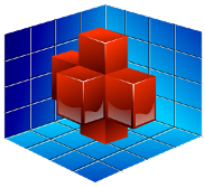
IRAMAT-CRP2A, Université Bordeaux Montaigne
Maison de l'Archéologie, Esplanade des Antilles
33607 Pessac Cedex, France



Rapport de stage

Développement sous JAVA de modules pour l'analyse de données isotopiques

Stage réalisé par Guitard Alan,
sur la période du 17 Mai 2016 au 31 Juillet 2017

Table des matières

Remerciements

Je tiens à remercier Norbert Mercier, directeur de recherche au CNRS et maître de stage, pour m'avoir proposer de continuer mon précédent stage à ses côtés et m'accorder une grande confiance et liberté dans ma manière de travailler.
Merci aussi à Loïc Martin, doctorant et développeur des logiciels DosiVox, DosiVox2D et DosiSeed, pour m'avoir accorder autant de son temps pour les explications et le suivi du programme.
Il as été très agréable de travailler avec eux et je les remercie pour leur souplesse.

Introduction et contexte

1.1 Présentation du laboratoire : IRAMAT-CRP2A

L'IRAMAT-CRP2A (Institut de recherche sur les Archéomatériaux – Centre de recherche en physique appliquée à l'archéologie) est un laboratoire bordelais associé au CNRS qui étudie les sociétés anciennes par l'étude des matériaux archéologiques et de leur environnement et nourrit des thématiques basés sur le peuplement humain des sociétés anciennes, sur l'usage de leur ressources, sur l'histoire de l'art et sur leur techniques. Dans le cadre de ces recherches au carrefour de la science humaine et de la science de la matière, les chercheurs sont amenés à développer des outils d'analyses et de datation des matériaux, de simulations informatiques ou de traitements statistiques inédits en vue de l'analyse chronologique d'événements archéologiques.

Leurs recherches s'organisent autour de trois grands axes :

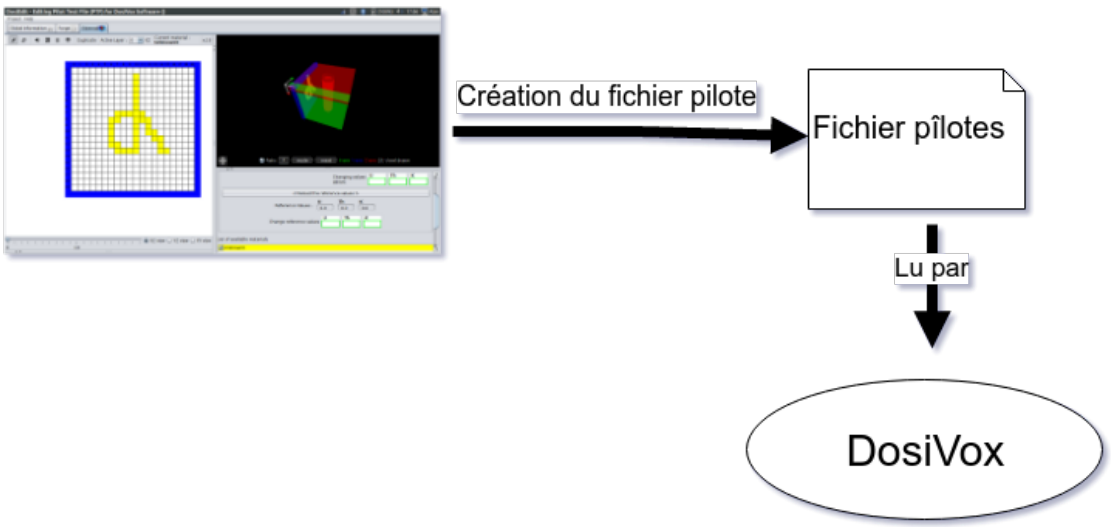
- Chronologie, Référentiels : Implantations Humaines et Paléoenvironnement,
- De la source à l'objet : ressources, diffusion, technique, altération, conservation,
- Architecture du bâti monumental : Histoire, Constructions, Matériaux, Art.
  Pour plus d'informations, voir le site internet officiel du laboratoire. [3]

1.2 Contexte de développement

1.2.1 Objectif

Ce stage s'inscrit dans la continuité d'un précédent stage réalisé dans les mêmes conditions et sous la supervision des mêmes chercheurs, Norbert Mercier et son doctorant, Loïc Martin, qui a développé le logiciel DosiVox dans le cadre du projet Dosi-Art. L'objectif était de développer une interface Java permettant d'éditer de façon simple des fichiers pilotes qui seront lus par le logiciel DosiVox[2]. Dans ce nouveau stage, l'objectif est de développer une suite logiciel de simulation dosimétrique dans la droite lignée de l'interface de DosiVox, ainsi qu'un plug-in ImageJ. Toutes ces interfaces doivent générer des fichiers textes qui piloteront leur logiciel respectifs.

