UNIVERSITÀ DEGLI STUDI DELL'AQUILA

Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica

CORSO DI LAUREA MAGISTRALE IN INFORMATICA (ASE)

Insegnamento Model Driven Engineering

NAME AND SURNAME	STUDENT NUMBER
Luca Francesco Macera	302123
Calogero Carlino	302154

Metamodel Description

The metamodel represents a structure for modeling a restaurant management system; therefore, it’s designed to capture details of a restaurant's structure, management, and menu organization, accommodating various components needed for operational and structural information.

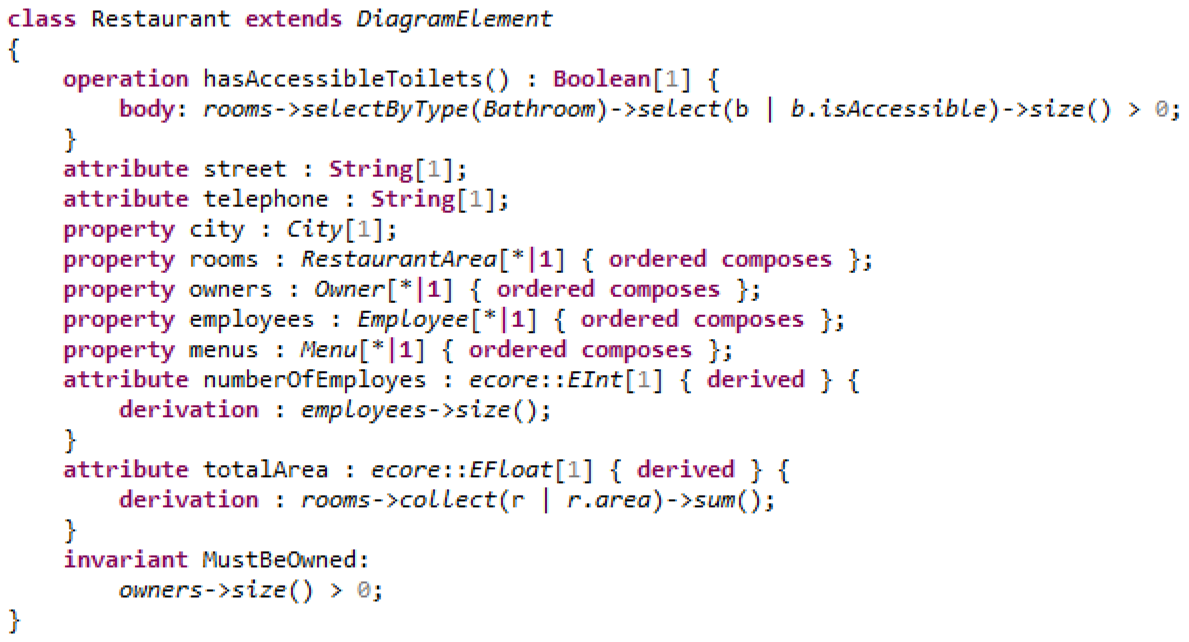
1. **NamedElement:** NamedElement is an abstract metaclass that only has a name attribute
2. **Diagram:** a Diagram concept that models our metamodel language and in which only DiagramElement concepts can be added as children
3. **DiagramElement:** a metaclass that extends from NamedElement and that can only be added as a Diagram concept child
4. **CourseType:** CourseType is an Enumerator that models the type of a restaurant dish, like for example fried, vegan, pizza, etc
5. **Gender:** Gender is an Enumerator that models the gender of a person
6. **Material:** Material is an Enumerator that models the materials a table can be made of, like for example glass, wood, plastic, etc
7. **Role:** Role is an Enumerator that models the type of job that an employee has inside a restaurant, like for example cashier, chef, waiter etc
8. **Bathroom:** the Bathroom concept models a restaurant's bathroom with their size, number of toilets and specified gender
9. **City:** the City concept models real-life cities belonging to a region
10. **Course:** the Course concept models a typical restaurant dish with its number of pieces, price, type of food and description
11. **Diagram:** the Diagram concept is the root element of our language, it comprehends all the elements of the languages like restaurants, dining rooms, owners, etc.
12. **DiningRoom:** the DiningRoom concept models a restaurant's dining room with their size and tables
13. **Employee:** the Employee concept extends from Person and models the employee that works in the restaurant. Employees have a name, a surname, a date and place of birth, contract start and end dates and, finally, the salary
14. **Kitchen:** the Kitchen concept models a restaurant's kitchen room with its sizes and number of stoves
15. **Menu:** the Menu concept models a restaurant menu made up by courses
16. **Owner:** the Owner concept extends from Person and models a restaurant owner that has a name, surname, data and place of birth and VAT
17. **Person:** the Person concept is an abstract metamodel class that models a real-life person with their name, surname and data and place of birth
18. **Region:** the Region concept models real-life regions in which cities belong to
19. **Restaurant:** the Restaurant concept models an actual restaurant located in a street belonging to a certain city
20. **RestaurantArea:** the RestaurantArea concept is an abstract metamodel class that models a generic restaurant room that could be a kitchen or a bathroom, for example, with its size and name

