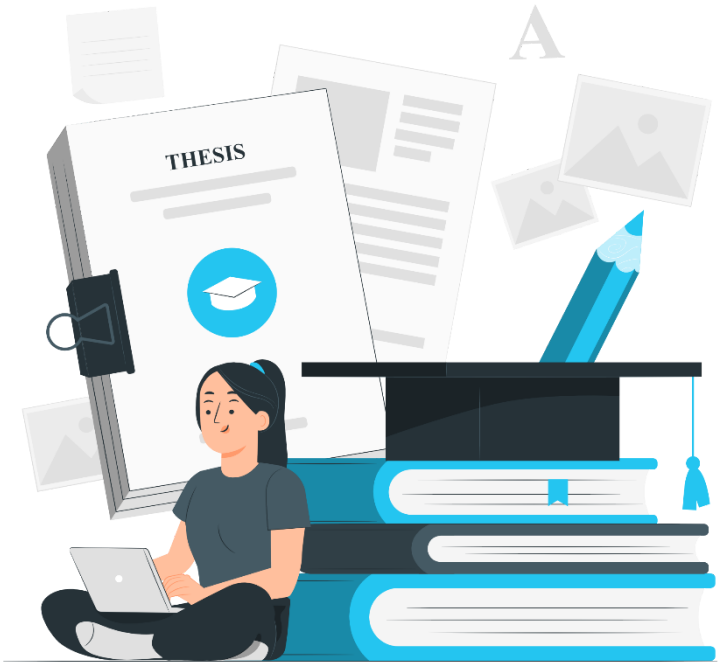
Engenharia Orientada a Modelos

Ano letivo 2023/24

Relatório de Projeto



Docente: Vasco Amaral

Dinis Silvestre 58763

Bruna Arroja 56751

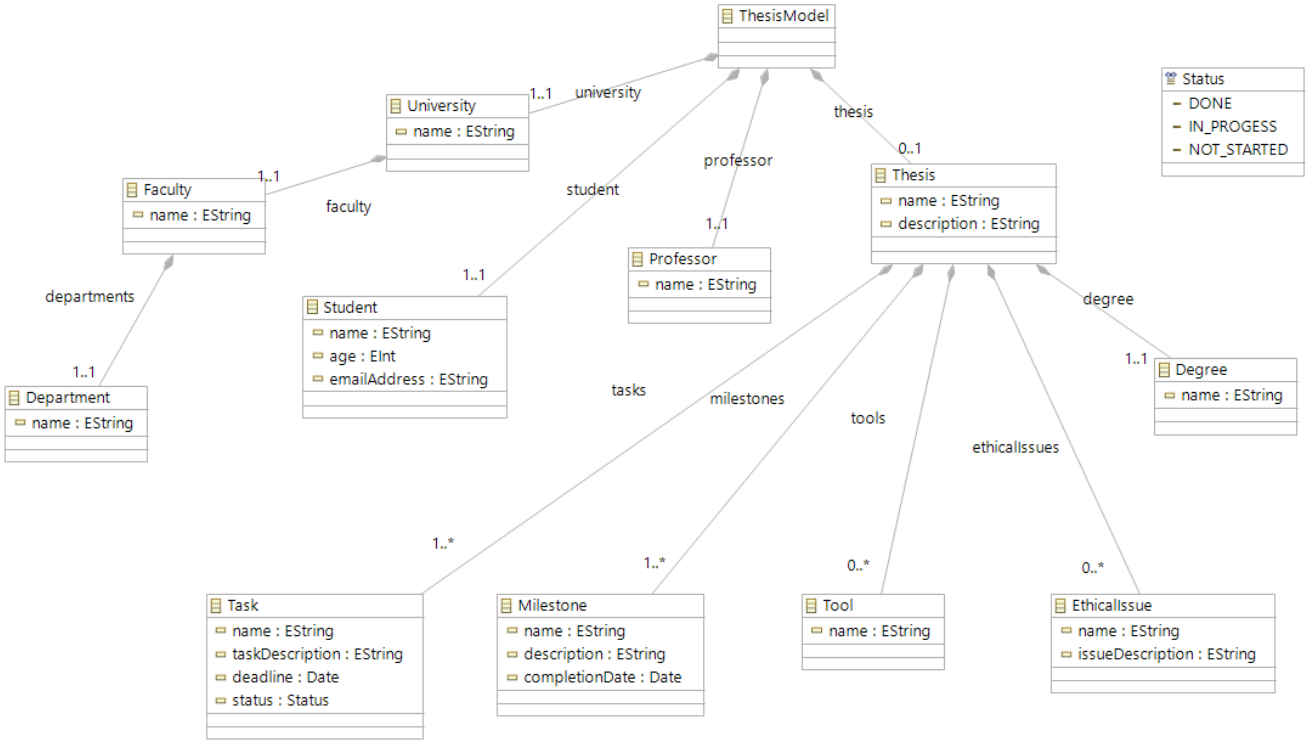
Francisca Corga 58218

Índice

Motivação

A dissertação de mestrado é uma jornada intelectual desafiadora que representa a culminação de esforços académicos. No entanto, enfrenta desafios, desde a definição da estrutura até à gestão de prazos. Diante dessa complexidade, propomos o Model-Driven Master Thesis Support System (MD-MTSS). Este sistema busca oferecer uma solução personalizada e eficiente, permitindo gerar a estrutura da dissertação apenas inserindo informações básicas como os dados do aluno, do professor orientador e da tese.

Metamodelo



Este metamodelo descreve um domínio relacionado com um sistema de gestão de teses. As entidades primárias neste metamodelo incluem "Task", "Milestone", "Student", "Tool", "EthicalIssue", "Faculty", "University", "Professor", "Thesis", "Status", "ThesisModel", "Department" e "Degree". Estas entidades estão interligadas através de várias características estruturais, tais como atributos e referências, formando um modelo abrangente para a gestão de uma tese.

As relações são estabelecidas através de referências entre entidades. Por exemplo, uma "Thesis" contém referências a "Degree", "Milestone", "EthicalIssue", "Task" e "Tool".

O metamodelo tem um tipo de enumeração "Status" para representar os diferentes estados em que uma tarefa pode estar, incluindo "DONE", "IN_PROGRESS" e "NOT_STARTED".

