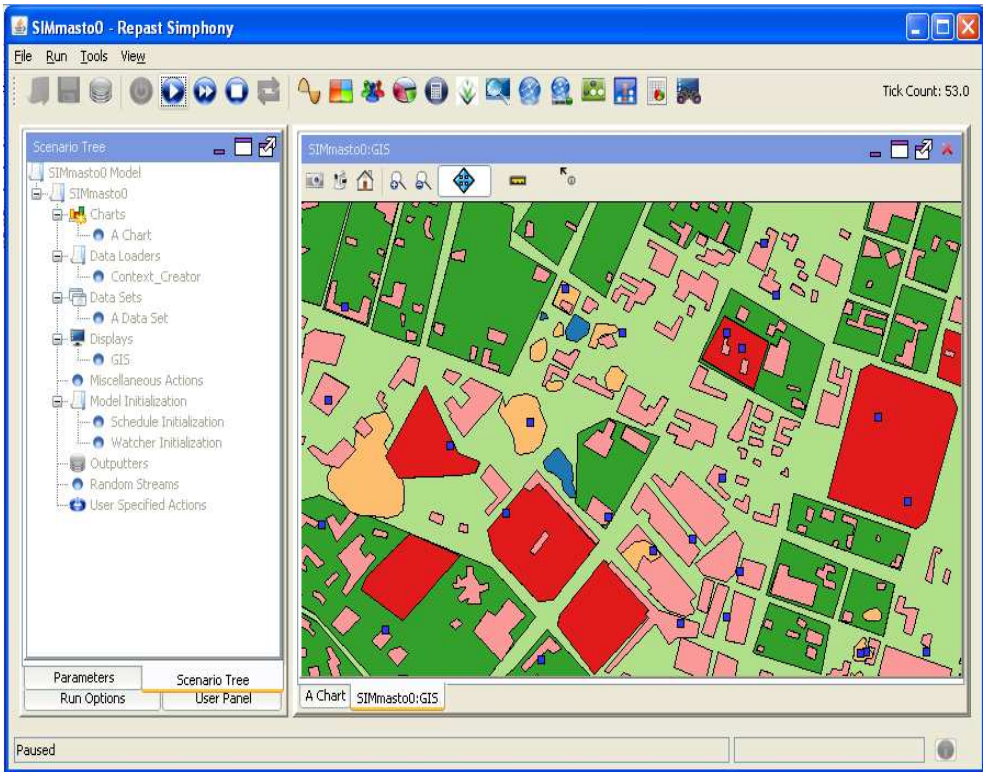
# Réalisation d'une chaîne de traitement conduisant à l'intégration de supports spatiaux et de données géoréférencées dans un simulateur multi-agents.

**Quentin BADUEL**

Rapport de stage




Stage effectué au Centre de biologie et de gestion des populations

Sous la direction de Jean Le Fur et Sylvain Piry

IUT de Montpellier, département informatique Stage du 12/02/09 au 24/04/09

# Remerciements

Je tiens à remercier l'ensemble des personnes qui ont rendu ce stage possible :

Je remercie mes maîtres de stage, Jean Le Fur et Sylvain Piry pour avoir proposé ce sujet qui m'a permis de découvrir des aspects de l'informatique que je n'avais encore jamais abordé et pour m'avoir soutenu lors de la réalisation de la chaîne de traitement ainsi que pour l'aide qu'ils m'ont apporté pour la rédaction. Grâce à eux, j'ai eu l'occasion d'apprendre des concepts algorithmiques et des méthodes et j'ai pu travailler dans de nombreux domaines différents (imagerie, bases de données, simulation, fichier de formes...).

Je remercie également M. Franck Dorkeld pour son soutien lors des installations des logiciels, M. Joannides pour ses conseils sur la rédaction du rapport ainsi que l'ensemble des professeurs de l'IUT pour leur enseignement au cours de ces deux années.

Pour finir, je tiens à remercier les trois autres stagiaires de l'IUT , Audrey Schneider, Xavier Rigollet et Pierre-Alain Sabatier pour la bonne ambiance que nous avons partagé durant ces onze semaines.

# Table des matières

# Glossaire

A

**ABMS** : Agent Based Modeling and Simulation

**Arcview/Arcgis** Logiciels de gestion des SIG semblables à Qgis

**Agent** : Dans le cadre de la simulation, un agent est considéré comme une entité autonome définie par ses caractéristiques et ses réactions. Un agent peut disposer d'une perception plus ou moins étendue de son environnement et planifier ses actions pour parvenir à ses buts

B

C

**CBGP**: Centre de Biologie et de Gestion des Populations

**CEFE**:Centre d'Ecologie Fonctionnelle et Evolutive

**CIRAD**: Centre de coopération International en Recherche Agronomique pour le Développement (crée en 1984)

**Clipping** : Le clipping est une méthode permettant d'extraire une partie d'une image.

**CSIRO** Commonwealth Scientific and Industrial Research Organisation. Organisme gouvernemental australien pour la recherche scientifique fondé en 1916

D

**DMS** (Degrés Minutes Secondes) : c'est un format de représentation des coordonnées géographiques, les valeurs sont des fractions d'angle. On exprime ainsi des positionnements par rapport aux parallèles et aux méridiens. C'est un des formats permettant de positionner un point en définissant sa latitude et sa longitude. Une minute d'angle représente 1/60 degrés et une seconde d'angle représente 1/60 minute d'angle.

**DD** ( Decimal Degree) : Il s'agit d'une version simplifiée du format DMS. Les coordonnées sont exprimées uniquement en degrés, mais on conserve des décimales. (on travaille donc en base 10)

E

**Eclipse** : c'est un environnement de programmation permettant de développer en Java. Il accepte de nombreuse extensions qui permettent de faciliter la programmation et d'intégrer des applications comme Repast Simphony.

**ESRI** :Environmental Systems Research Institute : C'est une firme informatique à l'origine du concept des logiciels SIG. Elle a créé de nombreux logiciels très répandus dont ArcGis. Elle est également à l'origine de certains standards ISO et de formats de données.

F

**Feature** : c'est une classe contenant les données d'un tuple d'une base de données. On l'utilise pour stoker les polygones des shapefiles avec les données qui leurs sont associées.

G

**Geography** : Dans Repast Simphony, une géographie est une projection permettant de définir un support virtuel pour des agents évoluant dans un environnement défini par un SIG.

**Géomatique**: La géomatique regroupe les outils et les méthodes permettant de représenter, et de manipuler des données géographiques.

**Geometry** : D'après le standard de JTS, une geometry est un enregistrement d'une forme géométrique constituée de un ou plusieurs points.

**Géoréférencement**: Le géoréférencement est une donnée associée à un élément (agent, carte, image...) permettant de situer cet élément sur le globe terrestre.

**Geotools** : C'est une librairie open source de Java permettant d'effectuer des traitements sur la géomatique et les SIG.

**GML** : Geography Markup Language : c'est un langage dérivé du XML permettant de stoker des informations géographiques ( par exemple les points définissants un polygone avec les données qui lui sont associées comme son géoréférencement et sa description).

H

I

**INRA**: Institut National de la Recherche Agronomique, organisme français de recherche en agronomie fondée en 1946

**IRD** : Institut de Recherche pour le Développement.

J

**Jointure** : Technique permettant de fusionner des éléments de deux tables de données en fonction d'attributs ayant des valeurs communes.

**JAI** : Java Advanced Imagery: bibliothèque Java spécialisée dans l'imagerie.

**Java** : Langage de programmation orientée objet.

**JTS** : Java topology suite : librairie de Java permettant la gestion de données géométriques.

K

**KML** :Keyhole Markup Language : c'est un langage destiné à l'affichage de données géoréférencées, il est compatible avec la plupart des logiciels SIG et avec les outils Google maps et Google earth.

L

**latitude/longitude** : la latitude et la longitude représentent des valeurs angulaires permettant de représenter un point sur la surface de la terre. La latitude varie de 0 degrés a l'équateur jusqu'à 90 degrés aux pôles, elle permet de tracer les parallèles (lignes horizontales parallèles à l'équateur utilisées pour les cartes) . La longitude quant à elle permet de donner une coordonnée en fonction des méridiens ( lignes verticales sur une carte joignant les deux pôles) par convention on compte 360 méridiens qui font le tour de la terre et dont l'origine se trouve être le méridien de Greenwich .

M

N

O

**OGC** : Open Geospatial Consortium: c'est un groupe international qui définit des standards garantissant l'interopérabilité des logiciels de géomatique.

P

**Projection** : dans Repast c'est un système servant de support et de repère pour situer les agents et leur permettre de se déplacer.

Q

**QGIS** : logiciel de traitement des SIG. Il a été conçu comme une plateforme permettant de faire des liens avec d'autres applications ( comme Grass et PostGis).

R

**Raster** : c'est une matrice de données. On peut s'en servir notamment pour stocker des pixels ordonnancés constituant une image.

**Repast** :Recursive Porous Agent : c'est le prédécesseur de Repast Simphony

**Repast-Simphony** (ne pas confondre avec symphonie) Plugin de Eclipse permettant la simulation multi-agents

S

**SVN** : Subversion: système de traitement de version.

**Shapefile**: (ou fichier de forme) c'est un format initialement développer par ESRI pour stoker des données issues des SIG.

**SIG** ou **GIS** : Système d'Informations Géographiques. Ce sont des outils permettant de stoker et de traiter des informations spatialement référencées.

**SRID :** IDentifiant du Système de Référence: il s'agit d'un nombre servant à identifier un système de coordonnées de référence particulier.

**Simulation multi-agents** : Une simulation multi-agents a pour but d'étudier le comportement d'une population d'agents dont les représentants agissent de façon autonome. Ils peuvent percevoir et interagir entre eux et avec leur environnement. On peut étudier les schémas d'évolution des populations en répétant les simulations et en mesurant l'impact de différents paramètres.

**SQL :** Structured Query Language : c'est un langage permettant d'interroger ou de manipuler des bases de données relationnelles.

T

**TIFF** : Tagged Image File Format : c'est un format d'image non compressé extrêmement flexible. Il est très utilisé pour la gestion des rasters car il supporte beaucoup de modifications. De plus il accepte les métadonnées. Ce qui permet d'attribuer par exemple un géoréférencement à l'image.

**Tuple** : ce terme désigne une collection ordonnée d'objet. Dans le cas d'une base de données un tuple représente une ligne d'une table dans une base de données.

U

**United States Departement of Agriculture** : l' USDA est le département de l'administration fédérale américaine en charge de la politique en matière d'agriculture et d'alimentation. Il fut créé en 1862

**UTM** (Universal Transverse Mercator : c'est un format permettant de représenter des coordonnées en mètres à n'importe quel emplacement sur le globe.

V

W

**WKT** : Well Know Text :format permettant d'enregistrer des formes géométriques dans un format texte ( exemple : « POINT(6 10) » représente un point défini par deux coordonnées entières.)

**WKB** : Well Know Binary : équivalent binaire du WKT. Il est interprétable par certaines fonctions SQL

**WLD** : World : il s'agit d'un format de fichier, il contient uniquement du texte et permet d'associer des coordonnées à une image. Pour géoréférencer « image1.jpg » on peut lui associer un fichier « image1.wld ».

X

**XML** : (eXtensible Markup Language). La norme XML est disponible à l'adresse www.w3.org/XML/. XML offre une façon pratique et standard de classer des données, afin de faciliter leur lecture, leur accès et leur manipulation. XML utilise une arborescence et une structure de balises identique à HTML

Y

Z

# Introduction

Dans le cadre de mes études en informatique en 2ème année d'IUT, nous sommes amenés à réaliser un stage en entreprise. La réalisation de ce stage a pour but de donner un aperçu de la vie professionnelle et de réaliser un projet correspondant à un besoin. C'est également une bonne occasion pour découvrir des outils et des techniques particulières pour l'organisation, la conception ou la réalisation de projets.

Un sujet de stage a été proposé pour réaliser la conception d'une chaîne de traitement visant à introduire des données dans une plateforme de simulation en Java.

Ce sujet s'inscrit dans le cadre d'un centre de recherche, et propose de concevoir des procédures qui seront réutilisables sur le long terme dans le cadre d'un projet scientifique.

Dans ce rapport, nous allons présenter le contexte dans lequel s'est déroulé le stage, avant d'expliquer les objectif attendus et de présenter les outils et les méthodes mises en place pour y parvenir. Finalement nous présenteront un bilan des activités réalisées.

Ce rapport est accompagné d'un dossier technique présentant la chaîne de traitement qui constitue le principal travail réalisé et un des attendus du stage.

# I. Contexte

## A. Le CBGP

Centre de Biologie et de Gestion des Populations

### *1. le cadre de recherche*

Le Stage s'est déroulé au Campus international de Baillarguet dans le bâtiment du CBGP.

Le CBGP(http://www1.montpellier.inra.fr/CBGP/) est un centre de recherche appartenant à l'Institut National de Recherche Agronomique(INRA). Le centre accueille les chercheurs de différents instituts de recherches ( IRD, CIRAD).

Ces différents organismes coopèrent en mettant en commun leurs moyens techniques autour de problèmes de biologie et de gestion des populations.

### *2. Le rôle du CBGP*

Le rôle du CBGP est de développer des études sur le comportement des populations animales (insectes, vers, rongeurs) pour permettre de dresser des modèles d'évolution.

Les résultats de ces recherches trouvent leurs applications dans plusieurs domaines, notamment l'agronomie et la santé humaine. en effet les modèles se révèlent utiles pour représenter l'évolution des populations de nuisibles qui menacent les cultures ou qui peuvent être des vecteurs de maladies.

Ces recherches permettent également de surveiller l'évolution de certaines espèces en fonctions des modifications de l'environnement provoquées par les changements climatiques, l'activité humaine, ou l'introduction de nouveaux prédateurs et parasites.

Le CBGP est également un centre de gestion de données, puisqu'il contient des bases de données relatives aux relevés effectués et aux particularités des espèces recensées.

# B. Le projet SimMasto

## *1.Présentation du projet*

La réalisation de ce stage s'est déroulé dans le cadre du projet SimMasto (http://www.mpl.ird.fr/ci/masto/index.htm).


Ce projet a été lancé à l'initiative de Monsieur Le Fur. Il a pour objectif de mettre en commun les résultats obtenus dans le cadre de recherches sur les rongeurs.

Le cœur du projet consiste à concevoir sur le long terme une plateforme de simulation multi-échelles permettant de regrouper les travaux des chercheurs concernés par ce thème.