21. **Table:** the Table concept models a table of a DiningRoom with their specified material, number and number of chairs

Metamodel Constraints

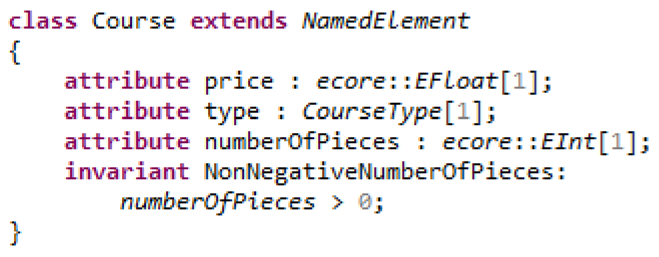
Our metamodel definition also has some constraints:

1. **MustBeOwned:** a Restaurant concept must be owned by at least one Owner Concept



```
class Restaurant extends DiagramElement
{
    operation hasAccessibleToilets() : Boolean[1] {
        body: rooms->selectByType(Bathroom)->select(b | b.isAccessible)->size() > 0;
    }
    attribute street : String[1];
    attribute telephone : String[1];
    property city : City[1];
    property rooms : RestaurantArea[*|1] { ordered composes };
    property owners : Owner[*|1] { ordered composes };
    property employees : Employee[*|1] { ordered composes };
    property menus : Menu[*|1] { ordered composes };
    attribute numberOfEmployes : ecore::EInt[1] { derived } {
        derivation : employees->size();
    }
    attribute totalArea : ecore::EFloat[1] { derived } {
        derivation : rooms->collect(r | r.area)->sum();
    }
    invariant MustBeOwned:
        owners->size() > 0;
}
```

2. **NonNegativeNumberOfPieces:** a Course concept numberOfPieces attribute must hold a non negative, greater than zero number



```
class Course extends NamedElement
{
    attribute price : ecore::EFloat[1];
    attribute type : CourseType[1];
    attribute numberOfPieces : ecore::EInt[1];
    invariant NonNegativeNumberOfPieces:
        numberOfPieces > 0;
}
```

3. **UniqueTableNumber:** all the Table concept in a DiningRoom concept must have unique numbers (there can not exist two tables with the same number)

```
class Table
{
    attribute number : ecore::EInt[1];
    attribute numberOfSeats : ecore::EInt[1];
    attribute material : Material[1];
    property diningRoom#tables : DiningRoom[?];
    invariant UniqueTableNumber:
        diningRoom.tables->select(t | t.number=number)->size() <= 1;
}
```

Metamodel Operations

Our metamodel definition also has some operations:

1. **hasAccessibleToilets():** an operation used to know if a Restaurant Concept has at least one Bathroom concept with isAccessible value set to true

```
class Restaurant extends DiagramElement
{
    operation hasAccessibleToilets() : Boolean[1] {
        body: rooms->selectByType(Bathroom)->select(b | b.isAccessible)->size() > 0;
    }
    attribute street : String[1];
    attribute telephone : String[1];
    property city : City[1];
    property rooms : RestaurantArea[*|1] { ordered composes };
    property owners : Owner[*|1] { ordered composes };
    property employees : Employee[*|1] { ordered composes };
    property menus : Menu[*|1] { ordered composes };
    attribute numberOfEmployes : ecore::EInt[1] { derived } {
        derivation : employees->size();
    }
    attribute totalArea : ecore::EFloat[1] { derived } {
        derivation : rooms->collect(r | r.area)->sum();
    }
    invariant MustBeOwned:
        owners->size() > 0;
}
```

2. **hasCourseType()**: an operation used to tell if a Menu concept has at least one Cource concept of the specified type

```
class Menu extends NamedElement
{
    operation hasCourseType(courseType : CourseType[1]) : Boolean[1] {
        body: courses->select(c | c.type = courseType)->size() > 0;
    }
    property courses : Course[*|1] { ordered composes };
    attribute numberOfCourses : ecore::EInt[1] { derived } {
        derivation : courses->size();
    }
}
```

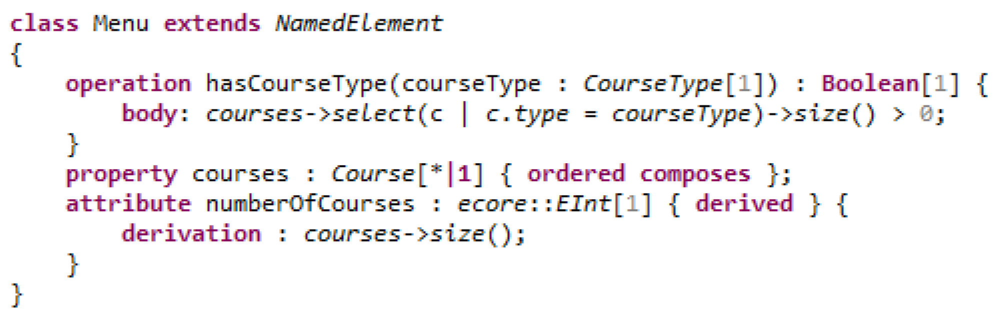
Metamodel Derived fields

Our metamodel definition also has some derived fields:

1. **numberOfEmployees:** a derived field that holds the Restaurant concept total number of employees
2. **totalArea:** a derived field that holds the Restaurant concept total area computed by summing up all the Restaurant concept's RestaurantArea concepts areas

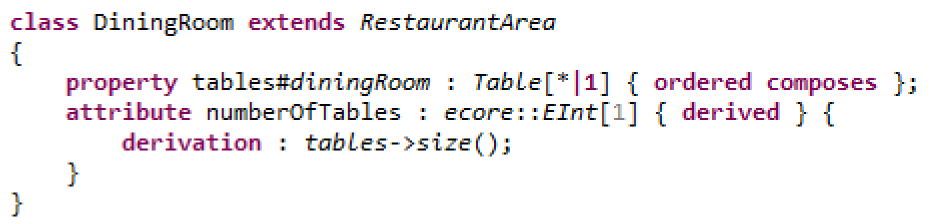
```
class Restaurant extends DiagramElement
{
    operation hasAccessibleToilets() : Boolean[1] {
        body: rooms->selectByType(Bathroom)->select(b | b.isAccessible)->size() > 0;
    }
    attribute street : String[1];
    attribute telephone : String[1];
    property city : City[1];
    property rooms : RestaurantArea[*|1] { ordered composes };
    property owners : Owner[*|1] { ordered composes };
    property employees : Employee[*|1] { ordered composes };
    property menus : Menu[*|1] { ordered composes };
    attribute numberOfEmployes : ecore::EInt[1] { derived } {
        derivation : employees->size();
    }
    attribute totalArea : ecore::EFloat[1] { derived } {
        derivation : rooms->collect(r | r.area)->sum();
    }
    invariant MustBeOwned:
        owners->size() > 0;
}
```

