Je ne détruits plus les agents.

Si un ou plusieurs soilCells sont modifiés (nouveaux villes, marchés ou routes) alors il faut refaire lendPlotsList, cityList et marketList.

Les répertoires que j’utilise pour centennale sont tous nommé CENTENAL

Dans inspecteur centenal : updateSCMandVL fait quoi ? JLF

updateSCMandVL est dans RasterGraphManager, c’est pour modifier l’affinité des soilCell et des des cellules de valueLayer à la création d’une nouvelle route, ville etc. PAM

J’ai supprimé data\_csv/SenCentenal (pour l'instant, on n’a qu’à tout mettre dans data\_csv (il faudrait d’ailleurs mettre tous les répertoires data dans un seul mais je crains que cela ne fasse beaucoup de changements à faire). JLF

Et pour data\_rater, est ce qu’on va mettre tous les répertoires data… dans data ? PAM

*currentMonth commence par 0 au lieux de 1 si on le change il aussi changer… s*

Dans C\_StepVariousProcedure.manageUniverse()

// UNIVERSE MANAGEMENT EACH NEW CURRENT MONTH/YEAR

groundManager.resetCellsColor(); //

valueLayer est tout simplement pour l’affichage (Pour l’instant)

C\_UnweightedGraph

C\_RasterGraphManager

Je regarde le fichier chrono, à chaque fois que je voix **bioClimate** dans le champ **EVENT**, alors je regarde sa valeur dans le champ **VALUE** pour renseigner le **fichier raster** avec les conversions suivantes :

Les valeurs de **raster value** seront les valeurs des **affinités des soilCells.**

Ce tableau de conversion est stocké dans I\_sim\_constants comme suit :

// 0 1 2 3 4 5 6 7 8 9 : Correspondence ou VALUE :

**public** **static** **final** **int** [] *RASTER\_BIO\_CLIMATES\_VALUES* = {0, 7, 5, 0, 0, 0, 0, 0, 6, 5 };

Donc je lits dans fichier chrono

Donc pour gérer les couleurs dans la classe C\_Style2d\_terrain, je veux connaitre suivant ces couleurs et la ou les cartes que nous devons utliser, la valeur du «**Raster value**» et si possible la **correspondance.**

Pour la première carte j’ai déjà mis la **correspondance**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **EVENT** | **VALUE** | **raster value** | **Couleur** | **RGB pour C\_Style2d\_terrain** | **My Raster Value(provisoire)** | **correspondance** |  |
| bioClimate | 0 | 1 |  | 29, 29, 165 | 14 | **Ocean and Abroad** |  |
| bioClimate | 1 | 7 |  | **248, 25, 252** | **7** | **Niayes : 1** |
| bioClimate | 2 | 5 |  | **255 , 255 , 0** | **4** | **Saloum : 2** |
| bioClimate | 3 | 0 |  | **51, 255, 255** | **0** | **Sine : 3** |
| bioClimate | 4 | 0 |  | **139, 20, 147** | **1** | **Grande côte et Delta : 4** |
| bioClimate | 5 | 0 |  | **153, 0, 0** | **2** | **Ferlo : 5** |
| bioClimate | 6 | 0 |  | **102 , 204 ,204** | **3** | **Soudanien : 6** |
| bioClimate | 7 | 0 |  | **51 ,102 , 255** | **9** | **Terres neuves et zone cotonnière : 7** |
| bioClimate | 8 | 6 |  | **165, 255, 255** | **6** | **Basse-Casamance : 8** |
| bioClimate | 9 | 5 |  | **17 ,178 ,151** | **5** | **Haute Casamance et Sénégal Oriental : 9** |

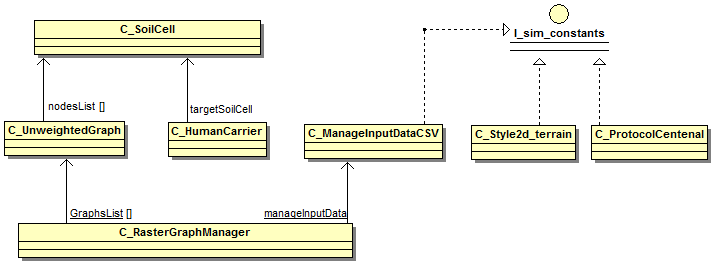
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **EVENT** | **VALUE** | **raster value** | **Couleur** | **RGB pour C\_Style2d\_terrain** | **My Raster Value(provisoire)** | **correspondance** |  |
| bioClimate | 0 | 1 | ou | 249, 248, 239 ou **29,29,165** | 14 | **Ocean and obroad** |  |
| bioClimate | 1 | 7 |  | **51, 204, 102** | **7** | **Niayes : 1** |
| bioClimate | 2 | 5 |  | **211, 238, 207** | **4** | **Saloum : 2** |
| bioClimate | 3 | 0 |  | **242, 226, 9** | **0** | **Sine : 3** |
| bioClimate | 4 | 0 |  | **212, 205, 205** | **1** | **Grande côte et Delta : 4** |
| bioClimate | 5 | 0 |  | **255, 204, 153** | **2** | **Ferlo : 5** |
| bioClimate | 6 | 0 |  | **255, 255, 0** | **3** | **Soudanien : 6** |
| bioClimate | 7 | 0 |  | **237, 180, 23** | **9** | **Terres neuves et zone cotonnière : 7** |
| bioClimate | 8 | 6 |  | **17, 178, 151** | **6** | **Basse-Casamance : 8** |
| bioClimate | 9 | 5 |  | **197, 216, 21** | **5** | **Haute Casamance et Sénégal Oriental : 9** |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Carte** | **Couleur** | **R** | **G** | **B** | **Raster value** | **correspondance** | **Raster value 2** | **Autres cartes** |
|  |  | **249** | **248** | **239** | **14** | **Arrière-plan** | 0 |  |
|  |  | **255** | **255** | **204** | 7 | **Grande Côte** | 1 |  |
|  | **255** | **255** | **0** | **3** | **Ferlo** | 2 |
|  | **255** | **204** | **153** | **2** | **Sahélien** | 3 |
|  | **237** | **180** | **23** | **9** | **Boundou** | 4 |
|  | **197** | **216** | **21** | 5 | **Fouladou** | 5 |
|  | **17** | **178** | **151** | 6 | **Basse-Casamance** | 6 |
|  | **242** | **226** | **9** | **0** | **Saloum** | **7** |
|  | **139** | **20** | **147** | **1** |  | **8** |
|  |  | **248** | **25** | **252** | **4** |  | **9** |
|  |  | **29** | **29** | **165** |  |  |  |
|  | **153** | **0** | **0** |  |  |  |
|  |  |  |  |  |  |  |
|  | **139** | **20** | **147** |  |  |  |
|  | **51** | **153** | **153** |  |  |  |
|  | **102** | **204** | **204** |  |  |  |
|  | **51** | **255** | **255** |  |  |  |
|  | **201** | **242** | **238** |  |  |  |
|  | **255** | **255** | **0** |  |  |  |
|  |  | **153** | **153** | **153** |  |  |  |
|  | **204** | **204** | **204** |  |  |  |
|  | **222** | **224** | **224** |  |  |  |
|  | **249** | **248** | **239** |  |  |  |
|  |  | **0** | **153** | **0** |  |  |  |
|  | **255** | **0** | **0** |  |  |  |
|  | **51** | **102** | **255** |  |  |  |
|  | **102** | **204** | **204** |  |  |  |
|  | **153** | **0** | **0** |  |  |  |
|  | **249** | **248** | **239** |  |  |  |
|  |  | **103** | **200** | **255** |  |  |  |  |

**Pour les valeurs du fichier raster et les couleurs dans C\_Style2d\_terrain :**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **EVENT** | **VALUE** | **raster value** | **Couleur** | **RGB pour C\_Style2d\_terrain** | **My Raster Value(provisoire)** | **correspondance** | **Carte et légende** |
| bioClimate | 0 | 1 |  | 29, 29, 165 | 8 | **Ocean and obroad** |  |
| bioClimate | 1 | 7 |  | **248, 25, 252** | **7** | **Niayes : 1** |
| bioClimate | 2 | 5 |  | **255 , 255 , 0** | **4** | **Saloum : 2** |
| bioClimate | 3 | 0 |  | **51, 255, 255** | **0** | **Sine : 3** |
| bioClimate | 4 | 0 |  | **139, 20, 147** | **1** | **Grande côte et Delta : 4** |
| bioClimate | 5 | 0 |  | **153, 0, 0** | **2** | **Ferlo : 5** |
| bioClimate | 6 | 0 |  | **102 , 204 ,204** | **3** | **Soudanien : 6** |
| bioClimate | 7 | 0 |  | **51 ,102 , 255** | **9** | **Terres neuves et zone cotonnière : 7** |
| bioClimate | 8 | 6 |  | **165, 255, 255** | **6** | **Basse-Casamance : 8** |
| bioClimate | 9 | 5 |  | **17 ,178 ,151** | **5** | **Haute Casamance et Sénégal Oriental : 9** |

**Voici les classes que j’ai toucher**



Maintenant je crois que je peux gérer plusieurs graphes, mais j’ai pas encore bien tester. a

On peut utiliser d’autres couleurs aussi en utilisant le tableau suivant. Et je crois qu’il me sera facile de remplir la colonne **Raster value** en me basant sur le premier tableau.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Carte** | **Couleur** | **R** | **G** | **B** | **Raster value** | **correspondance** | **Raster value 2** | **Autres cartes** |
|  |  | **255** | **255** | **204** |  | **Grande Côte** |  |  |
|  | **255** | **255** | **0** |  | **Ferlo** |  |
|  | **255** | **204** | **153** |  | **Sahélien** |  |
|  | **237** | **180** | **23** |  | **Boundou** |  |
|  | **197** | **216** | **21** |  | **Fouladou** |  |
|  | **17** | **178** | **151** |  | **Basse-Casamance** |  |
|  | **242** | **226** | **9** |  | **Saloum** |  |
|  | **249** | **248** | **239** |  | **Arrière-plan** |  |
|  |  | **29** | **29** | **165** |  |  |  |
|  | **153** | **0** | **0** |  |  |  |
|  | **248** | **25** | **252** |  |  |  |
|  | **139** | **20** | **147** |  |  |  |
|  | **51** | **153** | **153** |  |  |  |
|  | **102** | **204** | **204** |  |  |  |
|  | **51** | **255** | **255** |  |  |  |
|  | **201** | **242** | **238** |  |  |  |
|  | **255** | **255** | **0** |  |  |  |
|  |  | **153** | **153** | **153** |  |  |  |
|  | **204** | **204** | **204** |  |  |  |
|  | **222** | **224** | **224** |  |  |  |
|  | **249** | **248** | **239** |  |  |  |
|  |  | **0** | **153** | **0** |  |  |  |
|  | **255** | **0** | **0** |  |  |  |
|  | **51** | **102** | **255** |  |  |  |
|  | **102** | **204** | **204** |  |  |  |
|  | **153** | **0** | **0** |  |  |  |
|  | **249** | **248** | **239** |  |  |  |
| Autres couleurs |  | **103** | **200** | **255** |  |  |  |  |

Je crois que valueLayer ne gère que les couleurs dans l’affichage. Si c’est le cas…

**Suggestion sur la gestion des carriers et des groupes de villes dans lesquelles ils peuvent aller (Rejeté)**

Je propose qu’on ajoute dans le centenalEvents.csv une **colonne** permettant de spécifier pour chaque ville, les groupes dans lesquelles elle appartient.

Ou bien au lieu d’une nouvelle colonne, ajouter pour chaque ville, dans la colonne **value**, la liste des numéros des groupes dans lesquelles elle appartient et que les numéros soient séparés par des virgules ou un autre caractère.

On fait la même chose pour les carriers(train et truck) : on met dans la colonne **value** le nombre, son groupe de ville, et si on veut son type pour les trucks(CAMION, CAMIONNETTE, MINIBUS ou VOITURE).

**Ex :**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **X** | **Y** | **DATE** | **CELL\_ID** | **EVENT** | **VALUE** |
| 12 | 15 | 05/01/1910 | 1348 | city | **Banjul,0**  Nom des villes |
| 29 | 60 | 05/01/1910 | 5370 | city | **Dagana,1** |
| 0 | 34 | 05/01/1910 | 3027 | city | **Dakar,1**  Nombre de carriers |
| 18 | 34 | 05/01/1910 | 3045 | city | **Diourbel,2**  Les groupes On peut aussi créer une nouvelle colonne pour les groupes de villes |
| 15 | 26 | 05/01/1910 | 2330 | city | **Foundiougne,3** |
| 14 | 53 | 10/01/1910 | 4732 | train | **1,1** |
| 15 | 53 | 10/01/1910 | 4733 | train | **1,3** |
| 3 | 35 | 01/01/1924 | 3119 | truck | **4,0** |
| 0 | 34 | 01/01/1924 | 3027 | truck | **556,2** |

Comme ça moi je construis une structure de données dans **I\_sim\_constants** pour récupérer tous les groupes par leur numéro

Et dans la classe **HumanCarrier**, je crée un champ **cityGroupNumber** de type **int**, permettant de spécifier pour chaque carrier le groupe de villes dans lesquelles il peut aller.

Et via ce numéro, les carriers(train et truck) peuvent à leur groupe de ville dans **I\_sim\_constants** pour choisir une destination.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

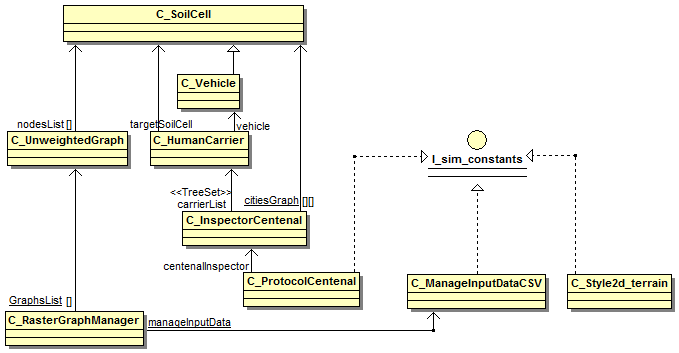
**29/05/2013**

**Couleurs :**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **EVENT** | **VALUE** | **raster value** | **Couleur** | **RGB pour C\_Style2d\_terrain** | **My Raster Value(provisoire)** | **correspondance** |  |
| bioClimate | 0 | 1 | ou | 249, 248, 239 ou **29,29,165** | 14 | **Ocean and obroad** |  |
| bioClimate | 1 | 7 |  | **51, 204, 102** | **7** | **Niayes : 1** |
| bioClimate | 2 | 5 |  | **211, 238, 207** | **4** | **Saloum : 2** |
| bioClimate | 3 | 0 |  | **242, 226, 9** | **0** | **Sine : 3** |
| bioClimate | 4 | 0 |  | **212, 205, 205** | **1** | **Grande côte et Delta : 4** |
| bioClimate | 5 | 0 |  | **255, 204, 153** | **2** | **Ferlo : 5** |
| bioClimate | 6 | 0 |  | **255, 255, 0** | **3** | **Soudanien : 6** |
| bioClimate | 7 | 0 |  | **237, 180, 23** | **9** | **Terres neuves et zone cotonnière : 7** |
| bioClimate | 8 | 6 |  | **17, 178, 151** | **6** | **Basse-Casamance : 8** |
| bioClimate | 9 | 5 |  | **197, 216, 21** | **5** | **Haute Casamance et Sénégal Oriental : 9** |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **EVENT** | **VALUE** | **raster value** | **Couleur** | **RGB pour C\_Style2d\_terrain** | **My Raster Value(provisoire)** | **correspondance** |  |
| bioClimate | 0 | 0 | ou | 249, 248, 239 ou **29,29,165** | 14 | **Ocean and obroad** |  |
| bioClimate | 1 | 10 |  | **51, 204, 102** | **7** | **Niayes : 1** |
| bioClimate | 2 | 11 |  | **211, 238, 207** | **4** | **Saloum : 2** |
| bioClimate | 3 | 5 |  | **242, 226, 9** | **0** | **Sine : 3** |
| bioClimate | 4 | 3 |  | **212, 205, 205** | **1** | **Grande côte et Delta : 4** |
| bioClimate | 5 | 2 |  | **255, 204, 153** | **2** | **Ferlo : 5** |
| bioClimate | 6 | 4 |  | **255, 255, 0** | **3** | **Soudanien : 6** |
| bioClimate | 7 | 6 |  | **237, 180, 23** | **9** | **Terres neuves et zone cotonnière : 7** |
| bioClimate | 8 | 12 |  | **17, 178, 151** | **6** | **Basse-Casamance : 8** |
| bioClimate | 9 | 9 |  | **197, 216, 21** | **5** | **Haute Casamance et Sénégal Oriental : 9** |
| city | city | 13 |  | **255, 0, 0** |  |  |  |
| river | river | 0 |  |  |  |  |  |
| road | road | 1 |  |  |  |  |  |

