|  |
| --- |
| PROJET INTER-TEXTES |
| Restructuration des documents |
| Demain un Autre Jour |

# OBJET DU DOCUMENT

Le présent document présente les travaux effectué dans le cadre de la tâche 4 qui a pour objectif d’extraire de façon automatique le texte et la structure des documents, et de les exporter au dans un format XML standardisé.

Nous allons détailler dans ce document les différentes étapes de processus ainsi que les principaux algorithmes.

# INFORMATIONS TECHNIQUES

Tout le processus de traitement des documents se fait au travers d’une API développé en C# sous le framework .NET 4.0. Nous nous appuyons sur le SDK d’Adobe Acrobat pour le traitement des fichiers PDF.

# ETAPES

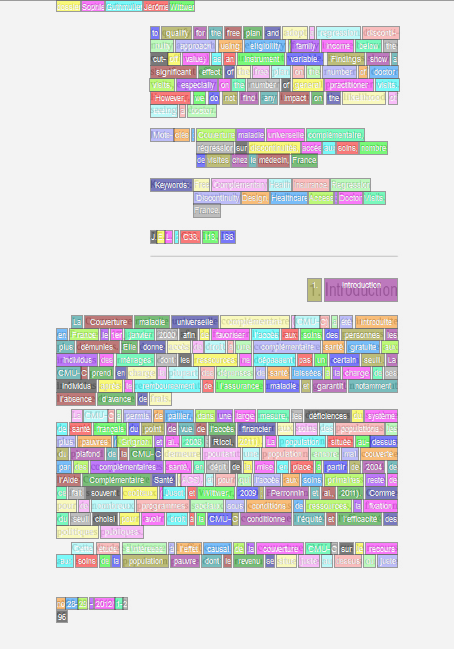
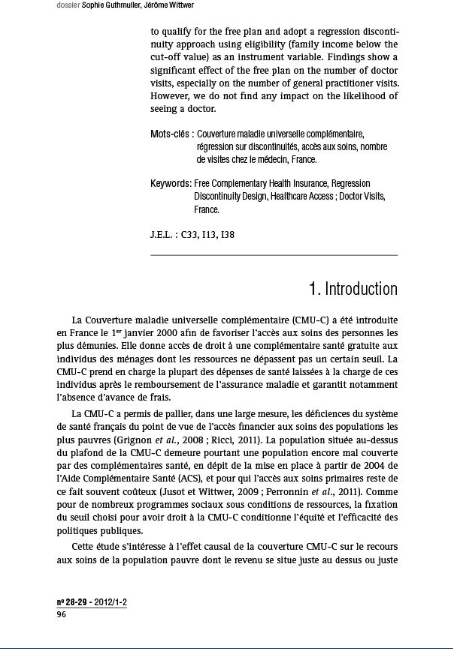
## Extraction du texte

Le texte extrait nous est donné par le SDK d’Acrobat. Nous récupérons une liste de mots avec pour chacun de ces mots différentes informations :

* Coordonnées du bloc contenant le mot.
* Coordonnées des blocs contenant chaque lettre du mot.
* Styles (fontes, couleur, taille, etc.) de chaque lettre du mot.
* Différentes propriétés : dernier mot sur la ligne, orientation, sens d’écriture, etc.

Malheureusement, de par la structure même des documents PDF, cette extraction de mots est imparfaite et nous avons dû développer un algorithme de correction afin d’améliorer cette extraction :

* Suppression des césures : les mots césurés sont extraits avec le caractère de césure, nous détectons les césures et supprimons le caractère indésirable.
* Gestion des ligatures : les ligatures sont présentes dans un seul bloc de caractère mais sont extraites sous la forme de deux caractères ce qui pose problème dans la suite des traitements.
* Gestion des mots tronqués : il arrive que des mots soient coupés en deux lors de l’extraction. Nous avons donc développé un algorithme se basant sur les espacements entre les différents caractères des mots d’une même ligne et les ruptures des styles pour retrouver et corriger les mots coupés.
* Gestion des lettrines : les lettrines sont vues comme un mot à part entière. Nous les détectons et reformons le mot correct.



