Modèle D’analyse

Etude dynamique avec un diagramme de classe et des opérations

Diagramme de classe phase d'analyse statique conçu lors du TP3

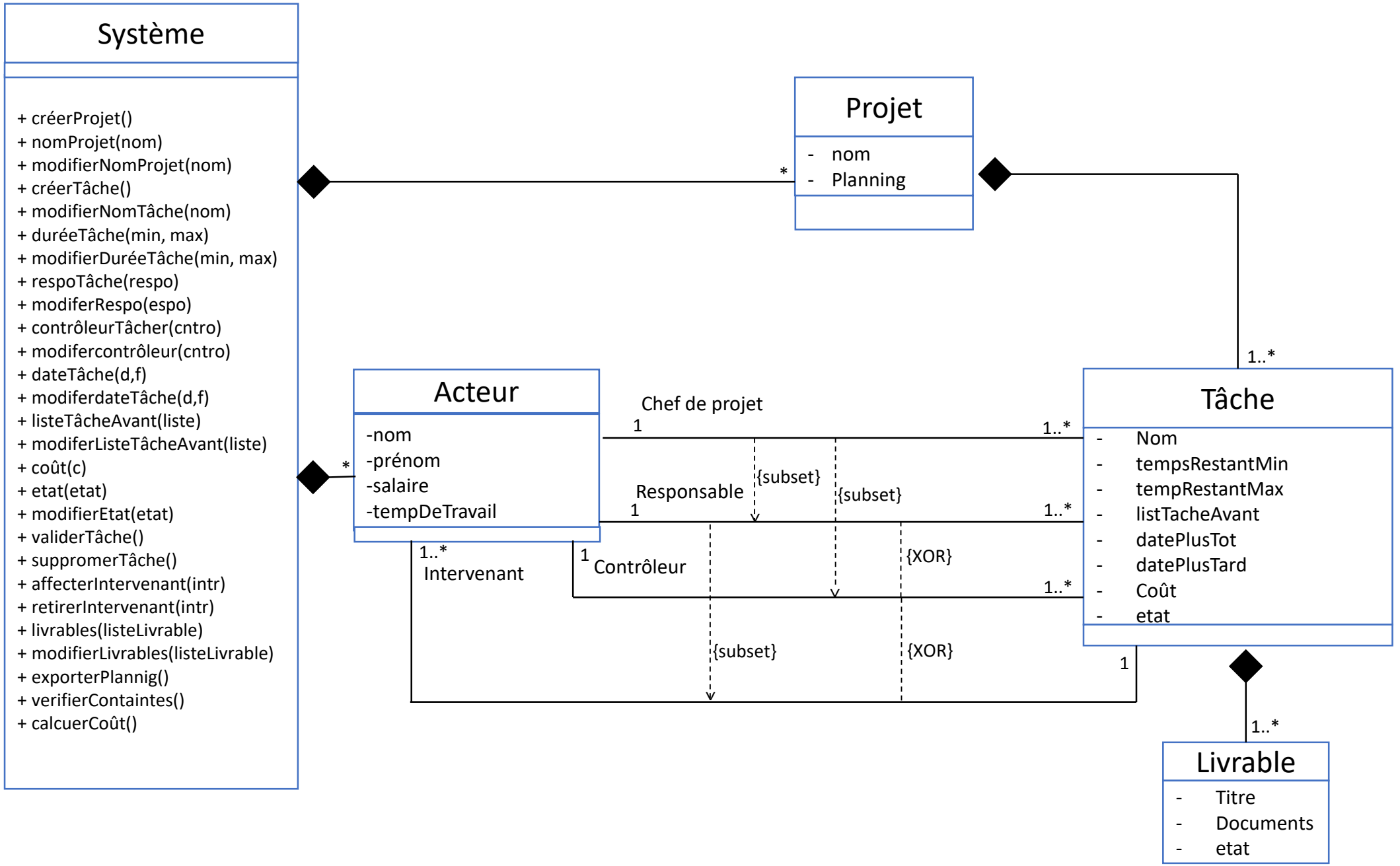


Diagramme de classe phase d'analyse dynamique

Système

- Projets: Collection<Projet>
- Acteurs: Collection<Acteur>

+ CréerProjet(nomProjet)
+-trouverProjet(nomProjet)
- messageErr( )
+ créerTâche(nomT, nomProjet)
+ modifierRespo(nomProjet, nomTâche, nomAncienRespo, nomNvRespo)

Tâche

- Nom
- tempsRestantMin
- tempRestantMax
- listTacheAvant
- datePlusTot
- datePlusTard
- Coût
- Etat
- Projet
- Livrables: Collection<Livrable>
- chefDeProjet
- Responsable
- Contrôleur
- Intervenant: Collection<Acteur>