**Voici les classes que j’ai touchées :**



**Résultats :**

Tick 5

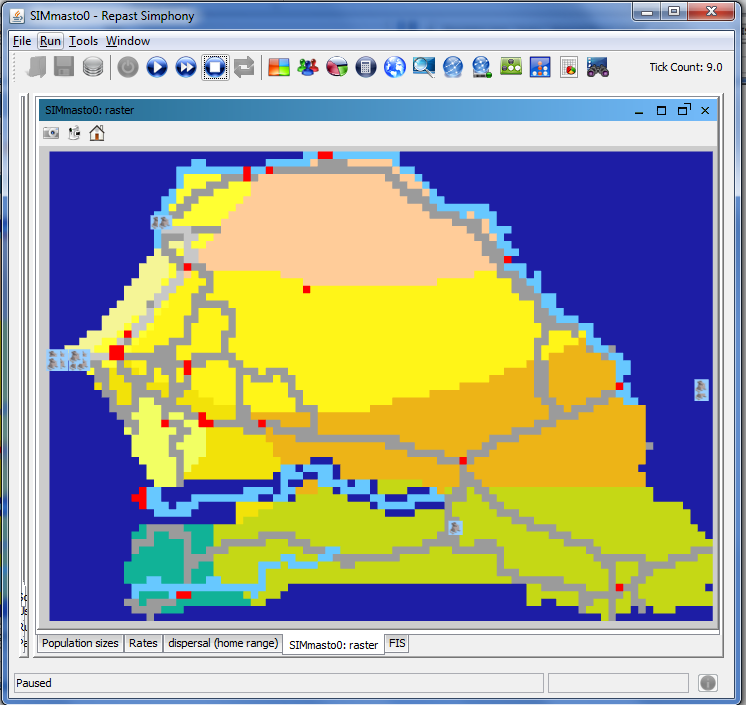
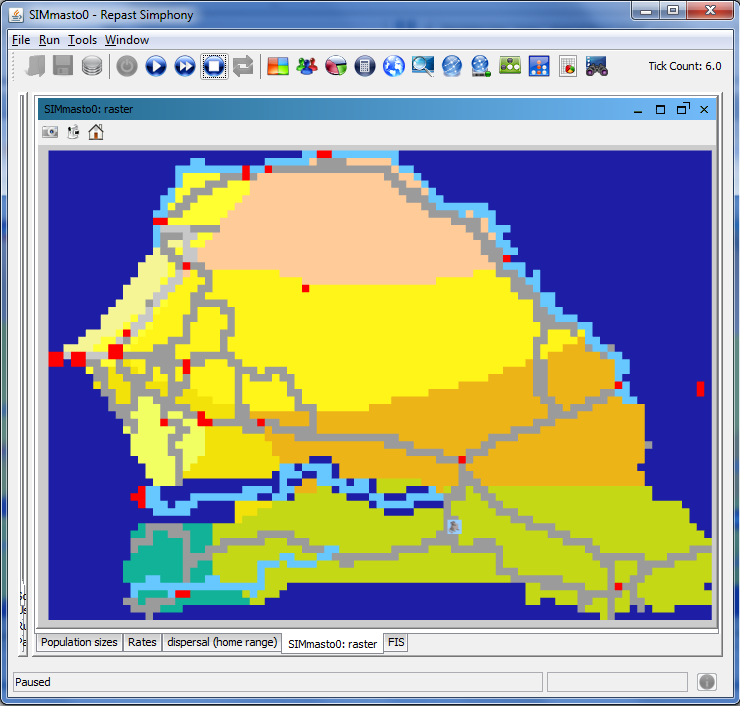
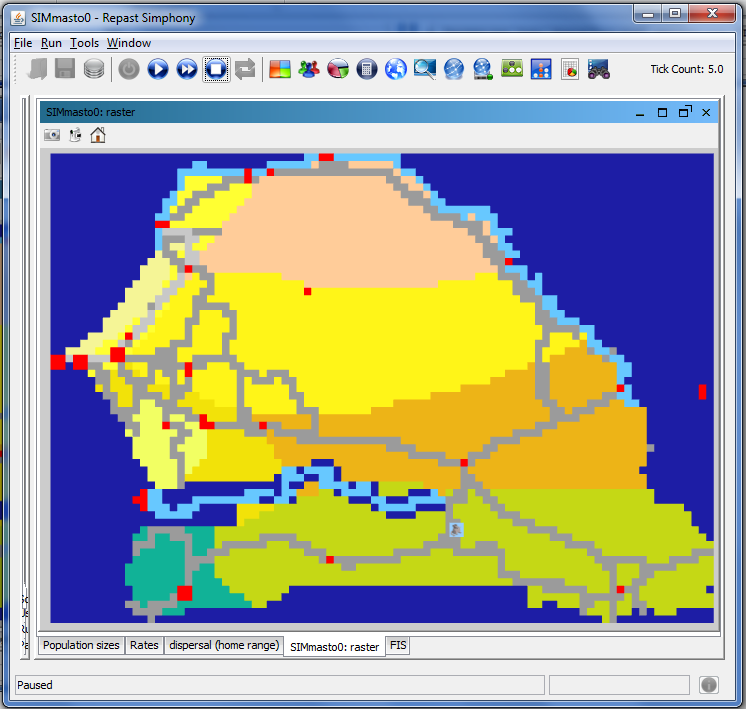
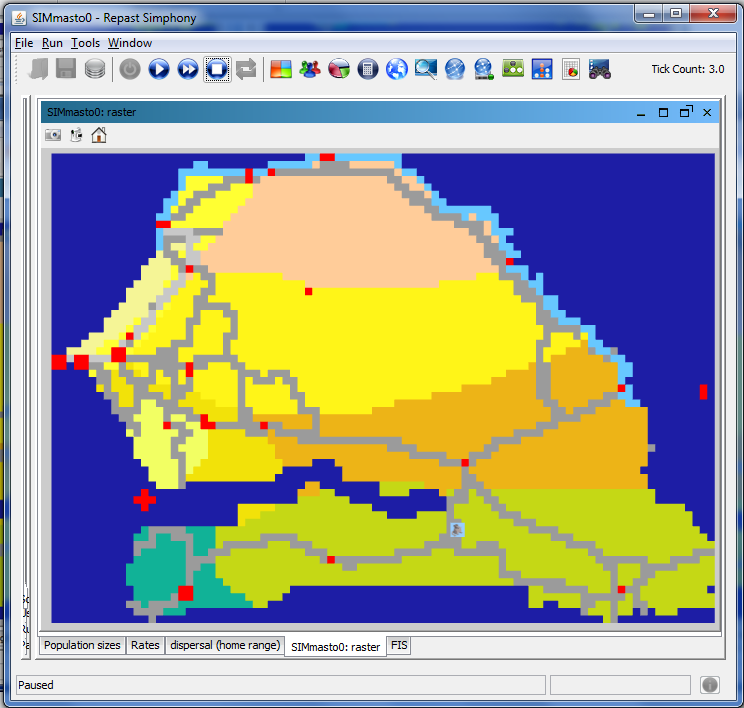
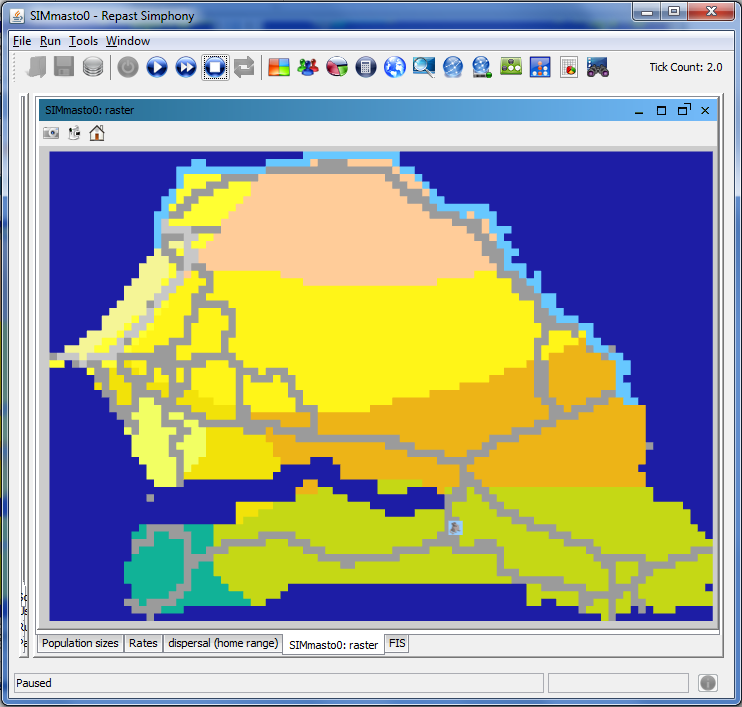
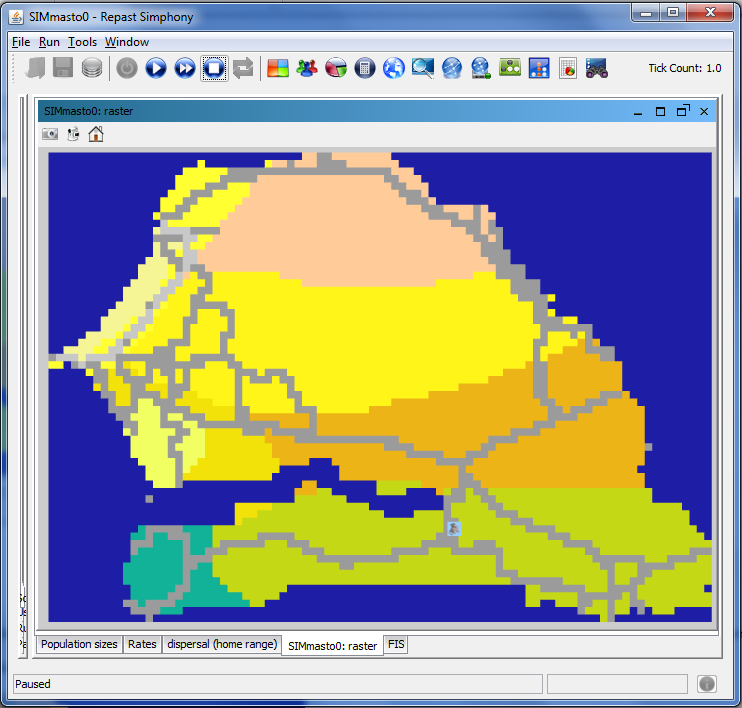
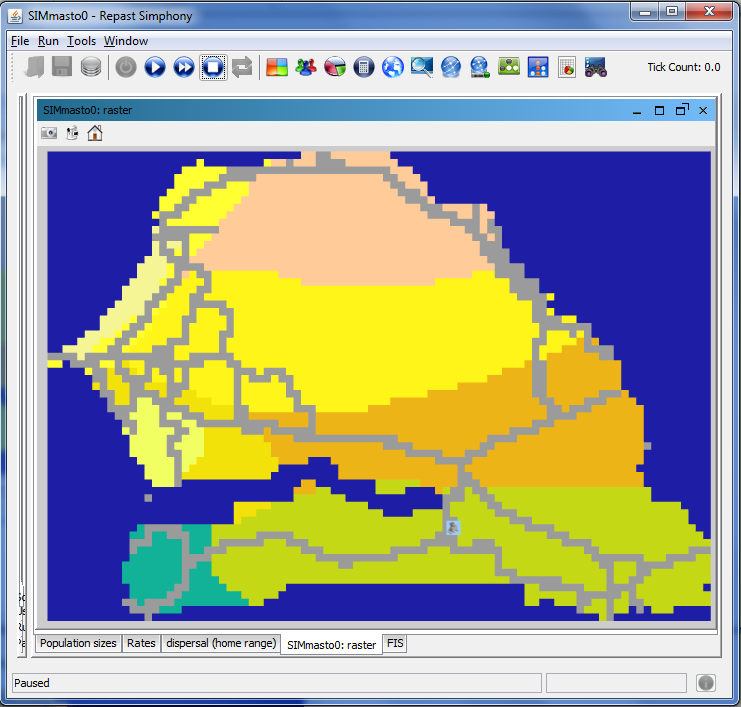
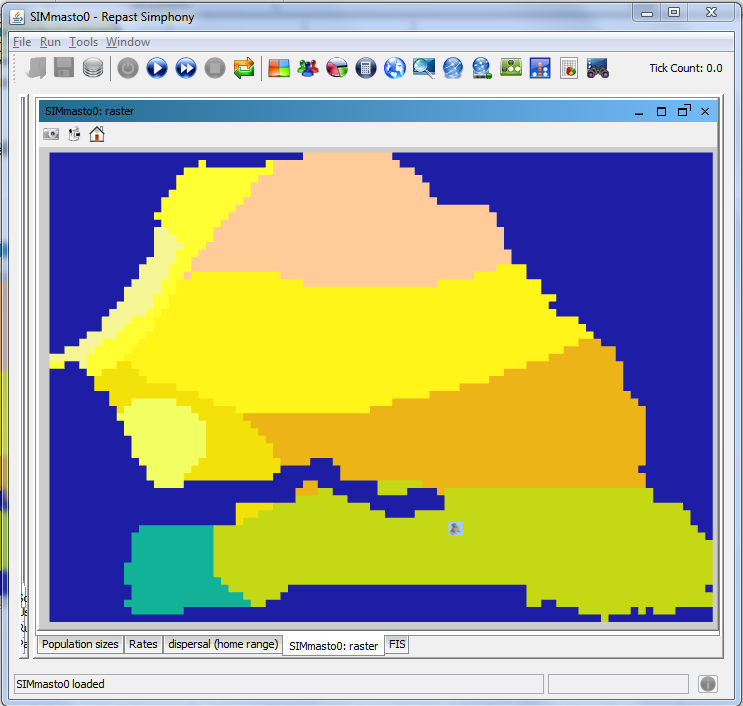
Tick 3

Tick 2

Tick 1

Tick 0

Tick 0



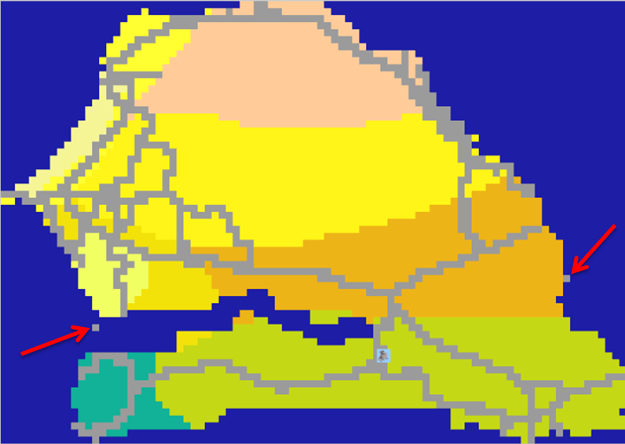
Est-ce qu’un soilCell peut apartenir à 2 landPlot à la fois ?

