L'écriture académique au format texte

Arthur Perret (Université Bordeaux Montaigne)

21/09/2021

L'auteur

Doctorant à l'université Bordeaux Montaigne, laboratoire MICA, axe E3D. Je contribue au programme de recherche ANR HyperOtlet. Je suis actuellement ATER à l'IUT Bordeaux Montaigne. Je participe aussi à l'organisation des journées d'étude Reticulum et du groupe de discussion Inachevé d'imprimer. J'ai fait mes études de master à l'Enssib, où certains enseignants (Éric Guichard, Antoine Fauchié) m'ont fait découvrir des outils d'écriture académique au format texte (LaTeX, Markdown, Pandoc). J'alimente depuis quelques années un site avec mes recherches. Je rédige actuellement ma thèse sur Paul Otlet.

Un traitement de texte est un outil polyvalent ; par conséquent, il tend souvent vers l'usine à gaz et s'avère gourmand en ressources (énergie, mémoire). Ancré dans le paradigme de l'imprimé, il est fondamentalement inadapté à la communication via le Web. Son modèle économique (en tout pas pour les logiciels propriétaires) est de moins en moins avantageux pour le consommateur (abonnement). Ses formats sont soit complètement fermés (`.doc`) soit difficiles à utiliser via d'autres outils (`.docx`). Le format texte constitue une alternative au traitement de texte, sans pour autant le remplacer totalement.

1 Qu'est-ce que le format texte ?

En informatique, au niveau le plus fondamental, tout est exprimé dans un alphabet numérique binaire : 0 et 1. Un format, c'est une convention qui établit la correspondance entre une certaine succession de 0 et de 1, et quelque chose d'autre : par exemple une couleur, ou une lettre de l'alphabet, ou la position d'un pixel sur un

écran. L'expression « format texte » désigne une catégorie de formats pour lesquels le contenu en binaire des fichiers encode des caractères textuels uniquement.

Le format `.doc` n'est pas lisible au format texte, on ne peut l'utiliser qu'avec un traitement de texte. Le format `.docx` lui est comparable au `.zip`, c'est en fait un répertoire compressé qui contient des fichiers au format texte, balisés en XML ; on peut les ouvrir dans un éditeur de texte mais concrètement ils sont destinés à être interprétés par la machine, pas par l'humain.

Le format texte a des avantages inhérents : légèreté, pérennité, interopérabilité, choix. Il fonde une offre logicielle variée, qui comprend aussi bien des plateformes intégrées alternatives à Google Docs (comme Authorea) que des systèmes modulaires bricolés avec des logiciels libres. Surtout, il existe des styles d'interface différents pour le format texte : on a le choix de travailler en WYSIWYG ou non.

2 Implications de l'utilisation du format texte

L'utilisation du format texte n'est pas plus ou moins simple que celle du traitement de texte, elle occasionne plutôt ce que j'appellerais un déplacement des efforts cognitifs. L'interface textuelle lève l'ambiguïté entre fond et forme. Conséquence : on a plus de temps de cerveau disponible pour élaborer le fond, mais on va devoir monter en compétence d'écriture et d'automatisation, notamment pour gérer la forme (via des styles, des modèles). Ce qui fait que deux principales difficultés entourent souvent le format texte. Premièrement, son utilisation nécessite un changement de culture technique, un apprentissage. Deuxièmement, sa pratique rentre en conflit avec les habitudes prises par notre entourage (collègues, industries diverses comme par exemple l'édition).

Sur ce deuxième point notamment, les solutions logicielles qui permettraient de faciliter la collaboration autour du format texte ne sont pas entièrement satisfaisantes. Il existe des solutions d'écriture collaborative synchrone reposant sur des formats ouverts, par exemple HackMD [1] ; il existe même des plateformes dédiées à l'écriture académique, comme Authorea. Mais le fait est que ces outils manquent parfois de fonctionnalités (ex : HackMD ne facilite pas spécifiquement des besoins comme les citations) ou bien complexifient considérablement des processus qui pourraient être beaucoup plus simples (ex : les citations dans Authorea).

On peut distinguer des formats de balisage « lourds », comme XML, HTML et LaTeX, par rapport à des formats de balisage dits « légers », comme Markdown, AsciiDoc, ou encore Org (le format du mode Org pour Emacs).

1. Voir également CodiMD, version ouverte proposée par l'équipe de HackMD, et HedgeDoc, une copie du projet gérée par une communauté bénévole.

Mais la meilleure façon de les distinguer est d'évoquer les traditions différentes dans lesquelles ils s'inscrivent : publication scientifique pour LaTeX, rédaction Web pour HTML et Markdown, rédaction technique pour XML et Asciidoc, organisation personnelle pour Org-mode, etc. Ce qui ne veut pas dire que les usages sont prescrits. Mais la philosophie d'un format influence en partie sa pratique.

