

# SAÉ 3 - R3.05

## Alizon - Délivraptor

### Contexte

La plateforme de vente en ligne Alizon va confier ses colis à des prestataires de transport. Ces prestataires fournissent un outil permettant :

- La prise en charge du colis
- L'interrogation de la situation du colis dans la chaîne de livraison
- La validation de la livraison

Vous allez concevoir et développer un simulateur de livraisons pour l'un de ces prestataires.

### Terminologie

Voici les termes utilisés dans ce document et que vous devez aussi utiliser dans la documentation à produire.

**Serveur** ou **Service** : le logiciel simulateur qui attend et traite les requêtes.

**Client** : le logiciel qui se connecte au serveur. Il s'agit du logiciel utilisé par Alizon pour interroger le service. Ne pas confondre avec le client destinataire (voir plus bas).

**Protocole** : les règles et la grammaire permettant à un client de soumettre des requêtes au serveur.

**Grammaire** : la syntaxe décrivant comment doivent/peuvent être formatées les requêtes du protocole.

**Requête** : une action faite sur le serveur par le biais d'un client en utilisant le protocole.

**Destinataire** : l'acheteur du produit à livrer. Rappel : ne pas confondre avec le "client" du service (voir plus haut).

# Attendus

## Protocole

Vous devrez concevoir un protocole pour vous identifier et interroger ou faire des actions sur le simulateur.

Vous devrez documenter ce protocole pour en expliquer les règles (sa grammaire, cf TP sur les Sockets)

## Langages et outils

Vous devrez fournir le code source ainsi que la documentation nécessaire à la compilation de votre serveur exclusivement sur une machine Linux.

### Langage C

Le simulateur devra être écrit en langage C, mettre en œuvre une programmation de sockets.

### Options

Le serveur devra être lancé en ligne de commande et disposer d'options (qui sont détaillées au fil de ce document).

Vous pouvez choisir la technique que vous souhaitez pour le passage des options, mais sachez qu'il existe une fonction prévue pour ça : **getopt()**.

### Langage PHP

Pour accéder au service depuis les pages du site Alizon, vous utiliserez évidemment PHP mais pour tester rapidement le protocole durant sa mise au point, vous pouvez aussi utiliser **telnet**.

## Installation des outils de développement

Vous avez 2 options possibles : une installation directe sur le serveur Proxmox ou un conteneur Docker.

### Installation directe (conseillé)

Vous devez installer les outils de développement sur votre serveur, en **root** :

```
apt-get update  
apt-get install gcc-15 build-essential libc6-dev
```

Travaillez dans un dossier **/docker/sae/delivraptor**

Docker (si vous avez la foi)

Le plus simple est d'utiliser une image officielle de GCC en version 15. Demandez-moi si vous optez pour cette solution.

## Documentation

Vous devrez produire une documentation suffisamment détaillée pour pouvoir compiler et exécuter votre serveur. Ceci doit pouvoir être fait par un développeur, sans nécessiter de connaissances particulières autre que savoir lancer des commandes dans un Terminal.

# Étapes de transit d'une livraison

Nous avons identifié les 9 étapes suivantes dans le processus de livraison d'un colis.

## Étape 1 - Création d'un bordereau de livraison

Au moment du paiement de la commande, le script PHP sur Alizon doit interroger le service du transporteur afin d'obtenir un numéro de bordereau de livraison qui constituera l'identifiant du colis chez le transporteur.

Le service doit mémoriser le numéro de commande Alizon, et en retour, le service émet un numéro de bordereau de livraison interne. Cette information doit être conservée sur Alizon car elle est l'identifiant permettant de suivre le colis. Si Alizon appelle le service plusieurs fois pour un même numéro de commande, le numéro de bordereau doit toujours être le même, il ne faut pas régénérer un nouveau bordereau s'il en existe déjà un pour une commande Alizon donnée.

À ce stade, le colis est toujours chez Alizon.

## Étape 2 - Prise en charge du colis chez Alizon

Le colis a quitté Alizon et est en cours d'acheminement vers la plateforme du transporteur.

## Étapes 3 et 4 - Chez le transporteur

Le colis arrive sur la plateforme du transporteur qui va le placer dans un camion en direction de la plateforme d'aiguillage régionale.

L'étape 3 est celle de l'arrivée chez le transporteur.

L'étape 4 est celle du départ vers la plateforme régionale.

