Analyse orientée objet

Diagramme d'interactions

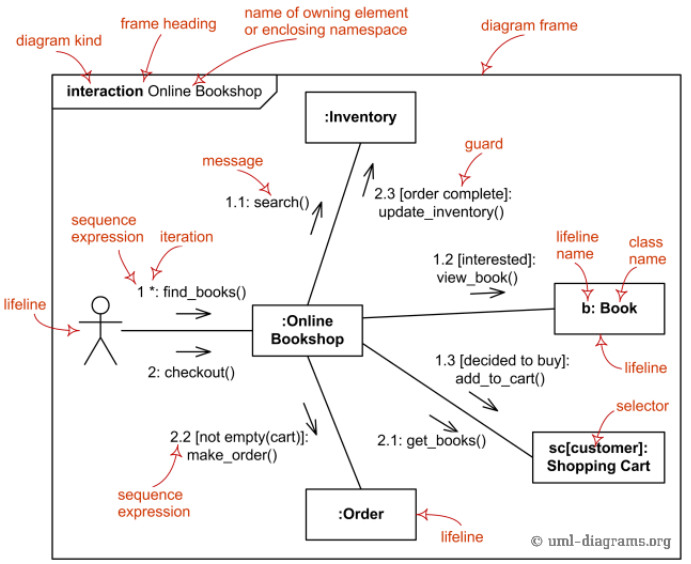
Les diagrammes d'interactions

- Les diagrammes d'interactions montre comment des objets (ou acteurs) communiquent entre eux
- Dans les autres diagrammes (comme le diagramme de classes), les liens indiquent une relation, mais jamais comment les objets communiquent entre eux
- Les diagrammes d'interactions prend un sous-ensemble du système pour montrer les interactions entre les entités

Les diagrammes d'interactions

- Il existe 4 diagrammes d'interactions, mais nous ne verrons que les 2 plus populaires:
  - Diagramme de communication
  - Diagramme de séquence
- Les 2 autres diagrammes sont trop spécifiques
- Vous verrez que les diagrammes d'interactions sont liés au diagramme de classes

Diagramme de communication



Source: https://www.uml-diagrams.org/communication-diagrams.html

Diagramme de communication

- Expression de séquence:
  - Permet de représenter l'ordre des messages. On peut utiliser plusieurs niveaux comme: 1.3 ou 2.4.7
  - Attention à l'ordre: 2 → 2.1 → 2.1.1 → 2.2 → …

Diagramme de communication

- Message:
  - Représente le message envoyé à un objet (souvent le nom d'une méthode de l'objet)
  - Il est possible de passer des paramètres dans les « () » de la méthode
  - [condition] → condition pour envoyer le message (« guard » en anglais)

Diagramme de communication

- Itération:
  - Permet d'indiquer combien de fois le message sera envoyé
  - « * » représente un nombre indéterminé (2, 10, 50, etc)
  - « *[x: 1..10] » représente une itération sur 10 éléments où chaque élément sera représenté à tour de rôle par x (on peut l'utiliser dans les méthodes)
  - « *||[x: 1..10] » comme le point précédent sauf que les itérations se font en parallèle

Diagramme de communication

- Ligne de vie (« lifeline »):
  - Représente l'existence d'un objet
  - Dans un diagramme de communication, son utilité est plus limitée que dans le diagramme de séquence
  - Il est possible de décrire précisément l'objet via une notation

Diagramme de communication

- Ligne de vie (« lifeline »):
  - La notation « nom de l'objet [selector] : type de l'objet »
  - « x: Livre » → il s'agit de l'objet x de type Livre (une instance)
  - « :Livre » → il s'agit d'un objet de type Livre (on dit qu'il est anonyme)
  - « x: » → il s'agit d'un objet x de type inconnu
  - « : » → objet anonyme de type inconnu