3 Pandoc

Pandoc est un convertisseur entre formats textuels. Le créateur de Pandoc est John MacFarlane, professeur de philosophie à Berkeley. Pandoc définit une variante de Markdown, le Pandoc Markdown, qui peut servir de source à tous les autres formats d'export de Pandoc et inclut des fonctionnalités liées à l'écriture académique. Ceci en fait un logiciel extrêmement utile lorsqu'on veut pratiquer le format texte en contexte académique et coexister pacifiquement avec les traitements de texte.

Pandoc peut être utilisé de manière autonome. Mais il peut aussi être utilisé via des scripts qui facilitent son automatisation, comme Pandoc Scholar, Manubot, Pandocomatic. Pandoc est au cœur du fonctionnement de certains éditeurs de texte, comme PanWriter, Fidus Writer, Zettlr, Stylo. Enfin, l'intégration de Pandoc à certains éditeurs de texte populaires (Vim, Emacs, Sublime, Atom, VS Code…) est disponible via des extensions, et son intégration à certains grands écosystèmes de programmation existe également : Pandoc est utilisé par exemple par le format R Markdown (R) et dans les carnets Jupyter (Python). Ces ressources et bien d'autres sont listées sur le wiki Pandoc Extras.

Il existe des cours permettant de découvrir simultanément Markdown et Pandoc : Élaboration et conversion de documents avec Markdown et Pandoc par Jean-Daniel Bonjour, Rédaction durable avec Pandoc et Markdown sur le site du *Programming Historian*. Une fois cette première étape passée, la principale ressource pour continuer à apprendre à utiliser Pandoc est le manuel. Ce long document constitue une mine d'or inépuisable. De nombreuses autres ressources traitent des différents usages spécialisés de Pandoc. J'ai publié moi-même quelques retours d'expérience sur mon site. Une simple recherche « pandoc + type d'utilisation » permet généralement de trouver rapidement des billets de blog contenant des exemples et des solutions.

Le format texte est parfois l'objet de vives critiques. Adam Hyde, fondateur de Coko, suggérait récemment que les auteurs n'ont pas nécessairement envie d'apprendre à écrire du code source, si on s'accorde à définir ainsi Markdown par exemple. Hyde est plutôt partisan d'outils fondés sur le format texte mais avec des interfaces d'écriture WYSIWYG. Par ailleurs, les partisans du format texte ont parfois une posture anti-traitement de texte caricaturale qui me paraît contre-productive, car elle favorise la réduction de la question à une querelle de chapelles, avec une

radicalisation des positions qui ne peut que nuire à la diffusion de nouvelles pratiques. Ainsi, certaines critiques virulentes de Word nous reviennent désormais en boomerang sous la forme d'une opposition de principe au format texte, qui crée des blocages regrettables. Il me paraît urgent de désamorcer ces conflits stériles, et je crois que Pandoc peut être un véritable instrument de médiation, pour désenclaver nos pratiques d'écriture.

4 Démonstration de Pandoc

Cliquez ici pour récupérer les fichiers de la démo.

Lancez un terminal et déplacez-vous à l'endroit où vous avez mis ces fichiers. Par exemple, si les fichiers sont dans un répertoire `démo` sur votre Bureau dans macOS :

```
cd /Users/ici-votre-login/Desktop/démo
```

4.1 Conversion simple

L'ordre des options n'a pas d'importance. Les commandes ci-dessous sont rédigées de manière à mimer le processus de conversion : on dit à Pandoc de prendre un fichier et de le convertir en un autre fichier.

Conversion dans un format pour traitement de texte :

```
pandoc doc-simple.md -o doc-simple.docx
```

Conversion en HTML :

```
pandoc doc-simple.md -o doc-simple.html
```

Conversion en PDF (nécessite d'installer une distribution LaTeX) :

```
pandoc doc-simple.md -o doc-simple.pdf
```

Très vite, on va ajouter des options aux commandes Pandoc, qui ne tiennent plus toujours sur une ligne. Dans le reste du document, les commandes sont donc présentées autrement : le fichier à convertir occupe la dernière ligne, et il y a un retour à la ligne entre chaque option. Or un retour à la ligne (touche entrée) dans un terminal déclenche l'exécution de la commande ; s'il intervenait au milieu de la commande `pandoc`, celle-ci n'aurait pas l'effet escompté. Solution : on ajoute une barre oblique inverse avant un retour à la ligne pour signaler que ce dernier ne doit pas être pris en compte par le terminal. On peut alors copier et coller en bloc une commande de plusieurs lignes puis appuyer sur Entrée pour l'exécuter.

```
pandoc \
  -o doc-simple.docx \
  doc-simple.md
```

