

FeatureLanguage: Automatic Generation of Application Backend for Model-Based Programming Course Projects

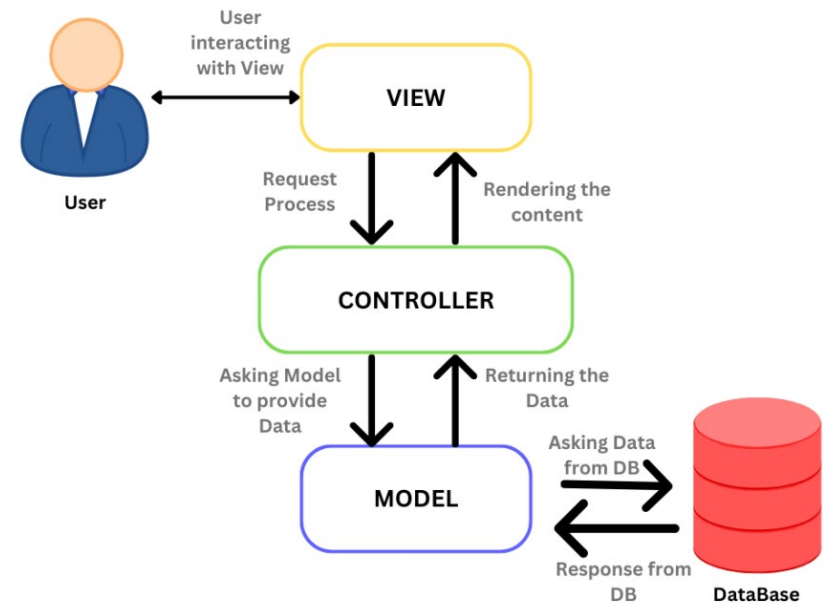
Erica De Petrillo, Gunter Mussbacher

McGill University

MoDRE'24, Reykjavik, Iceland ♦ June 25 2024

Model-Driven Engineering Courses

- Semester-long project: MVC application
- Learning by example?
 - Beneficial for students
 - Time-consuming for teaching staff



- Unless they use FeatureLanguage...

Typical MDE Course Project

- MVC app
 - Model layer generated from class diagram using Umple
 - Controller layer implemented by students using Java
 - View layer implemented by students using JavaFx
 - Test suite implemented by students using Cucumber
- Project size
 - ~ 16 features
 - ~ 25 constraints
 - ~ 150 unit tests
 - Derived from Gherkin scenarios

 Umple Online

 JavaFx™

 Java

cucumber 

Umple

- Modeling tool and programming language family
- Adds concepts from UML to OO languages
- Textual & graphical
- Umple features include:
 - Generating Java code from class diagrams
 - Generating Ecore diagrams from class diagrams

Umple Online Draw on the right, write (Umple) model code on the left. Analyse models and generate code.

Download Donate For help: User manual Ask questions Report issue

Line=61 E G S T D A M Generate Java Save as URL Hide Tabs Restore Saved State URL for 2DShapes example

```
1 // 2D Shapes class hierarchy - sample UML class
2 // diagram in Umple
3 // From Book by Lethbridge and Laganiere, McGraw Hill
4 // Object-Oriented Software Engineering:
5 // Practical Software Engineering using UML and Java
6 // See
7 // https://www.site.uottawa.ca/school/research/lloeng/
8 //Namespace for core of the system.
9 namespace Shapes.core;
10
11 class Shape2D {
12   center;
13 }
14 //Abstract
15 class EllipticalShape {
16   isA Shape2D;
17   semiMajorAxis;
18 }
19 //Abstract
20 class Polygon {
21   isA Shape2D;
22 }
23 class Circle {
24   isA EllipticalShape;
25 }
26 class Ellipse{
27   isA EllipticalShape;
28 }
29 class SimplePolygon {
30   orientation;
31   isA Polygon;
32 }
```

