MDE Tool

1. MDE Tool execution steps.

Code Generation using the MDE Tool involves three steps: architecture specification, PIM parameterization, and code generation.

1. Architecture specification:

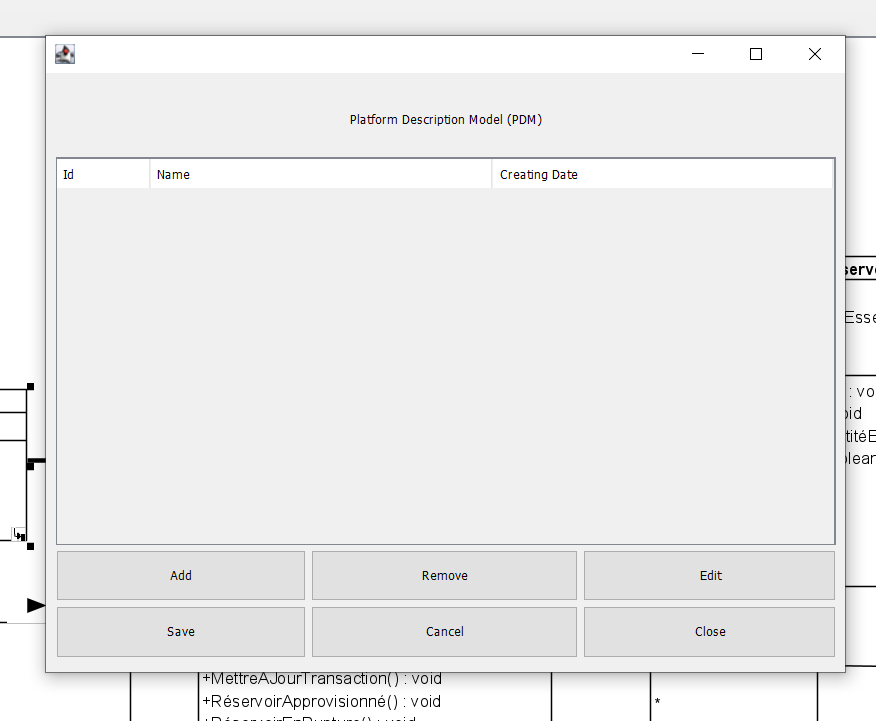
To specify the architecture, follow these steps:

* Open Visual Paradigm, then click on "Platform Description Model" in the "MDE" tab.

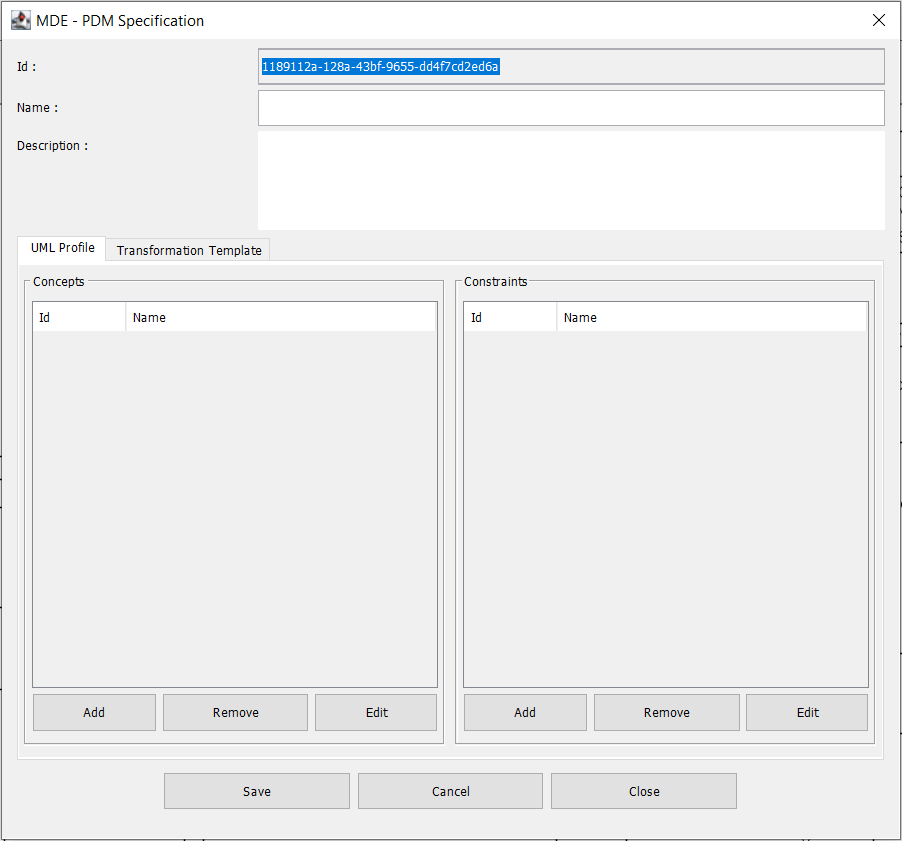
Une image contenant texte

Description générée automatiquement

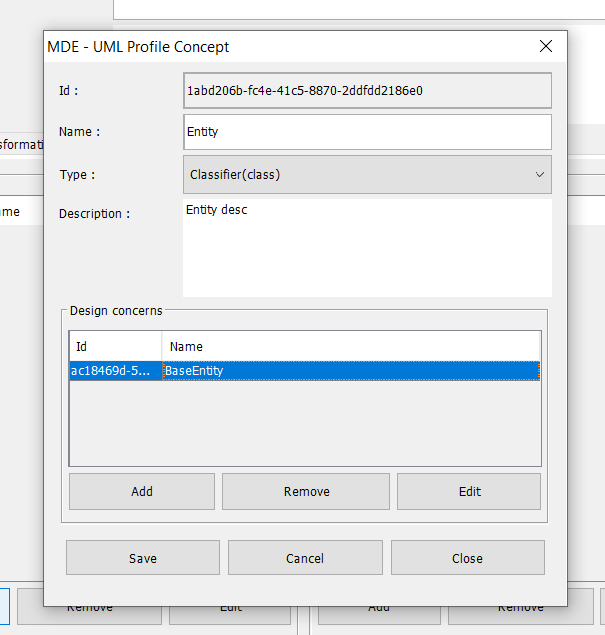
* Click on "Add" in the main window to add a PDM.



