Cahier de TD/TP

Génie Logiciel

UML

UML – diagrammes de cas d'utilisation et diagrammes de séquences

Exercice 1 : Entretien d'un TGV:

L'entretien d'un TGV nécessite la collaboration de plusieurs équipes. Le personnel de nettoyage doit nettoyer les wagons, le personnel technique doit faire des vérifications (freins, portes, …), et le service de restauration doit réapprovisionner le TGV en nourriture et en boissons.
- Donner le diagramme de cas d'utilisation

Exercice 2 : Virement bancaire :

On souhaite modéliser un système permettant d'effectuer des virements bancaires. Le système doit être accessible à des clients en local et par internet. Un client peut effectuer un virement après s'être identifié.
- Donner le diagramme de cas d'utilisation
- Décrivez un scénario nominal (happy Day) pour un virement sur internet
- Décrivez un scénario exceptionnel pour un virement sur internet
- Donner le diagramme de séquence pour ces 2 scénarios

Exercice 3 : Com pas com :

On souhaite modéliser le système de télécommunication d'une entreprise *ComPcom*. Ce système héberge une plateforme de services téléphoniques *PST* et propose un service de télécommunication *SCOM*. Le *SCOM* autorise les clients de l'entreprise *ComPcom* à configurer des scénarii automatiques pour des appels sortant de la plateforme de *ComPcom*.
Par exemple, l'automate peut appeler des gens chez eux pour le compte de l'entreprise *GLcom* (un client de la société *ComPcom*), et proposer aux gens une offre commerciale ou des invitations dans des salons, leur poser des questions, et enregistrer les réponses.

Via un site Web, le client peut choisir un message vocal à diffuser, la liste des destinataires et déclencher la campagne d'appels immédiatement ou bien à une date planifiée. Il peut aussi enregistrer un nouveau message de diffusion, ou une nouvelle liste de destinataires. Par exemple, l'automate d'appel appelle les habitants d'une commune en cas d'alertes aux inondations.
- Décrire les cas d'utilisations du service *SCOM*
- Donner un scénario nominal pour l'alerte des habitants d'une commune en cas d'inondation
- Donner un scénario d'exception
- Décrire par un diagramme de séquence comment un appel est déclenché chez un habitant de la commune.

Exercice 4 : Boutique de vêtements

Considérez une boutique de vêtements.
1. Énumérez 3 acteurs impliqués dans la conception d'un système de gestion de caisse.
2. Un cas d'usage est l'achat d'articles. Suivant le point de vue du consommateur, citez un autre cas d'utilisation, dont le niveau d'abstraction est identique.
3. Tracez un diagramme de cas d'utilisation du système de gestion de la caisse d'une boutique
4. Rédigez un scénario standard pour chaque cas d'utilisation.
5. Rédigez un scénario d'exception pour chaque cas d'utilisation

UML – diagrammes de classes

Exercice 5 : Gestion de vente

- Les clients utilisent un bon de commande sur lequel ils précisent leur numéro de client, leurs coordonnées ainsi que la date de commande.
- La partie commande comporte un certain nombre de lignes de commande composées de la référence unique de l'article, le prix unitaire et la quantité commandée.
- Le client doit également préciser le mode de paiement (chèque, carte, immédiat ou 3 mois après la commande). Dans le cas d'un paiement différé, seul le paiement par carte est accepté. Le client doit préciser le numéro de la carte et sa date d'expiration.
- A la livraison, un bon de livraison accompagne le colis. Une commande peut faire l'objet de livraisons partielles. Le bon de livraison comporte le détail des produits livrés (référence, quantité).

Décrire le diagramme de classe

Exercice 6 : Un timbre, une adresse et une enveloppe

Une enveloppe prête à poster contient une adresse, et un timbre. Dans le cas contraire, elle n'est pas valide. Vu sous cet angle, la relation entre timbre, enveloppe et adresse est donc une composition.

Si maintenant on considère que adresse est un concept à part entière, l'adresse peut être notée sur plusieurs enveloppes : elle est partagée, il s'agit d'une agrégation.

Si on considère la vie des objets, le timbre est isolée ou en carnet, avant d'être collé sur l'enveloppe. Cette dernière est vierge avant utilisation.

Faire le diagramme de classes correspondant.

Mettre en évidence un héritage pour l'état de l'enveloppe (Affranchie, Adressée, PrêteAPoster) et sa liaison avec l'enveloppe.

