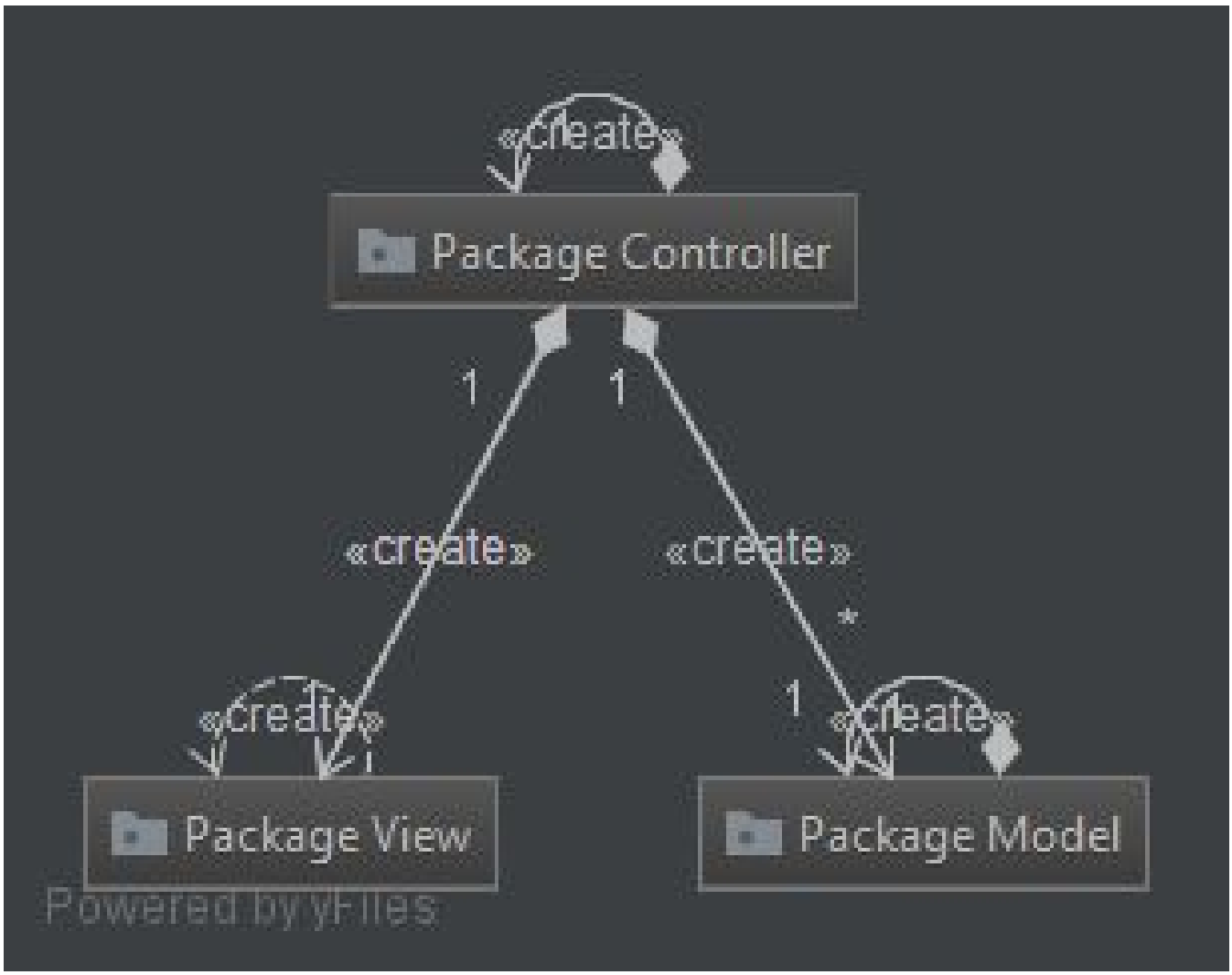
Modélisation UML

“Modélisation UML de l'application comprenant au minimum : un ou plusieurs diagrammes de classes pour l'ensemble de l'application, des machines à états pour les tâches, un diagramme de séquence système pour illustrer l'interaction de votre choix. Les diagrammes seront insérés dans un document pdf (à rendre) comprenant également des textes explicatifs, notamment au niveau des choix de conception.”