## Étape 5 et 6 - Sur la plateforme régionale

Le colis arrive sur la plateforme d'aiguillage régionale. C'est cette plateforme qui envoie le colis vers les centres locaux proches du client final.

L'étape 5 est celle de l'arrivée sur la plateforme régionale.

L'étape 6 est celle du départ vers le centre local.

## Étape 7 et 8 - Au centre local

Le colis arrive au centre local proche du client final.

Le centre va confier le colis soit à un de ses livreurs soit à un prestataire partenaire indépendant qui travaille en sous-traitance.

L'étape 7 est celle de l'arrivée au centre local.

L'étape 8 est celle du départ pour la livraison finale.

## Étape 9 - Livré ou refusé

Le colis est livré au destinataire final qui, soit a accepté la livraison, soit était absent et le colis est considéré livré, soit a refusé le colis.

Si le colis est livré en l'absence du destinataire, une image de la boîte aux lettres ouverte avec le colis à l'intérieur doit être remontée vers le client. Par simplicité, vous utiliserez toujours la même image, locale au service. Cette image doit être dans un fichier permettant ainsi de la changer aisément lors des tests.

Si le destinataire refuse le colis, la raison du refus du colis doit être remontée vers le client.

Si le colis est livré en mains propres, cette information doit remonter vers le client.

## Complément

Les appels au service doivent renvoyer les informations spécifiques à l'étape à laquelle se trouve un colis. À vous de proposer judicieusement ce que votre service renvoie.

Pensez à horodater toutes les étapes, notamment pour servir dans les réponses du protocole.

A l'exception de l'étape 1, vous ne choisissez pas à quelle étape se trouve un colis. C'est le simulateur (ci-après) qui fera la progression.

Attention, l'étape 9 est particulière car elle est (en principe) la seule à renvoyer une partie "binaire" : l'éventuelle image de la boîte aux lettres !

## Base de données

Par simplicité, vous pouvez utiliser la même base de données qu'Alizon pour stocker les états de transit des colis, dans des tables dédiées, mais le site Alizon **ne doit jamais** accéder à ces données lui-même. Tout doit se faire par l'accès au service Délivraptor (via sockets).

## Simulation de la progression

À chaque quantum de temps qui s'écoule, le transit de chaque colis progresse d'une étape.

Vous devez créer un script PHP autonome pour simuler l'écoulement du temps. Chaque lancement simule donc l'écoulement de **1** quantum de temps.

L'étape 9 est particulière car elle est associée à une image (boîte aux lettres témoin de livraison). Vous n'avez à coder ni la soumission ni le stockage de cette image. Pour la simulation de cette étape 9, votre script choisira aléatoirement un des 3 états possibles :

- livré en mains propres
- livré en l'absence du destinataire + photo
- refusé par le destinataire + une raison aléatoire parmi une liste de votre choix

## Exécution du script

Vous avez 2 possibilités, soit manuellement, soit à intervalles réguliers.

### Manuellement

Lancez le script manuellement si vous voulez un quantum de temps inférieur à la minute. Pratique pour les tests.

## Par **cron**

**cron** est un mécanisme Unix qui permet de lancer des commandes à intervalles réguliers ou à certains moments précis. La **crontab** est un fichier de configuration de **cron**.

Pour installer une ligne en **crontab**, tapez **crontab -e**, puis saisissez une ligne au format suivant :

```
m h dom mon dow /chemin/vers/script 2>&1 > /chemin_absolu_vers_fichier_logs
```

où :

- **m** : à quelle minute ou **\*** pour “chaque minute”
- **h** : à quelle heure ou **\*** pour “chaque heure”
- **dom** : quel jour du mois (dom = day of month) ou **\*** pour “chaque jour”
- **mon** : quel mois (mon = month) ou **\*** pour “chaque mois”
- **dow** : quel jour de la semaine (dow = day of week) ou **\*** pour “n’importe quel jour”  
**0** ou **7** = dimanche, **1** = lundi etc.

sauvez et quittez. Pour vérifier que la crontab est en place, tapez **crontab -l**

Ainsi, la granularité la plus fine sera de 1 min.

Chaque utilisateur dispose d’un accès à **cron** via une **crontab** personnelle. Pour vos tests il est sans doute préférable d’installer une **crontab** pour **root**.

Pour empêcher un lancement programmé, il suffit de mettre sa ligne en commentaire dans la **crontab** (en éditant avec **crontab -e**) en ajoutant un **#** en début de ligne à mettre en pause.

