ClassiPy

V1.0.0

RAPPORT DU PROJET UML : CLASSIPY V1.0.0

El Amrani Ilyas – El Jaouhari Mohamed

ENSA
ÉCOLE NATIONALE DES SCIENCES APPLIQUÉES
KHOURIBGA

PROJET UML : ClassiPy

Réalisé par :

El Jaouhari Mohamed *El Amrani Ilyas*

Encadré par:

Pr. Lamghari Nidal

Filière :

Informatique et ingénierie des données

Année universitaire:

2022/2023

Sommaire

I. Introduction :

L'utilisation de l'apprentissage automatique devient de plus en plus fréquente et derrière ces applications des algorithmes compliquées de point de vu mathématique, mais ils sont utilisé pour des taches d'automatisation surtout. Ces technologies son implémentés par une variété de langages, notamment Python, et utilisent beaucoup des ressources.
Dans notre application ClassiPy, nous essaierons de faciliter aux utilisateurs, surtout ceux qui n'aime pas coder mais sont passionnés par ce domaine la création, l'entrainement, le test, l'utilisation et la gestion de deux types de réseaux de neurones, tous on donnant l'utilisateur à chaque fois le choix des paramètres, de jeu de données et l'avertir en cas d'erreur ou de mauvaise utilisation de notre application simple.

II. 1ère version :

A. Diagramme des cas d'utilisation :

Permet d'Identifier les limites du système, les acteurs, les cas d'utilisation, ajout des relations entre cas d'utilisation et les classer par ordre d'importance. Nous avons les cas d'utilisations suivantes :

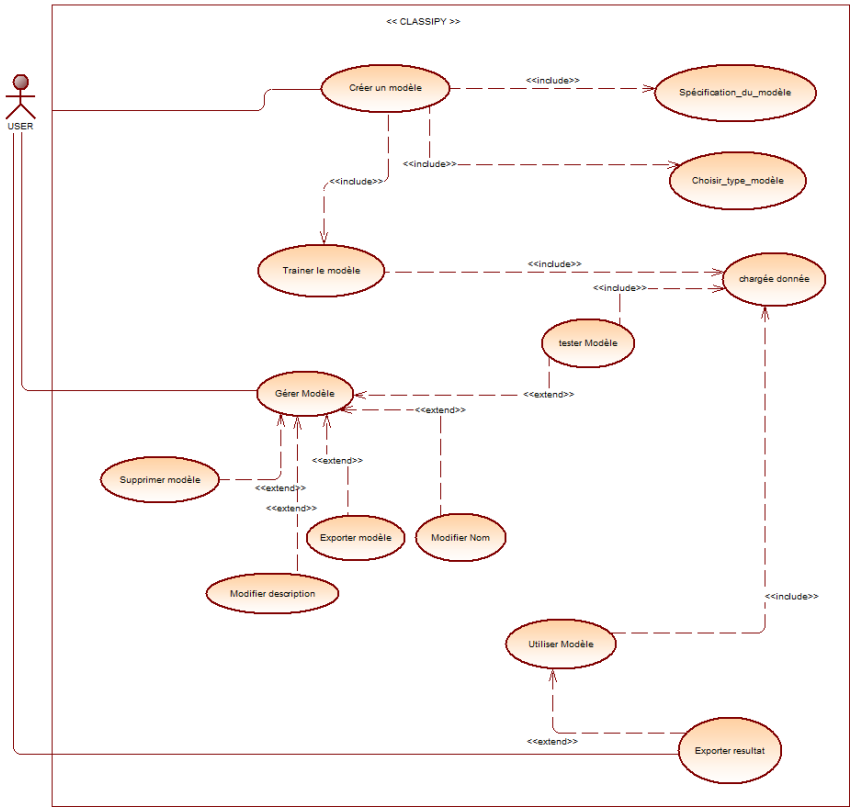
- Créer Modèle
- Spécification du modèle
- Choisir type de modèle
- Trainer le modèle
- Chargée les données
- Tester le modèle
- Gérer le modèle
- Supprimer le modèle
- Modifier la description
- Exporter le modèle
- Modifier le Nom
- Utiliser le modèle
- Exporter Résultat

Pour les acteurs, on a 1 seule acteur qui est l'USER.

Pour les relations :

- Créer un modèle inclus la spécification du modèle, choisir le type de modèle et trainer le modèle.
- Trainer modèle inclus chargée le modèle.

- Tester modèle inclus chargée donnée et elle est inclus comme option dans Gérer modèle.
- Gérer Modèle contient plusieurs options : Supprimer le modèle, Modifier la description, Exporter le modèle et Modifier le Nom.
- Utiliser modèle inclus chargée donnée, et comprend Exporter résultat.

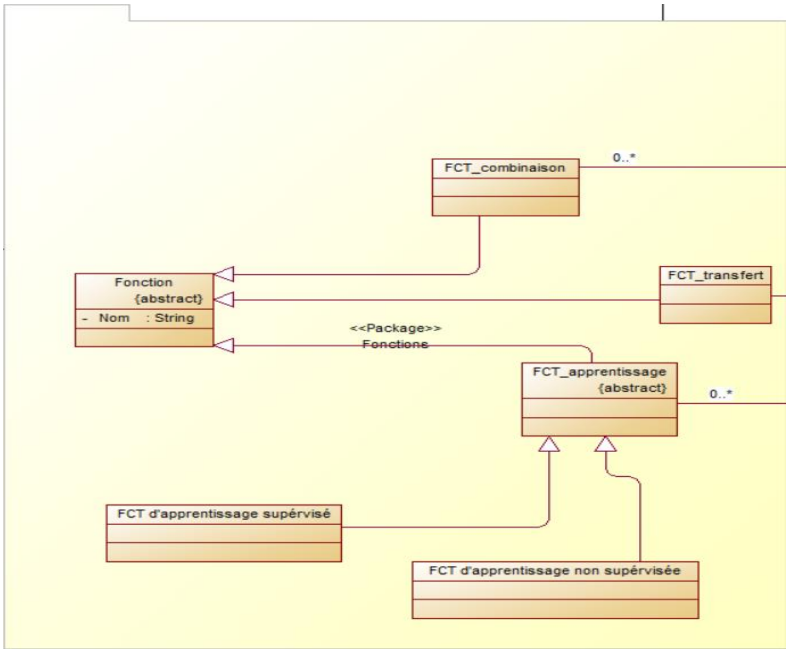


B. Modèle du domaine :

Notre modèle du domaine est constitué de 4 packages :

1. Package de fonction :

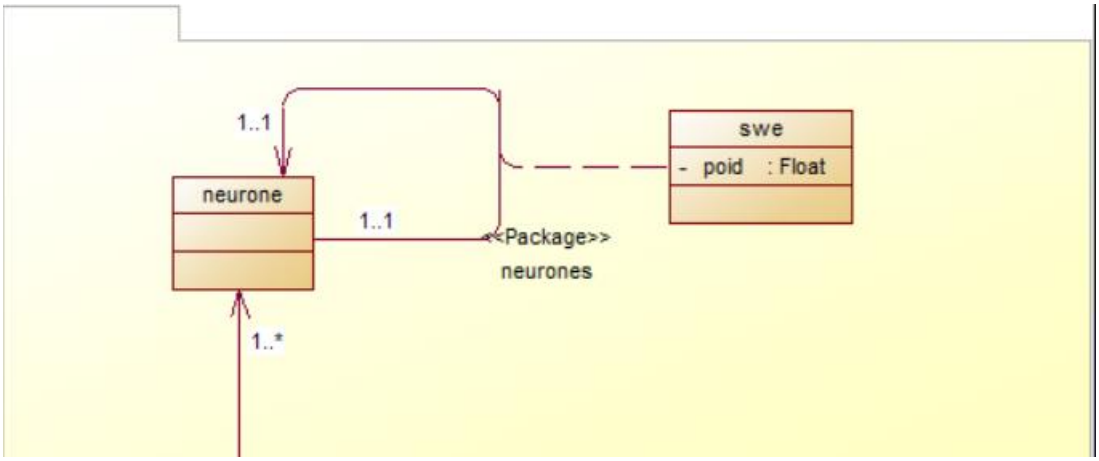
Ce package contient les classes représentant les types de fonctions à utiliser.



2. Package de neurone :

Ce package contient 2 classes :

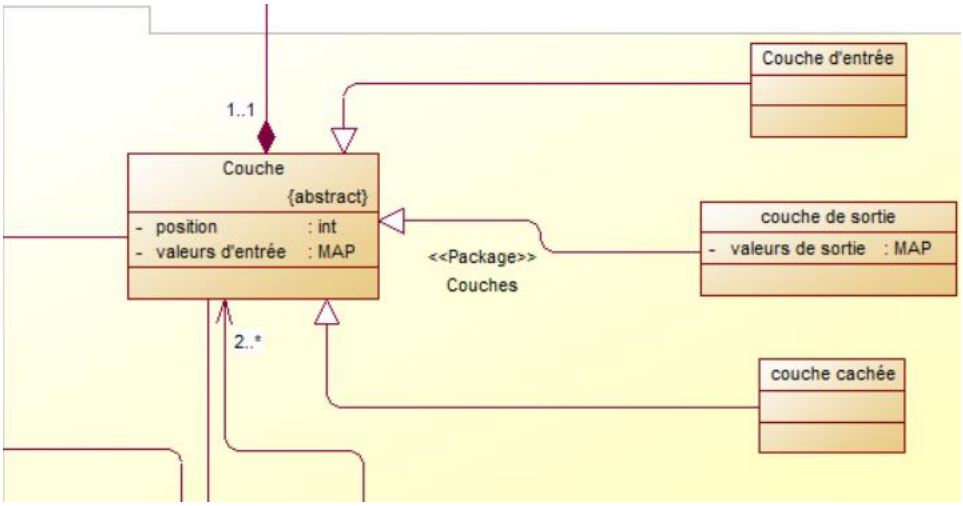
- Une de neurone
- Une des poids de chaque neurone.



3. Package de couche :

Ce package contient 4 classes :

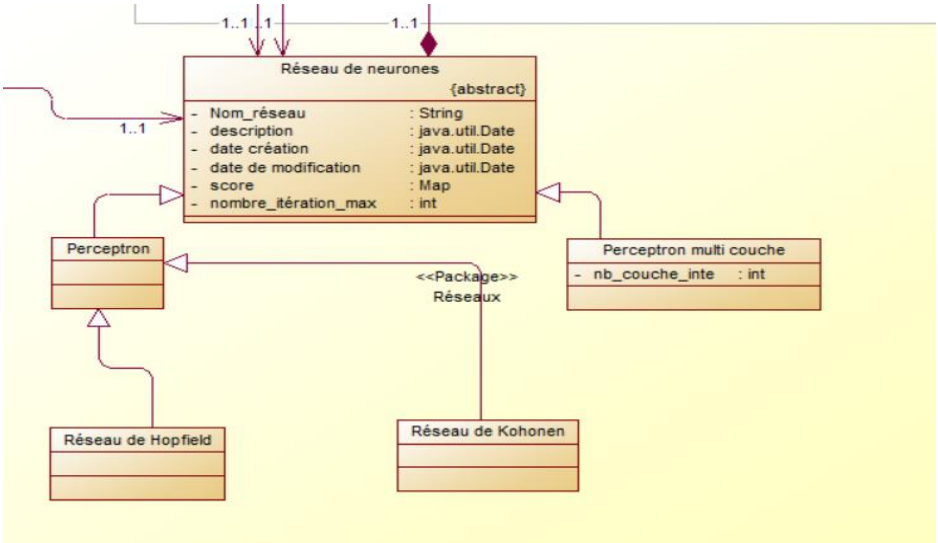
- Couche (classe mère) : présente la couche avec sa position (1ère couche, 2ème couche, etc.)
- Couche d'entrée : La première couche du réseau
- Couche de sortie : La dernière couche du réseau
- Couche cachée : L'ensemble des couches cachées



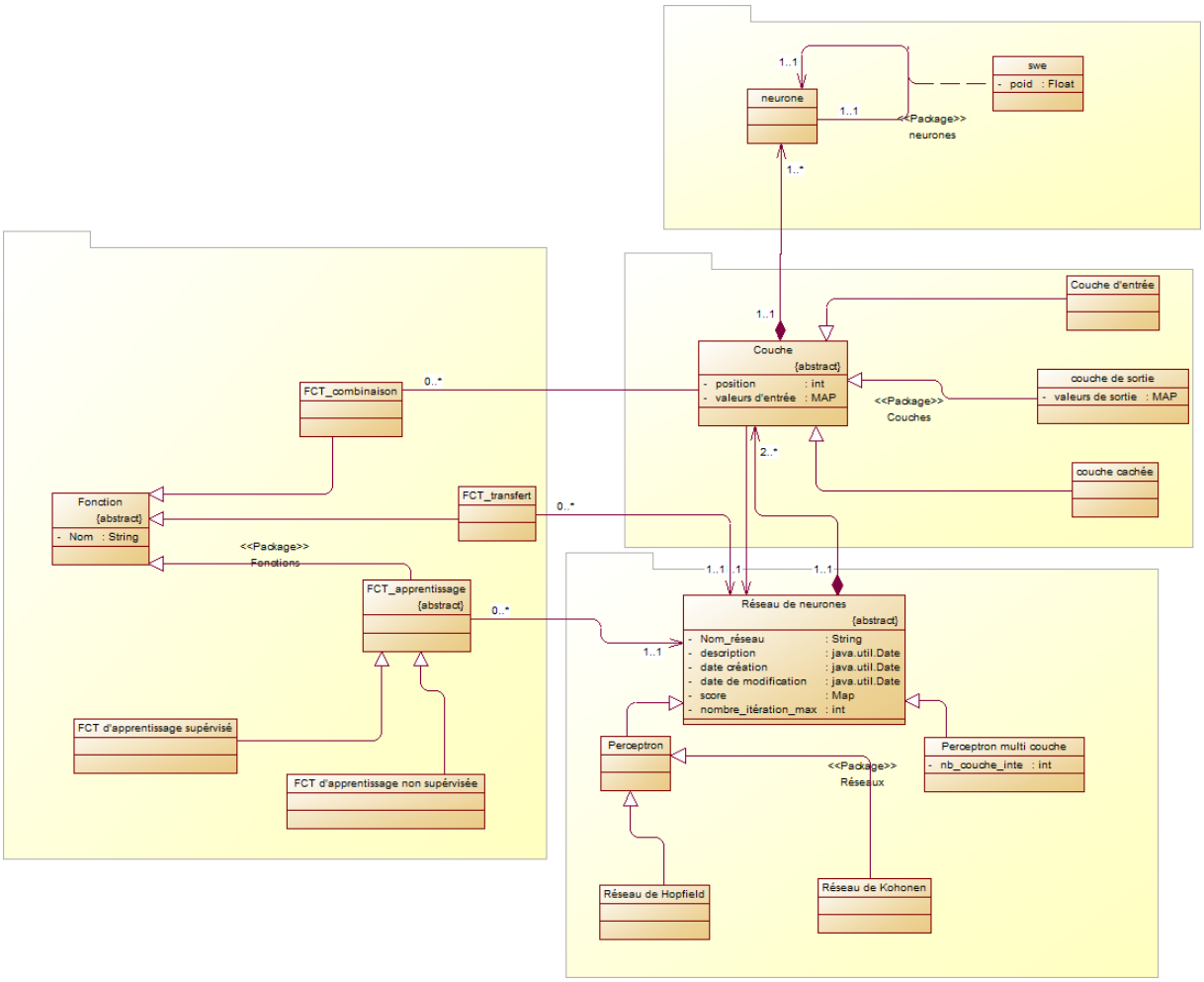
4. Package de réseau de neurone :

Ce package contient 4 classes :

- Réseau de neurones : présente le réseau de neurone toute entier.
- Perceptron : le réseau d'un seul perceptron
- Réseau de hopfield : hérite de la classe Perceptron, présente le réseau de hopfield.
- Réseau de Kohonen : hérite de la classe Perceptron, présente le réseau de Kohonen.