Exercice 7 : Echiquier

Un échiquier est un tableau de 8 par 8 cases. Les pièces sont de 2 couleurs (noir/blanc)
Les figures sont roi (1), reine (1), fou (2), tour (2), cavalier (2), pions (8)
Un mouvement est caractérisé par la position de départ et d'arrivée d'une pièce.

- Représenter une situation quelconque d'une partie d'échec en utilisant des classes-associations
- Donner une modélisation pour la mémorisation des mouvements

Exercice 8 : Métro

Pendant la phase de spécifications des besoins, l'utilisateur du futur logiciel Métro a écrit le texte suivant :
« Le logiciel Métro est destiné aux utilisateurs du métro. L'utilisateur indique la gare de départ et la gare d'arrivée et le logiciel Métro conseille un trajet reliant ces deux gares. Les gares ont un nom et sont situées sur des lignes. Une ligne a une couleur. Une gare est soit un terminus, soit une correspondance, soit une gare normale. Une ligne possède deux terminus, des gares normales et au moins une correspondance. Une correspondance relie au moins deux lignes. Un trajet est composé d'une gare de départ, d'une liste de segments et d'une gare d'arrivée. Un segment correspond à une ligne et possède une gare origine et une gare destination. »

1) Dessiner le diagramme de classes correspondant à ce texte.
3) Dans la classe Gare, écrire la signature des deux méthodes répondant à la phrase suivante :
« Pour une ligne et un terminus donnés, une gare a éventuellement une gare suivante et une gare précédente. »
4) Effectuer la spécification d'interfaces correspondant au **cas d'utilisation** dans lequel l'utilisateur demande le trajet reliant la gare de départ *pyramide*, située sur la ligne *rouge*, à la gare d'arrivée *mer*, située sur la ligne *bleue*. On supposera que le trajet trouvé par le logiciel Métro contient deux correspondances, *sphinx* et *phare*, situées sur la ligne *jaune*.
L'interface du logiciel est en mode clavier-écran. Pour les distinguer des sorties écran, on fera précéder les entrées tapées au clavier par un prompt *Métro>*.

UML – Diagrammes de classes
Diagrammes d'objets

Exercice 9 : RESTAURANT

Un *restaurant* {est composée de} *tables*. Il est midi. Des *clients* {sont à} table. Des *plats* et des *boissons* (de la *nourriture*) {sont posés sur} les tables. Un client {mange son} plat et {boit sa} boisson. Un client peut être un *adulte* ou un *enfant*. Les boissons peuvent être des *bouteilles de vin*, des *carafes d'eau* ou des *tasses de café*. Un plat peut être une *entrée*, un *plat de résistance* ou un *dessert*. Plusieurs clients peuvent boire la même boisson. Un client mange un seul plat mais peut boire plusieurs boissons. Les bouteilles de vin et les plats ont des prix variables, un café coûte 2 euros et une carafe d'eau est gratuite. Un enfant ne boit ni vin ni café.

1) On suppose que les classes du texte précédent correspondent aux mots ou groupe de mots en *italique*. On suppose qu'un prix est un entier. Dessiner un diagramme de classes (on s'intéresse dans un premier temps aux relations de généralisation).
2) On suppose que les associations correspondent aux mots {entre accolades}. Dessiner les associations entre classes (on placera les ordres de multiplicité 1 ou *).
3) Soit les classes suivantes : Restaurant, Table, Client, Adulte, Enfant, Nourriture, Boisson, Plat, CarafeEau, Café.

- Placer les attributs :
  .prix., .monClient., .mesClients., .maTable., .mesTables., .maBoisson., .mesBoissons., .monPlat., .mesPlats., dans les classes adéquates.
- Compléter la description avec les types des propriétés (int, Liste,…)
- Idem pour les méthodes :
  .void seMettreATable(Table)., .void poserSurLaTable(Table)., .void poserNourritureSur (Nourriture)., .void debarrasserLesTables()., .void boireDuVin(Vin)., .void boireDeLEau(Eau)., .void afficherLesClients()..

