: Integer
-Average: Float
*
take
*
Course
-courseName: String
(b) Sally's version (V_1)
Personnel
teach
Teacher
0..1
-teacherID: String
Room
assign
1
organize
1
*
0..*
*
Student
-studentID: Integer
*
take
*
Course
-courseName: String
-courseUnit: Integer
1
has
*
Grade
(c) Harry's version (V_2)
request
Seminar
organize
Staff
-staffID: String
Professor
0..1
instruct
0..*
*
*
Student
-studentNumber: Integer
-Average: Float
*
take
*
Course
-courseName: String
-courseUnit: Integer
1
has
*
Grade
(d) Alice's version (V_3)
enroll
*
Library
*
Room
assign
1
*
teach
Teacher
0..1
-teacherID: String
-faculty: String
0..*
Student
-stdID: Integer
-Average: Float
*
take
*
Course
-courseName: String
-courseUnit: Integer
1
has
*
Grade

Our goal is to create a unified merged model.

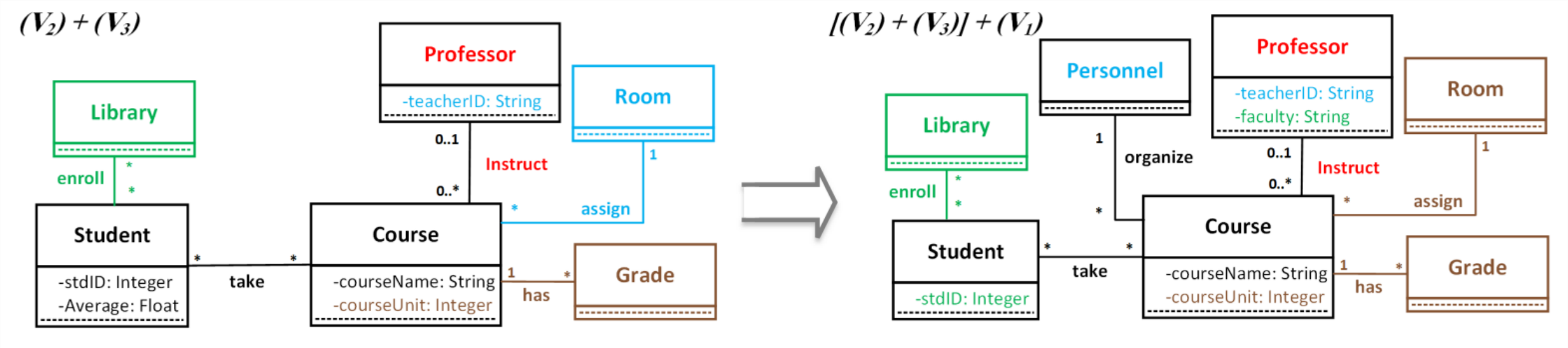