A estrutura global está organizada sob uma entidade "ThesisModel", que liga as entidades "University", "Thesis", "Professor" e "Student" de uma forma abrangente.

EVL

Para validar o modelo definimos as seguintes restrições:

- Contexto Task:

  TaskNameNotEmpty: A tarefa deve ter um nome não vazio.

  TaskDescriptionNotEmpty: A tarefa deve ter uma descrição não vazia.

  ValidTaskDeadline: A tarefa deve ter um prazo válido.

  UniqueTaskNames: Cada tarefa dentro de uma tese deve ter um nome único.

  ValidTaskStatus: O estado da tarefa é inconsistente com o prazo.

- Contexto Milestone:

  MilestoneNameNotEmpty: O marco deve ter um nome não vazio.

  ValidMilestoneCompletionDate: O marco deve ter uma data de conclusão válida.

  UniqueMilestoneNames: Cada marco dentro de uma tese deve ter um nome único.

  NonEmptyMilestoneDescription: A descrição do marco deve ser não vazia.

- Contexto Student:

  ValidEmailAddress: Endereço de e-mail inválido para o estudante.

  ValidAge: O estudante deve ter pelo menos 18 anos.

- Contexto University:

  UniversityNameNotEmpty: A universidade deve ter um nome não vazio.

- Contexto Thesis:

  ThesisHasMilestones: A tese deve ter pelo menos um marco.

  AtLeastOneTask: A tese deve ter pelo menos uma tarefa.

  OneDegree: A tese deve ter apenas um grau.

  ThesisDescriptionNotEmpty: A tese deve ter uma descrição não vazia.

- Contexto EthicalIssue:

  EthicalIssueNameNotEmpty: A questão ética deve ter um nome não vazio.

  ValidEthicalIssueType: O tipo e a descrição da questão ética devem ser especificados.

- Contexto Faculty:

  FacultyHasDepartments: A faculdade deve ter pelo menos um departamento.

  UniqueFacultyNames: Os nomes das faculdades devem ser únicos dentro da universidade.

- Contexto Professor:

  ProfessorNameNotEmpty: O professor deve ter um nome não vazio.

- Contexto Department:

  DepartmentNameNotEmpty: O departamento deve ter um nome não vazio.

- Contexto Tool:

  ToolNameNotEmpty: A ferramenta deve ter um nome não vazio.

- Contexto ThesisModel:

  ValidUniversityReference: O ThesisModel deve ter uma referência universitária válida.

  ValidThesisReference: O ThesisModel deve ter uma referência de tese válida.

  UniqueProfessorStudentCombination: O ThesisModel deve ter uma combinação única de professor e estudante.

  ThesisModelHasTools: A tese deve ter pelo menos uma ferramenta.

```
context Task {
 constraint TaskNameNotEmpty {
          check: self.name.size() > 0
          message: "Task must have a non-empty name."
           }
 constraint TaskDescriptionNotEmpty {
  check: self.taskDescription.size() > 0
  message: "Task must have a non-empty task description."
}
 constraint ValidTaskDeadline {
  check: self.deadline <> null
  message: "Task must have a valid deadline."
}
 constraint UniqueTaskNames {
  check: self.eContainer().tasks->select(t | t.name = self.name)->size() <= 1
  message: "Each task within a thesis must have a unique name."
```

```
 }
  constraint ValidTaskStatus {
      check: (self.status = "DONE" and self.deadline <= 2023-11-30) or
             (self.status <> "DONE" and self.deadline > 2023-11-30)
      message : "Task status is inconsistent with deadline"
   }
}
context Milestone {
 constraint MilestoneNameNotEmpty {
   check: self.name.size() > 0
   message: "Milestone must have a non-empty name."
 }
  constraint ValidMilestoneCompletionDate {
   check: self.completionDate <> null
   message: "Milestone must have a valid completion date."
 }
  constraint UniqueMilestoneNames {
   check: self.eContainer().milestones->select(m | m.name = self.name)->size() <= 1
   message: "Each milestone within a thesis must have a unique name."
 }
  constraint NonEmptyMilestoneDescription {
   check: not self.description.isEmpty()
   message: "The description of the milestone must be non-empty"
 }
}
context Student {
 constraint ValidEmailAddress {
   check: self.emailAddress.matches('[a-zA-Z0-9._%+\\-]+@[a-zA-Z0-9.\\-]+\\.[a-zA-Z]{2,}')
   message: "Invalid email address for the student."
 }
  constraint ValidAge {
   check: self.age >= 18
   message: "The student must have at least 18 years old"
 }
}
context University {
 constraint UniversityNameNotEmpty {
   check: self.name.size() > 0
   message: "University must have a non-empty name."
 }
}
context Thesis {
 constraint ThesisHasMilestones {
   check: self.milestones->size() > 0
   message: "Thesis must have at least one milestone."
 }
  constraint AtLeastOneTask {
   check: self.tasks->size() > 0
   message: "The thesis must have at least one task."
 }
  constraint OneDegree {
   check: self.degree->size() == 1
   message: "The thesis must have only one degree."
```