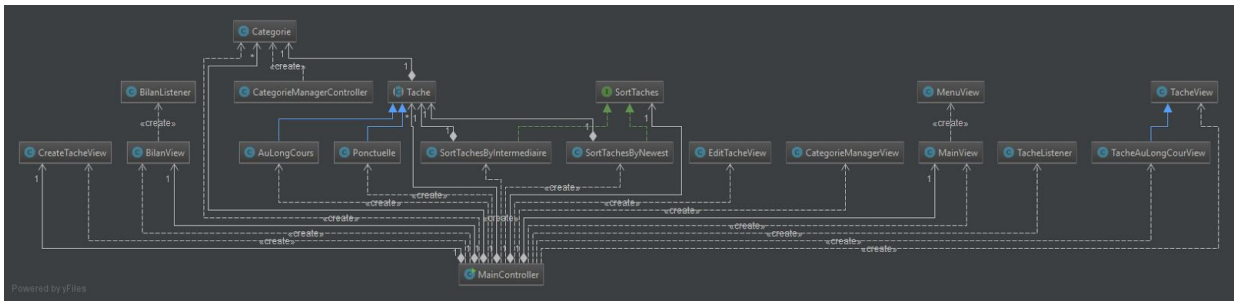
Sommaire

Diagramme des packages



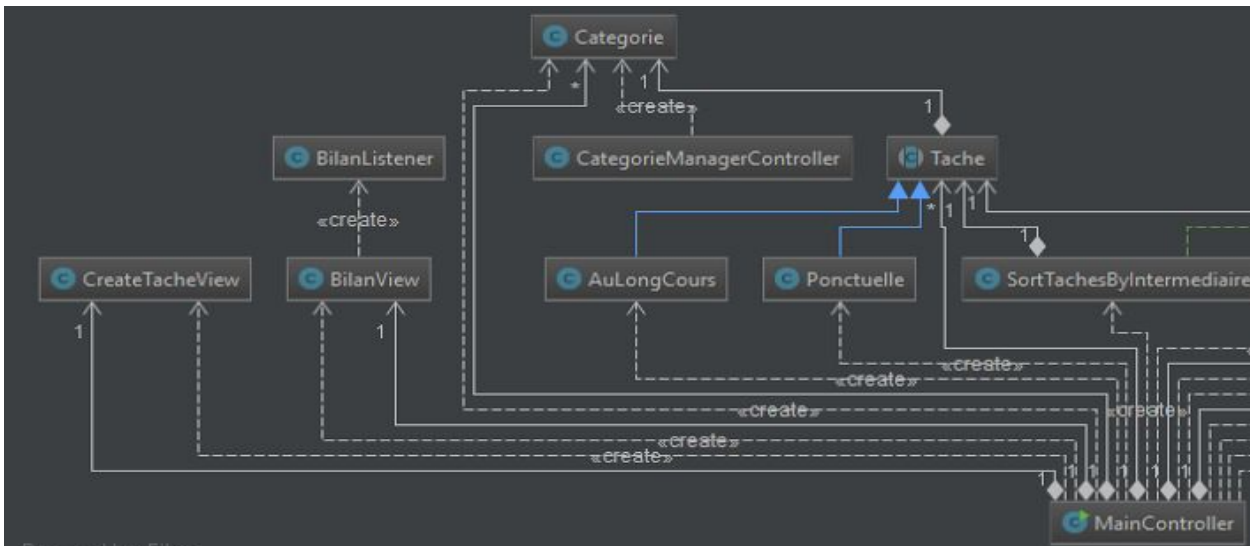
Dans ce diagramme UML, nous représentons les différentes liaisons qu'il y a dans notre application entre les différents packages.
Avec celui-ci, nous pouvons voir que, hormis les liaisons entre les packages eux-mêmes, le packages "Controller" est lié à la "View" et au "Model".
Les packages Model et View ne communiquent pas entre eux, ils transitent par le Controller pour cela.

Diagramme de classes



Dans ce diagramme UML, nous représentons toutes les liaisons existant entre les classes de l'application.