The question is: **HOW TO CREATE IT?**

- Pairwise Merging
- N-way Merging

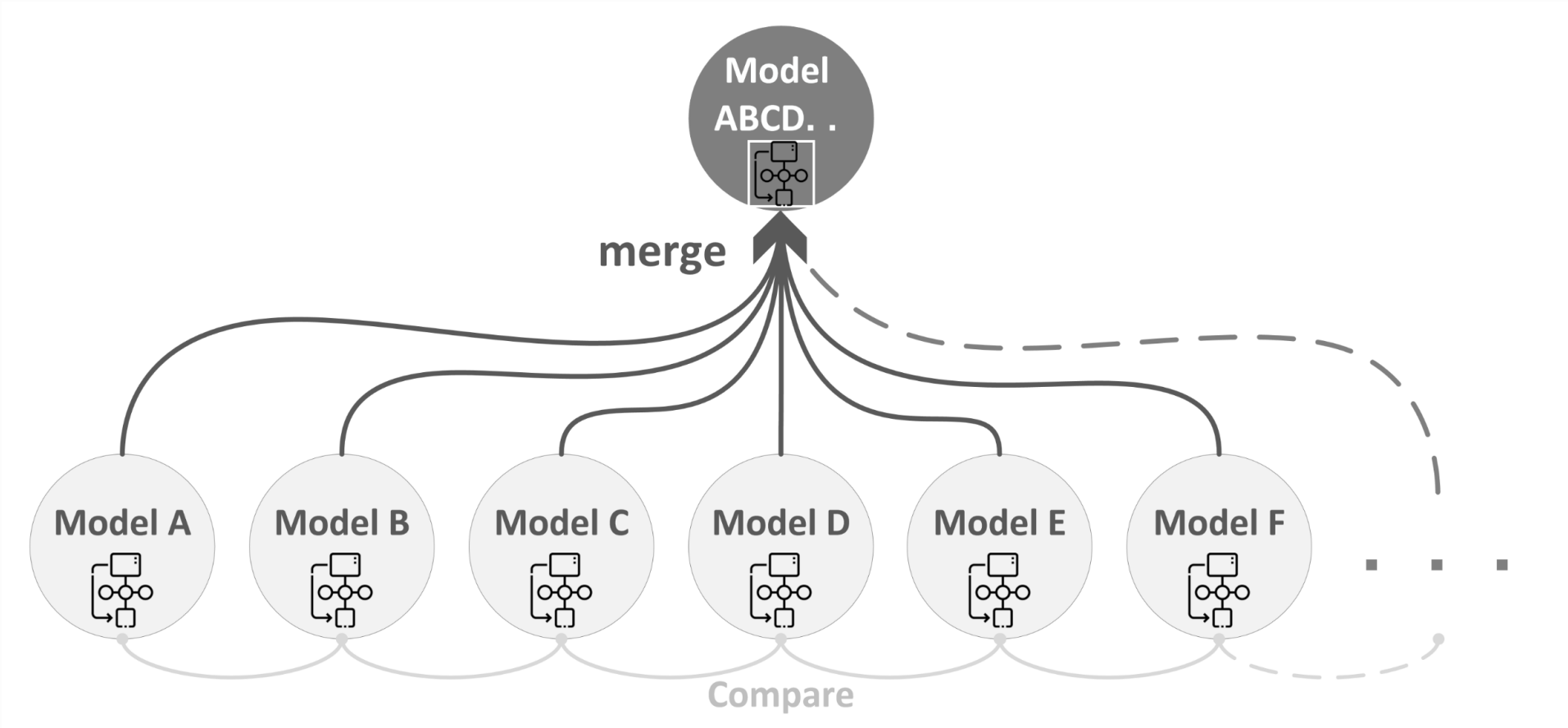
Pairwise Merging



Pairwise Merging

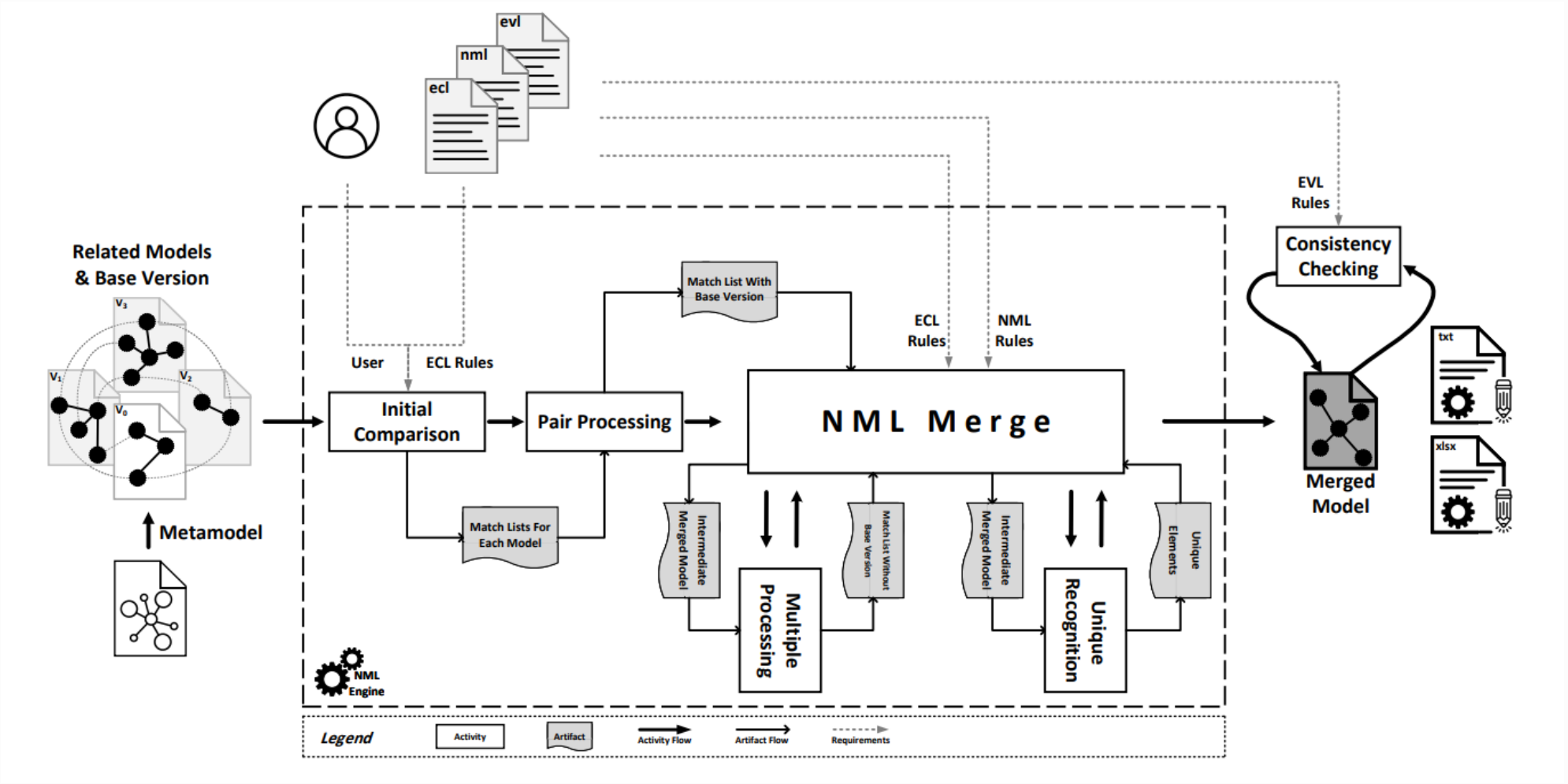


N-way Merging

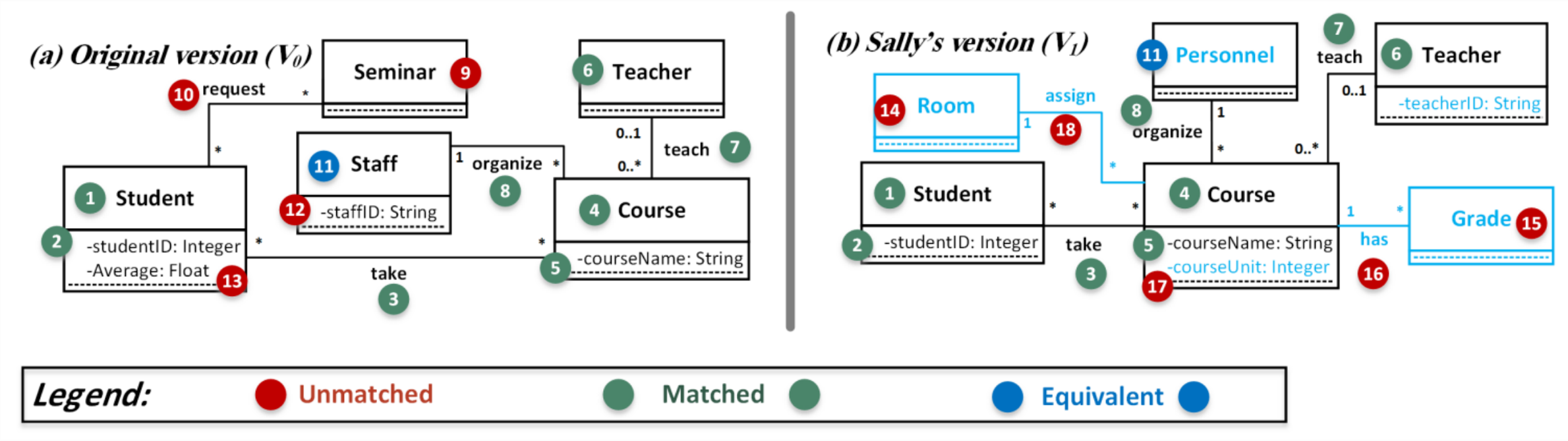


02 The Proposed Approach

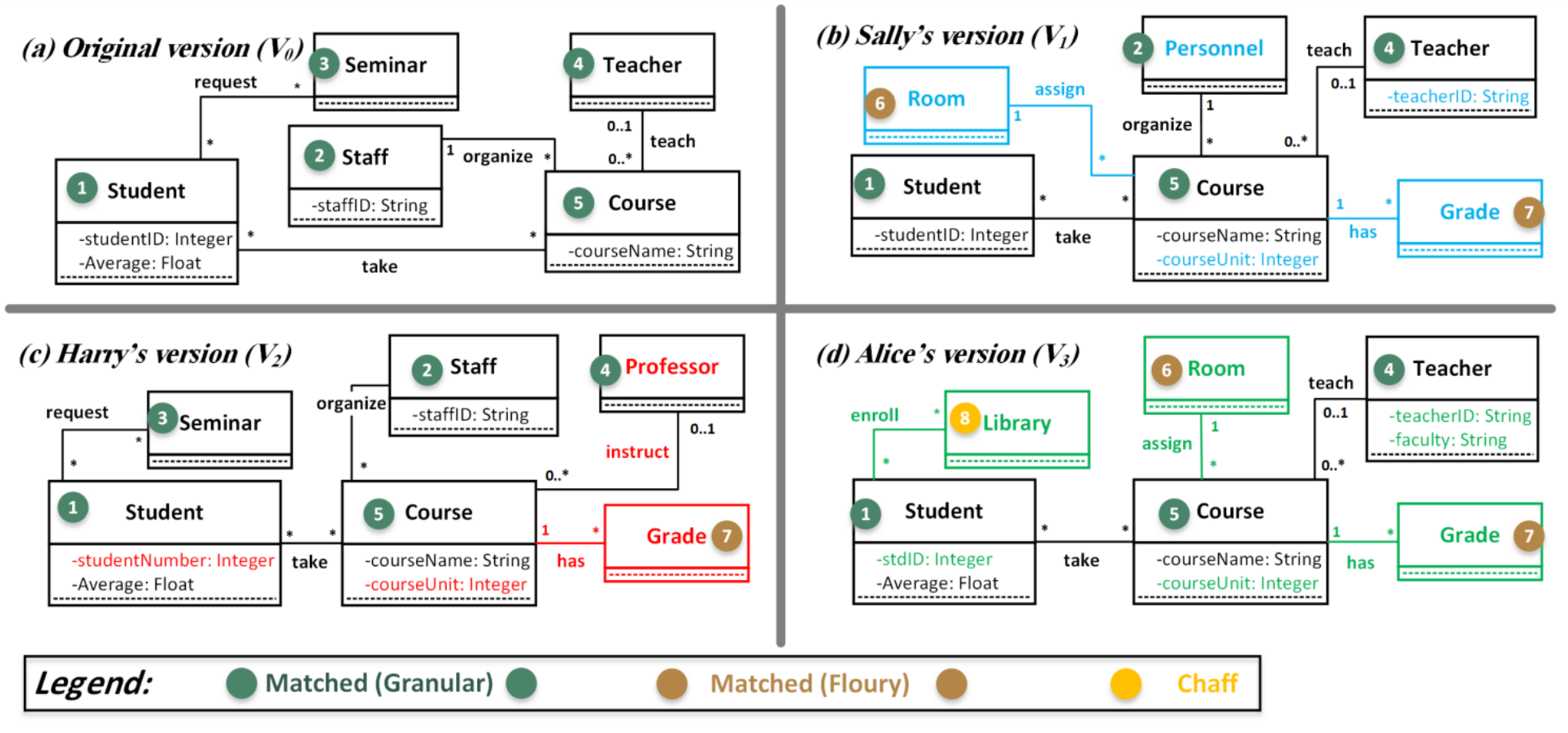
Workflow


evl
nml
ecl
Related Models & Base Version
Metamodel
User
ECL Rules
Initial Comparison
Pair Processing
Match List With Base Version
Match Lists For Each Model
ECL Rules
NML Rules
NML Merge
Intermediate Merged Model
Multiple Processing
Match List Without Base Version
Intermediate Merged Model
Unique Recognition
Unique Elements
NML Engine
EVL Rules
Consistency Checking
Merged Model
txt
xlsx
Legend
Activity
Artifact
Activity Flow
Artifact Flow
Requirements