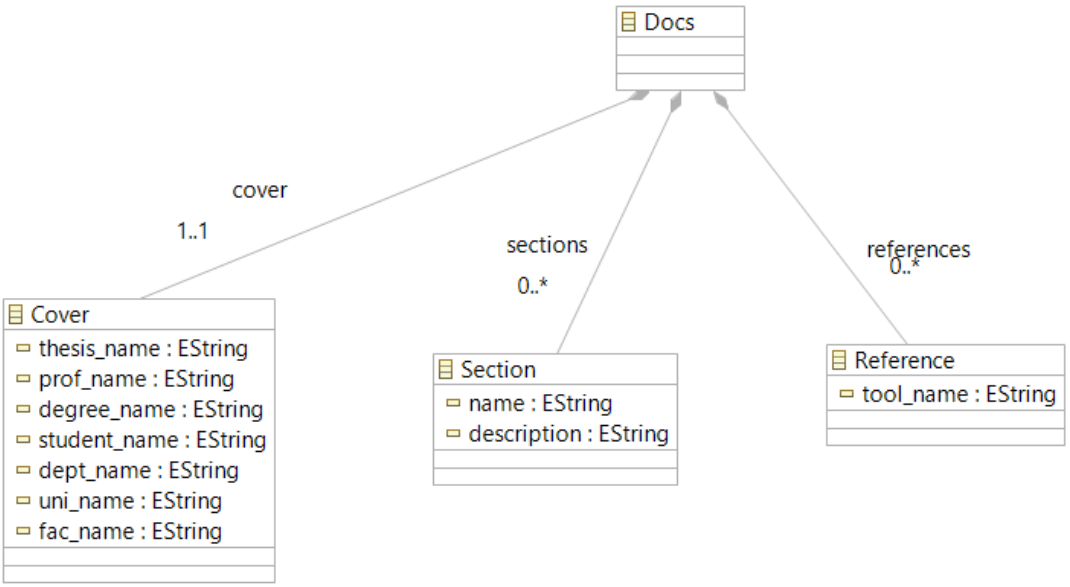
```
 }
  constraint ThesisDescriptionNotEmpty {
   check: self.description.size() > 0
   message: "Thesis must have a non-empty description."
  }
 }
 context EthicalIssue {
  constraint EthicalIssueNameNotEmpty {
   check: self.name.size() > 0
   message: "EthicalIssue must have a non-empty name."
  }
  constraint ValidEthicalIssueType {
   check: not self.name.isEmpty() and not self.issueDescription.isEmpty()
   message: "Ethical issue type and description must be specified."
  }
 }
 context Faculty {
  constraint FacultyHasDepartments {
   check: self.departments->size() > 0
   message: "Faculty must have at least one department."
  }
  constraint UniqueFacultyNames {
   check: self->select(f | f.name = self.name)->size() <= 1
   message: "Faculty names must be unique within the University."
  }
 }
 context Professor {
  constraint ProfessorNameNotEmpty {
   check: self.name.size() > 0
   message: "Professor must have a non-empty name."
  }
 }
 context Department {
  constraint DepartmentNameNotEmpty {
   check: self.name.size() > 0
   message: "Department must have a non-empty name."
  }
 }
 context Tool {
  constraint ToolNameNotEmpty {
   check: self.name.size() > 0
   message: "Tool must have a non-empty name."
  }
 }
 context ThesisModel {
  constraint ValidUniversityReference {
   check: self.university <> null
   message: "ThesisModel must have a valid university reference."
  }
  constraint ValidThesisReference {
   check: self.thesis <> null
   message: "ThesisModel must have a valid thesis reference."
  }
```

```
 constraint UniqueProfessorStudentCombination {
  check: ThesisModel.allInstances.select(t | t.professor = self.professor and t.student =
self.student)->size() <= 1
  message: "ThesisModel must have a unique combination of professor and student."
 }
 constraint ThesisModelHasTools {
  check: self.thesis.tools->size() > 0
  message: "Thesis must have at least one tool."
 }
}
```

PIM (Platform Independence Model)

O PIM cria uma camada de abstração, permitindo que o software seja independente do hardware ou software subjacente, facilitando a adaptação do software a ambientes diversos sem modificações significativas. Essa abstração promove a resiliência do software às mudanças tecnológicas e proporciona uma base sólida para a inovação contínua.

Docs



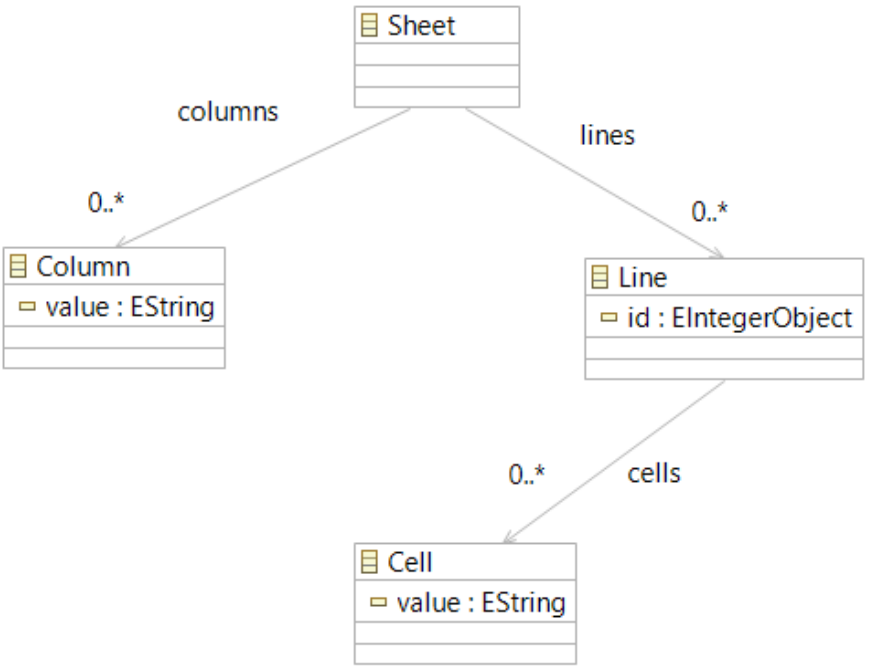
Sendo o PIM uma camada de abstração, foi feito um metamodelo para abstraír todo o tipo de documentos, sejam eles .pdf, .docx ou .tex.

Estes documentos seguem todos as mesmas especificações: uma capa com o nome da tese, professor, graduação, estudante, departamento, universidade e faculdade.

Após a capa, seguem-se várias secções, sendo cada uma delas representada por uma task, milestone ou ethicalIssue. As secções iniciais terão todas as tasks com a sua respetiva descrição seguidas por milestone e por último ethicalIssue.

Por último, as referências serão as tool names, como por exemplo o que foi usado para a criação do documento, como livros, artigos/dissertações pesquisados no Google scholar ou links de websites.

Sheets



Para além dos documentos, foi feito um metamodelo para abstraír todo o tipo de ficheiros do formato spreadsheets, sejam eles .xlsm, .xls ou xlsx.

Estes ficheiros spreadsheets seguem todos as mesmas especificações: existem várias colunas (neste caso nome, descrição, taskDeadline e status - todos os atributos das tasks).

Existem também várias linhas, representadas cada uma por cada task. Ora, cada linha tem várias células, sendo uma célula representada por uma linha e coluna associadas à mesma.

Por exemplo, uma célula na posição A1, irá ter como value o nome da primeira task, enquanto uma célula na posição B3 irá ter como value a taskDeadline da terceira task.