X Y CellId

13 16 1438 c’est une cellule de type route et de type ville mais qui n’est pas reliée au routes et ça fait qu’un véhicule disparaît et apparaît brusquement (image 1)

80 23 2128 c’est une cellule de type route mais qui n’est pas reliée au routes et ça me paraît anormal

1

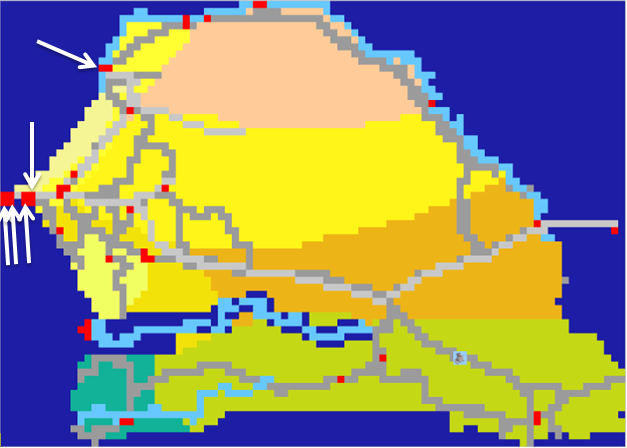
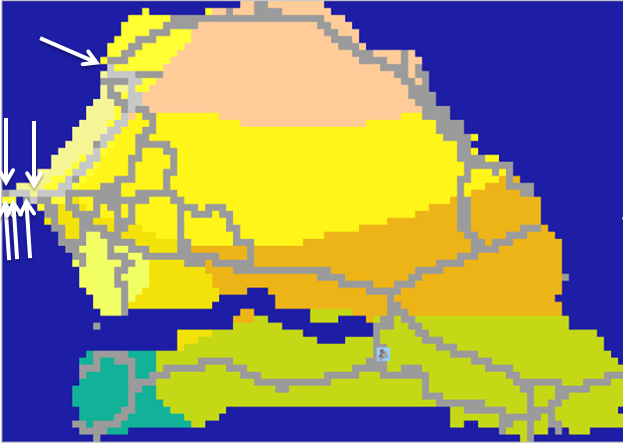


**Cellules de type ville en voisinage d’une route, rail ou river mais n’étant pas de type route, rail ou river :**

**Par rapport aux rails en gris clair :**

3

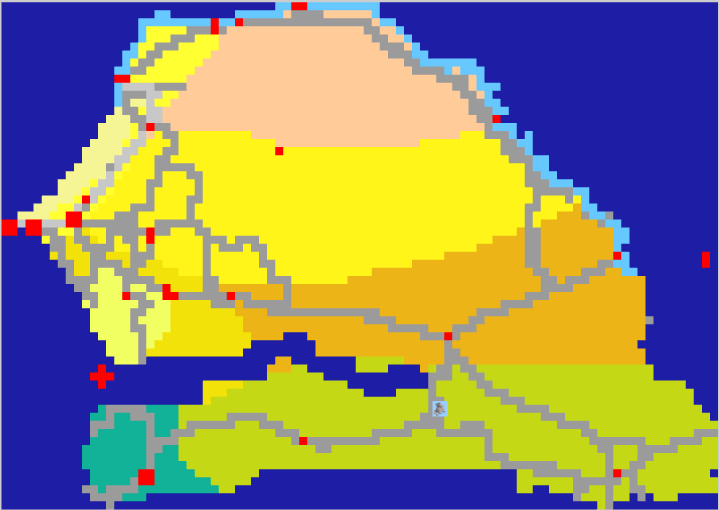
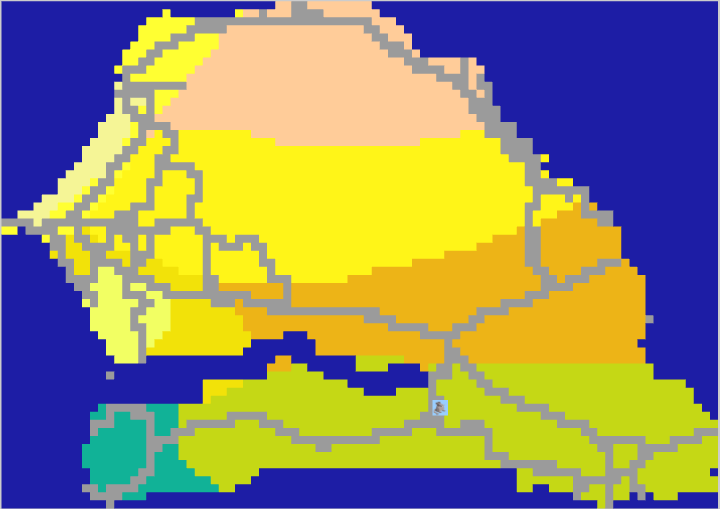
2



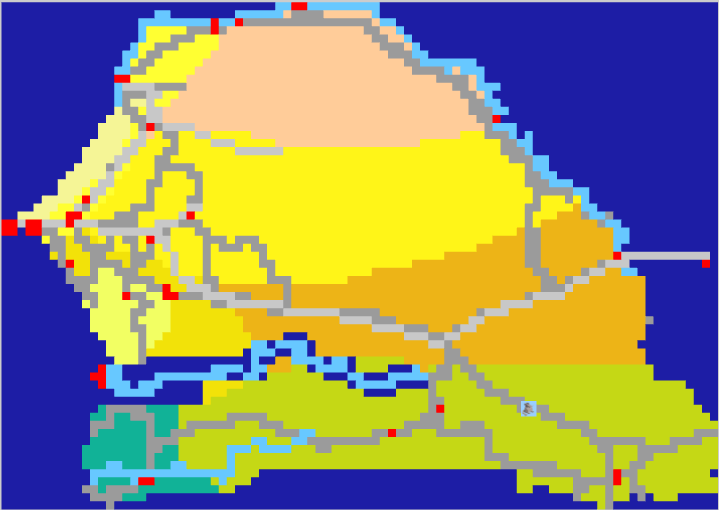
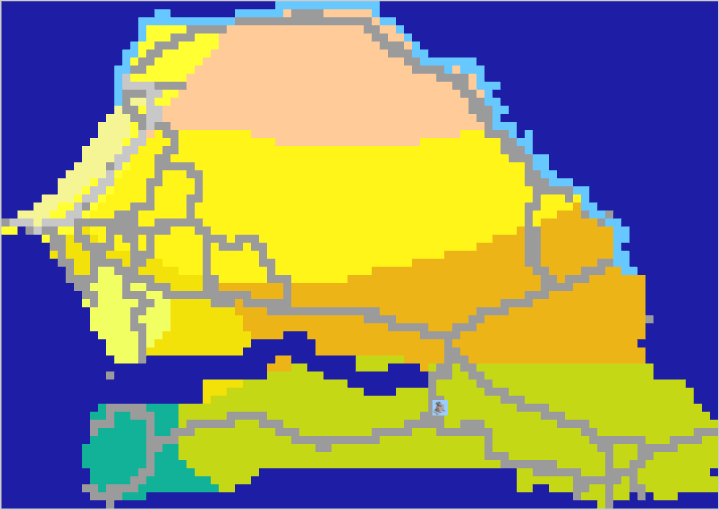
Par rapport aux routes en gris foncé :

4

4



Par rapport aux river en bleu clair :



Je ne sais pas, il n’y a pas getvaluelayer dans le mien j’ai exactement get**V**alueLayer

Sur mon ordi

Le \RepastSimphony-2.0\workspace\SIMmasto\_0\data\_csv

Si on a une nouvelle ville, on regarde dans quel graphe il appartient et on le mait dans son groupe de ville dans l’InspecteurCentenal. Comme ça on éliminera le **if**(events[*CITY*] == 1){ … // Nouvelle ville

Dans C\_ProtocolCentenal

J’ai touché C\_ProtocolCentenal

Dans C\_SoilCell j’ai ajour 2 champs :

Un tableau d’entiers : numberInGraph

Quand un soilcell devient une ville, je regarde dans son tableau numberInGraph là où on n’a pas -1 pour savoir qu’il appartient au graphe correspondant.

C\_ProtocolCentenal

C\_ManageInputDataCSV

I\_sim\_constants

C\_RasterGraphManager

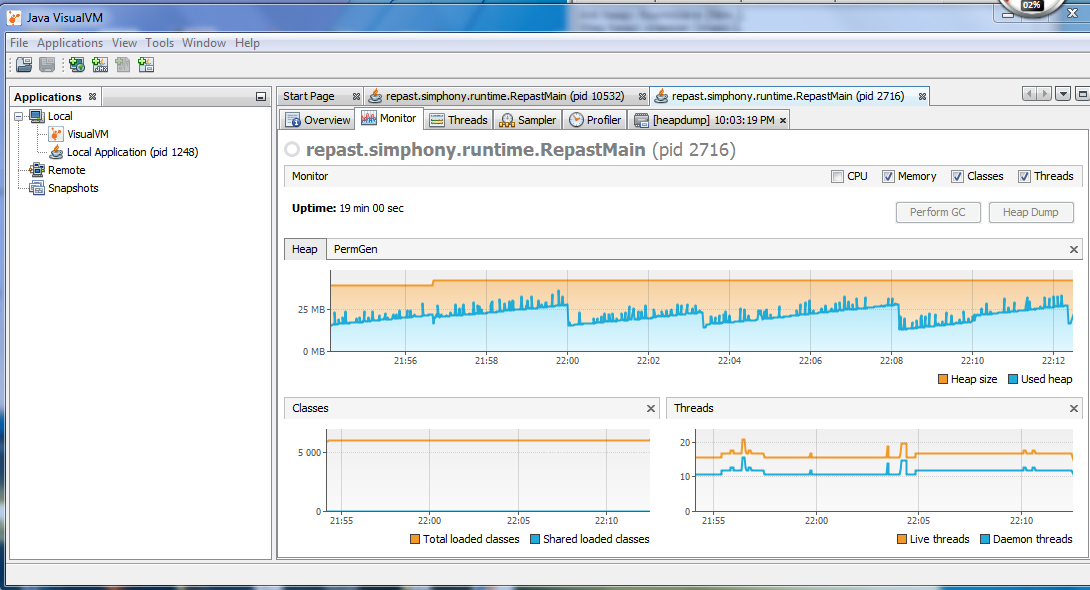
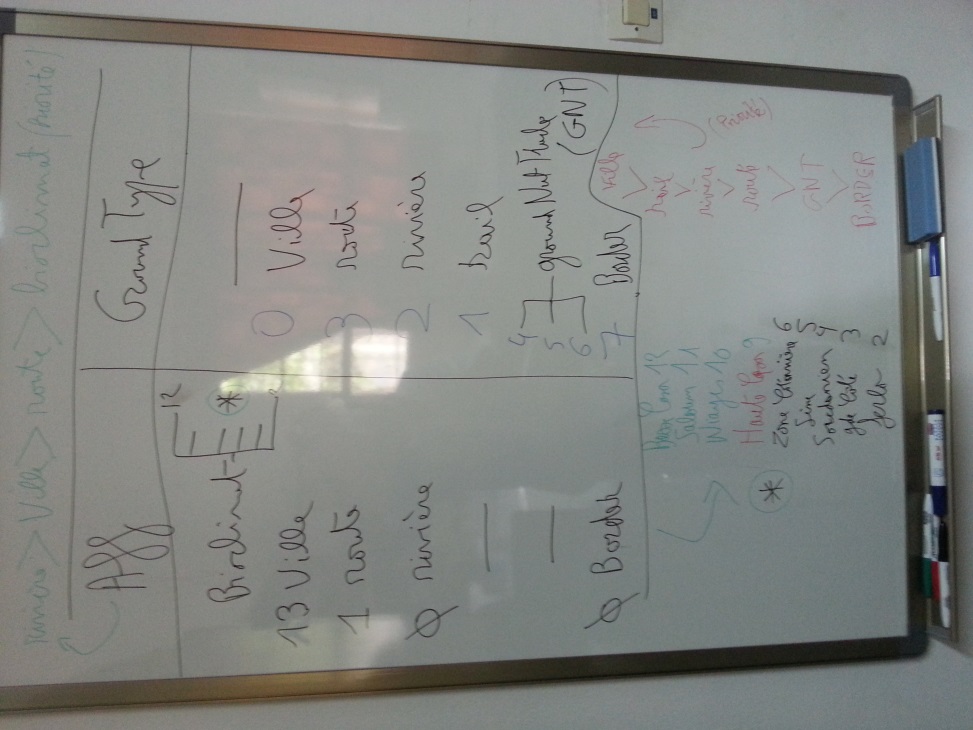
C\_UnweightedGraph

C\_SoilCell

C\_HumanCarrier

C\_Style2d\_terrain

C\_InspectorCentenal



Dans I\_Centenal\_Constants :

**public** **static** **final** **int** *X\_COL* = 0, *Y\_COL* = 1, *DATE\_COL* = 2, *CELL\_ID\_COL* = 3, *EVENT\_COL* = 4, *VALUE\_COL* = 5, *COMMENT\_COL* =6; //bien ordonnée

**public** **static** **final** String

*CITY* = "city", *RAIL* = "rail", *RIVER* = "river", *ROAD* = "road",

*GROUND\_NUT\_TRADE\_WEAK* = "groundNutTrade",

*GROUND\_NUT\_TRADE\_MEDIUM* = "groundNutTrade",

*GROUND\_NUT\_TRADE\_HEAVY* = "groundNutTrade",

*BORDER* = "border",

*RATS* = "rats", *TRUCK* = "truck", *TRAIN* = "train", *BOAT* = "boat";

**public** **static** **final** **int** *CITY\_INT* =0;

Dans C\_Protocol\_Centenal:

Dans le constructeur:

groundTypesCodes.put(*CITY*, 0);

groundTypesCodes.put(*RAIL*, 1);

groundTypesCodes.put(*RIVER*, 2);

groundTypesCodes.put(*ROAD*, 3);

groundTypesCodes.put(*GROUND\_NUT\_TRADE\_WEAK*, 4);

groundTypesCodes.put(*GROUND\_NUT\_TRADE\_MEDIUM*, 5);

groundTypesCodes.put(*GROUND\_NUT\_TRADE\_HEAVY*, 6);

groundTypesCodes.put(*BORDER*, 7);

groundTypesCodes.put(*RATS*, -1);

groundTypesCodes.put(*TRUCK*, -1); *TRUCK*

groundTypesCodes.put(*TRAIN*, -1);

groundTypesCodes.put(*BOAT*, -1);

04/07/2013

//**TODO** touchedEvents as a boolean array ok

* **SoilCell**
  + groundTypeAndName : Chaque soilCell aura une liste de groundType et un nom qui seront lié. Donc si un soilCell est une cellule de la ville de Dakar, Alors ce soilCell aura un groundType city de nom Dakar. Et s’il est en meme temps

pas groundTypeAndName  mais juste **groundTypeList** et **name OK**

Un soilCell peut avoir un meme groundType avec trois noms différents ex : **impossible**

La soilCell 3, 35 est un groundNutTrade de type MEDIUM et WEAK **c’est pas dans une meme date**

PourgroundNutTrade les types sont GNTmedium, GNTweak et GNTheavy et pas de nom

* **NumberInGraph** : si un soilCell appartient à un ou des graphes, ce champs permet de stocker les références de ces graphes avec les numéros de ce soilCell dans c’est graphes. **OK**
* **LandPlot**
* Est-ce qu’une cellule peut appartenir à plusieurs landPlot ? oui (ex (3,35) qui est en meme temps MEDIUM et WEAK)
* Est-ce que je dois construire automatiquement tous les landPlot possible ou je dois les construire à la demande ?

Je dois construire un landPlot à chaque arrivé d’un évènement nécessitant un landPlot **OK**

* Quand je construis un landPlot, est ce que je détecte automatiquement les villes dans ce landPlot ou bien je fais une méthode à part me permettant de construire la liste des villes dans un landPlot donné ? le 2ème **OK**
* Un seul graphe dans un landPlot
* Faire une sous classe de landPlot C\_City
* **City ICI**
  + Faire associer une liste landPlot de villes à tous les landPlot sauf les landPlot de type ville (un carrier va donc connaitre le LP où il va évoluer)
  + Ou bien faire une liste de landPlot de type ville avec des noms, comme ça chaque carrier aura sa liste de ville (pris dans le chrono)
* **Graphe**
  + **Je construis au besoin un graphe pour un landPlot donné (et qu’est ce qui m’indique pour cela)**
* **Carrier (dès sa création)**
  + **Il peut connaitre un LandPlot et donc la liste de ville associée**
  + **Et avoir un la référence de la liste de villes où il peut aller**
  + **Il peut stocker la référence du graphe dans lequel il va évoluer**
  + **Si j’ai le landPlot, je dois pouvoir accéder à sa liste de villes associées et à son graphe associé (Mais c’est pas tous les landPlot qui en ont)**

**Nouveaux Points**

* Construire des graphes dans des landPlot

|  |  |
| --- | --- |
| Liste des groundType | |
| 1 | city |
| 2 | groundNutTrade |
| 3 | rail |
| 4 | river |
| 5 | road |
| 6 | border |

|  |  |  |
| --- | --- | --- |
| **Liste des landPlot** | | |
| **N°** | **groundType** | **name** |
| 1 | city | Banjul |
| 2 | city | Dagana |
| 3 | city | Dakar |
| 4 | city | Diaobe |
| 5 | city | Diourbel |
| 6 | city | Foundiougne |
| 7 | city | Kaffrine |
| 8 | city | Kaolack |
| 9 | city | Kayes |
| 10 | city | Kedougou |
| 11 | city | Kidira |
| 12 | city | Kolda |
| 13 | city | Linguere |
| 14 | city | Louga |
| 15 | city | Matam |
| 16 | city | Mbour |
| 17 | city | Medina Gounass |
| 18 | city | Podor |
| 19 | city | Richard Toll |
| 20 | city | Rufisque |
| 21 | city | St\_Louis |
| 22 | city | Tambacounda |
| 23 | city | Thies |
| 24 | city | Tivaouane |
| 25 | city | Touba Mbacke |
| 26 | city | Ziguinchor |
| 27 | groundNutTrade | HEAVY |
| 28 | groundNutTrade | MEDIUM |
| 29 | groundNutTrade | WEAK |
| 30 | river | Casamance |
| 31 | river | Gambia |
| 32 | river | Senegal |
| 33 | road |  |
| 34 | rail |  |
| 35 | border |  |

J’ai créé une classe C\_CsvChronoReader me permettant de bien stocker les données à partir du fichier centenalEvents et de construire aussi le fichier raster.txt. ainsi cette classe à principalement 2 methodes :

<http://repast.sourceforge.net/docs/reference/SIM/GUI%20Parameters%20and%20Probes.html>

**A faire :**

1. Scheduler : C\_Calendar, C\_tableauDeBord, C\_Parameters

**???**

Size et randomGenerator

1. Le chrono ne prend pas encore en compte le suppression d’une partie du territoire ?

**A retenir :**

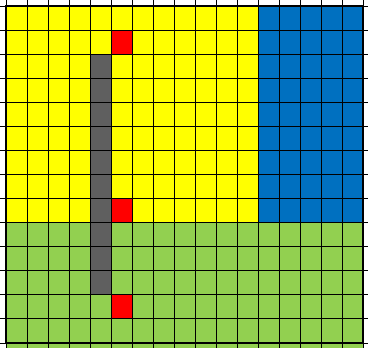
Dans mon cas dans C\_SoilCell je préfère HashSet que TreeSet par raison de compléxité

Dans mon cas dans C\_RasterGraphManager je préfère utiliser les HashMap que les TreeMap

paoiudslkgjatjgljfotiljvdql

Construction des graphe (graphe pour les GNT)

Cette cellule à le meme grapheTyphe que sa précédente, mais elle n’appartient pas dans GNT jaune …



**A faire dans le chrono officiel :**

**10/11/2013**

Créer la ville de Dakar et de Saint-Louis avant avant de mettre le train pour le trajet Saint-Louis Dakar.

