Rapport de stage

Développement d’une évolution d’un générateur de figures de Navon

IUT Informatique de Dijon,

[Titre du cours]

# Remerciements

Je tiens à remercier Madame Poulin Charronnat Bénédicte, la Directrice du LEAD[[1]](#footnote-1) pour m’avoir accepté au sein du laboratoire.

Je remercie les doctorants avec lesquels j’ai passé mon stage, pour leur accueil, leur bonne humeur et pour m’avoir permis de m’intégrer facilement.

Je remercie Monsieur Bard, mon maître de stage, pour sa confiance, pour m’avoir partagé ses connaissances et l’autonomie qu’il m’a offert pendant ce stage, me permettant d’affirmer mes compétences.

Je souhaite adresser mes remerciements au corps professoral et administratif de l’IUT Informatique de Dijon, pour la qualité de l’enseignement et plus particulièrement Madame Sologny pour m’avoir suivi durant ce stage.

Table des matières trop drôle

# Présentation du contexte

## Présentation générale

## Présentation du service

## Définition de la mission

### La problématique

Avant de vous présenter la problématique je souhaiterais vous expliquer ce qu’est une « Figure de Navon » : Une figure de Navon c’est un élément de grande taille (une lettre par exemple) composé d’éléments de plus petite taille comme :

INSERER UNE IMAGE

Une a

Au commencement, ce projet n’était qu’un sujet d’une discussion quelconque entre Monsieur Bard et INSERER UN NOM. Lorsque celui-ci faisait ses figures de Navon, il perdait beaucoup de temps à former lettre par lettre, une figure de Navon. Une expérience ne se faisant pas qu’avec une seule figure de Navon, la préparation lui prenait une éternité. C’est à partir de ce moment-là que se développa l’idée d’un logiciel permettant de faciliter la tâche. N’ayant trouvé aucun logiciel, ils décidèrent de donner ce sujet en tant que projet à ESIREM. Deux premières versions furent développées. Les deux étaient très différentes l’une de l’autre et aucune n’atteignait les attentes voulues.

### La tâche à effectuer

Le projet était initialement nommé : « Développement d’une évolution d’un générateur de figures de Navon » : Je devais reprendre le code des précédents étudiants sur ce projet et l’améliorer afin qu’il atteigne les attentes voulues. Mon projet s’est rapidement renommé « Développement ~~d’une évolution~~ d’un générateur de figures de Navon » : Les codes ne pouvaient être suivis par d’autres personnes que les programmeurs eux-mêmes : aucuns commentaires, le code possédait des fautes. J’avais donc décidé de repartir sur une base nouvelle. L’idée de ce projet est que l’utilisateur ait la possibilité de créer facilement des figures de Navon simples, telles que des lettres de l’alphabet et la possibilité de dessiner une figure de Navon et pour les deux façons, il doit avoir la possibilité de changer plusieurs options, principalement :

* Changer le caractère local
* Changer le caractère local en une « image locale » : Que la figure dessinée soit remplie de petites images.
* Changer la densité : le nombre l’éléments locaux (caractères ou images) utilisés afin de faire la figure

## Méthode retenue

### Les différentes solutions envisagées

Les solutions envisagées étaient celles recherchées et testées par les précédents groupes :

* La création d’une figure de Navon par géométrie analytique : On créer une figure de Navon à partir des coordonnées des éléments utilisés.
  + Par exemple : Un « A » majuscule est composé de trois segments.

INSERER UNE IMAGE

* Une grille avec des cases cliquables correspondantes aux emplacements où des lettres doivent apparaître.

INSERER UNE IMAGE

Il fallait dans un premier temps, tester ces possibilités, trouver comment celles-ci étaient améliorables, s’il y avait moyen d’implémenter le point le plus important : la densité.

La densité était, pour ces essais, le point problématique qui n’a pas été géré.

### Les matériels et logiciels utilisés

Afin de réaliser ce projet, voici ce que j’ai utilisé :

#### Python 3.9

Dans les projets précédents les langages utilisés étaient le Python et le C++. Pour ses bibliothèques diverses et son implémentation simple, j’ai décidé de faire mon projet en Python. La version utilisée est la 3.9, c’est à l’heure actuelle la version la plus stable qu’il existe. Le Python possède plusieurs bibliothèques qui m’ont été très utile dans la réalisation de ce projet, des bibliothèques puissantes et assez simple d’utilisation :