Zoom sur la partie gauche



Zoom sur la partie droite

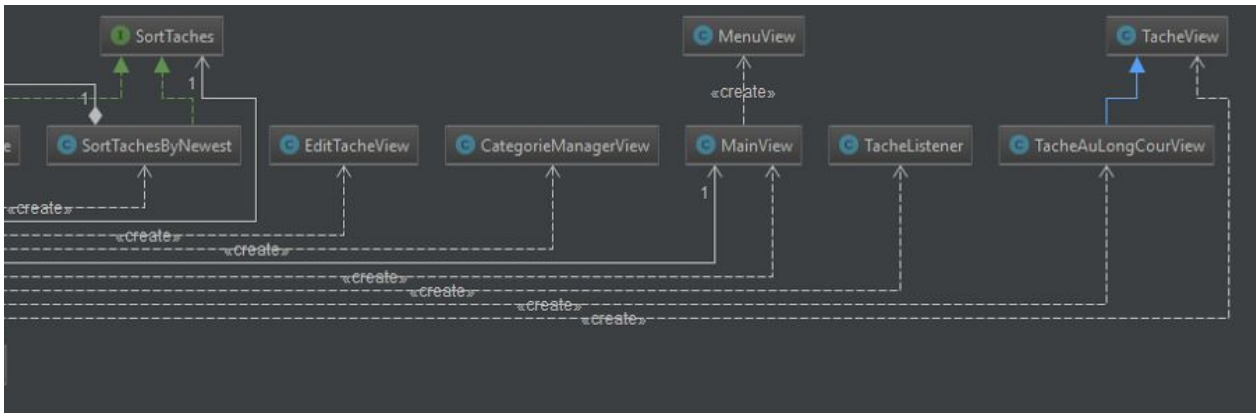
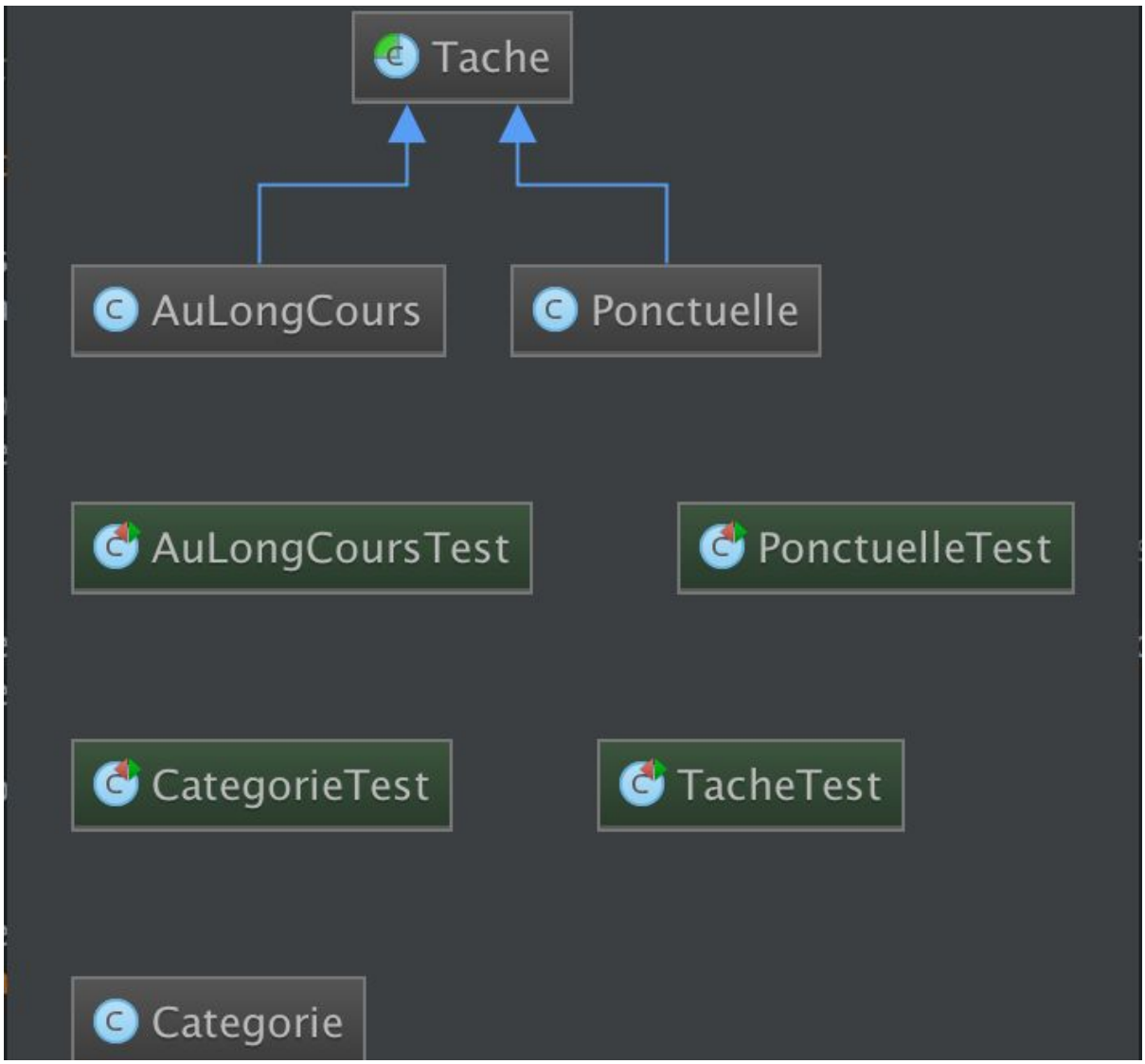


