DOCUMENTATION DU PROJET MDE

FOKO TAGNE BRICE ALBIN

I. DESCRIPTION DE L'OPTION ET DU LANGAGE DE PROGRAMMATION CHOISIE

Pour mener ce projet, nous avons utilisé **l'option 2** avec le langage **PHP natif**.

Nous avons réalisé le métamodèle du site pédagogique pour modéliser notre langage. Par la suite nous avons fait des instances validées sur les contraintes OCL écrite. Nous avons ensuite fait le métamodèle d'une REST API basé sur le langage PHP et fait une instance validée sur des contraintes OCL. Nous avons ensuite écris des requêtes ATL pour transformer les instances en RestApi

II. PRÉSENTATION DE VOTRE LANGAGE : MÉTAMODÈLE ET SPÉCIFICATIONS OCL

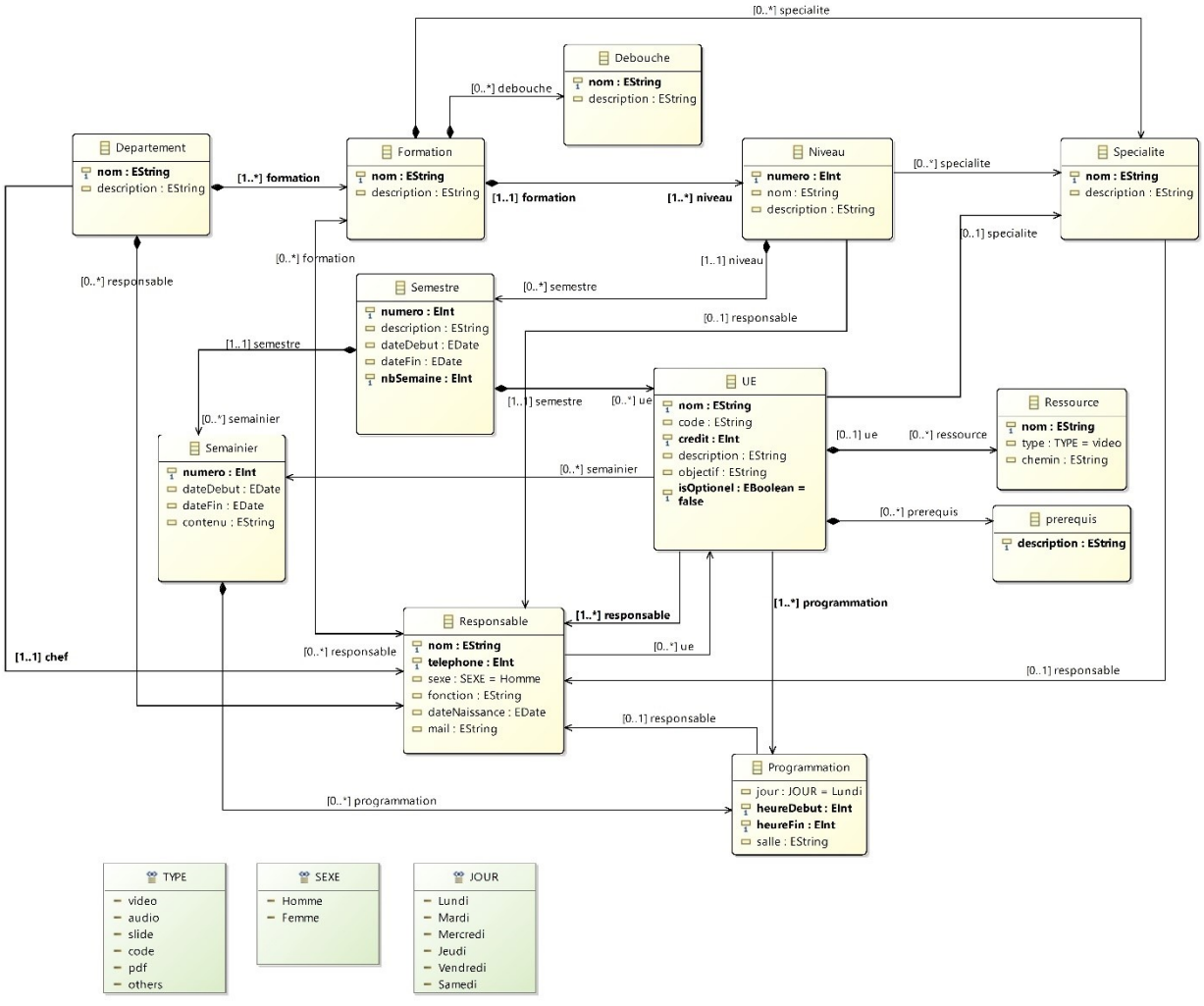
Pour concevoir le langage de modélisation, nous allons faire un langage personnalisé puis le formaliser en définissant un métamodèle après quoi nous ferons une formalisation des contraintes de modélisation à l'aide d'OCL.