Elle sera utilisée dans un premier temps pour effectuer des simulations particulières pour les différents pôles d'étude.

Le Projet SimMasto a pour ambition de permettre de représenter les résultats des recherches sur les rongeurs à tous les niveaux. Ainsi, les connaissances acquises par des chercheurs dans un domaine pourront être utilisées par les chercheurs d'un autre domaine.

## *2. Les contraintes à respecter*

Cette plateforme doit être très souple et très robuste car les applications seront très différentes. Par exemple, on peut vouloir étudier l'évolution d'une population de rongeurs dans un terrain défini de l'ordre du kilomètre, ou étudier la propagation des parasites entre les populations de rongeurs sur des zones couvrant plusieurs milliers de kilomètres.

Les différents niveaux de simulation sont les suivants :

- Clade : étude de l'évolution des différentes branches taxonomiques. Pour étudier leur co-évolution et co-adaptation à l'échelle d'un pays, d'un continent.
- Population : étude de l'évolution d'une population dans un biotope donné. On peut étudier une population à l'échelle d'une région, et faire varier le biotope en fonction des saisons
- Communauté on étudie les interactions des individus d'un groupe , leur évolution dans leur écosystème.
- Individuel : on étudie le comportement d'un individu particulier en fonction de son environnement (ressources disponibles, densité de rongeurs déjà présent...)

- Inter-individuel : il s'agit de l'étude de la relation hôte-parasite. On étudie la propagation des parasites en fonction des activités des hôtes (contamination, déplacement, transmission...)
- Intra-individuel : Cette approche permet d'étudier le comportement d'un agent en fonction de son état. (stress, maladie...)
- Sub-cellulaire : Ce dernier aspect permet d'étudier l'évolution génétique des rongeurs. On observe la progression, la régression ou l'apparition de gènes dans une population.

Pour chaque niveau de simulation, il faudra également prévoir des échelles spatiales et temporelles différentes.

## *3. Résultats attendus*

Cette mise en commun des informations à travers des simulations permettra de partager les connaissances de manière dynamique. Ce support permettra de partager des informations et de les distribuer en complément des travaux publiés sur les sites Web et les rapports.

Le stage réalisé s'inscrit dans le cadre de récupération de données dans la perspective de créer une telle plateforme. De plus la mise en place d'une structure de gestion permettant des les utiliser et de les manipuler de manière générique permettra de concevoir un squelette de cette plateforme.

# II. Présentation du stage

## A. Le cahier des charges

### *1. Objectif*

L'objectif de ce stage est de réaliser une chaîne de traitement pour utiliser des informations spatialement référencées dans un simulateur.

L'originalité de ce stage réside dans le fait que la chaîne n'est pas prédéfinie et à réaliser mais à concevoir pas à pas dans une logique d'amélioration continue. En effet il s'agit de concevoir des étapes permettant l'intégration de données dans un simulateur. Cependant les étapes permettant l'acquisition des ces données et les traitements nécessaires ne sont pas connus.

### *2. Principe de l'élaboration itérative*

Pour construire la chaîne de traitement, il est nécessaire de mettre en place un certain nombre d'étapes.

C'est lors de la définition de ces étapes que l'on va pouvoir s'apercevoir des différentes possibilités et choisir celles qui nous semblent appropriées. Puisque l'on ne connaît pas à l'avance les formats à utiliser et les méthodes à appliquer,il faut donc procéder de manière itérative et redéfinir les besoins et les résultats attendus après la réalisation de chaque étape. Ainsi, on peut avoir à définir des étapes intermédiaires permettant de transformer les données pour qu'elles puissent être traitées dans l'étape suivante.

Tout au long de la chaîne de traitement, on peut redéfinir l'objectif de chaque étape et les contraintes que l'on s'impose pour essayer de se rapprocher de la perspective de départ qui est de pouvoir utiliser des informations dans une simulation.

De plus, on peut à tout moment choisir de créer des dérivations dans la chaîne de traitement pour avoir plusieurs façons de procéder. Il faut alors adapter les modules de la chaîne pour obtenir un résultat cohérent.

### *3. Contraintes de réusabilité et d'efficacité*

La réalisation de la chaîne de traitement doit permettre de s'adapter à différents cas de figures. Il faut prendre en compte les différents formats existants et vérifier qu'ils puissent être utilisés.

De plus la rédaction de chaque étape est nécessaire pour permettre à une personne extérieures de comprendre et de reproduire chaque étape. Il faut donc penser que sur le moyen ou long terme, les procédures de cette chaîne de traitement seront utilisées pour intégrer différents supports dans différentes simulations.

La contrainte de respecter la réutilisabilité des éléments est assez forte, mais il faut également chercher les solutions les plus économiques en terme de ressources car les simulations doivent pouvoir porter sur de très nombreuses itérations. Le temps de calcul est alors très important.

## *4. Résultats attendus*

La réalisation de ce stage doit aboutir à trois résultats principaux.

Le premier résultat est la structure de la chaîne de traitement proprement dit comportant les étapes à réaliser, les branchements optionnels tenant compte des différents cas particuliers qui peuvent être rencontrés. La chaine doit aller de la donnée spatialisée en entrée jusqu'à la simulation de l'évolution d'agents mobiles dans cet espace.

Le second est l'application Java permettant de visualiser les éléments intégrés dans la simulation. Il faut donc mettre en place d'une structure de programmation et des interactions entre les différents éléments. Cette application permettra d'illustrer les résultats des étapes de la chaîne de traitement et servira d'exemple pour programmer des simulations de manière génériques quelque soit le type de support choisi.

Le troisième résultat attendu est le rapport sur la chaîne de traitement. en effet il faut laisser une trace des procédures réalisées pour permettre à un utilisateur d'utiliser ses données comme support pour une simulation. Il faut également détailler les problèmes qui peuvent apparaître et expliquer comment les résoudre.

# B. Les outils

## *1. Les systèmes multi-agents*

Le concept de système multi-agents se rapproche du concept d'objets. Les agents sont des entités qui possèdent des caractéristiques (attributs) et qui sont capables de réaliser des actions (méthodes).

En revanche, ils sont plus que de simples objets puisqu'ils agissent généralement en fonction de buts. Si il ne savent pas exécuter certaines fonctions, ils peuvent demander l'assistance d'un autre agent spécialisé. Les agents sont donc capable de coopérer.

On utilise les systèmes multi-agents dans le cadre de l'intelligence artificielle. Indépendamment, chaque agent possède une « intelligence » (capacité de délibération) propre plus ou moins restreinte. Mais les interactions mises en place avec tous les autres agents peuvent permettre la réalisation de taches complexes. On parle alors d'intelligence collective et d'intelligence distribuée.

Dans le cadre d'une simulation au sein du projet SimMasto, on va utiliser le concept d'agents pour représenter une population de rongeur. Chaque individu est représenté par un agent.

Ainsi, on va pouvoir étudier à partir des décisions de plusieurs individus autonomes l'émergence de comportement global des populations de rongeurs (leur nombre, leur déplacement en fonction du terrains...).

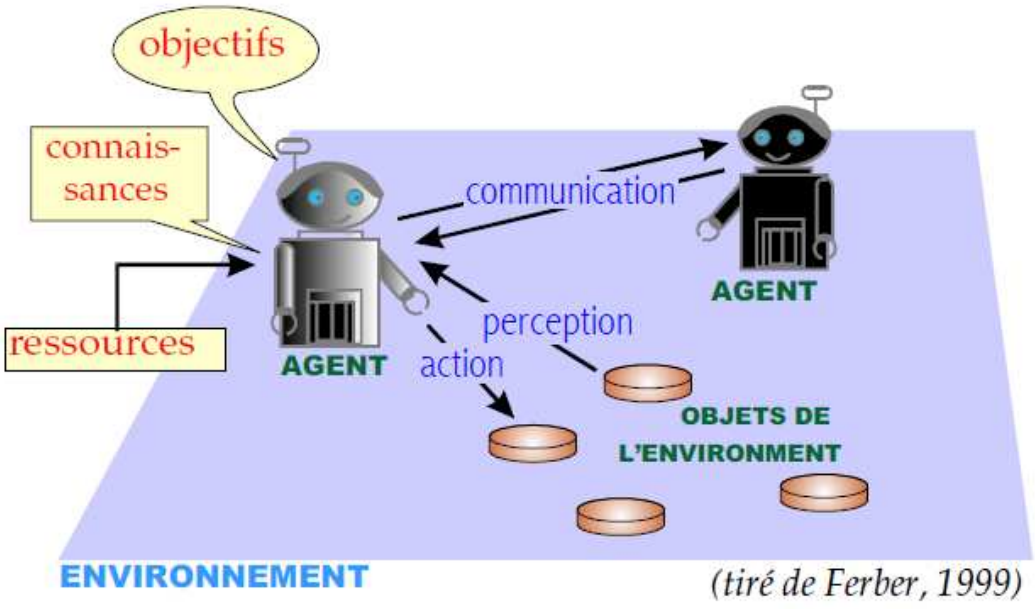



Schéma du comportement d'un agent

## *2. Les Systèmes d'informations géographiques (SIG)*

a) Les SIG en général

Les systèmes d'informations géographiques sont anciens, on estime que leur première utilisation date de 1854. La nécessité de pouvoir représenter des informations a encouragé le

développement de logiciels spécialisés qui sont apparus dès 1950.

Dans un premier temps, leur utilité était principalement réduite à la cartographie, mais au fur à mesure, ils ont évolué pour permettre de faire des correspondances entre des bases de données et des représentations géographiques.

Ils permettent maintenant d'enregistrer, de représenter et de manipuler des données avec une position géographique, on s'en sert dans de nombreux domaines.

b)Exemple d'application

Ils sont très utilisé pour représenter des éléments sur des surfaces particulières. Par exemple, on peut représenter les éléments d'un terrain avec une forme dont la couleur va dépendre de la végétation. On peut également combiner différentes données. Si on étudie les données d'élévation, les donnée de végétation et d'humidité, on peut par exemple retrouver des sources souterraines.

Alors que l'on ne peut pas déduire facilement les informations qui nous intéressent depuis les tuples d'une base de données, les SIG permettent par une simple représentation visuelle de retrouver rapidement toutes les portions voisines d'une portion défectueuse et procéder à des détournements.

La visualisation des données permet d'établir des relations qui ne seraient pas évidentes autrement. Il suffit ensuite d'utiliser des requêtes spatiales pour obtenir rapidement des informations sur une zone donnée.

On peut utiliser les SIG dans des contextes très différents, par exemple pour représenter une carte ou les bâtiments sont colorés en fonction de leur utilité, leur propriétaire, et toutes sortes de données.

c)Utilisation dans le cadre du projet SimMasto

Dans le cadre du CBGP, on utilise des simulateurs multi-agents pour représenter des rongeurs ( représentés sous la forme de points) qui possèdent différentes caractéristiques, des plus simples (vitesse, champs de vision...) au plus complexes (code génétique, infection par un parasite particulier...).

Par ailleurs, on va faire évoluer ces rongeurs en fonction de leur terrain. Celui ci est composé de différentes parcelles possédant des attributs particuliers (nourriture disponible pour les rongeurs, danger...).On va donc utiliser les SIG pour associer des données aux différents éléments du terrain

Les fichiers contenant des informations spatialement référencées peuvent être représentés sous différents formats. Nous allons nous intéresser à deux formats particuliers très utilisés : les shapefiles et les fichiers rasters

## Compléments sur les formats raster et vecteur

Raster et vecteur sont des modes de représentation d'une image, il est possible de convertir une image d'un format à l'autre mais des données peuvent être perdues.

Pour faire simple un fichier de raster peut être vu comme une grande matrice de données (ces données peuvent représenter n'importe quoi). Et les vecteurs sont composés d'un ensemble d'éléments distincts. Ils ont chacun leur forces et leur faiblesses, et on utilise l'un ou l'autre selon

l'objectif que l'on cherche à atteindre.

Dans les SIG cependant, on préfère généralement travailler avec des vecteurs

## Les rasters

Les raster fonctionnent comme des matrices de données. Souvent on s'en sert d'image en divisant l'espace manière régulière (généralement par pixel). Ce mode de représentation est facile à utiliser puisque la structure est simple d'accès (les pixels sont ordonnés dans une matrice, et donc accessibles par des coordonnées).

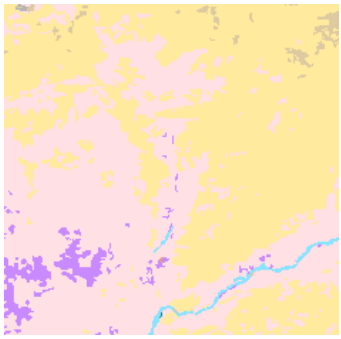


*Image raster représentant les types de végétation; extrait du landcover GLC2000*

Exemple du codage RGB ou un pixel est représenté par trois composantes: on peut donc utiliser 3 raster de même dimension dont les pixels ont des valeurs entre 0 et 255 pour recomposer une image couleur.

C'est un format assez souple qui correspond bien pour le traitement d'image, puisque toutes les données sont à la même échelle (le pixel) et que les opérations de traitement d'image peuvent être appliquées directement sur la matrice.

Ce format est bien adapté pour le traitement de valeurs continues (comme la température).

- En revanche ce format présente aussi quelques défauts, puisque les objets présents dans le raster ne sont pas différenciés (tout est inclus dans une couche de pixel).
- L'extraction d'élément reste possible en effectuant des tests sur les pixels, mais l'utilisation des vecteurs est souvent plus simple et plus rapide.

## Les vecteurs

Ils donnent un mode de représentation géométrique des objets, c'est à dire que tous les éléments sont représentés par des points, des lignes ou des surfaces (polygones).

Ce format présente un grand intérêt pour les SIG puisqu'il permet d'individualiser les objets et donc de définir des attributs. On peut obtenir les coordonnées et les dimensions de chaque élément avec précision.

On peut ainsi établir une correspondance entre des objets réels et leur représentation géométriques.

On peut accéder à des informations en effectuant des recherches sur les attributs des différents éléments. Le format enregistre les contours des objets et leurs attributs, il est souvent plus léger (on ne repère pas chaque pixel mais des zones limitées par des points).

Il est possible de comparer les relations entre les coordonnées et les dimensions des différents éléments pour effectuer des requêtes spatiales (intersection, union...).