4) Dessiner un diagramme d'objets correspondant au texte suivant :

Le restaurant "GL" comprend trois tables.
Laurence et Paul ont une fille Léa.
Ils sont à la table 1 avec Valérie.
Laurence et Valérie boivent une bouteille de bourgogne.
Laurence mange un riz cantonnais.
Valérie mange une salade de tomates.
Paul boit une tasse de café.
Léa mange son dessert et boit de l'eau.
A la table 2, Nacef et Anne-Lise boivent du vin mais n'ont pas faim.
Leur fille Mathilde mangeait une glace à la vanille qui est tombée par terre.
La table 3 est vide.

5) Indiquer les instanciations, généralisations, spécialisations, agrégations, associations ou attributions présentes dans les bouts de phrases suivants :

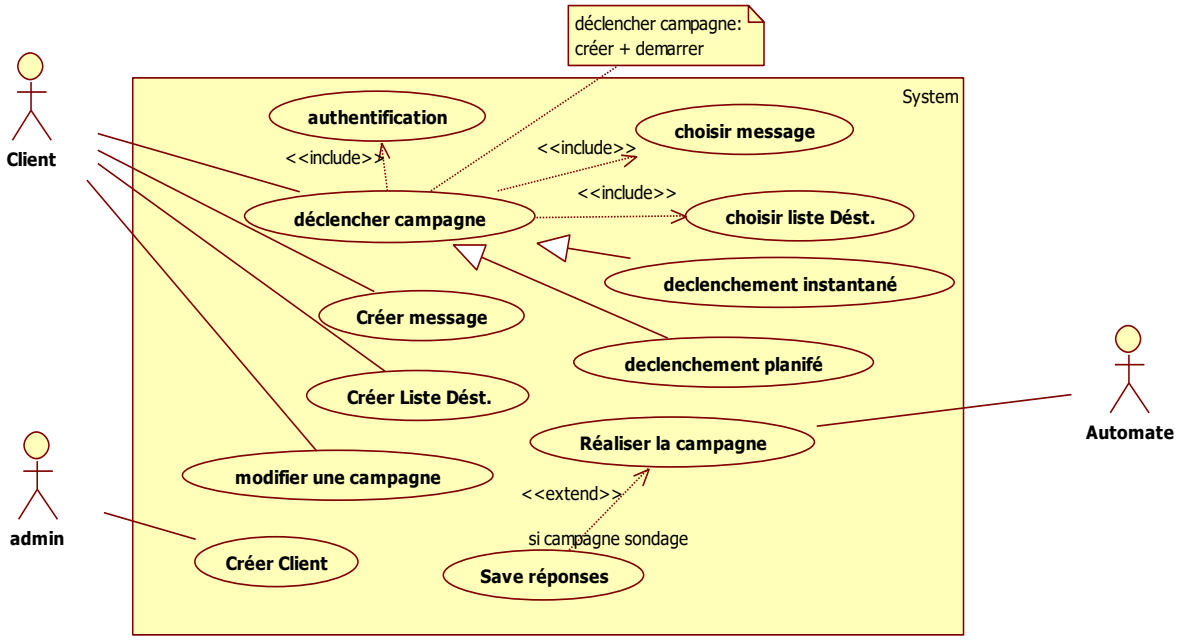
- Roger est un restaurateur.