ETL

- Docs

Ao receber como input instâncias criadas no ThesisModel.xmi, transforma-as em instâncias DocsModel.xmi representadas pelo novo metamodelo Docs.

Ficheiro: model2docs.etl

```
rule ThesisModel2Docs
  transform t : input!ThesisModel
  to ds : output!Docs {

        // Create the Cover instance
  var c = new output!Cover;
  c.thesis_name = t.thesis.name;
        c.prof_name = t.professor.name;
        c.degree_name = t.thesis.degree.name;
        c.student_name = t.student.name;
        c.dept_name = t.university.faculty.departments.name;
        c.fac_name = t.university.faculty.name;
        c.uni_name = t.university.name;
        ds.cover = c;
        for (task in t.thesis.tasks) {
    var s = new output!Section;
    s.name = task.name;
    s.description = task.taskDescription;
    ds.sections.add(s);
  }
  for (milestone in t.thesis.milestones) {
    var s = new output!Section;
    s.name = milestone.name;
    s.description = milestone.description;
    ds.sections.add(s);
  }
  for (et in t.thesis.ethicalIssues) {
    var s = new output!Section;
    s.name = et.name;
    s.description = et.issueDescription;
    ds.sections.add(s);
  }
   for (tool in t.thesis.tools) {
    var s = new output!Reference;
    s.tool_name = tool.name;
    ds.references.add(s);
  }
}
```

- Sheets

Ao receber como input instâncias criadas no ThesisModel.xmi, transforma-as em instâncias SheetsModel.xmi representadas pelo novo metamodelo Sheets.

Ficheiro: model2Sheets

```
rule Task2Column
```

```
transform t : input!ThesisModel
to ds : output!Sheet {
var s1 = new output!Column;
s1.value = "name";
ds.columns.add(s1);
var s2 = new output!Column;
s2.value = "description";
ds.columns.add(s2);
var s3 = new output!Column;
s3.value = "taskDeadline";
ds.columns.add(s3);
var s4 = new output!Column;
s4.value = "status";
ds.columns.add(s4);
var idCounter = 1;
for (task in t.thesis.tasks) {
    var t = new output!Line;
    t.id = idCounter;
    ds.lines.add(t);
    for (column in ds.columns) {
        var s = new output!Cell;
        if (column.value == "name") {
            s.value = task.name;
            if (task.name != null)
                t.cells.add(s);
        }
        else if (column.value == "description") {
            s.value = task.taskDescription;
            if (task.taskDescription != null)
                t.cells.add(s);
        }
        else if (column.value == "taskDeadline") {
            if (task.deadline != null) {
                s.value = task.deadline.toString();
                t.cells.add(s);
            }
            else
                s.value = task.deadline;
        }
        else if (column.value == "status") {
            if (task.status != null) {
                s.value = task.status.toString();
                t.cells.add(s);
            }
            else {
                s.value = task.status;
                t.cells.add(s);
            }
        }
    }
    idCounter++;
  }
}
```

EGL

- Código gerado para latex; (Ficheiro: modelToLatex.egl)
- Código gerado para excel sheets.
    - macros (Ficheiro: modelToSheetsMacros.egl)
    - xml (Ficheiro: modelToSheetsXML.egl)
    - gantt chart (Ficheiro: modelToSheetsGantt.egl)

modelToLatex

Ao receber como input instâncias do SheetsModel.xmi, transforma-as num documento LaTeX.

```
\documentclass{article}
\usepackage{graphicx}
\usepackage{hyperref}
\begin{document}
% Capa
[% for (Docs in Docs.allInstances()) { %]
\begin{titlepage}
\centering
\Large\textbf{[%=Docs.cover.student_name%]}
\vspace{0.5cm}
\\
\Large\textbf{Supervisor: [%=Docs.cover.prof_name%]}
\vspace{0.5cm}
\\
\Large\textbf{[%=Docs.cover.degree_name%]}
\\
\vspace{7cm}
\Huge\textbf{[%=Docs.cover.thesis_name%]}
\vspace{7cm}
\\
\Large\textbf{[%=Docs.cover.dept_name%]} ,
\Large\textbf{[%=Docs.cover.uni_name%]}
\\
\vspace{0.2cm}
\Large\textbf{\today}
\end{titlepage}
% Índice
\tableofcontents
\newpage
 \section{Tasks, Milestones and Ethical Issues}
 \begin{sectionPage}
   [% for (section in Docs.sections) { %]
   \Large\textbf{Name: }[%=section.name%]}
   \\
   \Large\textbf{Description: }[%=section.description%]
   \\
   \\
   [%}%]
```

```
\end{sectionPage}
        % Conclusão e Referencias
        \section{Conclusion}
        \begin{conc}
 [% for (ref in Docs.references) { %]
 \Large\textbf{Name: }[%=ref.tool_name%]
 \\
 \\
 [%}%]
\end{conc}
\end{document}
[%}%]
[%
 var t : Template = TemplateFactory.load("modelToLatex.egl");
 t.generate("latex.tex");
%]
```