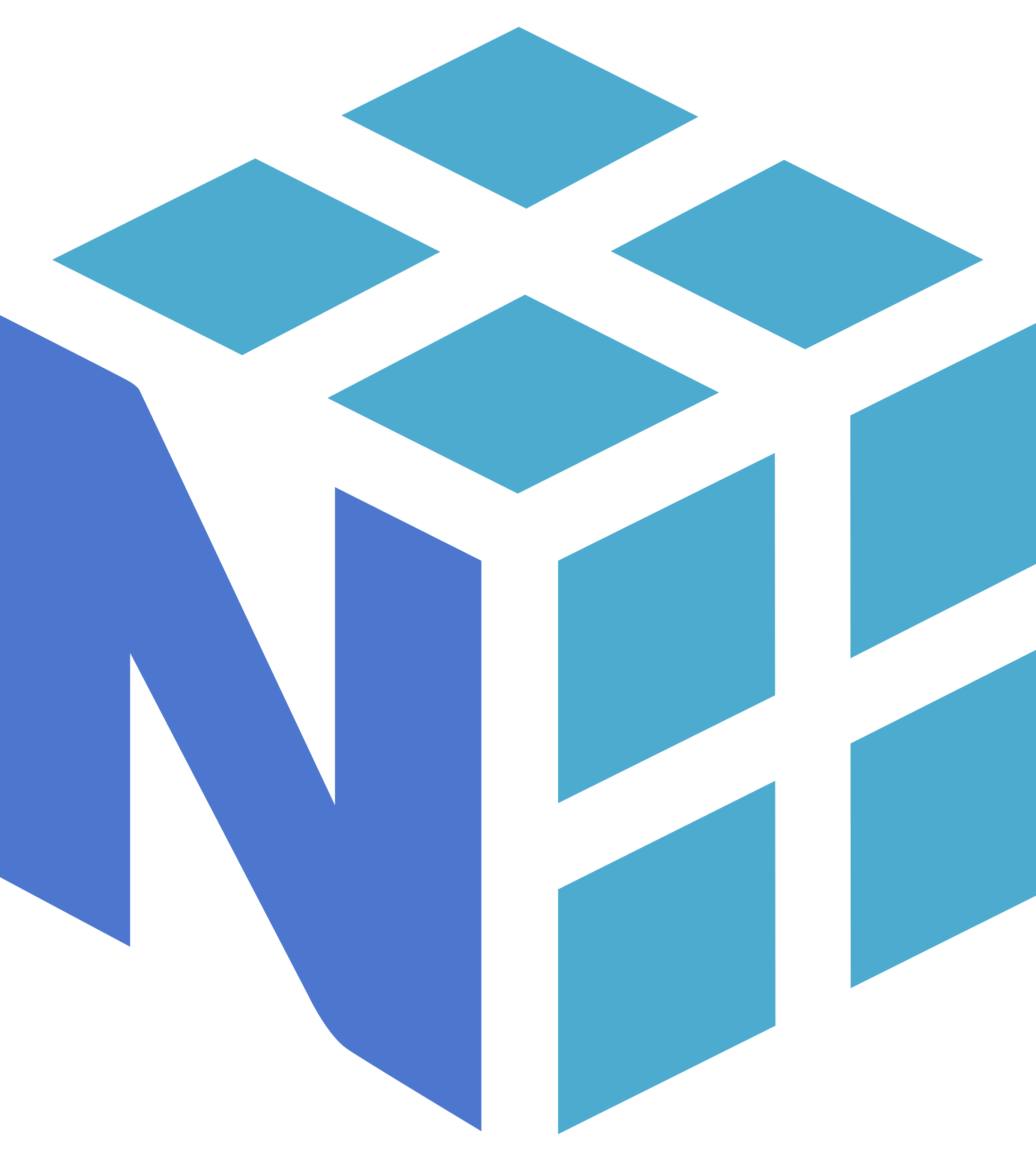


Image 1 : Numpy

Numpy : une bibliothèque permettant la création de tableaux et matrice à N dimensions, propose diverses fonctions mathématiques applicables directement aux tableaux créés et ce, sans réduire les performances du programme.



Image 2 : SciPy

SciPy : une bibliothèque à usage scientifique, utilisant d’autres bibliothèques telles que Numpy citée précédemment. Cette bibliothèque a été utilisé pour sa fonction d’interpolation.



Image 3 : Pillow

Pillow : une bibliothèque de gestion d’image « Python Image Library » me permettant de créer les images sur lesquelles sont dessinés les figures de Navon.



Image 4 : Tkinter

Tkinter : une bibliothèque de graphique, m’ayant permis de créer l’interface homme-machine du programme.

#### Environnements de développements

Le premier EDI utilisé, principalement lors de la phase de test est Pycharm Community Edition :

Pycharm est un EDI utilisé pour programmer en Python, un EDI développé permettant d’analyser son code, d’utiliser un débugueur graphique et d’effectuer des tests entre autres. Il m’a permis, dans les premières semaines de développement, de test et de recherche, d’effectuer beaucoup de test, me permettant de découvrir les différentes bibliothèques citées plus haut.