FIGURE 1 – Exemple avec DosiVox



Les trois interfaces de dosimétrie sont destinés à être en libre téléchargement sur le site de l'IRAMAT-CRP2A à côté des liens des logiciels DosiVox, DosiVox2D et DosiSeed. Des colloques ont été organisés par Loïc afin d'introduire le fonctionnement des interfaces aux chercheurs qui s'en serviront.

1.2.2 Environnement de développement

Le développement a été réalisé sur l'environnement de développement Eclipse, sa version 4.6.0.

1.2.3 Équipe et locaux

Le laboratoire ne possédant pas de pôle informatique de développement, j'ai travaillé dans la salle des stagiaires prévu à cet effet. J'ai travaillé seul sur l'écriture et nous faisions des points réguliers sur les avancées, les fonctionnalités à apporter et sur l'ergonomie des interfaces.

Interface de DosiVox : DosiEdit

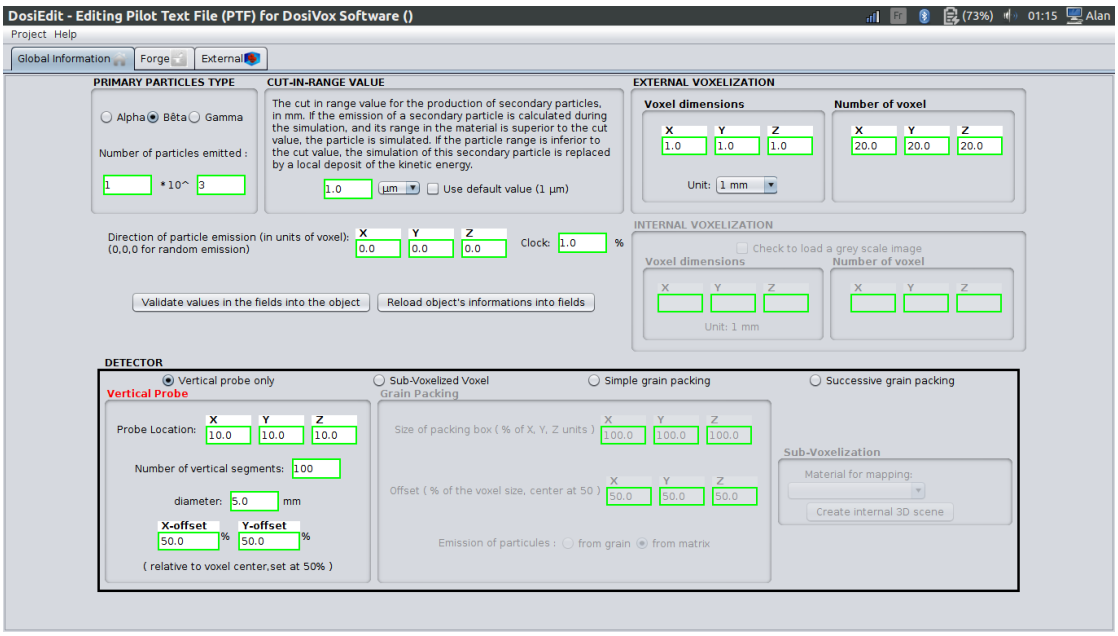
2.1 État de l'art

Présentation de DosiVox et DosiEdit DosiVox [2] est un logiciel de simulation dosimétrique, en ligne de commande, utile pour les Methode de datation par luminescence et détermine les débits de doses utiles pour le calcul de l'âge.

DosiVox prend en entrée un fichier texte qui contient des paramètres globaux de description du monde, des informations sur les matériaux et les composants créés et des matrices représentant le monde voxelisé, matrices de nombres qui sont l'indices du matériau dans la liste avec lequel le voxel est rempli. Ce fichier pilote pouvant être potentiellement très long et donc très fastidieux à écrire, l'interface a été codé dans le but d'écrire ces fichiers avec simplicité.