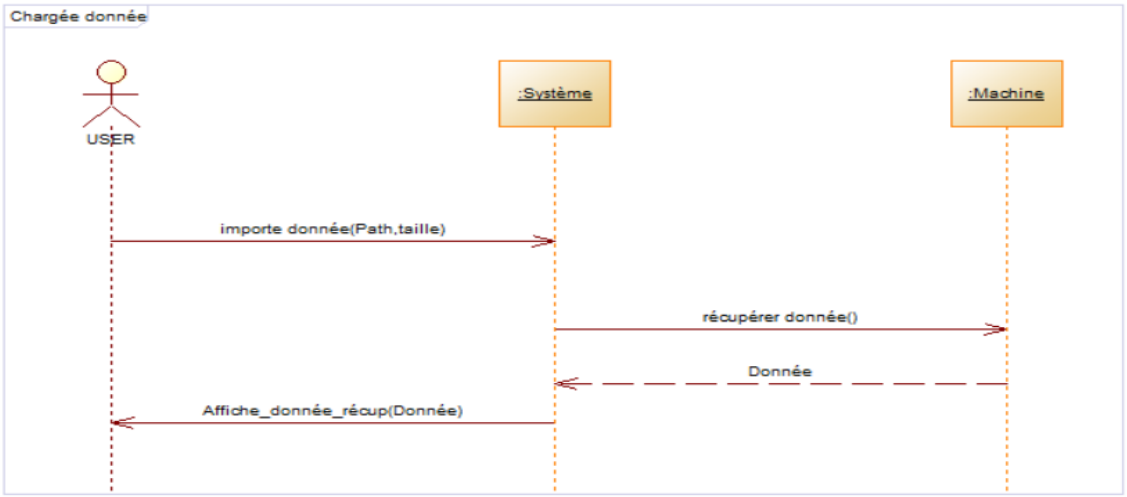
5. Modèle du domaine tout entier :



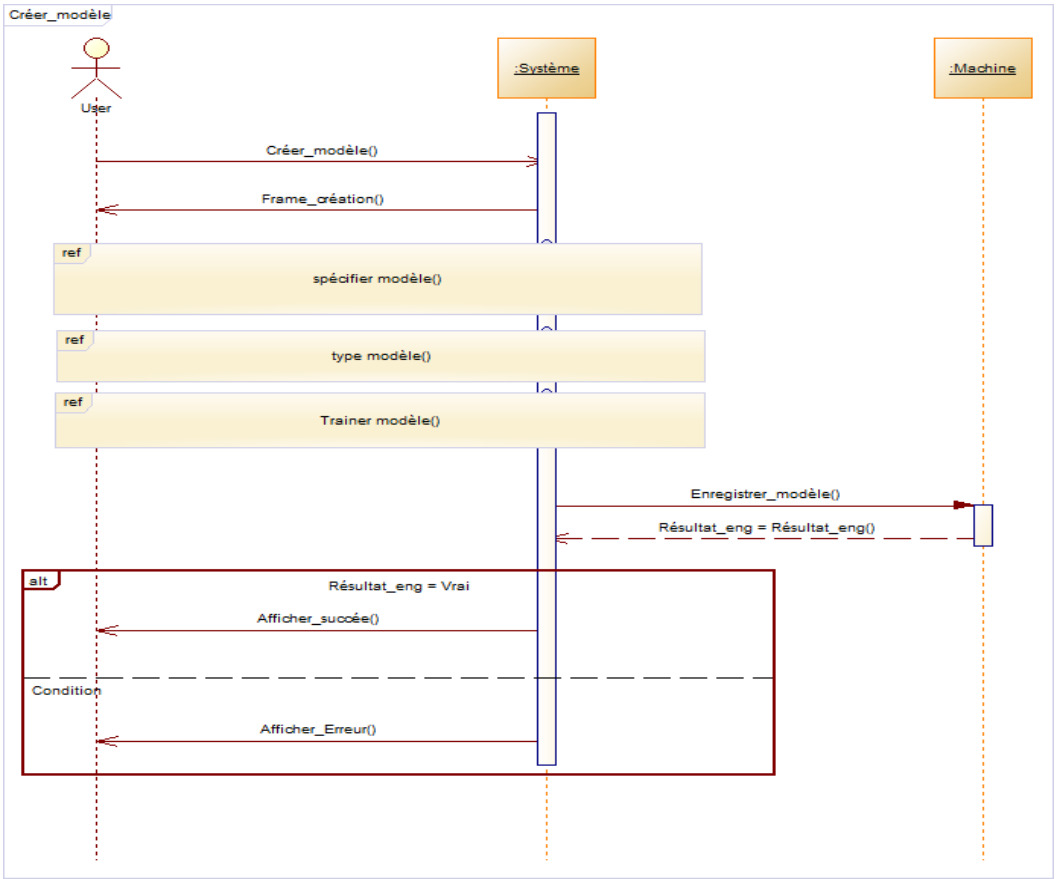
C. Diagrammes de séquences système :

Pour chaque cas d'utilisation, on a créé diagramme de séquence pour chaque cas d'utilisation. La classe système sera détaillé dans une autre version.

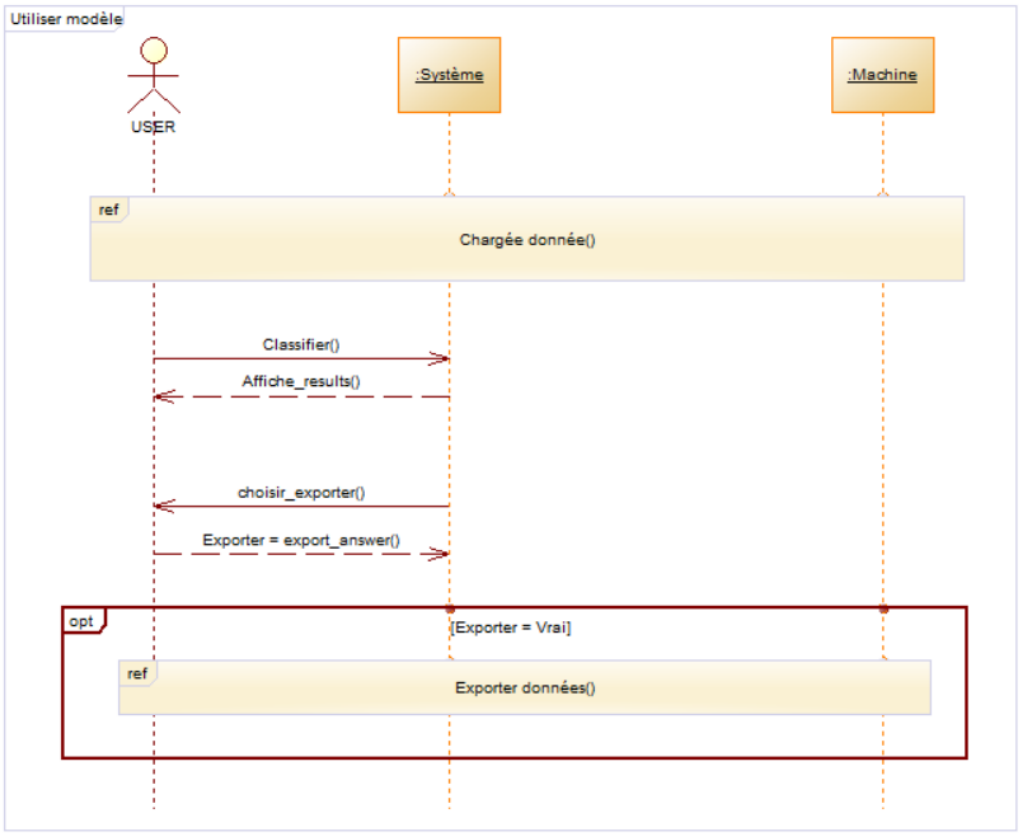
1. Chargée donnée : Nécessaire pour l'entrainement et l'utilisation du modèle.



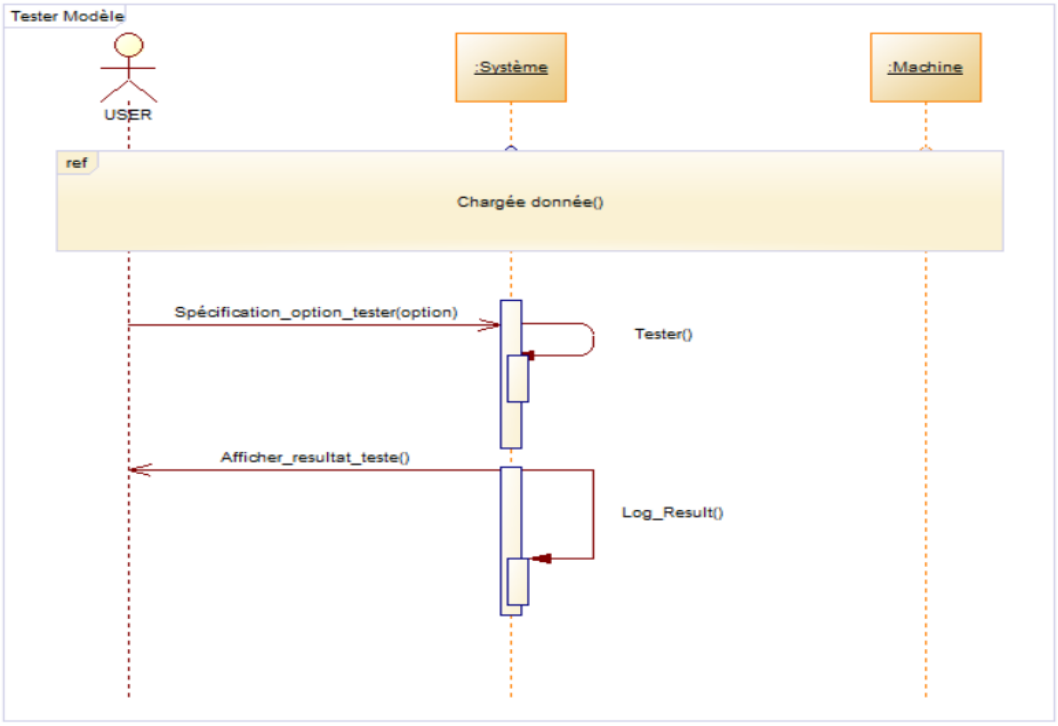
2. Créer modèle :



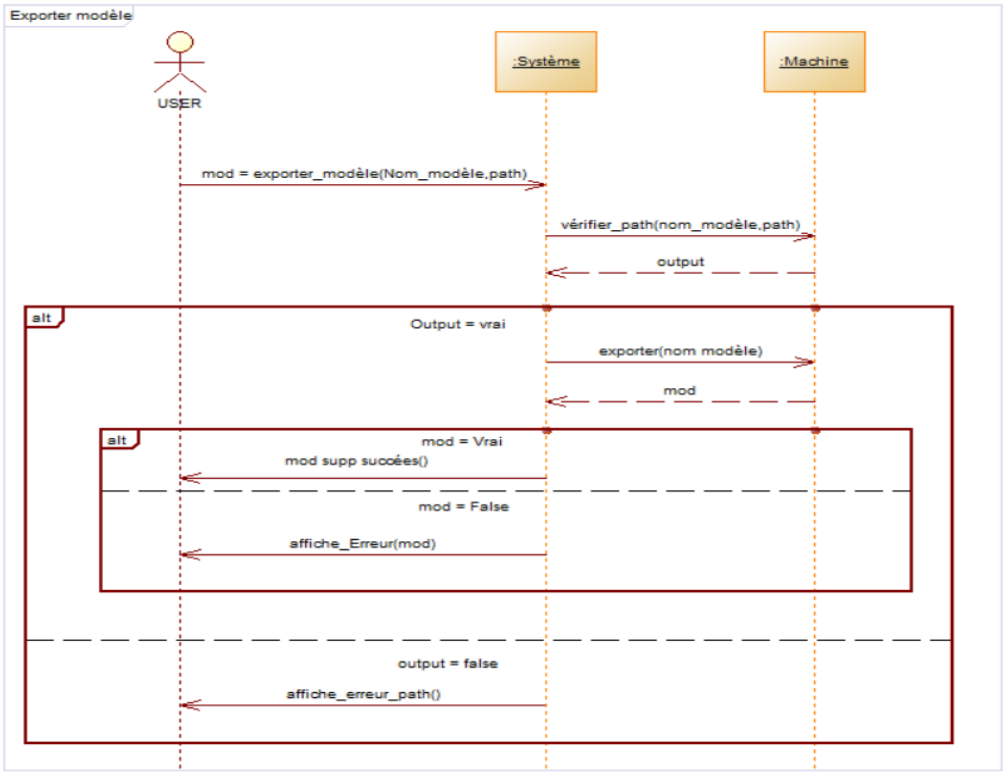
3. Utiliser le Modèle : Utilisation et exportation des résultats selon le choix de l'utilisateur