+ <<CREATE>>(nomT)
+ trouverRespo(nomAncienRespo)
+ ajouterRespo(nomNvRespo)
+ supprimerRespo(nomAncienRespo)

Acteur

-nom
-prénom
-salaire
-tempDeTravail
-taches: Collection<Tâche>

Livrable

- Titre
- Documents
- Etat
- tâche

Projet

- nom
- Planning
- Tâches: Collection<Tâche>

+ <<CREATE>>(nomProjet)
+ nouveauTâche(nomT)
+ vérifierContrainte( )
+ miseAJourCoût( )
+ vérifierContrainte( )
+ trouverTâche(nomTâche)

Diagramme de séquence d'opération créer un projet

CréerProjet(nomProjet)

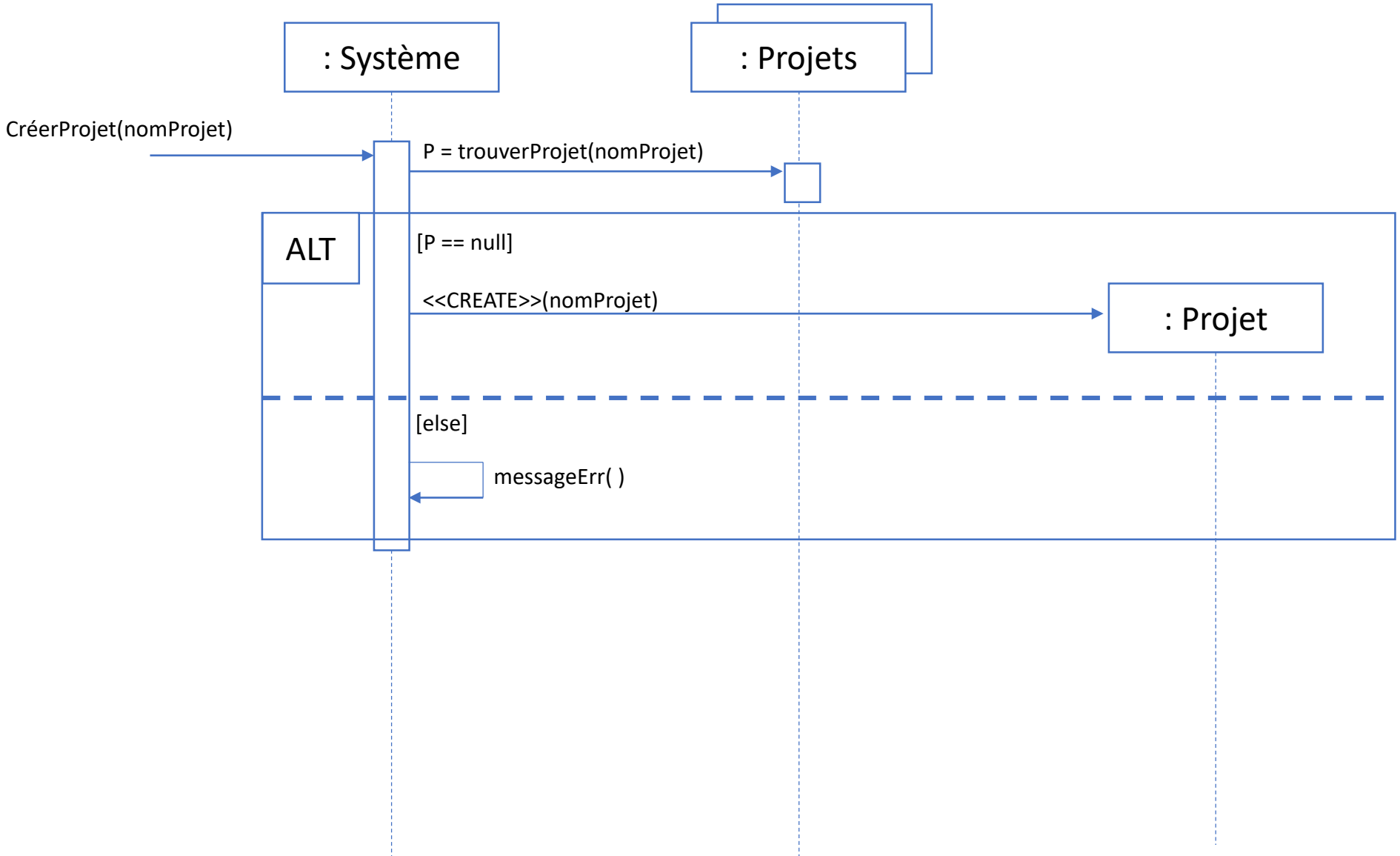


Diagramme de séquence d'opération créer une tâche

CréerTâche(nomT, Projet)

