Rapport PlagueINT

Modélisation de la propagation des épidémies

CHERRE Romain COROLLER Stevan
PAMART Pierrick PIPEREAU Yohan

Encadrant: Mr. Vincent Gauthier

22 mai 2017

Table des matières

Pré-Rapport

Analyse des besoins

Fonction du produit

- Mode de visualisation (écoulement du temps)
- Modélisation mondiale avec cellules de la taille d'un pays
- Possibilité d'exporter le résultat dans un fichier lisible
- Voies de transports prises en compte
- Possibilité d'ajouter des événements (blocage d'aéroports, gare, etc..) au début de la modélisation

Contraintes techniques

- Utiliser Java8 avec Eclipse et éventuellement d'autres languages si nécessaires
- Possibilité d'exécution en mode terminal puis graphique
- Portabilité Windows, Linux, MAC OS (gérée nativement par Java)

Critères d'acceptabilité et de réception

- Application performante avec un temps d'exécution raisonnable
- L'interface de l'application doit être conforme à la maquette suivante :

—— **Simulation de propagation de maladies** ——
(1) Lancer la simulation
(2) Paramètres de simulation
(3) Quitter la simulation

—— **Paramètres de simulation** ——
(1) Quitter les options sans sauvegarder
(2) Date de début et durée de la simulation
(3) Pays infectés et nombre d'infectés de départ
(4) Choisir les constantes de propagation
(5) Gérer les évènements
(6) Choisir une maladie pré-enregistrée
(7) Quitter et sauvegarder les paramètres

—— **Evènements** ——
(1) Créer un évènement
(2) Voir les évènements
(3) Supprimer un évènement
(4) Revenir aux paramètres

Extensions

- Interface - graphique
- Informations sur les celulles (petits graphiques, etc...)
- Modification de l'environnement (hygiène, température, etc...)

Juridique

Spécification fonctionnelle générale

Fonction du produit

Pour l'écoulement du temps, nous avons choisi de discrétiser le temps.
Pour l'importation des données et leur traitement, Python est fortement envisagé en tant qu'outil plus performant que Java à l'aide de certaines bibliothèques précodées. Pour l'exportation des résultats dans un fichier lisible, on exporterait les données dans un fichier csv en utilisant des fonctionnalités de lecture/écriture de fichier.
Pour modéliser les voies de transports, on utiliserait un seul graphe avec comme noeuds du graphe les pays et sur les branches, le nombre de passagers par jour.

Critères d'acceptabilité et de réception

Pour la résolution des équations différentielles, on utiliserait dans un premier temps une méthode d'Euler. Dans un second temps, on implémenterait une méthode de Runge Kutta qui nous permettrait de gagner en précision.
Pour l'interface utilisateur, on permettrait à l'utilisateur de définir l'échelle de temps afin de gérer la rapidité du programme. L'utilisateur pourrait également choisir les pays de lancement de la maladie. Il écrirait le nom du pays et on vérifierait si le nom correspond au nom d'un pays présent dans la liste d'une variable pays.
Enfin pour les coefficients de propagation des différentes maladies, il y a le mode manuel ou l'utilisateur saisit les coefficients à la main et lance ensuite notre programme de modélisation. Mais il y a aussi le mode d'utilisation où l'utilisateur rentre la maladie et où le programme va chercher dans des données que l'on a générées précédemment à partir de statistiques mondiales sur les maladies.