SAVE & LOAD

TOOLS

EXAMPLES

Class Diagrams

2DShapes *

DRAW

Class

Association

Generalization

Delete

Undo

Reindent Code

System Diagrams

GENERATE

Java Code

Generate It

OPTIONS

TASKS

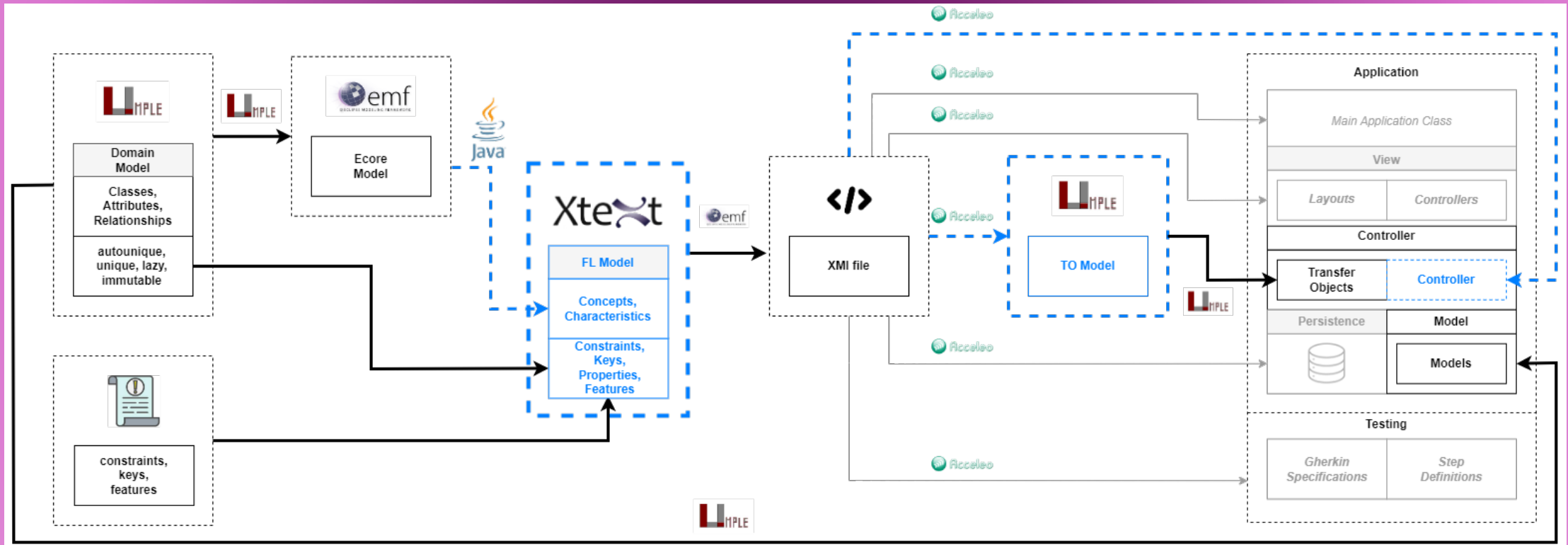
UML Class Diagram:

- Shape2D (center : String X) -- Add More --
- EllipticalShape (semiMajorAxis : String X) -- Add More --
- Polygon -- Add More --
- ArbitraryPolygon (points : String X) -- Add More --
- SimplePolygon (orientation : String X) -- Add More --
- Circle -- Add More --
- Ellipse -- Add More --
- RegularPolygon (numPoints : String X, radius : String X) -- Add More --
- Rectangle (height : String X, width : String X) -- Add More --

MVC App Generation Process

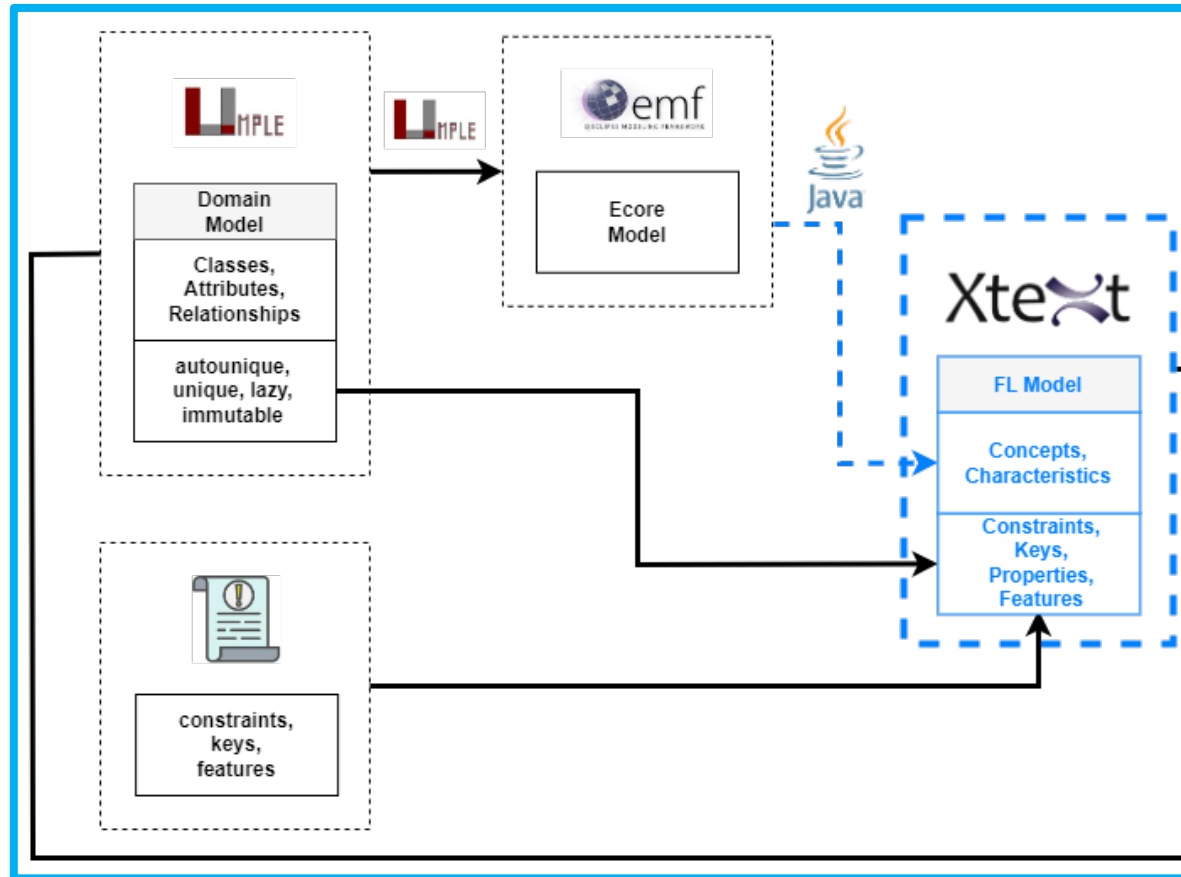
- MVC app
 - Handwritten code ✗
 - Generated code ✓
- Input
 - Domain model
 - Extra specification
- Output
 - Sample solution