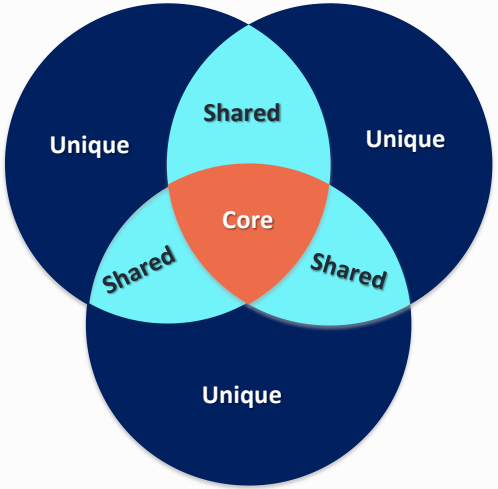
Comparison Phase



Conformance Checking Phase



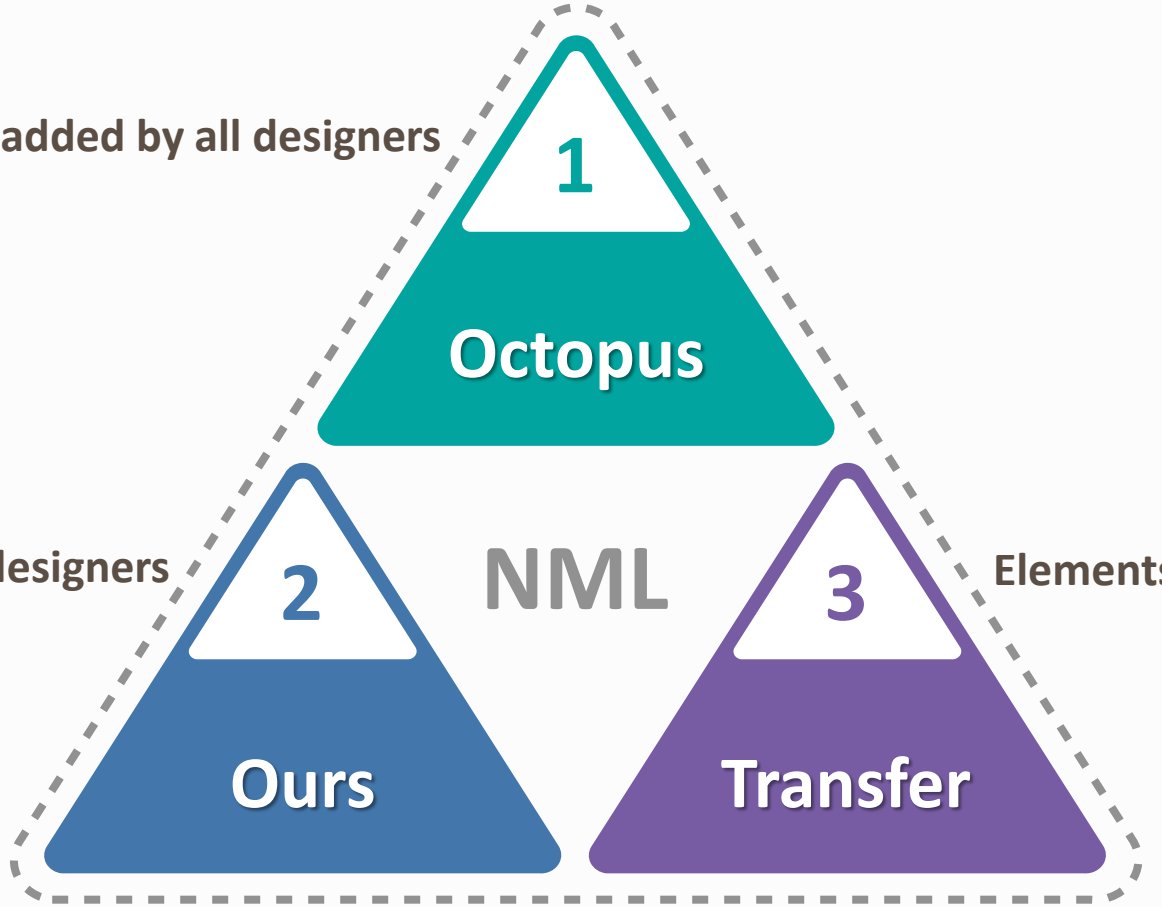
Merging Phase



Similar elements added by all designers

Similar elements added by some designers

Elements added by only one designer



```
rule <name>
   mergeOctopus <parameterName>:<versionParameterType>
   with  <parameterName>:<versionParameterType>
              (, <parameterName>:<versionParameterType>)*
  (withBase <parameterName>:<baseParameterType>)?
   into  <parameterName>:<targetParameterType>
           (, <parameterName>:<targetParameterType>)*
  (extends <ruleName> (, <ruleName>)*)?   {
              statementBlock
  }
```