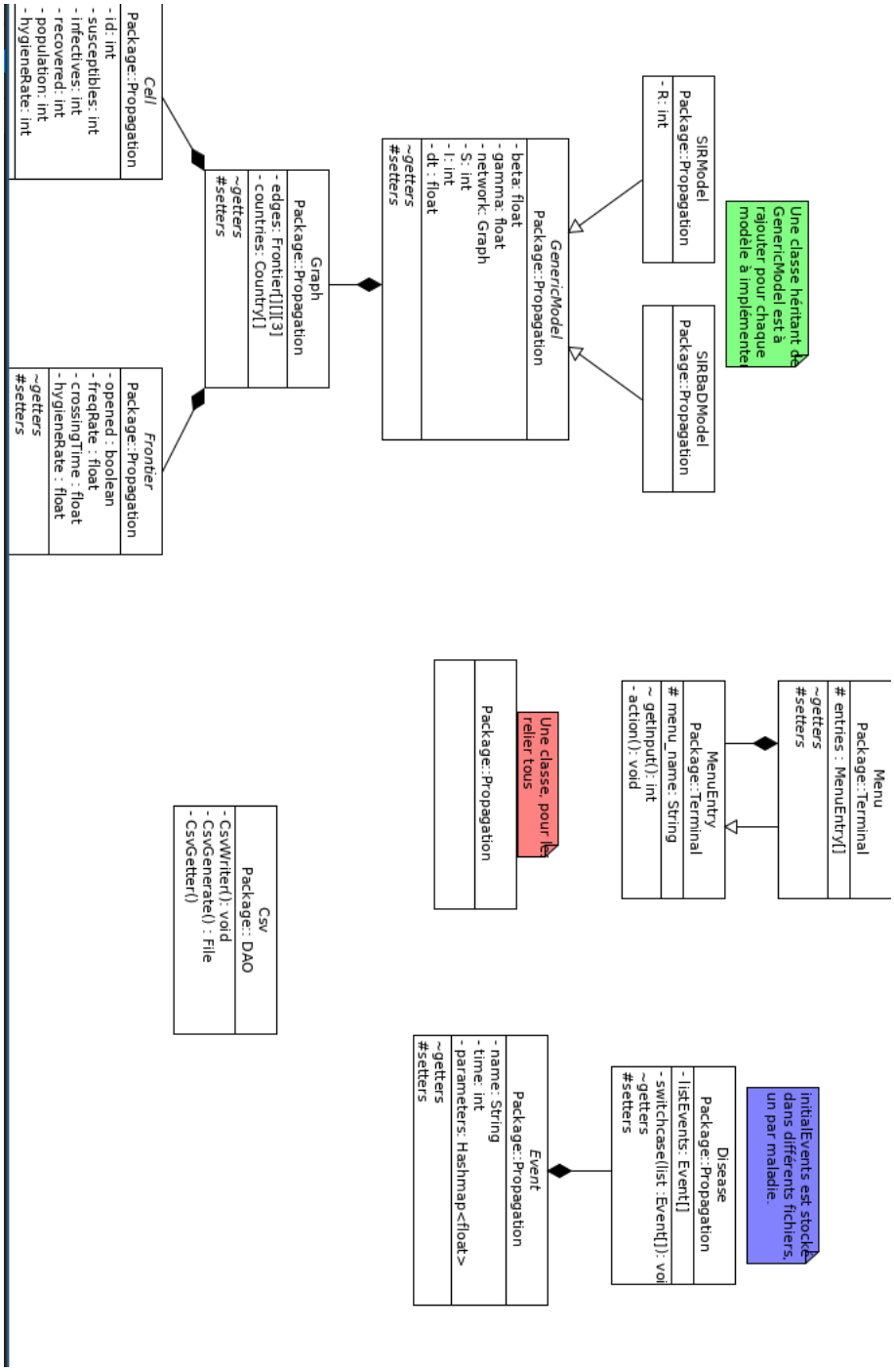
Extensions

Pour l'interface graphique qui permettra d'afficher une carte du monde ainsi que de tracer des graphiques relatifs aux données de la celulle et leur évolution dans le temps, nous utiliserons la librairie (toolkit) graphique JavaFX.
Pour la modification de l'environnement (hygiène, température, etc...), on modifie directement les coefficients de propagation de la maladie dans la cellule.

Regroupement modulaire des fonctionnalités

- Visualisation
  - Terminal
  - Graphique
  - Exportation en CSV
- Évènements
  - Blocage de lieux de transports
  - Blocage des frontières
- Statistiques
  - Par pays : évolutions du nombre d'infectés, ...
  - Générales
- Calcul des évolutions temporelles

Description du flux des données entre les modules



Le diagramme UML ci-dessus ne présente pas les relations entre toutes les classes car nous comptons voir comment regrouper certaines classes dans des phases ultérieures de notre développement.

Conception détaillée

Guide pour le développeur

L'aborescence des paquets est la suivante :

DAO : Data Access Object : Paquet contenant les classes relatives aux données. Dans notre projet les données sont stockées dans fichier .txt avec séparateurs.

- Main : Contient la classe Context.java qui est itérée tous les dt et qui lance la création des objets et permet le déroulement du programme.
- propagation : Ce paquet contient les éléments relatifs aux pays, frontières, les modèles épidémiques et les événements. Nous voulions modéliser trois types de frontières : Air, Land et Maritime. Les appels pour les extractions de données du CSV se font dans la classe Graph.java.
- service : Ce paquet contient une première version de résolution des équations différentielles avant l'implémentation de utils.
- terminal : Ce paquet contient l'ensemble des fonctions donnant le terminal utilisateur.
- utils : Ce paquet contient les outils de résolution des équations différentielles en proposant une implémentation de la méthode d'Euler.