*Document initial*

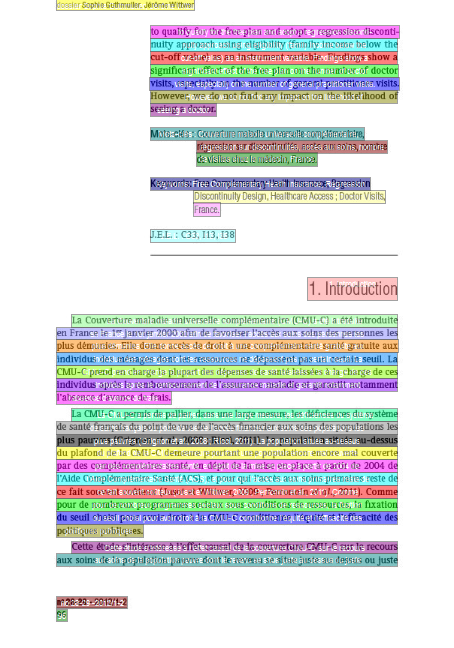
*Extraction des mots*

## Extraction des lignes

Nous regroupons les mots par lignes en considérant qu’un ensemble de mots contenus dans des blocs ayant une même position haute et basse appartiennent à une même ligne.

Si au sein d’une ligne on détecte un espace important entre deux blocs de mots, on crée deux lignes distinctes.

S’il existe une intersection entre deux lignes, on fusionne les deux lignes.

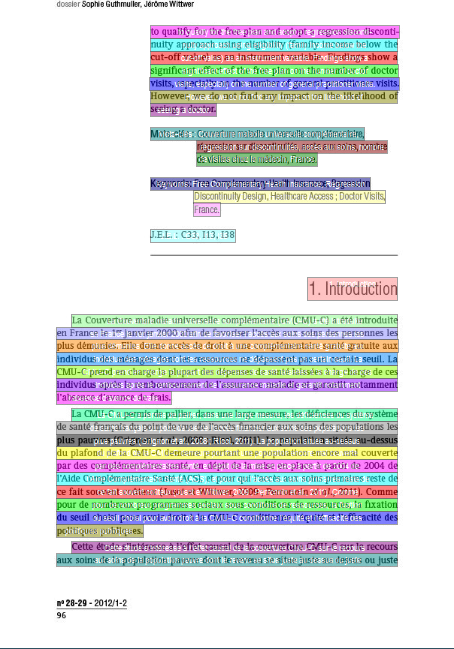


*Regroupement par lignes*

## Suppression des lignes redondantes

Les entêtes et pieds de page des documents provoquent du bruit dans l’extraction du texte. C’est pourquoi nous détectons et éliminons ces lignes suivant cet algorithme :

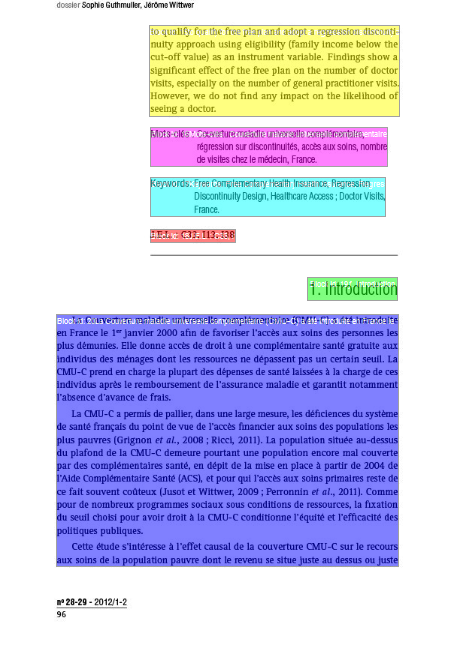
* Constitution de la liste des lignes présentes dans la partie haute ou partie basse des pages (1/5ième de la hauteur de page).
* Pour chacune de ces lignes, on cherche s’il existe d’autres lignes avec un texte « similaire » (algorithme de distance de Levenshtein).
* On regroupe ensuite les lignes similaires trouvées selon leurs positions sur la page. Si un couple « ligne similaire / position sur la page » est présent sur au moins 1/5ième des pages, toutes les lignes situées au-dessus (dans le cas où le couple est en partie haute de la page) ou au-dessous (dans le cas où le couple est en partie basse de la page) sont ignorées pour le reste du traitement.



*Suppression des lignes redondantes*

## Regroupement des lignes en blocs

Nous regroupons par la suite les lignes en fonction de la distance qui les sépare afin de créer des blocs de texte.