4. Tester le Modèle : Test et enregistrement/exportation des résultats.

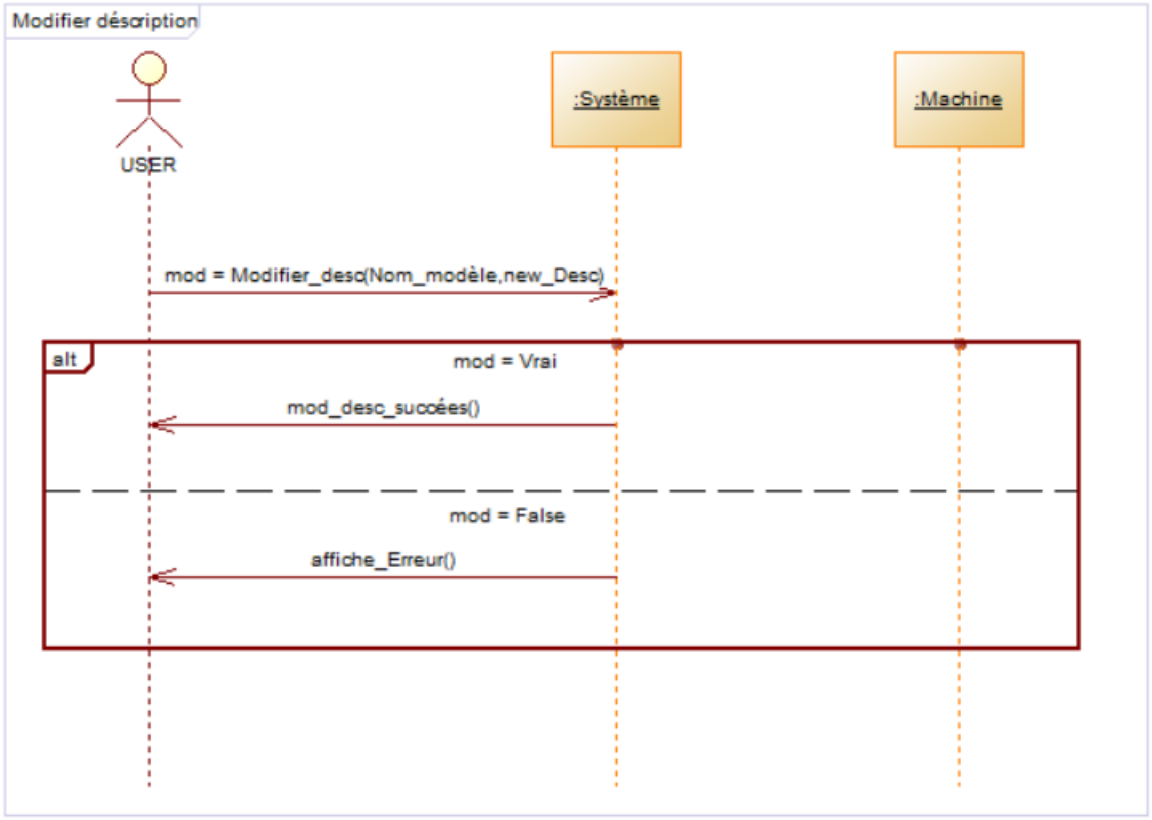


5. Exporter un modèle :

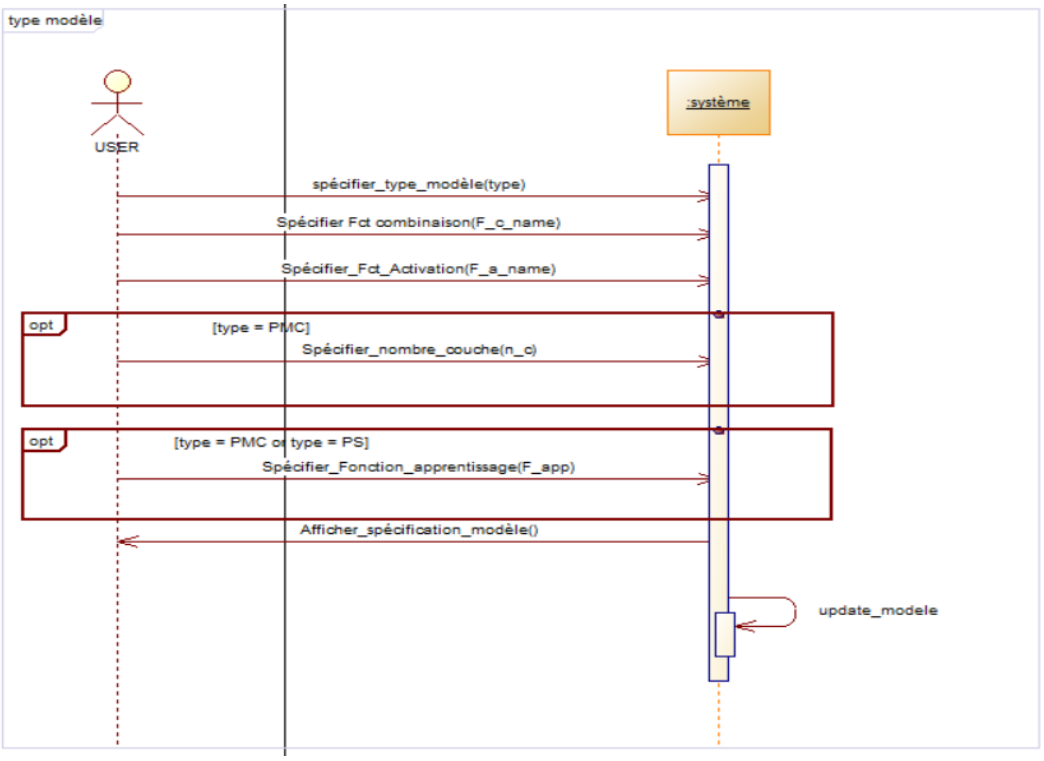


6. Modifier Nom :

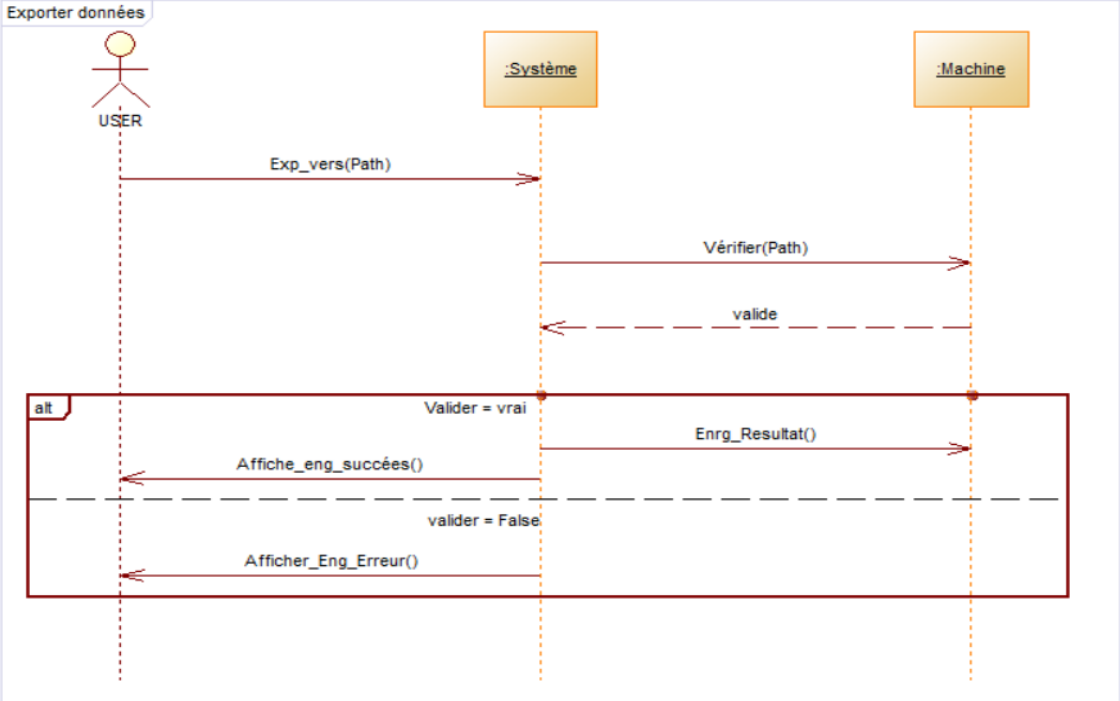
7. Modifier Description :



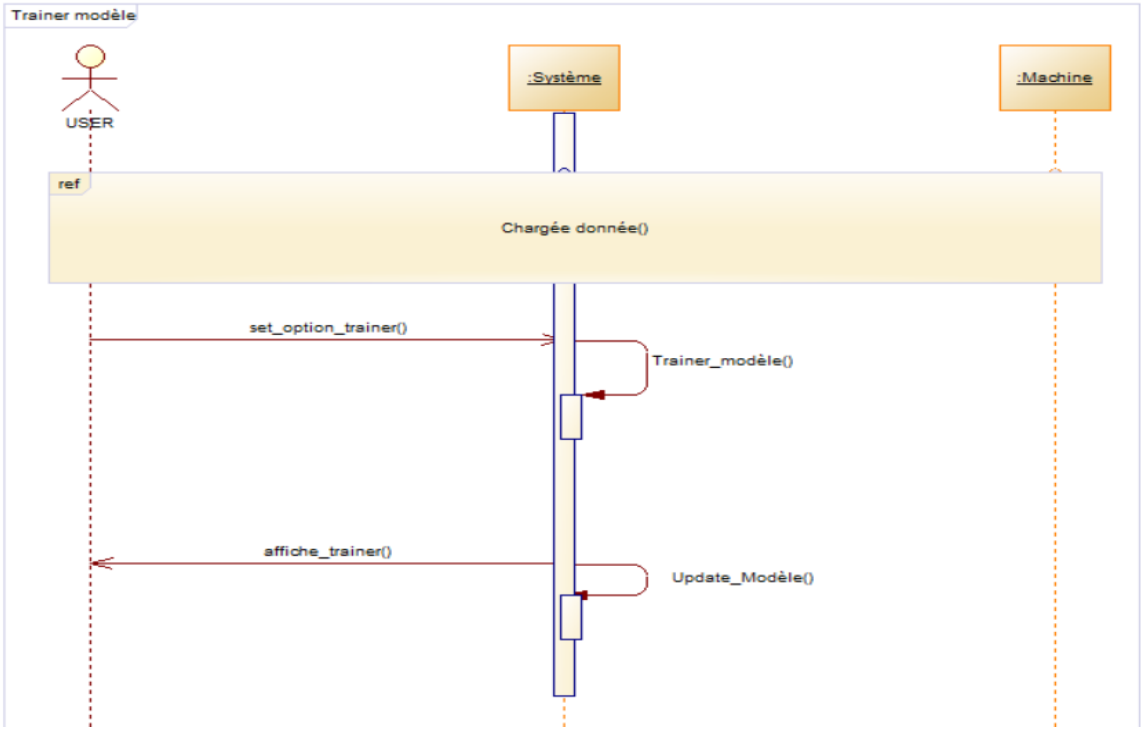
8. Spécifier type du modèle :



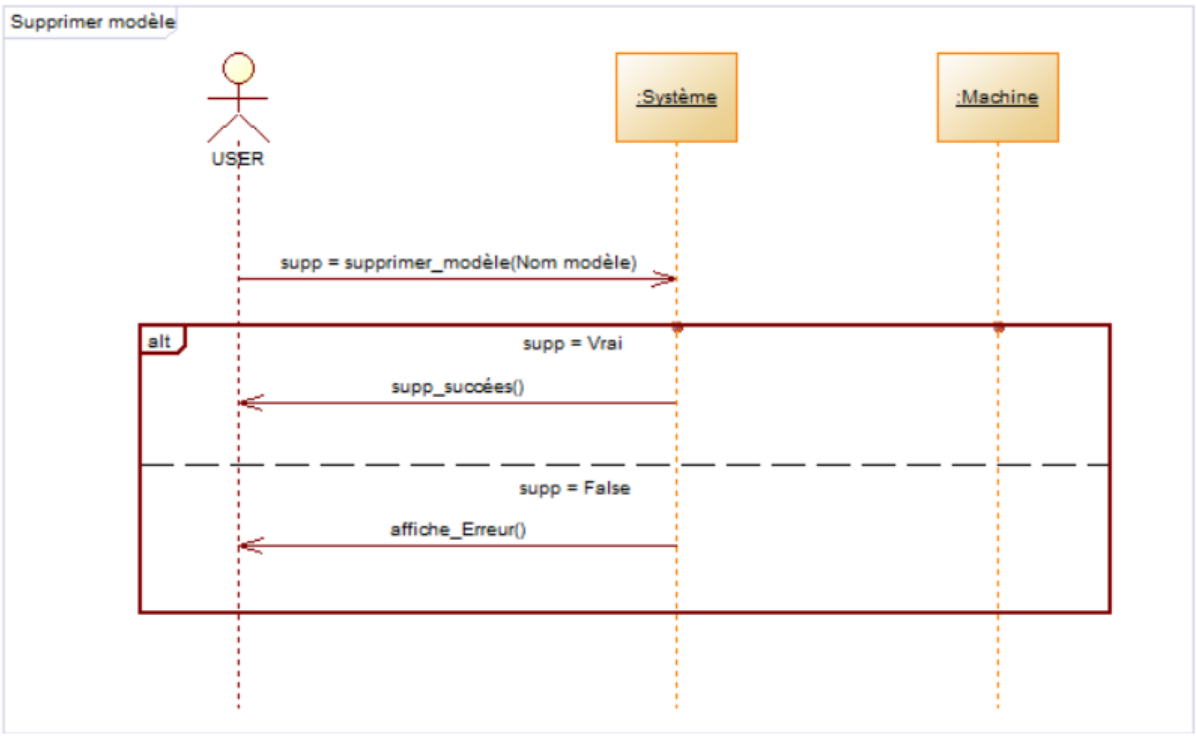
9. Exporter les données: Après un test ou une prédction



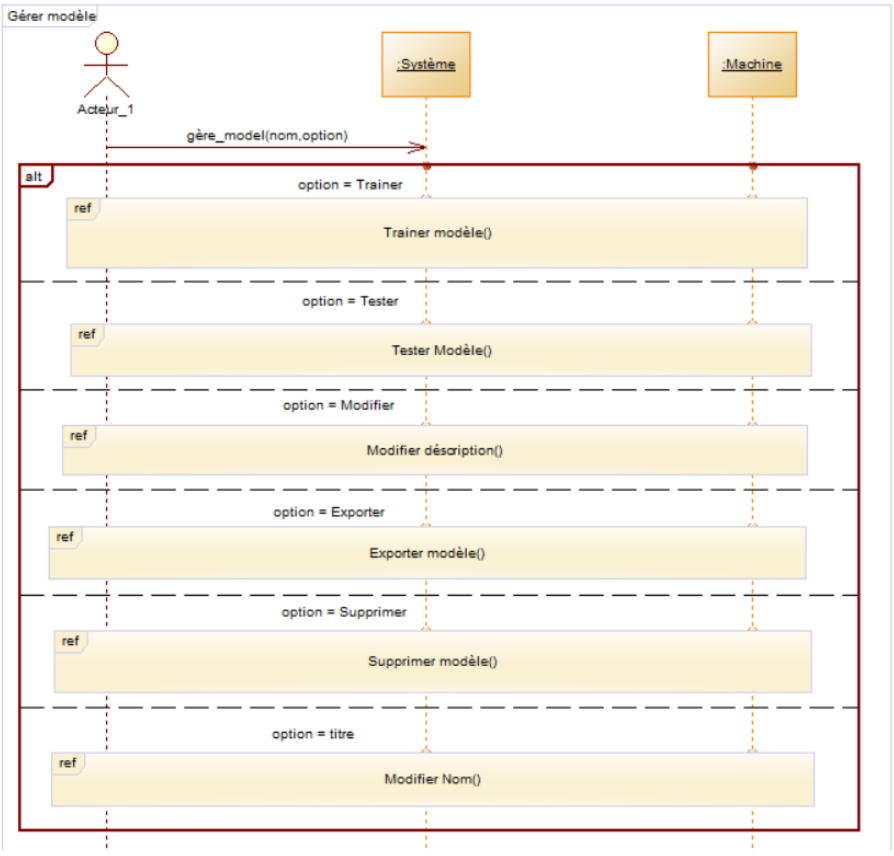
10. Trainer le modèle :



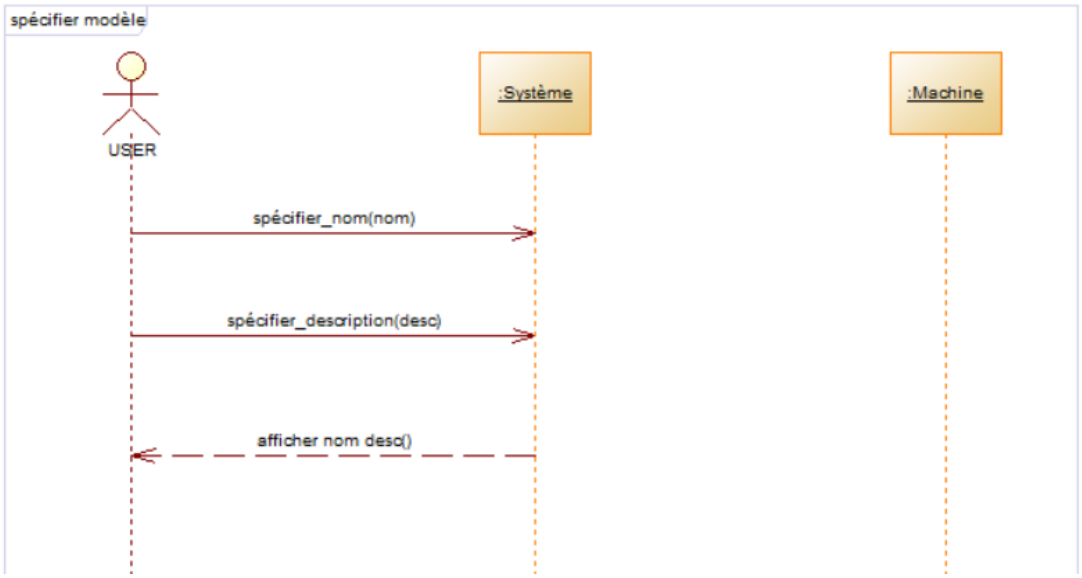
11. Supprimer le Modèle :