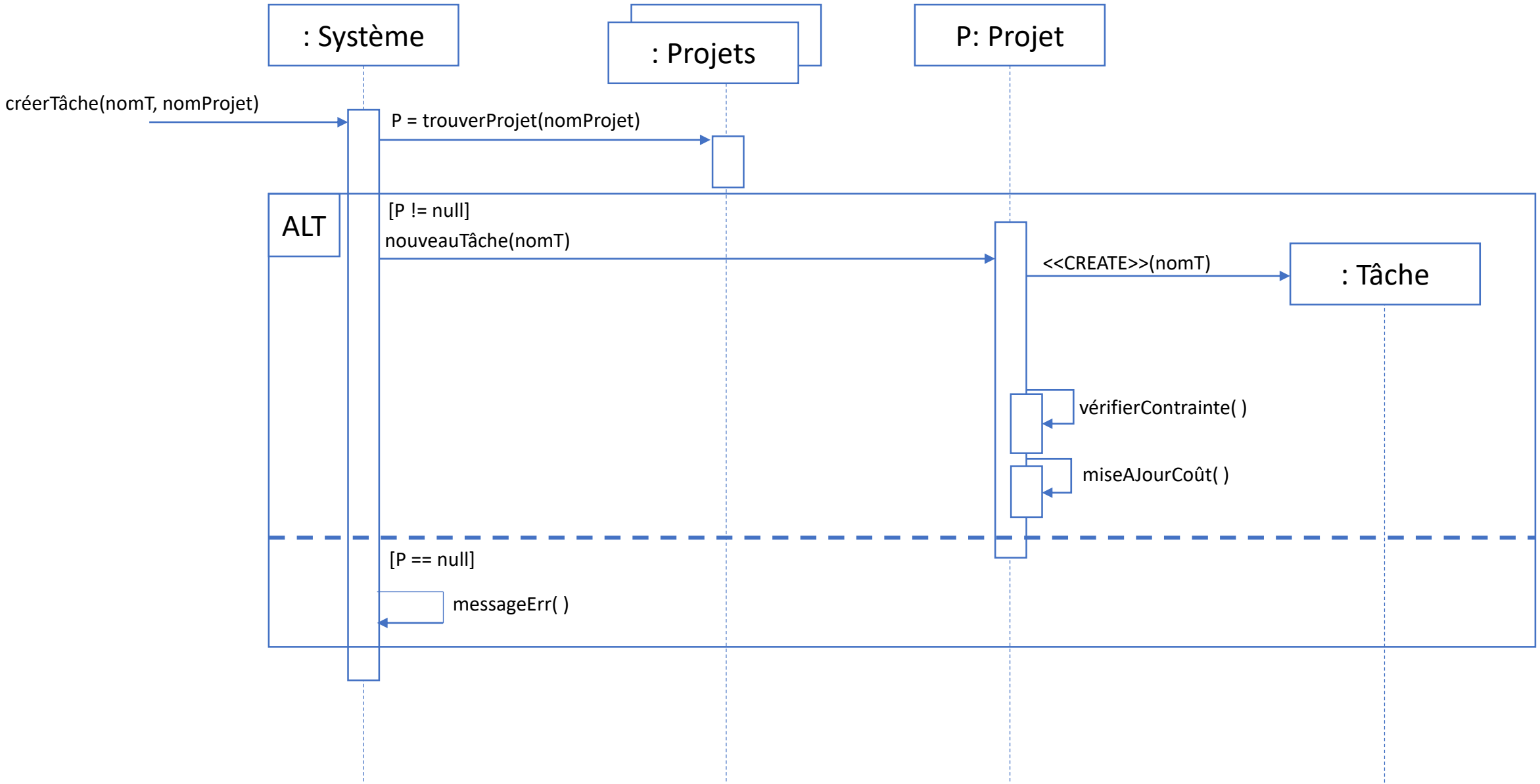
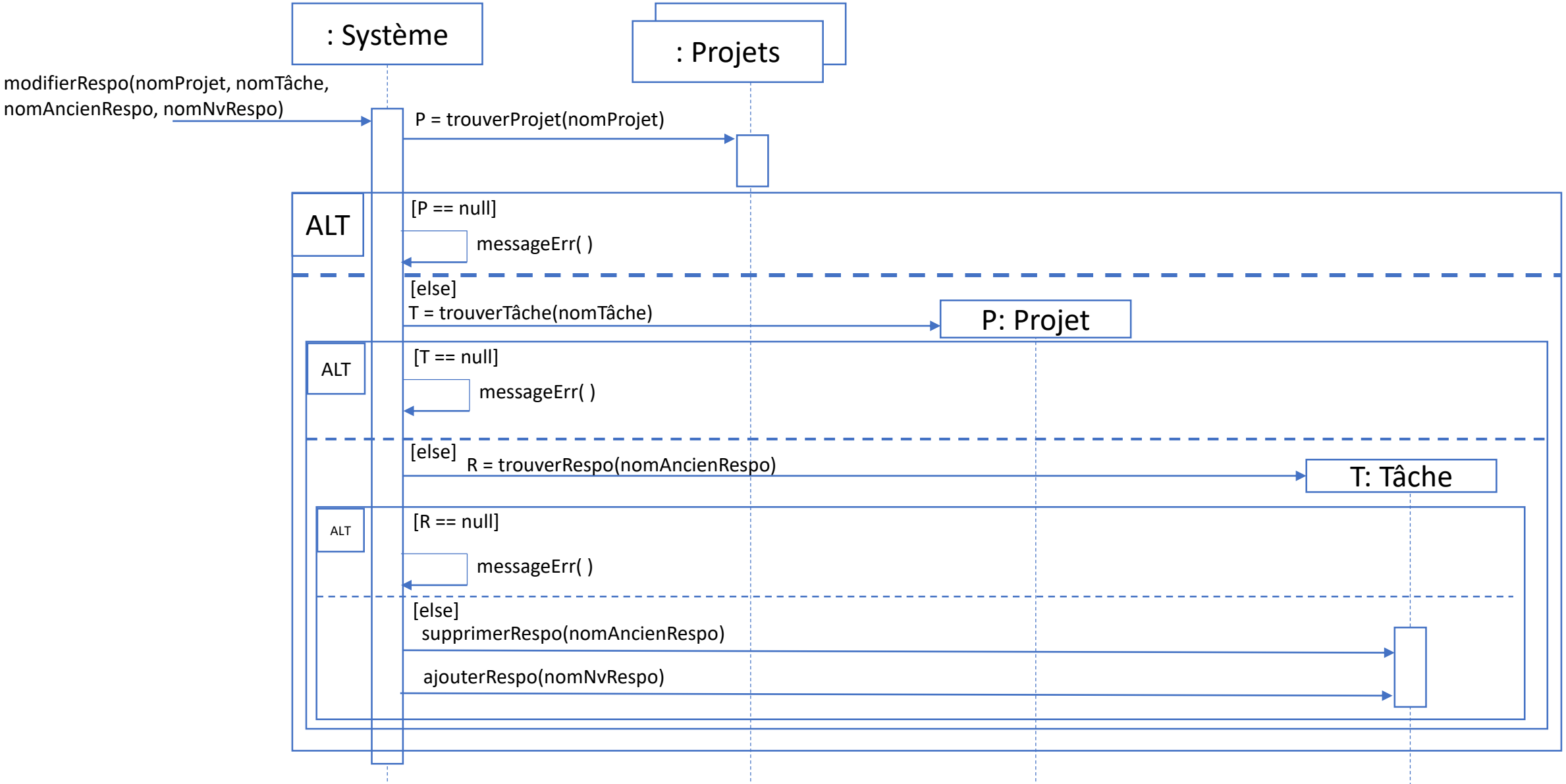