Transformation Pipeline



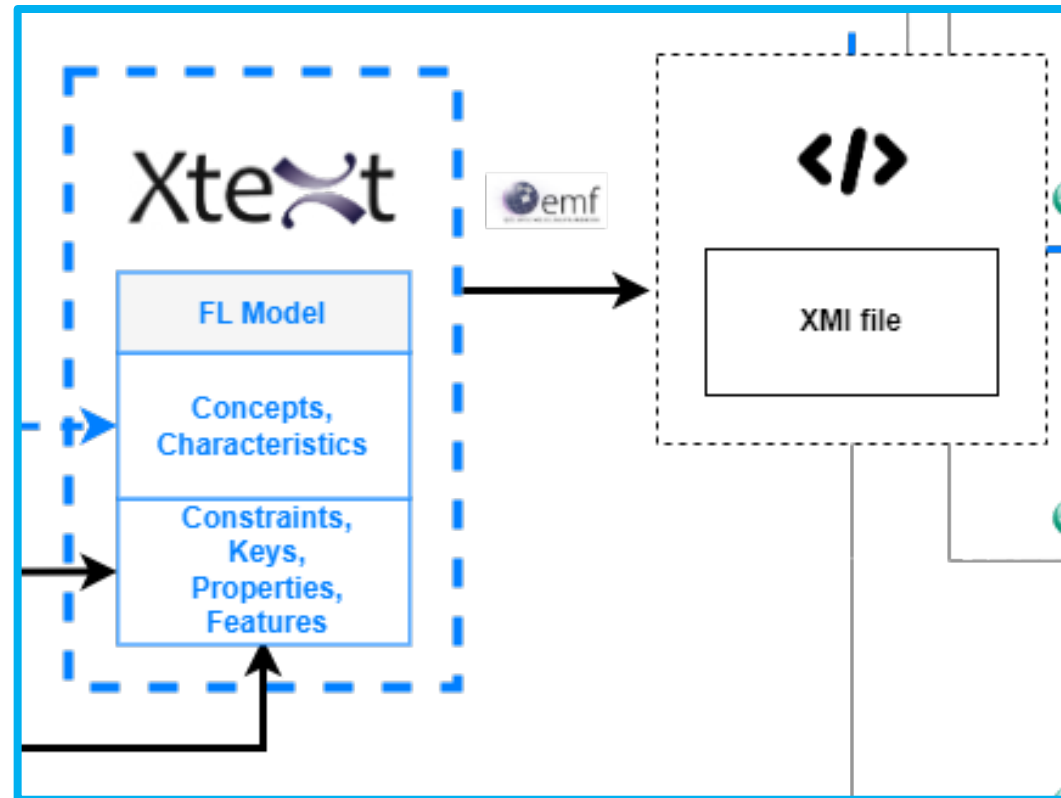
- Thick Black: Already Existed
- Dashed Blue: Current Implementation
- Thin Grey: Future Work