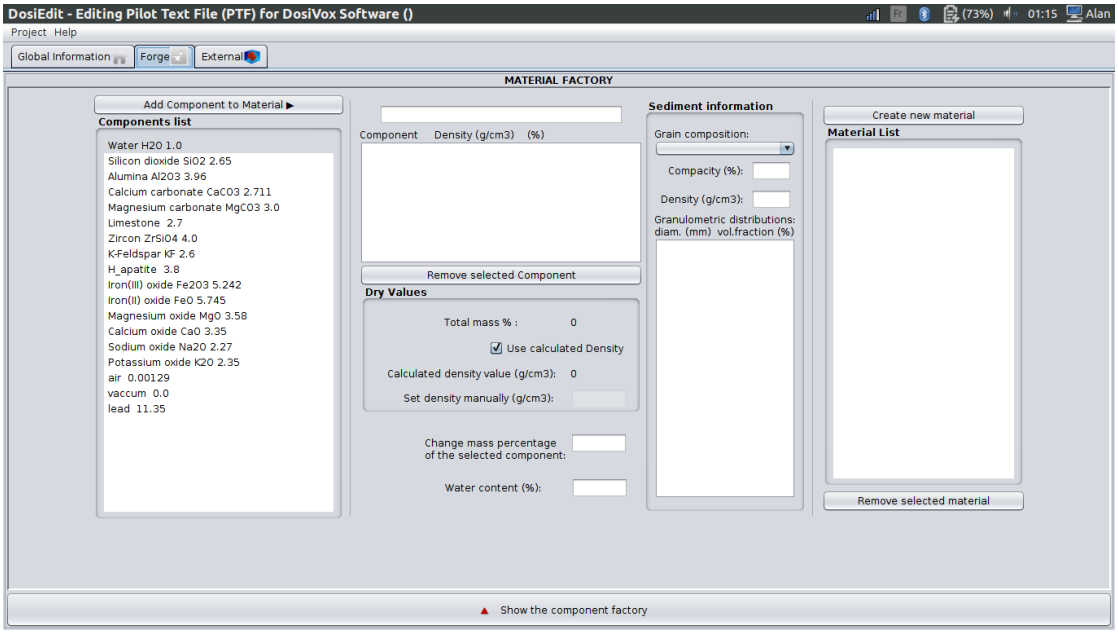
Description de l'interface L'interface est constitué de trois onglets. Le premier, "Global information" propose de paramétrer cer-

FIGURE 2 – Onglet d'information globales



taines valeurs globales du projet, valeurs qui n'ont pas d'impact à l'édition du monde mais à la simulation par DosiVox. Le deuxième onglet est appelé la "Forge". C'est ici que l'on créé des nouveaux composants qui constitue les matériaux que l'on peut également créé. A propos de la création des composants, l'interface propose de pouvoir cliquer sur les cases du tableau de Mendeleïev afin de définir la formule chimique du composant nouvellement créé. 18 composants largement utilisés par les chercheurs sont déjà créés. Enfin, le troisième onglet, appelé "Editor", montre la grille éditable et la vue en trois dimensions. Ici, on sélectionne un matériau avec lequel travailler, on modifie ses paramètres radio-actifs, et on peut alors remplir le monde avec ce matériaux en cliquant sur la grille qui représente la tranche sélectionné.

FIGURE 3 – Onglet de forge



2.2 Cahier des charges

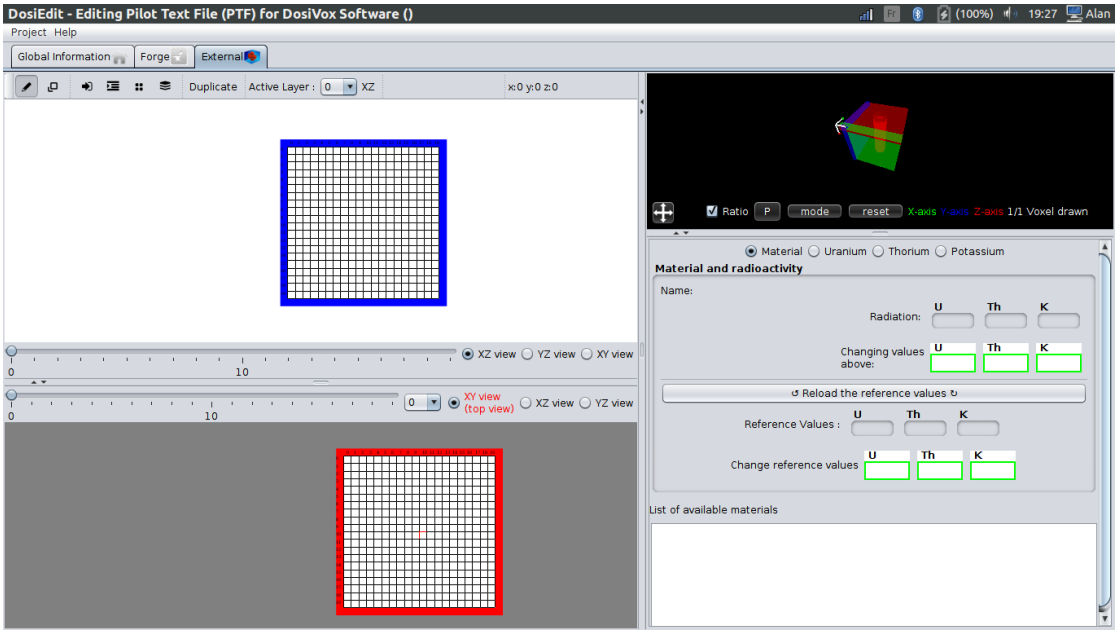
Correction de bugs Des bugs d'affichages ou de calcul bruts persistaient, il devaient donc être corrigés.