Bilan

Comparaison entre l'objectif et la réalisation

Les contraintes techniques ont bien été respectées : utilisation de Java8 et Eclipse, possibilité d'exécution dans un terminal cependant la portatibilité Windows n'est pas garantie car l'arborescence utilise au lieu de /. Le programme fonctionne correctement sous Linux et MacOS. Nous n'avons pas eu le temps de travailler sur l'interface graphique, considéré comme une extension du programme. Il possible d'obtenir des graphes sur l'état des pays en récupérant les fichiers .csv de données et en utilisant un tableur excel. Concernant les critères d'acceptabilité et de réception, l'interface utilisateur ne correspond pas complètement à la maquette initiale : il ressemble à la maquette mais l'ordre des menus n'est pas le même, la version finale nous paraîssait plus logique et intuitive. La méthode d'Euler est implémentée, mais la méthode de Runge-Kutta, plus précise, ne l'est pas encore. Contrairement à ce qui était prévu, nous n'avons pas utilisé python pour importer et traiter les données dont nous nous servons. En effet, nous sommes tous habitués à utiliser bash en tant qu'outil natif en administraiton système. Enfin, à l'exception des différents modèles de propagation des maladies (notamment le modèle SIR avec naissances et morts), tous les choix possibles qui étaient prévus pour la modification des paramètres de propagation, et qui n'étaient pas considérés comme des extensions (comme le taux d'hygiène des pays et frontières) sont implémentés. La fonction d'importation des données n'est pas fonctionnelle. En particulier, nous avons essayé de trouver et parser des données réelles sans succès. Nous avons alors décidé de créer un monde fictif linuxmap. Ci-dessous la carte linuxmap :