Project Info -> FeatureLanguage



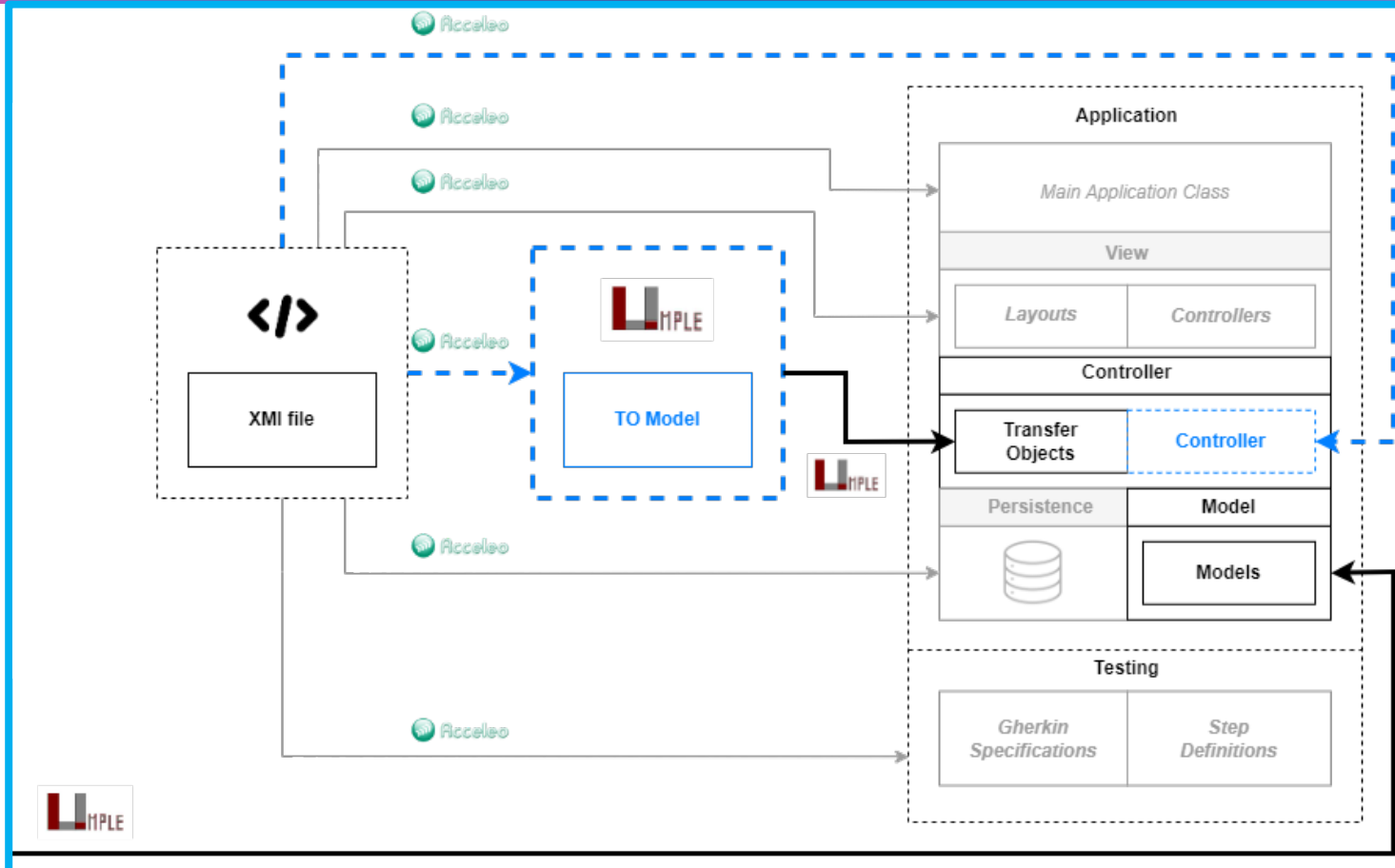
- Thick Black: Already Existed
- Dashed Blue: Current Implementation