modelToSheetsMacros

Ao receber como input instâncias do SheetsModel.xmi, transforma-as num ficheiro Macros().

```
Sub Macro5()
'
' Macro5 Macro
'
'
  ActiveCell.FormulaR1C1 = "Name"
  With Selection
    .HorizontalAlignment = xlCenter
    .VerticalAlignment = xlBottom
    .WrapText = False
    .Orientation = 0
    .AddIndent = False
    .IndentLevel = 0
    .ShrinkToFit = False
    .ReadingOrder = xlContext
    .MergeCells = False
  End With
  With Selection
    .HorizontalAlignment = xlCenter
    .VerticalAlignment = xlCenter
    .WrapText = False
    .Orientation = 0
    .AddIndent = False
    .IndentLevel = 0
    .ShrinkToFit = False
    .ReadingOrder = xlContext
    .MergeCells = False
  End With
  ActiveCell.Offset(0, 1).Range("A1").Select
  ActiveCell.FormulaR1C1 = "Description"
  With Selection
```

```
    .HorizontalAlignment = xlCenter
    .VerticalAlignment = xlBottom
    .WrapText = False
    .Orientation = 0
    .AddIndent = False
    .IndentLevel = 0
    .ShrinkToFit = False
    .ReadingOrder = xlContext
    .MergeCells = False
End With
With Selection
    .HorizontalAlignment = xlCenter
    .VerticalAlignment = xlCenter
    .WrapText = False
    .Orientation = 0
    .AddIndent = False
    .IndentLevel = 0
    .ShrinkToFit = False
    .ReadingOrder = xlContext
    .MergeCells = False
End With
ActiveCell.Offset(0, 1).Range("A1").Select
ActiveCell.FormulaR1C1 = "taskDeadline"
With Selection
    .HorizontalAlignment = xlCenter
    .VerticalAlignment = xlBottom
    .WrapText = False
    .Orientation = 0
    .AddIndent = False
    .IndentLevel = 0
    .ShrinkToFit = False
    .ReadingOrder = xlContext
    .MergeCells = False
End With
With Selection
    .HorizontalAlignment = xlCenter
    .VerticalAlignment = xlCenter
    .WrapText = False
    .Orientation = 0
    .AddIndent = False
    .IndentLevel = 0
    .ShrinkToFit = False
    .ReadingOrder = xlContext
    .MergeCells = False
End With
ActiveCell.Offset(0, 1).Range("A1").Select
ActiveCell.FormulaR1C1 = "status"
With Selection
    .HorizontalAlignment = xlCenter
    .VerticalAlignment = xlBottom
    .WrapText = False
    .Orientation = 0
    .AddIndent = False
```

```
    .IndentLevel = 0
    .ShrinkToFit = False
    .ReadingOrder = xlContext
    .MergeCells = False
  End With
  With Selection
    .HorizontalAlignment = xlCenter
    .VerticalAlignment = xlCenter
    .WrapText = False
    .Orientation = 0
    .AddIndent = False
    .IndentLevel = 0
    .ShrinkToFit = False
    .ReadingOrder = xlContext
    .MergeCells = False
  End With
  ActiveCell.Offset(1, -3).Range("A1").Select
  [% for (SheetsModel in Sheet.allInstances()) { %]
  [% for (line in SheetsModel.lines) { %]
        ActiveCell.FormulaR1C1 = "[% if (line.cells.size() > 0 and line.cells.at(0) != null) {
%][%=line.cells.at(0).value%] [%}%][% else { %]null[%}%]"
        With Selection
    .HorizontalAlignment = xlCenter
    .VerticalAlignment = xlBottom
    .WrapText = False
    .Orientation = 0
    .AddIndent = False
    .IndentLevel = 0
    .ShrinkToFit = False
    .ReadingOrder = xlContext
    .MergeCells = False
  End With
  With Selection
    .HorizontalAlignment = xlCenter
    .VerticalAlignment = xlCenter
    .WrapText = False
    .Orientation = 0
    .AddIndent = False
    .IndentLevel = 0
    .ShrinkToFit = False
    .ReadingOrder = xlContext
    .MergeCells = False
  End With
        ActiveCell.Offset(0, 1).Range("A1").Select
        ActiveCell.FormulaR1C1 = "[% if (line.cells.size() > 1 and line.cells.at(1) != null) {
%][%=line.cells.at(1).value%] [%}%][% else { %]null[%}%]"
        With Selection
    .HorizontalAlignment = xlCenter
    .VerticalAlignment = xlBottom
    .WrapText = False
    .Orientation = 0
    .AddIndent = False
    .IndentLevel = 0
```

```
    .ShrinkToFit = False
    .ReadingOrder = xlContext
    .MergeCells = False
  End With
  With Selection
    .HorizontalAlignment = xlCenter
    .VerticalAlignment = xlCenter
    .WrapText = False
    .Orientation = 0
    .AddIndent = False
    .IndentLevel = 0
    .ShrinkToFit = False
    .ReadingOrder = xlContext
    .MergeCells = False
  End With
        ActiveCell.Offset(0, 1).Range("A1").Select
        ActiveCell.FormulaR1C1 = "[% if (line.cells.size() > 2 and line.cells.at(2) != null) {
%][%=line.cells.at(2).value%] [%}%][% else { %]null[%}%]"
        With Selection
    .HorizontalAlignment = xlCenter
    .VerticalAlignment = xlBottom
    .WrapText = False
    .Orientation = 0
    .AddIndent = False
    .IndentLevel = 0
    .ShrinkToFit = False
    .ReadingOrder = xlContext
    .MergeCells = False
  End With
  With Selection
    .HorizontalAlignment = xlCenter
    .VerticalAlignment = xlCenter
    .WrapText = False
    .Orientation = 0
    .AddIndent = False
    .IndentLevel = 0
    .ShrinkToFit = False
    .ReadingOrder = xlContext
    .MergeCells = False
  End With
        ActiveCell.Offset(0, 1).Range("A1").Select
        ActiveCell.FormulaR1C1 = "[% if (line.cells.size() > 3 and line.cells.at(3) != null) {
%][%=line.cells.at(3).value%] [%}%][% else { %] null [%}%]"
        With Selection
    .HorizontalAlignment = xlCenter
    .VerticalAlignment = xlBottom
    .WrapText = False
    .Orientation = 0
    .AddIndent = False
    .IndentLevel = 0
    .ShrinkToFit = False
    .ReadingOrder = xlContext
    .MergeCells = False
```

```
  End With
  With Selection
    .HorizontalAlignment = xlCenter
    .VerticalAlignment = xlCenter
    .WrapText = False
    .Orientation = 0
    .AddIndent = False
    .IndentLevel = 0
    .ShrinkToFit = False
    .ReadingOrder = xlContext
    .MergeCells = False
  End With
        ActiveCell.Offset(1, -3).Range("A1").Select
  [%}%]
  [%}%]
End Sub
[%
 var t : Template = TemplateFactory.load("modelToSheetsMacros.egl");
 t.generate("sheetsMacros.xlsm");
%]
```

Quando corremos o ficheiro gerado Macro, este é output que obtemos no Excel.

