

Développement Logiciel, 2013

Projet - version simple

Un réseau **ad hoc** est un réseau sans fil décentralisé, qui comprend des machines (nœuds) à la fois statiques et mobiles (PC, ordinateurs portables, téléphones mobiles). Un réseau ad hoc ne repose pas sur une infrastructure préexistante, tels que les routeurs dans les réseaux câblés ou les points d'accès qui gèrent les réseaux sans fil. Au lieu de cela, chaque nœud participe au routage des unités des données (« paquets ») vers les autres nœuds. Le routage des paquets est déterminé dynamiquement en fonction de la connectivité du réseau. En plus du routage classique, les réseaux ad hoc peuvent utiliser les inondations (« flooding ») pour la transmission des données.

Sujet du projet : Il s'agit de développer une petite suite logicielle qui permettra de créer un réseau ad-hoc, dans lequel on transmet des paquets, et de simuler leurs déplacements jusqu'à ce qu'ils atteignent leur objectif : arriver dans un nœud de destination.

Cette suite est composée de trois programmes, dont nous décrivons pour chacun les caractéristiques dans la version initiale (a) et la version avancée (b) :

1. **L'usine** : permet la conception des types de nœuds pour notre réseau :
 - (a) Définition de l'aspect visuel et des propriétés des nœuds,
 - (b) Optimisation des compétences de transmission des messages et de mouvement des nœuds.
2. **L'éditeur de réseau** : permet la conception d'un monde (grille à deux dimensions de cellules) :
 - (a) Choix de l'aspect visuel du monde, de la position des nœuds statiques et des « blocages ».
 - (b) Changement des propriétés de cellules du monde qui affectent la transmission des paquets.
3. **Le simulateur de réseau** : simule la transmission des paquets dans le réseau :
 - (a) Simulation naïve où les nœuds sont statiques (mais peuvent disparaître) et où la distance entre nœuds influence la transmission,
 - (b) Prise en compte des obstacles et autres contraintes spécifiées par le monde, ainsi que le mouvement et les compétences de transmission des paquets, et l'interaction directe sur le monde (IHM).

Informations générales

Sur le site <http://dev-log-wiki.lri.fr/> vous trouverez des éventuels compléments d'information ainsi que les documents utiles à la réalisation du projet. Ce projet s'étend sur les six dernières semaines d'enseignement (jusqu'à la dernière séance de TP).

On vous demande de mettre en œuvre les différents concepts et outils vus en cours. En particulier, les trois programmes disposeront des éléments suivants :

1. Une interface graphique (fenêtre et dessin)
2. Un système de sauvegarde et chargement
3. Une utilisation des threads pour la simulation (chaque nœud devra être géré dans un thread indépendant)

De plus, d'autres éléments vus dans le cours de Développement Logiciel (ou dans vos autres cours portant sur Java) devront être mis en œuvre pour garantir la lisibilité de votre code : utilisation de concepts d'objets, des exceptions, commentaires, etc. . .

Evaluation

Le projet se découpe en six parties : les trois premières sont consacrées à l'élaboration des versions initiales de chacun des programmes, les deux dernières concernent les versions avancées. Chaque partie du projet est notée sur 4.5 (27/25 au total, 2 points bonus).

Les critères d'évaluation sont les suivants :

- qualité du code (organisation et structure, clarté)
- mise en œuvre des concepts vus en cours
- qualité de la soutenance
- qualité du rapport et de la méthodologie (voir les cours de *Génie Logiciel*)
- originalité

Des malus seront attribués en cas de retard, de non fonctionnement du code, d'absence de rapport...