3. **numberOfCourses:** a derived field that holds a Menu concept total number of Course concepts



```
class Menu extends NamedElement
{
    operation hasCourseType(courseType : CourseType[1]) : Boolean[1] {
        body: courses->select(c | c.type = courseType)->size() > 0;
    }
    property courses : Course[*|1] { ordered composes };
    attribute numberOfCourses : ecore::EInt[1] { derived } {
        derivation : courses->size();
    }
}
```

4. **numberOfTables:** a derived field that holds the total number of Table concepts of a DiningRoom concept



```
class DiningRoom extends RestaurantArea
{
    property tables#diningRoom : Table[*|1] { ordered composes };
    attribute numberOfTables : ecore::EInt[1] { derived } {
        derivation : tables->size();
    }
}
```

5. s

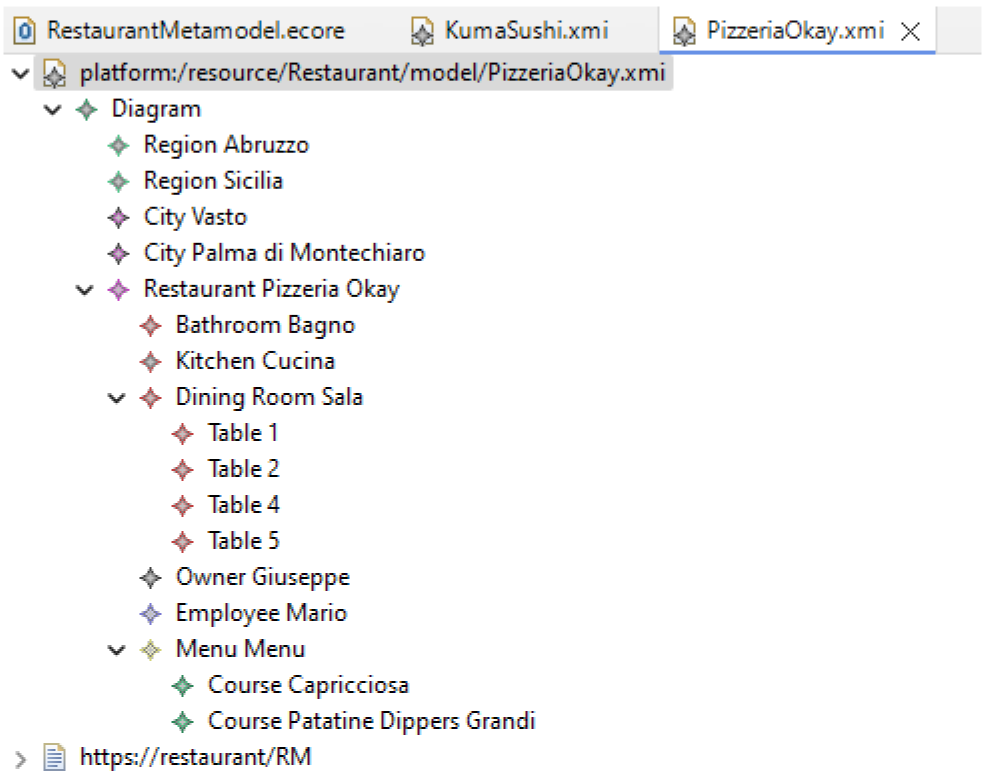
Plugin

We have created a plugin that creates, serializes, loads and validates models. The plugin project can be found in the **RestaurantPlugin** folder.