Octopus

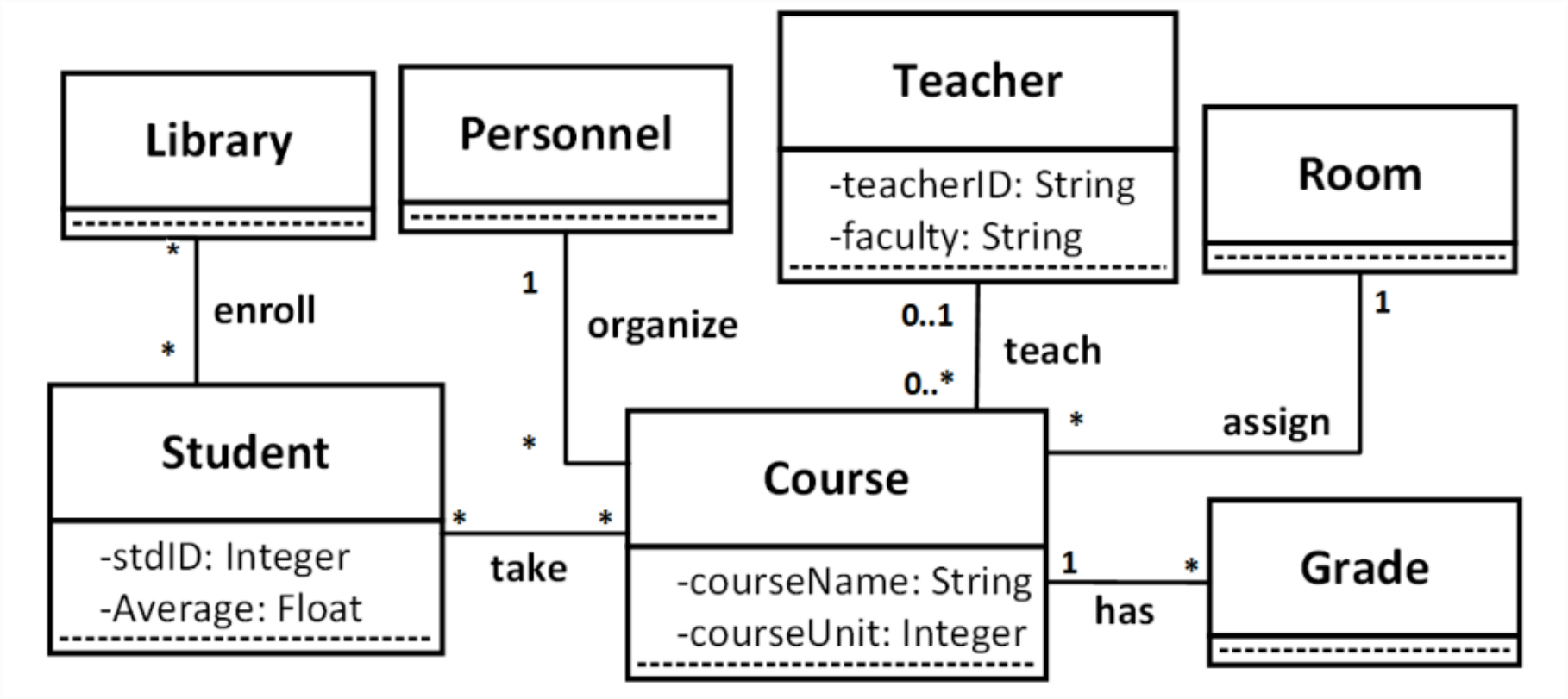
```
rule <name>
    transfer <parameterName>:<sourceName>!<parameterType>
    from <sourceName> :
              (<versionName>(, <versionName>)*)
    to  <parameterName>:<targetParameterType>
              (, <parameterName>:<targetParameterType>)*
    (extends <ruleName> (, <ruleName>)*)?   {
                statementBlock
  }
```

Transfer

```
rule <name>
   mergeOurs <parameterName>:<versionParameterType>
   with  <parameterName>:<versionParameterType>
              (, <parameterName>:<versionParameterType>)*
   (withBase <parameterName>:<baseParameterType>
        (exists_in <numberOfVersions>)? )?
   priority <priorityListName> :
                    [<parameterName>(, <parameterName>)*]
          (, <priorityListName> :
                    [<parameterName>(, <parameterName>)*])*
   into  <parameterName>:<targetParameterType>
          (, <parameterName>:<targetParameterType>)*
   (extends <ruleName> (, <ruleName>)*)?   {
              statementBlock
 }
```

Ours

Reconciliation Phase



Thank you for your attention

Department of Software Engineering
University of ISFAHAN, Isfahan, Iran

Model Driven Software Engineering Research Group

Department of Computer Science and Networks
Télécom Paris, Palaiseau, France