	A	B	C	D
1	Name	Description	taskDeadline	status
2	Literature Review	This is the description of the literature review	01/11/2023	DONE
3	Research	This is the description of the research	21/11/2023	IN_PROGESS
4	Data Collection	This is the description of the Data Collection	20/12/2023	NOT_STARTED
5				

modelToSheetsXML

Ao receber como input instâncias do SheetsModel.xmi, transforma-as num ficheiro XML.

```
<?xml version="1.0" encoding="UTF-8" ?>
<Table>
  [% for (SheetsModel in Sheet.allInstances()) { %]
  [% for (line in SheetsModel.lines) { %]
  <Row>
      <Name>[% if (line.cells.size() > 0 and line.cells.at(0) != null) { %][%=line.cells.at(0).value%] [%}%][%
else { %] null [%}%]</Name>
    <Description>[% if (line.cells.size() > 1 and line.cells.at(1) != null) { %][%=line.cells.at(1).value%]
[%}%][% else { %] null [%}%]</Description>
    <taskDeadline>[% if (line.cells.size() > 2 and line.cells.at(2) != null) { %][%=line.cells.at(2).value%]
[%}%][% else { %] null [%}%] </taskDeadline>
    <status>[% if (line.cells.size() > 3 and line.cells.at(3) != null) { %][%=line.cells.at(3).value%] [%}%][%
else { %] null [%}%]</status>
  </Row>
  [%}%]
  [%}%]
</Table>
```

```
[%
 var t : Template = TemplateFactory.load("modelToSheetsXML.egl");
 t.generate("sheets.xml");
%]
```

Quando corremos o ficheiro gerado XML, este é output que obtemos no Excel.

	A	B	C	D	E
1	Name	Description	taskDeadline	status	
2	Literature Review	This is the description of the literature review	01/11/2023	DONE	
3	Research	This is the description of the research	21/11/2023	IN_PROGESS	
4	Data Collection	This is the description of the Data Collection	20/12/2023	NOT_STARTED	
5					

modelToSheetsGantt

Ao receber como input instâncias do SheetsModel.xmi, transforma-as num gráfico estilo Gantt Chart no Excel. Isto é obtido através do Macro.

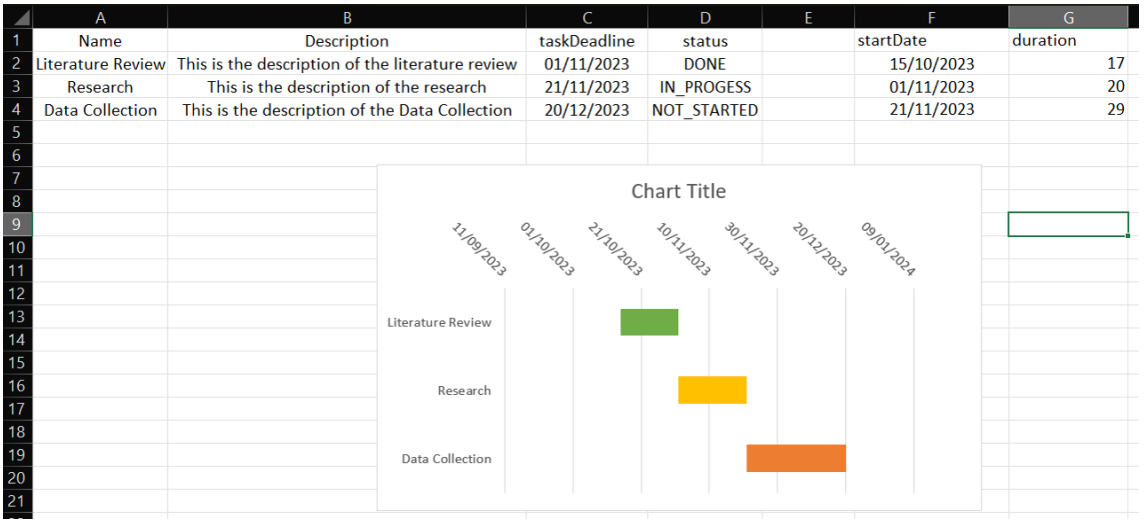
```
Sub Macro7()
'
' Macro7 Macro
'
'
  Application.CutCopyMode = False
  With Selection
     .HorizontalAlignment = xlCenter
     .VerticalAlignment = xlCenter
     .WrapText = False
     .Orientation = 0
     .AddIndent = False
     .ShrinkToFit = False
     .ReadingOrder = xlContext
     .MergeCells = False
  End With
  ActiveCell.FormulaR1C1 = "startDate"
  Range("G1").Select
  With Selection
     .HorizontalAlignment = xlCenter
     .VerticalAlignment = xlCenter
     .WrapText = False
     .Orientation = 0
     .AddIndent = False
     .ShrinkToFit = False
     .ReadingOrder = xlContext
     .MergeCells = False
  End With
  ActiveCell.FormulaR1C1 = "duration"
  Range("F2").Select
  ActiveCell.FormulaR1C1 = "10/15/2023"
  Range("G2").Select
  Application.CutCopyMode = False
  ActiveCell.FormulaR1C1 = "=RC[-4]-RC[-1]"
  [% for (SheetsModel in Sheet.allInstances()) { %]
  [% var rowNum = 3; %]
          [% var countNum = 2; %]
          [% var dateI = 2; %]
          [% var counter = 1; %]
          [% var counterLines = 1; %]
          [% var range = 1; %]
  [% for (line in SheetsModel.lines) { %]
  [% if (counter < SheetsModel.lines.size()) { %]
  Range("F[%=rowNum%]").Select
  ActiveCell.FormulaR1C1 = "[% if (line.cells.size() > 2 and line.cells.at(2) != null) { %]
  [%=line.cells.at(2).value%] [%}%][% else { %]null[%}%]"
```

```
    Selection.NumberFormat = "m/d/yyyy"
    Range("G[%=rowNum%]").Select
    Application.CutCopyMode = False
    ActiveCell.FormulaR1C1 = "=RC[-4]-RC[-1]"
    [% rowNum = rowNum + 1; %]
    [% counter = counter + 1; %]
    [%}%]
    [%}%]
    [% for (line in SheetsModel.lines) { %]
    Range("F[%=countNum%]").Select
    Selection.NumberFormat = "General"
    [% countNum = countNum + 1; %]
    [% counterLines = counterLines + 1; %]
    [%}%]
    Range("A1:A[%=counterLines%],F1:G[%=counterLines%]").Select
    Range("F1").Activate
    ActiveSheet.Shapes.AddChart2(297, xlBarStacked).Select
    ActiveChart.SetSourceData
Source:=Range("Row!$A$1:$A$[%=counterLines%],Row!$F$1:$G$[%=counterLines%]")
    ActiveChart.FullSeriesCollection(1).Select
    ActiveChart.FullSeriesCollection(1).Points(3).Select
    ActiveChart.ChartArea.Select
    ActiveChart.FullSeriesCollection(1).Select
    ActiveChart.Axes(xlValue).Select
    Selection.Format.Fill.Visible = msoFalse
    [% for (line in SheetsModel.lines) { %]
    Range("F[%=dateI%]").Select
    Selection.NumberFormat = "m/d/yyyy"
    [% dateI = dateI + 1; %]
    [%}%]
    ActiveSheet.ChartObjects("Chart 1").Activate
    ActiveChart.Axes(xlCategory).Select
    ActiveChart.Axes(xlCategory).ReversePlotOrder = True
    ActiveChart.Legend.Select
    Selection.Delete
    ActiveSheet.ChartObjects("Chart 1").Activate
    ActiveChart.Axes(xlValue).Select
    Selection.TickLabels.Orientation = -45
    ActiveChart.ChartTitle.Select
    ActiveChart.ChartTitle.Text = "Gantt chart"
    Selection.Format.TextFrame2.TextRange.Characters.Text = "Gantt chart"
    With Selection.Format.TextFrame2.TextRange.Characters(1, 11).ParagraphFormat
        .TextDirection = msoTextDirectionLeftToRight
        .Alignment = msoAlignCenter
    End With
    With Selection.Format.TextFrame2.TextRange.Characters(1, 11).Font
        .BaselineOffset = 0
        .Bold = msoFalse
        .NameComplexScript = "+mn-cs"
        .NameFarEast = "+mn-ea"
        .Fill.Visible = msoTrue
        .Fill.ForeColor.RGB = RGB(89, 89, 89)
        .Fill.Transparency = 0
```

```
    .Fill.Solid
    .Size = 14
    .Italic = msoFalse
    .Kerning = 12
    .Name = "+mn-lt"
    .UnderlineStyle = msoNoUnderline
    .Spacing = 0
    .Strike = msoNoStrike
  End With
  ActiveChart.FullSeriesCollection(2).Select
  [% for (line in SheetsModel.lines) { %]
  ActiveChart.FullSeriesCollection(2).Points([%=range%]).Select
  With Selection.Format.Fill
    .Visible = msoTrue
    .ForeColor.ObjectThemeColor = msoThemeColorAccent4
    .ForeColor.TintAndShade = 0
    .ForeColor.Brightness = 0
    .Transparency = 0
    .Solid
  End With
  [% range = range + 1; %]
  [%}%]
  Range("I19").Select
          [%}%]

End Sub
```