Diagramme de séquence d'opération modifier Responsable

modifierRespo(nomProjet, nomTâche, nomAncienRespo, nomNvRespo)



Commentaires TP analyse dynamique

I. Les opérations :

- *L'opération créer un projet : CréerProjet(nomProjet)*
  - Les arguments :
    - nomProjet : le nom du projet à créer.
  - Les méthodes utilisées dans cette opération :
    - *CréerProjet(nomProjet) :* méthode publique car elle est appelée par IHM, et elle appartient à la classe Système. elle permet de créer un nouveau projet avec le nom « nomProjet »
    - *trouverProjet(nomProjet)* : méthode appartient à la classe système, elle est privée car elle est accessible que par la classe système.(" :Projets " n'est pas une classe, c'est une Collection de la classe Projet qui se trouve dans la classe système).elle permet de trouver un objet Projet dans la collection Projets en lui communiquant le nom du projet.
    - *messageErr( )* : méthode appartient à la classe système, elle est privée car elle est accessible que par la classe système. Elle permet de renvoyer une erreur.
  - Logique de l'opération :
    - Si le projet n'existe pas (trouverProjet(nomProjet) retourne null) alors on appel au constructeur de la classe Projet (notation : <<CREATE>>(nomProjet)).
    - Si non (il existe un projet du même nom) alors en retourne une erreur par la méthode messageErr().