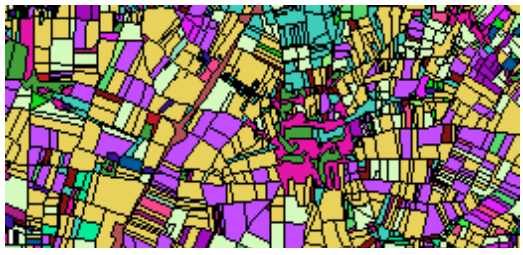


*image au format vecteur*

- En revanche l'organisation des éléments dépend d'un index, il est donc plus difficile d'effectuer des traitements d'image sur des représentations en vecteur.

.

## 3. Les logiciels utilisés

Durant le déroulement du stage, nous avons utilisé plusieurs outils qui nous ont permis de réaliser les différentes étapes. Voici une liste des logiciels et progiciels utilisés avec une description de chacun d'eux.

a)les outils de traitements SIG:

• Qgis est un logiciel de gestion de SIG, il permet d'importer des fichiers de différents formats, (en particulier les shapefiles). Ses fonctions de base permettent ensuite de visualiser les données en superposant les couches et en les coloriant selon la valeur d'un attribut particulier. La gestion des SIG inclut également un certain nombre d'actions possibles pour la manipulation des données (opération sur les tables d'attributs) et permet de modifier le modèle (récupération de zones grâce au des opérations géométriques), et de changer le système de référencement, ou de modifier les coordonnées.


•ArcGis: c'est un logiciel semblable à Qgis à ceci près que arcgis est une version professionnelle et propose davantage de fonctions


•Google Earth: bien qu'il soit souvent utilisé pour la cartographie, Google earth possède les fonctions permettant de lire et de créer des SIG simples.


•Grass : C'est un logiciel de traitement de SIG de conception modulaire, il propose un grand nombre de traitement et l'appel aux différents modules peut se faire en console, ce qui permet de créer des scripts permettant d'automatiser certains traitements répétitifs.

Bien que Grass soit une application indépendante, elle peut être intégré à Qgis en tant que Plugin et dispose alors d'une interface un peu plus accessible. Cependant tous les traitements ne sont pas toujours possibles depuis l'interface graphique et il est parfois nécessaire d'utiliser la console.


•Postgres/PostGis :Postgres est un SGBD (système de gestion de base de données), qui permet de créer des tables et d'effectuer des requêtes SQL.A l'aide d'un utilitaire, on peut convertir les données d'un fichier shapefile en une table de données contenant tous les éléments géométriques du shapefile avec les attributs associés. PostGis quant à lui est un greffon à ajouter sur des tables de données Postgres, il permet de faire appel à l'intérieur des requêtes SQL à des fonctions de requêtes spatiales. PostGis permet donc d'effectuer rapidement des modifications sur un fichier shapefile.

Bien que la combinaison de fonctions géométriques avec le langage SQL soit déroutante, il s'avère être un outil très puissant pour la gestion SIG.

Pour pouvoir utiliser ces outils, il faut toutefois installer un serveur Postgres, créer au moins une base de donnée puis configurer les droits des utilisateurs. Il faut ensuite ajouter les fonctions des PostGis à cette base. Pour finir les utilisateurs doivent pouvoir se connecter, ( il faut connaître l'hôte du serveur, les identifiants, les mots de passes et les ports...

Une fois la connexion avec la base configurée, Qgis permet d'accéder directement aux tables et au vues crées et les interpréter comme des shapefile ( a condition qu'il existe une clé primaire explicite, cad le gid).Pour finir on peut enregistrer les tables importées dans Qgis en tant que shapefile.

•The Gimp : (*GNU Image Manipulation Program* ) il s'agit d'un logiciel libre de traitement d'image. Il permet de nombreuses opérations et propose une large gamme d'outils.

Dans le cadre de la chaîne de traitement, on l'utilise pour ses outils de sélection , ses opérations sur les contrastes et sa capacité à fonctionner en utilisant différentes couches de calques.

b)Outils utilisés pour la partie programmation :


•Eclipse : Il s'agit d'un environnement de programmation pour le langage Java. Il fonctionne comme une plateforme acceptant de très nombreux plugin permettant une grande flexibilité dans la conception et dans la programmation. Il dispose d'un outils de mises à jour capable de reconnaître des sites comme sources de mise à jour possibles. Le grand nombre de plugin disponible fait d'Eclipse un environnement de programmation très adaptables.


•Repast Simphony: Ce logiciel est libre et le code source est disponible. Repast Simphony est le successeur de Repast (REcursive Porous Agent Simulation Toolkit).

Le projet a été créé à l'université de Chicago, puis à été repris par des organisations comme « Argonne National Laboratory ». Actuellement Repast est développé par une équipe de volontaire : ROAD : Repast Organization for Architecture and Development).

Plutôt qu'un logiciel prêt à utiliser, Repast s'apparente plus à une bibliothèque Java mettant à disposition une structure de classes et une interface permettant de représenter la simulation que le programmeur a définit grâce aux outils de Repast et d'autres bibliothèques.

Pour créer une simulation, il faut définir tous les éléments (contexte, projection, agent, interactions...) à l'intérieur de classes Java, mais la structure générale est très adaptable et permet de programmer des simulations très différentes en adaptant le simulateur au besoin.

En revanche puisqu'il est en développement permanent et que ses fonctions sont très ciblées, il est difficile de trouver de la documentation et il est parfois nécessaire de connaître certaines classes du simulateur pour pouvoir programmer certains éléments (par exemple l'affichage ).


•Geotools : Biobliothèques Java permettant la gestions de données géoréférencées, on l'utilise principalement pour lire les shapefile et pour la gestion SIG.


•JTS : Java Topology Suite, librairies Java développée par Vivid Solution permettant une gestion des formes géométriques. Utilisé pour les traitement sur les objets en mode de représentation GIS dans Repast.

# III. Déroulement

## A. La chaîne de traitement (recherche, conception, et utilisation).

### *1. Analyse de l'existant*

La réalisation s'est déroulée dans deux domaines différents : celui de la géomatique et celui de la simulation. Bien que la chaîne de traitement s'inscrive dans un domaine particulier, certains des problèmes abordés sont récurrents et ont déjà été résolus. Il existe des outils permettant de faciliter certaines taches, en proposant des fonctions de traitements que ce soit pour la gestion d'informations spatialement référencées ou pour la mise en place d'une simulation.

L'univers des SIG est très vaste, et il existe des dizaines d'outils déjà existants.

Du fait du caractère international du traitement des données géographiques et des systèmes de représentation, il existe des milliers de formats pour représenter une carte (le format change en fonction du lieu et de l'échelle pour prendre en compte la courbure de la terre). De plus il existe un grand nombre de fonctions existantes pour leur manipulation. On retrouve des fonctions de traitement géométrique (union, intersection, différence symétrique...) et des fonctions de bases de données (jointure). Mais ces fonctions sont implémentées différemment selon les logiciels et ne produisent pas toujours le même résultat.

Les fonctions d'union et d'intersection par exemple n'agissent pas toujours dans le sens mathématique de l'union et de l'intersection d'ensembles.

Le premier objectif était donc de trouver des standards et de définir les fonctionnalités pour effectuer des traitements particuliers. Même si les outils existent, il faut comprendre leur fonctionnement et reproduire une procédure en fonction de différents cas.

Les méthodes existent également, mais il faut faire des recherches pour retrouver celles qui peuvent nous permettre de modifier les paramètres qui nous intéressent et les comprendre pour pouvoir les adapter et les enchaîner correctement.

Pour la partie concernant le simulateur, il existe des exemples de codes sources de simulations mettant en place de nombreux éléments différents, mais ils sont tous orientés dès le début vers un objectif précis. La conception d'une simulation particulière (et non la recopie d'une déjà existante) demande quelques recherches pour permettre de s'adapter au simulateur.

Dans le cadre de la Simulation SIG, bien que certaines fonctions de bases de traitement existent et soient intégrées par le simulateur, il faut néanmoins faire des recherches pour comprendre leur fonctionnement (bibliothèques JTS et Geotools).Il faut ensuite programmer en s'aidant de ces fonctions.

## 2. Les étapes

Après avoir passé un premier temps pour étudier les différentes possibilités, il est apparu que certains aspects de la chaîne de traitement seraient incontournables. Ce qui a permis de dégager des axes principaux de recherches :

1. Pour commencer, il faut concevoir et tester toutes les étapes permettant de récupérer des données spatialement référencées (il peut s'agir du type de terrain, de l'hydrométrie etc , représentées sous la forme d'une carte).
2. Une fois ces données récupérées, on peut définir les traitements nécessaires pour rendre ces données intégrables par le simulateur en les convertissant dans un format particulier.

   Il faut également tester la cohérence et la pertinence de certaines données et mettre en place des procédures de traitement pour effectuer si nécessaire des modifications et des conversions.
3. Le temps de calcul étant important, et les données en entrée étant parfois volumineuse, il faut mettre en place des procédures permettant de réduire ou de simplifier les fichiers de données.
4. Après que les données soient transformées dans un format acceptable pour la simulation, il faut programmer les classes Java permettant d'utiliser ces données pour créer les objets qui vont constituer la « matière première » de la simulation (les éléments représentant un terrain).
5. Il faut ensuite définir des classes Java permettant la gestion de ces éléments en respectant des contraintes d'encapsulation assez forte, puisqu'il faut permettre aux agents d'interagir entre eux et avec leur environnement de manière générique.

   Il faut rendre possible des modifications sur des paramètres tel que l'échelle spatiale, l'échelle temporelle, les caractéristiques du support ou des agents sans avoir à reprogrammer les autres classes Java.

   Pour rendre le code adaptable, le support de la simulation ne doit pas être défini à l'avance. On peut ainsi travailler aussi bien avec un espace continu qu'avec une grille sans modifier les classes définissant les agents.

   Lors de la programmation il faut également essayer de rendre le code le plus optimisé possible. Lors d'une simulation, les calculs sont très nombreux et concernent chaque agent à chaque pas de temps. Il faut donc s'assurer de réduire les parcours.
6. Pour finir, il faut synthétiser les différentes étapes et détailler tous les éléments de la chaîne de traitement, les tester et proposer différentes façons de procéder selon la situation.

   Lors de la rédaction il faut également présenter et expliquer le fonctionnement de tous les outils utilisés.

   Les éléments d'information doivent permettre d'anticiper les sources d'erreurs en précisant les précautions à prendre lors de la réalisation de certains traitements et de la mise en place de la simulation elle même.

## 3. Conception du squelette de la chaîne de traitement

Avant de commencer à rédiger la chaîne de traitement, il a fallu un certain temps pour installer tous les outils et se familiariser avec les options proposées.

Lors des deux premières semaines, j'ai donc travaillé parallèlement sur le fonctionnement du simulateur et sur la manipulation de fichiers shapefile existants à l'aide des logiciels de gestion de SIG .

Repast Simphony ne permettant pas d'intégrer directement des fichiers au format shapefile, j'ai fait des recherches sur les librairies proposées par JTS et Geotools permettant de gérer les données d'un fichier de forme. Grâce à leur Javadoc, j'ai pu manipuler les fonctions de Geotools pour intégrer les données du shapefile comme éléments de support de la simulation et les afficher dans une projection adaptée.

Dans un deuxième temps, je suis parti d'un fichier image représentant une vue aérienne de champs et j'ai cherché à le convertir dans un format adapté pour l'implanter dans le simulateur.

Une fois que nous avons définies les méthodes permettant d'utiliser des fichiers shapefiles dans le simulateur, et celles permettant de générer des shapefiles à partir d'images aériennes, nous avons pu commencer à complexifier la chaîne.

## 4. Mise en place des traitements intermédiaires

Après la réalisation de la structure minimale de la chaîne de traitement, il a fallu mettre en place des traitements pour rendre la simulation réaliste.

1. Il faut faire en sorte que les distances représentées dans le simulateur soient cohérentes. Si les agents se déplacent et « raisonnent »en mètres, il ne faut pas que le plan sur lequel ils évoluent aie des coordonnées en degrés.
2. La taille des fichiers de forme n'étant pas limitée, il a fallu mettre en place des procédures de clipping pour extraire des sous parties d'un fichier de formes.
3. De plus, on utilise les données du fichier pour retrouver la nature et les caractéristiques du terrain sur lequel l'agent évolue. Il a donc fallu mettre en place des procédures permettant de récupérer ces attributs et les modifier.

Après avoir réussi l'implantation d'un shapefile. Nous avons repris la procédure en amont, pour créer un fichier shapefile depuis une simple image satellitaire.

La première méthode que nous avons imaginé pour transformer l'image en shapefile consistait à utiliser un logiciel de traitement d'image (Gimp) pour isoler les différentes zones de l'image à l'aide de l'outil de sélection par couleur, et démarquer les différentes catégories de terrain en utilisant une couleur unique pour chaque. Une fois la carte entièrement recouverte par des couches de couleur représentant les différents éléments, il devient possible d'utiliser un traitement permettant de vectoriser les différents éléments par reconnaissance des zones de couleurs.

Cette solution, bien que fonctionnelle a cependant amené des inconvénients, puisque les fichiers ainsi générés étaient trop « lourd » : chaque élément était défini par plusieurs centaines de points et ralentissait le fonctionnement du simulateur

Il a donc fallu mettre en place une autre procédure consistant a créer manuellement les polygones à l'aide d'un logiciel SIG en superposant les polygones créés à la carte réelle.

Cette solution marche très bien mais il y a des écueils à éviter. il faut géoréférencer l'image, puis gérer la superposition des polygones. En effet pour représenter un arbre dans un champ, on ne peut pas « dessiner » le champs avec un polygone en dessinant des trous là ou il y a des arbres.

Outre la mise en place de cette nouvelle procédure, nous avons conservé l'ancienne qui peut être utile dans le cas où la carte étudiée présente un contraste fort. Par ailleurs nous avons trouvé des méthodes permettant de simplifier les fichiers ainsi générés.

Nous avons ensuite repris le code du simulateur pour essayer d'avoir la meilleure encapsulation possible tout en respectant les contraintes posées par les logiciels et les besoins de la simulation.

Lorsque la chaîne de traitement à été suffisamment aboutie, nous avons cherché à l'améliorer et à optimiser les résultats.

## *5. Structure et rédaction de la chaîne de traitement*

A la fin du stage, l'élaboration de la chaîne a été obtenue. Le résultat est représenté ci-dessous sous formes d'organigrammes. Les modules et procédures sont décrits dans le rapport technique joint.

La réalisation de la chaîne de traitement ne s'est pas effectuée de façon linéaire. Nous sommes parti de la fin de la chaîne pour savoir quels éléments le simulateur pouvait interpréter.