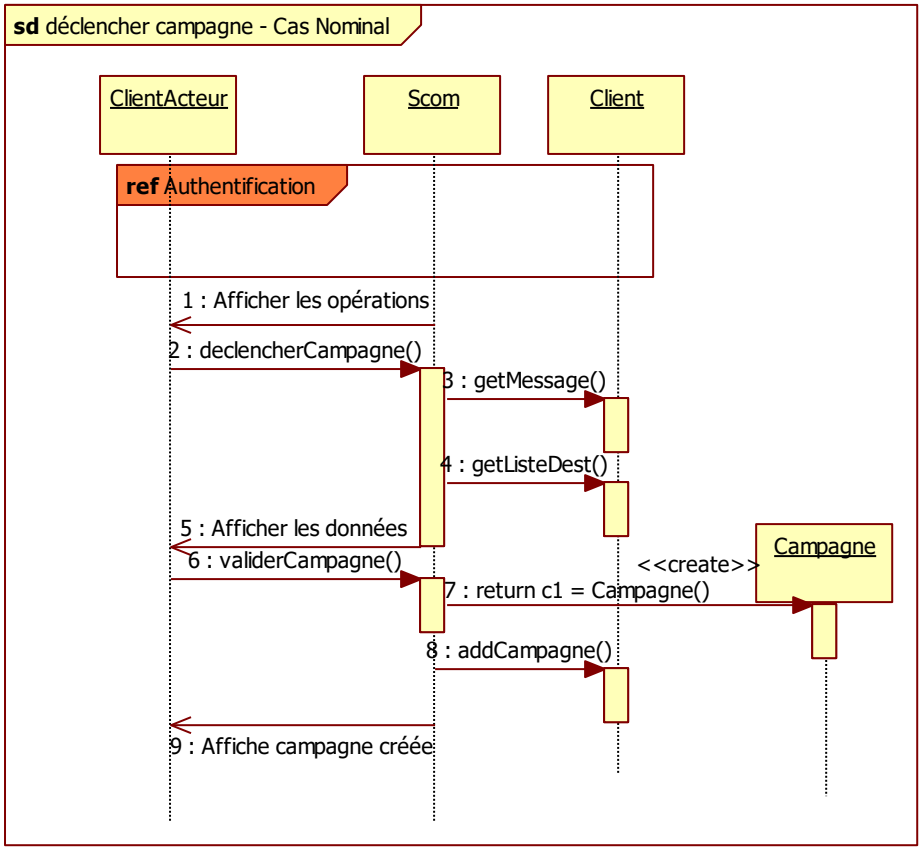
- il tient le restaurant « GL » qui est un restaurant.
- un restaurateur est une personne,
- donc Roger est une personne.
- un repas complet est composé d'une entrée, d'un plat de résistance et d'un dessert.
- un plat est une entrée ou un plat de résistance ou un dessert.
- la table 3 est une table du restaurant ;
- elle est vide.
- un vin est une boisson ;
- une boisson est soit du vin, soit de l'eau, soit du café.
- L'eau est froide ; le café est chaud.
- le café va bien avec le dessert ;
- d'ailleurs le dessert contient du sucre.

Exercice 10 : Com pas Com – la suite

On souhaite tracer le diagramme de classes du système de télécommunication de l'entreprise *ComPcom* (voir la série d'exercice précédente)

Exemple de Diagramme de cas d'utilisation et de séquence :





Le diagramme de séquence doit être corrigé pour permettre une meilleure modélisation orienté objet.

UML – Diagrammes d'états transitions

Exercice 11 : Etat composite

Soit les deux diagrammes d'états A et B. Le diagramme B contient un état composite.