FeatureLanguage -> XMI



- Thick Black: Already Existed
- Dashed Blue: Current Implementation

XMI -> MVC App



- Thick Black: Already Existed
- Dashed Blue: Current Implementation

- Thin Grey: Future Work

FeatureLanguage

- DSL
- Created using Xtext
- Sections
 - Concepts
 - Constraints
 - Keys
 - Properties
 - Features

```
1 concept MinimalRestoApp
2   Table tables 0..*
3   Order orders 0..*
4
5 concept Table
6   int number
7   int maxNumberSeats
8   MinimalRestoApp minimalRestoApp 1..1
9   Seat seats 0..*
10  Order orders 0..*
11  Location location { Indoors Patio }
12
13 concept Seat
14   boolean isArmChair
15   Table table 1..1
16
17 concept Order
18   Date date
19   Time time
20   int number
21   MinimalRestoApp minimalRestoApp 1..1
22   Table table 0..1
23
24 constraints
25   Table.number > 0 "The table number must be greater than 0"
26
27 keys
28   Table.number unique
29   Table.seats index
30   Order.number autounique
31
32 properties
33   MinimalRestoApp root
34   Table.maxNumberSeats lazy
35
36 features
37   Add Table
38   Remove Table
39   Display Table
40   Update Table
41   Add Table.seats
42   Remove Table.seats
43   Display Table.seats
44   Update Table.seats
45   Update Table.location
46   Add Order
47   Remove Order
48   Display Order
49   Update Order
```

Concepts

- Generated from domain model
- Textual representation of domain model
- Uml class = FL Concept
- Uml attribute = FL Characteristic
- Uml association = FL Characteristic
- Uml inheritance

Built-in Types:

- int
- byte
- short
- long
- float
- double
- boolean
- char
- String
- Date
- Time

```
1 concept MinimalRestoApp
2   Table tables 0..*
3   Order orders 0..*
4
5 concept Table
6   int number
7   int maxNumberSeats
8   MinimalRestoApp minimalRestoApp 1..1
9   Seat seats 0..*
10  Order orders 0..*
11  Location location { Indoors Patio }
12
13 concept Seat
14   boolean isArmChair
15   Table table 1..1
16
17 concept Order
18   Date date
19   Time time
20   int number
21   MinimalRestoApp minimalRestoApp 1..1
22   Table table 0..1
```

Constraints

```
24 constraints
25     Table.number > 0 "The table number must be greater than 0"
```

- Filled by hand from the project instructions
- Dot notation: `Concept.characteristic`
- Operators:
 - Less than `<`
 - Less than or equal `<=`
 - Equal `=`
 - Greater than or equal `>=`
 - Greater than `>`
- Value
- Error Message
- Limitations

Keys

- Filled in by hand using
 - Domain model information
 - Project instructions
- All Concepts except root need a Key

• Unique keys

• Autounique keys

• Index keys

27

28

29

30

31

keys

Table.number **unique**

Table.seats **index**

Order.number **autounique**

Properties


- Filled in by hand using:
 - Domain model information

- `ConceptProperty`

- Mandatory
- Declares root

- `CharacteristicProperty`

- Optional
- Defines `Characteristic` as lazy or immutable



```
32 properties  
33   MinimalRestoApp root  
34   Table.maxNumberSeats lazy
```

Features

- Filled in by hand using:
 - Project instructions
- Four types:
 - Add
 - Can act on Concepts
 - Can act on Characteristics representing associations
 - Remove
 - Can act on Concepts
 - Can act on Characteristics representing associations
 - Display
 - Can act on Concepts
 - Can act on Characteristics representing associations
 - Update
 - Can act on Concepts
 - Can act on Characteristics representing associations
 - Can act on Characteristics representing attributes