Models

We have also generated two models based on our defined language

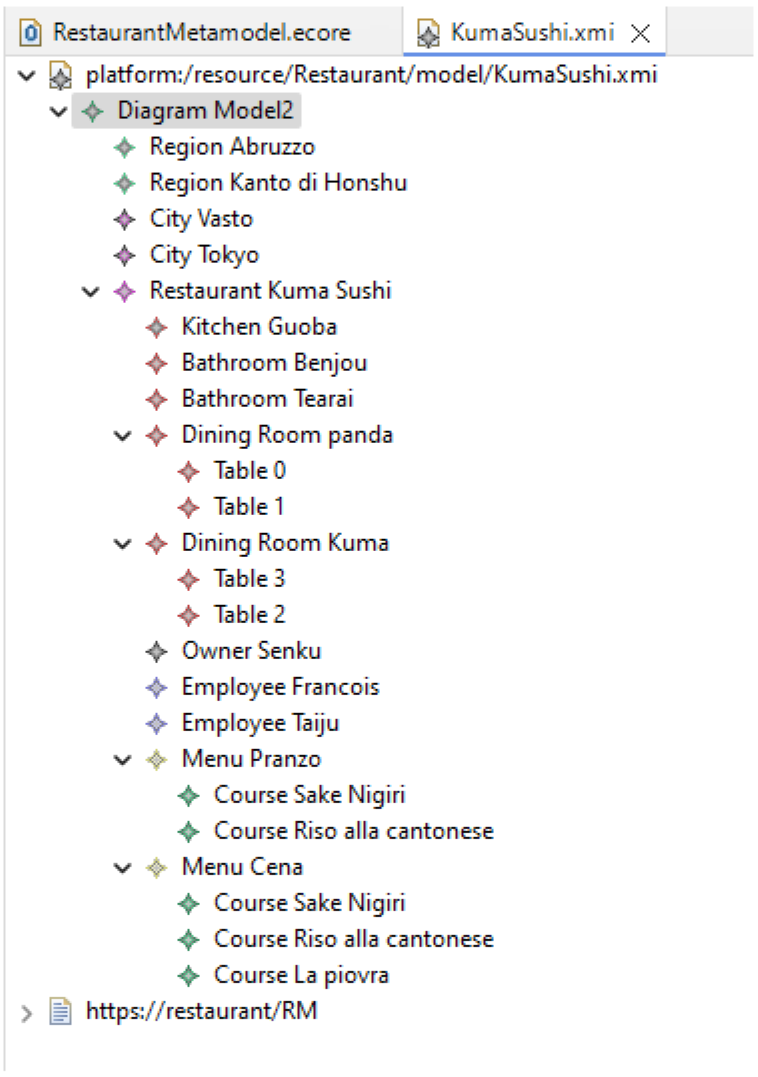
- Model 1



RestaurantMetamodel.ecore   KumaSushi.xmi   PizzeriaOkay.xmi

```
platform:/resource/Restaurant/model/PizzeriaOkay.xmi
  Diagram
    Region Abruzzo
    Region Sicilia
    City Vasto
    City Palma di Montechiaro
    Restaurant Pizzeria Okay
      Bathroom Bagno
      Kitchen Cucina
      Dining Room Sala
        Table 1
        Table 2
        Table 4
        Table 5
      Owner Giuseppe
      Employee Mario
      Menu Menu
        Course Capricciosa
        Course Patatine Dippers Grandi
https://restaurant/RM
```

- Model2



```
RestaurantMetamodel.ecore   KumaSushi.xmi
platform:/resource/Restaurant/model/KumaSushi.xmi
  Diagram Model2
    Region Abruzzo
    Region Kanto di Honshu
    City Vasto
    City Tokyo
    Restaurant Kuma Sushi
      Kitchen Guoba
      Bathroom Benjou
      Bathroom Tearai
      Dining Room panda
        Table 0
        Table 1
      Dining Room Kuma
        Table 3
        Table 2
      Owner Senku
      Employee Francois
      Employee Taiju
      Menu Pranzo
        Course Sake Nigiri
        Course Riso alla cantonese
      Menu Cena
        Course Sake Nigiri
        Course Riso alla cantonese
        Course La piovra
https://restaurant/RM
```