12. Gérer le modèle :



13. Spécifier le modèle :



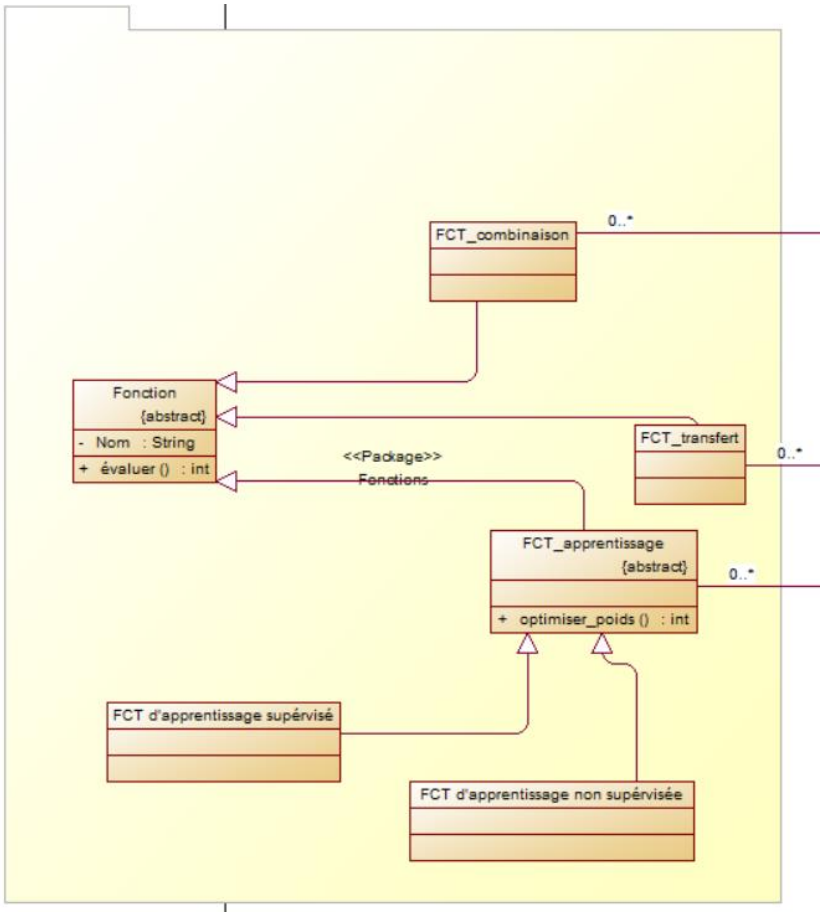
III. 2ème version :

Dans Notre 2ème version, on a ajoutée des modifications sur les diagrammes suivants :

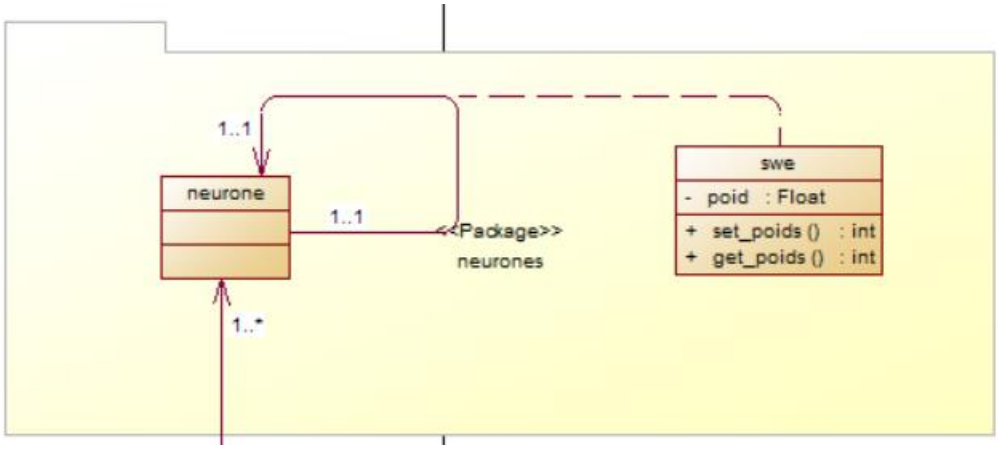
A. Diagramme de classe participante :

Pour les 4 package, on a pour :

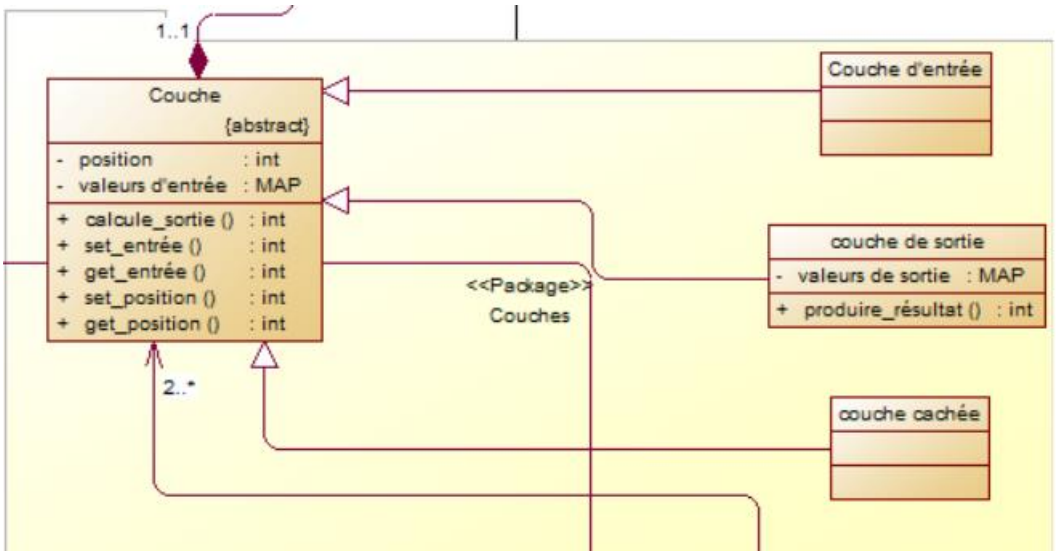
1. Package des fonctions :



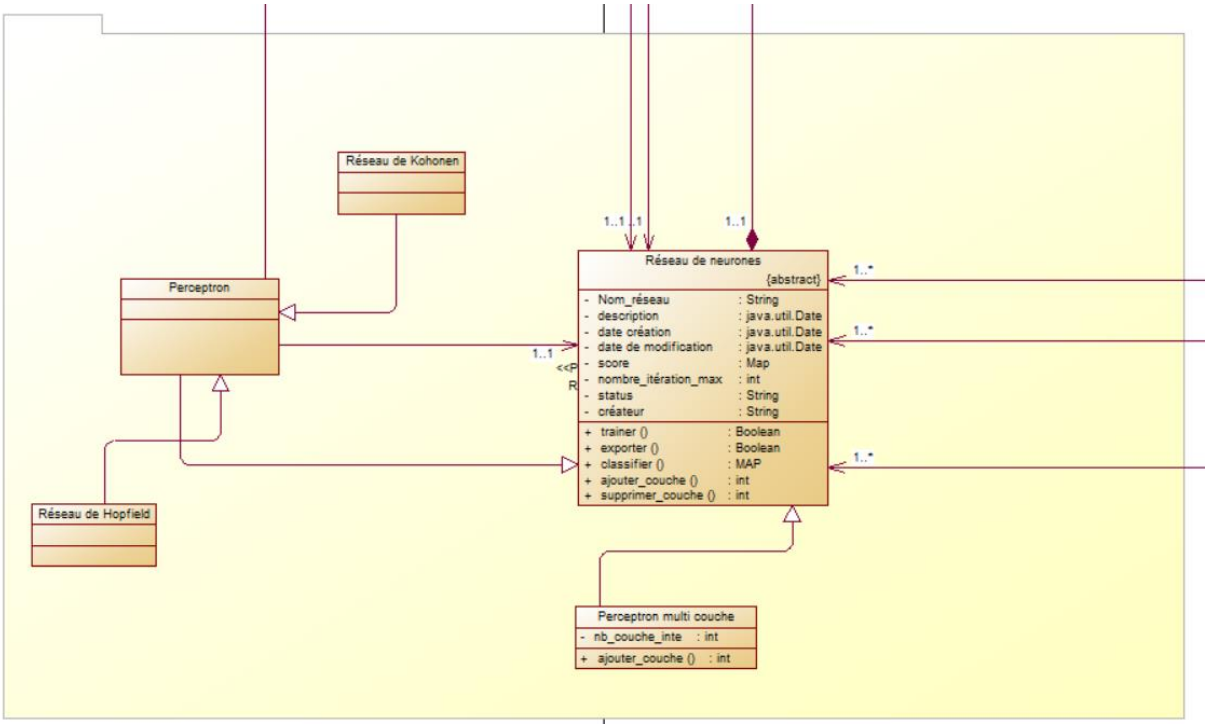
2. Package des neurones :



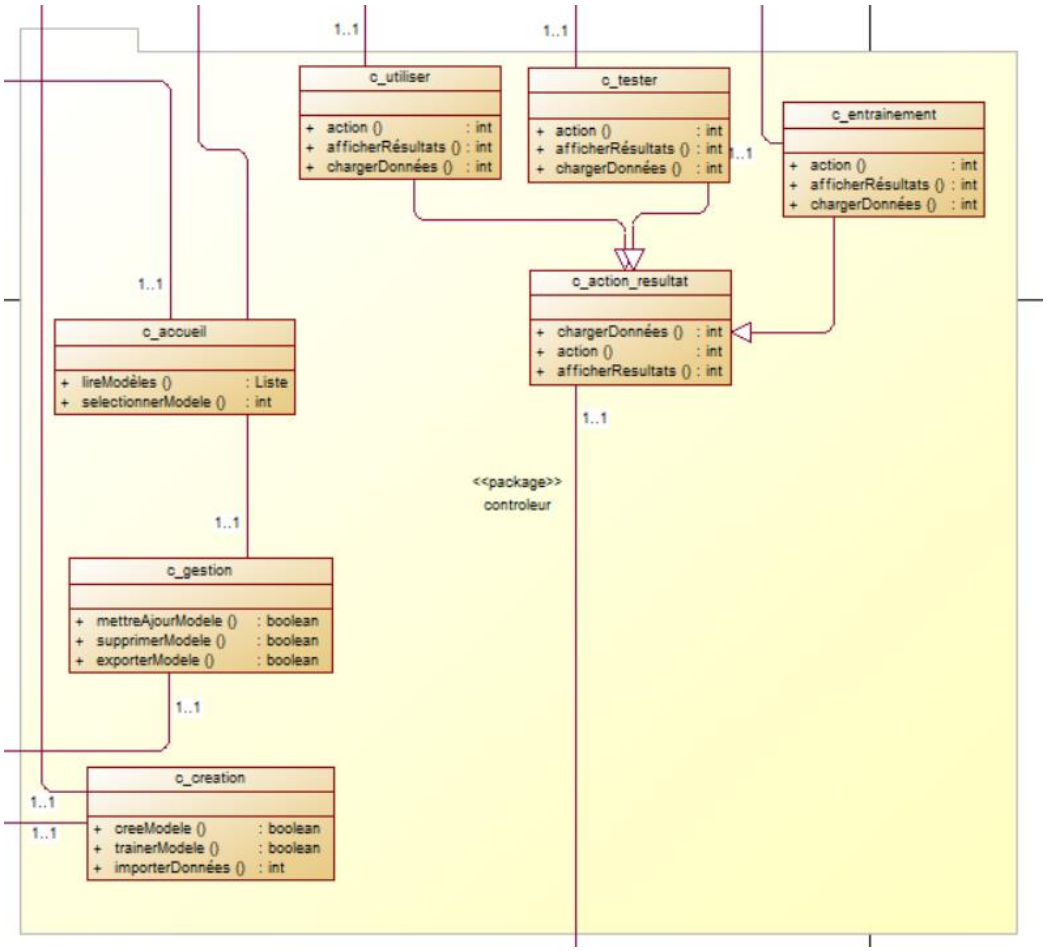
3. Package des couches :



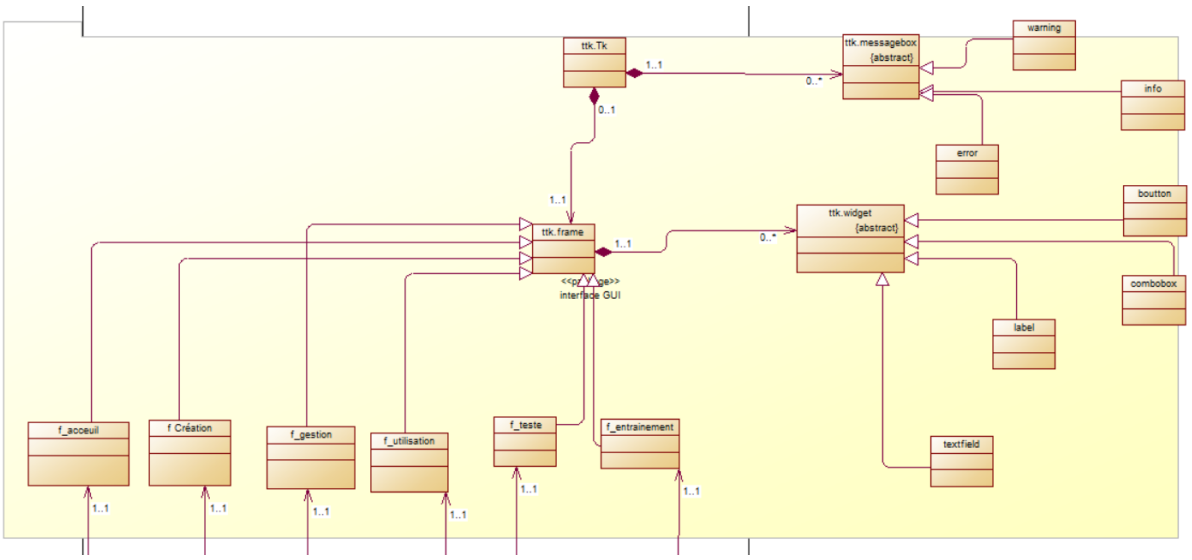
4. Package de réseau :



5. Package des contrôleurs :



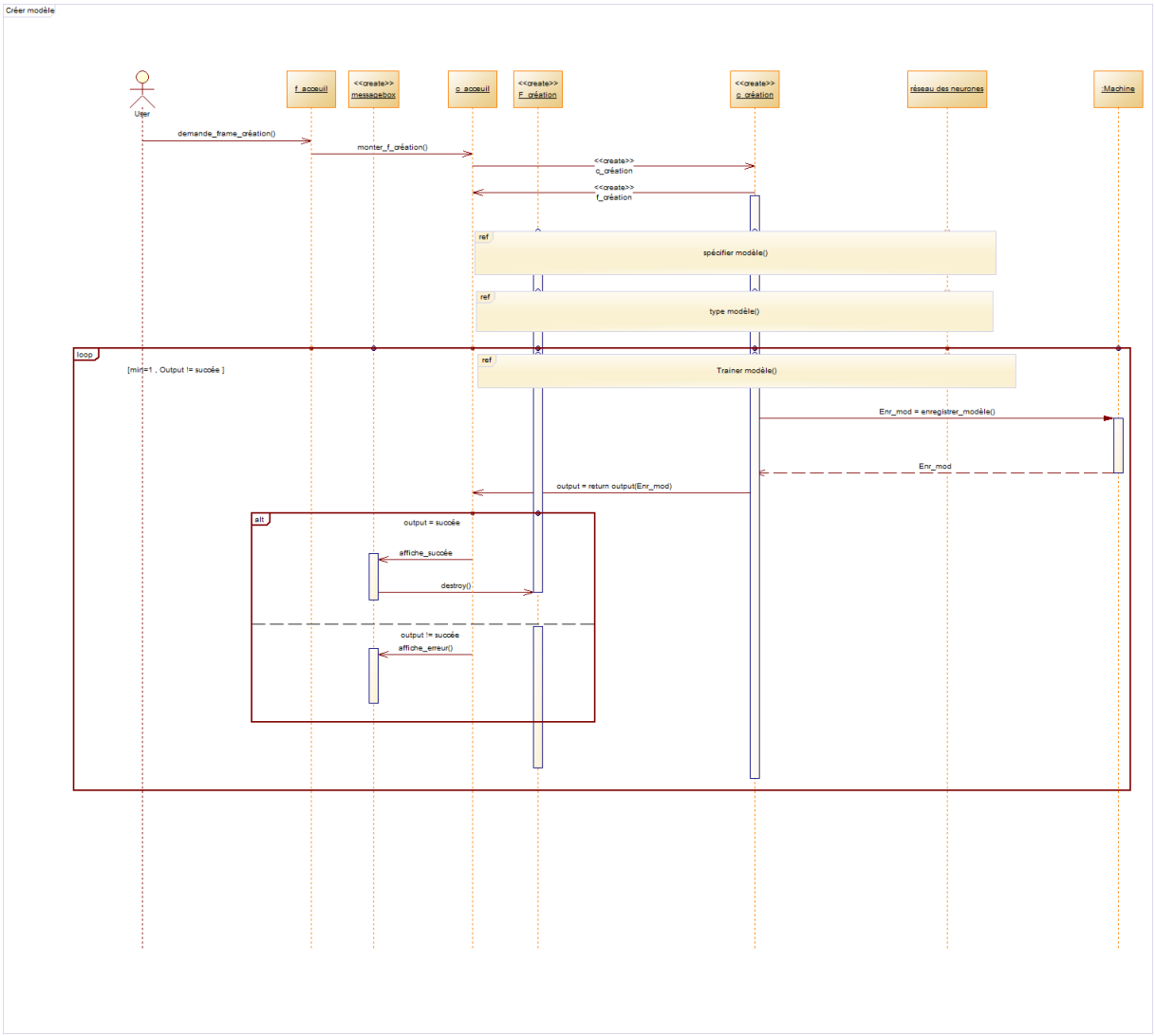
6. Package d'interface GUI :