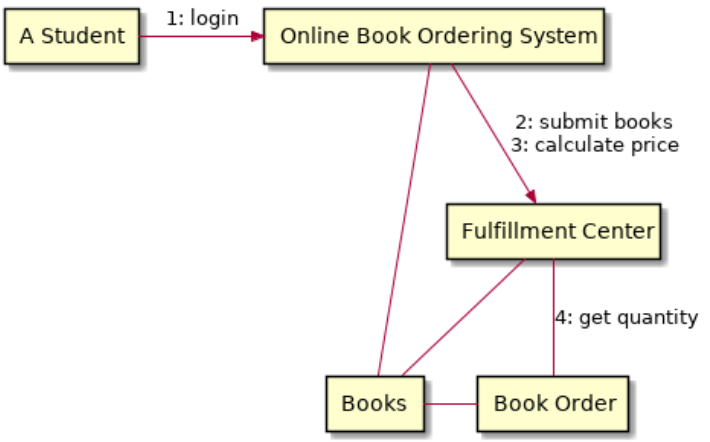
Diagramme de communication

- Ligne de vie (« lifeline »):
  - Pro tips → essayer toujours de typer un maximum vos objets
  - Le sélecteur permet de choisir un objet spécifique dans un ensemble d'objets.
  - Dans l'exemple: on choisit le panier (« shopping cart ») du client parmi tous les paniers du système.

Diagramme de communication

- Est-ce que PlantUML est capable de représenter des diagrammes de communication ?
- Avec un peu de « bricolage », on peut réaliser quelque chose de minimal…

Diagramme de communication



Source: https://bormueller.com/2020/03/07/uml-diagrams-with-plantuml/

Diagramme de communication

Diagramme de communication

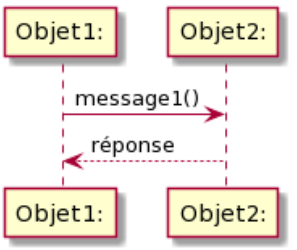
- **Avis personnel**: si vous voulez absolument faire un diagramme de communication, utilisez un autre logiciel que PlantUML. Si vous vous désirez n'utiliser que PlantUML, dirigez-vous vers un diagramme de séquence.

Diagramme de séquence

- Les diagrammes de séquence et ceux de communications se ressemblent beaucoup
- Logique, car leur but est le même: représenter les interactions
- Cependant, le diagramme de séquence permet une représentation plus fine et permet de représenter visuellement l'ordre d'exécution

Diagramme de séquence

- Ligne de vie:
  - Elle est représentée par un rectangle suivi d'une droite en pointillé.
  - Suit la même notation que le diagramme de communication

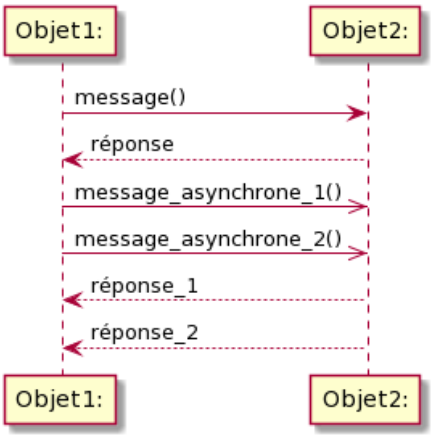


Source:
https://www.planttext.com/?text=SoWkIImgAStDuU9Iya_AIYqnj59IqBLJW72CW1mhXTpKukB4z5H33KqkXB0kN15GXvL2qU6L1VbvnQbSN5mEgNafG0C1

Diagramme de séquence

- Message:
  - Mêmes principes que le diagramme de communication
  - Possibilité d'avoir un message synchrone (on attend la réponse) ou d'avoir un message asynchrone (on continue sans tenir compte de la réponse).
  - Message synchrone → « flèche à tête pleine »
  - Message asynchrone → « flèche à tête ouverte »

Diagramme de séquence

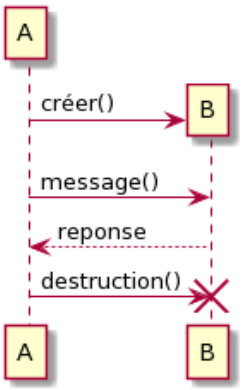


Source:
https://www.planttext.com/?text=SoWkIImgAStDuU9Iya_AIYqnj59IqBLJW72CW1mhXTpKukB4z5GDJIw4g2vS3L23bKBHuPK5-Nd5gIY689a2DoG-iRWoBpcZA3ylDTuGRY9nwes8SK6y8H5gZBYuk1nIyrA0_WO0