Amélioration de l'inter-compatibilité L'interface a initialement été compilée avec la version 1.5 du Kit de développement d'Oracle. Des problèmes de compatibilité avait été détectés sur les ordinateurs avec l'environnement d'exécution de version 1.7 ou supérieur. Ces problèmes se traduisaient par une diminution très conséquentes de la fluidité de l'interface.

L'interface doit donc être recompilée avec une version plus récente du kit de développement.

Mauvaise écriture de fichiers pilotes Les fichiers pilotes contenaient parfois des erreurs sous de rares conditions, rendant inutile l'uti-

FIGURE 4 – Onglet d'édition



lisation de l'interface puisque que le fichier généré était illisible par DosiVox. Il était primordiale de régler ces points rapidement.

2.3 Développement

Compatibilité des versions de Java Des tests ont été effectués à propos des différentes version du SDK d'Oracle, pour savoir avec quelles environnement d'exécution elles étaient compatibles. Le code source a été codé en version 1.7, la dépendance envers cette version de l'environnement d'exécution devient inévitable. La diminution de la fluidité de l'interface venaient des utilisateurs qui utilisaient le version 1.6, version qui n'est pas mis à jour par les dernières mise à jour de sécurité.[1]

La version 1.7 est la version minimum a utilisée. Oracle, ayant

1. `http://www.oracle.com/technetwork/java/javase/downloads/java-archive-downloads-javase6-419409.html`

sortie une dernière mise à jour de la versione 1.8 le 19 Juillet 2016,[2] recommande d'utiliser cette dernière version dans le cadre d'un développement. Toutes les interfaces seront donc compilé avec le kit de développement, et un message d'avertissement sera affiché à l'ouverture du logiciel dans le cas où l'environnement d'exécution est de version 1.6 ou moins.

Correction de bugs Cette interface DosiEdit, en tant que développement pur, n'as fait l'objet que de corrections de bugs ou de réécriture de codes. La grande partie du stage était consacré aux autres interfaces. Un exemple de bug était les caractères autorisés par les champs de saisie de texte, qui devait n'accepter que des nombres entiers ou flottants correctement écrits. Or, les champs autorisaient plusieurs points (ou virgules), ce qui déclenchaient une exception de type `NumberFormatException` qui n'était pas traités.

Les fichiers pilotes avait quelques erreurs, comme le nombre de nouveaux composants toujours à 0, des espaces manquants pourtant indispensable à la compréhension du fichiers par DosiVox, ou encore des erreurs de cohérence entre les données (le nombre de composant doit par exemple correspondre au nombre de composant décrit après ce nombre).

Pour en savoir plus sur le développement de DosiVox, le rapport de stage correspondant est disponible en téléchargement à l'adresse disponible dans la bibliographie [1].

2. `https://www.java.com/fr/download/faq/release_dates.xml`

DosiEdit2D : Une version 2D de DosiEdit

3.1 Cahier des charges

3.1.1 Besoin principal

DosiVox2D, aussi écrit par Loïc Martin, est un logiciel identique à DosiVox, il pratique des simulations dosimétriques sur un monde décrit par un fichier pilote, mais il est spécifique à la simulation sur des mondes représenté par des matrices 1xn (où n est un entier), sur des tranches donc.

Il est possible de faire ce genre de simulation sur DosiVox mais beaucoup de données, même globales, sont alors inutiles. Une interface simplifiée de DosiEdit est donc envisagée, nous utiliserons donc DosiEdit comme base de travail.

3.1.2 Simplification des paramètres globaux

- **Direction des particules :** Dans DosiVox, il fallait indiquer un vecteur 3x3 de direction d'émission des particules (Alpha, bêta ou gamma). DosiVox2D est assez simplifié pour ne pas avoir besoin de cette information, même par un vecteur 2x2, on peut donc supprimer ce champs.
- **Choix de l'élément radio-actif à simuler :** DosiEdit, à la génération des fichiers pilotes, en créé quatre. Les matrices qu'ils décrivent sont les mêmes mais les valeurs de radiations diffèrent car l'un décrit la simulation de l'élément radio-actif potassium, un autre l'uranium, le troisième le thorium et un quatrième appelé "user-defined" que l'utilisateur peut définir à sa guise. Une amélioration a apporté est de permettre à l'utilisateur quel

fichier générer, plutôt que de créer systématiquement les quatre.

- **Sondes :** DosiVox simule une sonde au milieu de son monde où récupérer les informations utiles pour la recherche. On peut y choisir un type de sonde parmi quatre : une sonde vertical, un monde voxelisé à l'intérieur d'un voxel et les deux dernières sont des sondes à paquet de grains, l'une simple et l'autre à paquet de grains successifs. Aucune précision ne sera apportées ici pour ces trois dernières sondes et pour cause, elle sont inutiles à DosiVox2D qui n'utilise qu'une sonde vertical.
- **Forge :** Étant donné que la seule sonde possible dans DosiVox2D est la sonde verticale, les matériaux n'ont plus utilité de contenir des informations de grains, ou sédiments. Ces informations seront donc supprimer de la forge.