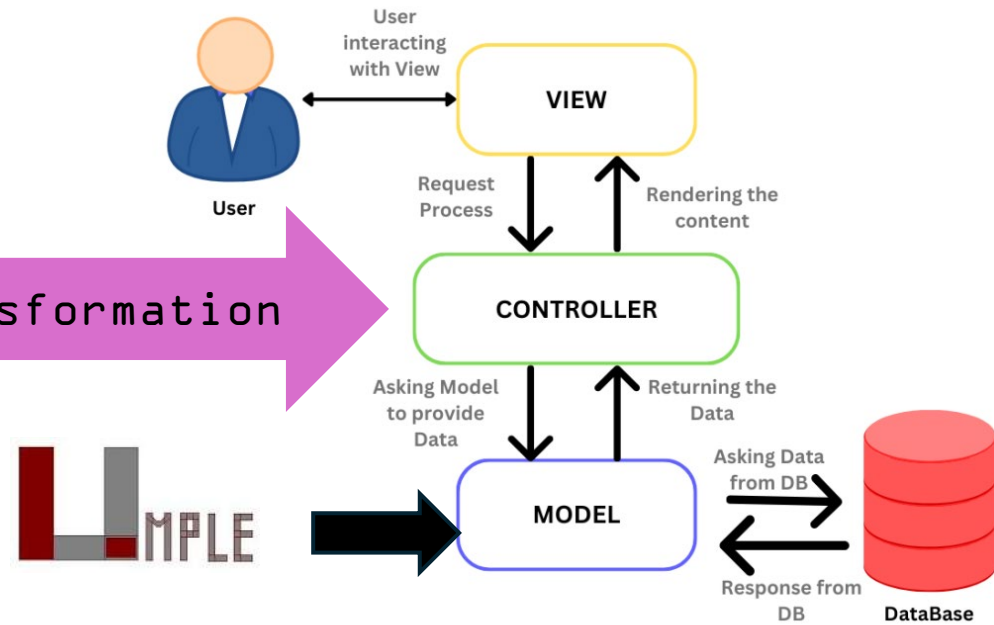
```
36 features
37   Add Table
38   Remove Table
39   Display Table
40   Update Table
41   Add Table.seats
42   Remove Table.seats
43   Display Table.seats
44   Update Table.seats
45   Update Table.location
46   Add Order
47   Remove Order
48   Display Order
49   Update Order
```

```

1 concept MinimalRestoApp
2   Table tables 0..*
3   Order orders 0..*
4
5 concept Table
6   int number
7   int maxNumberSeats
8   MinimalRestoApp minimalRestoApp 1..1
9   Seat seats 0..*
10  Order orders 0..*
11  Location location { Indoors Patio }
12
13 concept Seat
14  boolean isArmChair
15  Table table 1..1
16
17 concept Order
18  Date date
19  Time time
20  int number
21  MinimalRestoApp minimalRestoApp 1..1
22  Table table 0..1
23
24 constraints
25   Table.number > 0 "The table number must be greater than 0"
26
27 keys
28   Table.number unique
29   Table.seats index
30   Order.number autounique
31
32 properties
33   MinimalRestoApp root
34   Table.maxNumberSeats lazy
35
36 features
37   Add Table
38   Remove Table
39   Display Table
40   Update Table
41   Add Table.seats
42   Remove Table.seats
43   Display Table.seats
44   Update Table.seats
45   Update Table.location
46   Add Order
47   Remove Order
48   Display Order
49   Update Order

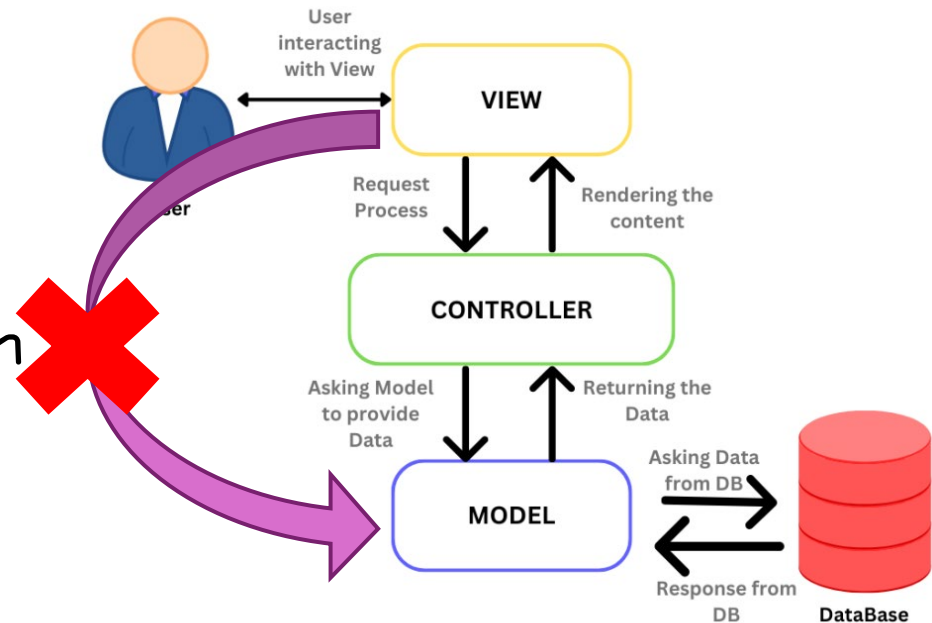
```

Acceleo transformation



Acceleo Transformation

- Uses
 - OCL queries
 - Templates
- Outputs
 - Controller class (.java)
 - Transfer Object generation file (.ump)