I.e. pour Thiès et Kaye OK

Remplacer : OK

groundNutTrade MEDIUM par GNT-MEDIUM

groundNutTrade WEAK par GNT-WEAK

groundNutTrade HEAVY par GNT-HEAVY

Richard Toll par Richard-Toll

Touba Mbacke par Touba-Mbacke

Medina Gounass par Medina-Gounass

population par deprecated

Si deux ville ont une même date de création, faire avancer l’une de 1 jour, pour que les noms des villes dans la citiesList puisse respecter l’ordre des noms des villes en paramètre ou en constante. Car si deux villes sont créées dans la même date, identifyLandPlot ajoute en premier lieu la première ville qu’elle rencontrera quand il scanne la matrice.

Une autre solution serai de changer la manière de créer les landPlots à partir du chrono en ne pas utiliser identifyLandPlot() Et comme ça on pourra respecter l’ordre d’arrivée des villes dans le chrono

Cette manière de faire consiste à :

Ex :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **N°** | **Date** | **x** | **y** | **event** | **value1** |
| 1 | d1 | 0 | 34 | city | Dakar |
| 2 | d2 | 1 | 34 | city | Dakar |
| 3 | d3 | 0 | 35 | city | Dakar |
| 4 | d4 | 1 | 35 | city | Dakar |

A la ligne 1, si ça n’existe pas encore, je crée le landPlot de type city et du nom de Dakar avec la seule soilCell (0, 34).

A la ligne 2, si le landPlot de type city et du nom de Dakar, existe, je lui ajoute la soilCell (1, 34).

…

Pas de problème si d1 == d2s ou si deux villes ont la même date de création, car le travail est indépendant de la date et est fait dans manageOneEventLine().

**12/12/2013**

On utilisait rats au lieu de rat (Il a suffi de changer rats par rat dans I\_Centenal\_Constant) OK

Créer toutes les villes 02/01/1910 (au moins une ville doit être construite au commencement de la simulation pour être conforme à l’écriture de données output)

Changer toutes les dates 01/01/1910 par 02/01/191 OK

**15/12/2013**

Il y a une population qu’on doit mettre en depricated

**Fin A faire dans le chrono officiel**

**-------------------------------------------------------------------------------------------------------**

**A faire dans le simulateur :**

**17/12/2013**

Les affinités des soilCells ne sont pas mises à jour, ils ont les affinités des **bioclimats**, ie du **valueLayer1**

Ie les rats ne savent pas s’il y a une route au un chemin de fer quelque part…

**Fin A faire dans le simulateur :**

**-------------------------------------------------------------------------------------------------------**

**Idées**

**10/12/2013**

* Si un rat fait un findObjectsOnGrid(), tanqu’il ne sont pas de son SC où il a fait findObjectsOnGrid(), alors il n’a pas besoin de refaire findObjectsOnGrid ou du moins il n’a pas besoin de faire tout le findObjectsOnGrid()… Donc scinder cette méthode en 2 …
* Les SC stockent toutes les cellules de son voisinage le rayon de perception des rats en en 2 tableau :
  + L’un contient les cellules sont entièrement dans le raton de perception (cellules dont tout ce qu’elles contiennent est perçu par le rat),
  + L’autre celles qui ne sont pas entièrement dans le rayon de perception (cellules dont tout ce qu’elles contiennent n’est pas forcement perçu par le rat).

Comme ça on pourra simplifier findObjectsOnGrid() jusqu’à ce qu’il ne reste que la partie de JLF (qui sera adapté suivant les 2 tableaux).

**Fin idées**

**-------------------------------------------------------------------------------------------------------**

**Déjà fonctionnel**

**18/12/2013**

Des transporteurs sur une zone spécifique avec le mot clé area-type Ex : area-type:GNT-HEAVY

**Fin déjà fonctionnel**

**-------------------------------------------------------------------------------------------------------**

**Questions :**

1. Comment prendre en compte la reproduction des rats dont un HC devait amener dans une ville au 5ième et qu’avec le multi temps, ils ne sont arrivés que 7 mois après ?
   * Faire une simulation pour trouver un bon facteur multiplicatif peut être.
   * Réveiller les hc à chaque saison de reproduction des rats
     + Une idée en multi temps est de réveiller les agents pendand leur saison d’activité les plus importantes.
2. Faire une simulation pour le super temps pour voir si cette notion est bonne ou pour le paramétrer et un article la dessus pour CARI 2014.

**Fin Questions**

**10/03/2014**

La gestion des dates de début de la simulation

Date de début de la simulation est récupérée en paramètre : touchés :

parameters.xml, parameters\_Centenal.xml, C\_Parameters, C\_Calendar

**Idée** pour ne plus changer les parameters.xml pour passer d'une sim à une autre :

Ecrire une petite méthode dans une bonne place dans C\_Parameters, permettant

de copier automatiquement la parameters\_Centenal.xml sur parameters.xml. ~~S'il n'est~~

~~pas déjà laba (mettre en commentaire à la deuxième ligne du .xml le nom du protocole)~~

ie pour scenario...

en fait on peut meme faire parameters\_scenario\_nomProtocol.xml et y mettre tout ce qui concerne ce

protocole en parametre et en scenarion (avec un séparateur bien sûr)

C\_ReadWriteFile

C\_OutputData

I\_sim\_constants

I\_Bandia\_constants (adaptation de l’appelle d’une méthode)

C\_ProtocolCentenal(adaptation de l’appelle d’une méthode)

C\_RasterGraphManager

**C\_OutputData et C\_ReadWriteFile**

NowC\_OutputData s’occupe de tout ce qui est écriture dans un fichier

**C\_ReadWriteFile s’occupe de tout ce qui est copier ou transformer les données d’un fichier vers un autre**

**Il utilise** C\_OutputData pour écrire dans un fichier

**C\_CsvChronoReader et C\_ReadWriteFile**

buildRasterFileFromChrono() est enlevée de la classe C\_CsvChronoReader.

Elle est now dans **C\_ReadWriteFile**

**Et pourquoi pas** retrieveChrono() aussi. Comme ça on renommera C\_CsvChronoReader par C\_CsvInterneChrono

**Idée** faire de sorte qu'il y ait qu'une seule fois le nom du protocole dans toute la simulation

Mettre le nom du protocole que dans les paramètres (la 1ère ligne)

Implémentation de la gestion de time Condensed

Si la méthode activateTimeCondensed() de A\_TimeCondensed\_Mammal n'est pas appeller (juste après la création de l'agent)

alors la time condensed n’est pas activé pour cet agent.

* On peut travailler avec et ou sans time condensed
* Et en time condensed on peut travailler en tick et ou en date.
  + En date on peut travailler en heure, jour, mois et/ou année.

Ajout de d’une classe abstraite et de de classes qui étendent la classe abstraite :

* + - A\_ManageActAccordTimUnit
    - C\_ManageAct\_Udate
    - C\_ManageAct\_Utick

J'ai découper la méthode manageOneEventLine() du CPC

Tout ceci doit etre testés **Fin 10/03/2014**

**11/03/2014**

Test des travaux du **10/03/2014**

**Bug : on décharge d’un véhicule un rat qui n’existe plus :**

Le problème dans C\_RodentCommensal

manque de cohérence entre step() deliberation() tryBoardingVehicle() interact()

**12/03/2014**

Petite incohérence dans C\_RodentCommensal et entre C\_RodentCommensal et A\_Animal :

1. Ces appelle font un tour, ce qui donne une mauvaise valeur à trappedOnBoard

C\_RodentCommensal (

Deliberation (

A\_Animal (

Deliberation (

interact(

C\_RodentCommensal(

interact))))

1. Dans C\_RodentCommensal :
   * step(), tryBoardingVehicle() et deliberation() ont des parties qui font exactement la meme chose

**Bug réglé :**

Maintenant un C\_RodentCommensal fait :

perception() retourne [TreeSet](eclipse-javadoc:%E2%98%82=SIMmasto_0/src%3Cthing%7BA_VisibleAgent.java%E2%98%83A_VisibleAgent~perception%E2%98%82TreeSet)<[I\_situated\_thing](eclipse-javadoc:%E2%98%82=SIMmasto_0/src%3Cthing%7BA_VisibleAgent.java%E2%98%83A_VisibleAgent~perception%E2%98%82I_situated_thing)>

deliberation(reçoi [TreeSet](eclipse-javadoc:%E2%98%82=SIMmasto_0/src%3Cthing%7BA_VisibleAgent.java%E2%98%83A_VisibleAgent~perception%E2%98%82TreeSet)<[I\_situated\_thing](eclipse-javadoc:%E2%98%82=SIMmasto_0/src%3Cthing%7BA_VisibleAgent.java%E2%98%83A_VisibleAgent~perception%E2%98%82I_situated_thing)>) retourne [ArrayList](eclipse-javadoc:%E2%98%82=SIMmasto_0/src%3Cthing%7BC_RodentCommensal.java%E2%98%83C_RodentCommensal~deliberation~QTreeSet%5C%3CQI_situated_thing;%3E;%E2%98%82ArrayList)<[C\_SoilCell](eclipse-javadoc:%E2%98%82=SIMmasto_0/src%3Cthing%7BC_RodentCommensal.java%E2%98%83C_RodentCommensal~deliberation~QTreeSet%5C%3CQI_situated_thing;%3E;%E2%98%82C_SoilCell)>

selectDestination (reçoi [ArrayList](eclipse-javadoc:%E2%98%82=SIMmasto_0/src%3Cthing%7BC_RodentCommensal.java%E2%98%83C_RodentCommensal~deliberation~QTreeSet%5C%3CQI_situated_thing;%3E;%E2%98%82ArrayList)<[C\_SoilCell](eclipse-javadoc:%E2%98%82=SIMmasto_0/src%3Cthing%7BC_RodentCommensal.java%E2%98%83C_RodentCommensal~deliberation~QTreeSet%5C%3CQI_situated_thing;%3E;%E2%98%82C_SoilCell)>) retourne [C\_SoilCell](eclipse-javadoc:%E2%98%82=SIMmasto_0/src%3Cthing%7BC_RodentCommensal.java%E2%98%83C_RodentCommensal~deliberation~QTreeSet%5C%3CQI_situated_thing;%3E;%E2%98%82C_SoilCell)

interact(reçoi (C\_Vehicle) [C\_SoilCell](eclipse-javadoc:%E2%98%82=SIMmasto_0/src%3Cthing%7BC_RodentCommensal.java%E2%98%83C_RodentCommensal~deliberation~QTreeSet%5C%3CQI_situated_thing;%3E;%E2%98%82C_SoilCell)) // iteract ne peut par recevoir un HC ici sauf si on change en haut

tryBoardingVehicle(C\_Vehicle) suivant une proba, met le rat dans le véhicule

C\_RodentCommensal.step(){ ...

C\_SoilCell scDestinationTmp = selectDestination(deliberation(perception()));

**if** ((scDestinationTmp **instanceof** C\_Vehicle)) {

interact((C\_Vehicle) scDestinationTmp);

}...

**Questions :**

selectDestination() et deliberation() ne devrait–il pas etre la meme chose ???

ou bien devront-il pas appeler selectDestination() dans deliberation() ???

ou bien selectDestination() et interact() … ???

**13/03/2014**

Continuation des Test des travaux du **10/03/2014**

Test des travaux du **10/03/2014 OK**

J’ai mis undeprecated dans le chrono en 05/01/10957

Procédure pour la construction des routes suivant des dates…

Modélisation spatiale et temporelle

**17/03/2014**

Pour la gestion des populations des rats je divise la pop par 1000

DansC\_ProtocolCentenal.managePopRatEvent()

Contrainte du chronogramme :

1. respecter l’ordre : date, x, y, event, value1, value2
2. bien trier suivant la date,
3. bien unifier les mots utilisés (Ex ne pas mettre à la fois : St-Louis, saint-Louis, Saint-Louis …)
4. Ne pas mettre à une même date une instruction de création d’un évènement et une instruction qui utilise le même évènement créé

Ex :

02/01/1910 20 26 city Kaolack

02/01/1910 20 26 population Kaolack 1553

C’est-à-dire on crée la ville de Kaolack et on y met le même jour une population de 1553 rats.

Il faut attendre au moins le lendemain.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 05/01/1910 | 9 | 35 | truck | 1 | city:Kidira,Thies |

1. Pour spécifier à un agent une liste de villes, il faut respecter dans la colonne value2 la syntaxe

City:Ville1,ville2,ville3… sans espace

Ex :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 05/01/1910 | 9 | 35 | train | 1 | city:Thies,Kayes |

1. Pour que des transporteurs restent sur une zone spécifique, il faut utiliser la syntaxe

area-type:zone-spécifique

Ex : area-type:GNT-HEAVY

Instruction de construction de la ville de Dakar

Instruction de construction d’un chemin de fer

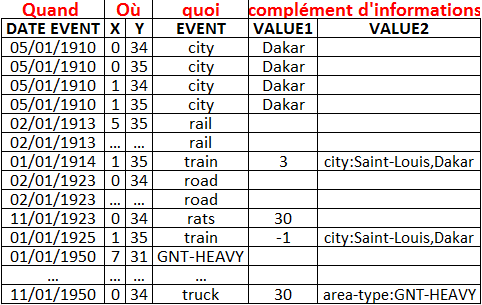
Instruction de fabrication de 3 trains ne faisant que le trajet Saint-Louis Dakar

Instruction de fabrication de 30 à la cellule de coordonnées 0, 34

Instruction de suppression de 1 train parmi ceux qui font le trajet Saint-Louis Dakar

Instruction de fabrication de la zone bassin arachidier GNT-HEAVY

Instruction de fabrication de 30 véhicules n’évoluant que dans la zone GNT-HEAVY



Time Condensed

**19/03/2014**

**A faire dans le chrono officiel :**

Mettre deprecated ou bien remplir cette ligne

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 05/01/1957 | 15 | 26 | deprecated | population | 10000 | 2330 | Dépt. … | 14/10/2013 | JLF-PAM | touché |

Si non on aura

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 05/01/1957 | 15 | 26 | population | 10000 |  | 2330 | Dépt. … | 14/10/2013 | JLF-PAM | touché |

date d’arriver des HC

Population et rat

Refait les dates pour les routes  bitumées OK :

route 1980 Podor Bakel ajout Matam cell(1) et d’autre cells cachées par la rivière

route 1960 Thiès St Louis ajout Thiès cell(2), SL cell(1), Louga cell(1)

route 1970 SL RT ajout RT cell(2) et cell(3) autour de RT

route 1960 DK Diourbel

route 1950 Rufisque Diourbel

route 1970 Kaolack Kafrine

route 1940 bleu près de Kafrine ajout Kafrine cell(1)

route 1980 rouge, Tamba ajout Tamba cell(1)

route 2000 Tamba Kidira ajout Kidira cell(1) et cell(1) en dessous de Kidira

route 1996 Tamba Ked ajout cell(75, 4)

route 1980 Diourbel Kaolack

route 2000 près de Diourbel

route 1940 bleu Kaolak vers touba

route quittant Kaolack et traversant la fleuve Gambie (j’ai mis provisoirement 2003 pour son année d’apparition)

**Chronogramme pour Time condansed :**

Les points de plus dans time condansed :

* Test isActive
* LoadHistory
* manageHistory

Pour mémo graphe :

Cas GNT, je voulais construire les graphes pour les portions des routes ne se trouvant qu’à l’intérieur des GNT. Pourquoi ? par ce que pour construire un chemin à partir d’un graphe il faut utiliser tous les nœuds du graphe. Ce qui est complexe si on utilise le graphe général même si les HC reste sur les villes du GNT. Mais ce problème de complexité ne se pose plus car les chemins sont calculés une seule fois car ils sont stockés. Donc il suffisant de dire à des HC de ne connaitre que les villes dans un GNT…

## 14/05/14 procédure pour test effet time condensed avec bruit de fond

Remettre dans l'ordre les city dans le csv

1-sans time condensed une graine (au moins 2x)  
 a) vérifier que les résultats soit identiques

2-sans time condensed une autre graine (conserver l'originale)

1-2 (pour une ville par exemple) -> bruit de fond lié à la graine

3-avec time condensed graine initiale

1-3 -> bruit de fond + effet du time condensed

<http://www.wikicfp.com/cfp/call?conference=multi-agent%20systems>

<http://jasss.soc.surrey.ac.uk/JASSS.html>

## 27.05.14 A FAIRE

- Lancer le cluster sur centenal pour compléter l’article cari (sans timeCondenser)

- Chercher des cibles de publication.