Diagramme de séquence

- Message de création/destruction:
  - Le message de création crée un objet → La flèche va sur un rectangle
  - Le message de destruction détruit un objet (très pratique pour des langages comme C/C++ où on doit gérer la mémoire dans le code) → La flèche se termine par un « X »

Diagramme de séquence

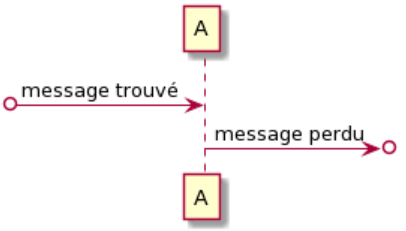


Source:
https://www.planttext.com/?text=SoWkIImgAStDuU9oLD2rKt3Iqh9II2uwl3ArI4CJAcIba9jQN9oOdWeKSLBG1GevMYaKfHRavnMdGfKeA2fQAQIa5fSKbIQNPERd0LLoSJcavgK0JGO0

Diagramme de séquence

- Message perdu/trouvé:
  - Un message perdu est un message dont l'expéditeur est connu mais il n'arrive pas à destination
  - Un message trouvé est un message qui arrive à destination mais l'expéditeur est inconnu

Diagramme de séquence

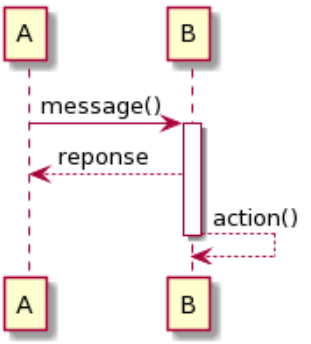


Source:
https://www.planttext.com/?text=SoWkIImgAStDuUAApjUrSxHIo4qjBavCJrKeASelBJkyaijHrou_5Y5Sa5gKKie510wfUIb0VG00

Diagramme de séquence

- Activation:
  - On peut montrer qu'un objet est actif via un rectangle sur sa ligne de vie.
  - Généralement, un objet est actif à partir de la réception d'un message jusqu'à l'envoi d'une réponse
  - Cependant, il est envisageable d'avoir des cas où l'activation continue après l'envoi de la réponse
  - Possibilité de chevauchement d'activation

Diagramme de séquence



Source:
https://www.planttext.com/?text=SoWkIImgAStDuUBYSbJGjLDmibB8JIqkJanFrT3aSamkoInBB4bLSE9oKj05oZfQAHIb5kJd5QT21O3A4u6AylE0gbrI4n7gkHnIyrA0vW80

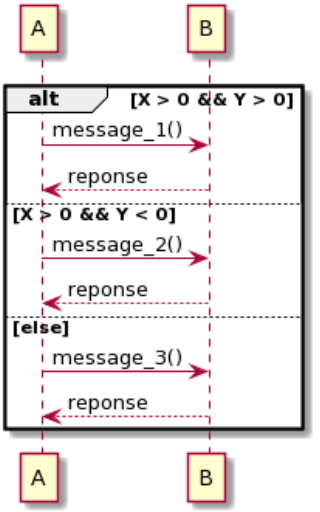
Diagramme de séquence

- Les fragments d'interaction combinés permettent d'introduire un comportement non linéaire dans le schéma
- Ainsi, il est possible de représenter des « if..else… », des exécutions parallèles, etc.
- Les plus importants seulement seront vus

Diagramme de séquence

- Alternative (alt):
  - Permet de représenter deux (ou plus) exécutions possibles grâce à des conditions
  - Il est possible d'utiliser la garde « else » pour avoir une exécution si aucune autre condition n'est remplie
  - Il faut au moins 2 exécutions possibles
  - Correspond à « if.. else… » ou « if… else if… else… »