1. Ebauche de notre langage pour décrire le site

```
Département [Informatique] {
    Formations {
        [LicenceFonda] {
            Semestres {
                [Semestre1] {
                    UEs {
                        [Algorithmique] {
                            Code UE1
                            Enseignants {
                                Dr. Melat Paulin
                            }
                            Ressources {
                                cours1.pdf
                                cours2.pdf
                            }
                            Programmation [S1-UE1] {
                                Lundi 10h_12h S08
                                Jeudi 14h_16h S06
                                 }
                        }
                        [Software Engineering] {
                            Code UE2
                            Enseignants {
                                Dr. Kimbi
                                Dr. Abessolo
                            }
                            Ressources {
                                slide1.pptx
                                syllabus.pdf
                            }
```

```
                    Programmation [S1-UE2] {
                        lundi_10h_12h S08
                        jeudi_14h_16h S06
                    }
                }
            }
        }
        [Semestre2] {
            …
        }
    }
    [LicenceICT] {
        …
    }
}
```

2. Métamodèle représentant le langage pour décrire les sites



3. Contraintes OCL sur le métamodèle pour décrire les sites

- **Département**

```
invariant Named: self.nom <> null;
invariant uniqueDepName: self.oclType().allInstances()->forAll(id1, id2 |id1 <> id2 implies id1.nom <> id2.nom);
invariant atLeastOneFormation: self.formation->size() > 0;
invariant UniqueFormationList: self.formation->forAll(f1, f2 |f1 <> f2 implies f1.nom <> f2.nom);
invariant auMoins1Responsable: self.responsable->notEmpty();
```

- **Formation**

```
invariant Named: self.nom <> null and self.description <> null;
invariant nomUnique: self.oclType().allInstances()->isUnique(nom);
invariant auMoins1Niveau: self.niveau->size() >= 1;
invariant niveauDistinct: self.niveau->forAll(n1, n2 |n1 <> n2 implies n1.numero <> n2.numero and n1.nom <> n2.nom);
```

- **Niveau**

```
invariant Named: self.numero <> null and self.nom <> null;
invariant auMoins1Responsable: self.responsable->notEmpty();
invariant numeroSemestreUnique: self.semestre->forAll(s1, s2 | s1 <> s2 implies s1.numero <> s2.numero);
invariant associeFormation: self.formation->notEmpty();
invariant multiSpecialite: self.specialite->size() >= 0;
invariant associeSemestre: self.semestre->notEmpty();
invariant
NotSpecialityWthDiffLevelWthSameName: self->select(n | n.formation = self.formation)->select(n | n <> self)->forAll(n | n.specialite->select(s | s.nom = self.nom)->size() = 0);
invariant semestreUnique: self.semestre->forAll(s1, s2 |s1 <> s2 implies s1.numero <> s2.numero);
```

- **Semestre**

```
invariant Named: self.numero <> null and self.description <> null;
invariant numeroUniqueParNiveau: self.niveau.semestre->forAll(s | s <> self implies s.numero <> self.numero);
invariant AssocieNiveau: self.niveau <> null and self.niveau->size() = 1;
invariant nombreSemaine: self.nbSemaine >= 1 and self.nbSemaine <= 24;
invariant NombreSeancesValide: self.semainier->size() <= self.nbSemaine;
invariant auMoins1Ue: self.ue->size() >= 1;
```

- **Specialite**

```
invariant Named: self.nom <> null;
invariant nomUnique: self.oclType().allInstances()->isUnique(nom);
```

- **Semainier**

```
invariant Named: self.numero <> null;
invariant ContenuDefini: self.contenu <> null;
invariant DatesSeancesValides: self.dateDebut >= self.semestre.dateDebut and self.dateFin <= self.semestre.dateFin;
invariant pourUnSeulSemestre: self.semestre->size() = 1;
```