- grille décennal

- transformer la grille gravitaire en chrono de rats et de population.

- planning jusqu’à la fin de la thèse

- revoir code (C\_ProtocolCentenal)

- CR réunion

- revoir avec debug pas à pas qu’est-ce qu’il se passe quand les HCarriers sont en pause.

**Rmq sur les règles de codage:**

1. Dans une classe, commencer les noms des champs par this. Pour éviter les risques de confusion si une variable prend le même nom qu’un champ dans une méthode de la classe. Sinon ça peut créer un bug grave et quasi indétectable
2. Il faut instancier tous les champs dans les constructeurs, car Repast ne détruit pas les champs quand on redémarre la simulation, mais en appelant les constructeurs, les champs sont réinitialisés.
3. Quand on fait le refactoring, dans certain cas il aussi faire ctrl+H pour prendre en compte le refactor dans les commentaires et les printes.
4. Ne pas formater automatiquement le code. Car ça cause de difficultés de relectures et pire encore ça cause des problèmes de recherche avec ctrl+H

Ex. si on recherche « citiesListByAreaAndGraphLandPlot.get », alors ceci « citiesListByAreaAndGraphLandPlot

.get(areaLp).get(oneOldLandPlot)) » n’est pas trouvé à cause du retour à la ligne (formatage).

Ainsi à cause de la recherche avec ctrl+H il ne faut pas aussi écrire avec des espaces inutiles et des retours à la ligne n’importe comment.

Ex. citiesListByAreaAndGraphLandPlot .get

citiesListByAreaAndGraphLandPlot. Get

ceci doit etre écrit sur la même ligne ou du moins comme ça :

**if** (getCitiesListByAreaAndGraphLandPlot(areaLp, oneOldLandPlot) != **null** &&

oneCarrier.getMyCitiesList() == citiesListByAreaAndGraphLandPlot.get(areaLp).get(oneOldLandPlot))

Unifier les espaces entourant les opérateurs (+, =, ==, - …)

Ou bien ne pas chercher des du texte comme « citiesListByAreaAndGraphLandPlot.get ».

1. Mettre le développeur dans de bonne condition, ça augmente leur concentration, perception, rapidité et vigilance.

### 01/07/2014 cible de publication possible

* Colloque CNRIA <http://cnria.cci.ucad.sn/>
* Revue GEOINFORMATICA <http://link.springer.com/journal/10707>
* CYBERGEO <http://cybergeo.revues.org/>

Piste et route

Début tout en piste en terre

Et les mis à jours c’est pour bitumer les piste

Plus lent sur les pistes que sur les routes bitumées (track).

J’ai plus de speed\_UcellsByTick car speed\_UmeterByTick n’est plus constante et que speed\_UcellsByTick = speed\_UmeterByTick / C\_Parameters.CELL\_WIDTH\_UMETER

Maintenent si le plus

**08/09/2014 15:52**

From Jean

Bonjour Pape; Bonjour Malick, ce message te sera aussi utile je pense.

* J'ai presque fini la révision de la gestion des événements mais après le rasterGraphManager maintenant je tombe sur l'inspecteur transportation qui fait buguer le système. Problème.
* Je suis depuis vendredi sur ces sources, je m'arrache les cheveux, c'est extrêmement compliqué.
* Pour inspector Transportation, très mauvais point  :'( : record absolu du nombre de méthodes dans une classe, et en plus même pas pour une classe centrale, pour un simple inspecteur. Problème.
  + A passer ces dernières semaines sur ton code, je pense qu'en fait tu fais de la programmation linéaire : les procédures sont appelées une seule fois dans toute la plate-forme et en séquence); tu n'utilises pas assez le principe objet (ci-dessous) d'où le problème des sources trop longs notamment :
  + En effet, même si l'on maitrise bien la programmation objet ce qui est important c'est de bien assimiler, et mettre en œuvre**,le principe de** **réification**: les objets sont encapsulés et on peut leur déléguer tout un ensemble de tâches. Ceci permet de ne pas faire tout gérer par un inspecteur qui va s'empêtrer dans de la gestion de listes : noms de champs à rallonge, multiplication de procédures, code qui devient incompréhensible, qu'il faut exécuter dans l'ordre (!) et donc en tous cas pas fiable.
    - Par exemple dans inspector transportation 1) tu multiplies les listes, je redis encore une fois que les champs de listes sont une source d'erreur, 2) les map de map sont a priori le signe que le code est mal conçu, 3) les noms de variables sont composés de trop de mots, c'est aussi un signe de problème de conception, 4) avoir à la fois une méthode updateLandPlotCities(groundtype) et une updateCitiesByLandPlot(groundtype) est une erreur, cela pose des problèmes de cohérence et de compréhension  (et c'est un exemple).

updateLandPlotCities et updateCitiesByLandPlot ne font pas la même chose.

* + Pour continuer avec la réification et l'approche objet: un graphe peut être un agent exactement au même titre qu'un rongeur, il peut répondre à des requêtes et élaborer des listes qui sont à sa portée. Il faut exploiter ces possibilités (pas à fond, au mieux :-)
  + C'est la même chose pour une C\_City. Il faut encapsuler et déléguer les tâches aux agents, chacun fait sa part du travail (système distribué) et l'organisation est simplifiée et plus robuste.
    - par exemple une C\_City dispose d'une population, c'est <=> à son weight, il ne doit pas y avoir plus à gérer; il suffit de demander à la C\_City son weight. L'inspecteur n'a que la liste des C\_City à gérer c'est tout.
  + Chaque objet dispose de son petit bout de connaissance et de savoir faire et l'inspecteur leur envoie des requêtes pour répondre au protocole.
  + Tout est comme ça, il faut favoriser autant que possible la construction de listes à la volée plutôt que de stocker ça dans des champs qui sont ingérables (même si c'est au prix d'une perte de vitesse de simulation, mieux vaut des codes plus simples même s'ils sont plus longs à exécuter).
  + Pour faire ça, il faut que l'inspecteur sollicite les objets/agents et recombine les réponses.
  + Cela doit se faire pour les C\_Graph, les C\_City (avec des populationSize, CarrierList, ...), les C\_LandPlot ( avec des groundTypeList, des cityList) , les C\_HumanCarrier avec des cityList, ...). On ne fait pas une méthode A\_inspectorTransportation.setCitiesByLandPlot(oneLandPlot) mais plutôt C\_LandPlot.oneLandPlot.setCities();
    - A ce propos, tu peux faire une classe C\_Region qui étend C\_LandPlot et qui contient en plus un champ cityList avec toutes les procédures associées (attention pour l'héritage multiple entre C\_LandPlot et C\_City). Très bonne idée (Ok y aura pas de l’héritage multiple)
  + Pour prendre encore une image assez fidèle, simmasto est comme une petite usine, il y a des sections pour l'affichage, pour les données, etc. tout ceci au service de la section thing et avec quelques managers ayant une 'vision' globale. Il faut que chaque acteur de l'entreprise fonctionne bien pour que l'ensemble soit solide, certains postes sont occupés par un seul acteur, il ne faut pas les surcharger inutilement ou ils vont dérayer, il faut au moins les décharger au maximum. Certains postes sont occupés par beaucoup d'acteurs, il faut leur déléguer à chacun les actions qu'ils sont à même de réaliser pour distribuer la résolution de problème et alléger la gestion (système distribué), etc. etc.
* Pour C\_humanCarrier,
  + Pourquoi appeler deliberation() selectCity() ? Tu perds tout l'héritage de A\_Animal.
  + C'est la même chose d'ailleurs pour setCarriersMustDie de protocol transportation qui n'utilise pas le même schéma que pour les rongeurs (je travaille dessus, on en reparlera)
  + Il n'y a pas besoin de previousCityWeight dans C\_humanCarrier, c'est une inutile surcharge.
  + ...
* En réifiant bien, on peut simplifier le code (et faciliter sa compréhension) mais surtout gagner en robustesse. C'est grâce à cette robustesse que la plate-forme peut traiter des cas complètement différents et que tout le code développé même une année avant reste complètement applicable. Je ne dis pas ça en l'air... c'est ça tout le cahier des charges de la plate-forme et du projet SimMasto.

Tout cela est important pour vous car si on ne dispose pas d'un bon protocole \_et\_ inspecteur pour Transportation on ne pourra pas développer ni décennal, ni Dodel, et encore moins multiscale.

Il y a donc un effort de révision important à faire sur la base de ce principe de réification. Le problème est qu'il faut tout revoir notamment les procédures update... qui font des dizaines de lignes. Certaines sont très bien commentées (comme updateAllLandPlots par exemple et ne sont sans doute pas facilement modifiables (réifiables).  
  
Je viens de passer au moins trois semaines pleines à réviser du code, il faut maintenant vraiment que je travaille sur d'autres urgences. Donc on fera la révision ensemble mais peut-être pas tout de suite   
En attendant pouvez-vous voir ce que j'ai fait au niveau de la gestion des événements ? 

1. -> Le mieux est de passer une fois en debug par toutes les étapes de lecture d'une ligne de chrono jusqu'au bout (protocole centennal).
2. Vous pouvez faire le dernier update. Attention C\_RasterManager s'appelle C\_Raster et C\_RasterGraphManager C\_RasterGraph (ce sont aussi des agents du système, ils pourraient presque aller dans le package thing.ground mais c'est encore prématuré).
3. Vous démarrez ensuite le debug à C\_ProtocolTransportation.step\_Utick et avancez uniquement avec F5 pour voir toutes les méthodes qui ont été impliquées/révisées
4. vous pouvez même le faire ensemble pour plus d'efficacité).

Le centenal n'est toujours pas fonctionnel, le nouveau bug en cours est sur : C\_InspectorTransportation.managePopulationEvent()  
  