Diagramme de séquence

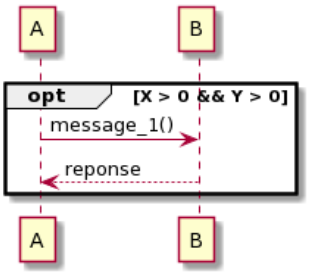


Source:
https://www.planttext.com/?text=SoWkIImgAStDuU9Ap2bHY52mKp1GK5DJY0Gnk1mLTEqKdAmKSbEBYnFJK-CDDJIvd1HqWOAELWf5gGNvUSMfNAavnQd86csmQ3J2gH54O2WtngeyBuKB21o0DD2z0000

Diagramme de séquence

- Optionnel (opt)
  - Comme alt sauf qu'il n'y a qu'un seul chemin qui est optionnel (en fonction de la condition)
  - Correspond à un « if… »

Diagramme de séquence

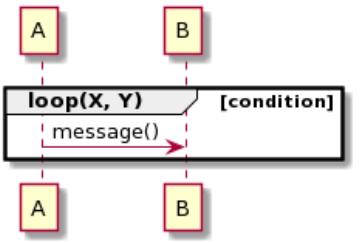


Source:
https://www.planttext.com/?text=SoWkIImgAStDuUBABoXHY52mKp1GK5DJY0Gnk1mLTEqKdAmKSbEBYnFJK-CDDJIvd1HqWOAELWf5gGNvUSMfNAbvALn0u02aWUO00000

Diagramme de séquence

- Boucle (loop):
  - Permet de répéter une séquence selon des conditions
  - Il est possible d'indiquer un minimum d'itérations, un maximum et/ou une condition sous cette forme: loop(X, Y) [condition]
  - Si X = Y, on peut omettre un des deux
  - Si la condition n'est plus respectée, on stoppe les itérations même si X itérations n'ont pas été faites
  - Correspond à un « for » ou un « while »

Diagramme de séquence

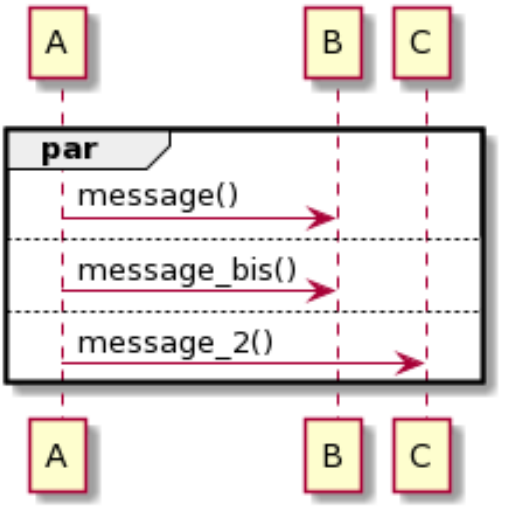


Source:
https://www.planttext.com/?text=SoWkIImgAStDuU9AByelBL38oSylq8ZGKOZKL8XEpizBoIp9pC-BvN9KqBLJSB9Io4qjBavCJzNGv4hDI-420WSW3TGE0000

Diagramme de séquence

- Parallèle (par)
  - Permet d'indiquer que deux (ou plus) séquences d'exécution se font en parallèle

Diagramme de séquence



Source:
https://www.planttext.com/?text=SoWkIImgAStDuU8gI2pYKW02HmLTEmKdAuNSrEBYnFHK3Kskr9pYL6nIyKcPnHZIpWXf8v1aNWeN43W0QQ0j0000

Exercice

- Représentez l'énoncé suivant avec un diagramme de séquence:

Exercice

Quand un client arrive sur le site web d'un cinéma, le site lui affiche les films projetés pour la journée. Quand le client clique sur une des affiches, les informations sur la séance lui sont données (heure, salle, etc). S'il essaie d'acheter un ou plusieurs tickets, le serveur vérifie qu'il y aura assez de places. Si ce n'est pas le cas, le serveur affiche qu'il n'y a plus de place. Si c'est le cas, le serveur va réserver les places et générer en même temps un mail avec un code barre. Ce mail sera ensuite envoyé au client qui n'aura plus qu'à le présenter à l'accueil

PlantUML

- Documentation pour les diagrammes de séquence: https://plantuml.com/fr/sequence-diagram