- `invariant SemainierValideDate: self.dateDebut < self.dateFin;`

- **UE**

- `invariant Named: self.nom <> null and self.description <> null and self.isOptionel <> null;`
- `invariant auMoins1Responsable: self.responsable->size() >= 1;`
- `invariant dans1SeulSemestre: self.responsable->size() = 1;`
- `invariant plusieursRessources: self.ressource->size() >= 0;`
- `invariant EnseignantsValides: self.programmation->forAll(p | self.responsable->includes(p.responsable));`
- `invariant semanierValide: self.semainier->forAll(sem | self.semestre.semainier->includes(sem));`

- **Responsable**

- `invariant UesDansFormationsOuIntervient: self.ue->forAll(u | self.formation->includes(u.semestre.niveau.formation));`

- **Ressource**

- `invariant Named: self.nom <> null and self.type <> null and self.chemin <> null;`
- `invariant nomUnique: self.oclType().allInstances()->isUnique(nom);`
- `invariant uneSeuleUe: self.ue->size() = 1;`

- **Programmation**

- `invariant`
- `ContenuDefini: self.jour <> null and self.heureDebut <> null and self.heureFin <> null and self.salle <> null;`
- `invariant HoraireValid: self.heureDebut < self.heureFin and self.heureFin - self.heureDebut >= 1;`
- `invariant`
- `trancheHoraire: self.heureDebut <= 20 and self.heureDebut >= 7 and self.heureFin <= 21 and self.heureFin >= 8;`

- **Prerequis**

- `invariant Named: self.description <> null;`

- **Debouche**

- `invariant Named: self.nom <> null;`

III. DESCRIPTION D'UNE INSTANCE DE MODÈLE DU MÉTAMODÈLE

Ici nous avons réalisé une instance avec département informatique.

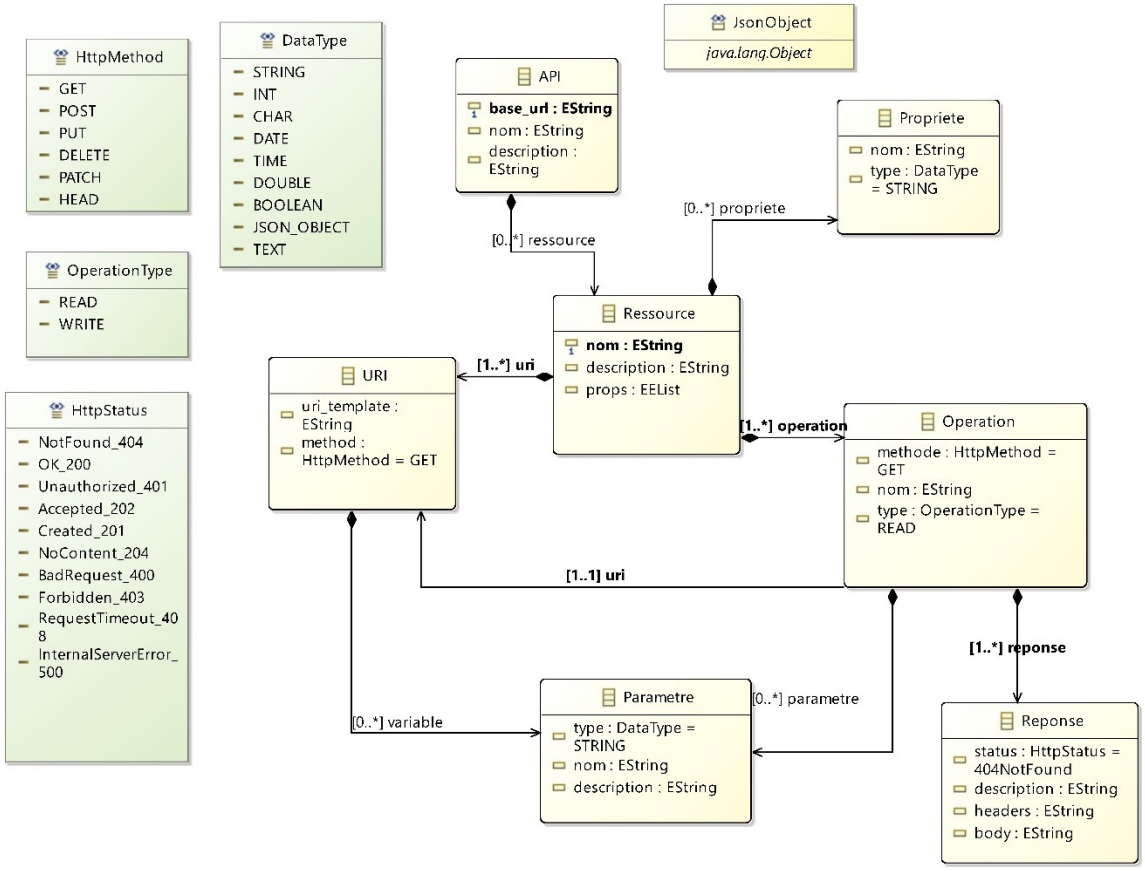
Le département est un ensemble de formations et de responsable. Chaque formation est une collection de niveau, de spécialité et de débouché. Les niveaux sont une collection de semestres qui eux aussi sont une collection d'unité d'enseignement et de semainier. Les Ues collectionnes les prérequis et les semainiers collectionnent les programmations de Ues dans la semaine.