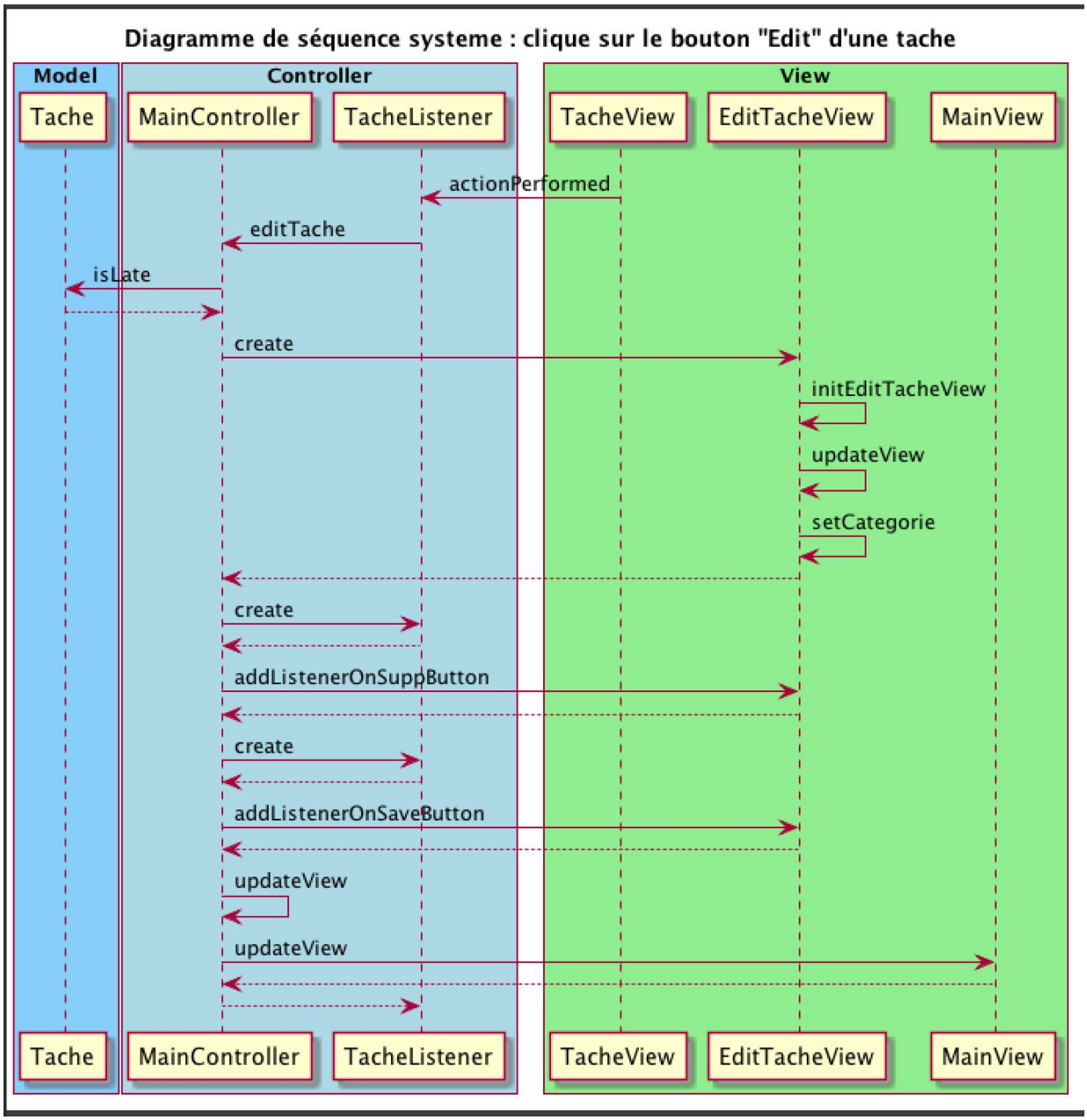
Diagramme de classes métiers



On peut voir que nous avons choisi d'implémenter le “Model” Tache sous forme d’une classe abstraite. Celle-ci définit les méthodes communes à tous les types de tâche. Et ensuite nous avons créé les classes pour chaque type de tâche. Chacune implémentant d’une certaine manière les fonctions abstraites définies par la classe tâche et en définissant d’autres plus spécifiques.

Diagrammes de Séquence Système

Action lancé lors du clic sur le bouton “Edit” d’une TacheView