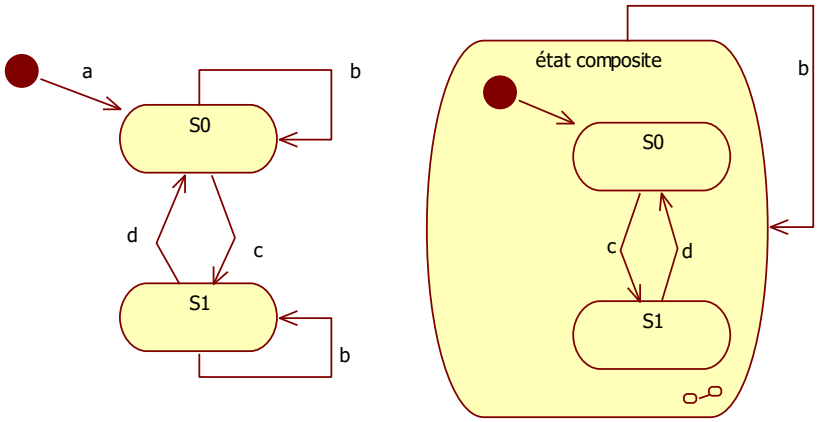


Diagramme d'états A Diagramme d'états B

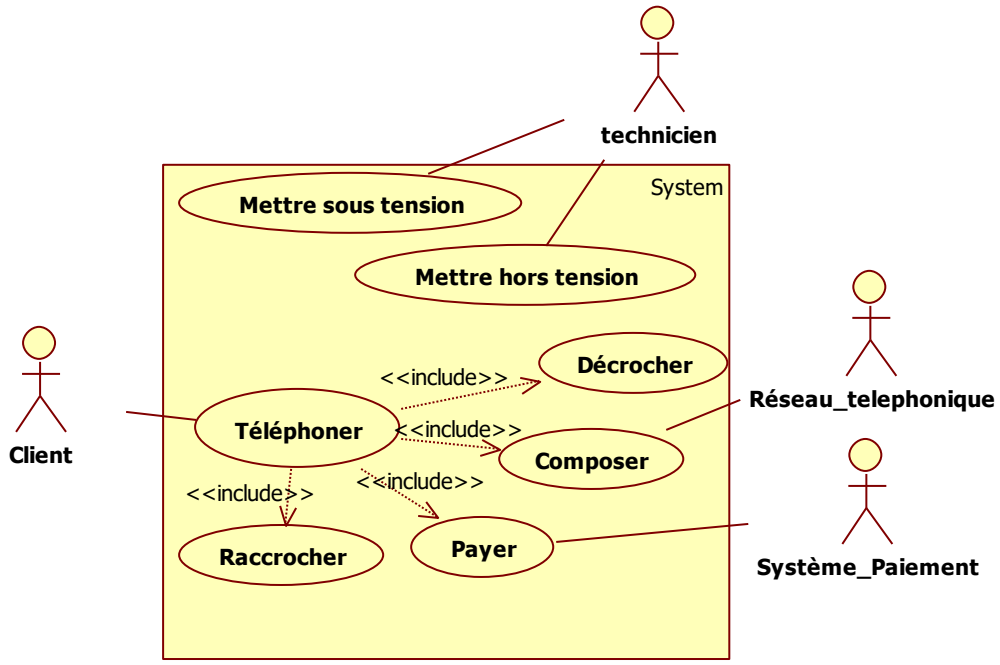
Montrer que les deux diagrammes ci-dessus ne sont pas équivalents en trouvant une séquence d'événements qui n'amène pas dans le même état en A et en B. On supposera que les séquences débutent à l'état initial.

Exercice 12 : Publiphone (Cabine téléphonique)

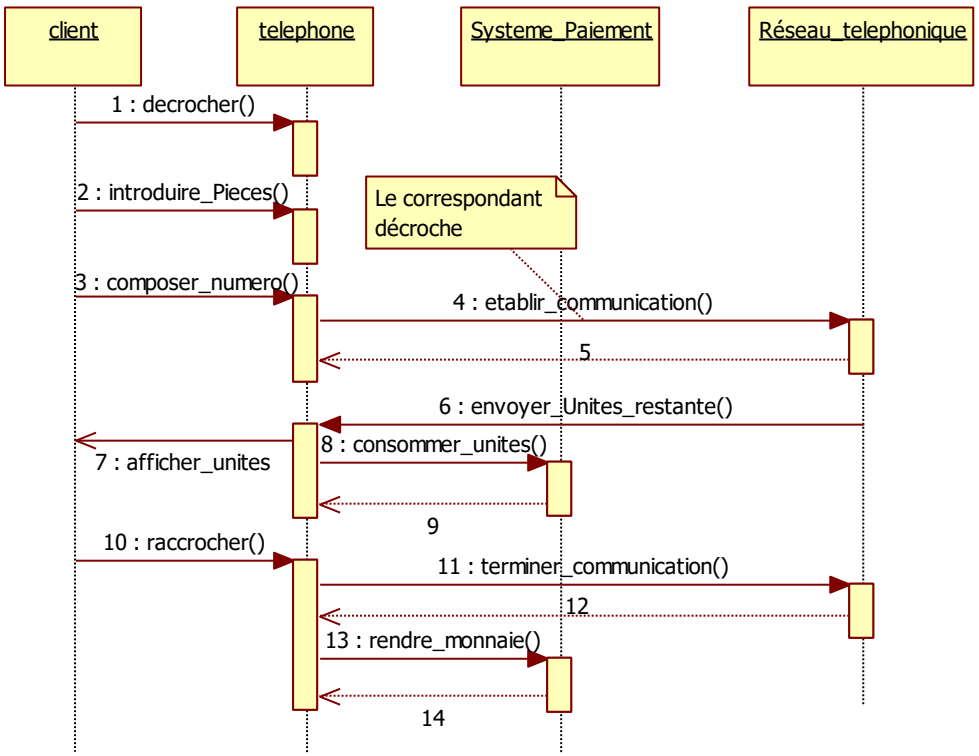
L'objectif est de réaliser une application qui simule le comportement d'un Publiphone à Pièces sur un système d'exploitation classique à partir de quelques périphériques standard dans la micro-informatique: microphone, haut-parleur, lecteur de carte bancaire type *Moneo*.
Pour débuter ce projet, on part de l'existant en analysant un système simplifié de Publiphone à pièces :