La figure ci-dessous nous montre notre instance de modèle

- platform:/resource/mymetamodel/Model/siteInfo.xmi
  - Departement Departement d'informatique
    - Formation Licence Fonda
      - Niveau L1
        - Semestre S1-L1
          - Semainier Semanier S1-L1 Licence Fondamentale
            - Programmation A250 Mercredi 12h-14h
          - UE Algortihmique
            - Prerequis Raisonnement étapes par étapes
        - Semestre S2-L1
          - Semainier Semanier S2-L1 Licence Fondamentale
            - Programmation A250 Mardi 16h-18h
          - UE web design
            - Prerequis Analyse fonctionelle et non fonctionelle des besoins
      - Niveau L2
        - Semestre S1-L2
          - Semainier Semanier S1-L2 Licence Fondamentale
            - Programmation A250 Lundi 8h-10h
          - UE Object-Oriented Programming (OOP)
        - Semestre S2-L2
          - Semainier Semanier S2-L2 Licence Fondamentale
            - Programmation A250 Lundi 14h-16h
          - UE Système transactionelle (SGBDR)
      - Niveau L3
        - Semestre S1-L3
          - Semainier Semanier S1-L3 Licence Fondamentale
            - Programmation A250 Mercredi 15h-17h
          - UE Système d'exploitation (SE)
        - Semestre S2-L3
          - Semainier Semanier S2-L3 Licence Fondamentale
            - Programmation A250 Samedi 10h-12h
          - UE Bases de données relationnelles
        - Debouche analyste-programmeur
        - Debouche développeur web
    - Formation Master Fonda
      - Niveau M1
        - Semestre S1-M1
          - Semainier Semanier S1-M1 Master Fonda
            - Programmation S107 Jeudi 10h-12h
          - UE Algorithmique avancée
        - Semestre S2-M1
          - Semainier Semainier S2-M1 Master fonda
            - Programmation S107 Vendredi 8h-10h
          - UE Compilation
      - Niveau M2
        - Semestre S1-M2
          - Semainier semainier M2 avec les programmations des UEs de la filière Geni logiciel
            - Programmation S006 Jeudi 8h-10h
            - Programmation S006 Lundi 10h-12h
            - Programmation S006 Lundi 16h-18h
            - Programmation S008 Mercredi 10h-12h
            - Programmation S107 Jeudi 16h-19h
          - Semainier Semanier M2 avec les programmations des UEs de la filière science de données
            - Programmation S008 Lundi 8h-10h
            - Programmation S008 Lundi 12h-14h
            - Programmation S008 Lundi 16h-19h
            - Programmation S006 Lundi 8h-10h
          - UE Fouille de donnée
            - Prerequis connaitre les structure algorithmiques de bases
          - UE Search-Based Software Engineering
            - Prerequis connaitre les heuristiques
          - UE Big Data
            - Prerequis manipuler les BD relationelle
          - UE Buisiness Intelligent (BI)
            - Prerequis connaitre les SI
          - UE Methodologie de recherche scientique
            - Prerequis savoir faire une recherche en ligne
            - Ressource Slide Veille scientifique et technologique
          - UE Ingénierie Dirigée par le Modèles (IDM)
            - Prerequis Maitrise Outils developpement Eclipse
            - Prerequis Maitrise UML
            - Ressource Introduction à l'IDM
            - Ressource métamodelisation
            - Ressource contraintes OCL
            - Ressource Transformation ATL
        - Debouche Concepteur de solutions ERP
        - Debouche Concepteur de SI
        - Debouche Expert Business Intelligent
        - Specialite Science des données
        - Specialite Génie Logiciel
      - Responsable Dr. Paulin
      - Responsable Pr. Nobert
      - Responsable Pr. Roger
      - Responsable Dr. Xaviera
      - Responsable Dr Fidèle
      - Responsable Dr. Valery Monthe
      - Responsable Dr Aminou
      - Responsable Dr. tapamo