Quando corremos o ficheiro gerado Macro, este é output que obtemos no Excel - é criada uma nova tabela à direita com a start date e a duração em dias. A data 15/10/2023 é a data de início do projeto e assumimos que, mal acabe uma task, uma nova é criada. As cores diferentes representam se a task está concluída, em progresso ou por fazer.



	A	B	C	D	E	F	G
1	Name	Description	taskDeadline	status		startDate	duration
2	Literature Review	This is the description of the literature review	01/11/2023	DONE		15/10/2023	17
3	Research	This is the description of the research	21/11/2023	IN_PROGESS		01/11/2023	20
4	Data Collection	This is the description of the Data Collection	20/12/2023	NOT_STARTED		21/11/2023	29

RESTful APIs

- Automation of tasks - **Trello**

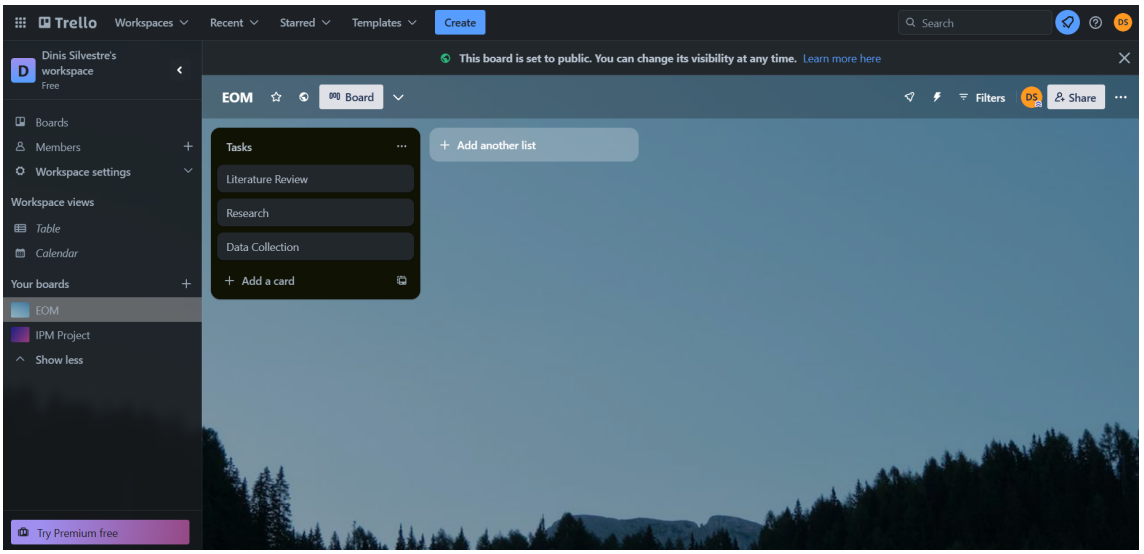
Ao receber como input instâncias do ThesisModel.xmi, transforma-as num código Python em que é usado uma API e Token IDs para fazer um pedido de criação de um cartão ao Trello.

```
import requests
import json
def create_card(api_key, api_token, list_id, names, board_id):
  url = "https://api.trello.com/1/cards"
  headers = {
    "Accept": "application/json"
  }
  for task_name in names:
    query = {
      'key': api_key,
      'token': api_token,
      'idBoard': board_id,
      'idList': list_id,
      'name': task_name
    }
    response = requests.post(url, headers=headers, params=query)
    print("Creating task:", task_name)
    print("Response Code:", response.status_code)
    print("Response Body:", response.text)  # Print the raw response text
    try:
      print("Parsed Response Body:", json.dumps(response.json(), sort_keys=True, indent=4,
separators=(",", ": ")))
    except json.decoder.JSONDecodeError:
      print("Unable to parse response body as JSON.")
if _name_ == "_main_":
  api_key = "23face52cb51932bb6eb8a8b187b0a73"  # Replace with your actual API key
  api_token =
"ATTAc75512c90fc22df7b4ae5fd79d67d9ecab4441a70a1e625f156e873f8cb4abcb80B07654"
# Replace with your actual API token
  list_id = "656927a76fc7953fecc37331"  # Replace with your actual list ID
  board_id = "rNksPOTU"
  [% for (ThesisModel in ThesisModel.allInstances()) { %]
  task_names = []  # Create an empty list to store task names

  # Add task names to the list dynamically
  [% for (task in ThesisModel.thesis.tasks) { %]
  task_names.append("[%=task.name%]")
  [%}%]
  [%}%]

  create_card(api_key, api_token, list_id, task_names, board_id)
[%
 var t : Template = TemplateFactory.load("API_modelToTrello.egl");
 t.generate("trello.py");
%]
```