A+,  
Jean

1 lecture du chrono

Update les cellules de l’environnement pour tous les évènements d’une date

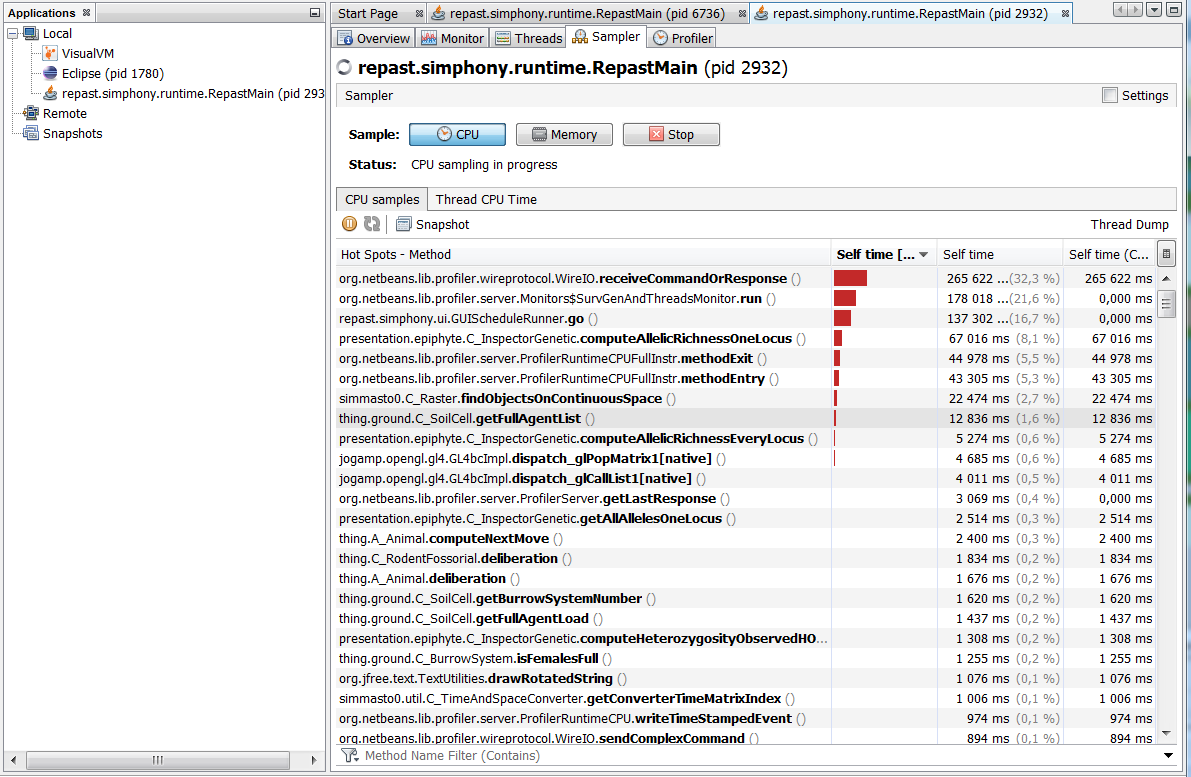
Protocole MUS\_TRANSPORT

CAGES

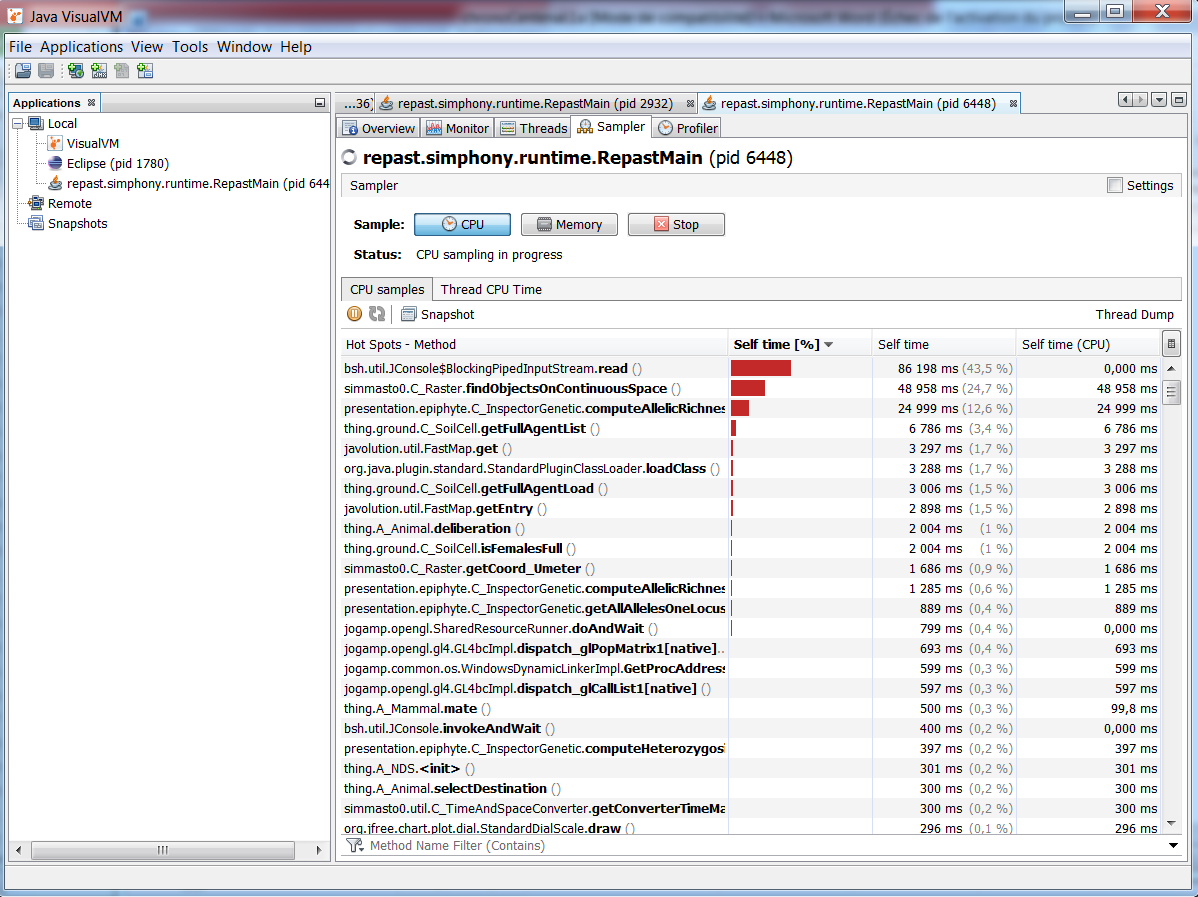
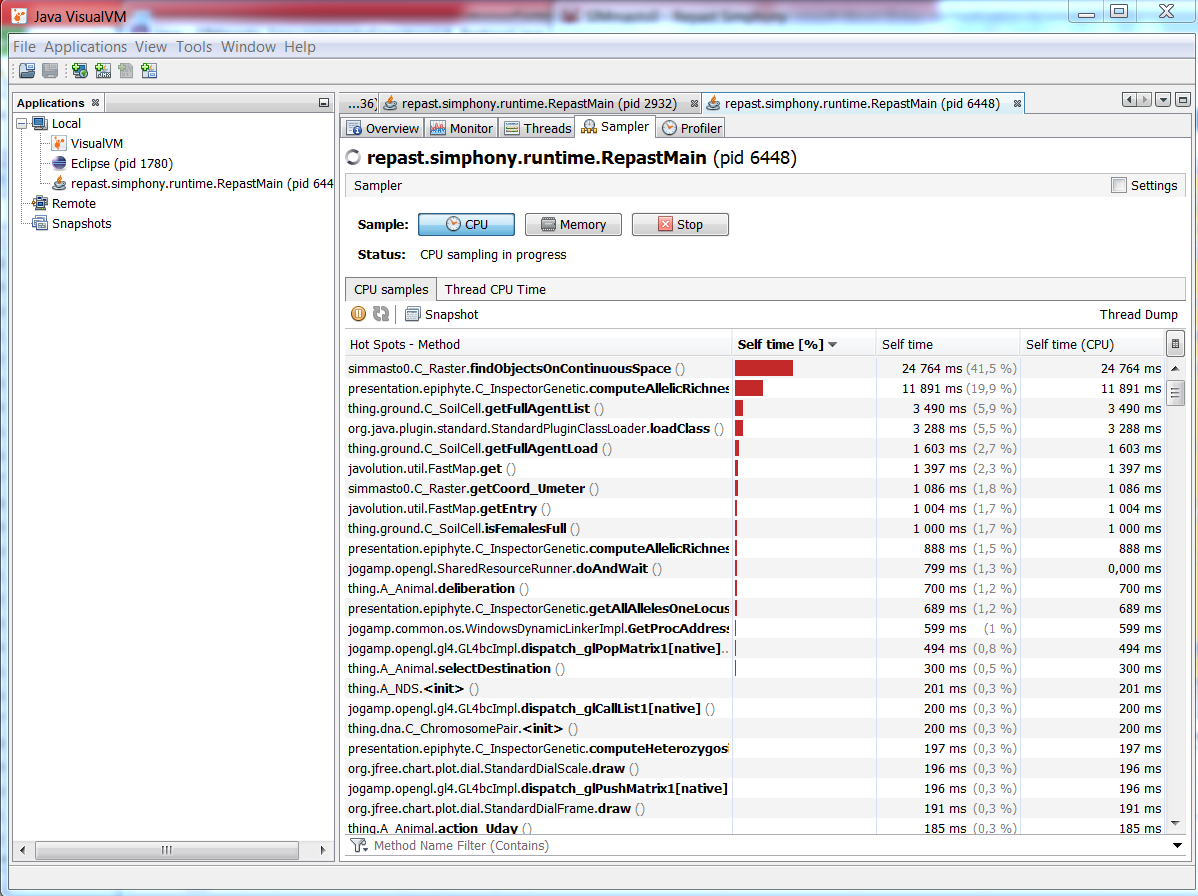
**20/06/2015**

**Teste de lenteur avec VisualVM, avec CHIZE, CENTENAL, HYBRID\_UNIFORM**

Pour CHIZE

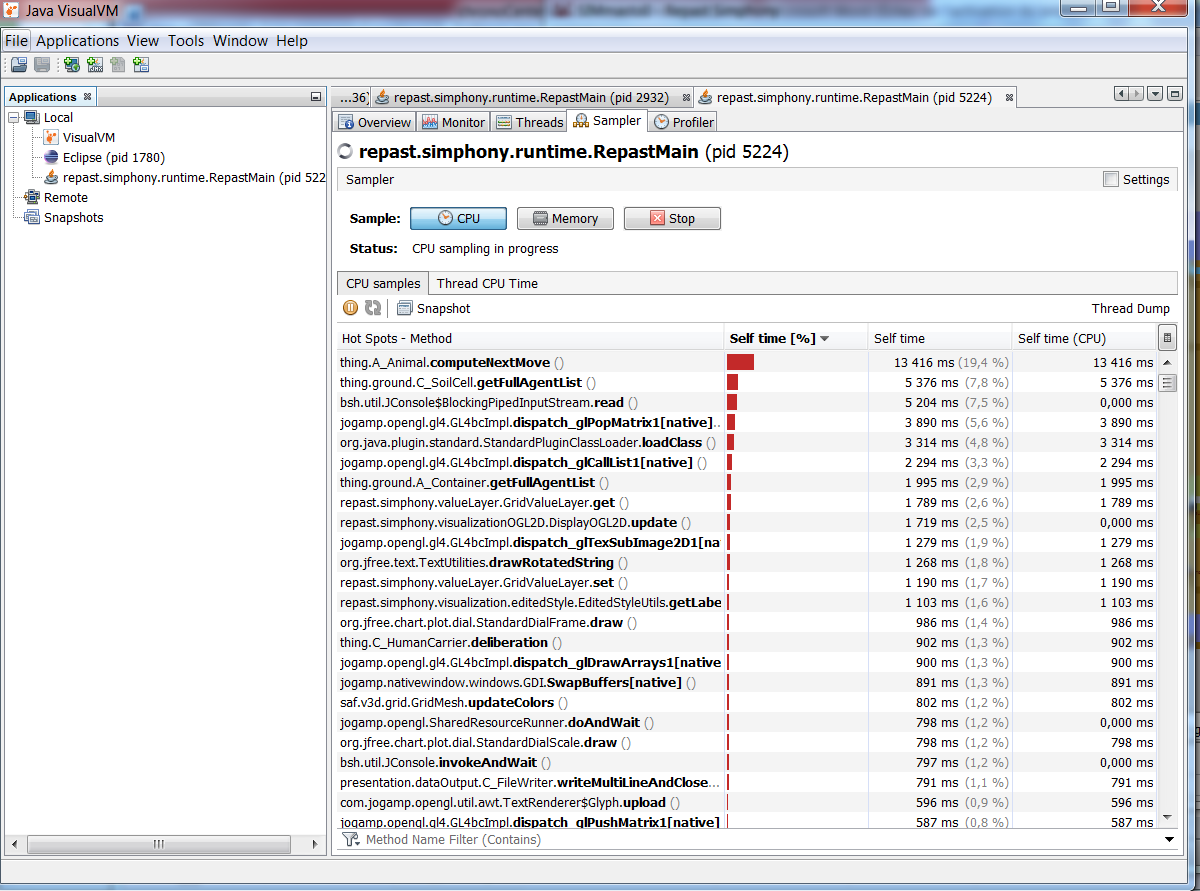


Pour HYBRID\_UNIFORME

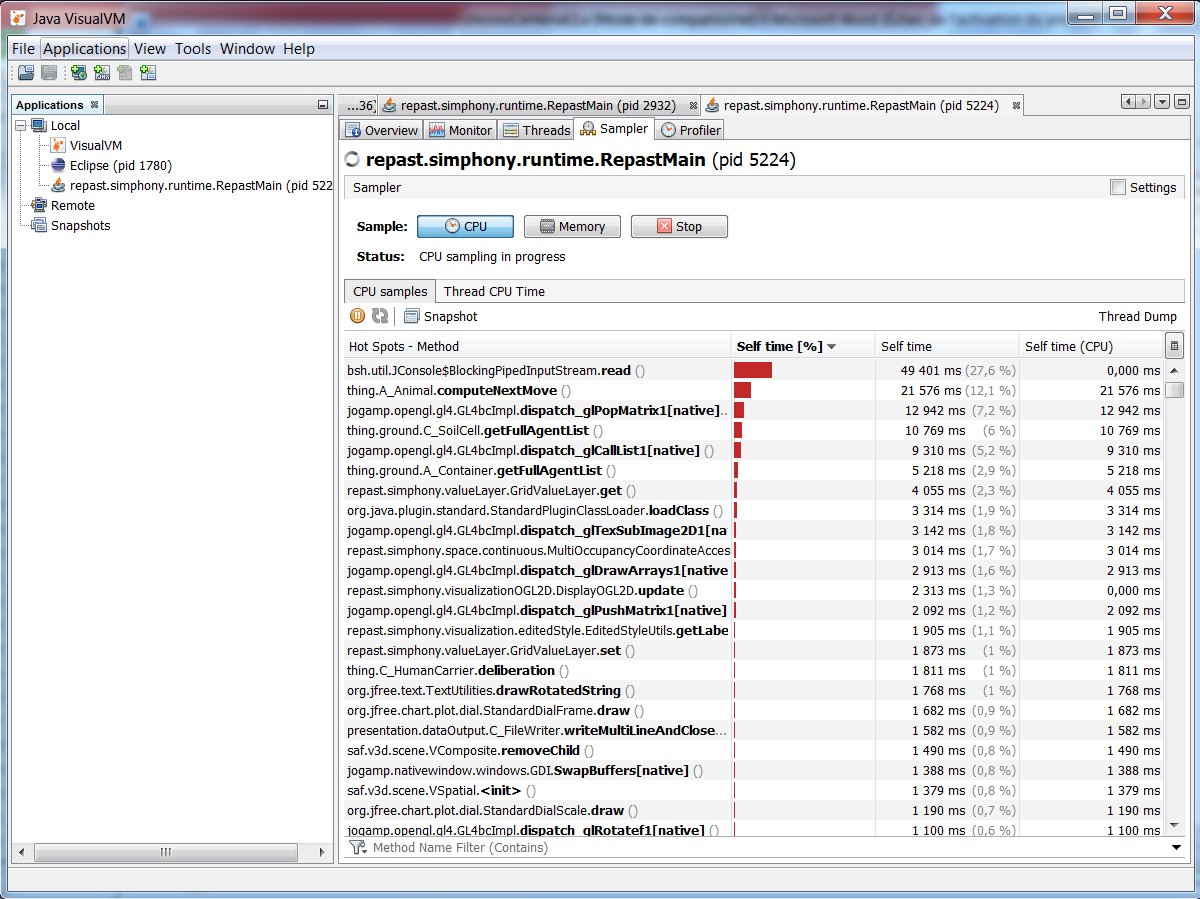


Pour CENTENAL

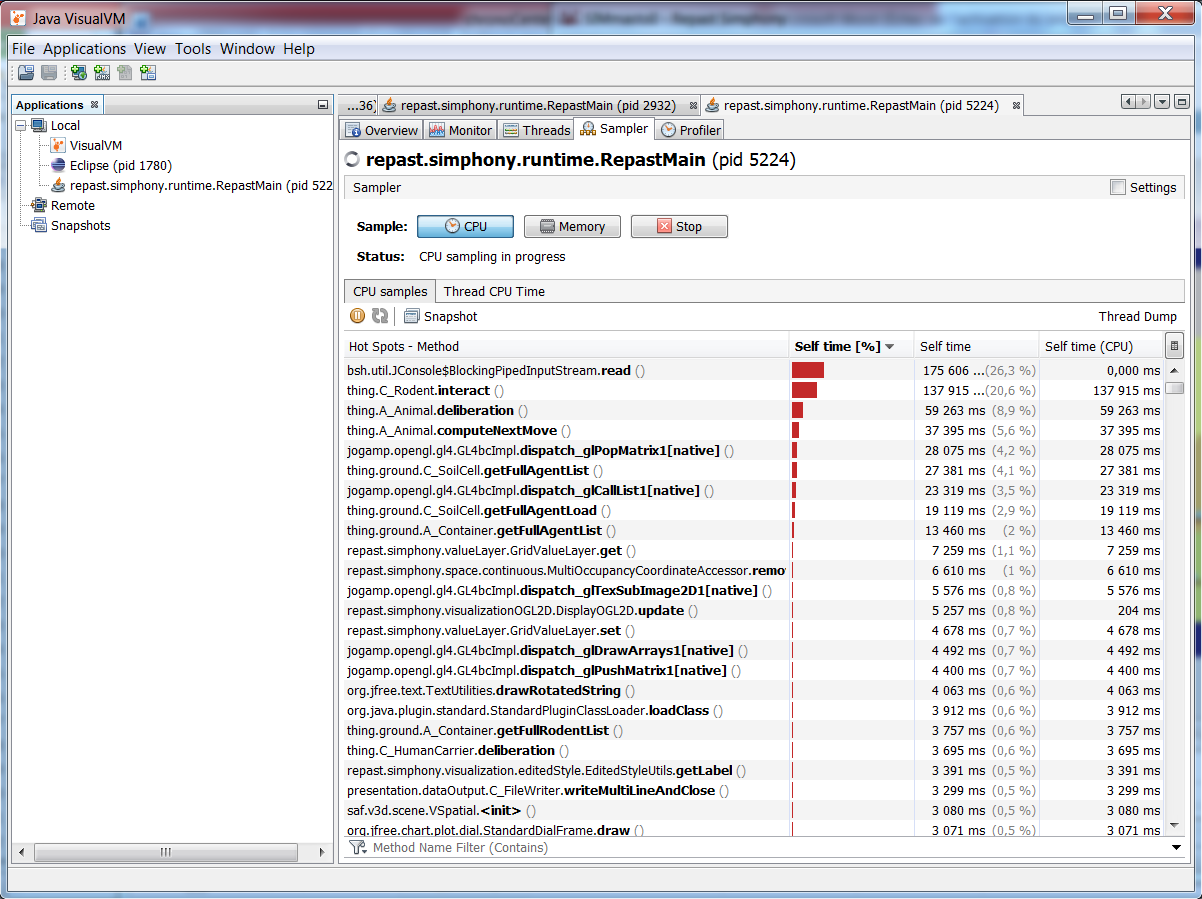
Au début et c’était pas lent, 1tick = 14jour



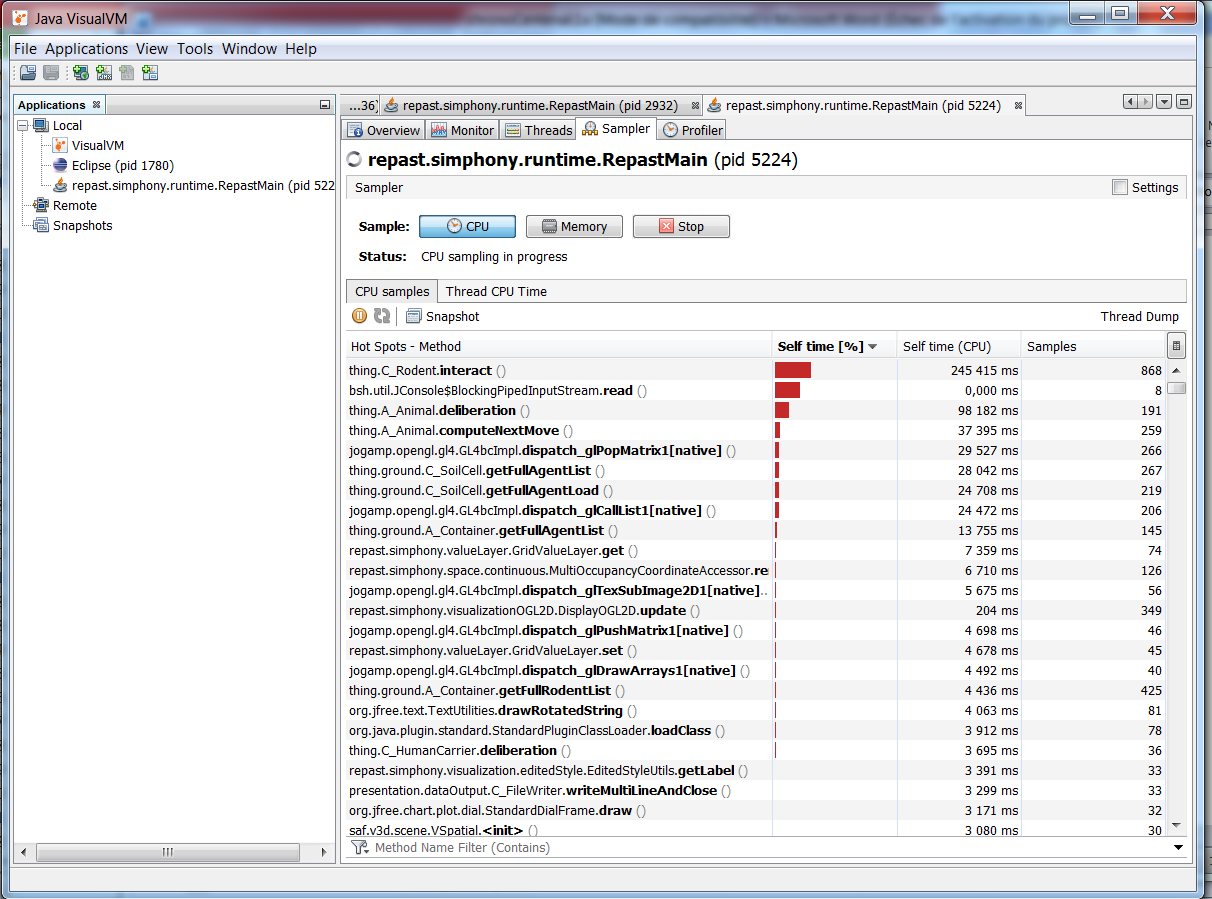
Au début et c’était pas lent, 1tick = 14jour