L'évaluation proprement dite aura lieu lors de deux soutenances avec votre chargé de TP :

1. La **première soutenance** sur les versions initiales aura lieu lors de la **3e séance** de TP-projet : **5 avril**
 - Présentation des versions initiales des programmes.
 - Durée de **15 minutes** maximum.
 - Rendu du **rapport imprimé** de 5 pages maximum comportant :
 - une brève introduction des fonctionnalités de chaque programme,
 - des copies d'écran illustrant le fonctionnement de chaque programme,
 - un bref résumé des fonctionnalités envisagées pour la version avancée,
 - un bref résumé des bonnes pratiques de programmation mises en œuvre.
 - Remise des **sources** de votre programme **sans bugs** sous la forme d'un fichier ZIP ou TGZ par mail.
2. La **seconde soutenance** sur les versions avancées aura lieu lors de la **dernière séance** de TP : **26 avril**
 - Présentation des versions avancées des programmes.
 - Durée de **15 minutes** maximum.
 - Rendu du **rapport imprimé** de 5 pages maximum comportant :
 - une brève introduction des fonctionnalités de chaque programme,
 - des copies d'écran illustrant le fonctionnement de chaque programme,
 - un bref résumé des difficultés rencontrées et des perspectives pouvant être envisagées,
 - un bref résumé des bonnes pratiques de programmation mises en œuvre,
 - et en plus vos observations sur les paramètres de la simulation.
 - Remise des **sources** de votre programme **sans bugs** sous la forme d'un fichier ZIP ou TGZ par mail.

Un point **crucial** est qu'il est **absolument** nécessaire de fournir un programme fonctionnel lors de la soutenance, c'est à dire **sans bugs**. Il est plus important de garantir que votre programme fonctionne plutôt que d'ajouter une option sans la tester. Les sources de votre programme doivent être envoyées à votre chargé de TP, **en mettant votre chargé de cours en copie**.

Par ailleurs, il est primordial d'arriver à la soutenance avec un **rapport imprimé** (il peut être en noir et blanc) comportant vos noms, prénoms, l'UE et la date.

Finalement, vous devez traiter les présentations avec la même attention que l'examen. Vous devez avoir votre rapport avec vous, être préparé pour votre présentation, et être présent à l'heure.

Usine

Version initiale à rendre pendant la 3e séance de TP-projet

L'usine permet à l'utilisateur de :

- charger la description (propriétés) d'un type de nœud existant,
- créer un nouveau type de nœud en modifiant les propriétés d'un type existant.

Fonctionnalités souhaitées :

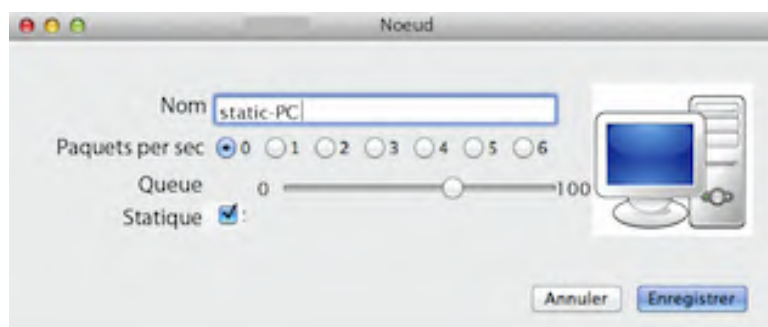
- Création, sauvegarde et chargement de type de nœud.

Ressources :

- Vous trouverez sur le site du cours une archive contenant des fichiers images des nœuds.



(a) sélection, options pour créer, supprimer, etc



(b) génération de nouveaux types de nœud

FIGURE 1 – Usine version initiale

Objectif : Permettre la création d'un type de nœud à partir de zéro, aléatoirement ou à partir d'un autre.

- Chaque type de nœud est défini par :
- une vitesse de production des données (paquets) par seconde à envoyer aux autres nœuds (un entier entre 0 et 6),
- un nom,
- une taille de queue de paquets (entre 10-100, nous allons explorer d'autres alternatives plus tard),
- un paramètre « Vrai /Faux » qui définit si le nœud est statique ou non (mobile),
- une portée de communication définie en pixels
- et une représentation visuelle, par exemple une image ou une forme géométrique.

Éditeur de réseau

Version initiale à rendre pendant la 3e séance de TP-projet

L'éditeur de réseau permet à l'utilisateur de :

- charger la description (propriétés) d'un réseau existant,
- créer un nouveau réseau en modifiant les cellules pour y placer les nœuds statiques,
- ajouter une option permettant de « bloquer » quelques cellules pour une certaine période.