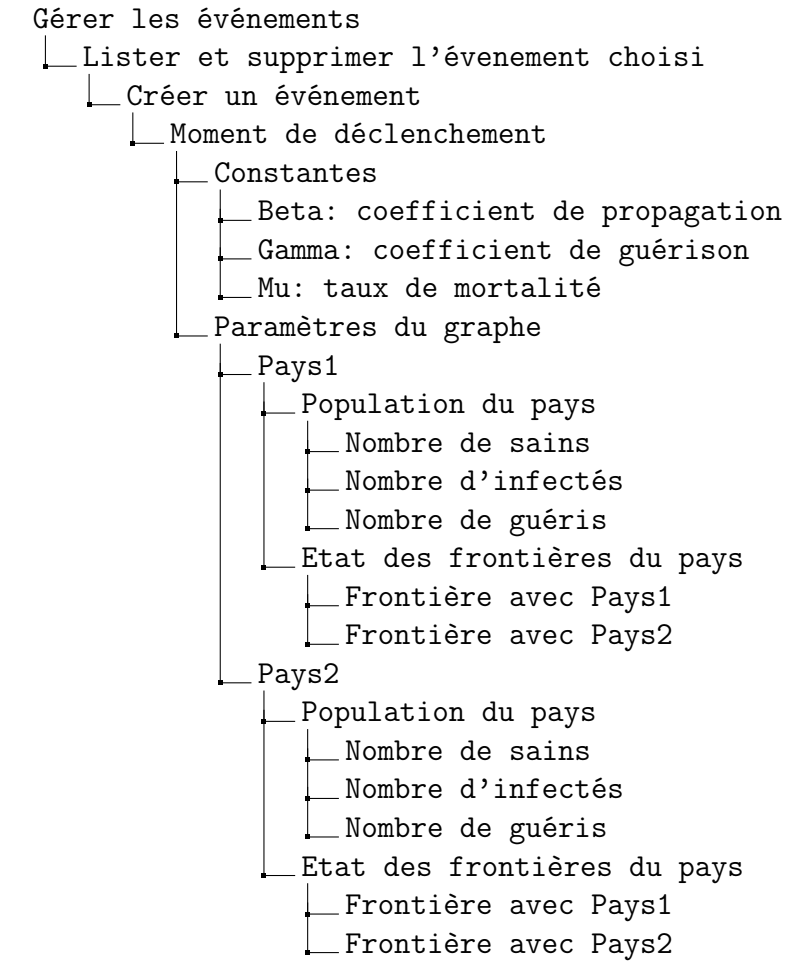

THE GREAT COMPILE
CENTOSIA
THE CHAKRA BAY
THE HIGHLANDS OF GENTOO
SABAYON SEA
NOVELL SEA
THE INDEPENDIENT REPUBLIC OF FEDORA
THE RED HAT COMONWEALTH
THE SCIENTIFIC ARCHIPELAGO
THE OPENSUSE KINGDOM
CAPE SUDO
NEW MAGEIA REPUBLIC
MOUNT CHROOT
ANDROID SEA
PCLINUX OS SEA
TIZEN ISLAND
THE ROGUE STATE OF MINT
THE UBUNTU BASED EMIRATES
KUBUNTUSTAN
GPL SEA
RPM STRAITS
CINNAMON FORTRESS
MATE FORTRESS
THE ELEMENTARY STATE
THE LINUS BAY
GNOME 3 HILLS
FIREFOX OS FEDERATION
KINGDOM OF SLACKWARE
POPULAR REPUBLIC OF UBUNTU
XUBUNTIA
FOSS CHANNEL
THE GNU OCEAN
UNITY FORTRESS
NOOB DESERY
CANONICAL SEA
THE ISLE OF MARK
THE DEBIAN REPUBLIC
THE STALLMAN SEA
LUBUNTIA

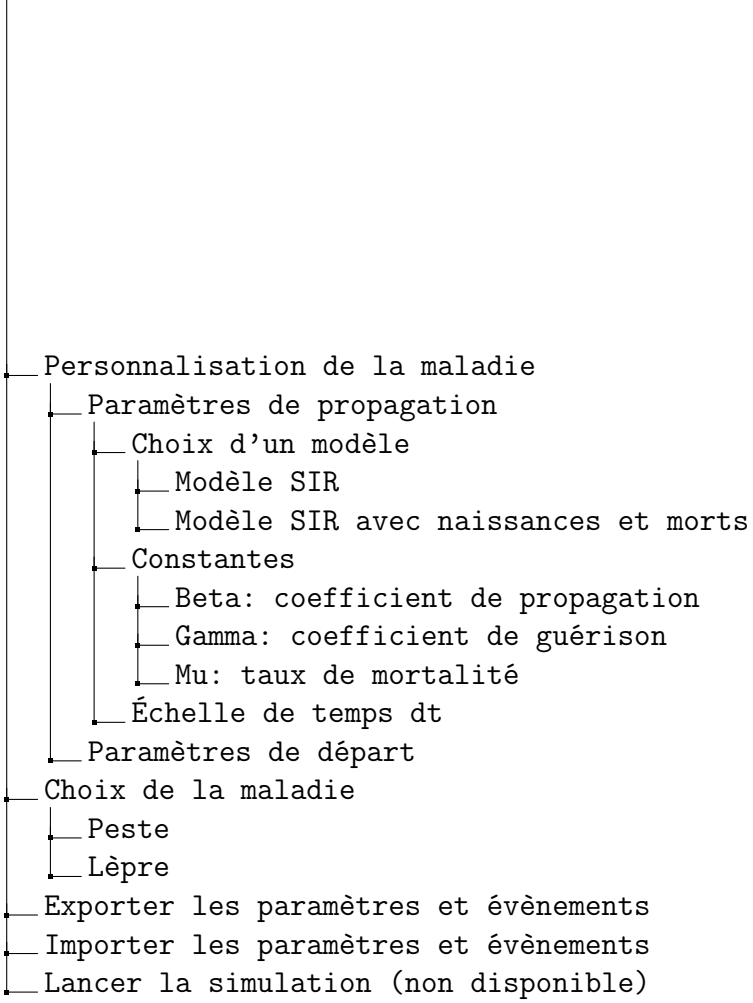
Manuel utilisateur

Guide d'utilisation du programme PlagueINT

Aborescence des Menus

L'interaction entre l'utilisateur et le programme se fait par l'utilisation d'un terminal divisés en menus.





Le choix d'une maladie permet de charger les données pré-enregistrées (constantes, population de départ, évènements...) d'une maladie connue. Nous n'avons pas eu le temps de générer ces données, mais la fonctionnalité elle-même est implémentée. La personnalisation de la maladie permet de modifier les données de départ de la modélisation (que l'on ait choisi une maladie précédemment ou non). Ainsi on peut modifier l'état de départ des frontières (ouverture/fermeture et flux de population), la population de départ de tous les pays (répartie entre les sains, les infectés et les guéris) et les paramètres de propagation de la maladie (coefficients Beta et Gamma, Mu n'étant pas implémenté). Les événements sont des phénomènes modifiant les données du problème à un instant t. Ils permettent de modifier les mêmes paramètres que le menu de personnalisation de la maladie, mais à un instant donné non nul. Exporter les paramètres et évènements permet de sauvegarder les modifications apportées aux paramètres de la simulation, et importer les paramètres et évènements permet de charger les sauvegardes précédentes.

Les test unitaires peuvent être exécutés dans DAO.CsvTest.