1950, tick = 1jour



1950, tick = 1h



**20/06/2015-28/06/2015**

Un déplacement d’un agent se fait en 2 phases :

1. sur l’espace continu, qui suit celui de repast
   * soit avec moveToLocation(thing, position)
   * soit avec moveObject(thing, distance) ou mouveObjectByDistance(thing, distance)
2. sur la grille ou soilCellMatrix qui se fait à partir du déplacement sur l’espace continu (1) :
   * en récupérant les coordonnées dans l’espace continu (unité continuous space \_Ucs),
   * le convertir en unité cellule (\_Ucell) : qu’on appelle newCell,
   * tester si thing.currentCell et newCell ont les mêmes coordonnées :
     + si oui on laisse comme ça,
     + sinon on utilise la méthode moveToCell(thing, newCell) pour le dépacer du currentCell au newCell

**Le déplacement d’un agent transporteur** : un transporteur se déplace sur des cellules de qualité différentes et donc avec une vitesse qui varie.

|  |  |  |  |
| --- | --- | --- | --- |
| **Boucle while** | **Distance max à parcourir dans une entrée de la boucle** | **Distance parcourue réellement** | **Distance intermédiaire parcourue = 0 au début** |
| **1** | **3 - 0 = 3** | **1,414 / 1** | **0 + 1,414 = 1,414** |
| **2** | **3 - 1,414 = 1,586** | **1 / 3 = 0,333** | **1,414 + 1 = 2,414** |
| **3** | **3 - 2,414 = 0,586** | **1 / 3 = 0,333** | **2,414 + 1 = 3,414** |

Tableau 2

**Remarque** : Pour passer du nœud 6 au nœud 7, le véhicule passe par une cellule qui n’est pas du graphe. Alors s’il teste si elle est orange ou grise il n’aura rien.

Si un véhicule doit faire 3m par tick, et que sa vitesse est divisée par **3** dans les cellules orange. Alors pour chaque step, voir tableau 2.

Pour une entrée dans la boucle while, le véhicule ne doit pas parcourir une distance supérieure à celle de la cellule (voir tableau 1)

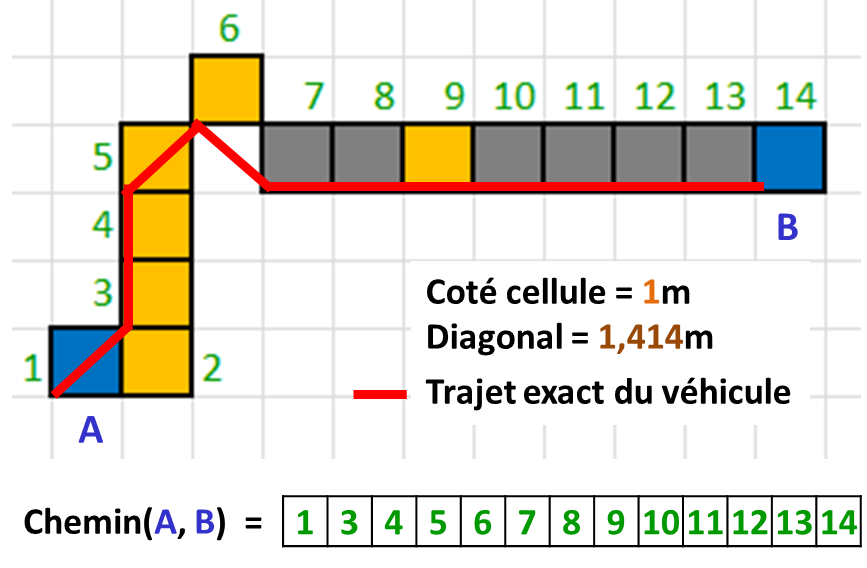


Tableau 1

|  |  |
| --- | --- |
| **Distance de déplacement dans une cellule** | |
| Diagonal cellule bleue | **1,414** |
| Coté cellule bleue | **1** |
| Diagonal cellule orange | **1,414/3 = 0,471** |
| Coté cellule orange | **1/3 = 0,333** |

**Déplacement des rats à l’intérieur d’une ville** :

**SelectDestination()**

Pour trouver aléatoirement un point dans la ville, il suffit de choisir deux nombres aléatoires (x, y) compris respectivement entre [100-300] et [100-200], ce point sera le targetPointe\_Umeter.

**computeNextMove()**

Il suffira ensuite de faire computeNextMove() avec la vitesse du rat et targetPointe\_Umeter

**moveObjet() ou moveObjectByDistance()**

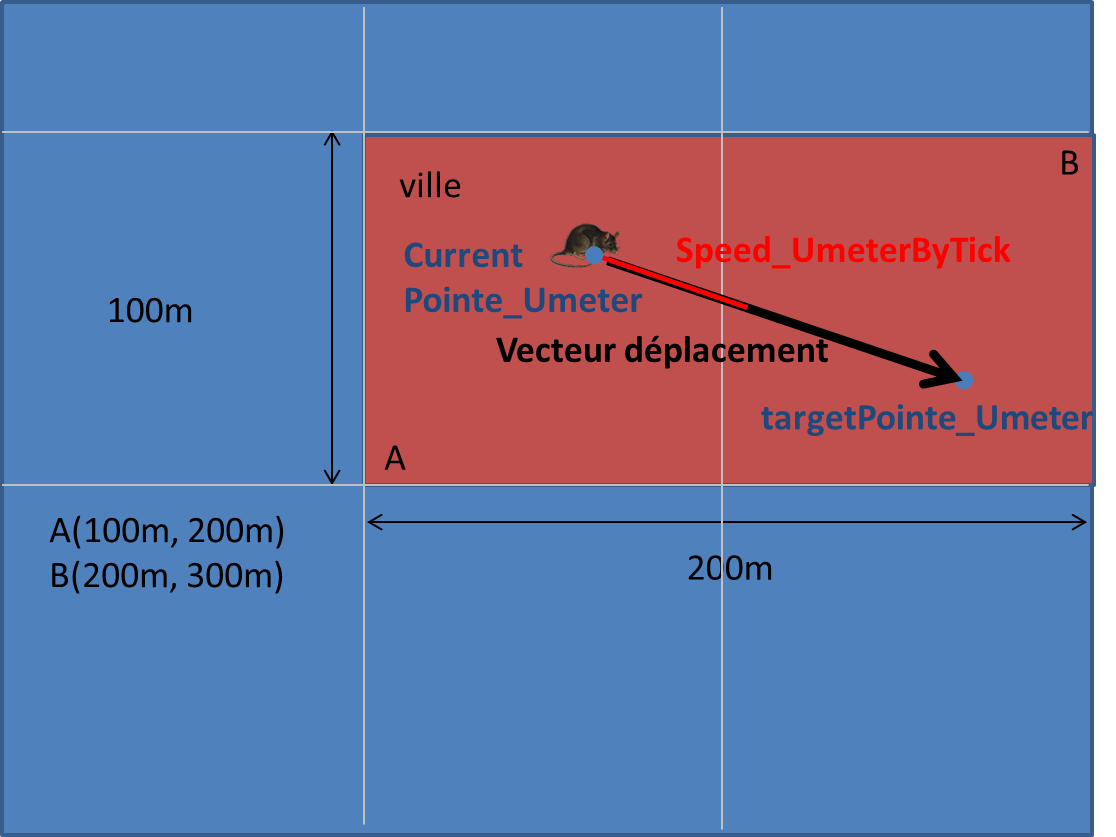
Et en fin faire un moveObject

**Formule pour générer un nombre aléatoire entre min et max :**

Soient plage = max – min, et randNombre un nombre aléatoire compris entre 0 et 1 ( [0, 1[ ). Alors on a :

Min + (randNombre \* plage) ∈ [min, max[

Il suffit donc qu’une ville ait un champ **origineCoord\_Umeter** pour min et un champ **plageCoord\_Umeter** pour plage



Ce déplacement évite au rats de s’entasser sur une position, de se persevoir tous et de persevoir tous les véhicules parqués dans la ville.

**Comment remédier à la lourdeur de SimMasto Global ?**

**Idées** :

1. mettre un champ booléen "bord" dans soilCel. Bord = true si la cellule est à l’extrémité du domaine.
   1. moveObject(), au lieu de faire checkGoalPosition() et bordure() pour chaque déplacement d’agent et pour chaque position, le faire que si currentSoilCell est de bord == true.
   2. Le nombre de rangés de cellule considérés comme bord dépend du speedUmeterByTick des rats ou des agents.
2. stocker le voisinage (cellules) de chaque cellule suivant le rayon de perception des rats. Ce qui permettra de supprimer toute 1ère partie de la méthode findObjectsOnContinuousSpace(). (On pourra même revoir cette méthode)
3. Si ce n’est pas suffisant, revoir le nombre d’interact entre les rats (peut-être faire de sorte que si deux rats interagissent, ils interagissent plus avec un autre rats dans un même tick…). J’ai un peut essayer celui-ci mais ce n’était pas suffisant.

**Idées pour CENTENAL** :

1. Les rats s’entassent dans une même position dans une ville. Ce qui augmente leur possibilité d’interaction et de se reproduire (chacun perçoit tous les autres rats et tous les véhicules parqués dans la ville). **OK**
2. Il faut donc permettre aux rats de se déplacer aléatoirement à l’intérieur de la ville
   1. Ceci est déjà réaliser, voir C\_RodentCommensalSimplified.getNewRandomDisplacement()**OK**
   2. Cette manière de faire nous facilitera le point 4.b suivant.
3. Analyses de sensibilité (paramètres)  (*diviser pour mieux régner*):
   1. Faire une simulation à part pour les rats **pour caler la dynamique des populations des rats dans une ville**, avec les paramètres suivants :
      1. Rayon de perception des rats
      2. Distance de déplacement par tick (par jour)
      3. Probabilité d’interagir avec un autre rat perçu
      4. Nombre de rats par portée ou bien le fixé mais suivant une probabilité comprise entre 4 et 6 …
      5. Probabilité de mourir ou d’être tué
   2. Faire une simulation à part pour les transporteurs, **pour caler leur dynamique (origine-destination),** avec les paramètres suivants :
      1. ~~Durée de stationnement des transporteurs après leurs step~~ **~~OK~~**
      2. Vitesse des transporteurs **(multiplicateur : faire à peu près 20km/h à 120 km/h)** **OK**
      3. ~~Probabilité de choix de destination des véhicules suivant une alternative pondérée par la taille de la population des villes ou par la distance entre les villes. Sinon prendre une alternative pondérée par la taille de la population des villes, divisées par la distance entre les villes~~
      4. Taille des supers agents **1, 5, 10, 20, 50, 100, 150 OK**
      5. Durée tick**: s,m,h,d,mon,y OK**
   3. Faire une simulation de CENTENAL à partir de 1999 (juste avant la construction de la route Tambacounda-Kédougou), en maintenant des quantités de rats dans les villes qu’on sait qu’il en avait en 1999 (pas de reproduction), **pour caller la probabilité de monter des rats dans les véhicules pour aller à Kédougou**. Car on connait :
      1. La fréquence à laquelle les transporteurs prennent la route Tambacounda-Kédougou (simulation b)
      2. La date d’arrivée des rats à Kédougou (à peu près)
   4. On pourra aussi faire de l’analyse de sensibilité sur la diffusion des rats par les bateaux coloniaux si on connait :
      1. La date du début de l’arrivée des bateaux coloniaux (voir JMD)
      2. La fréquence à laquelle ils venaient au Sénégal
      3. La date à laquelle les rats sont apparus au Sénégal
4. **En fin faire une simulation CENTENAL sur un siècle pour retracer la dynamique des rats dans vers les villes du Sénégal : les dates approximatives** **d’arriver des rats dans chaque ville du Sénégal :**
   1. Supposant qu’il n’y avait pas de rats au Sénégal vers 1900, il n’y avait qu’en Europe. Si on a réalisé le point 3.d. alors on pourra inclure l’Europe dans la simulation pour retracer l’histoire de la diffusion des rats, en disant qu’initialement il y a des rats qu’en Europe.

Sinon on pourra commencer avec une population de rats dans certaines villes su Sénégal à la date où les rats sont arrivés.



* 1. En ce qui concernera le step des rats (utile/inutile et lourdeur) ou pourra le diviser en deux groupes, ceux qui sont déjà été transporté et ceux qui ne sont pas encore :
     1. Pour les rats qui ne sont jamais été transportés, ils ne font récupérer agentList de la cellule où ils se trouvent, fait un tryBoardingVehicle() du premier transporteur dans agentList (puisque c’est un treeset il peut être trié de telle sorte que les transporteurs soient en tête ou en fin de liste. Ce qui permettra de récupérer un transporteur de la liste en une opération : agentList.first() ou agentList.last()). Ces rats ne se reproduisent pas. On maintient juste constant le nombre de rats dans les villes où ils se trouvent.
     2. Pour les rats qui sont une fois été transportés (leur descendance aussi hérite cette propriété), ils font un step normal en se reproduisant si possible (instinct de survie). Si leur nombre atteint la taille de la population humaine alors ils arrêtent et font comme dans le point i. Ils deviennent tous transportable. A cette étape si un nouveau rat arrive dans cette ville, il sera marqué non tansported comme les autres. (une ville aura un champ booléen fullofRodent qui sera testé par les transporteurs, pour savoir la situation de la ville en nombre de rats…).

1. Une autre idée pour montrer que les rats se sont diffusés via le transport humain est de minimiser au maximum la probabilité qu’un rat monte dans un véhicule (en ajoutant par exemple qu’un rat ne puisse être transporté que s’il a un certain âge qu’il atteint rarement). Si après toute cette minimisation des rats sont transporté alors ça prouvera que les rats se sont réellement diffusés par le transport humain. Ensuite on fait 5. Pour déterminer la fréquence et la probabilité.