Mal é feito o pedido, cada task vai sendo adicionada à lista Tasks. Cada task representa cada instância task presente no ThesisModel.xmi.



- Collaboration tools - **Slack**

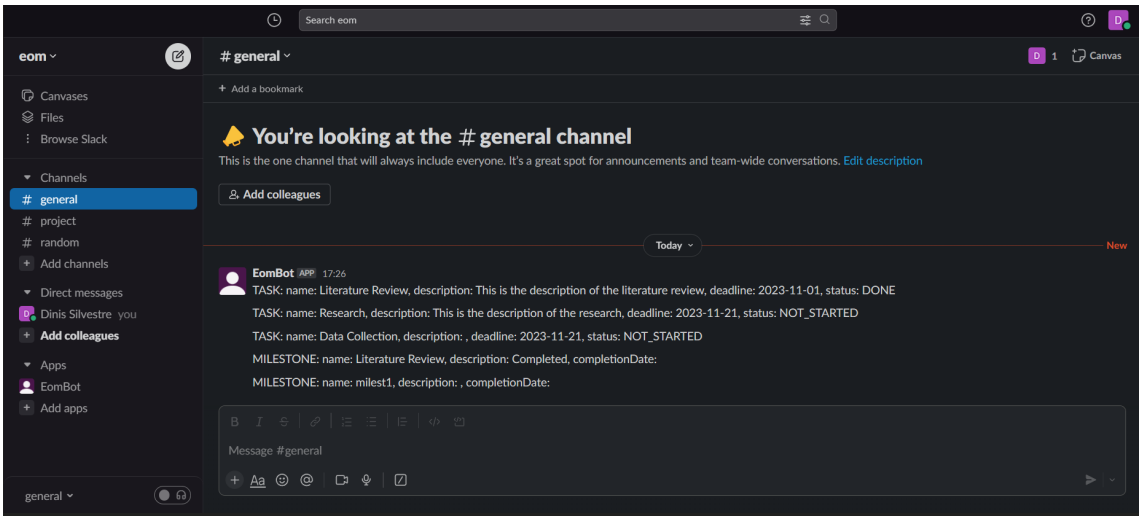
Ao receber como input instâncias do ThesisModel.xmi, transforma-as num código Python em que é usado um Token e Channel IDs para fazer um pedido de mensagem ao Slack.

```
import requests
import json
# Replace these values with your actual Slack token and channel ID
slack_token = 'xoxb-6264888934199-6276668934437-sp8Y21jz7Eo23iOcSWwpd2cw'
channel_id = 'C06879FFTK5'
# API endpoint for posting messages
api_url = 'https://slack.com/api/chat.postMessage'
[% for (ThesisModel in ThesisModel.allInstances()) { %]
# List of messages to send
messages = []  # Create an empty list to store task names

[% for (task in ThesisModel.thesis.tasks) { %]
messages.append("TASK: name: [%=task.name%], description:
[%=task.taskDescription%], deadline: [%=task.deadline%], status: [%=task.status%]")
[%}%]
[% for (milestone in ThesisModel.thesis.milestones) { %]
messages.append("MILESTONE: name: [%=milestone.name%], description:
[%=milestone.description%], completionDate: [%=milestone.completionDate%]")
[%}%]
# Set the authorization header with your Slack token
headers = {
  'Content-Type': 'application/json',
```

```
  'Authorization': f'Bearer {slack_token}',
}
# Iterate over the messages and make the API request to post each message
for message_text in messages:
  # Message data
  message_data = {
    'channel': channel_id,
    'text': message_text,
  }
  # Make the API request to post the message
  response = requests.post(api_url, headers=headers, data=json.dumps(message_data))
  print(response.content)
  # Check the response status for each message
  if response.status_code == 200:
    print(f'Message posted successfully: {message_text}')
  else:
    print(f'Error posting message. Status code: {response.status_code}, Response:
{response.text}')
[%}%]
[%
 var t : Template = TemplateFactory.load("API_modelToSlack.egl");
 t.generate("slack.py");
%]
```

Mal é feito o pedido, as mensagens vão sendo enviadas para o canal *general* (primeiro as tasks e depois as milestones) com os seus respetivos atributos.



Conclusão

O desenvolvimento do Model-Driven Master Thesis Support System (MD-MTSS) representa um avanço significativo na simplificação e personalização do processo de elaboração de dissertações de mestrado. A introdução do metamodelo, juntamente com as regras definidas em EVL (Epsilon Validation Language), contribui para garantir a consistência e validade do modelo. A implementação do Platform Independence Model (PIM) proporciona uma camada de abstração, promovendo a independência do software em relação ao ambiente subjacente.

A geração de documentos e planilhas, utilizando EGL (Epsilon Generation Language) e ETL (Epsilon Transformation Language), demonstra a versatilidade do sistema ao adaptar-se a diferentes formatos, como LaTeX e folhas de cálculo. A automatização de tarefas, tanto na geração de documentos quanto na integração com a plataforma Trello por meio de RESTful APIs, destaca a eficiência e a integração do MD-MTSS com ferramentas externas.

Em resumo, o MD-MTSS não apenas simplifica o processo de elaboração de dissertações, mas também oferece flexibilidade e interoperabilidade, proporcionando uma abordagem inovadora e eficaz para os desafios enfrentados durante a jornada académica de mestrado.