Controller Class - Headers

```
168 [template public fileHeader(aFeatureLanguage : FeatureLanguage)]
169 package ca.mcgill.[getRoot(aFeatureLanguage).toLowerCase()] controller;
170
171 [getImports(aFeatureLanguage)/]
172
173 public class [getRoot(aFeatureLanguage).toUpperCase().concat('Controller')/] {
174
175     public [getRoot(aFeatureLanguage).toUpperCase().concat('Controller')/]() {
176     }
177
178 [/template]
179
180 [template public getImports(aFeatureLanguage : FeatureLanguage)]
181 import ca.mcgill.[getRoot(aFeatureLanguage).toLowerCase()].application.[getRoot(aFeatureLanguage).toUpperCase()]Application;
182 import ca.mcgill.[getRoot(aFeatureLanguage).toLowerCase()].persistence.[getRoot(aFeatureLanguage).toUpperCase()]Persistence;
183 [for (aConcept : Concept | aFeatureLanguage.concepts)]
184 import ca.mcgill.[getRoot(aFeatureLanguage).toLowerCase()].model [aConcept.name]
185 [/for]
186 [/template]
```

```
1 package ca.mcgill.minimalrestoapp controller;
2
3 import ca.mcgill.minimalrestoapp.application.MinimalRestoAppApplication;
4 import ca.mcgill.minimalrestoapp.persistence.MinimalRestoAppPersistence;
5 import ca.mcgill.minimalrestoapp.model.MinimalRestoApp.*;
6 import ca.mcgill.minimalrestoapp.model.Table.*;
7 import ca.mcgill.minimalrestoapp.model.Seat.*;
8 import ca.mcgill.minimalrestoapp.model.Order.*;
9
10
11 public class MinimalRestoAppController {
12
13 public MinimalRestoAppController() {
14
15 }
16 }
```

```
4 [query public getRoot(aFeatureLanguage : FeatureLanguage) : String =
5     aFeatureLanguage.properties->any(x : Property | x.ocliIsTypeOf(ConceptProperty)).concept.name
6 /]
```

Controller Class - Feature Methods

- For each Feature => 1 Controller method
 - Different templates for
 - Different Feature types
 - Acting on Concepts of Characteristics
 - Special case: Display Concept.Characteristic
 - Where Characteristic's upper bound multiplicity > 1
 - Then 2 methods:
 - getCharacteristicFromConcept => gets 1 Characteristic
 - getCharacteristicsFromConcept => gets Characteristic list

Controller Class - Add Table Example

Uses:

- Feature
- Concept
- Characteristic
- ConceptProperty
- Key
- CharacteristicProperty

Uses:

- ConceptProperty

Uses:

- Constraint
- Concept
- Characteristic

Uses:

- Characteristic
- CharacteristicProperty
- Concept

```
16 public static String addTable(int number, String location) {
17
18     MinimalRestoApp root = MinimalRestoAppApplication.getMinimalRestoApp();
19
20     if (!(number > 0)) {
21         return "The table number must be greater than 0";
22     }
23
24     Location parsedLocation;
25     try {
26         parsedLocation = Location.valueOf(location);
27     }
28     catch (Exception e) {
29         return "The table location must be Indoors or Patio.";
30     }
31
32     try {
33         new Table(number, parsedLocation, root);
34     }
35     catch (RuntimeException e) {
36         return "The table number must be unique.";
37     }
38
39     try {
40         MinimalRestoAppPersistence.save();
41     }
42     catch (RuntimeException e) {
43         return e.getMessage();
44     }
45
46     return "";
47 }
```

Controller Class - Conversion Methods

- Convert Model Objects into Transfer Objects (TOs)
- 2 types of Conversion methods:
 - Convert 1 Model Object to 1 Transfer Object
 - Convert Model Object list to Transfer Object list
 - Only when Concept's upper bound multiplicity > 1

```
302 private static T0Table convertToT0Table(Table table) {
303
304     return new T0Table(
305         table.getNumber(),
306         table.getMaxNumberSeats(),
307         convertToT0Seat(table.getSeats());
308         convertToT0Order(table.getOrders());
309         table.getLocation()
310     );
311 }
```