# Protocole

Votre protocole doit permettre de faire les actions suivantes et vous devez proposer une grammaire précise et sans ambiguïté.

Rappel : un protocole est un échange de messages formalisés entre deux pairs, le client et le serveur. Cet échange n’est pas forcément une simple question suivie d’une simple réponse. Il peut (et c’est souvent le cas) être constitué de plusieurs échanges successifs durant la connexion.

## Identification

Pour pouvoir faire une quelconque action, le client doit être authentifié au préalable.

La liste des identifiants possibles et des mots de passe associés (hashés en **MD5**) sera placée dans un fichier dont on passera le chemin par une option au lancement.

## Prise en charge

Le transporteur a une capacité de prise en charge limitée. Elle est variable et paramétrable par une option passée au lancement du service. Afin de faciliter vos tests, vous utiliserez une petite valeur (2 ou 3 par exemple).

La prise en charge n'est plus possible si la file de livraison est pleine. Dans ce cas, c'est l'Étape 1 qui s'en trouve bloquée. Votre client PHP doit réagir en conséquence si cette situation survient.

Dès qu'une place se libère dans la file, la prise en charge est de nouveau possible.

Un colis passant de l'Étape 4 à l'Étape 5 libère une place dans la file.

## Service

Votre service est mono-processus et les états sont stockés en BDD afin de pouvoir offrir une persistance au redémarrage du service.

Votre service doit afficher une aide en ligne par l'utilisation d'une option **--help** et, dans ce cas, s'arrêter immédiatement. Inspirez-vous de ce qu'un **gcc --help** affiche, par exemple.

Le service écoute sur toutes les interfaces et sur un port que vous devrez passer en option de lancement sur la ligne de commande.

Vous devez produire des logs détaillés, dans un fichier, afin de suivre précisément ce que fait, ce que reçoit, ce que répond, votre service, comment il a été lancé, avec quelles options, etc. Préfixez chaque ligne de log par une date+heure ainsi que l'IP du client<sup>1</sup>.

## Client du service

Vous pouvez utiliser un client **telnet** durant la mise au point du protocole, mais le vrai client final du service que vous allez créer est le site Alizon lui-même !

Vous devez coder la partie cliente en utilisant les fonctions PHP :

**fsockopen + fread + fwrite + fclose.**

La page de paiement doit être celle qui déclenche la génération du numéro de bordereau de livraison (Étape 1)

---

<sup>1</sup> Voir le paramètre 2 du **accept()**

La page de consultation d'une commande payée doit être celle qui interroge de service de livraison pour connaître l'état de la livraison et doit afficher l'image de la boîte aux lettres si cette information existe.

# Livrables

Voici la liste des livrables attendus, à déposer sur Moodle.

L'absence de certains livrables ou le non-respect des consignes ou des noms des fichiers impactera la note finale. Les fichiers **.md** sont en markdown et rien d'autre, merci.

- Tout ce qui suit doit être dans une **archive <num\_equipe>.zip**, dans les dossiers indiqués.
- **[dossier src/]** Code source C du serveur et autres éventuels outils complémentaires + Documentation pour compiler et pour exécuter en ligne de commande.
- **[fichier doc/proto.md]** Protocole détaillé (technique)
- **[fichier doc/usecase.md]** Documentation de cas d'utilisation par l'exemple (pratique) : se logger, soumettre une prise en charge, interroger la situation.
- **[dossier php/]** Code PHP de la partie cliente du site Alizon et rien d'autre (uniquement les pages PHP concernées)..
- **[fichier video.md]** Un lien Web vers une vidéo de démonstration complète de tous les aspects et fonctionnalités demandées :
  - Démarrage
  - Paramètres en ligne
  - Connexion au service
  - Création d'une prise en charge d'un nouveau colis
  - Passage par toutes les étapes (en mode manuel du script de progression)
  - Affichage des retours du service
  - Affichage des images collectées (boîte aux lettres)
  - Interrogation et affichage de l'état d'un colis
  - Interrogation et affichage des colis en transit

La vidéo doit être capturée en **1280** pixels de largeur, toutes les fenêtres doivent occuper **100%** de cette largeur. Attention, il ne s'agit pas d'enregistrer un écran en 4K et de réduire à 1280, ça serait flou !

Le son doit être parfaitement audible.

- **[fichier doc/participation.md]** Une répartition (en %) du travail individuel par tâche. Au minimum ceci, à détailler/compléter si utile :
  - Conception du protocole
  - Codage et tests
  - Documentation