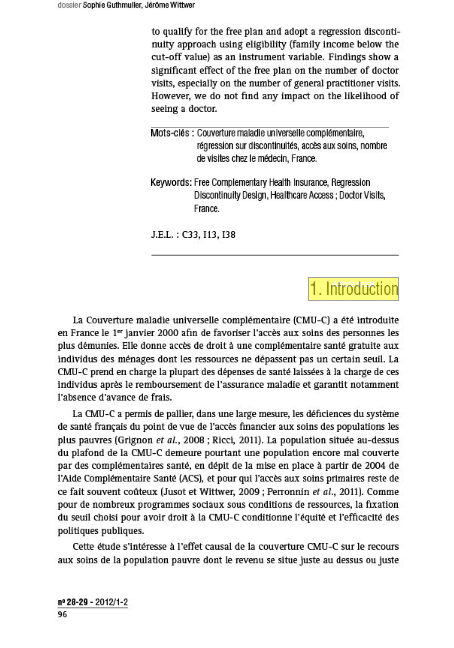


*Regroupement en blocs de texte*

## Recherche des titres et sous-titres

Nous déterminons ensuite les titres et sous-titres en fonction des styles et d’expressions régulières en appliquant l’algorithme suivant :

* Nous considérons que le style principal du document, c’est-à-dire le style du texte, est celui qui compte le plus de caractères correspondants.
* Pour chaque style ayant une taille de police supérieure à la police du style principal, on évalue sa probabilité pour que celui-ci soit un style rattaché à un titre.  
  Pour chaque bloc de texte qui correspond au style :
  + Si le bloc contient plus de trois lignes, la probabilité de style rattaché à un titre diminue.
  + Si le bloc contient plus de deux styles, la probabilité de style rattaché à un titre diminue.
  + Si le rapport nombre de caractères/nombre de ligne est trop faible, on ignore le style.
  + Si le texte du bloc correspond avec les expressions régulières définissant un titre, la probabilité de style rattaché à un titre augmente :
* Au final, si la probabilité que le style soit rattaché à un titre est suffisante, on marque le style comme candidat.
* Si aucun style n’est candidat, on relance la procédure sans tester la correspondance avec les expressions régulières.
* Chaque bloc correspondant à un style candidat est marqué comme bloc de « rupture » (titre).
* Pour chaque style candidat, on essaye de déterminer la « profondeur » du titre en appliquant des expressions régulières sur le texte de chaque bloc correspondant au style.



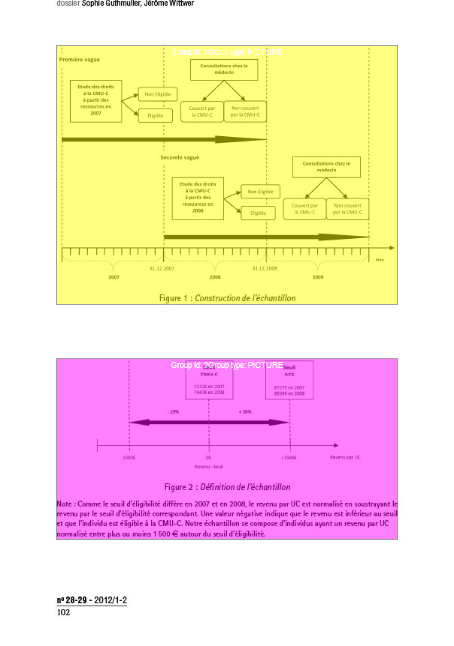
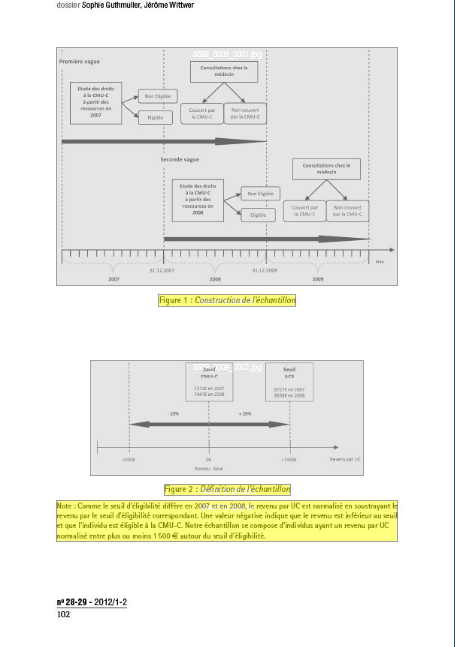
*Recherche des titres*

## Recherche des légendes

Nous rattachons les légendes aux infographies et photos du document en appliquant l’algorithme suivant :

* Pour chaque image du document, nous recherchons les blocs les plus proches de l’image.
* Pour chaque bloc trouvé, nous extrayons extrait ses styles et nous les ordonnons par taille de police décroissante.
  + Si on trouve un style qui a une taille inférieure ou égale à la taille du style par défaut + 2 points, ce style est marque comme « légende ».

Des blocs « images » sont ensuite créés regroupant l’image et sa légende.