IV. MÉTAMODÈLE RESTAPI

1. Métamodèle



2. Contraintes OCL

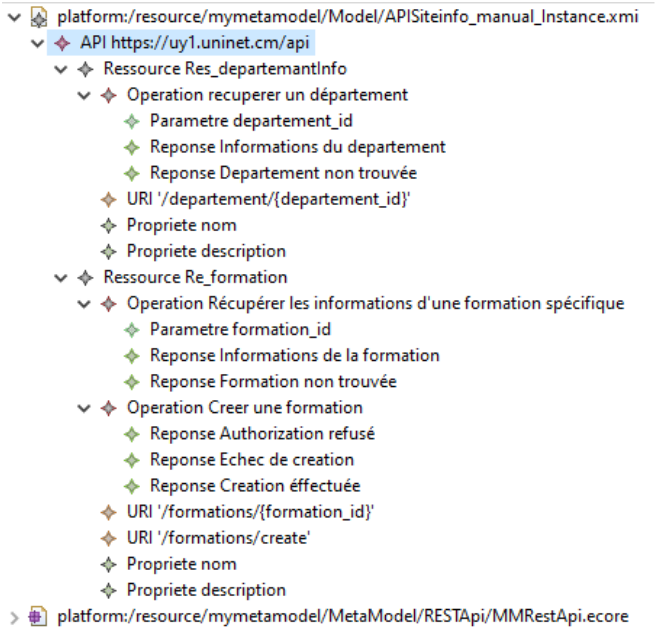
- **Ressource**

- `invariant at_least_one_operation: self.operation->size() >= 1;`
- `invariant unique_name: self.oclType().allInstances()->isUnique(nom) ;`

- **Operation**

- `invariant one_uri: self.uri->size() = 1;`
- `invariant http_method_consistency: (self.type = OperationType::READ and (self.methode = HttpMethod::GET or self.methode = HttpMethod::HEAD)) or (self.type = OperationType::WRITE and (self.methode = HttpMethod::POST or self.methode = HttpMethod::PUT or self.methode = HttpMethod::PATCH or self.methode = HttpMethod::DELETE));`
- `invariant read_operation_http_method: self.type <> OperationType::READ or (self.methode <> HttpMethod::POST and self.methode <> HttpMethod::PUT and self.methode <> HttpMethod::PATCH and self.methode <> HttpMethod::DELETE);`
- `invariant uris_have_defined_variable: self.uri->forAll(u | u.variable->forAll(v |u.uri_template.indexOf(v.nom) <> -1));`
- `invariant read_not_modify: self.type <> OperationType::READ or self.methode <> HttpMethod::POST and self.methode <> HttpMethod::PUT and self.methode <> HttpMethod::PATCH and self.methode <> HttpMethod::DELETE;`
- `invariant have_httpMethod: not self.methode.oclIsUndefined() implies self.uri.method = self.methode;`