Fonctionnalités souhaitées :

- Sauvegarde et chargement de réseaux,
- Choix des nœuds statiques seulement (types des nœuds spécifiés dans l'usine) à mettre dans les cellules,
- Possibilité de spécifier si une cellule est bloquée pour une période aléatoire ou définie par l'utilisateur (donc on ne peut pas y accéder). Le début de période de blocage est défini par l'utilisateur.

Ressources :

- Vous trouverez sur le site du cours une archive contenant des images des cellules.

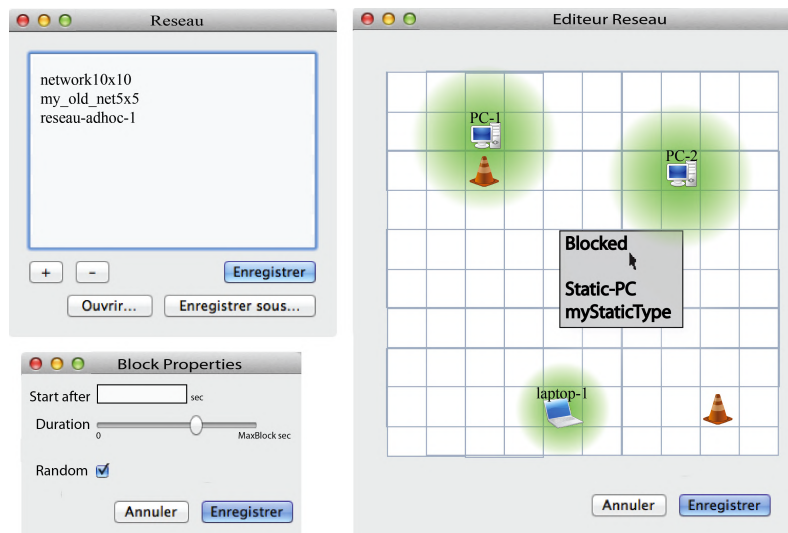


FIGURE 2 – exemple de fenêtres pour ouvrir, pour ajouter le types des cellules et pour éditer les paramètres de blocage (version initiale)

Objectif : Permettre la création d'un réseau de taille $N*N$, des nœuds statiques, et des cellules bloquées.

- un réseau est représenté par une grille de $N*N$ cellules,
- les cellules sont soit des cellules normales, soit bloquées.
- chaque cellule a une représentation graphique,
- l'utilisateur peut placer des nœuds statiques dans les cellules non bloquées,
- les cellules bloquées :
 - ne permettent pas d'y placer des nœuds pour une période aléatoire (entre $0.2 - MaxBloc$ sec), ou définie par l'utilisateur. Vous pouvez définir un temps de blocage maximal *Max-Bloc*.
 - l'utilisateur peut définir à partir de quand il faut bloquer la cellule (ex. 2 sec après le début de la simulation). Pendant la simulation, l'image de « blocage » ne sera visible que pendant la période de blocage.

Simulateur

Version initiale à rendre pendant la 3e séance de TP-projet

Le simulateur permet à l'utilisateur de visualiser les positions des nœuds et les déplacements des paquets dans le réseau.

Fonctionnalités souhaitées :

- Chargement de nœuds à partir des types créés et sauvegardés en fichiers dans l'usine,
- Chargement d'un réseau à partir du fichier créé dans l'éditeur de réseau,
- Visualisation du réseau (nœuds et transmission)