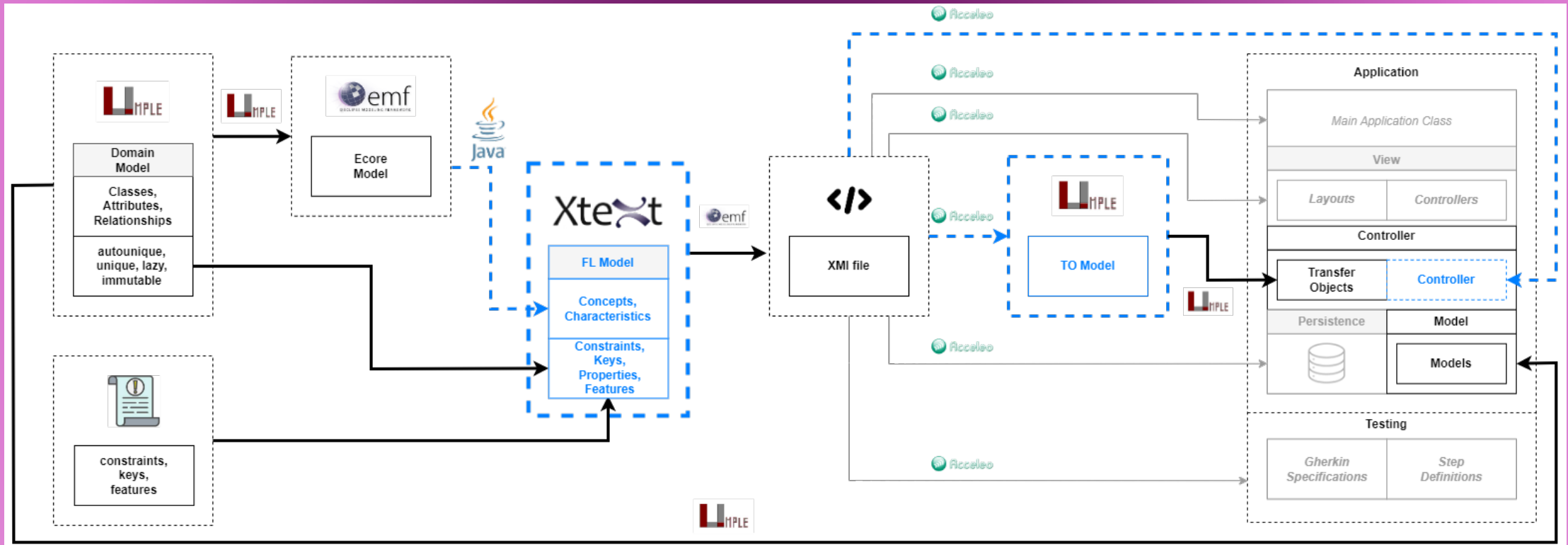
```
313 private static List<T0Table> convertToT0Table(List<Table> list) {
314     List<Table> list = new ArrayList<>();
315
316     for (Table element: list) {
317         list.add(convertToT0Table(element));
318     }
319
320     return list;
321 }
```

Transfer Object Uml File

- Can generate the TO classes from this file using Uml
- FL Concept => TO Class
 - Except for root
- Immutable
- All associations => directed associations

```
1 namespace ca.mcgill.minimalrestoapp.controller;
2
3 class TTable {
4     immutable;
5     Integer number;
6     String location;
7     Integer maxNumberSeats;
8     0..* -> 0..* TSeat seats;
9     0..* -> 0..* TOrder orders;
10}
11
12 class TSeat {
13     immutable;
14     Boolean isArmChair;
15     0..* -> 1..1 TTable table;
16}
17
18 class TOrder {
19     immutable;
20     Date date;
21     Time time;
22     Integer number;
23     0..* -> 0..1 TTable table;
24}
```

Transformation Pipeline



- Thick Black: Already Existed
- Dashed Blue: Current Implementation
- Thin Grey: Future Work

Future Work

- Replacing basic constraints by OCL constraints
- Generating Gherkin scenarios

```
1 Feature: Create Table
2 As the manager, I want to add a table in the system.
3
4 Background:
5   Given the following table exists in the system
6     | number | location |
7     |    1 | Indoors |
8
9 Scenario Outline: Successfully create a table
10  When the manager attempts to create a new table in the system with number "<number>" and location "<location>"
11  Then the number of tables in the system shall be "2"
12  Then the table "<number>" with location "<location>" shall exist in the system
13
14 Examples:
15   | number | location |
16   |    2 | Patio   |
17   |    3 | Indoors |
18
19 Scenario Outline: Unsuccessfully create a table
20  When the manager attempts to create a new table in the system with number "<number>" and location "<location>"
21  Then the number of tables in the system shall be "1"
22  Then the error "<error>" shall be raised
23
24 Examples:
25   | number | location | error |
26   |    1 | Patio   | The table number must be unique. |
27   |    2 | SecondFloor | The table location must be Indoors or Patio. |
28   |    0 | Indoors | The table number must be greater than 0. |
```




java

Xtext



FL Model

Concepts,
Characteristics

Constraints,
Keys,
Properties,
Features

***What else
would you like
to have
generated for a
course project?***



UML

```

@Acolleo
@Acolleo
@Acolleo
minimalRestoApp minimalRestoApp 1..1
table 0..1
> 0 "The table number must be greater than 0"
in the system with number "<number>" and location "<location>"
be "2"
"<location>" shall exist in the system
Examples:
| number | location |
| 2 | Patio |
| 3 | Indoors |
Scenario Outline: Unsuccessfully create a table
When the manager attempts to create a new table in the system with number "<numbers>" and location "<location>"
Then the number of tables in the system shall be "1"
Then the error "<error>" shall be raised
Examples:

```