3.1.3 Visualisation

S'affranchir des données en trois dimensions Étant donné que le monde devient en seulement deux dimensions, nous pouvons s'affranchir de la visualisation en trois dimensions réalisé avec OpenGL dans DosiEdit. Nous pouvons donc supprimer la dépendance à JOGL et tout les composants Swing qui y référait.

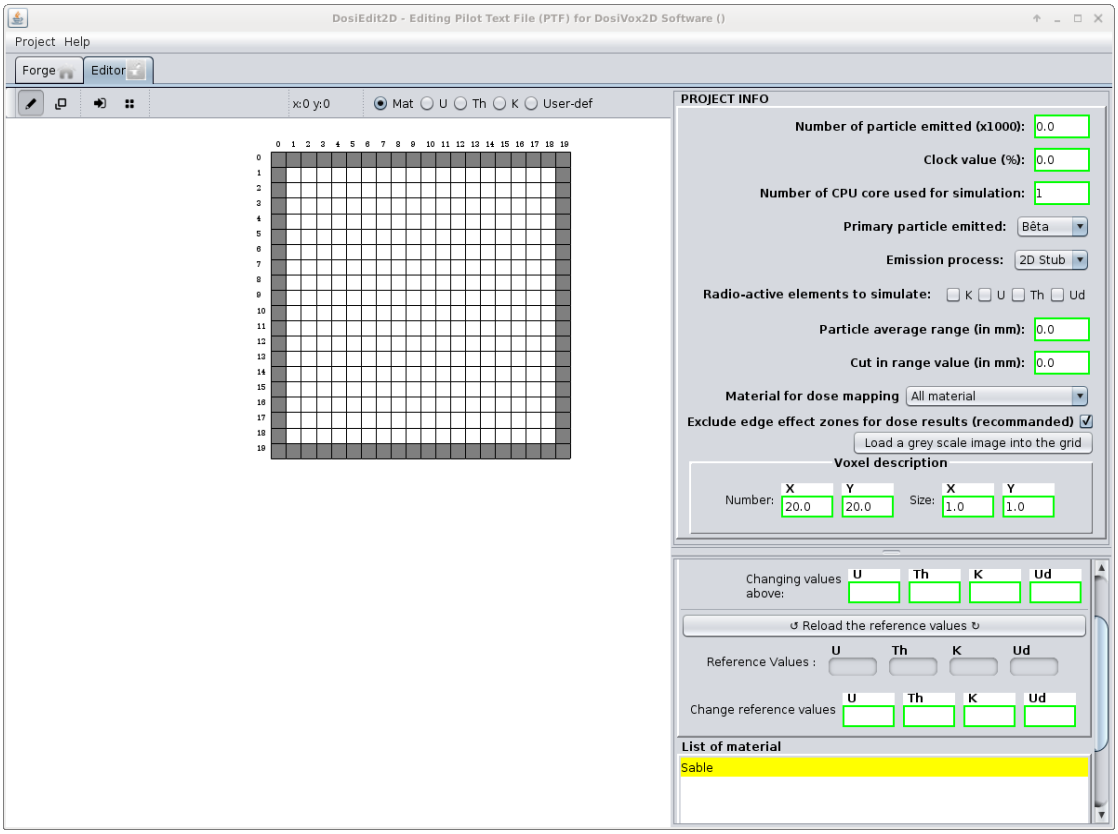
Pour la même raison, la deuxième grille qui visualisait une deuxième tranche dans DosiEdit (celle du dessous sur la figure 4) peut être également supprimée.

3.2 Développement

3.3 Étude technique avant écriture

Une auto-critique a été réalisé sur le développement de DosiVox. En effet, aucun patron de conception n'as été suivi dans son développement. Plus de 10000 lignes de codes ont été écrites pour ce logiciel, la maintenabilité s'en ait retrouvé compliquée et la robustesse était diminuée.

FIGURE 5 – Onglet d'édition de DosiEdit2D



Également, le logiciel n'as pas été pensé pour avoir une suite logiciel susceptible d'utiliser ses composants. Très peu de modularité n'as été pensé et le couplage est bien trop fort pour être reproduit dans d'autres interfaces.

Le temps de développement qu'un ré-usinage de DosiVox susciterai serait important et ce n'est pas le sujet du stage, nous commencerons donc à faire attention à ces points à partir de DosiEdit2D.

3.4 Patrons de conception

Pour les composants Swing, il est naturel d'utiliser le patron de conception Observer car les classes écrites par Oracle ont été pensées

pour ce paradigme. Aussi, quand un logiciel interface-utilisateur est envisagée, le réflexe acquis par le cours en Approche Objet que j'ai suivi est de penser au patron de conception Modèle-Vue-Contrôleur, patron qui offre une modularité acceptable et une cohérence des classes très adaptée.