**29/06/2015**

**Reconstruction des données réelles**

**Code pour déterminer la liste de tronçons possibles avec ces 34 villes**

String villeInSimulateur [] = {"Bakel", "Banjul", "Dagana", "Dahra", "Dakar", "Diaobe", "Diourbel", "Farafenni", "Foundiougne",

"Janjanbureh", "Kaffrine", "Kaolack", "Kayes", "Kebemer", "Kedougou", "Kidira", "Kolda", "Linguere", "Louga",

"Matam", "Mbour", "Medina-Gounass", "Pikine", "Podor", "Richard-Toll", "Rufisque", "Saint-Louis", "Sedhiou",

"Tambacounda", "Thies", "Tivaouane", "Touba-Mbacke", "Velingara", "Ziguinchor"};

**for**(**int** i=0; i< villeInSimulateur.length; i++)

**for**(**int** j=0; j< villeInSimulateur.length; j++){

**if**(i==j) **continue**;

System.*err*.println(villeInSimulateur [i]+" "+ villeInSimulateur [j]); }

**Détermination des villes en commun entre simulateur et données réelles**

Map<String,String> villeInSimulateur = **new** HashMap<String, String>() {

{

put("Bakel", "Bakel");

put("Banjul", "Banjul");

put("Dagana", "Dagana");

put("Dahra", "Dahra");

put("Dakar", "Dakar");

put("Diaobe", "Diaobe");

put("Diourbel", "Diourbel");

put("Farafenni", "Farafenni");

put("Foundiougne", "Foundiougne");

put("Janjanbureh", "Janjanbureh");

put("Kaffrine", "Kaffrine");

put("Kaolack", "Kaolack");

put("Kayes", "Kayes");

put("Kebemer", "Kebemer");

put("Kedougou", "Kedougou");

put("Kidira", "Kidira");

put("Kolda", "Kolda");

put("Linguere", "Linguere");

put("Louga", "Louga");

put("Matam", "Matam");

put("Mbour", "Mbour");

put("Medina-Gounass", "Medina-Gounass");

put("Pikine", "Pikine");

put("Podor", "Podor");

put("Richard-Toll", "Richard-Toll");

put("Rufisque", "Rufisque");

put("Saint-Louis", "Saint-Louis");

put("Sedhiou", "Sedhiou");

put("Tambacounda", "Tambacounda");

put("Thies", "Thies");

put("Tivaouane", "Tivaouane");

put("Touba-Mbacke", "Touba-Mbacke");

put("Velingara", "Velingara");

put("Ziguinchor", "Ziguinchor");

}

**private** **static** **final** **long** *serialVersionUID* = 1L;

};

Map<String, String> villeInDonneesReelles = **new** HashMap<String, String>() {

{

put("Bakel", "Bakel");

put("Bambey", "Bambey");

put("Banjul", "Banjul");

put("Bignona", "Bignona");

put("Birkilane", "Birkilane");

put("Bounkiling", "Bounkiling");

put("Dagana", "Dagana");

put("Dakar", "Dakar");

put("Diourbel", "Diourbel");

put("Fatick", "Fatick");

put("Foundiougne", "Foundiougne");

put("Gossas", "Gossas");

put("Goudiry", "Goudiry");

put("Goudomp", "Goudomp");

put("Guediawaye", "Guediawaye");

put("Guinee", "Guinee");

put("Guinee Bissau", "Guinee Bissau");

put("Guingineo", "Guingineo");

put("Kaffrine", "Kaffrine");

put("Kanel", "Kanel");

put("Kaolack", "Kaolack");

put("Kayes", "Kayes");

put("Kebemer", "Kebemer");

put("Kedougou", "Kedougou");

put("Kolda", "Kolda");

put("Koumpentoum", "Koumpentoum");

put("Koungheul", "Koungheul");

put("Linguere", "Linguere");

put("Louga", "Louga");

put("Malem Hodar", "Malem Hodar");

put("Matam", "Matam");

put("Mauritanie", "Mauritanie");

put("Mbour", "Mbour");

put("Medina Yoro Foulah", "Medina Yoro Foulah");

put("Nioro du Rip", "Nioro du Rip");

put("Oussouye", "Oussouye");

put("Pikine", "Pikine");

put("Podor", "Podor");

put("Ranerou", "Ranerou");

put("Rufisque", "Rufisque");

put("Rusfisque", "Rusfisque");

put("Saint-Louis", "Saint-Louis");

put("Salemata", "Salemata");

put("Saraya", "Saraya");

put("Sedhiou", "Sedhiou");

put("Tambacounda", "Tambacounda");

put("Thies", "Thies");

put("Tivaouane", "Tivaouane");

put("Touba-Mbacke", "Touba-Mbacke");

put("Velingara", "Velingara");

put("Ziguinchor", "Ziguinchor");

}

**private** **static** **final** **long** *serialVersionUID* = 1L;

};

**for**(String key : villeInSimulateur.keySet()){

**if**(villeInLiterature.containsKey(key))

System.*err*.println(key+" "+key);

**else**

System.*err*.println(key+" --");

}

**for**(String key : villeInDonneesReelles.keySet()){

**if**( ! villeInSim.containsKey(key))

System.*err*.println(key+" -");

}

**Détermination des données réelles pour les tronçons en commun entre les villes des données réelles et les villes du simulateur**

String villeCommun [] = {"Kaffrine", "Linguere", "Velingara", "Louga", "Pikine", "Kaolack", "Thies", "Sedhiou", "Kebemer",

"Tambacounda", "Foundiougne", "Kolda", "Rufisque", "Kayes", "Kedougou", "Diourbel", "Dakar", "Saint-Louis", "Matam", "Mbour", "Podor", "Dagana", "Bakel", "Ziguinchor", "Banjul", "Touba-Mbacke", "Tivaouane", "Fatick"};

HashMap<String,Integer> tronconInReelles = **new** HashMap<String, Integer>() {

{

put("Dakar:Guediawaye", 34);

put("Dakar:Pikine", 4);

…

}

};

**for**(**int** i=0; i< villeCommun.length; i++)

**for**(**int** j=0; j< villeCommun.length; j++){

String str = villeCommun[i]+":"+ villeCommun[j];

**if**(tronconInReelles.containsKey(str)){

System.*err*.println(villeCommun[i]+" "+ villeCommun[j]+" "+tronconInReelles.get(str));

}

}

**30/06/2015**

**Fatick** à ajouter dans les villes de CENTENAL **OK**

**07/07/2015**

Revoir le nombre de Hcarrier par rapport à la taille de HUMAN\_SUPER\_AGENT\_SIZE

La cellule de la ville de Farafenni n’est pas de type route (impossible d’y mettre ou d’y faire passer un véhicule) et il n’y a pas d’information pour cette ville dans le chrono sauf la date de sa création en 1960. La première partie est réglée

Pour le tronçon Mbour-Dakar, j’ai mis 979 au lieux de 9790

**29/07/2015**

**Les GNT et test :**



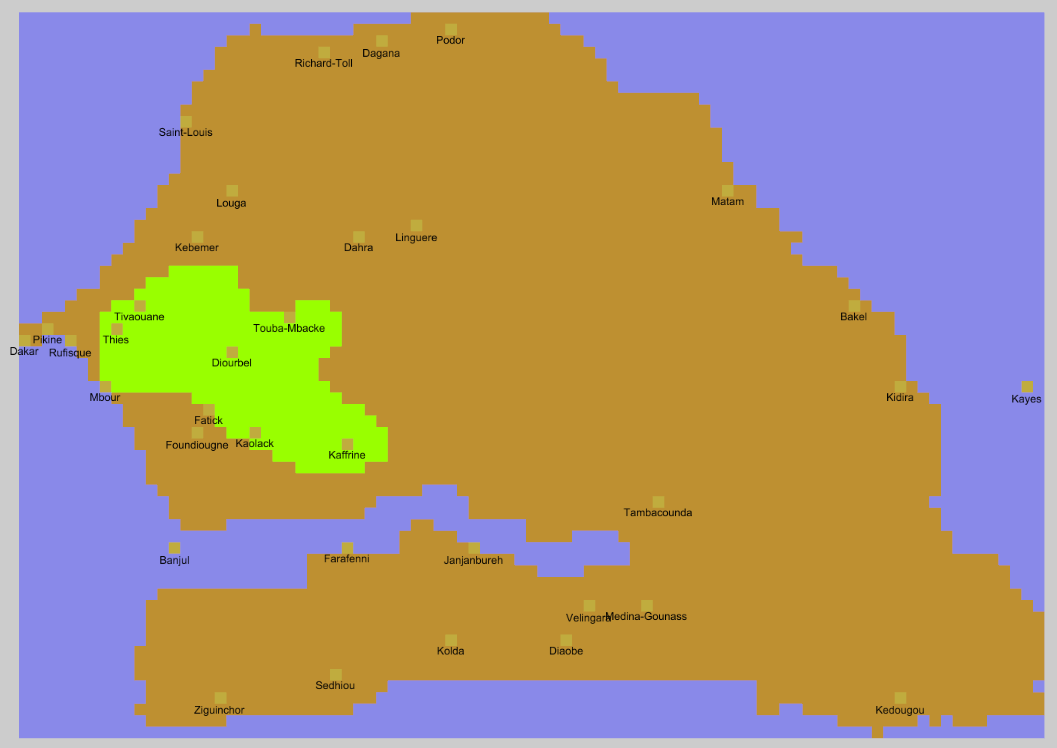
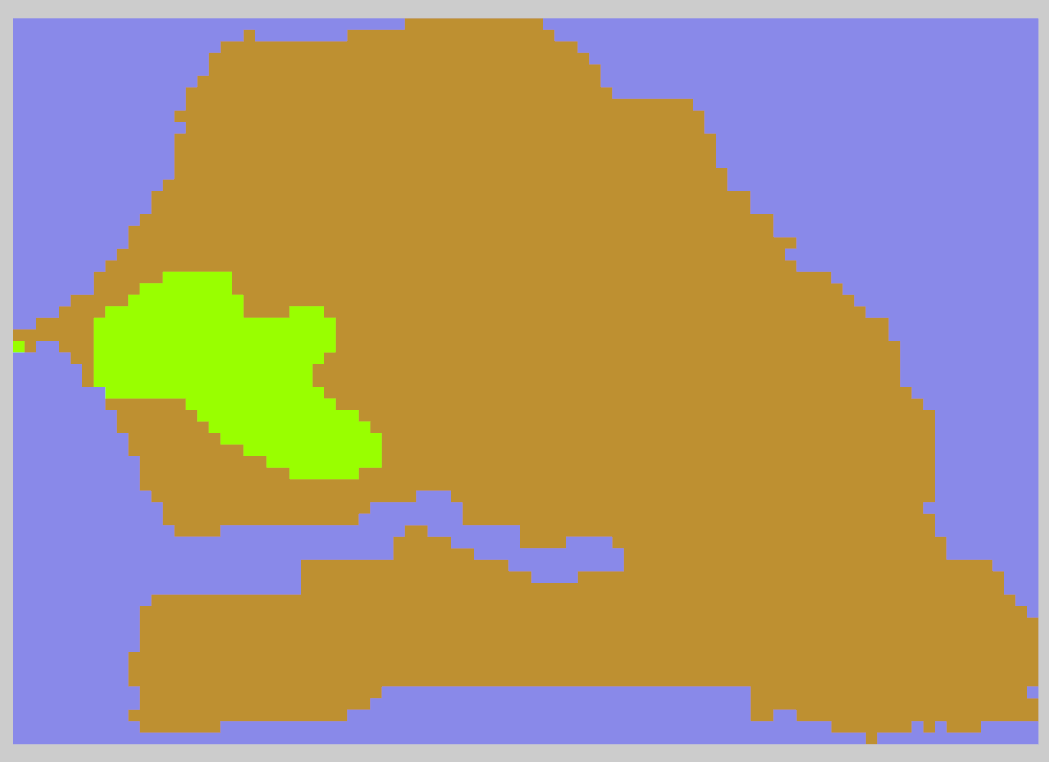
GNT-WEAK



GNT-WEAK 23 [Kidira, Tambacounda, Velingara, Diaobe, Janjanbureh, Kolda, Linguere, Dahra, Kaffrine, Sedhiou, Touba-Mbacke,

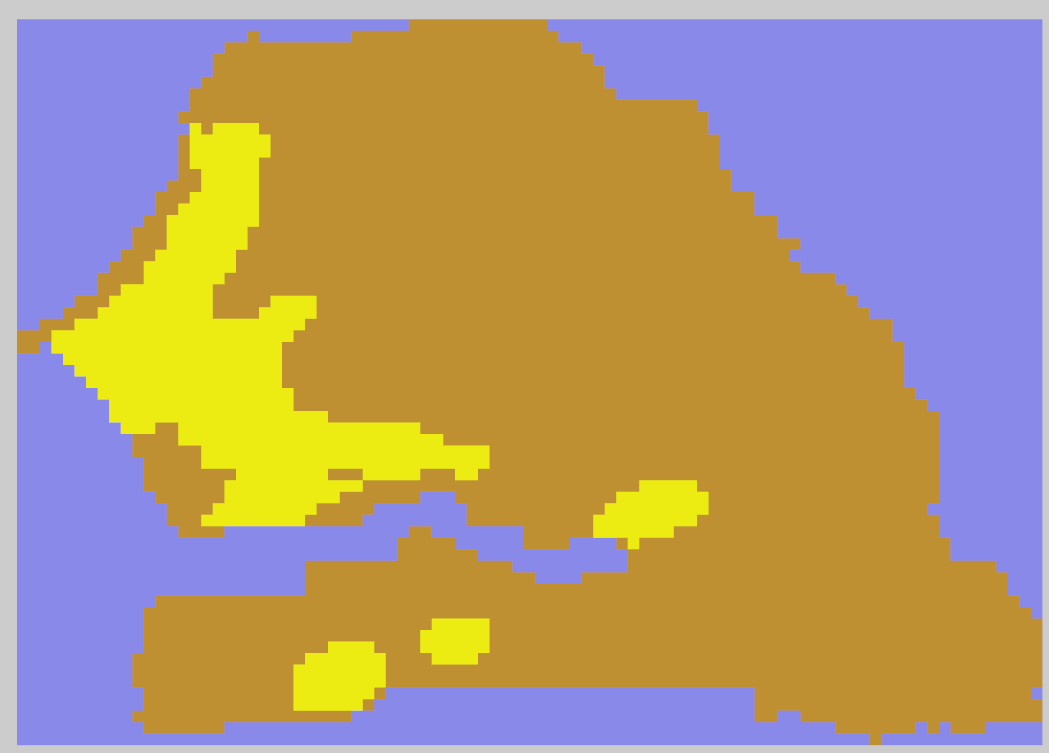
Kaolack, Louga, Diourbel, Ziguinchor, Fatick, Kebemer, Foundiougne, Tivaouane, Thies, Mbour, Rufisque, Dakar]

GNT-HEAVY



GNT-HEAVY 8 [Kaffrine, Touba-Mbacke, Kaolack, Diourbel, Fatick, Tivaouane, Thies, Dakar]

GNT-MEDIUM



GNT-MEDIUM 16 [Tambacounda, Kolda, Kaffrine, Sedhiou, Touba-Mbacke, Kaolack, Louga, Diourbel,

Fatick, Kebemer, Foundiougne, Tivaouane, Thies, Mbour, Rufisque, Dakar]

**03/07/2015**

**PAM pour la simulation des transporteurs (origine destination).**

J’ai ajouté un protocole complet pour la simulation des transporteurs (origine destination). Ainsi j’ai ajouté 4 feuilles que sont :

* **C\_ProtocolCentenal\_OriginDestinationTransport**
* **C\_HumanCarrier\_OriginDestinationTransport**
* **C\_InspectorCentenal\_OriginDestinationTransport**
* **C\_Vehicle\_OriginDestinationTransport**

Par rapport à ChooseProtocol j’ai ajouté :

* **else** **if** (C\_Parameters.*PROTOCOL*.equals(*CENTENAL\_OriginDestinationTransport*))

*protocol* = **new** C\_ProtocolCentenal\_OriginDestinationTransport((C\_RasterGraph) *groundManager*, context);

dans selectProtocol() de la class C\_ContextCreator

* Le mot **CENTENAL\_OriginDestinationTransport** dans la liste déroulante
* **parameters\_scenario\_CENTENAL\_OriginDestinationTransport.txt** dans SIMmasto\_0.rs

Pour sa destriction il suffura de les supprimer et de faire un ctrl+H et chercher " OriginDestinationTransport " pour s’assurer que tout est parti.

**Très important pour le Display**

Dans ce scénario CENTENAL\_OriginDestinationTransport, j’utilise le display du scénario de CENTENAL. Comment ?

Pour CENTENAL et pour CENTENAL\_OriginDestinationTransport les noms du protocole dans C\_Parameters contiennent tous "CENTENAL" :

C\_Parameters.PROTOCOL = CENTENAL

C\_Parameters.PROTOCOL = CENTENAL\_OriginDestinationTransport

Ainsi j’ai remplacé des C\_Parameters.PROTOCOL.equals("CENTENAL") par C\_Parameters.PROTOCOL.contains("CENTENAL")

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**01/08/2015**

Dans A\_ProtocolTransportation

/\*\*Elaborate its known city list with cumulated population\*/

**protected** **void** cityChoiceMap\_GravityByDistanceTest() {

**double** distanceBetweenCities;

Coordinate aCoord , bCoord;

System.*err*.println("Origines Destinations pop distance pop/distance");

**for**(C\_City currentCity : inspectorTransportation.getCityList()){

aCoord = currentCity.getCoord\_Umeters();

**for**(C\_City oneCity : inspectorTransportation.getCityList())

**if** (oneCity != currentCity) {// Does not choose the city where it stands

bCoord = oneCity.getCoord\_Umeters();

distanceBetweenCities = Math.*sqrt*(

(aCoord.x - bCoord.x)\*(aCoord.x - bCoord.x) +

(aCoord.y - bCoord.y)\*(aCoord.y - bCoord.y) );

System.*err*.println(currentCity+" "+oneCity+" "+oneCity.getHumanPopSize\_Uindividual()

+" "+distanceBetweenCities+" "+(oneCity.getHumanPopSize\_Uindividual()/distanceBetweenCities));

}

}

}

Dans step :

Date startDate = **null**;

// Elaborate its known city list with cumulated population

**try** {

startDate = C\_Calendar.*shortDatePattern*.parse("02/01/2013 00:00:00");

} **catch**(ParseException e) {

e.printStackTrace();

}

**if**(A\_Protocol.*protocolCalendar*.getTime().after(startDate)){

cityChoiceMap\_GravityByDistanceTest() ;

System.*err*.println("-----------------------------------");

System.*exit*(0);

}

**01/08/2015**

Remarque : Pikine n’est pas bitumée jusqu’en 2014