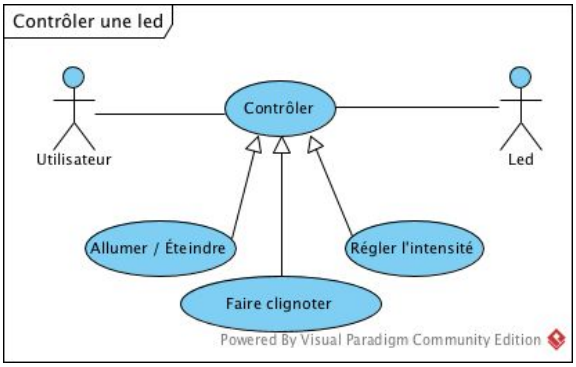
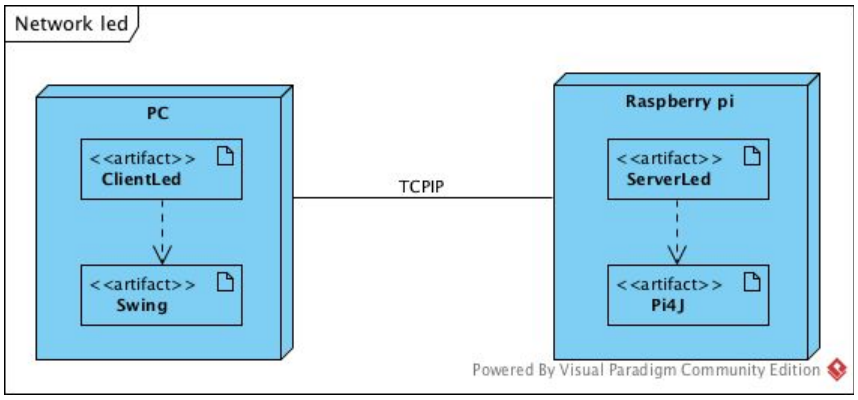
TP “Network Led”

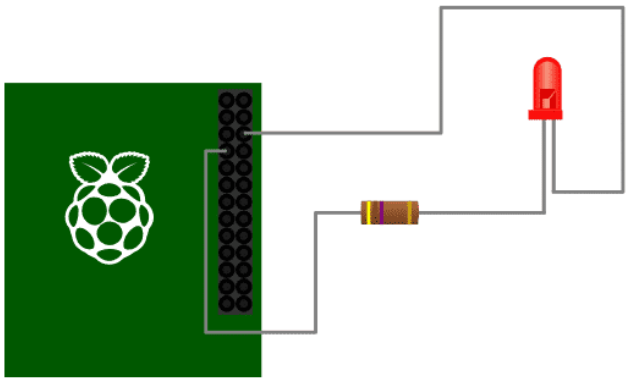


On souhaite piloter un éclairage avec une application communicant à travers le réseau local. L’application sera un client développé en Java swing (IHM) et l’éclairage sera représenté par une led connectée à un Raspberry pi configuré comme serveur.



La led sera connectée au port GPIO.1 du Raspberry pi et l’application permettra de :

- Allumer ou éteindre la led,
- Définir une vitesse de clignotement,
- Définir une intensité lumineuse.



Travail demandé :

1. Concevoir un protocole de communication le plus simple possible (trame ASCII = chaîne de caractères) permettant au client d'envoyer au serveur des commandes permettant d'obtenir les fonctions souhaitées. Le serveur devra renvoyer un acquittement au client pour indiquer si la commande a été exécutée ou pas.
2. En utilisant un programme client déjà réalisé en TP, développer progressivement une application console serveur pour le Raspberry pi en validant les étapes suivantes :
    2.1. Acceptation de la connexion du client sur le port 2222
    2.2. Affichage des commandes reçues du client
    2.3. Décodage des commandes reçues du client
    2.4. Conception en UML d'une classe "Led" répondant aux fonctionnalités de l'application
    2.5. Codage et test de la classe Led
    2.6. Conception en UML d'un diagramme de séquence montrant l'appel des méthodes de la classe Led en fonction des commandes du client
3. En utilisant le programme serveur réalisé à l'étape 2, développer progressivement une application graphique client en validant les étapes suivantes :
    3.1. Saisie des paramètres et demande de connexion / déconnexion
    3.2. Affichage de l'état de la connexion
    3.3. Ajout de composants graphiques pour allumer / éteindre la led
    3.4. Ajout de composants graphiques pour faire clignoter la led et régler la vitesse de clignotement de 1 à 10 (clignotements par secondes)
    3.5. Ajout de composants graphiques pour régler l'intensité de la led de 1 à 100 (%)
    3.6. Rétro-ingénierie du diagramme de classes de l'application client
4. Apporter les améliorations suivantes :
    4.1. L'application client doit recevoir les réponses du serveur, selon le protocole créé à l'étape 1.
    4.2. La classe Led doit pouvoir être instanciée pour n'importe quelle broche du GPIO
    4.3. L'application serveur doit se remettre à l'écoute si le client se déconnecte
    4.4. Il n'est pas possible d'utiliser un contrôle (ex : allumer) si le client n'est pas connecté
5. Rédiger une fiche de tests fonctionnels :
    5.1. Prévoir les tests des scénarios nominaux prévus par les cas d'utilisation
    5.2. Prévoir les tests d'exception (ex: les contrôles sont désactivés si le client n'est pas connecté)
    5.3. Les tests seront faits par quelqu'un qui n'a pas développé l'application testée
6. Documents à fournir pour la note finale :
    6.1. Diagramme de cas d'utilisation
    6.2. Diagramme de classes de l'application montrant :
        6.2.1. La classe ServeurLed
        6.2.2. La classe ClientLed
        6.2.3. La classe Led