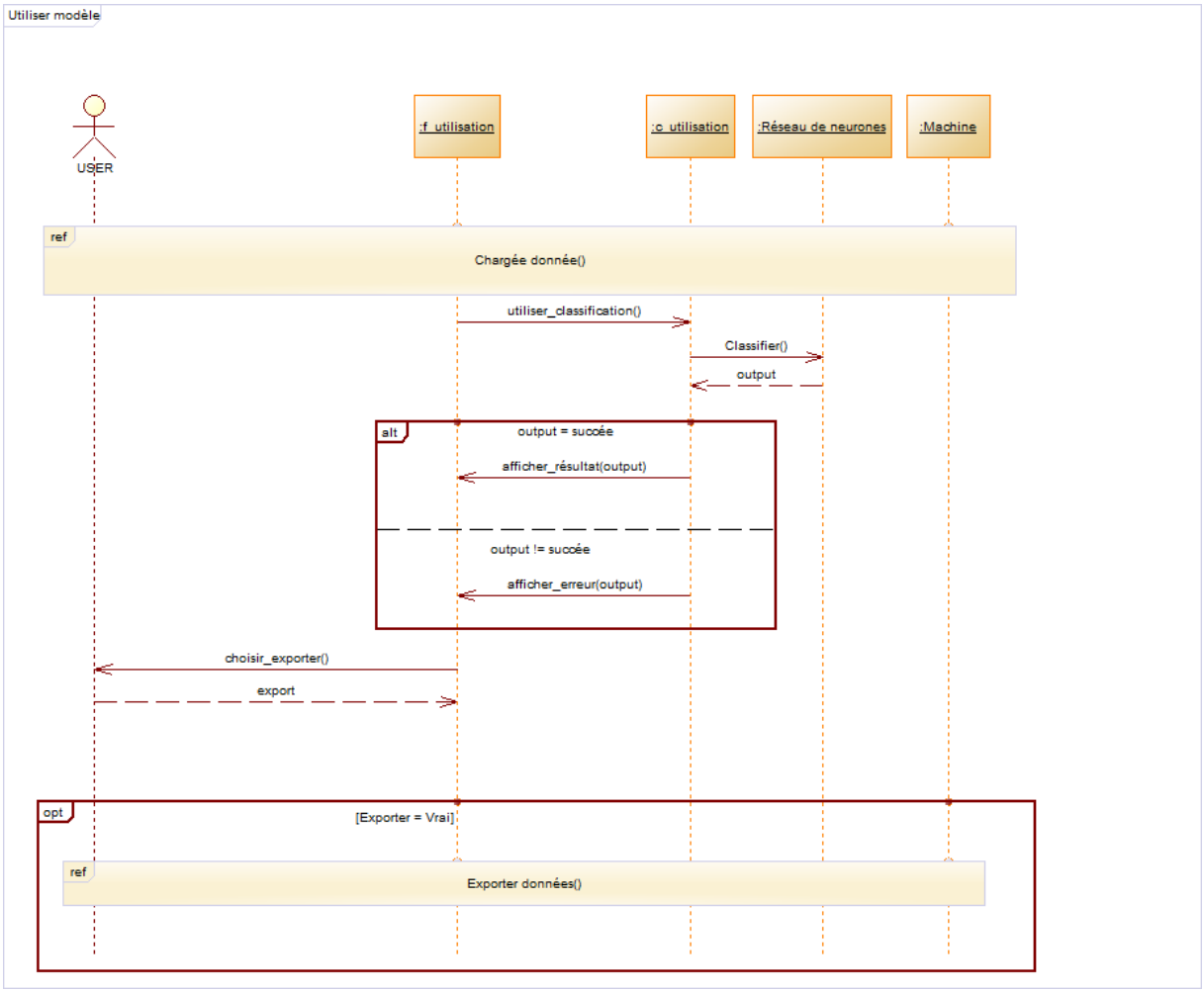
B. Diagramme de séquences système :

Dans cette partie, on détaillera la classe système de chaque diagramme de séquence.

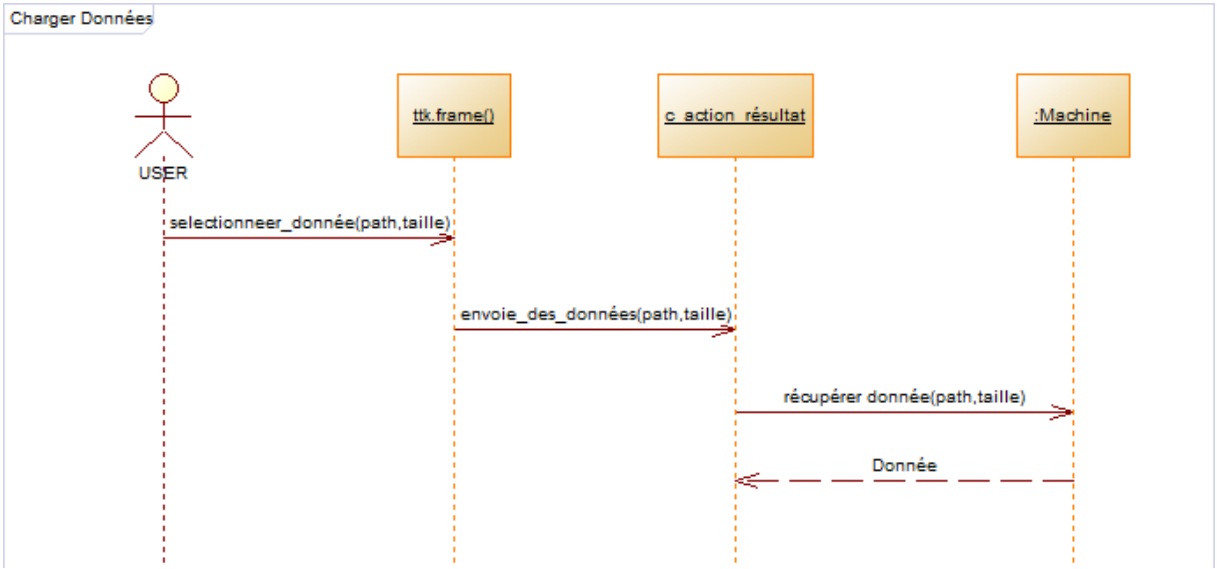
1. Créer modèle :



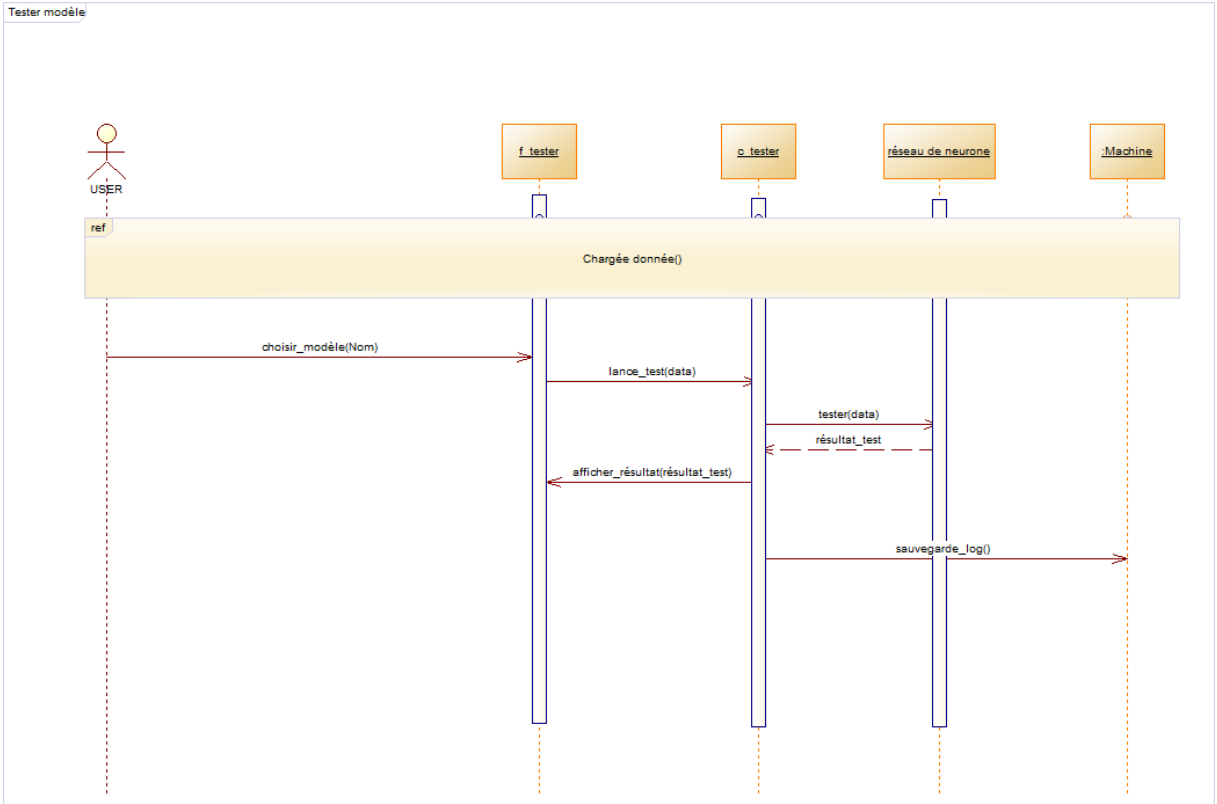
2. Utiliser le modèle :



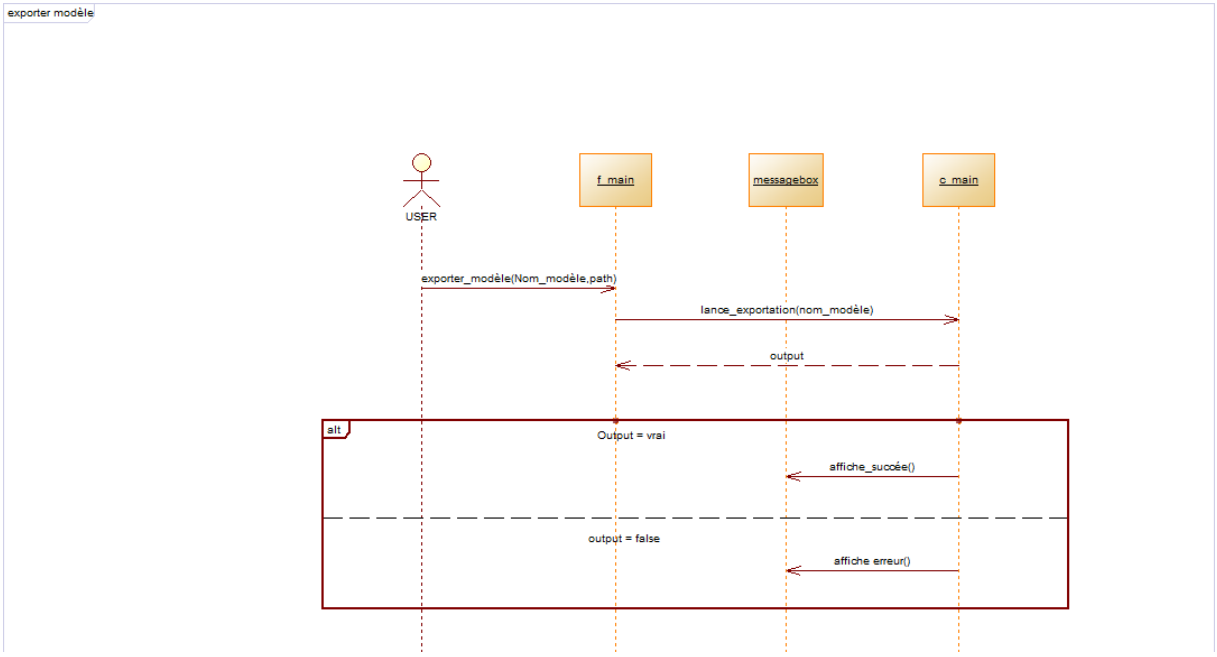
3. Chargée des données :



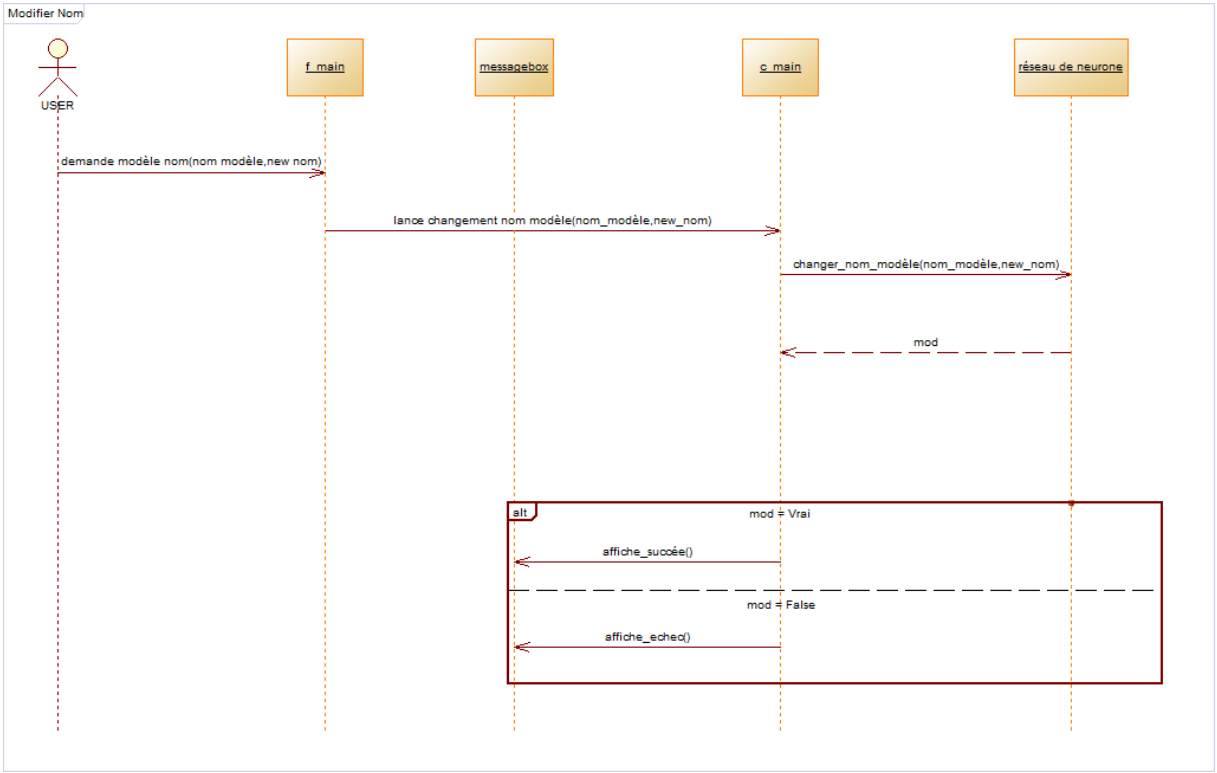
4. Tester le modèle :