Nous avons effectué des tests sur les fichiers existants dans différents formats. Une fois les objets en entrée du simulateur mieux définis (fichier shapefiles et rasters), nous avons repris le début de la chaîne de traitement pour mettre en place des méthodes permettant de créer des fichiers dans ces formats.

Nous avons ensuite établi des procédures de traitement permettant d'adapter ces fichiers aux besoins de la simulation (coordonnées, échelles, superposition des éléments...).

En situation réelle, le lecteur n'est pas obligé de réaliser toutes les étapes, il peut vouloir connaître la méthode permettant de faire un traitement précis. Nous avons donc opté pour une structure en modules indépendants. Une fois toutes les étapes et les variantes possibles mises en place, nous avons ordonnancé ces modules de manière logique en respectant les dérivations possibles.

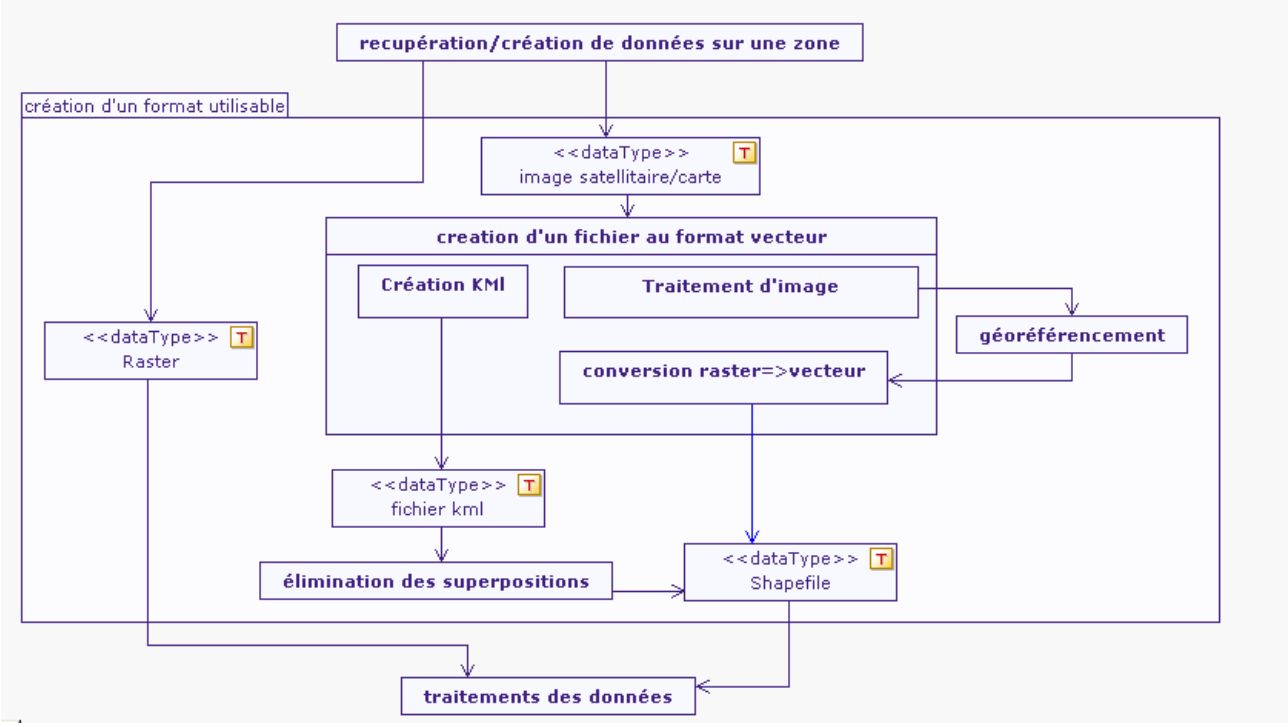



*Illustration 1: organisation des modules pour la création d'un format utilisable par le simulateur*

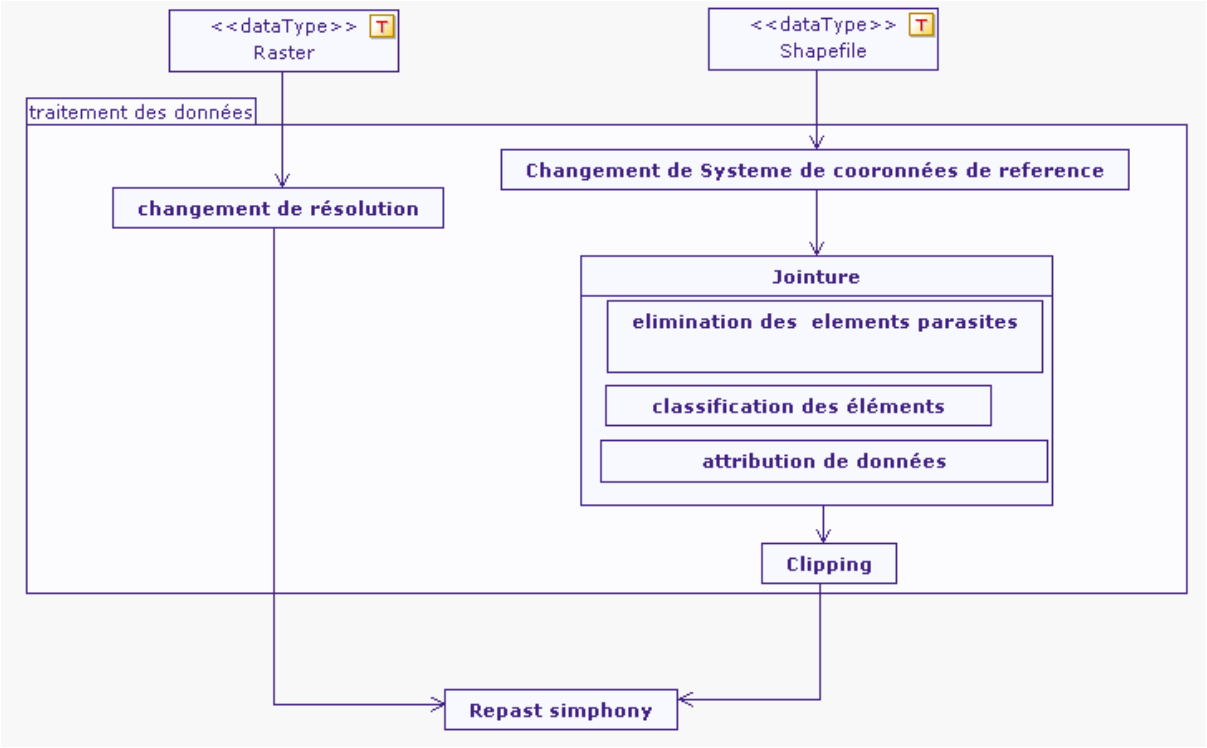



*Illustration 2: Organisation des modules permettant d'effectuer des traitement sur les formats utilisés*

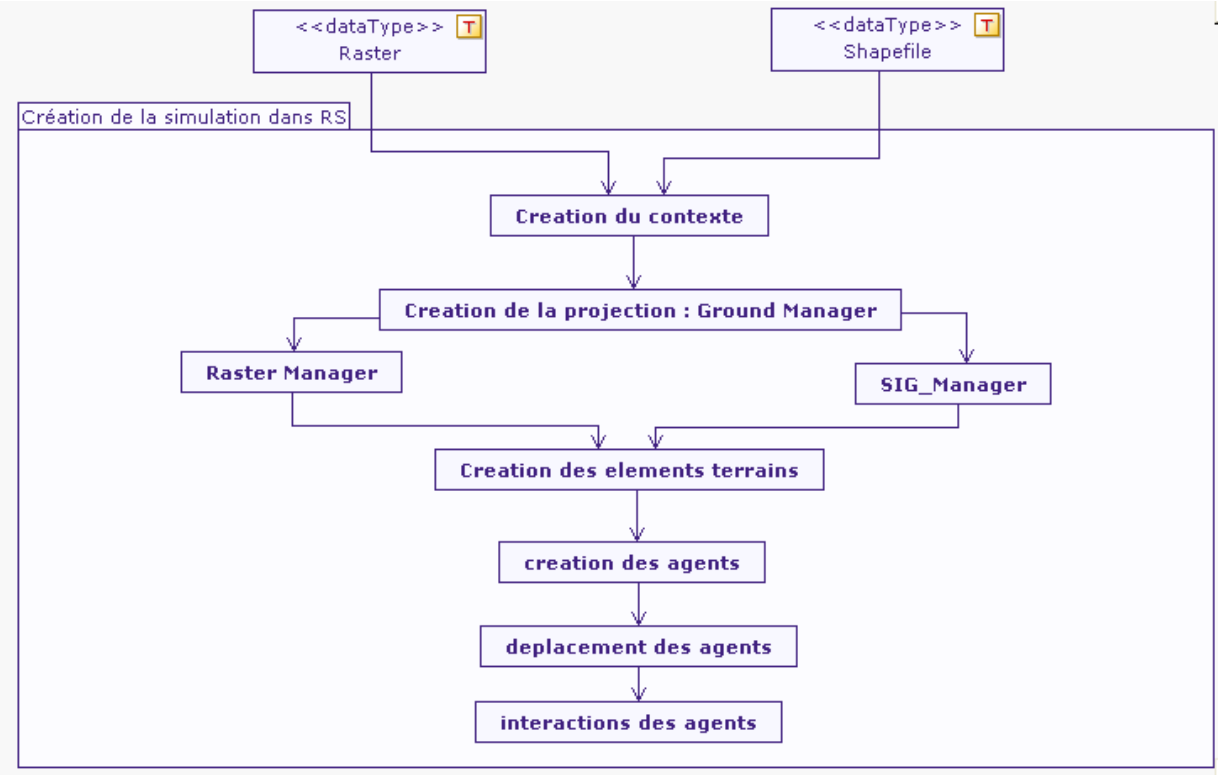



*Illustration 3: schéma de l'organisation des éléments permettant la gestion du simulateur*

# B. Validation des résultats

La validation des procédures réalisées a été réalisée à deux niveaux. Le premier vient du résultat lui même. On a réalisé chaque procédure avec un objectif précis. On défini les données en entrée et un résultat en sortie. L'analyse du résultat s'effectue donc par comparaison du résultat avec celui attendu. Les procédures ont été adaptées et modifiées jusqu'à ce que l'on parvienne à réaliser chaque étape.

Le deuxième point vient de la reproductibilité de la procédure.

- Il faut que la méthode trouvée pour parvenir au résultat fonctionne dans différents cas de figure.
- Par ailleurs il faut que la rédaction du module de la chaine de traitement soit suffisamment claire pour que toute personne extérieure puisse la comprendre et la réaliser.

En effet, une fois immergé dans une procédure on ne s'aperçoit pas toujours qu'une étape n'est pas intuitive et il faut faire attention à rester claire en expliquant les raisons de chaque action.

Avec l'aide de M Le Fur, nous avons donc repris les modules rédigés et nous avons suivi les étapes écrites du point de vue d'un utilisateur extérieur pour cerner d'éventuels oublis ou des imprécisions.

Lorsque toutes les procédures ont été fonctionnelles, nous avons commencé à chercher des dérivations possibles. Par exemple nous avons étendue le type des données en entrée. Ces dérivations ont entrainé des cas particulier dans les modules suivants et il a fallut les adapter pour

garder une cohérence dans leur enchainements.

Chaque module a donc été rédigé en plusieurs versions.

Au niveau du code Java, les classes ont été reprises plusieurs fois, pour garantir une plus grande généricité, puis pour optimiser les parcours et accélérer la simulation.

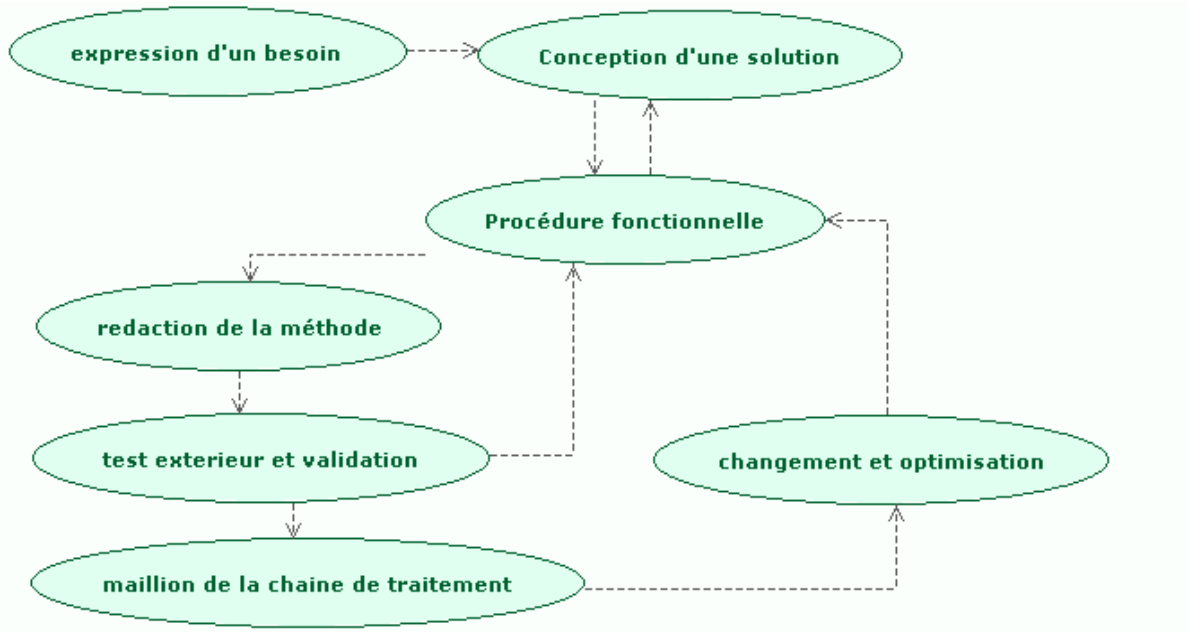
Récapitulation de la méthode élaborée et suivie

Le schéma ci dessous présente la méthode mise en place pour la conception de chaque module.

A partir d'un besoin bien défini, on imagine une solution puis on la réalise. Une fois que l'on parvient à un résultat satisfaisant, on rédige les différentes étapes en précisant les actions et les outils mis en œuvre.