Image 5 : Pycharm



Image 6 : Sublime Text

Le second outil, celui que j’ai utilisé la majorité du temps est Sublime Text afin de développer ce programme. Sublime Text est un éditeur de texte, simple d’utilisation qui permet une navigation et une visualisation simple des fichiers utilisés.

#### Conception

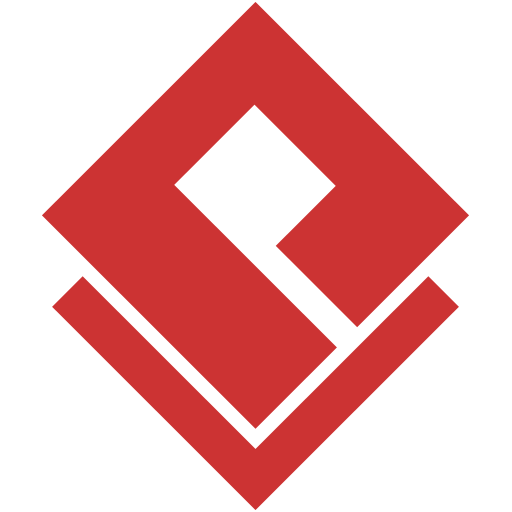


Image 7 : Visual Paradigm

Visual Paradigm : Logiciel me permettant de faire les diagrammes UML[[2]](#footnote-2). Nous avions déjà utilisé ce logiciel en cours afin de créer la conception de projet.

#### Suivi de l’avancée

Afin de suivre la progression globale du projet, j’ai utilisé plusieurs logiciels à ma disposition :



Image 8 : GitHub

GitHub : Afin de faire des sauvegardes régulières de mon code et de faire un pas en arrière si besoin, mon avancée de la programmation du projet est enregistrée dans un dossier que seul moi peut accéder. Ces enregistrements permettent en cas de pépins (comme cela a pu m’arriver) de reprendre à un point précédent sans tout recommencer.



Image 9 : OneNote

OneNote : Le programme de prise de note de Microsoft, me permet de faire un suivi quotidien de mon avancée dans mes recherches.



Image 10 : GanttProject

GanttProject : Un logiciel me permettant de modéliser le diagramme de Gantt de ce projet.