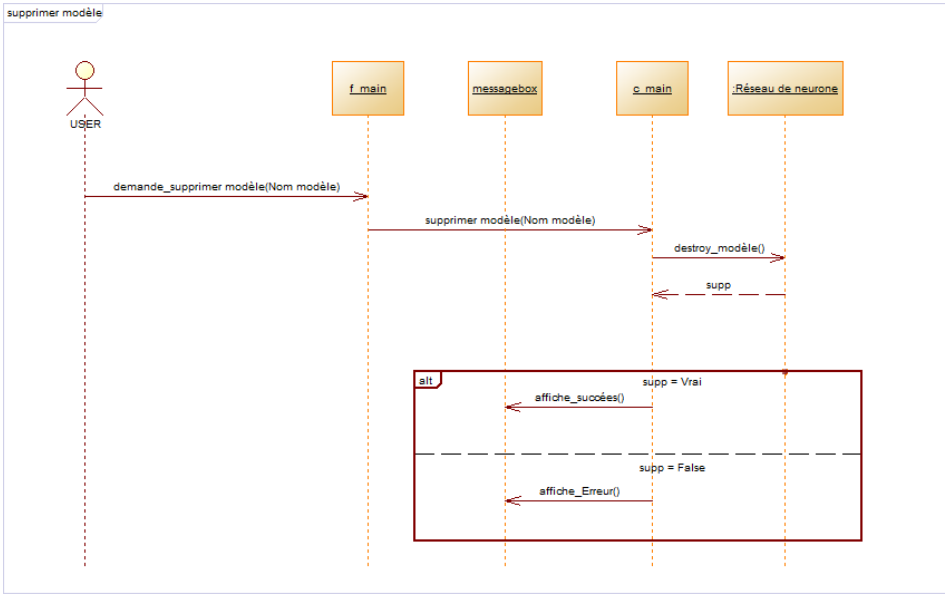
5. Exporter le modèle :



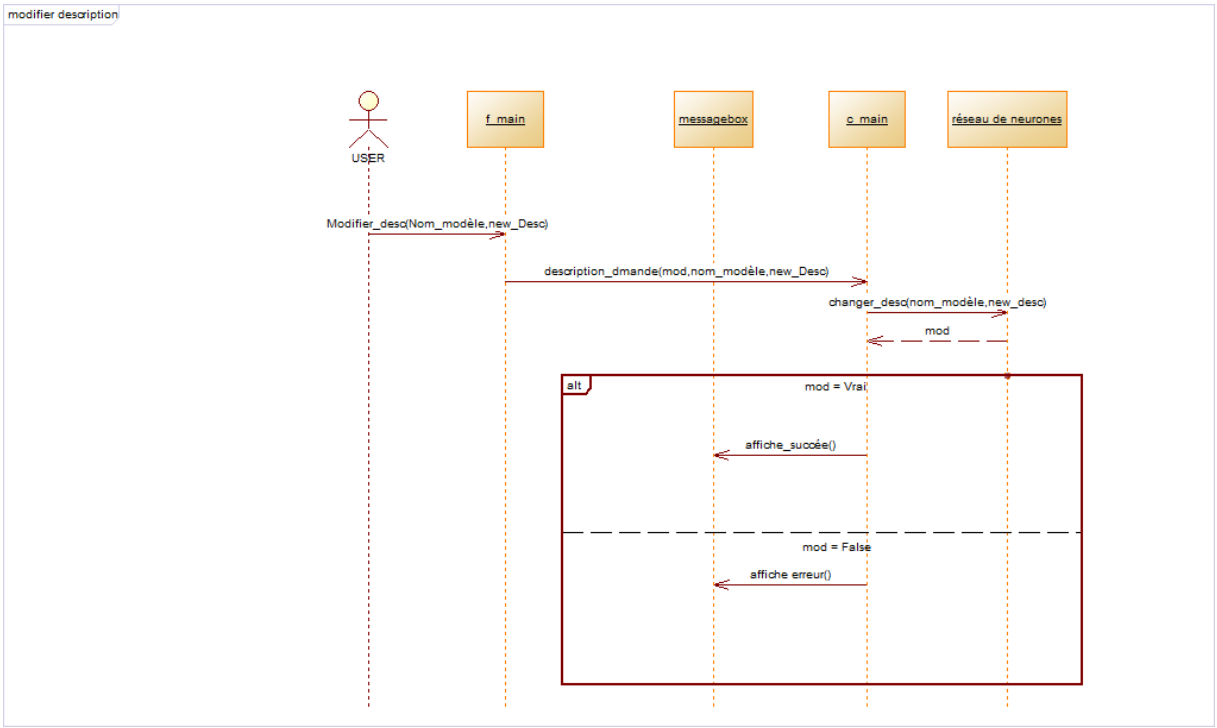
6. Modifier Nom :



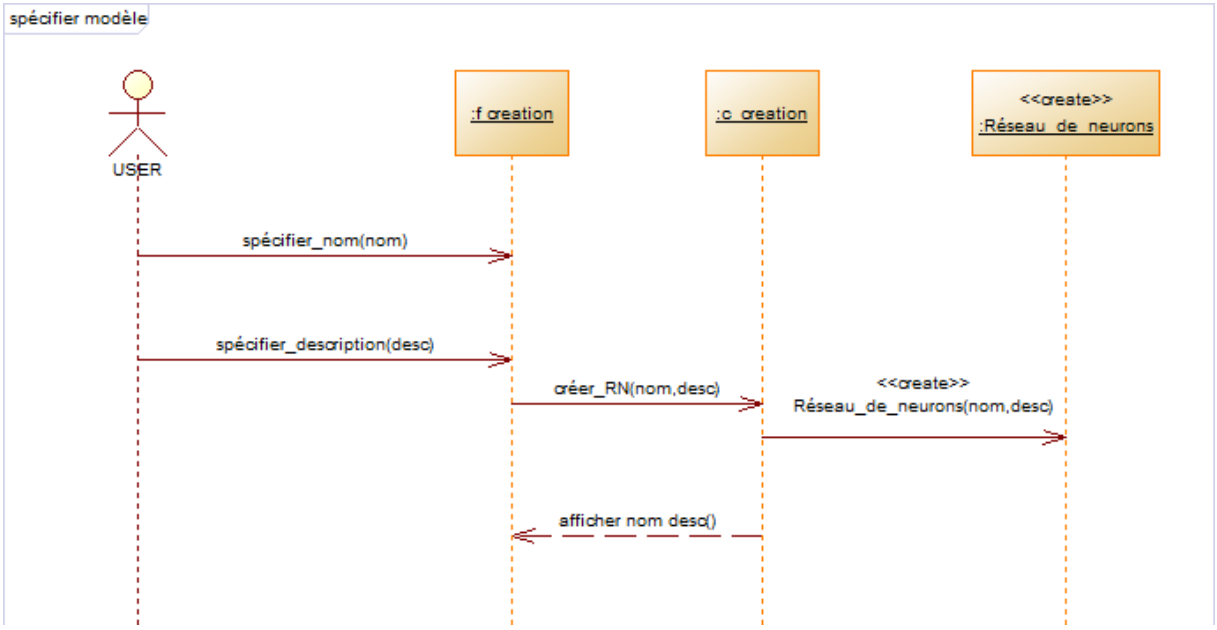
7. Supprimer le Modèle :



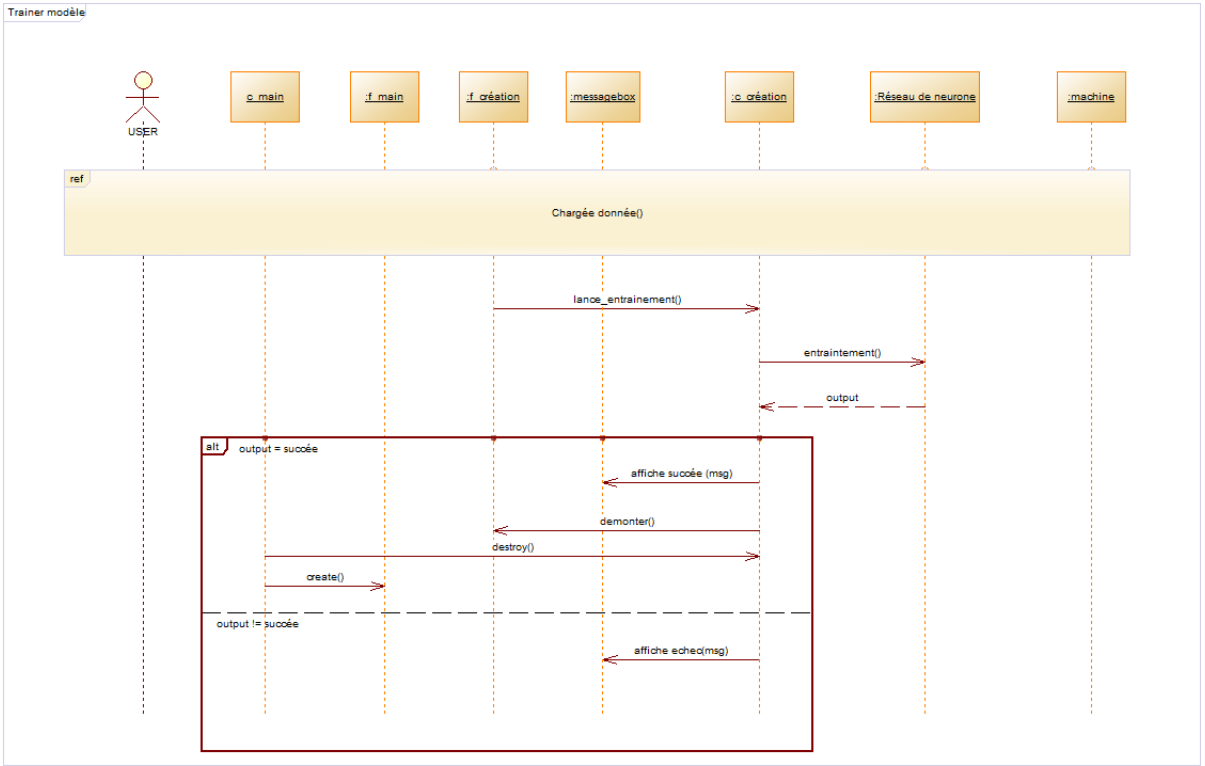
8. Modifier la description :



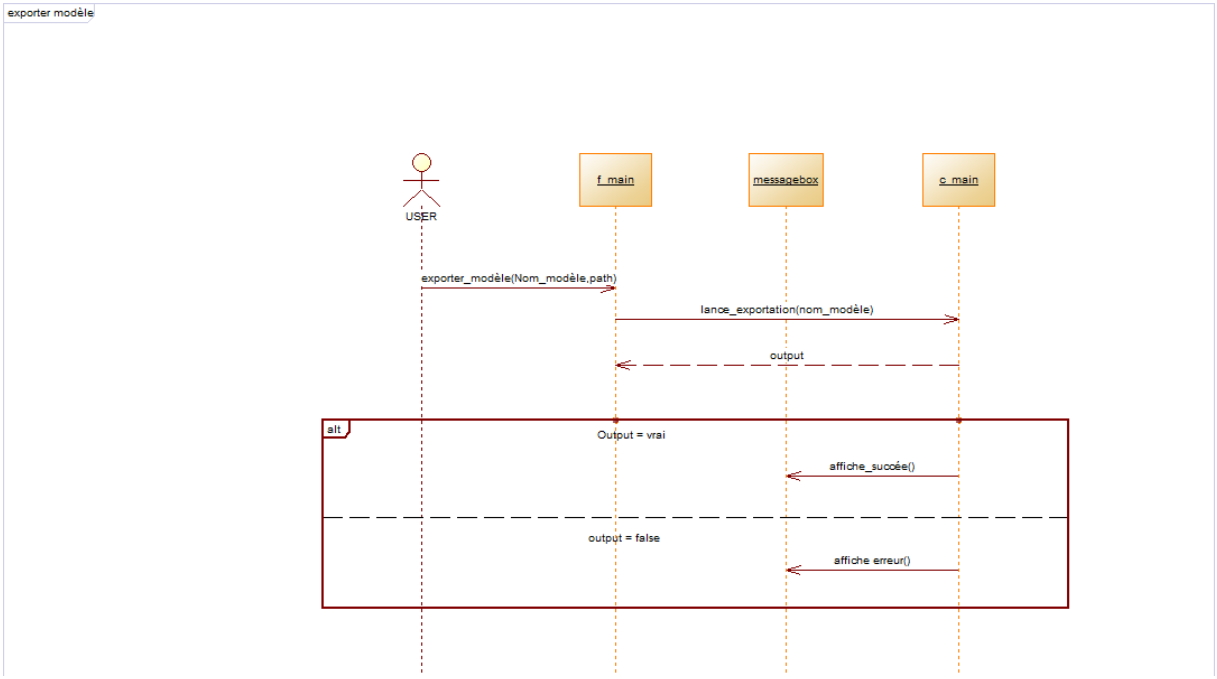
9. Spécifier le modèle :



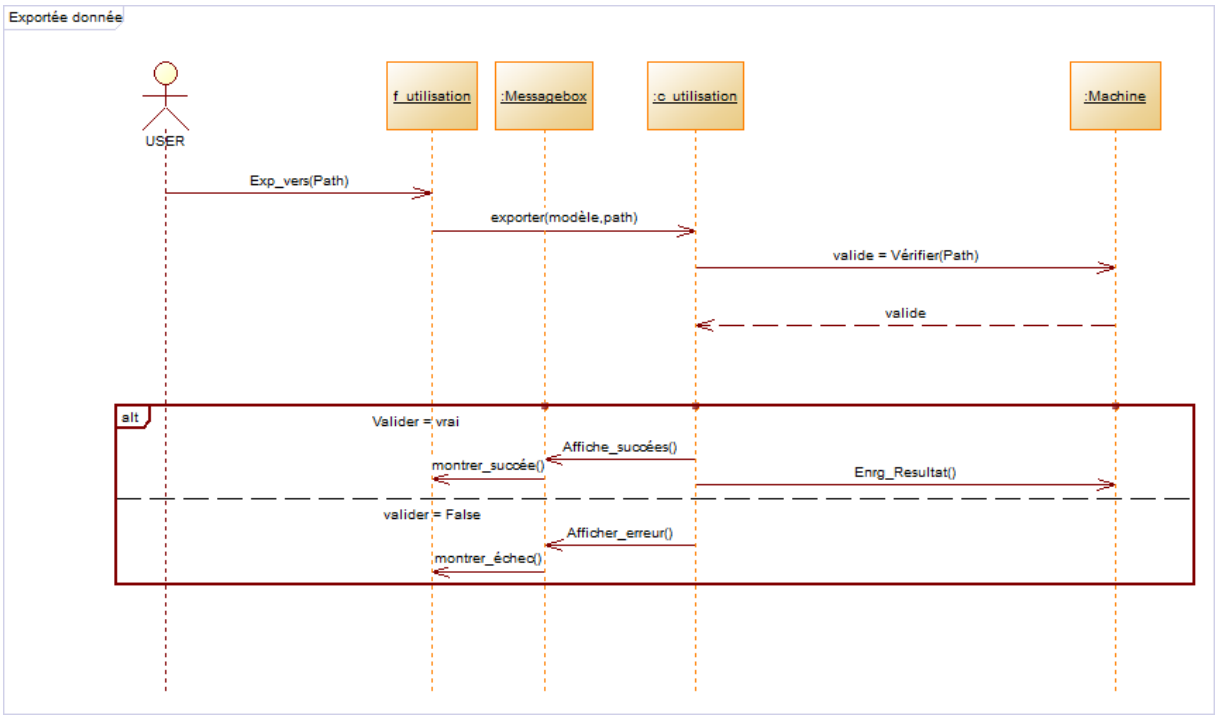
10. Trainer le modèle :