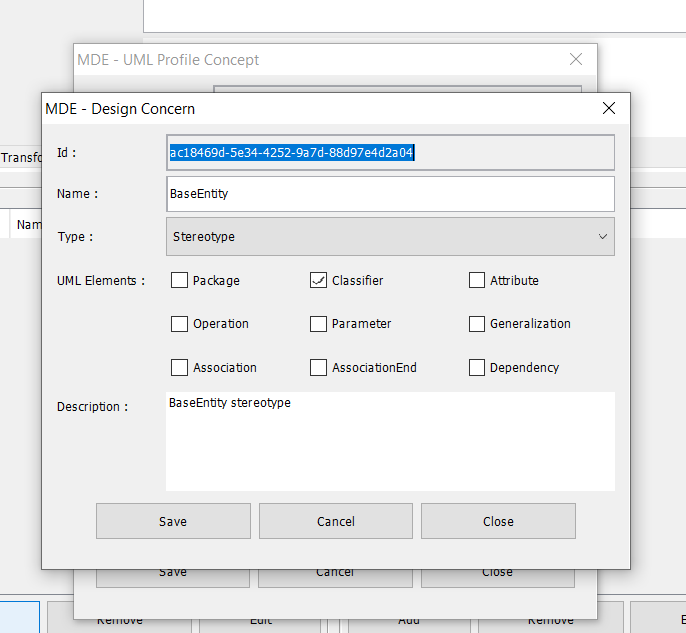
* A PDM is defined by a UML profile and a set of transformations. Click on "Add" in each section to add a UML profile concept or constraint, or a transformation.



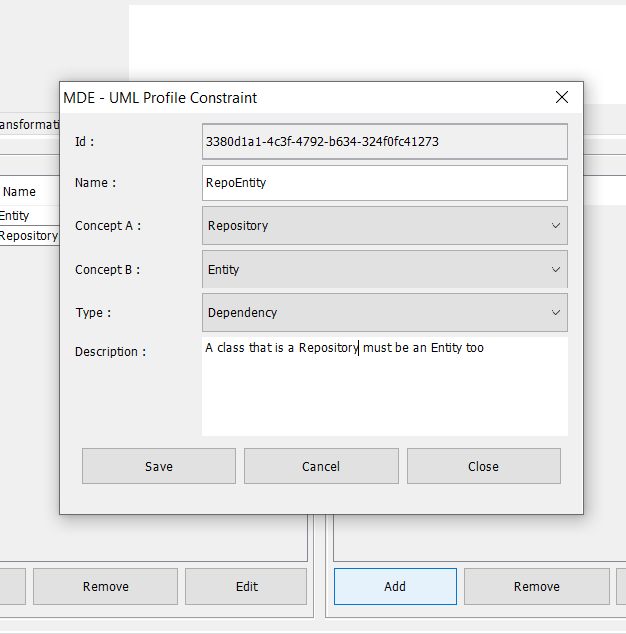
* To add a concept, enter: the name, type, description, and design concerns. To add a design concern, click on "Add" in the "Design concerns" section.



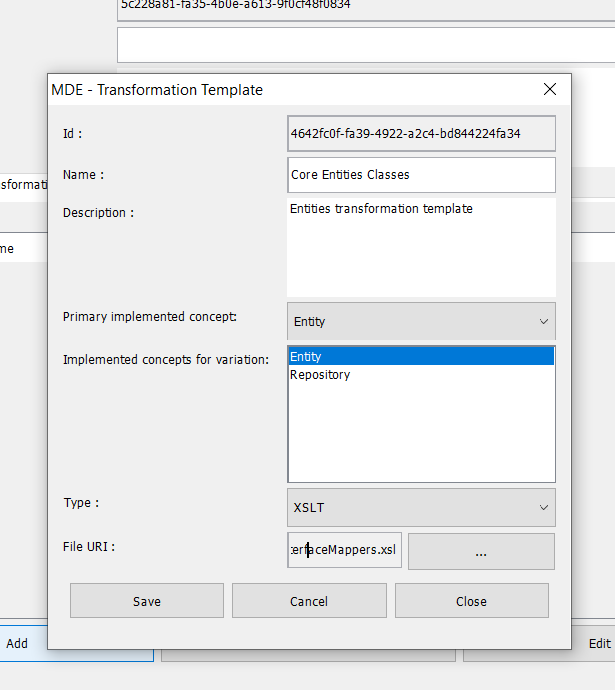
* To add a "design concern," enter the name, type, concerned UML elements, and description.



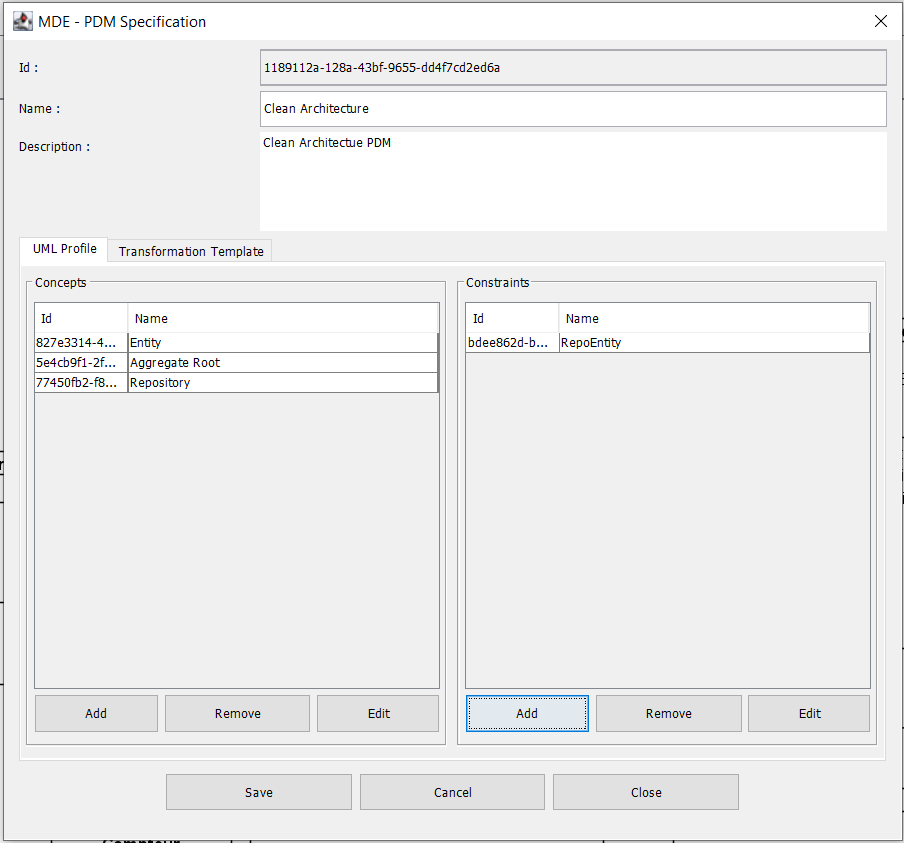
* After defining the concepts, we need to establish constraints between them. To add a constraint, enter the name, the first concept, the second concept, the type, and the description.