- *L'opération créer une tâche : CréerTâche(nomT, Projet)*
  - Les arguments :
    - nomT : le nom de la tâche à créer.
    - nomProjet : nom du projet qui va contenir cette nouvelle tâche.
  - Les méthodes utilisées dans cette opération :
    - *créerTâche(nomT, nomProjet) :* méthode publique car elle est appelée par IHM, et elle appartient à la classe Système. Elle permet de créer une nouvelle tâche nommée « nomT » dans un Projet nommé « nomProjet »
    - *trouverProjet(nomProjet)* : méthode appartient à la classe système, elle est privée car elle est accessible que par la classe système.(" :Projets " n'est pas une classe, c'est une Collection de la classe Projet qui se trouve dans la classe système).elle

permet de trouver un objet Projet dans la collection Projets en lui communiquant le nom du projet.

- *nouveauTâche(nomT)* : méthode publique car elle est appelée par la classe système, et elle appartient à la classe Projet.elle permet de créer une tâche dans le projet « nomProjet ».
- *vérifierContrainte( ) :* méthode appartient à la classe Projet , elle est privée car elle est accessible que par la classe Projet. Elle permet de vérifier les contraintes de la création d'une tâche.
- *miseAJourCoût( ) :* méthode appartient à la classe Projet , elle est privée car elle est accessible que par la classe Projet. Elle permet de mettre à jour le Coût après la création d'une tâche.
- *messageErr( )* : méthode appartient à la classe système, elle est privée car elle est accessible que par la classe système. Elle permet de renvoyer une erreur.

- Logique de l'opération :
  - Si le projet existe (trouverProjet(nomProjet) retourne un objet de classe Projet) alors on appel au nouveauTâche(nomT) de la classe Projet qui créer la nouvelle tâche (notation : <<CREATE>>(nomT)).
    Vérifie les contraintes de création et met à jour le Coût du projet.
  - Si non (trouverProjet(nomProjet) retourne null) alors en retourne une erreur par la méthode messageErr().

- *L'opération modifier Responsable : modifierRespo(nomProjet, nomTâche, nomAncienRespo, nomNvRespo)*
  - Les arguments :
    - nomProjet : nom du projet qui va contenir cette nouvelle tâche.
    - nomTâche : le nom de la tâche à créer.
    - nomAncienRespo *: le nom de responsable à modifier.*
    - nomNvRespo *:* le nom du nouveau responsable.
  - Les méthodes utilisées dans cette opération :
    - *trouverProjet(nomProjet)* : méthode appartient à la classe système, elle est privée car elle est accessible que par la classe système.(‘’ :Projets ‘’ n'est pas une classe, c'est une Collection de la classe Projet qui se trouve dans la classe système).elle permet de trouver un objet Projet dans la collection Projets en lui communiquant le nom du projet.
    - *trouverTâche(nomT) :* méthode publique car elle est appelée par IHM, et elle appartient à la classe Système. Elle permet de

créer une nouvelle tâche nommée « nomT » dans un Projet nommé « nomProjet ».

- *trouverRespo (nomAncienRespo) :* méthode publique car elle est appelée par la classe système , et elle appartient à la classe Tâche. Elle retourne un objet responsable de la classe Acteur d'une tâche. Si non retourne nulle.
- *supprimerRespo(nomAncienRespo)* : méthode publique car elle est appelée par la classe système, et elle appartient à la classe Tâche. Elle supprime le responsable « *nomAncienRespo »* de la tâche.
- *ajouterRespo(nomNvRespo)* : méthode publique car elle est appelée par la classe système, et elle appartient à la classe Tâche. Elle ajoute le responsable « nomNvRespo*»* à la tâche.
- *messageErr( )* : méthode appartient à la classe système, elle est privée car elle est accessible que par la classe système. Elle permet de renvoyer une erreur.

- Logique de l'opération :
  - Si le projet existe (trouverProjet(nomProjet) retourne un objet de classe Projet) et il contient la tâche « *nomT », et si cette tâche contient le responsable « nomAncienRespo »* alors on appel à *supprimerRespo(nomAncienRespo)* de la classe Têche qui supprime l'Acteur « *nomAncienRespo* ». En suite en ajoute à cette tâche le nouveau responsable ajouterRespo(nomNvRespo).
  - Si non alors en retourne une erreur par la méthode messageErr().