- Le prix minimal d'une communication nationale est de 0.1 Euro TTC.
- Après l'introduction de la monnaie, l'utilisateur a 2 minutes pour composer son numéro du correspondant (ce délai est décompté).
- La ligne appelée peut être libre ou occupée.
- Le correspondant peut raccrocher le premier.
- Le Publiphone consomme de l'argent dès que le correspondant décroche, et à chaque unité de temps (U.T.) engendrée par le standard.
- On peut ajouter des pièces à tout moment.
- Lors du raccrochage, le solde de monnaie est rendu.

Exemple de Diagramme de cas d'utilisation du Publiphone à pièces:



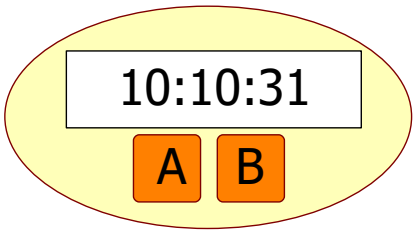
Exemple de Diagramme (préliminaire) de séquence qui décrive le Scénario nominal du cas d'utilisation: *Téléphoner*.



Questions :
1. Réaliser un premier Diagramme d'états qui décrive le comportement nominal du publiphone, d'après le diagramme de séquence.
2. Sur le Diagramme d'états précédent, comment représenter le fait que l'appelant peut raccrocher à tout moment, et pas seulement dans l'état *Conversation* ?

Exercice 13 : Montre

On considère une montre digitale simplifiée, dotée de deux boutons A et B.



L'état courant est l'état *Affichage Heure*. Quand on appuie une fois sur le bouton A **(**bouton Mode**)**, la montre passe en mode *modification de l'heure*. Chaque pression sur le bouton **B (**bouton Avance**)** incrémente l'heure d'une unité. Quand on appuie une nouvelle fois sur le bouton **A**, la montre passe en mode *modification des minutes*. Chaque pression sur le bouton **B** incrémente les minutes d'une unité. Quand on appuie une nouvelle fois sur le bouton **A**, la montre repasse en mode *Affichage*.

Questions :

1. Réaliser un Diagramme d'états correspondant aux spécifications ci-dessus, sans oublier les actions accompagnant certaines des transitions.

2. Pour régler l'heure de l'alarme, il faut exercer une double pression (à l'instar d'un double clic sur une souris) sur le bouton A. La montre s'arrête lorsque l'énergie ne parvient plus à la montre (« batteries (piles) trop faible », « batteries retirées », …).

    a) Expliquer pourquoi le réglage de l'alarme introduit un non-déterminisme dans le comportement de *l'interface utilisateur* de la montre.
    b) Montrer comment réintroduire un comportement déterministe en ajoutant un état supplémentaire non précisé dans l'énoncé.
    c) Ajouter autant de transitions que nécessaire aux états pour représenter le passage dans l'état final. Quelle astuce du formalisme du diagramme d'états pourrait-on utiliser pour alléger la représentation.

Exercice 14 : Tâche (Thread)

Dessiner un **diagramme d'états-transition** correspondant à la dynamique d'un thread définie de la manière suivante :

Le thread est :

non démarré	Au début
en cours	Lorsqu'il possède toutes ses ressources applicatives plus le processeur
en attente	Lorsqu'il lui manque une ressource applicative
Prêt	Lorsqu'il a toutes ses ressources applicatives et pas le processeur
terminé	Lorsqu'il a terminé son exécution

On supposera que les événements reçus par le thread sont :

Début	correspond au démarrage du thread (start en java). avant la réception de « début » le thread est « non démarré »
ressource attendue	correspond à l'appel d'une réservation de ressource lorsque celle-ci n'est pas disponible.
ressource OK	correspond à la libération d'une ressource par un autre thread et donc à la réservation effective de la ressource par le thread qui l'attendait
processeur OK	correspond à la libération du processeur par un autre thread et à l'utilisation effective du processeur par le thread qui l'attendait
Fin	correspond soit à l'exécution de la dernière instruction du programme exécuté par le thread soit à l'envoi d'un événement pour tuer définitivement le thread. Sur réception de « fin », le thread devient « terminé »

On supposera qu'un thread n'envoie pas d'événement. Il ne fait que les recevoir.