V. UNE INSTANCE DE MODÈLE DU MÉTAMODÈLE RESTAPI



VI. REGLES TRANSFORMATION ATL

```
-- Transformation des departements
Rule TransformDepartement {
           from
               s : Site!Departement
           to
               a : API!Resource (uri_template <- 'departement/{id}')
            do
                -- Copie des propriétés de la formation dans la ressource API
                a.nom <- s.nom;
                a.description <- s.description;
                -- Création d'une sous-ressource pour formation du departementn
                for (form in s.formation) {
                    form_a : API!Resource (uri_template <- '/departement/{id}/formations/{id}')
                    form_a.nom <- form.nom;
                    -- Ajout de la sous-ressource au modèle API
                    a.subresources <- form_a;
                }
        }

-- Transformation des formations
Rule TransformFormation {
          from
                s : Site!Formation
          to
                a : API!Resource (uri_template <- '/formations/{id}')
          do
                -- Copie des propriétés de la formation dans la ressource API
                a.nom <- s.nom;
                a.description <- s.description;
                -- Création d'une sous-ressource pour chaque semestre de la formation
                for (sem in s.semestre) {
                    sem_a : API!Resource (uri_template <- '/formations/{id}/semesters/{numero}')
                    sem_a.sem_number <- sem.numero;
                    -- Ajout de la sous-ressource au modèle API
                    a.subresources <- sem_a;
```

```
                }
    }

-- Transformation des semestres
Rule TransformSemestre {
            from
                s : Site!Semestre
            to
                a : API!Resource (uri_template <-
'/formations/{formation_id}/semesters/{semester_number}')
            do
                -- Copie des propriétés du semestre dans la ressource API
                a.nom <- s.nom;
                a.description <- s.description;
                -- Création d'une sous-ressource pour chaque UE du semestre
                for (ue in s.ues) {
                    ue_a : API!Resource (uri_template <-
'/formations/{formation_id}/semesters/{semester_number}/ues/{ue_id}')
                    ue_a.ue_id <- ue.id;
                    -- Ajout de la sous-ressource au modèle API
                    a.subresources <- ue_a;
                }
    }

-- Transformation des UE
Rule TransformUE {
            from
                s : Site!UE
            to
                a : API!Resource (uri_template <-
'/formations/{formation_id}/semesters/{semester_number}/ues/{ue_id}')
            do
                -- Copie des propriétés de l'UE dans la ressource API
                a.nom <- s.nom;
                a.description <- s.description;
                -- Ajout d'une propriété pour les ressources associées à l'UE
                a.resources <- s.resources;
                -- Ajout d'une sous-ressource pour les responsables de l'UE
                responsables_a : API!Resource (uri_template <-
'/formations/{formation_id}/semesters/{semester_number}/ues/{ue_id}/responsables')
                -- Ajout de la sous-ressource au modèle API
                a.subresources <- responsables_a;
    }

-- Transformation des responsables d'UE
Rule TransformResponsables {
            from
                s : Site!UE
            to
                a : API!Resource (uri_template <-
'/formations/{formation_id}/semesters/{semester_number}/ues/{ue_id}/responsables')
            do
                -- Copie des propriétés des responsables dans la ressource API
                a.nom <- s.responsable.nom;
                a.email <- s.responsable.mail;
    }


-- OPERATIONS
-- Récupérer toutes les formations
rule get_all_formations {
        from
                s: Site
        to
                a: API
        do {
                -- Créer une nouvelle ressource de type "formation" dans l'API
                var formationResource: RESOURCE = API!RESOURCE.create(name := "formation", api := a);
                -- Ajouter une opération GET sur la ressource pour récupérer toutes les formations
```

```
		var getAllOperation: OPERATION = RESOURCE!OPERATION.create(name := "getAll", resource :=
	formationResource, method := "GET");
		-- Ajouter une URI pour l'opération GET
		var uriTemplate: URI_TEMPLATE = OPERATION!URI_TEMPLATE.create(value := "formations",
	operation := getAllOperation);
	}
}
-- Créer une nouvelle formation
rule create_formation {
	from
		f: Formation
		s: Site
	to
	a: API
	do {
		-- Créer une nouvelle ressource de type "formation" dans l'API
		var formationResource: RESOURCE = API!RESOURCE.create(name := "formation", api := a);
		-- Ajouter une opération POST sur la ressource pour créer une nouvelle formation
		var createOperation: OPERATION = RESOURCE!OPERATION.create(name := "create", resource :=
	formationResource, method := "POST");
		-- Ajouter une URI pour l'opération POST
		var uriTemplate: URI_TEMPLATE = OPERATION!URI_TEMPLATE.create(value := "formations",
	operation := createOperation);
		-- Ajouter une représentation pour l'opération POST
		var representation: REPRESENTATION = OPERATION!REPRESENTATION.create(mediaType :=
	"application/json", operation := createOperation);
		-- Ajouter une contrainte pour l'opération POST
		var constraint: CONSTRAINT = OPERATION!CONSTRAINT.create(expression := "self.nom-
	>notEmpty()", operation := createOperation);
	}
}
-- Récupérer une formation spécifique
rule get_formation {
	from
		f: Formation
		s: Site
	to
		a: API
	do {
		-- Créer une nouvelle ressource de type "formation" dans l'API
		var formationResource: RESOURCE = API!RESOURCE.create(name := "formation", api := a);
		-- Ajouter une opération GET sur la ressource pour récupérer une formation spécifique
		var getOperation: OPERATION = RESOURCE!OPERATION.create(name := "get", resource :=
	formationResource, method := "GET");
		-- Ajouter une URI pour l'opération GET
		var uriTemplate: URI_TEMPLATE = OPERATION!URI_TEMPLATE.create(value := "formations/{id}",
	operation := getOperation);
		-- Ajouter un paramètre pour l'URI
		var uriParameter: URI_PARAMETER = URI_TEMPLATE!URI_PARAMETER.create(name := "id",
	uriTemplate := uriTemplate);
	}
}
```