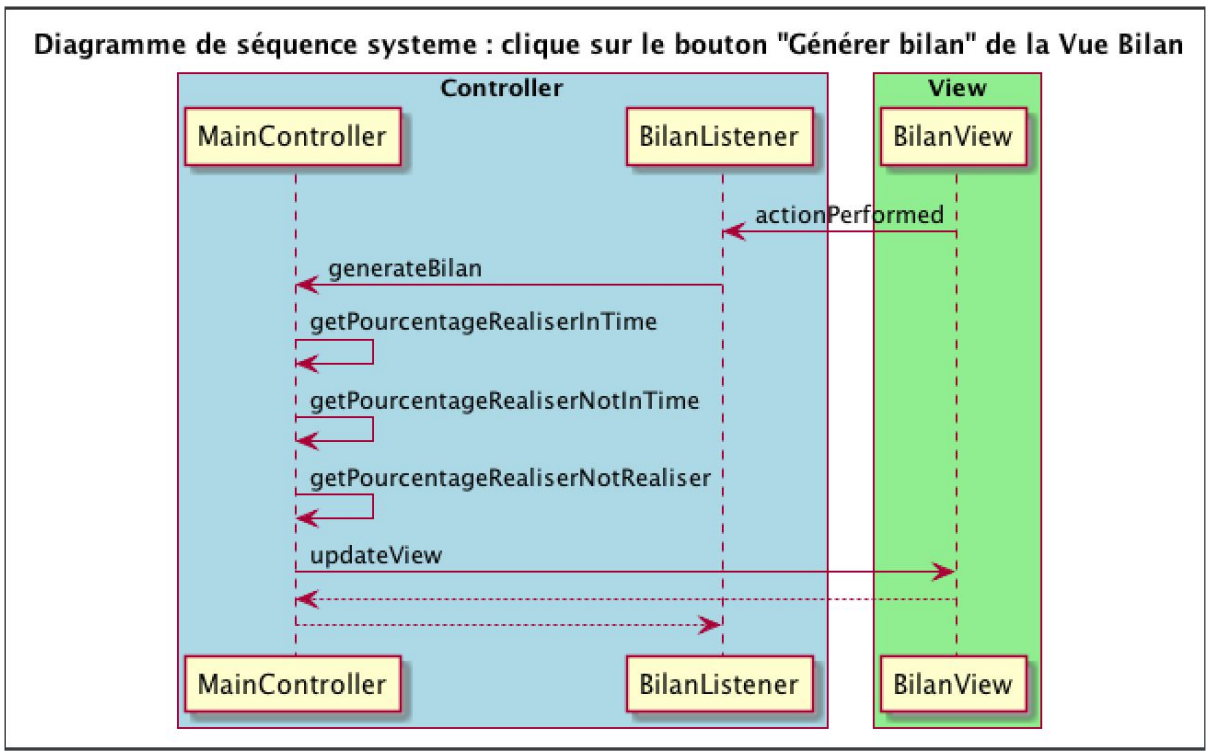
Dans ce diagramme de séquence système, nous modélisons, tous les appels de fonctions qui sont réalisées au clic sur le bouton “Edit”.

Lors de la réalisation d’une action, un “listener” est appelé. Celui-ci s’occupe d'appeler la fonction correspondant à l’action dans le controller “MainController”.
Ici, “TacheListener” est appelé, et il se charge d’appeler la fonction “editTache” du “MainController”.

Dans celle-ci, le “MainController”se charge de vérifier si la tâche est en retard en réalisant l’appel au “model” correspondant.