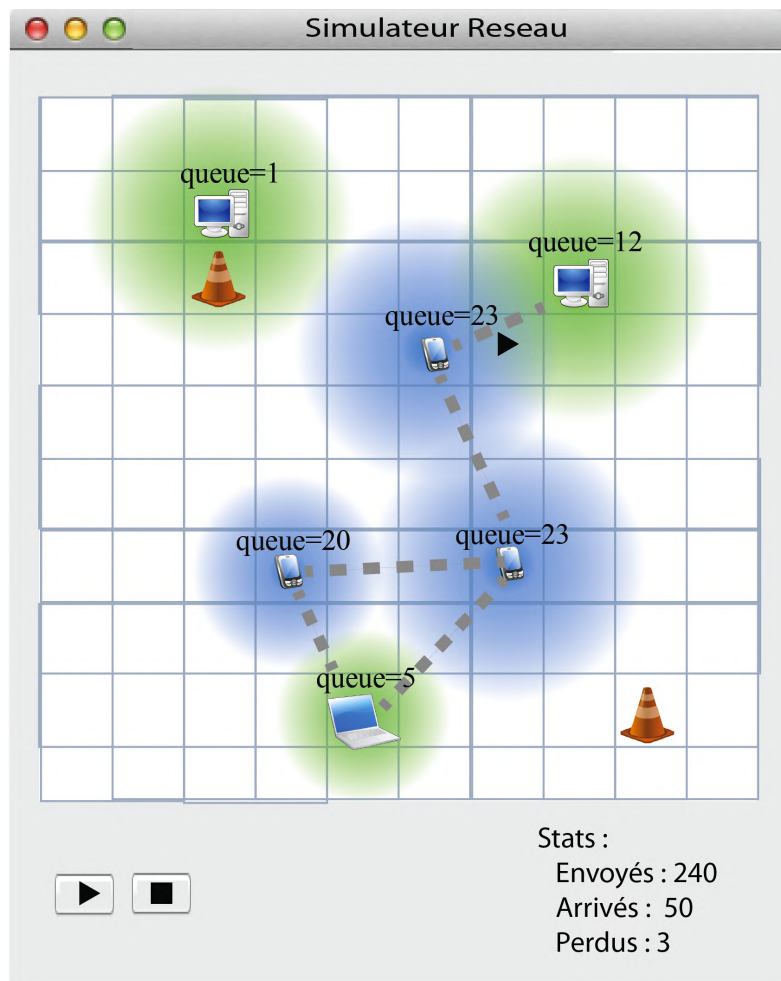


FIGURE 3 – exemple d'écran du simulateur de réseau (version initiale)

Objectif : Simuler un réseau de taille $N*N$ comprenant des nœuds statiques (prédéfinis dans l'éditeur), et des nœuds mobiles qui entrent et partent du réseau.

- Le simulateur contient une liste de tous les nœuds « actifs » et peut calculer leurs distances les uns des autres.
- Le simulateur « ajoute » un nouveau nœud mobile avec un taux de *CreationNœud* par sec (défini par l'utilisateur avant la simulation). On peut avoir un nombre maximal des nœuds mobiles (*MaxNoeudCount*).
- Pendant la création d'un nœud, le type de nœud ajouté est choisi aléatoirement parmi les types de nœuds mobiles définis dans l'usine, ainsi que sa position dans une cellule non bloquée.

- Chaque nœud mobile a une durée de vie dans la simulation, définie de façon aléatoire pendant sa création (entre 0.5 et *MaxNœudVie* ; *MaxNœudVie* est défini par l'utilisateur avant la simulation).
- Chaque nœud a une liste des ses voisins (nœuds à une distance inférieure de sa portée, en termes de distance euclidienne).
- Chaque nœud produit des données (paquets) à envoyer à un autre nœud du réseau avec sa vitesse de production des données. Le nœud « destination » du paquet est choisi de façon aléatoire par tous les nœuds qui sont « en ligne » quand le paquet est créé.
- Un paquet créé ou reçu par un nœud (plus tard) est mis dans la queue de transmission du nœud.
- Chaque nœud qui a des paquets dans sa queue, transmet un paquet à la fois à tous ses voisins. Les voisins qui reçoivent ce paquet le mettent dans leur queue de transmission, sauf s'ils sont le nœud destination. Quand un paquet arrive à sa destination il est considéré comme arrivé. Naturellement, il peut avoir des versions de ce paquet encore transmises dans le réseau.
- Pour assurer que le réseau ne devienne pas chargé avec de vieux paquets (qui peuvent être déjà arrivés), on va ajouter une durée de vie aux paquets *PaquetVie*. Si un nœud reçoit un paquet qui est plus vieux que *PaquetVie* il ne le retransmet pas et l'enlève de sa queue de transmission. Si un paquet disparaît complètement du réseau sans arriver à sa destination, il est caractérisé comme « perdu »
- Le simulateur donne des statistiques en temps réel sur les paquets créés, perdus, et arrivés à leur destination. Faites attention à ne compter les paquets arrivés qu'une seule fois (plusieurs copies peuvent arriver au destinataire par des directions différentes).
- Le simulateur montre des paquets envoyés, c.à.d. il a une façon de représenter visuellement les paquets qui sont arrivés et transmis par un nœud. Cette visualisation peut être une animation, ou texte qui affiche le nombre de paquets parties/arrives pour chaque nœud à coté de son image.