*Recherche des légendes*

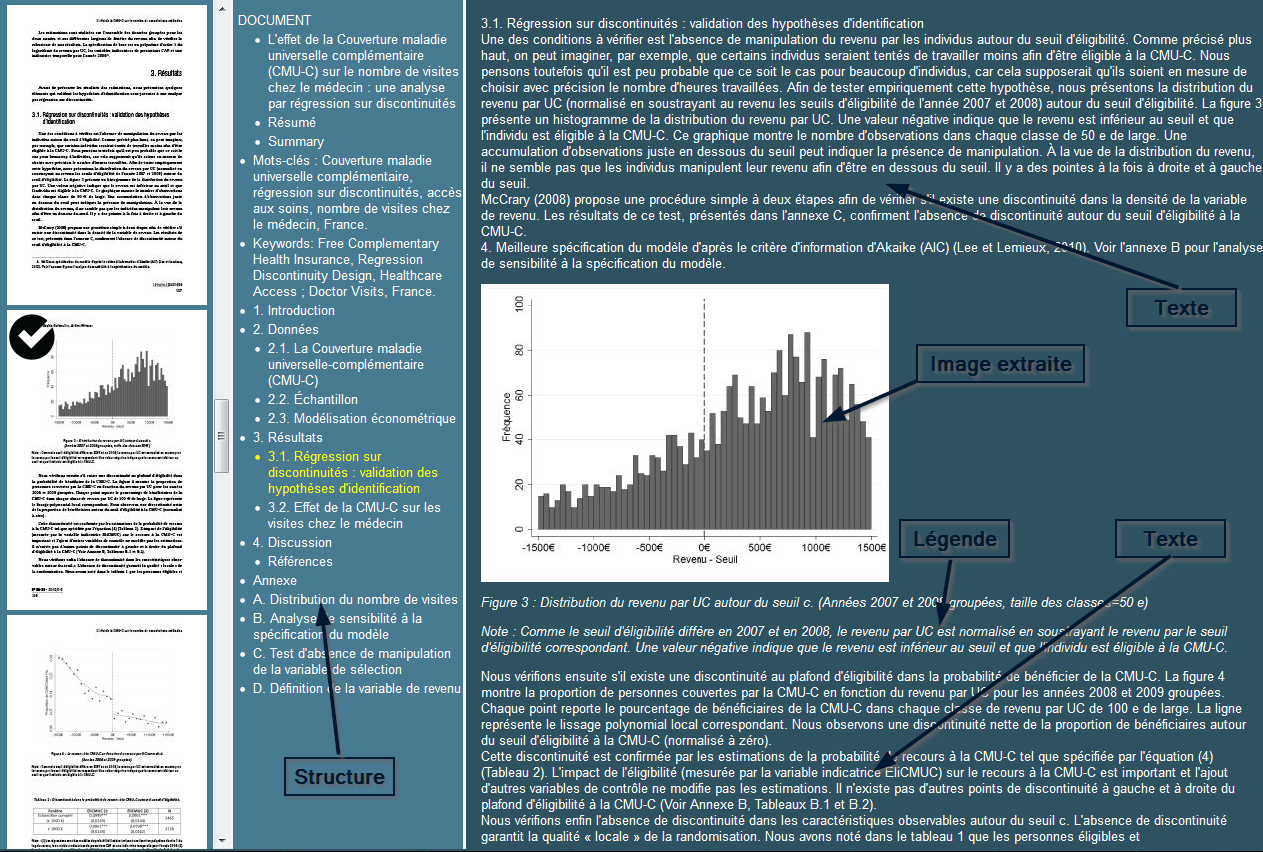
*Bloc image/légende*

## Regroupement par section

Cette étape consiste à recréer la hiérarchie du document et à placer les différents blocs de textes et d’images correctement ordonnés en vue de l’export XML.

# INTERFACE DE VISUALISATION

Nous avons développé une interface de visualisation permettant d’afficher le résultat de l’extraction. Cette interface a été développée en PHP et utilise le XML issue du regroupement par section.



# EVOLUTIONS

Il reste à retravailler sur le corpus issu de l’OCR pour améliorer la qualité de l’extraction de de texte en général, ou au moins émettre une alerte sur la qualité éventuellement suspecte du document.

Les algorithmes d’extraction doivent maintenant se spécialiser pour reconnaître de manière syntaxique des références bibliographiques et donc mettre en valeur cette zone qui sera validée ou non comme contenant des références bibliographiques par BILBO.

Ces zones peuvent être des bas de pages comme des « chapitres » à part entière dans un document.

Détecter les tableaux et les transformer en infographie afin de les traiter comme tel.