3.5 Structures de données

Pour cause de la quantité de donnée potentiellement grande manipulable par l'interface en raison du monde en trois dimensions, une étude avant l'écriture de DosiEdit a été faite des structures de données proposées par Java (`HashMap`, `ArrayList`, `LinkedList`,...). Une `HashMap` de ?`HashMap` de `HashMap` a été la solution la plus optimale pour représenté le monde en trois dimensions. La même solution sera utilisée pour le monde en deux dimensions de DosiEdit2D, mais en enlevant une dimension du tableau.

DosiSeed : Étude de grains dans un milieu

4.1 Présentation

DosiSeed est un logiciel, également en ligne de commande avec en entrée un fichier pilote comme ses prédécesseurs, qui simule des radiations sur des grains de matériaux dans un milieu. Ce ne sont plus les voxels qui irradie le monde autour mais des grains sphériques de différentes tailles et types dispersés aléatoirement ou de manière homogène dans leur matrice.

4.2 Cahier des charges

4.2.1 Similitude avec les autres interfaces

L'interface DosiSeed s'inscrit elle aussi dans la suite logiciel dosimétrique, elle doit donc avoir les mêmes besoins fondamentales :

- Maniabilité familière : L'interface doit se comporter en termes d'ergonomie comme DosiEdit et DosiEdit2D. Les classes de composants graphiques doit être réutilisés au mieux pour ne pas déstabiliser l'utilisateur qui s'attendrait à la même ergonomie générale.
- Possibilité de créer des matériaux : La forge doit être la même que pour DosiEdit2D. La même classe Java doit être réutilisée.
- Création de fichiers pilotes : L'interface doit pouvoir générer les fichiers pilotes lisibles par DosiSeed, en respectant les règles de lectures.

4.2.2 Points à apporter

L'interface de DosiSeed n'as aucun besoin de visualisation en trois dimensions ou en deux dimensions. Nous pouvons donc supprimer ces grilles et les remplacer par ce qui décrira les grains, leur différentes tailles, leurs différents matériaux qui les composent et leur taux de radiation.

Dans le fichier pilote que doit généré l'interface, DosiSeed attend ces informations sous formes de tableaux.

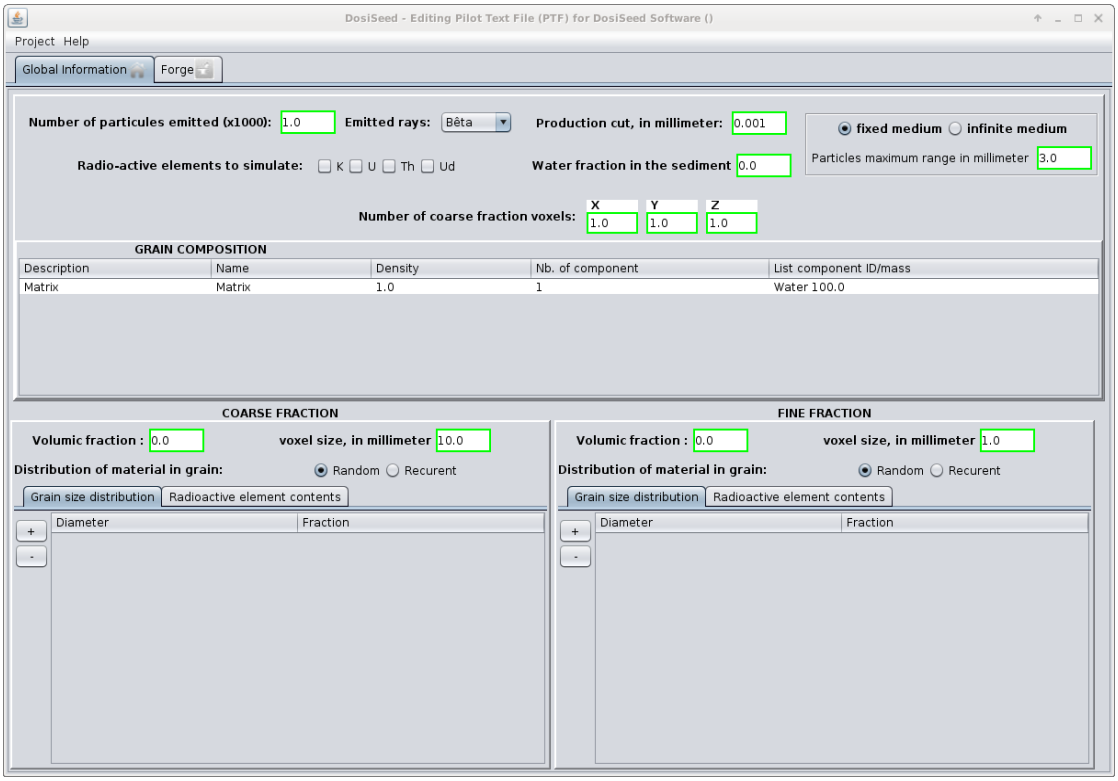
4.3 Développement

4.3.1 Description rapide de l'interface

L'interface contient deux onglets :

- La "Forge" : La forge est la même que DosiEdit2D (Comme expliqué dans la section 3.1.2).
- L'onglet d'information globales : Voici l'onglet (en image à la figure 6) qui permet non seulement de paramétrer les valeurs

FIGURE 6 – Onglet d'information globales



globales du projet, mais aussi les paramètres des grains, présentés en tableaux.