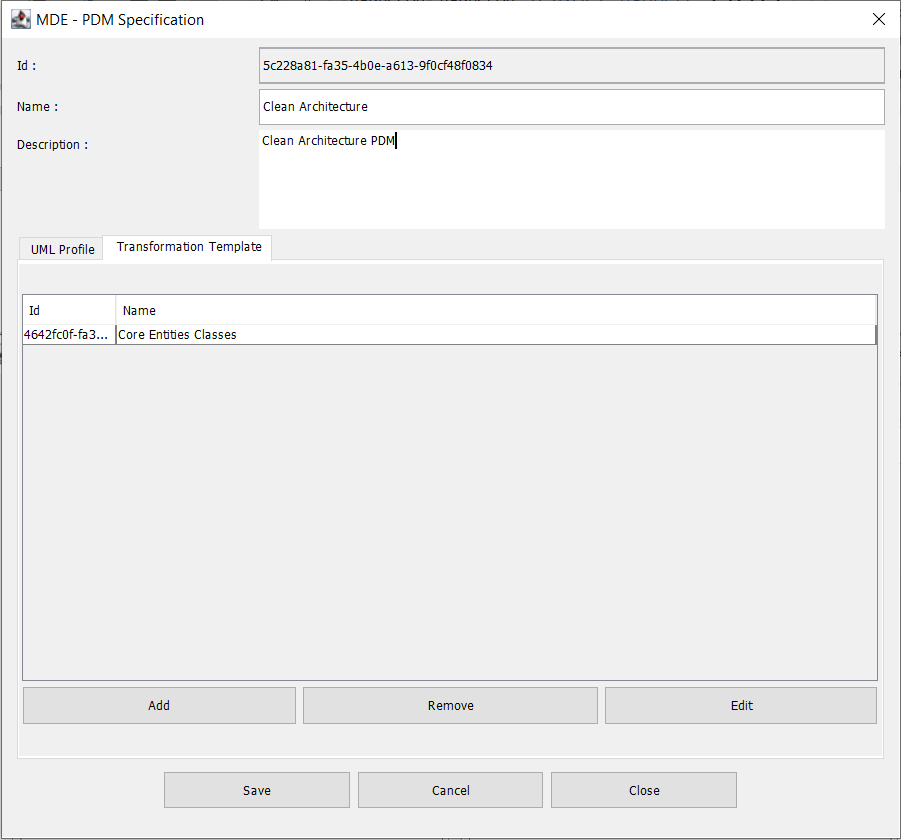
* The second part of architecture specification involves defining transformations. To add a transformation, enter the name, description, main implemented concept, implemented concepts for variation, type, and the path to the transformation.



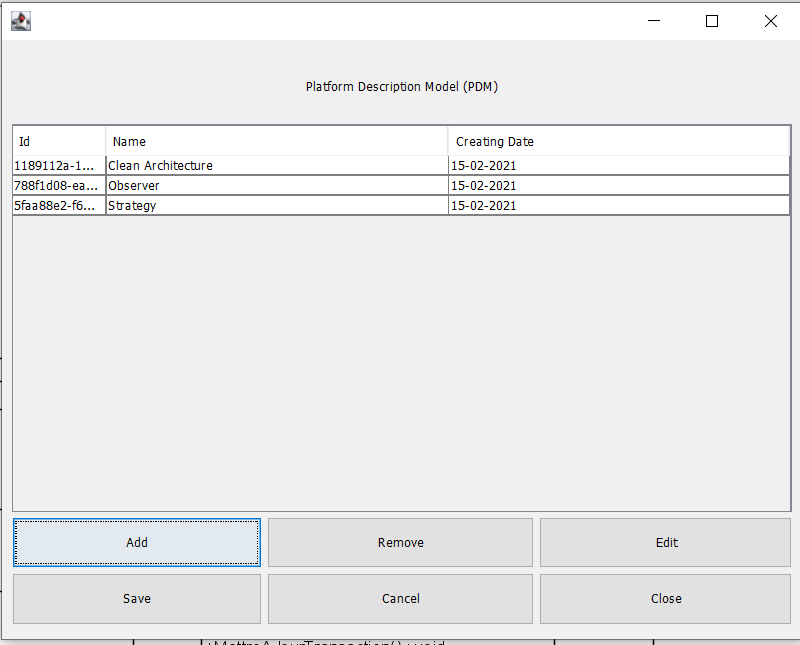
* Here is the UML profile after its definition. The user can add, modify, or delete a concept or constraint.



* Here are the transformation models after their definition. The user can add, modify, or delete a transformation model.
* After architecture specification, click on "Save".



* After adding the "clean architecture" PDM, I added others for design patterns (observer and strategy). The user can add, modify, or delete a PDM.
* Finally, click on "Save" to save the PDMs.

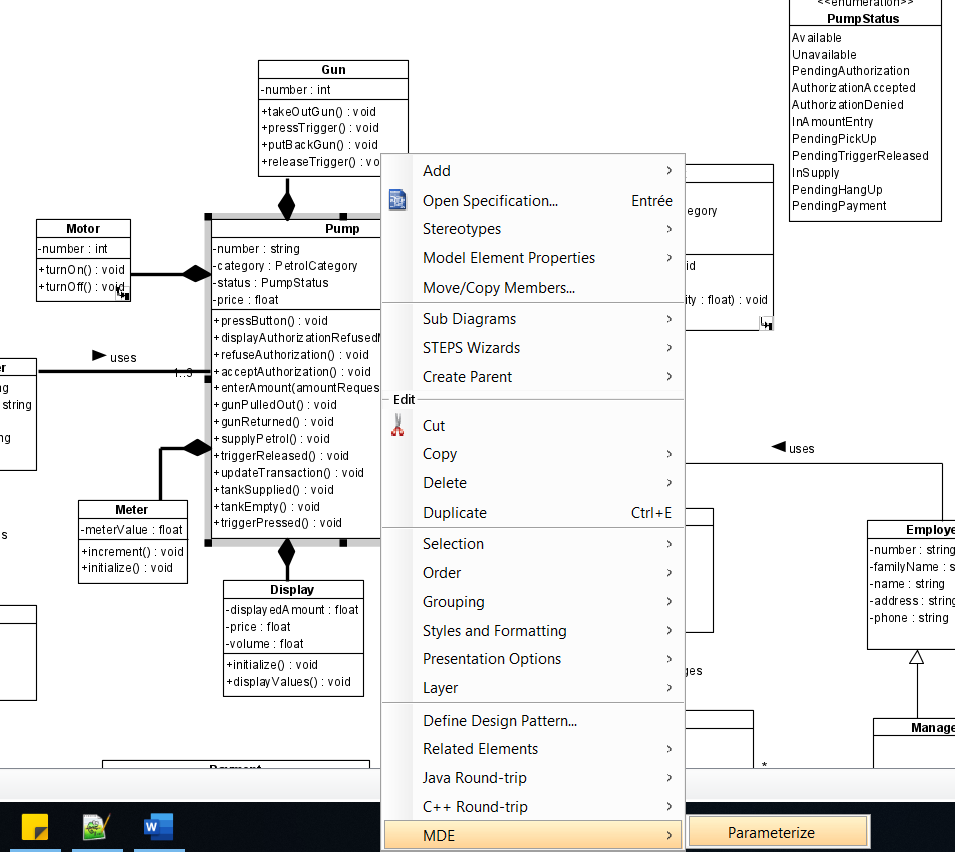


* Les The PDMs are saved in XML format in the following path:

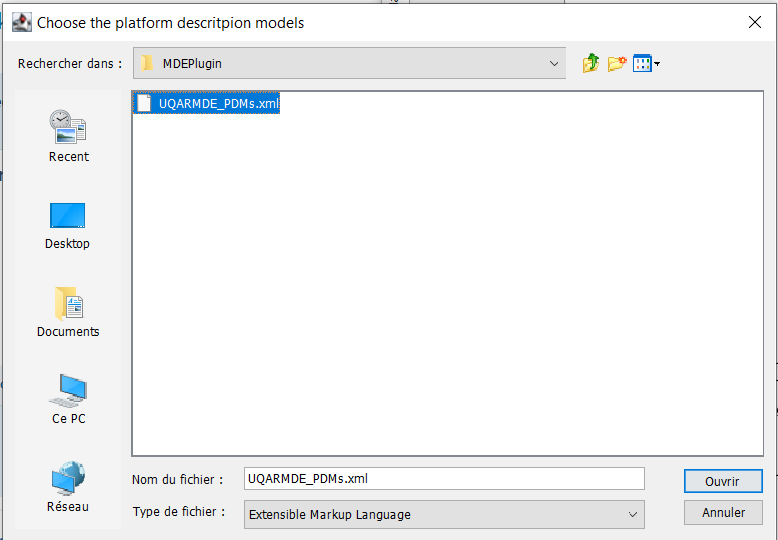
C:\Users\<User\_Name>\Documents\MDETool\<PDM\_Name.xml>

1. PIM parameterization:

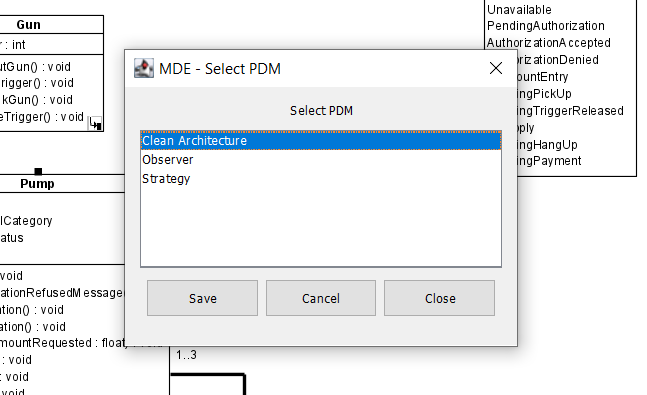
* Select the UML elements to be parameterized, then right-click and choose "MDE," and then click on "Parameterize." In the following example, I have selected the "Pump" class.



* Choose the previously defined PDMs.



* Select one or more PDMs to parameterize the PIM and click on "Save." In this example, I have chosen only the "Clean Architecture" PDM.

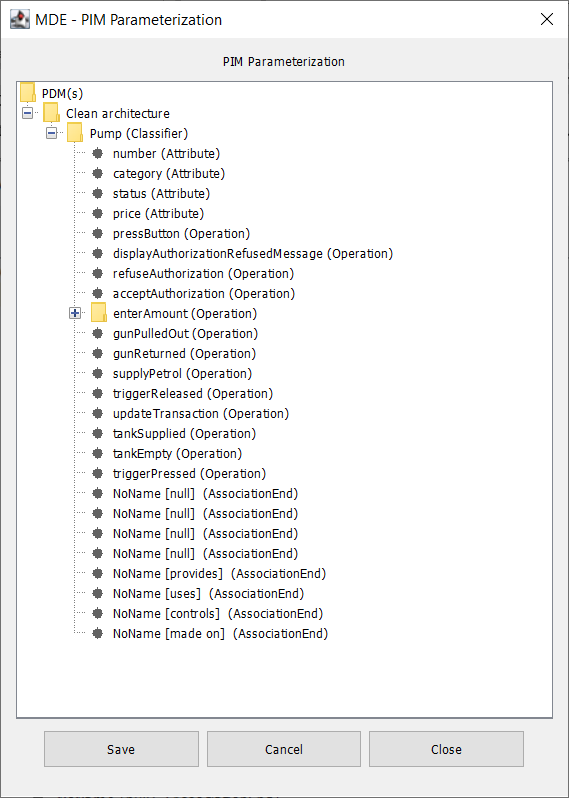


* The opened window consists of a hierarchy of elements. The root nodes of the hierarchy are the chosen PDMs. The branches from the root nodes represent the UML elements selected in Visual Paradigm. Furthermore, certain selected UML elements will have sub-branches, leading to:

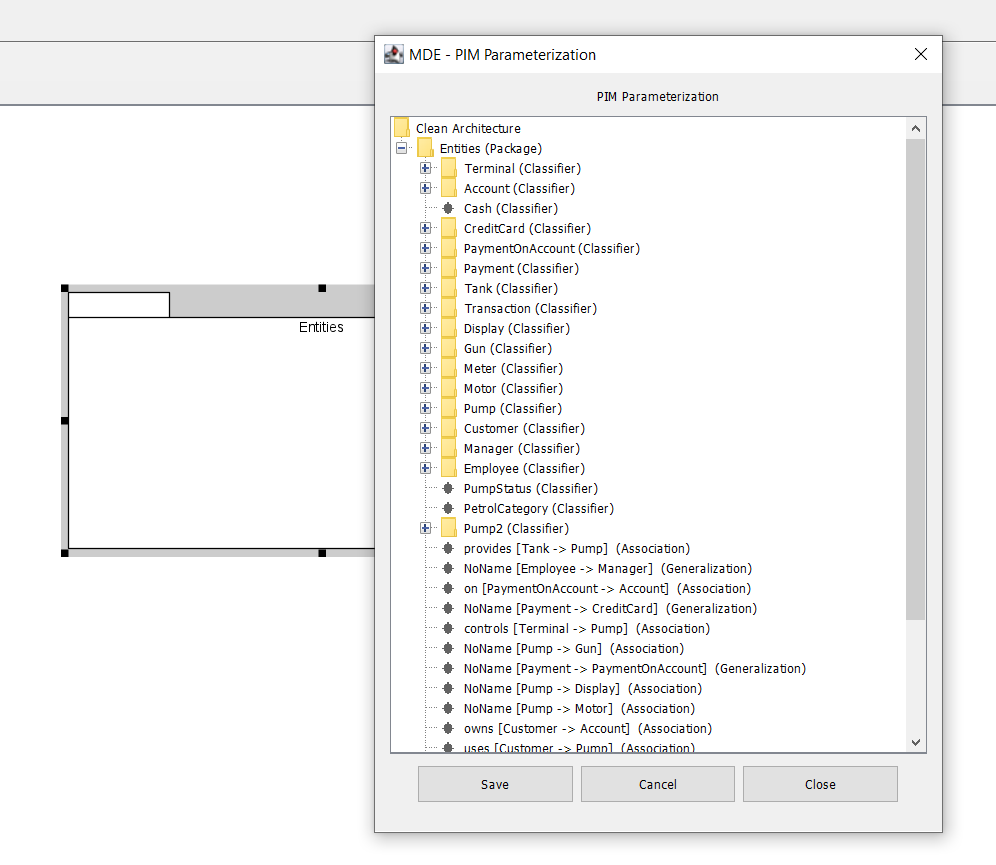
- If the selected UML element is a package, its branches will consist of classes and associations.