Avec l'aide de M. Le Fur nous avons ensuite pu tester chaque procédure en suivant les étapes décrites mais en adoptant un point de vue extérieur. On peut alors procéder à des modifications dans la rédaction pour rendre le texte plus compréhensible. Si le résultat n'est pas satisfaisant ou que la méthode n'est pas assez adaptable, on doit reprendre la création de la procédure.

Si le module réalisé passe la validation, il est alors considéré comme un maillon de la chaîne de traitement. On pourra ensuite éventuellement le reprendre plus tard pour l'optimiser ou prendre en compte d'autres cas de figure.




*Procédure de création et de vérification des maillons (modules) de la chaîne de traitement*

# C. Les problèmes rencontrés

## *1. Installation de logiciel et compatibilité*

Lors de la première semaine, nous avons été confrontés à un grand nombre de problèmes d'incompatibilité des programmes et des librairies nécessaires pour faire fonctionner certains d'entre eux.

Nous avons commencé par installer une version de Debian (Etch), mais certains logiciels se sont révélés incompatibles avec cette version. Nous avons donc du formater le disque et installer une autre version de Linux, Kubuntu.

Les installations sous Kubuntu sont particulières (ajouts de dépôts) mais après deux jours nous avons réussi à tous les installer.

Les problèmes sont apparus lorsque l'on a voulu commencer à programmer des simulations. Repast Simphony nécessite plusieurs librairies Java qui ne sont pas incluses lors de l'installation (JTS, JAI, Geotools, Groovy, SVN, JOGL..)

Par ailleurs les procédures d'installation des ces librairies ne sont pas standards, certaines d'entre elles sont exécutables, d'autres doivent être placées dans le JDK, d'autres dans des répertoires indépendants...

Malheureusement, certaines des librairies provoquent des conflits avec celles déjà présentes dans le JDK et entraînent l'instabilité d'Eclipse. Il n'y a alors pas d'autre choix que de supprimer le JDK et de le réinstaller.

Au bout d'une semaine, les problèmes d'installation n'avaient toujours pas été résolus. Nous avons donc installé Windows sur un autre ordinateur pour pouvoir installer Repast Simphony (sous Windows, Repast Simphony est installé avec toutes les librairies).

## *2. Quelques erreurs curieuses...*

Lors de la réalisations des traitements ou de la simulation plusieurs erreurs sont apparues, elle sont parfois illogiques et difficilement détectables. En voici quelques exemples :

- Dans le simulateur Repast, il est nécessaire que chaque élément représenté possède dans sa classe Java associée au moins un attribut avec ses accesseurs. Sans quoi, Repast est incapable de générer une représentation (impossible d'en éditer une) et le simulateur s'interrompt.
- Les extensions générés par les logiciels ne correspondent pas toujours au format réel. Les extensions n'ayant pas de sens sous Linux, on peut rencontrer un fichier « .shp » mais codé en XML.
- L'utilisation d'accent dans les variables ou attributs pose problème lors du passage de Linux vers Windows.
- Certains programmes peuvent bloquer l'accès à certains fichiers pour les autres programmes. Dans le cas d'Excel, il interdit aux autres programmes de modifier les fichiers ouverts.
- Dans un fichier KML représentant des formes géométriques, il faut faire attention à ne pas introduire d'éléments possédant des géométries différentes, sinon le format du fichier lui

même ne sera pas accepté. ( dans un fichier de 300 polygones, si le simulateur trouve une géométrie de type point utilisée comme repère, le fichier sera refusé).

- Le changement d'une projection d'un plan sphérique vers une surface plane ou inversement peut provoquer d'importantes déformations.
- Le nombre de point définissant un polygone est aussi important que le nombre de polygones en terme de ressource. Il est préférable de travailler avec 4 polygones de 5 points que avec 2 polygones de 130 points...
- Même si les données sont identiques, le simulateur fera la différence entre les formats Polygones et multipolygones. Ainsi des formes identiques seront incompatibles si le format n'est pas correctement interprété.
- Certain formats de fichiers évoluent vite et il est parfois nécessaire de changer l'en-tête pour pouvoir les utiliser avec certaines applications. C'est le cas de certains fichier KML dont les dernières version ne sont pas toujours reconnues par tous les logiciels SIG.
- Il est très délicat de travailler avec des valeurs continues dans une grille. On peut provoquer des oscillations ou des immobilisations des éléments.

# IV. Bilan

## 1. Les apports

### *A. Organisation*

Ce stage m'a permis de travailler en autonomie et de prendre des initiatives. Les trois premières semaines de stages se sont effectuées en autonomie pour la partie concernant la programmation. J'ai reçu l'aide de M. Piry pour la manipulation des logiciel SIG et M Le Fur m'avait heureusement laissé à disposition des références pour trouver de la documentation et des tutoriaux sur le fonctionnement de Repast Simphony.

La conception de la chaîne de traitement a apporté de nombreux problèmes, chaque étape a posé un certain nombre de problème et il a fallu mettre en place des outils et des procédures pour pouvoir les résoudre. Dans certains cas, la cause des erreurs était invisible, en particulier dans Repast Simphony ou le fonctionnement du simulateur peut être perturbé par de petits éléments, la seule solution a alors été de tout modifier jusqu'à trouver l'élément qui entraîne l'erreur.

La réalisation de la chaîne de traitement à nécessité de travailler de manière méthodique de façon à conserver une cohérence entre tous les modules de la chaîne de traitement et de rester suffisamment clair pour qu'une personne extérieure au contexte puisse suivre et reproduire la majorité des traitements.

La mise en place des différents éléments a également nécessité la recherche d'information.

Cette activité de recherche s'est faite durant toute la durée du stage , grâce au Javadoc de Sun, Repast, JTS et Geotools, les forums des sites consacrés aux logiciels utilisés et les divers tutoriaux.

Tous ces éléments m'ont permis d'avoir un aperçu du monde du travail en entreprise, et de savoir comment je pourrais réagir pour résoudre un problème donné en passant par chaque phase de la réalisation du travail. En passant par l'analyse du problème, la conception puis la réalisation de solutions adaptées.

## *B. Apprentissage*

Les travaux réalisés pendant ce stage m'ont également apporté des connaissances pratiques :

1. Linguistique : J'ai l'occasion par exemple travailler l'anglais, dans le cadre de la chaîne de traitement, les logiciels utilisés ne sont pas tous couramment utilisé (en particulier Repast Simphony) il n'existe donc pas de documentation complète en français. Il est nécessaire de faire des recherches sur des sites spécialisés anglophones (site, forum...). La Javadoc des modules utilisés par Repast est également rédigé en anglais. La recherche de documentation m'a permis d'apprendre la traduction de différents termes techniques en anglais. Par ailleurs, en accord avec mon responsable de stage, j'ai rédigé le commentaire des codes sources en anglais.

2. Linux et Windows : La majorité des logiciels utilisés tournait sous Linux. J'ai donc procédé à l'installation d'une Version Debian, puis suite à des problèmes d'incompatibilité j'ai formaté l'ordinateur pour installer une version de Kubuntu. Pour faire fonctionner le simulateur Repast Simphony, j'ai travaillé sur un deuxième poste fonctionnant sous Windows. La manipulation des fichiers entres les différentes plateformes m'a permis de mieux prendre en compte certains critères de compatibilité ( accent, extension...).

3. Travail en console ; l'installation des logiciels étant particulière sous Linux, j'ai réalisé un certain nombre d'installation depuis la console Linux (gestion des droits pour l'exécution, création de liens symboliques entre les bibliothèques...).

   De plus certaines applications, comme Grass et PostGis fonctionnent en partie en console (l'intérêt étant qu'il est possible de concevoir des scripts pour les gestions.

4. Traitement d'image, opération de numérisation : pour transformer une image, il y a plusieurs façons de procéder. Dans le cas où on utilise un logiciel de reconnaissance de zone par couleur. On doit effectuer des traitements sur l'image pour faciliter la découpe (augmentation du contraste, simplification des zones, pixelisation...).j'ai donc eut l'occasion d'utiliser certains fonctionnalité de Gimp pour préparer une image raster avant de transformer les différentes plage de couleurs en polygones.

5. La géomatique : un des éléments les plus enrichissant lors du stage fut la découverte de l'univers des SIG, ils sont couramment utilisés et s'adaptent à un grand nombre de situations.

   Parallèlement, j'ai découvert quelques éléments du monde de la cartographie, notamment en ce qui concerne les problèmes de changement de projection d'un repère sphérique vers un repère orthonormé.

6. Programmation, simulation et optimisation en Java : j'ai passé une grande partie du stage à essayer de créer les éléments de base d'une simulation pertinente sur un terrain quelconque (carte SIG ou grille/raster).

   La première difficulté a été de concevoir des interactions générique sans utiliser l'héritage

(Repast impose des contraintes sur les classes héritées pour la représentation). J'ai pu résoudre le problème en manipulant des interfaces.

Je me suis retrouvé confronté à des problèmes d'optimisation. En effet les interactions entre les agents et le terrain pouvant être complexes. Il faut faire attention à réduire au maximum les parcours (un raster qui contient 120 pixels sur 120 pixels peut être très lourd si on doit recalculer chaque pixel). Dans cette optique, j'ai appris plusieurs techniques dont l'utilisation de table de hachage pour réduire les parcours et l'utilisation d'interface comme moyen de conserver une gestion générique du terrain sans connaître la nature du terrain ou des agents qui vont le peupler.

Lors de la fin du stage, j'ai également été amené a faire des opérations sur les rasters depuis les classes de gestion d'image java. Certaine étapes sont délicates puisque les rasters sont souvent codé en byte, il faut alors les convertir en « int » pour associer une valeur entière et l'utiliser comme attribut.

Ce qui n'est pas toujours facile car les valeurs sont interprétées comme des « bytes signé » (c'est à dire avec des valeur comprises entre -128 et 127 au lieu de 0 et 256), il faut alors faire une opération pour ignorer le bit de signe). J'ai alors eut l'occasion d'apprendre des manipulation que je ne connaissait pas en utilisant les opérateur bit à bit avec un masque en hexadécimal.

7. Le langage SQL : pour effectuer des opérations sur les shapefile, j'ai utilisé des requête SQL grâce à PostGis, de plus j'ai travaillé indirectement sur des bases en effectuant des jointures sur des tables d'attribut pour créer de nouveaux shapefiles Bien que la partie SQL utilisée soit restreinte, cela m'a permis de découvrir une des extensions possibles de ce langage en utilisant des requêtes spatiales.

# 2. Bilan professionnel

L'objectif annoncé pour la réalisation de ce stage était de concevoir une chaîne de traitement permettant d'utiliser des fichiers contenant des informations géoréférencées comme support d'une simulation.

La réalisation de la chaîne de traitement elle même a été relativement rapide, mais une grande partie du stage à été utilisée pour la recherche de solutions alternatives (comme l'utilisation des rasters à la place des shapefiles) et des « améliorations » des solutions déjà mises en place (par exemple utilisation de tables de hachages pour améliorer les parcours).

Le résultat final est satisfaisant puisqu'il permet plusieurs solutions différentes et que les modules permettent de s'adapter à un grand nombre de besoins différents. La présentation des logiciels n'est pas complètement orientée sur la simulation puisque l'on a fait en sorte que les modules soit indépendants.

La partie proposant une structure pour la plateforme de simulation reste également relativement ouverte. Les composants sont peu nombreux mais proposent une généricité et une encapsulation solide. Même si dans certains cas on aurait préféré associer certains éléments à une classe particulière (par exemple qu'un agent dynamique connaisse ses coordonnées ou qu'un agent parcelles connaisse ses voisins) chaque aspect a été prévu pour s'adapter à des supports très différents, ce qui permettra de mettre en place des simulations très diverses et sur différentes échelles.

De plus conformément aux spécifications attendues, il est possible de modifier rapidement tous les paramètres de la simulation et de changer les types de support simplement en modifiant les constantes d'une interface.

# 3. Bilan personnel

Au lieu de se contenter de la programmation, j'ai eu l'occasion de faire des recherches depuis l'expression d'un besoin jusqu'à la solution finale.

La chaîne de traitement est le résultat de nombreuses étapes de tests et de conceptions, d'analyse des résultats et de corrections.

La portée du projet m'a permis d'apprendre beaucoup de choses sur des univers très différents

En particulier les SIG et la simulation, mais également pour l'imagerie 2D, les requêtes spatiales depuis Postgres et les manipulations en console sous Linux.

J'ai été amené à utiliser différents outils parfois très particuliers et dont la logique m'était complètement inconnue. Mais j'ai heureusement réussi à utiliser tous les traitements de bases et à utiliser les fonctionnalités nécessaires pour la chaîne de traitement.

Pour finir la programmation de la simulation a permis de tester les résultats de tous les traitements précédents.

La liberté que l'on m'a donné pour la conception et la réalisation de cette chaîne de traitement m'a permis de découvrir les attentes et les résultats attendus lors d'un développement professionnel et la rédaction des modules m'a permis de faire en sorte que le travail réalisé soit utilisable par la suite.

En plus des connaissances et de la méthodologie que m'ont apporté ce stage, je suis satisfait d'avoir pu mener à bout le projet que l'on m'a confié et qui servira de base pour les développement des simulations à venir.

Quentin Baduel**, DUT Informatique, du 09/02/09 au 24/04/09.**

**Sujet: conception et mise en œuvre d'une chaîne de traitement conduisant à la création d'un environnement SIG (système d'information géographique) de base pour un simulateur. Responsables : Jean Le Fur et Sylvian Piry.**

Descriptif et objectifs : Ce travail s'inscrit dans le cadre d'un projet de recherche sur la dynamique des populations de rongeurs. Il correspond à un module visant au développement d'une plate-forme générique de simulation multi-agents des rongeurs dans leur environnement. Dans le cadre de cette plate-forme les agents simulés sont amenés à évoluer dans un environnement le plus réaliste possible. Les travaux réalisés dans le centre de recherche se fondent sur l'utilisation de cartes géographiques disponibles numériquement dans des systèmes d'information géographique.

Objet du stage : concevoir et formaliser une chaîne de traitement permettant la récupération des cartes et leur implémentation dans la plate-forme de simulation. Une application sera réalisée à titre d'illustration représentant des populations d'agents (de rongeurs) évoluant dans un espace hétérogène (forêts, villages, champs) simulé à partir des cartes terrain. Produit attendu du stage : formalisation du processus complet de passage de la carte numérisée au support numérique des agents dans le simulateur.