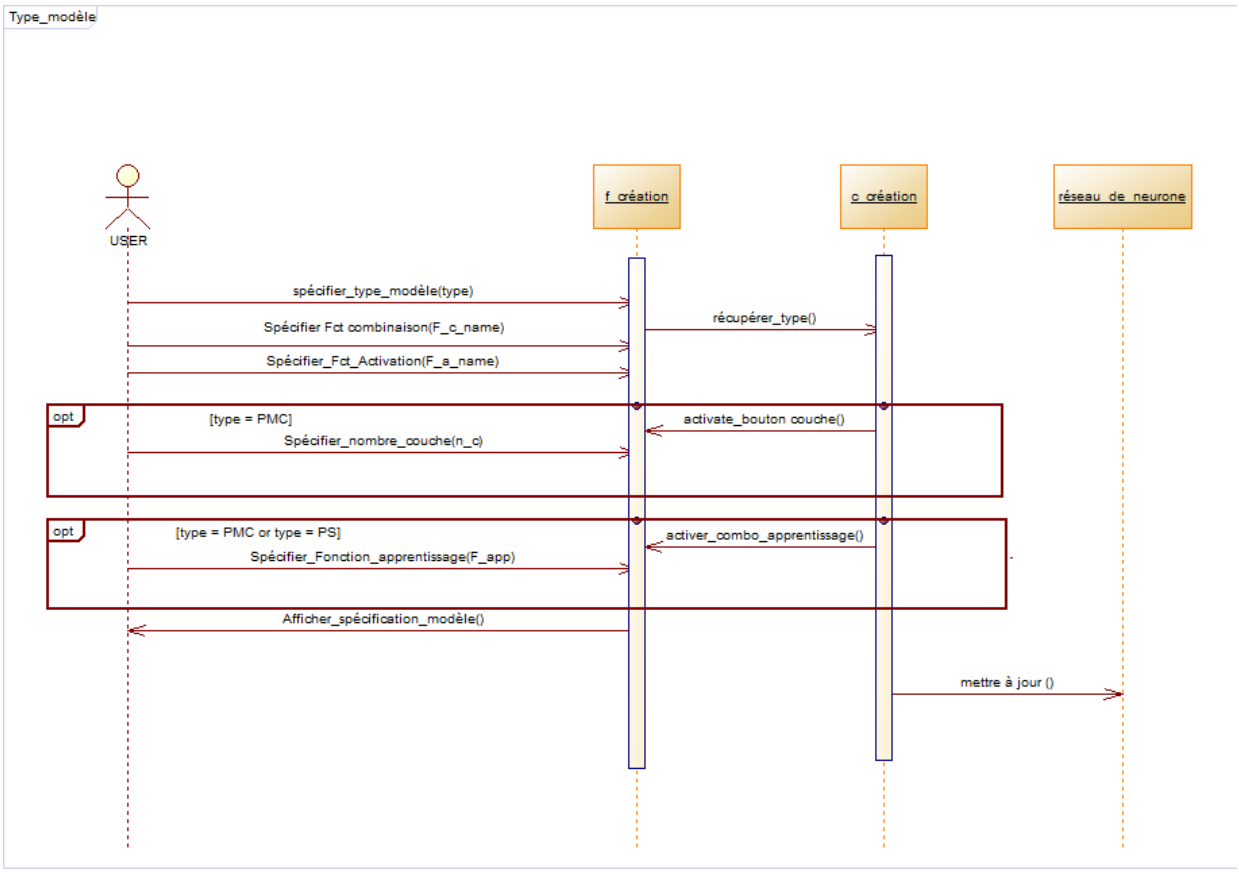
11. Exporter le modèle :



12. Exporter les données :



13. Type de modèle :

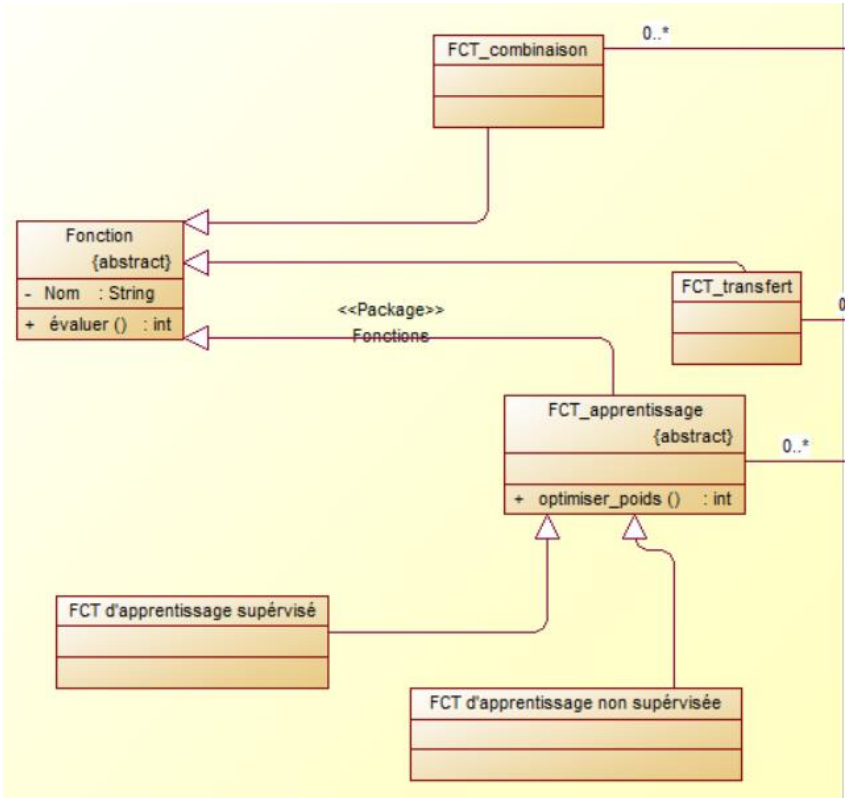


IV. 3ème version :

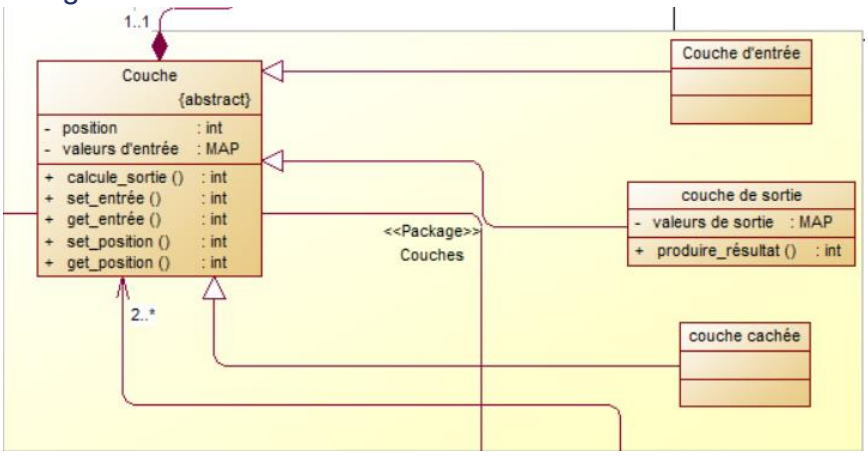
A. Classe de conception :

Notre dernier diagramme de classe participante contient 6 package en totale. Les packages sont comme suivants :

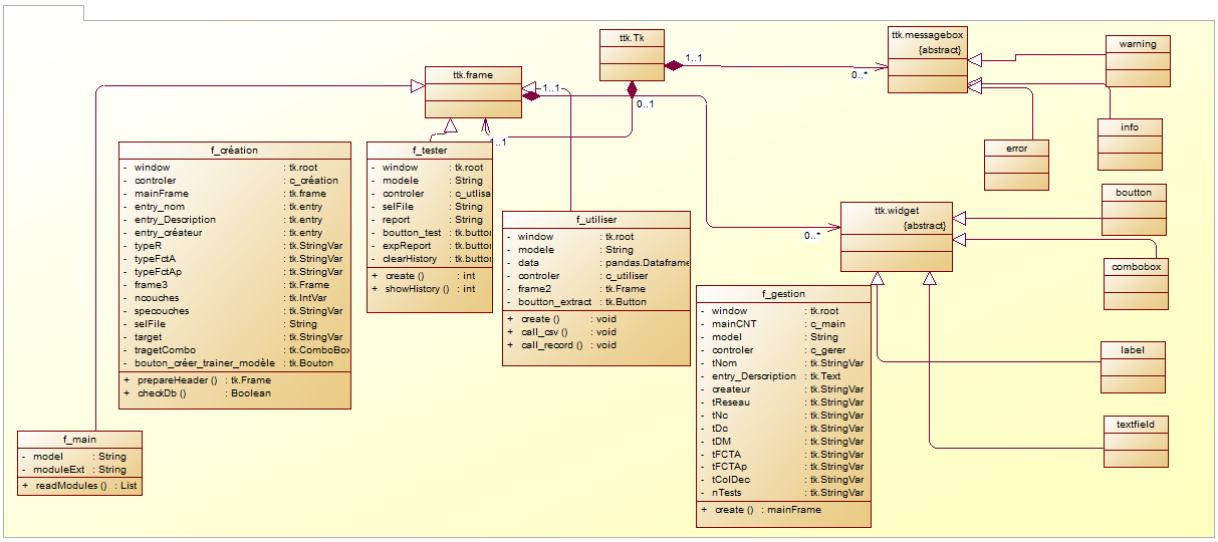
1. Package des fonctions :



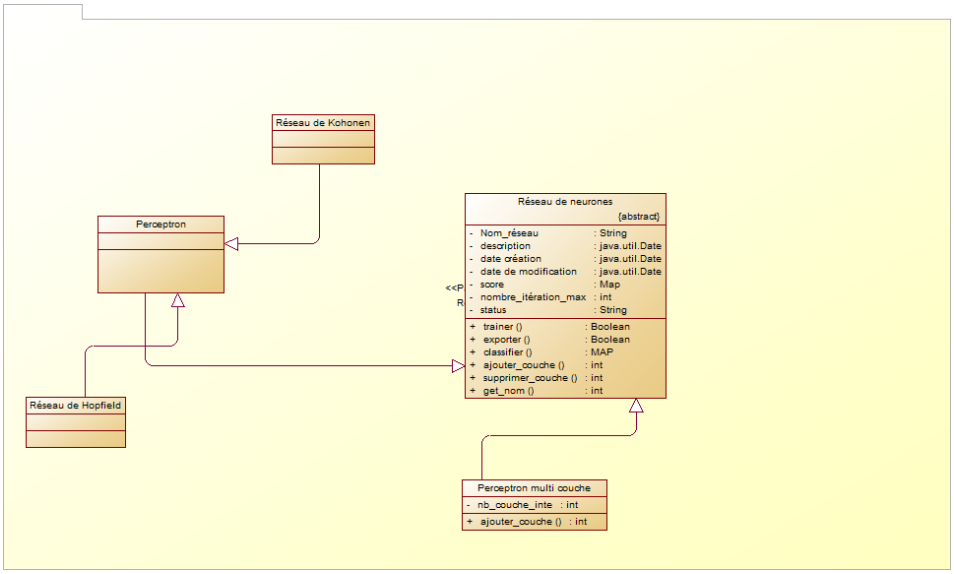
2. Package des couches :



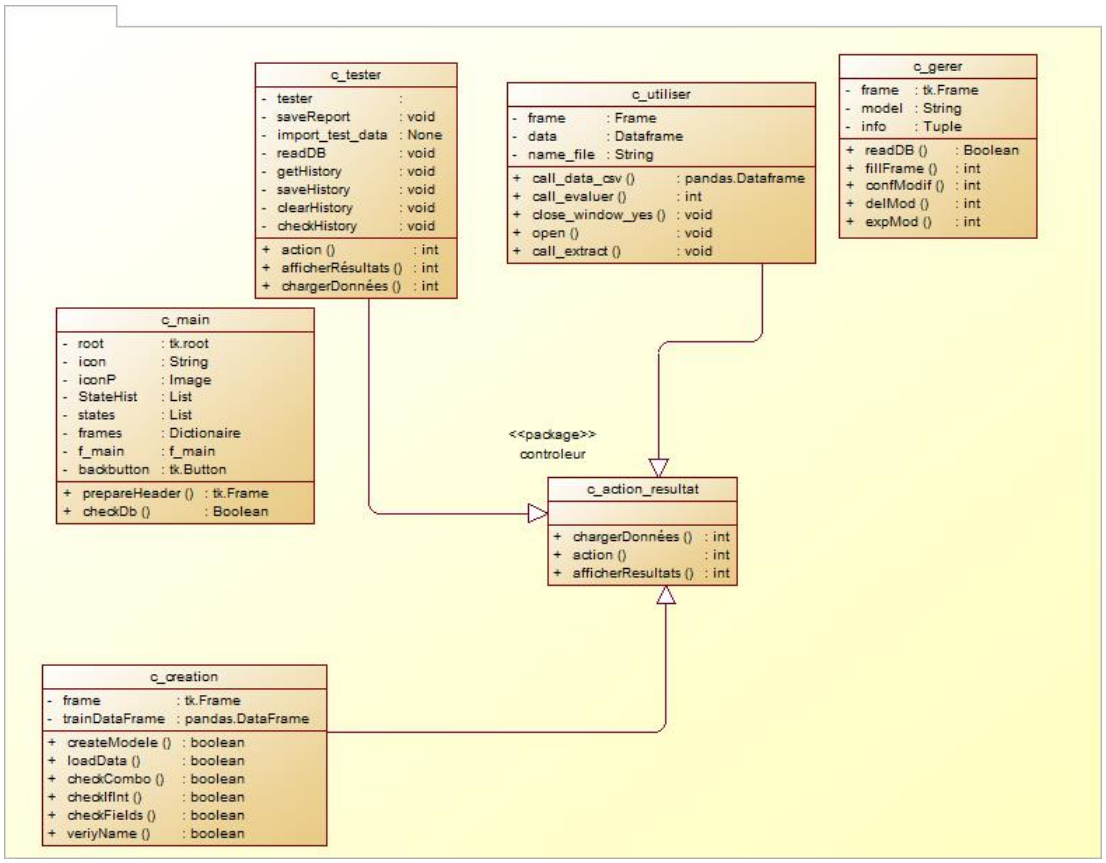
3. Package d'interface :



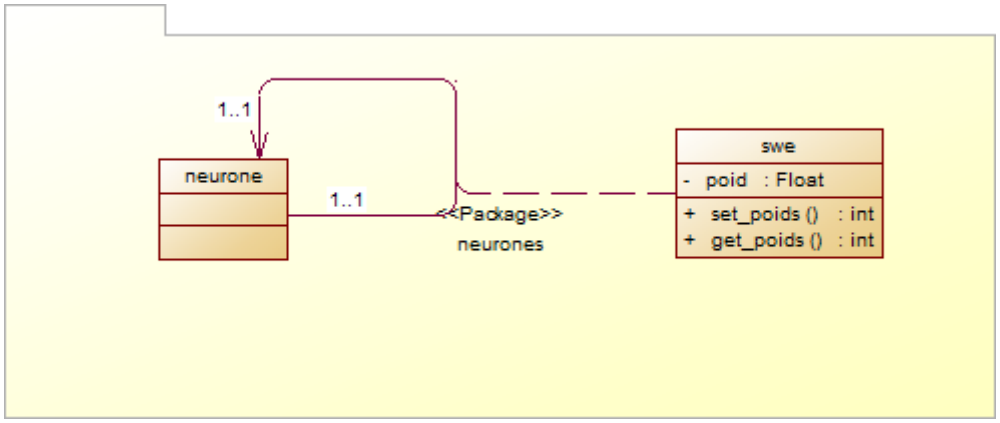
4. Package des réseaux :