- If the selected UML element is a class, its branches will include attributes, operations, and association ends.

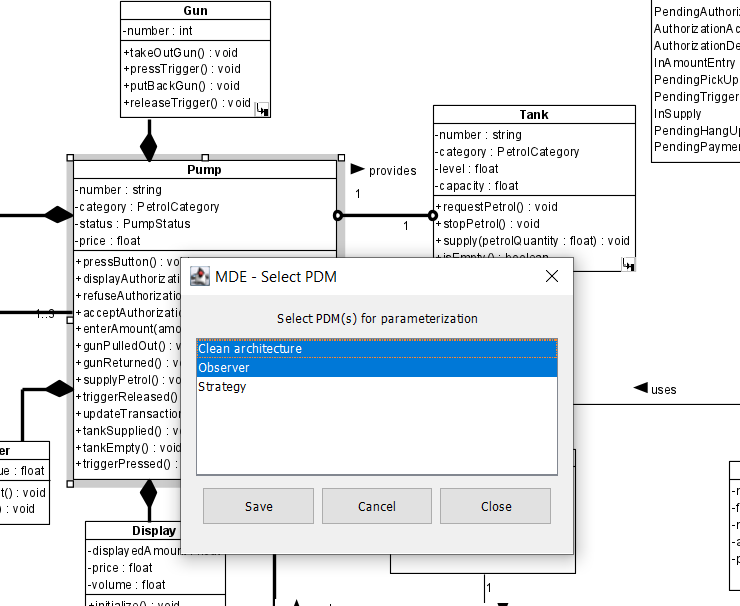
- If the selected UML element is an operation, its branches will include parameters.



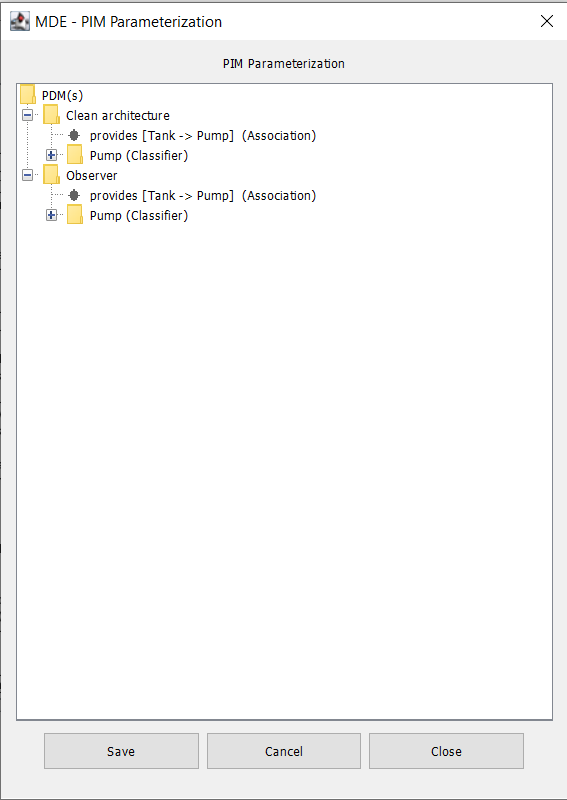
* Here is the hierarchy of elements when I selected a package for parameterization.



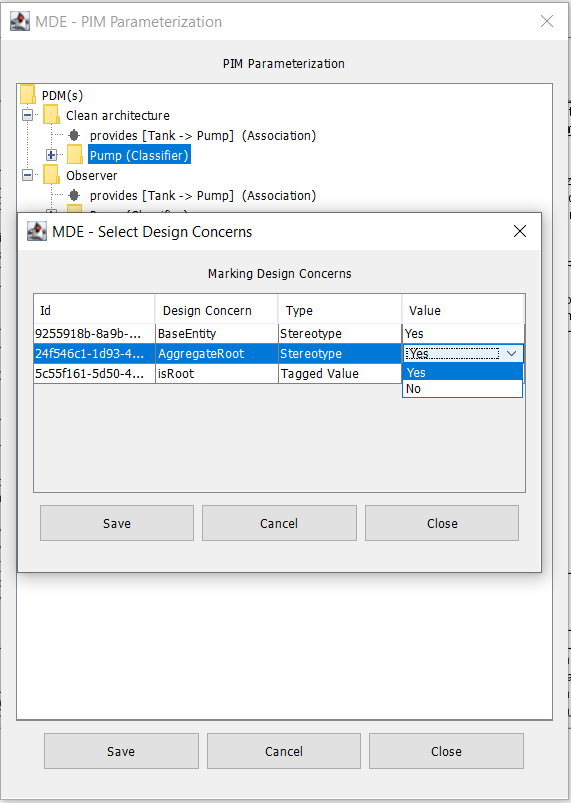
* The user can choose multiple PDMs and select multiple UML elements simultaneously. For instance, I selected the PDMs "Clean architecture" and "Observer," and I chose the class "Pump" and the association "provide" (between "Pump" and "Tank").