# Annexes

## Références

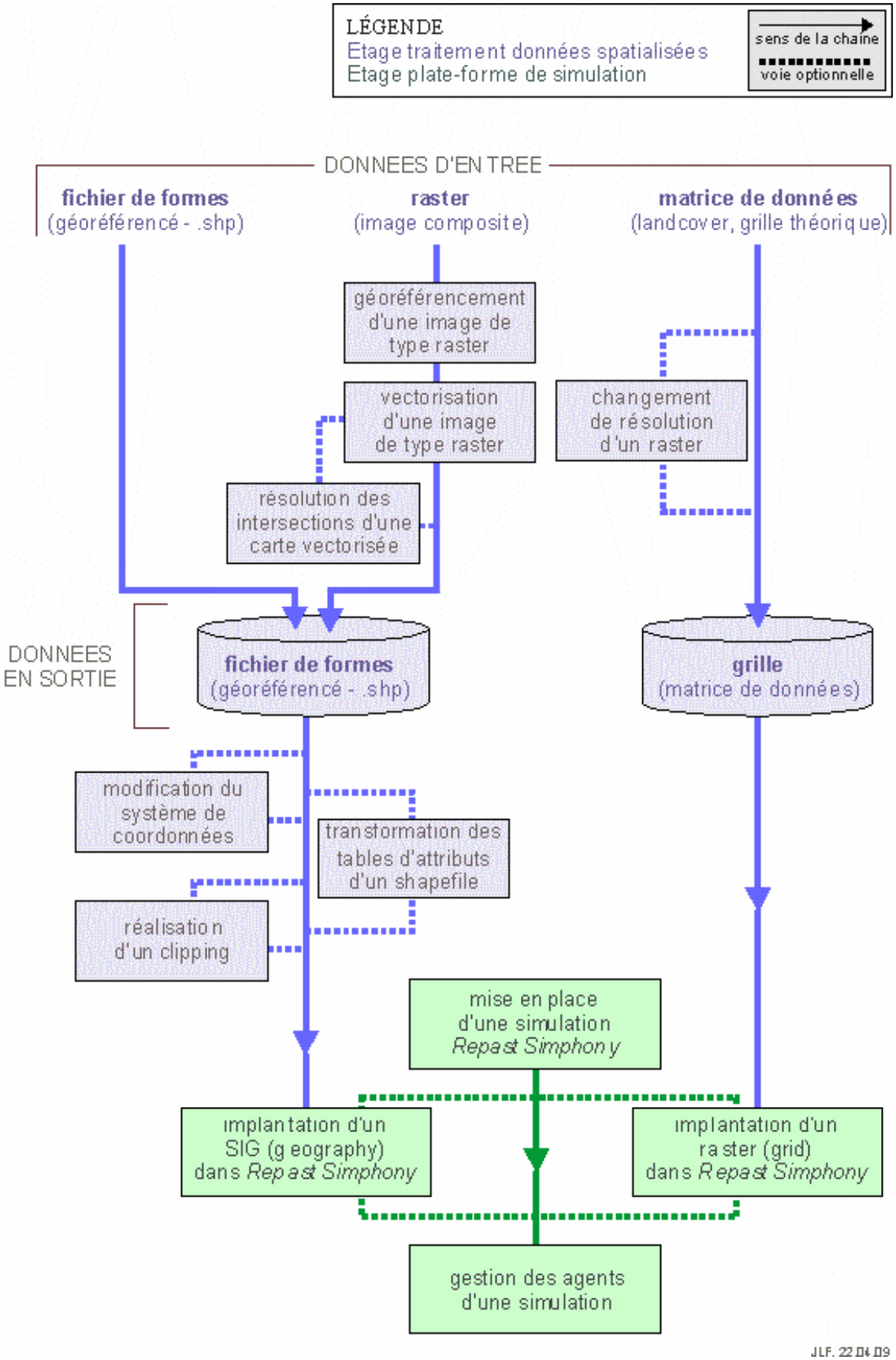
### Références bibliographiques


- GIS for web developers de Scott Davis
- Les systèmes multi-agentss, vers une intelligence collective de Jacques Ferber
- Programmer en Java : de Claude Delannoy


### Liens utiles

- Site de *Repast* Simphony : http://repast.sourceforge.net/index.html
- Forum utilisateurs : http://sourceforge.net/mailarchive/forum.php?forum_name=repast-interest
- Tutoriaux pour Repast Simphony:http://repast.sourceforge.net/docs/tutorial/SIM/
- Site de JTS (manipulation des objets geometry): http://www.vividsolutions.com/jts/jtshome.htm
- Site de Geotools (geometry, sig, shp...) : http://geotools.codehaus.org/
- Site de Google Earth: http://earth.google.fr/
- Site de Qgis : http://www.qgis.org/
- Site de GRASS: http://grass.itc.it/gdp/index.php
- Site de Postgres: http://www.postgresql.org/
- Site de Postgis : http://postgis.refractions.net/ & http://www.postgis.fr/
- Site de Gimp: http://www.gimp.org/

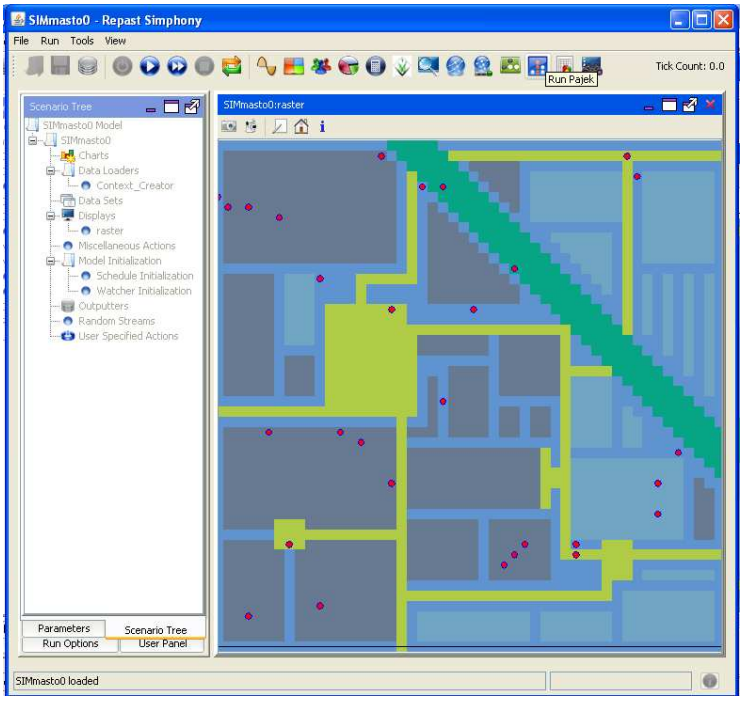
# Schéma Synoptique de la chaîne de traitement



# Représentation d'un raster

Les rasters sont des matrices de données, mais il est possible d'associer à chaque donnée un pixel, c'est à dire une petite surface dont la couleur est définie par la donnée elle même ou par une table de correspondance entre la donnée et une liste de couleur. Ci-dessous on représente une matrice de valeur avec sa représentation graphique résultante

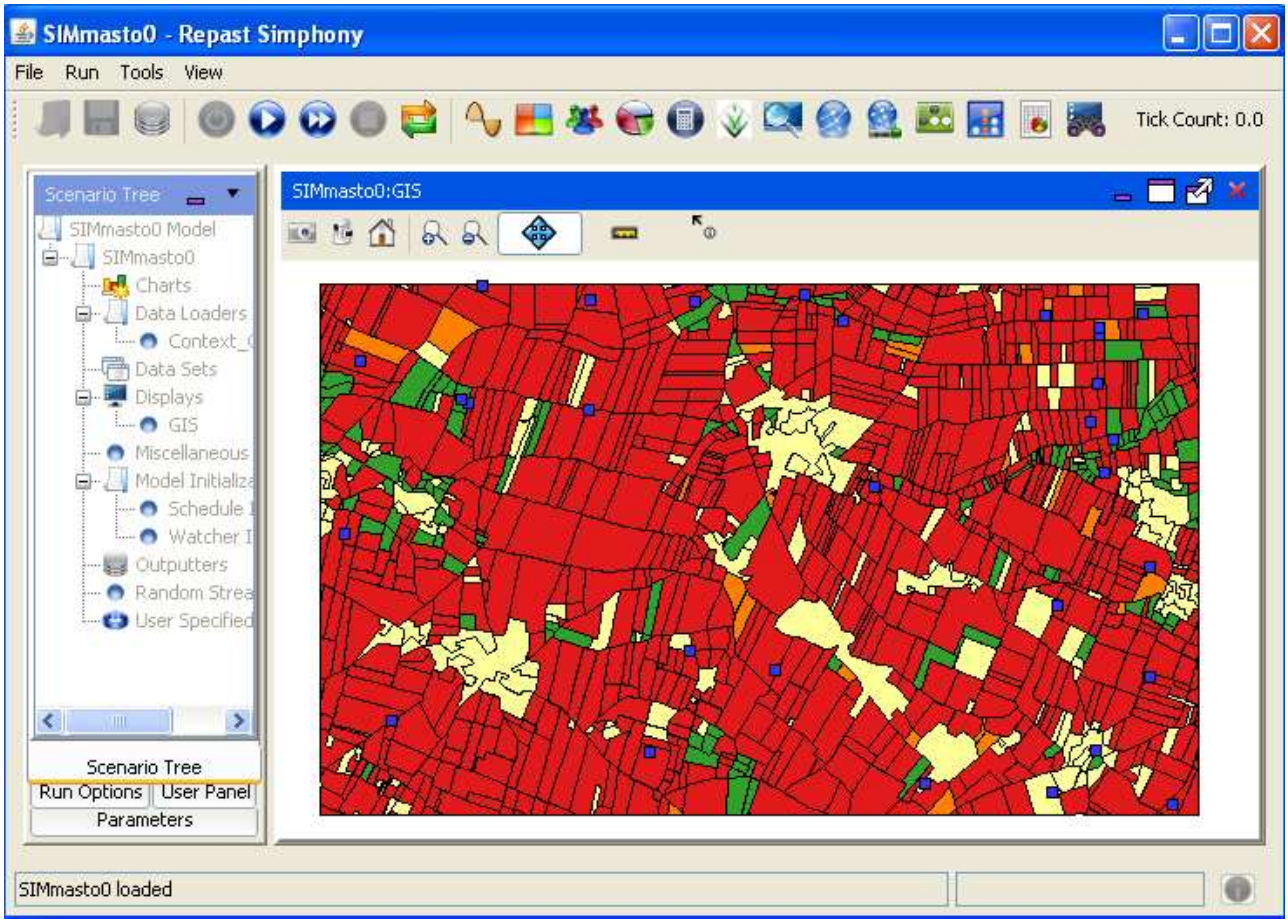
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 0 | 0 | 0 | 0 | 0 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 |
| 2 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 9 | 3 | 3 | 3 | 3 | 3 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 1 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 |
| 4 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 1 | 0 | 0 | 0 | 0 | 0 | 9 | 3 | 3 | 3 | 3 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 |
| 5 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 1 | 9 | 0 | 0 | 0 | 0 | 0 | 9 | 3 | 3 | 3 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 |
| 6 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 1 | 9 | 9 | 0 | 0 | 0 | 0 | 0 | 9 | 3 | 3 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 |
| 7 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 1 | 9 | 3 | 9 | 0 | 0 | 0 | 0 | 0 | 9 | 3 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 |
| 8 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 1 | 9 | 3 | 3 | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 9 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 |
| 9 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 1 | 9 | 3 | 3 | 3 | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 |
| 10 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 1 | 9 | 3 | 3 | 3 | 3 | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 9 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 |
| 11 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 1 | 9 | 3 | 3 | 3 | 3 | 3 | 9 | 0 | 0 | 0 | 0 | 0 | 9 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 |
| 12 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 1 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 0 | 0 | 0 | 0 | 0 | 9 | 9 | 7 | 7 | 7 | 7 | 7 | 9 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 |
| 13 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 1 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 0 | 0 | 0 | 0 | 0 | 9 | 9 | 7 | 7 | 7 | 7 | 9 | 1 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 |
| 14 | 9 | 3 | 3 | 3 | 3 | 3 | 9 | 7 | 7 | 7 | 9 | 9 | 9 | 9 | 1 | 1 | 1 | 1 | 1 | 1 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 9 | 9 | 7 | 7 | 7 | 9 | 1 | 9 | 7 | 9 | 7 | 7 | 7 | 9 |
| 15 | 9 | 3 | 3 | 3 | 3 | 3 | 9 | 7 | 7 | 7 | 9 | 9 | 9 | 9 | 1 | 9 | 9 | 9 | 9 | 9 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 9 | 9 | 7 | 7 | 9 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 9 |
| 16 | 9 | 3 | 3 | 3 | 3 | 3 | 9 | 7 | 7 | 7 | 9 | 9 | 9 | 9 | 1 | 9 | 9 | 9 | 9 | 9 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 9 | 9 | 9 | 1 | 9 | 7 | 9 | 7 | 7 | 7 | 7 | 7 | 9 |
| 17 | 9 | 3 | 3 | 3 | 3 | 3 | 9 | 7 | 7 | 7 | 9 | 9 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 9 | 9 | 1 | 9 | 7 | 7 | 9 | 7 | 7 | 7 | 7 | 9 |
| 18 | 9 | 3 | 3 | 3 | 3 | 3 | 9 | 7 | 7 | 7 | 9 | 9 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 9 | 1 | 9 | 7 | 9 | 7 | 7 | 7 | 7 | 7 | 9 |
| 19 | 9 | 3 | 3 | 3 | 3 | 3 | 9 | 7 | 7 | 7 | 9 | 9 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 9 | 7 | 9 | 7 | 7 | 9 | 7 | 9 |
| 20 | 9 | 3 | 3 | 3 | 3 | 3 | 9 | 7 | 7 | 7 | 9 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 0 | 0 | 0 | 0 | 0 | 9 | 9 | 7 | 7 | 9 | 7 | 9 |
| 21 | 9 | 3 | 3 | 3 | 3 | 3 | 9 | 9 | 9 | 9 | 9 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 9 | 0 | 0 | 0 | 0 | 0 | 9 | 9 | 7 | 9 | 7 | 9 |
| 22 | 9 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 9 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 9 | 9 | 0 | 0 | 0 | 0 | 0 | 9 | 9 | 7 | 9 | 7 |
| 23 | 9 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 9 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 9 | 9 | 9 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 9 | 9 | 7 | 9 | 7 |
| 24 | 9 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 9 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 9 | 3 | 9 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 7 | 7 | 7 | 7 | 9 | 0 | 0 | 0 | 0 | 0 | 9 | 9 | 9 |
| 25 | 9 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 9 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 9 | 3 | 9 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 7 | 7 | 7 | 7 | 9 | 9 | 0 | 0 | 0 | 0 | 0 | 9 | 7 |
| 26 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 9 | 3 | 9 | 3 | 3 | 3 | 3 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 1 | 7 | 7 | 7 | 7 | 9 | 9 | 9 | 0 | 0 | 0 | 0 | 0 | 9 |
| 27 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 9 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 7 | 7 | 7 | 7 | 9 | 9 | 9 | 0 | 0 | 0 | 0 | 0 | 9 |
| 28 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 1 | 9 | 9 | 3 | 9 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 7 | 7 | 7 | 7 | 9 | 7 | 7 | 9 | 9 | 0 | 0 | 0 | 9 |
| 29 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 7 | 7 | 7 | 7 | 9 | 7 | 7 | 9 | 9 | 9 | 0 | 0 | 0 |
| 30 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 1 | 7 | 7 | 7 | 7 | 9 | 7 | 7 | 9 | 9 | 9 | 9 | 0 | 0 |
| 31 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 1 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 0 | 0 |
| 32 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 1 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 9 | 0 |
| 33 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 1 | 1 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 9 | 0 |
| 34 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 1 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 9 | 9 |
| 35 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 1 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 3 | 9 |
| 36 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 1 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 3 | 9 |
| 37 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 1 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 3 | 9 |
| 38 | 9 | 3 | 3 | 3 | 3 | 3 | 1 | 1 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 1 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 3 | 9 |
| 39 | 9 | 9 | 9 | 9 | 9 | 9 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 3 | 9 |
| 40 | 9 | 3 | 3 | 3 | 3 | 3 | 1 | 1 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 9 | 9 | 1 | 1 | 1 | 9 | 9 | 9 | 9 | 9 | 9 | 9 |
| 41 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 42 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 9 | 9 | 9 | 1 | 1 | 1 | 9 | 9 | 9 | 9 | 9 | 9 |
| 43 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 9 | 9 | 9 | 1 | 1 | 1 | 9 | 7 | 7 | 7 | 7 | 9 |
| 44 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 1 | 9 | 9 | 9 | 9 | 9 | 9 | 1 | 9 | 9 | 9 | 9 | 9 | 9 |
| 45 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 7 | 9 |
| 46 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 7 | 9 |
| 47 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 7 | 9 |
| 48 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 7 | 9 |
| 49 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 9 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 1 | 9 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 9 | 7 | 9 |
| 50 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 1 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 1 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 9 |