VII. RÉSULTAT JSON


```
{
  "id": "https://www.fil.univ-lille.fr/portail",
  "name": "Portail FIL",
  "description": "Portail pédagogique de la Faculté d'Informatique de Lille",
  "base_url": "https://www.fil.univ-lille.fr/portail/api",
  "resources": [
    {
      "name": "formations",
      "description": "Liste des formations",
      "url": "/formations",
      "methods": [
```

```
      {
        "name": "GET",
        "description": "Récupérer la liste des formations",
        "parameters": [],
        "responses": [
          {
            "code": 200,
            "description": "Liste des formations",
            "schema": {
              "type": "array",
              "items": {
                "$ref": "#/definitions/Formation"
              }
            }
          }
        ]
      }
    ]
  },
  {
    "name": "formation",
    "description": "Formation spécifique",
    "url": "/formations/{formation_id}",
    "methods": [
      {
        "name": "GET",
        "description": "Récupérer les informations d'une formation spécifique",
        "parameters": [
          {
            "name": "formation_id",
            "type": "string",
            "description": "Identifiant de la formation"
          }
        ],
        "responses": [
          {
            "code": 200,
            "description": "Informations de la formation",
            "schema": {
              "$ref": "#/definitions/Formation"
            }
          },
          {
            "code": 404,
            "description": "Formation non trouvée"
          }
        ]
      }
    ]
  },
  {
    "name": "ues",
    "description": "Liste des UEs d'une formation",
    "url": "/formations/{formation_id}/ues",
    "methods": [
      {
        "name": "GET",
        "description": "Récupérer la liste des UEs d'une formation",
        "parameters": [
          {
            "name": "formation_id",
            "type": "string",
            "description": "Identifiant de la formation"
          }
        ],
        "responses": [
          {
            "code": 200,
            "description": "Liste des UEs de la formation",
```

```
            "schema": {
              "type": "array",
              "items": {
                "$ref": "#/definitions/UE"
              }
            }
          },
          {
            "code": 404,
            "description": "Formation non trouvée"
          }
        ]
      }
    ]
  },
  {
    "name": "ue",
    "description": "UE spécifique",
    "url": "/ues/{ue_id}",
    "methods": [
      {
        "name": "GET",
        "description": "Récupérer les informations d'une UE spécifique",
        "parameters": [
          {
            "name": "ue_id",
            "type": "string",
            "description": "Identifiant de l'UE"
          }
        ],
        "responses": [
          {
            "code": 200,
            "description": "Informations de l'UE",
            "schema": {
              "$ref": "#/definitions/UE"
            }
          },
          {
            "code": 404,
            "description": "UE non trouvée"
          }
        ]
      }
    ]
  }
}
```


VIII. PROCÉDURE À SUIVRE POUR EXPLOITER OU TESTER VOTRE PROJET

- ☞ Installer EMF dans Eclipse puis installer les modules OCL et ATL depuis la marketplace
- ☞ Copier les différents dossiers du projet dans le Workspace de son logiciel Eclipse :
  - Mymetamodel
  - Site2API
  - Mymetamodel.edit
  - Mymetamodel.editor
- ☞ Pour tester la génération avec ATL, il suffit de paramétrer les inputs et output avec respectivement le Métamodèle du langage de site et le Métamodèle de l'API. Ils sont disponibles depuis le module Site2API et le module Mymetamodel
- ☞ Insérer alors le modèle à transformer et préciser la sortie
- ☞ Cliquer sur Run pour lancer la génération ATL