## Application de la méthode et résultats

### Les différentes phases de la réalisation

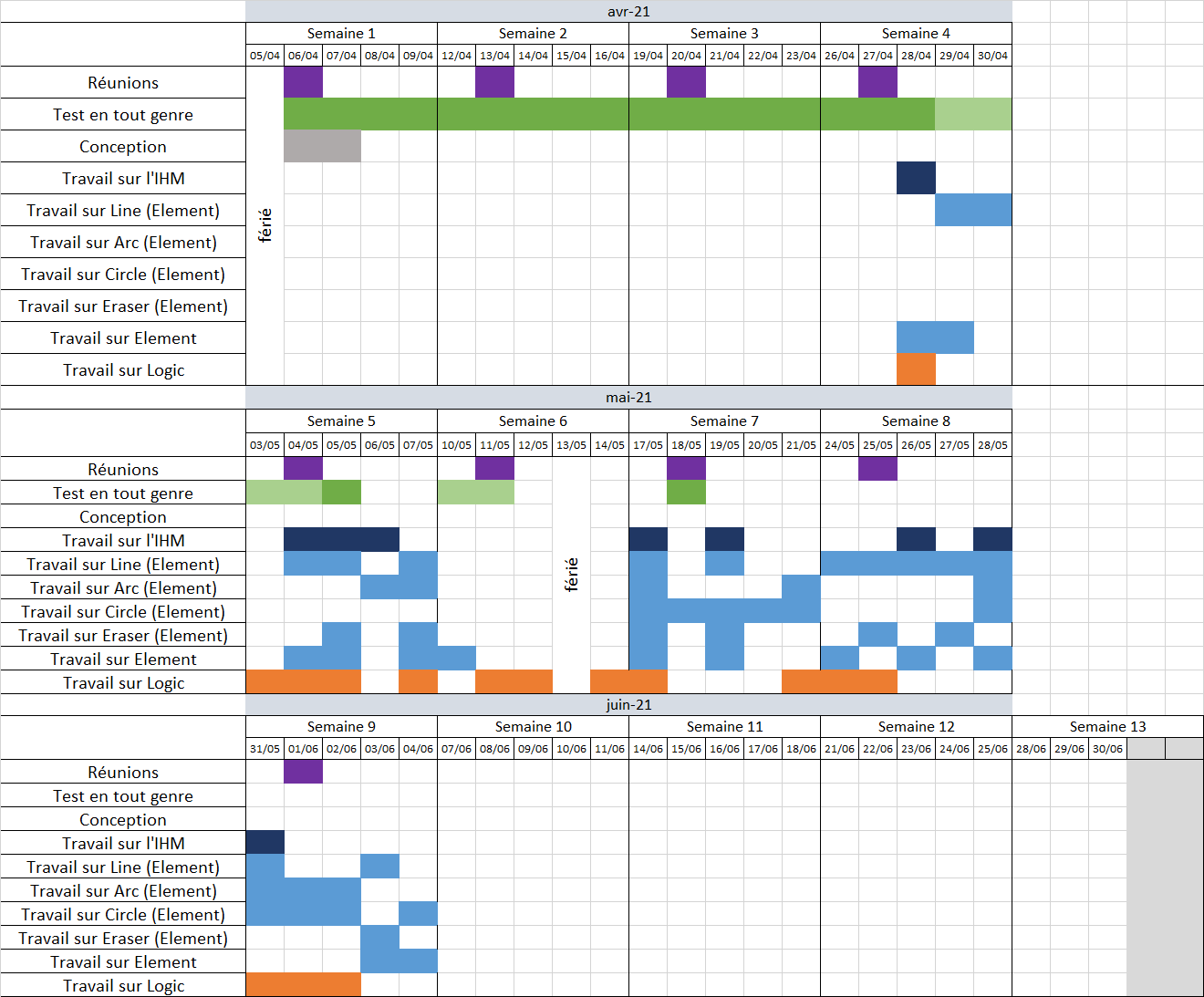


Image 11 : Diagramme Gantt de mon stage

Comme nous pouvons le voir sur ce diagramme de Gantt : mon travail s’est décomposé en deux grandes phases :

1. Une première phase d’une durée d’environ un mois : une phase de documentation, d’essais
2. La seconde phase étant la programmation du projet s’étant de la fin du mois d’Avril à aujourd’hui.

#### Première phase : documentation et essais

Dès notre première réunion, nous avons parlé des solutions explorées précédemment. Le principal problème de ce projet est la gestion d’une densité : le nombre d’élément locaux positionnés dans la longueur d’un élément global (Image 12 : Démonstration densité).

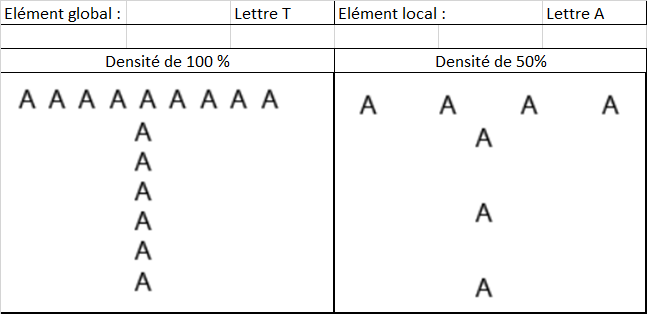


Image 12 : Démonstration densité

La première idée dont nous avons discuté était d’avoir une grille sur laquelle on pouvait cliquer afin de décider des emplacements où faire apparaître des éléments locaux (Image 13 : Exemple grille).

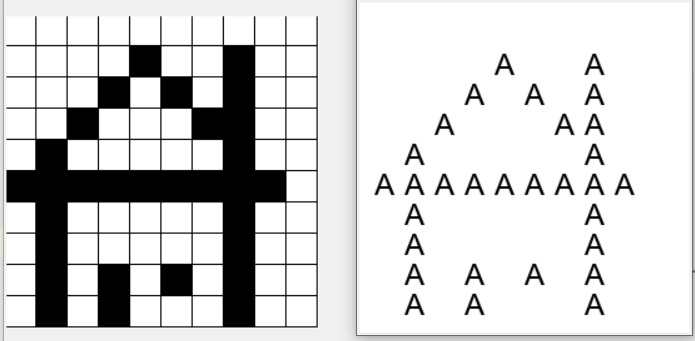


Image 13 : Exemple grille

Mais il était difficile de gérer la densité avec cette grille de dessin : plusieurs idées ont été mentionnées :

1. La première était d’avoir un « masque de densité » : ce masque serait une grille de la même taille que la grille de dessin, et elle serait composée de 1 et de 0 en fonction du pourcentage de densité donné. Et avant de créer l’image, on « superpose » les deux grilles et on ne garde de la grille de dessin que les cases noircies (Image 14 : Exemple densité (50%) par masque).



Image 14 : Exemple densité (50%) par masque

Comme on peut le voir sur cet exemple, le masque peut finir par faire perdre tout sens à la figure de Navon voulue.

### Les difficultés rencontrées

## Conclusion

### Les suggestions pour l’entreprise

### Les leçons tirées de ce travail

1. Laboratoire d’Etude de l’Apprentissage et du Développement [↑](#footnote-ref-1)
2. Unified Modeling Language, Langage de modélisation unifié : langage de modélisation graphique à base de pictogrammes [↑](#footnote-ref-2)