Remarques :

- Chaque noeud est géré par son propre thread, qui commence quand le noeud est ajouté (mobile) ou au début de la simulation (statique).
- La simulation termine, au mise en pause et reprise si l'utilisateur l'indique avec des boutons.
- Chaque nœud a une stratégie de communication, mais pour le moment, tous les nœuds envoient les paquets à tous leurs voisins (« flooding »), un paquet à la fois. De plus, nos nœuds mobiles ne se déplacent pas encore.

Dans un premier temps penser à coder et tester votre programme avec des nœuds et cellules bloqués qui ont une durée de vie fixe, la même vitesse de production, etc avant de traiter toutes les fonctions mentionnées ici.

Editeur de réseau

Version avancée à rendre pendant la dernière séance de TP-projet

à venir

Usine

Version avancée à rendre pendant la dernière séance de TP-projet

à venir

Simulateur

Version avancée à rendre pendant la dernière séance de TP-projet

à venir

Conseils pour réaliser le projet

Des parties des programmes demandés sont relativement indépendantes, vous pouvez donc vous partager le travail (mais ce n'est pas une obligation). Voici quelques conseils :

- 1. commencez simplement** : inutile de vouloir directement faire les versions avancées de chaque programme. Ces versions seront beaucoup plus simples à envisager lorsque vous aurez terminé la première partie du projet.
- 2. décomposez vos problèmes** : concentrez vous sur un sujet précis plutôt que de tenter de réaliser les trois programmes en parallèle.
- 3. faites le lien avec ce que vous savez déjà faire** : les notions et programmes que vous avez fait en TP sont suffisantes pour construire vos premières versions de programme.

Remise des documents et du programme

Pour la remise des documents, vous devez remettre en mains propres un **document papier** lors de la séance de cours ayant lieu à la date donnée. En *haut de la première page*, n'oubliez pas de mettre vos **noms, prénoms**, la mention **“Développement Logiciel : projet”** et l'intitulé du document (ex. manuel d'utilisation). Inutile de prendre une page entière pour ces informations. **Agrafez** (ou reliez) vos documents.

Pour la remise de votre programme, on vous demande d'envoyer aux **deux** enseignants (chargé de cours et votre chargé de TP), une archive au format **zip** ou **tgz** de vos sources. Vous devez envoyer cette archive lors de la dernière séance.

Attention : Testez votre archive. Vous pourrez consulter le site du cours pour voir si nous avons reçu (ou non) votre archive.

Contacts pour l'envoi des documents et des sources de votre projet

Chargés de TP :

Mounir Assaf,
Roza Lemdani,
Oleg Lodygensky

Chargée de cours :

Anastasia Bezerianos

Merci d'envoyer tous les documents à votre chargé de TP uniquement, en mettant en copie votre chargée de cours.

=> n'oubliez pas de mettre “[DevLog]” au début du sujet de votre mail (pour éviter d'être classer en spam). Par exemple : “[DevLog] Question sur ...”. Nous vous pourrions vous donner des indications générales mais en aucun cas résoudre des problèmes de programmation.