# Représentation d'un shapefile

Les shapefiles sont des fichiers contenant des éléments géométriques définis par un ou plusieurs points. Chaque élément peut contenir différentes données. Ci dessous, on a représenté un shapefile dont les éléments représentent des parcelles de terrains, ces parcelles contiennent des attributs (catégories, humidité, surface, propriétaire...)

On peut représenter ces éléments avec une couleur représentant un attribut. Dans cet exemple, la couleur représente l'affinité des rongeurs pour chaque terrain.



# Chronologie

**le 26/01/09**:

Rendez-vous essentiellement administratif et présentation de l'équipe du CBGP

**le 30/01/09:**

Lors de ce rendez vous, nous avons repris les différents points à aborder lors du stage,

nous avons vu les outils à utiliser (Arcview , Qgis , Repast-Simphony et différentes applications pouvant être utiles dans les étapes intermédiaires du traitement ( Google earth, Grass).

**Le 09/02/09 –Debut du stage**

installation de Debian Etch (+ mise à jour de la liste des dépôts et des paquets)

-installation d'Eclipse 3.4

-installation du JDK et JRE

-essai d'installation de Qgis => il est apparu que la distribution de Qgis pour Debian n'était compatible qu'avec la version "Lenny" de Debian, après plusieurs tentatives nous avons décidé de désinstaller Debian et de recommencer sous KUbuntu

-installation de kubuntu intrepid

**le 10/02/09**

-mise à jour de la liste des dépôts et des paquets pour KUbuntu

-installation de QGis (Ajout d'un dépôt particulier dans /etc/apt/sources.list puis téléchargement et installation via apt-get)

-installation du (JDK téléchargement sur le site de Sun pour disposer de la version 1.6)

-installation de Eclipse (téléchargé depuis le site du projet Eclipse)

=> Eclipse ne démarre pas, ( fenêtre vide au démarrage)

=> La dernière version d'Eclipse ne possède pas les bons composants de compatibilité sous Kubuntu: la version Eclipse Ganymède ( 3.4 ) n'est disponible que depuis le site d'Eclipse par paquet, et se décompresse en dossier "pret à l'emploi" mais sans les dépendances. Pour remédier au problème, suite aux conseils de Sylvain Piry , nous avons installé la version 3.2 de Eclipse par l'intermédiaire de APT-GET qui installe en parallèle les dépendances nécessaires à son fonctionnement. Une fois la version 3.2 installée, la version 3.4 démarre normalement.

-ajout d'un serveur de mise à jour dans les paramètres de Eclipse (http://mirror.anl.gov/pub/RepastSimphony) puis téléchargement de

Repast-Simphony.

-Premiers pas sous Qgis, chargement de carte et activation de la coloration des polygones selon leur catégories

**le 11/09/02**

-entrainement à Repast symphony grâce aux tutoriels -" Creating a basic Gis Agent environement "& de Karl D.Libert

-"Repast Simphony, non GuiTutorial" de nick Malleson

=> au cours de la réalisation de ces deux tutoriels un problème est apparu, le code est bon et aucune erreur n'est détectées. Le simulateur démarre mais au moment de choisir le style d'un agent une erreur survient dans la console: impossible de trouver la librairie

"Javax.media.jai.utils.range"

-Essai de correction du problème ,

-diverse mise à jours d'Eclipse,

=> le réseau à tendance a provoquer la coupure des téléchargement, et Eclipse ne redémarre pas les téléchargements de mise a jour interrompus. impossible d'installer de grosses mises à jour, juste quelques éléments qui semblent pouvoir débuguer la situation. (de plus le trafic du réseau est irrégulier chaque téléchargement est lent estimation à 49 476 jours pour télécharger 52Mo...)

-téléchargement et installation de la librairie JAI (pour le JDK) dans le JDK

-téléchargement et installation de la librairie JAI (pour le JRE) dans le JRE

-recherche de solution sur internet

-Lecture des documents concernant les tutoriels Repast-Simphony

-Lecture du livre « GIS Web devellopers" de Scott Davis et réalisation de quelques exercices associés à l'aide de Qgis.

**le 12/09/02**

Par défaut de tableur de Open Office ne permet pas d'ouvrir les format « .dbf ». Après recherche il est apparu qu'il fallait mettre à jour Open Office et installer Open Office Database pour permettre la gestion des DBF

-installation de Open Office Database

-édition du fichier dbf de la zone « chizé » pour redéfinir des catégories

-on effectue un tri de la feuille de calcul selon le champs OCS , puis on effectue une association entre les différents champs OCS et une références vers une nouvelle catégorie ( libre, mi_culture, culture, anthropisé ou problème)

-on peut réordonner la feuille en effectuant un second tri sur le champs d'index

-on charge le nouveau fichier dbf dans Qgis, puis on utilise le champs nouvellement créé pour colorer les polygones selon leur catégories et non leur références de zone.

-Suite à la référence donnée par Sylvain, installation de ftools, un plugin pour qgis.

-installation d'extensions en python pour Qgis, fournis par le gestionnaire d'extension en relation avec le site "http://www.ftools.ca/"

poursuite des recherches pour débuguer Repast Simphony,

-création d'un "mini-code" Java sous nano pour essayer de vérifier la compilation "à la main" , les bibliothèques standards sont chargée normalement mais impossible d'atteindre la bibliothèque "Javax.media"

-en parcourant le forum pour l'aide à l'utilisation de Repast ("") il semble que Repast Simphony ait des problèmes de compatibilité entres les différentes version des plugins dont il a besoin pour fonctionner, utiliser un plugin de vérification de compatibilité de version devrait permettre de résoudre les problèmes

-installation de Subversion SVN ( ajout d'un site dans le gestionnaire de mise à jour d'Eclipse et téléchargement)

il faut ensuite ouvrir une perspective SVN et lancer la procédure de comparaison de version entre son projet et la version mis en ligne sur le répertoire "

http://Repast.svn.sourceforge.net/svnroot/Repast "

lancement de svn pour vérifier la compatibilité des éléments de Simphony.

=> suite aux coupures réseaux, la vérification du projet par la svn est régulièrement interrompu et ne reprend pas toujours.

réalisation d'un clipping d'une zone de chizé dans Qgis

-création d'une nouvelle couche vide

-passage en mode d'édition de la couche vide

-création d'un carré grâce à l'outil de dessin de polygone fourni par ftools

-enregistrement de la nouvelle couche

-utilisation d'un outils d'extraction fourni par ftools pour récupéré les éléments de la couche "chizé" selon l'intersection avec le carré dessiné dans l'autre couche.

**le 13/09/02**

-relancement de la procédure de vérification par SVN qui n'a pas réussit à aboutir la veille

-des éléments de l'environnement nécessaires à Repast Simphony semblent mal installés, (EMF) mais l'outil de mise à jour n'arrive pas a charger les éléments manquants.

-tentative d'ajout des bibliothèques manquantes par ajouts dans les dossiers en suivants la procédure indiqués dnas le site "http://wiki.Eclipse.org/EMF/Installation" mais Eclipse est devenu instable ( redémarage infini) et il a fallut le supprimer

- reinstallation de Eclipse, reparamétrage et ajout de plusieurs site de mise à jour. Il a fallut plusieurs mise a jour pour permettre le téléchargement de EMF

-ajout des différnets composants de l'environnement nécessaire pour Repast Simphony

# Ganymede -> Models and Model Development -> EMF - Eclipse Modeling Framework Runtime Modeling Framework and Tools

# Ganymede -> Graphical Editors and Frameworks -> Graphical Editing Framework GEF

# Ganymede -> Graphical Editors and Frameworks -> Graphical Editing Framework GEF SDK

# Groovy (http://dist.codehaus.org/groovy/distributions/update/) -> GroovyFeature 1.5.7

# SVN (http://subclipse.tigris.org/update_1.4.x) (All Except "Integrations (optional)")

-essai de réinstallation de Repast Simphony ( problème lors du téléchargement)

-tutoriel sur les format XML et KML

http://code.Google.com/intl/fr/apis/KML/documentation/KMLreference.html

**le 16/09/02**

-problème de connexion impossible lancer des mises à jour

-début de la réalisation d'une mini application pour créer des rectangle en KML (dans le but d'effectuer plus simplement des clips de cartes)

-Après conseil auprès du responsable informatique, création d'une session Windows pour installer Repast Simphony (téléchargement de Eclipse window + JDK )

-problème l'exécutable d'installation du JDK ne fonctionne pas (erreur 1311 impossible d'accéder aux fichier ....cab lors de l'installation)

-recherche pour solutionner ce nouveau problème

-problème référencé sur certains forums mais non solutionné

problème résolu par Sylvain, installation de netBeam, livré avec le JDK

**-le 17/02/09**

-lancement du téléchargement de Repast pour Windows (300Mo)...

-reprise de la fonction de création de carré en KML

-reprise des tutoriels Repast-Simphony

gis_basique et non gis

installation d'utilitaire (oppenoffice, firefox, pdfreader)

-tutoriel predator-prey-continuous mais problème lors de l'exécution

**le 18/02/09**

-correction et fin du tutoriel predator-prey-continuous

-début des recherches pour intégrer le SIG, avec ses formes et ses données dans le simulateur

=> les fonctions Repast ne suffisent pas il faut utiliser des bibliothèque de JTS pour gérer les formes, mais JTS ne lit que les format WKT ou WKB

-parcours de la doc JTS, et Repast, plusieurs solutions sont envisageable.

-la gestion de JTS ne se fait que sur les aspects géométriques et pas les bases de données, il faut apparemment utiliser Geotools

**Le 19/02/09**

recherche sur la façon d'utiliser Geotools

=> problème la Javadoc de Geotools est hs

travail sur l'affichage des différents éléments dans le simulateur.

=> lorsque l'on ajoute un agent à une géométrie dans une géographie, l'agent et la géométrie sont associés, ( l'agent prend la forme de la géométrie dans laquelle on l'ajoute)

=> on peut donc considérer chaque « parcelle » comme un agent, les agents acteurs proprement dit auront des géométries propres et se déplaceront selon des coordonnées libre que l'on devra mettre en correspondance avec les parcelles.

**Le 20/02/09**

-téléchargement d'une Javadoc pour Geotools ( version 2,5,0 au lieu de la 2,5,3)

-recherche et tests sur les api de Geotools

-problème d'affichage avec le simulateurs

-intégration d'un SIG

-ajout d'agent point sur le SIG

-récupération des attributs via le shapefile

-colorisation du SIG en fonction de la valeur d'un attribut ( échelle de valeur).

**Le 23/02/09**

intégration du SIG de chizé dans le simulateur

-(erreur dans le format, le fichier shp de chizé était en réalité un fichier XML avec une extension shp)

mise en place de la détection de l'enveloppe contenant le SIG

-mise en place d'une fonction de répartition aléatoires des agent dans le SIG

-mise en place d'une fonction permettant aux agents de se dé placer ( à l'intérieur du SIG)

mise en place de l'interaction des agents avec leur environnement ( les agents détectent le type de parcelle sur laquelle ils se trouvent.

**-le 24/02/09**

-Rédaction du début du rapport sur la chaine de traitement et du tutoriel pour l'implantation du SIG

**Le 25/02/09**

-reprise du rapport

-réalisation d'une jointure grâce a ftools dans qgis

-manipulation des dbf

-Manipulation de arcgis : -importation du shapefile de schizé

-coloration selon l'attribut zone

-clipping d'une zone

-jointure de la table de cette zone avec une table de correspondance pour les catégories.

-remise d'une première version du rapport

**le 26/02/9**

mise en place de fonction de création de géométrie depuis le simulateur

-carré ( ok mais il faut entrer 5 coordonnée, avec la première et la dernière identiques)

-cercle( impossible directement mais on )peut end dessiner en tant que multi_poligone)

Attention, puisque les agent de même types doivent avoir la même géométrie, il faut que les géométries crées soient du même type ( à savoir multi_polygone)

mise ne place d'un champs de vision pour chaque agent

-les agents se détectent à distance et se regroupe

-les agent recherchent les zones où ils sont une grande affinité.