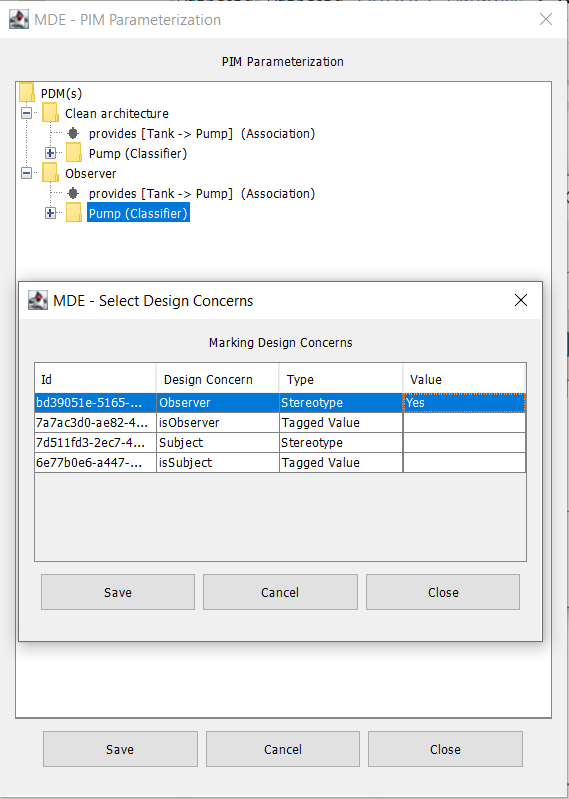
* The tool will list each chosen PDM, and for each PDM, it will list the selected UML elements. Therefore, the user can parameterize a UML element with multiple PDMs simultaneously.



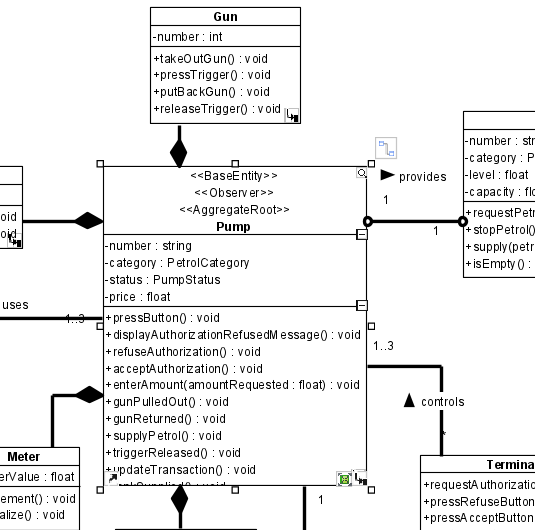
* Click on a UML element to parameterize it.
* In the following example, I clicked on the "Pump" class within the "Clean architecture" PDM. Subsequently, a window opens containing the applied design concerns for UML elements of type "Classifier."
* For stereotypes, select "yes" or "no" as the value. And for tagged values, enter the desired value.
* Click on "Save."



* The user can do the same thing for the "Pump" class, but with the design concerns from the "Observer" PDM.



* After saving the parameterization, the user can visualize the result in the PIM.

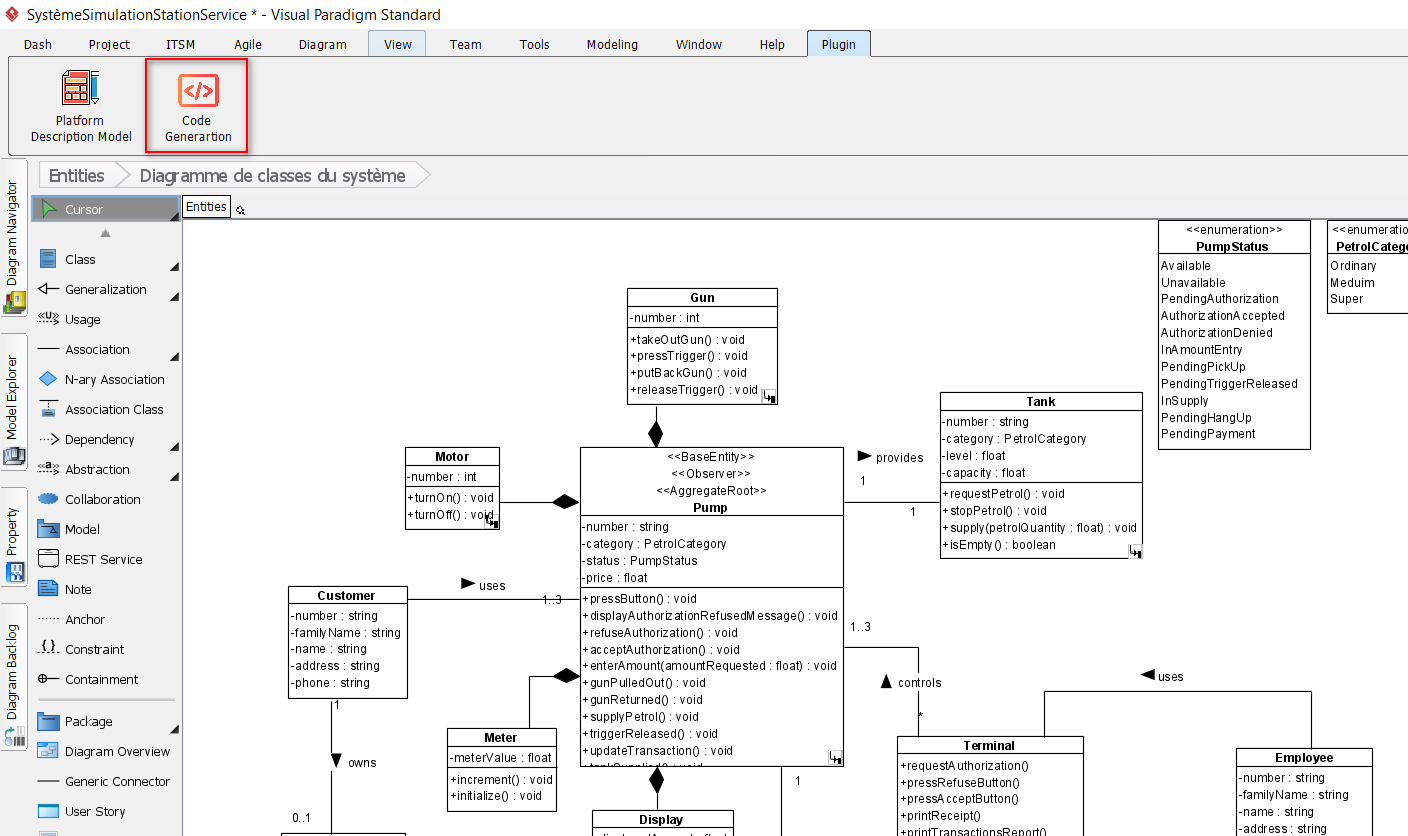


* The XML file containing the result of the parameterization is saved in the path:

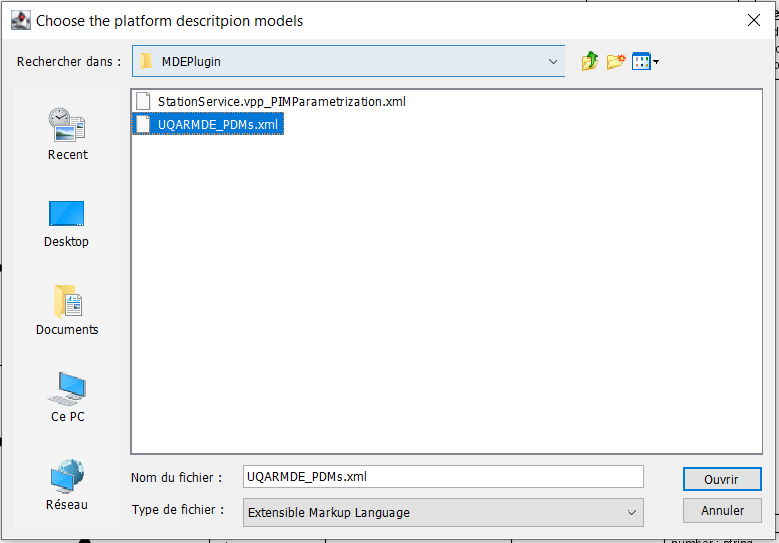
C:\Users\<User\_Name>\Documents\MDETool\<VP\_Project\_Name>\<VP\_Project\_Name>\_PIMParametrization.xml

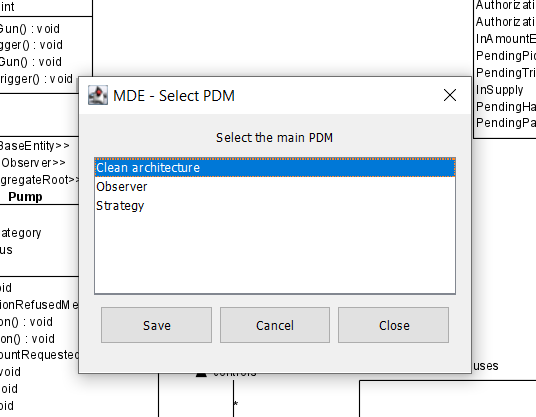
1. Code generation:

* To generate the code, click on "Code Generation" in the "MDE" tab.

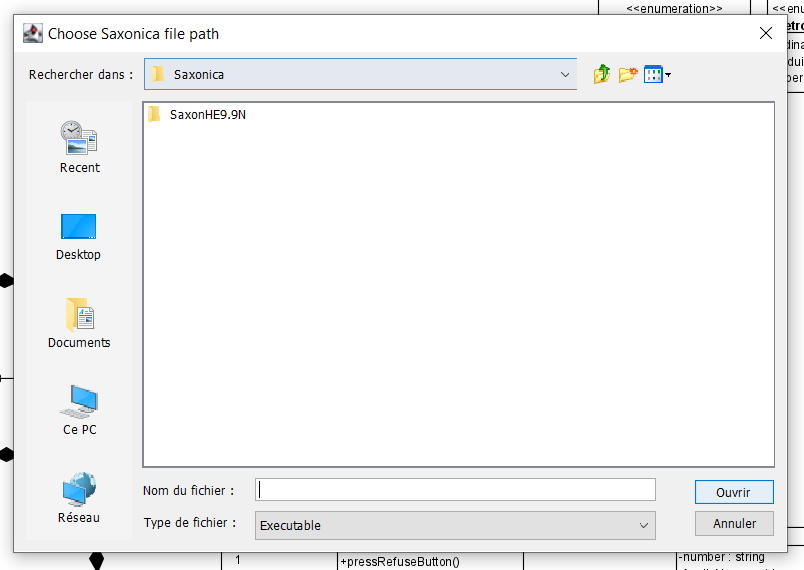


* Select the main PDM that contains the transformation models.

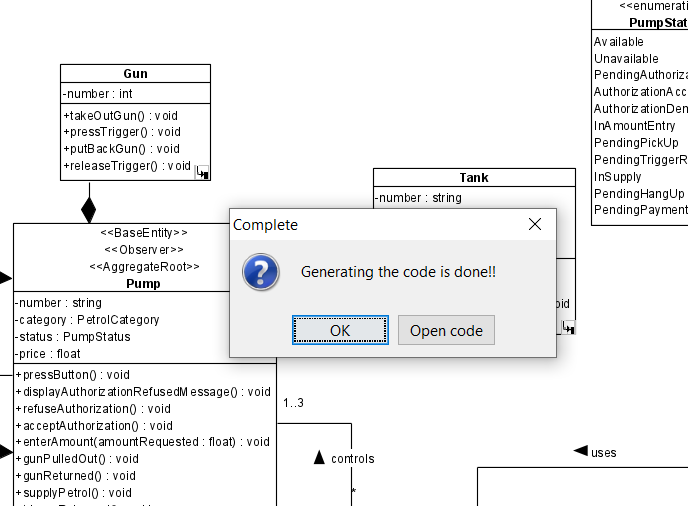




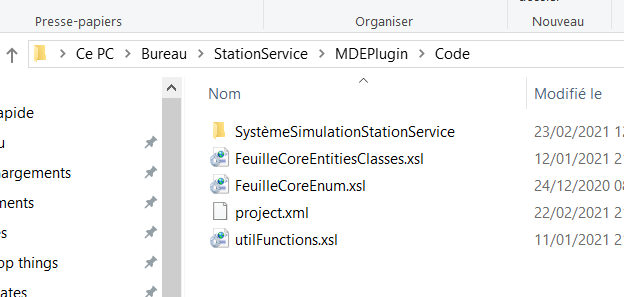
* Select the path to the transformation tool "Saxonica." For example:

"C:\Program Files\Saxonica\SaxonHE9.9N\bin\Transform.exe".

* Wait for the code generation to complete, then click on "Open code" to navigate to the code location.



* Le code est généré dans le dossier « Code ».



1. Tester le plugin

Pour tester le plugin dans Visual Paradigm (version 15.2) :

Copier le dossier «Code Generation» dans le dossier «Plugins» de Visual Paradigm. Dans ma machine il est dans le chemin suivant : « **C:/Users/hamza/AppData/Roaming/VisualParadigm/plugins** »

1. Configuration de Visual Studio code pour développer le plugin.

* <https://forums.visual-paradigm.com/t/using-vs-code-to-develop-vp-plugins/14987>
* Installer VS Code.
* Installer java JDK (jdk-11.0.3) et ajouter le chemin (ex : C:\Program Files\Java\jdk-11.0.3\bin) dans les variables d’environnement.
* Ajouter le chemin vers Apache Ant(dossier : apache-ant-1.10.7-bin) dans les variables d’environnement.