4.2 Options

Exemple (ajout d'une feuille de styles CSS à un export HTML) :

```
pandoc \
  --css=styles.css \
  --standalone \
  -o doc-simple.html \
  doc-simple.md
```

4.3 Métadonnées

Le fichier `doc-avec-entete.md` commence par un en-tête au format YAML. On peut y inclure des métadonnées qui seront utilisées quelque soit le format d'export :

```
title: L'écriture académique au format texte
author: Arthur Perret (Université Bordeaux Montaigne)
date: 21/09/2021
lang: fr-FR
```

On peut aussi inclure des options qui seront utilisées en fonction du format :

```
documentclass: scrartcl
colorlinks: true
```

Les métadonnées peuvent être stockées dans un fichier séparé, auquel cas il faut l'indiquer dans la commande `pandoc` :

```
pandoc \
  --metadata-file=metadonnees.yml \
  --css=styles.css \
  --standalone \
  -o doc-sans-entete.html \
  doc-avec-entete-separe.md
```

4.4 Modèles par défaut et personnalisation

Pandoc utilise un modèle par défaut pour chaque format. Cliquez ici pour les consulter. Ils sont construits pour vous permettre de modifier l'export via des options, qu'elles soient dans l'en-tête de votre fichier Markdown ou dans la commande Pandoc utilisée pour le convertir. Lorsque la personnalisation voulue est plus poussée, il est possible de récupérer un modèle par défaut, le modifier et l'utiliser pour la conversion.

Mettez vos templates dans le dossier utilisateur de Pandoc afin qu'ils soient utilisés automatiquement. Pour localiser ce dossier, voici un extrait traduit en français du manuel de Pandoc :

> « Sur les systèmes type Unix et macOS, il s'agira du sous-répertoire `pandoc` du répertoire de données de l'utilisateur, par défaut, `$HOME/.local/share`. Si ce répertoire n'existe pas et que `$HOME/.pandoc` existe, il sera utilisé (pour la rétro-compatibilité). Sous Windows, le répertoire de données utilisateur par défaut est `C:\Users\USERNAME\AppData\Roaming\pandoc`. Vous pouvez trouver le répertoire de données utilisateur par défaut sur votre système en saisissant `pandoc --version` dans un terminal. Les fichiers de données placés dans ce répertoire (par exemple, `reference.odt`, `reference.docx`, `epub.css`, `templates`) remplaceront les valeurs par défaut normales de pandoc. »

Prenons un exemple. LaTeX est un système modulaire, c'est-à-dire que beaucoup de fonctionnalités sont disponibles sous formes de modules appelés « paquets » (en anglais *packages*) et peuvent être utilisées à la carte en fonction de vos besoins. En ajoutant une ligne contenant `\usepackage{ici le nom du paquet}` dans l'en-tête d'un document LaTeX ou dans un modèle de document LaTeX, cela ajoute la fonctionnalité correspondante au moment de la compilation en PDF.

Le template LaTeX par défaut de Pandoc peut être contrôlé via des options. Ainsi, si vous ajoutez une ligne contenant `fontfamily: times` dans l'en-tête de votre fichier Markdown, l'export PDF utilisera le paquet `times`, ce qui modifie la police du document. Ceci repose sur le template LaTeX, sans que vous ayiez eu à modifier ce dernier.

Mais parfois, il faut modifier le template. Dans les classes de document LaTeX KOMA-Script, plus adaptées aux normes typographiques européennes que les classes par défaut, plusieurs polices sont utilisées dans un même document (souvent une police sans empattements pour les titres, et avec empattements pour le corps du texte). On peut vouloir utiliser une famille de polices comme IBM Plex. Or l'option `fontfamily` ne fonctionnera pas, car on ne peut déclarer qu'un paquet avec, et on ne peut pas utiliser plusieurs fois une même option en YAML. La solution est d'ajouter les lignes suivantes au modèle LaTeX par défaut de Pandoc :

```
\usepackage{plex-serif}
\usepackage{plex-sans}
\usepackage{plex-mono}
```

Dans le fichier `template.latex` fourni, cette modification a été effectuée (lignes 39-41).

```
pandoc \
  --template=template.latex \
  --number-sections \
  -o doc-sans-entete.pdf \
  doc-avec-entete.md
```

4.5 Citations

Pandoc accepte plusieurs formats de données bibliographiques. Les principaux utilisés dans l'écosystème Pandoc sont le BibTeX et le CSL JSON.

```
pandoc \
  --citeproc \
  --bibliography=references.bib \
  --csl=styles.csl \
  -o article.pdf \
  article.md
```

Zotero propose un répertoire en ligne de fichiers de styles bibliographiques.