4.3.2 Exploitation de la modularité des classes

La "Forge" fait l'objet d'une classe partagé par DosiEdit2D et DosiSeed. Pour des soucis de références croisées entre les classes de ces deux interfaces, toutes les classes partagés ont été définis dans le projet de DosiSeed. [3]

Il est donc question de la classe `ForgeWindow` mais aussi de cer-

3. Les références croisées arrive lorsque deux projets utilisent des classes de l'autre, ce qui à la compilation entraîne une boucle, qui s'évite en plaçant un des deux projets comme source de toutes références.

taines classes graphiques tel que `TripleFormattedField`.

4.3.3 Description des tableaux

Il sera ici objet des tableaux de description de grains, disponible au moyen d'onglet au sud de l'interface montré en figure 6. Pour chaque type de grains (un type représente un matériau), les différentes tailles de grains possibles sont décrits. Enfin, pour chaque taille de grains, leurs proportions dans le milieu en fonction des autres tailles sont décrites également. Ceci fait office d'un premier tableau, à noter qu'un type de base appelé "matrix" est obligatoire car il décrit le milieu dans lequel sont les grains.

Le deuxième tableau décrit la radio-activité de chaque type (en Potassium, Uranium, Thorium et en particule défini par l'utilisateur).

Toutes ces descriptions sont proposés pour deux échelles de grains, une fraction grosse et une fraction fine. La fraction grosse est simplement la description dans tout le milieu voxelisé et la fraction fine leur description dans un seul voxel (cette description est donc homogène voxel à voxel).

Plug-in Java sous ImageJ

5.1 Présentation du logiciel ImageJ

ImageJ est un logiciel open-source développé par NIH Image conçu pour la rechercher scientifique et pour la visualisation d'image multidimensionnelle. Il permet de pratiquer des opérations ou des traitements sur des images par le biais de nombreux plug-in et macros que les utilisateurs sont libres d'intégrer et de partager, ce qui en fait un outil

indispensable pour la science. Il a pour toute dépendance la machine virtuelle Java en version 1.6 ou supérieure.

5.2 Cahier des charges

Loïc Martin utilise ImageJ pour examiner et observer des images de coques d'œufs d'autruche afin de les dater par leur taux en radio-activité. Pour faire ressortir les zones de radio-activité, il utilisait une routine pour calibrer l'image, pour pouvoir "la faire parler" :

- Conversion de l'image en 16 bit : C'est une prérogative pour les fonctions utilisés après.
- Calibration : Permet de calibrer les pixels pour replacer l'intervalle des valeurs à partir de 0.
- Agrandissement : Permet d'agrandir et de réduire l'image en gardant le ratio.
- Application de filtre : Permet d'appliquer un filtre à l'image.
- Contraste : Permet de changer le contraste de l'image.

Toutes ces fonctionnalités sont déjà natives à ImageJ mais Loïc devait allait réaliser cette routine pour chaque image, et comme toutes ces fonctionnalités sont à beaucoup d'endroits différents de l'interface ImageJ, cela devenait long et fastidieux. Un plug-in pour réaliser cette procédure automatiquement était nécessaire.

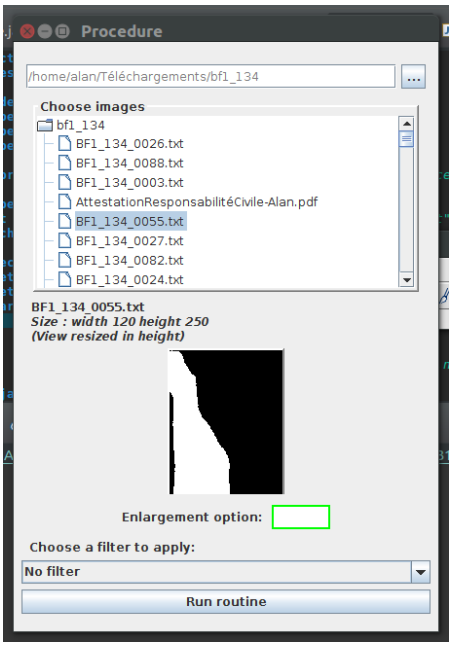
5.3 Développement

5.3.1 Environnement

Le développement de ce plug-in s'est fait surtout par l'apprentissage du code d'ImageJ, par sa structure et l'application de l'environnement de développement.
Pour ce faire, une version source du projet ImageJ doit se télécharger sur leur site et en faire un projet Eclipse. Un deuxième projet Eclipse doit alors être ouvert dans le même espace, qui contiendra les sources du plug-in à créer, et un script Ant[4] doit être lancé pour placer l'exécutable du plug-in dans le dossier "plug-in" d'ImageJ.