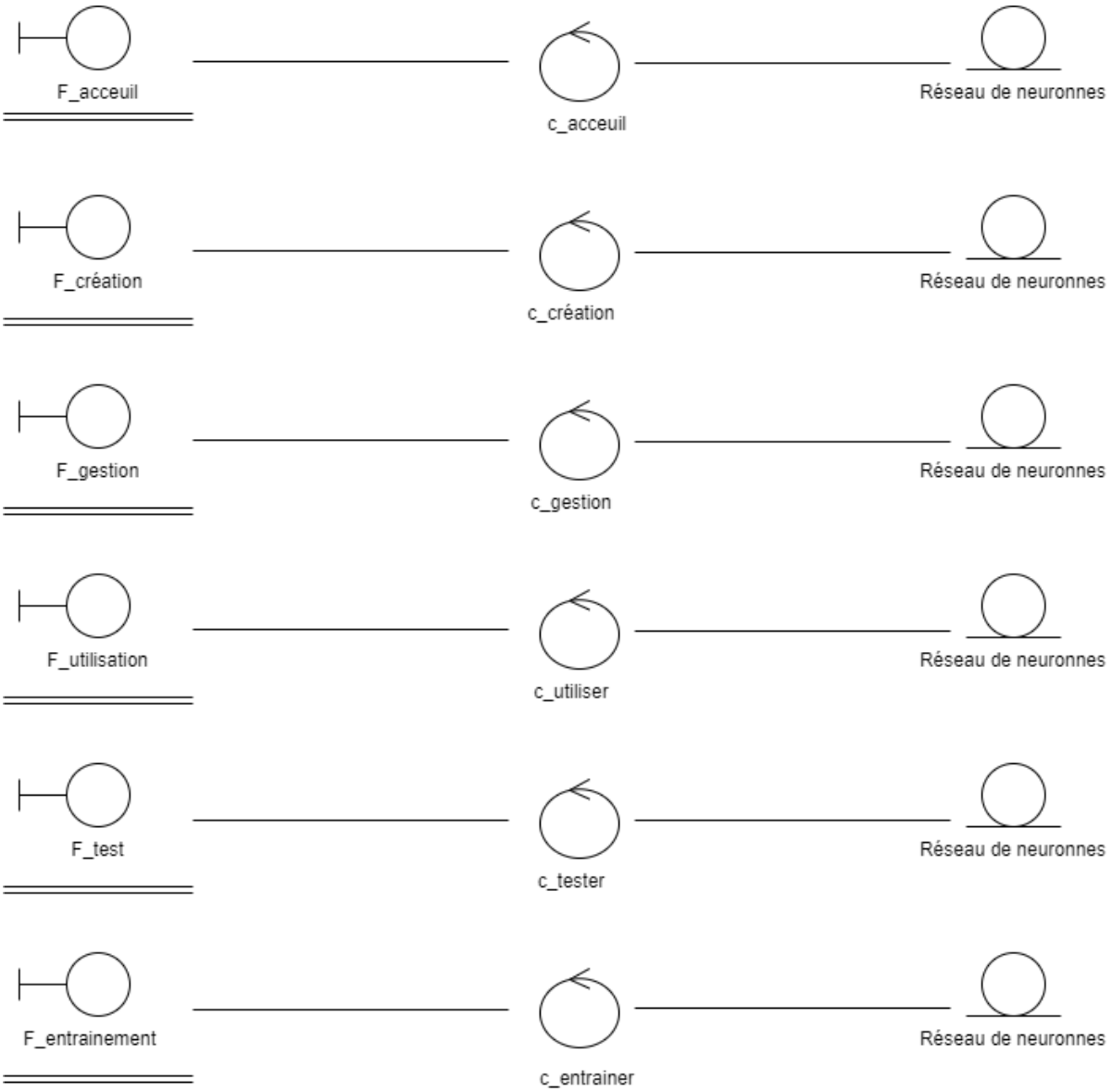
5. Package des contrôleurs :



6. Package des neurones :



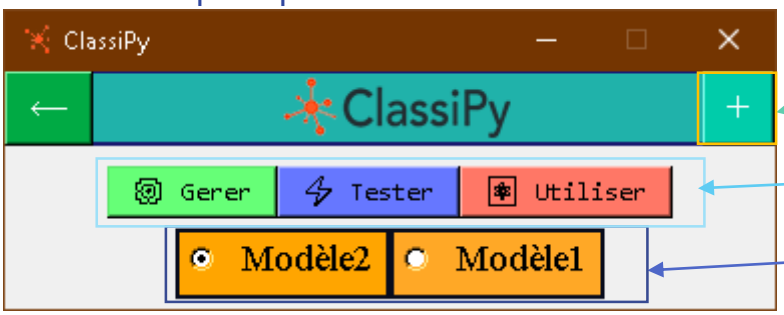
B. Diagramme de classe participante :



V. Maquette :

C'est la première partie de la réalisation de notre application, elle présente une interface graphique qui implémente les différents cas d'utilisation.

1. Menu principale :

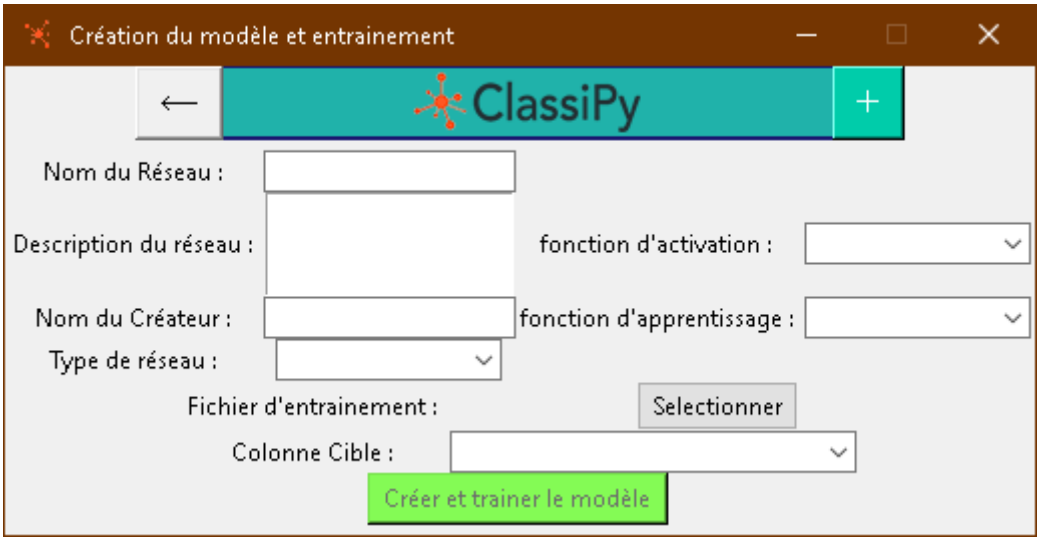


Créer un modèle

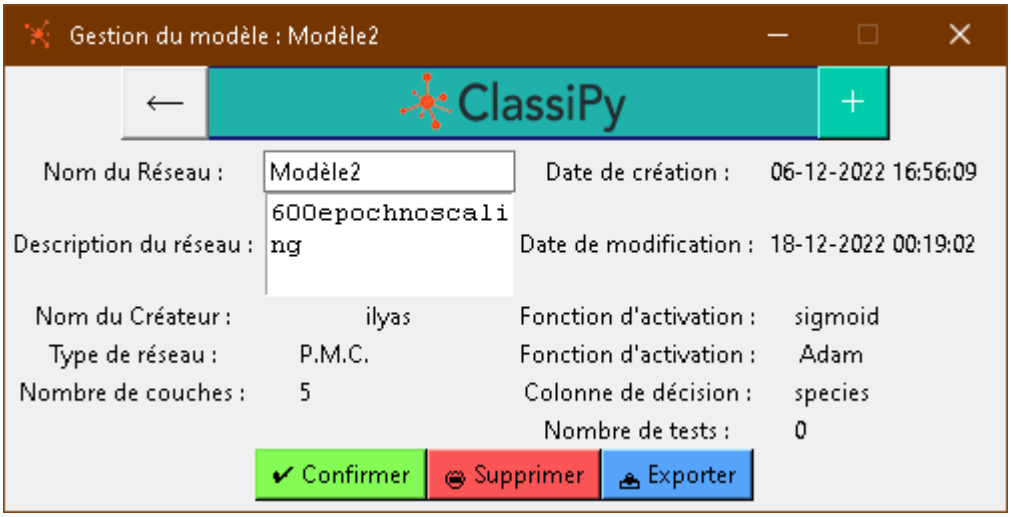
Actions à effectuer sur le model sélectionné

Les modèles existants

2. Création d'un modèle :

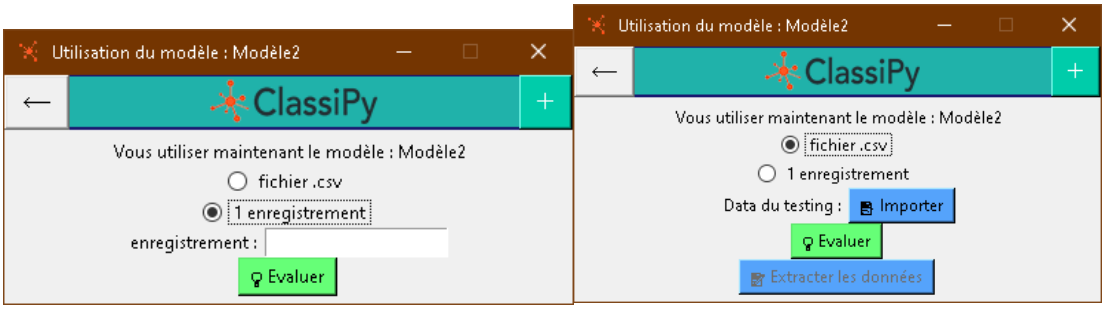


3. Gestion d'un modèle :



4. Tester un modèle :

5. Utiliser un modèle :



VI. Implémentation :

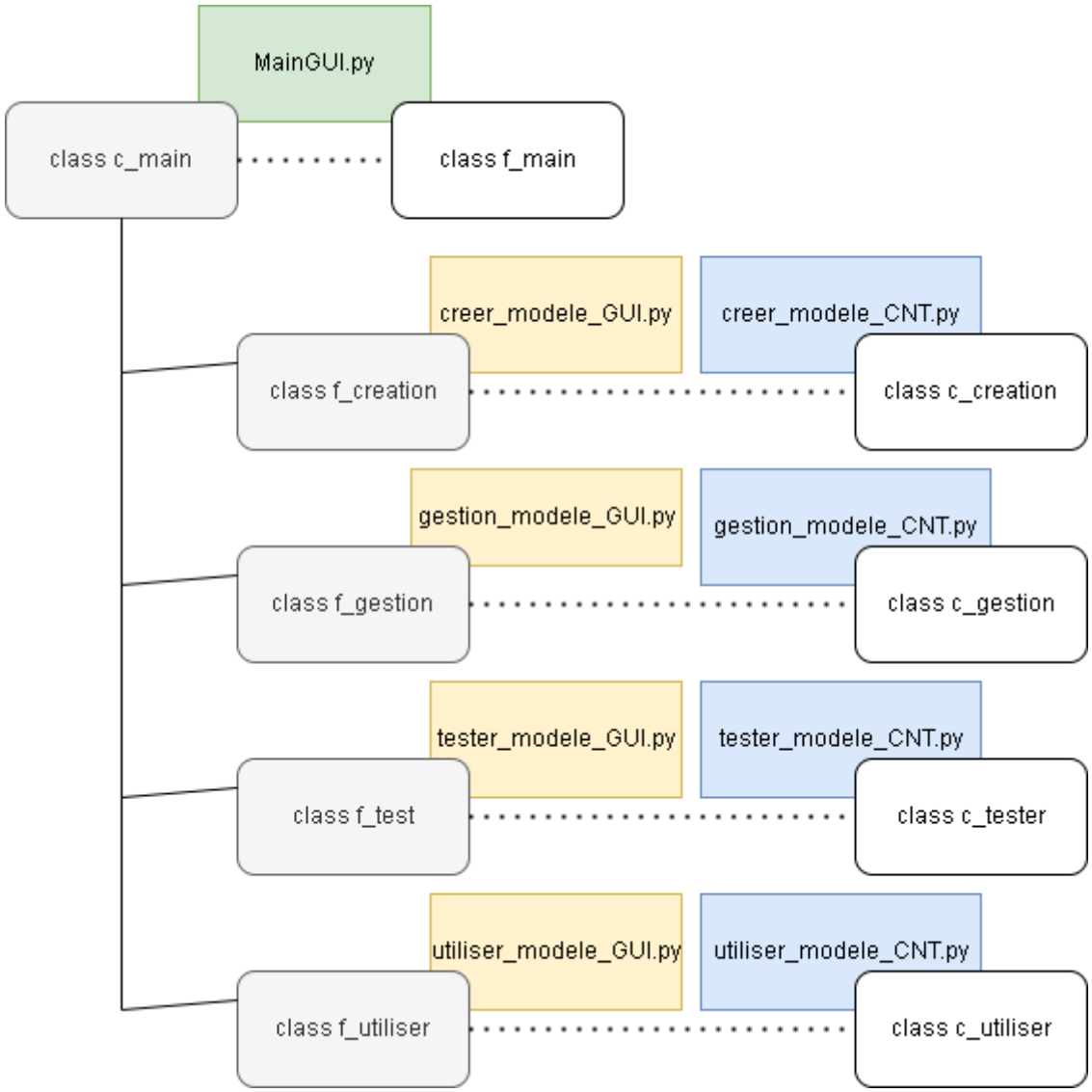
1. Module et Packages utilisés

Pour implémenter cette solution, nous avons opté à utiliser le langage Python et ses fameux package 'Tkinter' pour l'interface graphique, 'TensorFlow' pour la création, l'entrainement, l'utilisation et l'exportation des modèles, 'Sqlite3' pour des manupilation sur une base de données qui stock les données sur les différents modèles et les tests qui étaient effectués sur eux suivant le schéma suivant :
Tables :
Modeles (**id**, nom, description, createur, created, modified, tested, type, nbCouches, FCT_APP, FCT_AG, deci_col) ;
Tests(**id_test** , testFile, testStart, testDurrHrs, testScoreReport, #id_model) ;

Aussi le package 'OS' pour l'interaction avec la machine et d'autre pour des opérations élémentaires.

2. Structure en termes de fichiers (modules) :

Le graphe ci-dessous présente la structure du logiciel ClassiPy V1.0.0



VII. Bibliographie :

1. Documentation TensorFlow : https://www.tensorflow.org/api_docs
2. Documentation Tkinter : https://docs.python.org/3/library/tk.html
3. TensorFlow for Deep Learning : by Bharath Ramsundar, Reza Bosagh Zadeh - Publisher(s): O'Reilly Media, Inc.
4. Méthode Minimale en conception : https://masterrssi.wordpress.com/2015/01/01/methode-minimale/

VIII. Remerciement

Nous tenons à remercier notre professeur encadrante Mme. Nidal Lamghari pour son encouragement et son soutien durant toute les phases par lesquelles a passé ce projet, aussi pour ses efforts en faveur des étudiants pour les informer et les former en termes de conception et du génie logiciel.