Début de la création d'histogramme pour étudier l'évolution de certains paramètres

**le 27/02/09**

modification des méthodes de détection de l'environnement des agents, les méthodes existantes utilisées donnent des géométries très éloignées, redéfinition des fonction effectuées la veille, on récupère les géométries et on teste leur intersection pour récupérer celles qui sont réellement visibles.

Résolution d'un bug de méthodes de déplacement en concurrence .

Début de la transformation de la carte du marché de kedougou avec l'aide de gimp.

Création de différents calques selon les zones et récupération des éléments à l'aide du sélecteur de couleurs.

**Le 02/03/09**

(retour de M,Le Fur)

-présentation sommaire des procédures mise en place( clipping, jointure et représentation pour les SIG et présentation du code réalisé dans le simulateur Repast Simphony).

-remise d'une version du rapport et du code

-reprise du rapport, suite à l'avis de M Piry, pour organiser une mise en page plus contrasté entre les partie tutoriels et exemples.

**Le 03/03/09**

-Correction d'un bug de déplacement des agents, la sélection des géométrie n'est pas aléatoire mais favorise la sélection des géométries situé dans la partie supérieur de la zone de sélection.

-reprise de la carte de Kedougou avec Gimp

-recherche sur les moyens de numériser une carte depuis le résultat obtenu après le traitement de l'image sous gimp.

Des logiciel sont mis en avant sur internet mais sont propriétaires (exemple autocad ).

**Le 04/03/09**

reprise du rapport (mise en page , correction et début de la mise en place de la procédure de numérisation d'une image

reprise du code du simulateur afin de le rendre plus structuré, création d'une Javadoc et de commentaires en anglais ( changement des noms de fonctions et de variables)

-essai de numérisation de la carte avec Grass (impossible ou compliqué a ouvrir sans créer de projet Grass depuis qgis)

avec Sylvain: Manipulation de gimp, qgis et Grass pour numériser une carte après traitements

-importation georérencement, conversion pour Grass, vectorisation du raster,

il reste a traiter le résultat pour conserver les données pertinentes, création des attributs, conversion en shapefile, modification du crs ( coordinate reference system)

-rédaction sommaire de la première partie de la procédure.

**Le 05/03/09**

-réalisation de la première partie de la numérisation d'une image: il semble plus simple de travailler avec une seule image raster, plutôt que de travailler calque par calque ( la fusion de shapefile n'est pas évidente)

reprise de la rédaction de la procédure de numérisation,

-présentation du code modifié

-discussion avec M,Piry et M.Le Fur sur les méthodes à employer pour la conversion des système de projection en fonction des zones choisies et de leur surface ( problématique d'un terrain courbe que l'on ne peux pas exprimer en termes de mètres sans erreurs, la conversion latitude longitude pose problème car elle entraine des déformations.

-reprise de la procédure de clipping avec M.Le Fur. Des points sont à modifier

**le 06/03/08**

-correction de bug dans le simulateur

-reprise du rapport

reprise de la carte de kedougou

numérisation de kedougou

jointure et création d'attribut en fonction du type de polygone ,

-implantation de kedougou dans le simulateur

-discussion sur l'évolution de la simulation avec M;Piry et M Le Fur

**le 09/02/09**

reprise du rapport

recherche pour modifier le CRS ( Coordinate reference system) d'une carte pour convertir un format «latitute longitude » vers un système de coordonnées projetées en mètre (UTM)

>>Qgis affiche les coordonnées géographique sous le format longitude -latitude

-recherche d'outil s et de méthode de conversion

-intégration de la carte de kedougou transformé au format utm dans le simulateurs

**le 10/02/09**

-reformulation de la partie jointure

recherche de méthode sous Grass pour éliminer les parasites ( polygone de taille inférieur à un seuil donné)

-utilisation de la fonction v.clean rmarea

**le 11/02/09**

-le shp généré lors de la vectorisation est très lourd, même si on en réduit le nombre d'éléments car, la précision de Grass correspond au contour des formes au format « raster » chaque polygone est donc définit par plusieurs centaines voire millier de point

recherche de traitement gimp pour permettre la simplification des forme généré par sélection de plage de couleur

-filtre => amélioration ( contraste)

=> distorsion ( propagation de valeur) => élargie ou couche pour combler les trous

=> flou : pixeliser => permet de simplifier les formes en pixelisant chaque partie

recherche avec M.Le Fur et M, Piry des solutions parralèle pour vectoriser directement une carte depuis Google earth

reprise de la partie simulation:

-définition d'une interface regroupant les constantes

-création des paramètres de début de la simulations

-création d'une relation entre la taille du terrain (fixé en mètre) la vitesse de l'agent et le pas de temps décidé.

**Le 12/03/09**

-reprise du rapport

-modification des classes du simulateurs pour différencier les rôle ( un agent ne doit pas pouvoir connaître sa position car celle ci dépend de son type de support. Il peut la connaître en appelant l'objet support.

-recherche avec M.Piry d'une procédure de traitement pour permettre de réduire l'épaisseur d'un fichier KML à une seule couche ( on élimine les partie de polygones qui se recoupent ou qui se superposent pour qu'un point ne puisse appartenir qu'à un seul polygone.

**le 13/03/09**

-problème lors de la définition des mouvement aléatoire à l'initialisation , les nombres générés ne sont pas aléatoires...

-modification des classes du simulateur pour permettre une plus grande généricité en utilisant les interfaces comme déclaration des paramètres de fonctions.

- reprise de la numérisation de kedougou depuis Google earth

-résolution d'un problème de compatibilité des fichiers KML généré par Google earth, non compatible avec qgis.

**Le 16/03/09**

numérisation de kedougou

-problème qgis n'ouvre pas le KML généré par Google earth même en changeant l'en tete

-recherche de solution au problème au milieu des 12 000 lignes générés

Solution trouvée par M.Piry des point utilisé comme repères ont été introduit dans le fichier, celui ci ne possède donc plus d'un type de géométrie unique ( point + polygone);

-amélioration de la partie de traitement sous Google earth et de la partie numérisation

-entrainement sous PostGis ( création d'une table, connexion et importaiton sous qgis, requête simple... )

**le 17/03/09**

-réalisation de la procédure de traitement sous PostGis : travail avec M Piry pour écraser les polygones inclus avec une requête SQL

-rendez vous avec M.Le Fur pour décider d'un changement de structure du rapport, il faut découper le rapport pour le rendre plus modulaire et faire en sorte que les maillions de la chaine de traitement soit indépendants.

**Le 18/03/09**

généralisation de la procédure SQL pour la fusion des polygones d'après leur superposition et leur taille.

Reprise de la carte de kedougou pour donné une affinité aux zones d'après leurs nature ( opération de jointure)

mise en place d'une fonction de changement de CRS lors du traitement PostGis.

Reprise du simulateur en fonction des attributs propres de kedougou, pour que les agents se regroupent selon l'affinité qu'ils ont avec leur zone.

**Le 19/03/09**

reprise d'un shapefile de chize avec redéfinition des catégories et association avec une affinité, intégration dans le simulateurs.

Modification du code pour le rendre plus générique

**le 20/03/09**

Correction de la première partie du module numérisation des données.

**le 23/03/09**

-reprise du code pour que le simulateur puisse intégrer comme support une grille sans avoir à redéfinir la gestion des agents.

-recherche sur le chargement de fichier et la lecture des flux a travers un buffer.

Reprise du rapport sur la partie « géoréférencement d'un raster »

**le 24/03/09**

reprise de la partie « Vectorisation d'une image de type raster » et traitement des shapefiles.

correction et ajout de précision sur la partie concernant la suppression des superpositions.

reprise de la programmation de la grille

**le 25/03/09**

discussion avec M. Piry pour modifier la gestion des agents dans un espace continu dont les propriétés sont reliées à une grille d'agent terrain. Essai d'optimisation des parcours de listes à l'aide d'une table de hachage.

**Le 26/03/09**

reprise des parties traitement clipping et traitements pour le changement de CRS

Modification du programme Java pour mettre en place le traitement d'un support double ( grille et plan continu)

**le 27/03/09**

modification du programme pour pouvoir charger une image dans un format compatible et récupérer son raster pour l'utiliser comme grille.

Test du programme sur des cartes raster: un problème apparaît, le traitement est lourd car Repast effectue un parcours sur chaque case du raster à chaque tic de temps.

**Le 30/03/09**

Recherche pour modifier la solution précédente. Plutôt que l'utilisation d'une grille, on va utiliser une GridValueLAyer pour représenter les données. La partie de traitement des agent se fera dans un matrice d'agent parcelle qui ne sera pas représentée.

Rédaction du tutoriel sur Repast Simphony.

**Le 31/03/09**

Mise en place de la solution utlisant le Grid Value Layer. On ne lit plus les valeur calorimétrique des pixels d'un raster mais les valeurs contenues dans le raster. On construit ensuite une gridvalue layer à partir des valeur récupérée et on l'associe à une classe de style pour pouvoir afficher une image en fonction des ces valeurs

Rédaction de la partie « gestion des agents ».

**01/04/09**

reprise du code du simulateur pour récupérer le colormodel de l'image de départ et le transformer pour conserver la correspondance des couleurs lors de l'affichage dans le simulateur (conversion d'une color model en « hastable » pour faire la correspondante.

Modification du code qui permet de réduire les parcours. Les agents parcelles peuvent contenir des agents mobile. Le ground Manager fait des parcours plus petit pour pouvoir déplacer les agent mobile d'un agent parcelle à un autre.

Cette fonction permet de réduire les parcours pour savoir quels agents sont visibles par un autres agents.

**02/04/09**

rédaction du rapport de stage

correction sur la partie gestion des agents

**03/04/09**

Conception d'une procédure permettant de réduire la résolution d'un raster pour l'utiliser dans une simulation

utilisation des outils Grass et qgis

**06/04/09**

Rédaction de la procédure de changement de résolution d'un raster

rédaction de la partie intégration d'un support raster dans Repast

modification sur la partie gestion des agent pour détailler les cas où on utilise un SIG et celui où on utilise un raster

**07/04/09**

création d'un projet Repast Simphony SimMasto_0 qui réutilise toutes les procédure mise en place pour programmer une simulation. (reprise de l'ancien projet, modification de la structure en package, (data, business, management) changement des nom de classes et de variables, reprise des commentaires dans le code en anglais.

**Le 08/04/09**

création d'un table de hachage pour associé à chaque agent dynamique sa parcelle courante

reprise du rapport , mise, en page, ajout d'illustration.

**le 09/04/09**

reprise du code de l'application créant des carrés KML pour gérer certaines exeptions.

reprise du rapport chaine de traitement , mise, en page, ajout d'illustration.

**le 10/04/09**

installation de objectering et conceptions de diagrammes uml pour structurer les maillions de la chaine de traitements

reprise du rapport chaine de traitement , mise, en page, ajout d'illustration.

**Le 14/04/09**

rédaction du rapport de stage, mise, en page, ajout d'illustration.

**Le 15/04/09**

reprise de la gestion du gis manager pour utiliser des tables de hachages pour optimiser les parcours sur les agents parcelles et leurs voisins

rédaction du rapport de stage, mise, en page, ajout d'illustration.

**Le 16/04/09**

reprise du SIG manager, remplacement de toute la gestion des tables de hachages avec des objets hashtable existant.

On n'effectue plus ni de parcours ni de test, l'objet hashtable s'occupe de gérer les structures.

rédaction du rapport de stage, mise, en page, ajout d'illustration.

**Le 17/04/09**

rédaction du rapport de stage, mise en page

=> création d'un dossier regroupant des fichiers utilisés lors de la réalisation de la chaîne de traitement.

**le 20/04/09**

Remise d'une première version complète du rapport de stage

=>La structure est à revoir, des paragraphes sont à supprimer ou fusionner, les grandes lignes ne sont pas assez démarquer

=> reprise de l'ensemble du rapport de stage

**le 21/04/09**

=>finalisation de la seconde version du rapport de stage

=> remise d'une version 2

=> discussion sur l'ordonnancement des modules de la chaîne de traitement et réflexions sur d'éventuelles extensions pour un gestion plus complète des rasters. La partie raster est assez brève, mais la reconnaissance des terrains doit être fait par une expertises. De plus il y a tropde format de raster différent pour pouvoir adapter simplement la chaîne, il sera peut être nécessaire de créer une nouvelle chaîne de traitement permettant de convertir les données contenue dans un raster à un format acceptable.

**Le 22/04/09**

recherche et mise en place d'un code permettant de créer une colormap pour une image en niveaux de gris.

modification du code dans la classe Raster manger pour tester la nature de l'image lue et récupérer ou créer une colormap en fonction

**Le 23/04/09**

Fin de la rédaction du rapport, dernière mise en pages

# Résumé

Ce rapport présente une chaîne de traitement développée dans le cadre de mon stage de deuxième année d'IUT informatique à Montpellier.

Cette chaîne de traitement s'inscrit dans un projet de création d'une plateforme de simulation au centre de recherche CBGB[1] de Montferrier ; elle a pour ambition de détailler toutes les procédures permettant d'intégrer des informations spatialement référencées en tant qu'élément d'une simulation.

A terme, le travail s'insérera dans la plateforme et permettra de développer des simulations multi-agents exploitant une variété de supports spatiaux géoréférencés.

Les résultats obtenus sont la réalisation des procédures constituant la chaîne de traitement, la création de simulations programmées en Java pour illustrer les résultats et mettre en place une structure permettant des interactions entre les agents et leur support de manière générique et enfin la rédaction de la chaîne présentant les étapes réalisées.

Mots clés : Système d'information géographique (SIG), simulation multi-agents (SMA), Java, fichier de formes (shapefile), rasters, chaîne de traitement logiciel.

# Summary

This report describes a processing chain realized in the frame of my training period completed during my second year at the IUT of Montpellier.

The chain is part of a project which aims at the development of a simulation platform at the CBGP research centre of Montferrier. The training period objectives were to elaborate, develop and document all procedures allowing integration of spatial referenced data as a geographical support to simulation's elements.

This work will eventually insert itself in the platform and will allow developing multi-agents simulations using several georeferenced spatial supports.

The results obtained are the realization of procedures constituting the treatment chain, the creation of a structure allowing interactions between agents and their support in a generic way and finally the documentation of the chain explaining the different realized stages.

Key Words: Geographic Information System (GIS), Agent Based Modeling and Simulation (ABMS), Java, shapefiles, rasters.

[1] CBGP : Centre de Biologie et de Gestion des Populations