5.3.2 Procédure

FIGURE 7 – Début du plug-in



Le plug-in débute par la fenêtre décrite par la figure 7. Elle permet de sélectionner une image dans un dossier de l'ordinateur. L'image doit s'accompagner à ses côtés d'un fichier du même nom concaténé à la chaîne de caractère "Scale" qui est un fichier qui contient 4 valeurs utiles pour la calibration. Si il ne s'y trouve pas, une fenêtre invitera l'utilisateur à en choisir un ailleurs sur l'ordinateur. L'image s'ouvre ensuite avec tout les paramètres appliqués sur celle-ci, à savoir l'éventuel filtre, l'agrandissement, la calibration, et une fenêtre pour gérer le contraste s'ouvre à ses côtés.

Conclusion

6.1 Améliorations possibles

En termes de fonctionnalités, toutes celle demandées par Loïc Martin et Norbert Mercier ont été intégrés. Les améliorations sur les interfaces se feront au fil des utilisations, lorsque des nouveaux bugs seront détectés ou lorsqu'ils auront des idées de nouvelles fonctionnalités. Mon mail est présent dans la section "About" des menus afin de pouvoir maintenir les logiciels dans le futur, comme pour DosiVox.

Les principales améliorations à apporter sont en termes de structure et de forme. DosiEdit, codé en été 2015, a été écrit maladroitement et avec peu d'expérience. En termes de Génie Logiciel, la structure du code a besoin d'un fort ré-usinage afin d'augmenter la robustesse et la maintenabilité. Pour la structure en termes de packages, il serait intéressant de définir un projet qui contiendra les classes partagés par toutes les interfaces, et les classes de bases, en les réusinant pour augmenter leur ré-utilisabilité. La suite logiciel Dosi aura ainsi sa propre mini API pour tout chercheurs voulant créer une nouvelle interface dans cette suite.

Il serait aussi intéressant d'avoir un installeur de mise à jour automatique et un seul et unique lanceur pour toutes les interfaces, des recherches ont été faites mais le temps manquaient, ces idées ont alors été abandonnés.

6.2 Bilan global

Avoir prolongé un stage d'avant la première master m'as permis de voir tout ce que j'avais appris durant cette année en termes de projet à moyenne échelle. En effet, je me suis rendu compte d'habitude naïve que j'avais, comme l'oubli de penser à la structure de mon code avant de l'écrire. Cela devait m'obliger à travailler sur des fondations tremblantes, c'est pour cela qu'en termes général, aucune classe de DosiEdit n'as été réutilisé.

A propos du plug-in, cette première approche du logiciel ImageJ a été très intéressante. Les scientifiques de divers domaines l'utilisent dans leur recherche, je serais donc amené à le réutiliser car mon projet professionnel est d'appliquer l'informatique au science de la recherche fondamentale, comme cela as été le cas pour ces deux stages.

Références

[1] Guitard Alan. Développement en java d'une interface 3d d'édition de fichiers destinés à piloter dosivox, Été 2016.

[2] FD. DosiVox, logiciel de simulation dosimétrique développé à l'IRAMAT-CRP2A, Janvier 2015.

[3] Site officiel. Présentation de l'IRAMAT-CRP2A, Novembre 2014.

[4] Patrick Pirrotte. The ImageJ Eclipse Howto, Août 2010.

Glossaire

Methode de datation par luminescence La thermoluminescence, un phénomène qui apparaît avec la température, décrit le fait que certaines substances émettent une lueur transitoire après avoir été exposées à un rayonnement ionisant. L'explication simplifiée est de dire que lors de son irradiation, le composé thermoluminescent a "piégé" des électrons dans des défauts de sa structure, et les rejettent quand il est chauffé produisant ainsi de la lumière. Dans son milieu naturel, un composé est constamment irradié par les radio-éléments naturels (principalement l'Uranium, le Thorium et le Potassium) et va accumuler un nombre croissant d'électrons dans ses pièges. La luminescence, la quantité de lumière émise lors de la chauffe, sera donc fonction de la durée d'irradiation du composé dans son milieu naturel. Il est important de noter que si le composé est chauffé une deuxième fois, il n'émettra plus de lumière. Les composés datés par cette technique sont prélevés dans les couches inférieures du sol, et la technique est mise en oeuvre dans un laboratoire à lumière "noire" (en fait, une très faible intensité lumineuse est admise). DosiVox a pour fonction de fournir des informations plus précises sur ces irradiations, grâce à des simulations numériques, en particulier pour traiter des cas complexes pour lesquels on ne peut déterminer de solution analytique. 5