Dans ma machine : C:\Users\hamza\Downloads\apache-ant-1.10.7-bin\apache-ant-1.10.7\bin

* On doit ajouter quatre extensions à Visual Studio Code :
* Java Extension Pack : <https://marketplace.visualstudio.com/items?itemName=vscjava.vscode-java-pack>
* XML : <https://marketplace.visualstudio.com/items?itemName=redhat.vscode-xml>
* Ant Target Runner : <https://marketplace.visualstudio.com/items?itemName=nickheap.vscode-ant>
* Ant Tree Viewer : <https://marketplace.visualstudio.com/items?itemName=pedroguerra.ant-tree-viewer>
* Ouvrir le dossier «CodeGenerationPlugin» avec VS Code
* Changer le chemin vers « Open API » dans le fichier «.classpath ». Dans ma machine il existe dans : «C:/Program Files/Visual Paradigm 15.2/lib/openapi.jar».
* Pour le « Build », Il faut changer les deux propriétés « vp » et « openapi » dans le fichier « build.xml ». Dans ma machine :

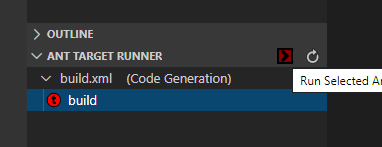
Openapi (emplacement d’open api) :

« C:/Program Files/Visual Paradigm 15.2/lib/openapi.jar »

Vp (destination de l’exécutable du plugin après le build):

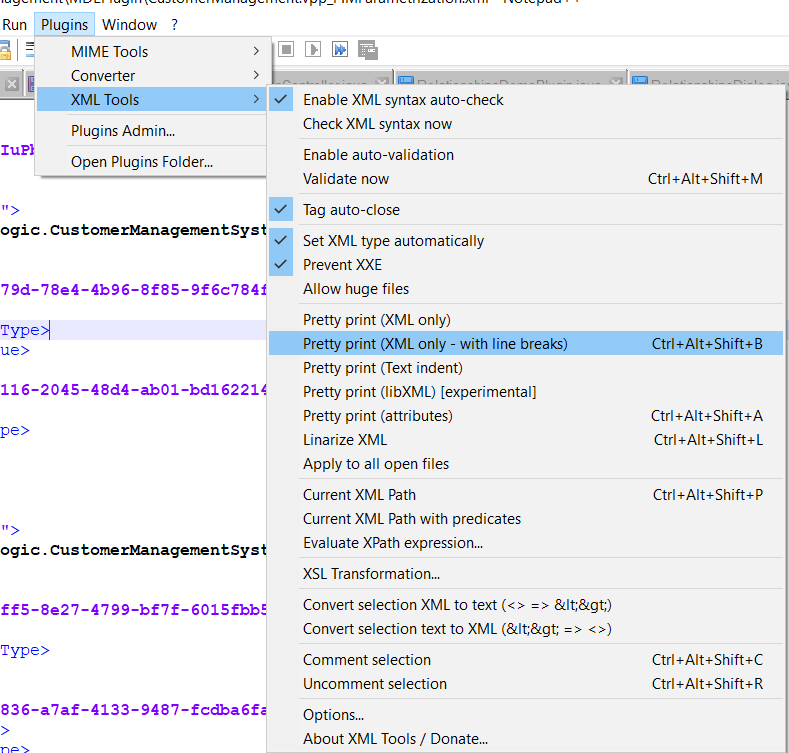
 « C:/Users/hamza/AppData/Roaming/VisualParadigm/plugins »

* Pour faire le build et le déploiement du plugin après chaque changement dans le code. Il faut cliquer sur la flèche rouge dans l’onglet « ANT TARGET RUNNER »



1. Remarques:

* Pour visualiser les fichiers XML j’ai utilisé « Notepad++ ». Le plugin sauvegarde les fichiers XML dans une seule ligne, pour les rendre lisible il faut installer le plugin «XML Tools » dans Notepad++.



1. Debugging:

* XML : problème de création de fichiers XML dans certains projets.
* UI : Problème dans les champs « description » lorsque le texte dépasse la largeur du « textbox ».
* UI : bouton pour choisir le chemin vers le modèle de transformation.
* Diviser le tableau « DesignConcernMarkingHandler » en deux : un tableau pour les stéréotypes et l’autre pour tagged value.
* Dans le tableau « DesignConcernMarkingHandler », ajouter le champ « Concept ».
* Vérifier le nom de « AssociationEnd » dans l’arborescence.
* Déplacer les modèles de transformation vers le dossier « Code » manuellement ou automatiquement?
* Lors la parameterisation d’un élément UML avec deux PDMs, les valeurs sont perdues dans la première « Save »
* Paramétrisation de PIM : Les énumérations sont considérées comme des classificateurs dans l’arborescence des éléments UML.

1. Questions :

----------------------------------------------------------------------------------------------------------------------

Rencontre avec prof 25 décembre – Questions et explication de l’outil MDE

1. Constructeur dans le modèle UML ou les templates de transformation ?

**Réponse** : généralement on ne fait pas les constructeurs dans le modèle UML, mais si on a besoin des constructeurs spéciaux on va les faire dans le modèle UML. Donc on va les faire dans les templates de transformation selon l’architecture.

Pour l’architecture clean on va faire constructeur par défaut (exigé par EF Core) et constructeur pour initialiser les propriétés de la classe.

1. Navigation entre deux classes et le choix ou mettre la clé étrangère ?

**Réponse** : On va faire la navigation dans les deux sens.

Le choix de la clé étrangère c’est une décision de conception donc c’est la paramétrisation (stereotype). On cas d’une relation 1—1 on doit choisir ou mettre la clé étrangère. Donc dans une association on va choisir « associationEnd » qui va avoir le stereotype « HasForeignKey ».

1. Processus de la spécification de l’architecture :

* Dans chaque projet (ex : Core) voir les « stereotypes » ou les « tagged values » qu’on est besoin. (Ex core : Entity, aggregateRoot, valueObject)
* La définition ou la réutilisation des bibliothèques (d’autres noms : accélérateurs, sous architecture…) par exemple accélérateur des design patterns.
* Donc la possibilité d’implémenter plusieurs spécifications d’architecture dans une seule solution. Par exemple on a l’architecture principale et l’architecture clean et on peut réutiliser d’autres spécifications d’architecture (biblio, accélérateur, PDM, profil, morceaux…) par exemple pour les design pattern.
* Donc profil (clean) est composé de sous profils (design pattern, cas des test).
* Exemple d’accelereteur : design pattern observer je vais l’appliqué. Contient les stereotype « subject », « observer ».
* Why concept has design concerns ??
* The concept specify which compnent will be generated. For example if concept has type of classifier(class) it will generate a class, but design concerns will tell where the mark will be done in the UML element in VP. For example FK (foreign key) will be marked in association end.

----------------------------------------------------------------------------------------------------------------------------

Use saxon as the transformer in java (Doesn’t work for now):

1. Add saxon.jar to classpath

<https://stackoverflow.com/questions/50232557/visual-studio-code-java-extension-how-to-add-a-jar-to-classpath>

1. Use saxon in Java

<https://stackoverflow.com/questions/2968190/how-to-select-saxon-transformerfactory-in-java>

Create custom ribbon

<https://knowhow.visual-paradigm.com/openapi/custom-ribbon/>