Enfin notre “MainController” s’occuper d’initialiser la vue “EditTacheView” en la créant et en lui passant les listeners dont elle a besoin.

Action lancé lors du clic sur le bouton “Générer le bilan” de la vue “Bilan”

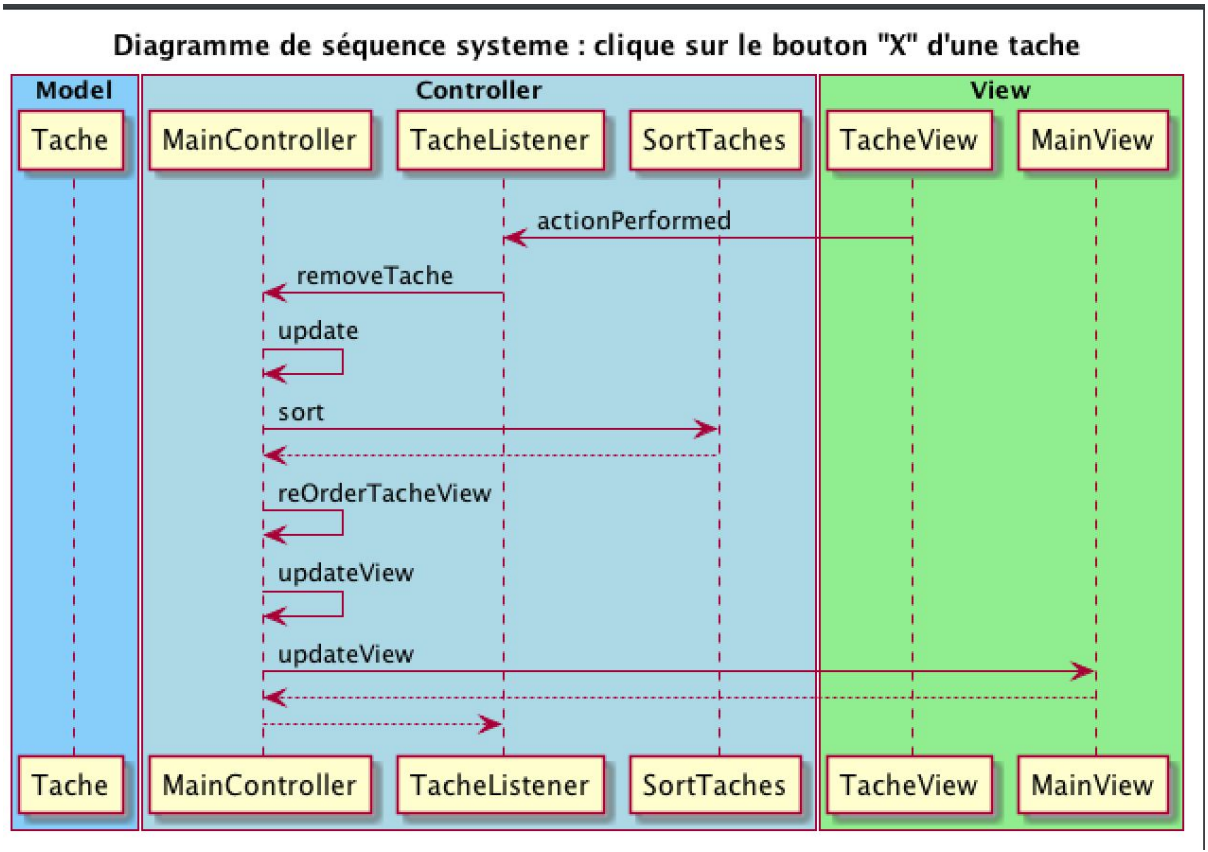


Dans ce diagramme de séquence système, nous modélisons tous les appels de fonctions qui sont réalisés au clic sur le bouton “Générer le bilan”.

Le schéma suivi et le même que ci-dessus. Une fois l’action réalisé, le listener correspondant est appelé, et celui-ci appel la méthode correspondante dans notre “MainController”.

Ici, celui-ci réalise plusieurs appels fonction afin de pouvoir mettre à jour, par la suite, la vue “BilanView”.

Action lancé lors du clic sur le bouton “X” d’une tâche



Le même schéma est une nouvelle fois suivi.

Cependant, ici le MainController fait appel à l’instance de “SortTache” afin de trier les tâches. La délégation de cette tâche nous permet par la suite de changer facilement de type de tri en changeant l’instance de “SortTache”.