Université Paris Nanterre
200 av. de la République
92001 Nanterre Cedex
www.parisnanterre.fr

Université Paris Nanterre
École doctorale 138 Lettres, langues et spectacles
DICEN-IdF
Membre de l'Université Paris Lumières
Université de Montréal
Département de littératures et langues du monde
CRCEN

Université de Montréal

Roch DELANNAY

Environnements d'écriture numérique et production du sens

Thèse présentée et soutenue publiquement le 05 novembre 2025 en vue de l'obtention du doctorat en cotutelle de Sciences de l'information et de la communication de l'Université Paris Nanterre, et de Littérature, option humanités numériques de l'Université de Montréal sous la direction de Mme Marta SEVERO (Professeure des universités, Université Paris Nanterre), de M. Marcello VITALI-ROSATI (Professeur titulaire, Université de Montréal) et de M. Emmanuel CHÂTEAU-DUTIER (Professeur agrégé, Université de Montréal).

Composition du jury

Rapporteur	M. Bruno BACHIMONT	Professeur des universités, Université de technologie de Compiègne
Rapporteur	M. Samuel SZONIECKY	Maître de conférences HDR, Université Paris 8 Vincennes - Saint-Denis
Présidente du jury	Mme Susan KOVACS	Professeure des universités, ENSSIB
Membre du jury	M. Emmanuel CHÂTEAU-DUTIER	Professeur agrégé, Université de Montréal
Membre du jury	Mme Camille PRIME-CLAVERIE	Maîtresse de conférences, Université Paris Nanterre
Membre du jury	Mme Marta SEVERO	Professeure des universités, Université Paris Nanterre
Membre du jury	M. Marcello VITALI-ROSATI	Professeur titulaire, Université de Montréal

Environnements d'écriture numérique et production du sens

Roch Delannay

Remerciements

Mes remerciements vont d'abord à Marta Severo et Marcello Vitali-Rosati, pour avoir rendu cette thèse possible. Le projet de thèse a été évoqué pour la première fois avec Marta, entre deux réunions éditoriales tandis que je travaillais comme chargé d'édition pour l'EUR ArTeC à Paris 8. Merci d'avoir accepté de suivre cette thèse et aidé, par la suite, à ficeler le montage en cotutelle. Je remercie Marcello de m'avoir accueilli si chaleureusement et généreusement à Montréal. Il y avait cette revanche à prendre sur un stage de recherche que je devais effectuer à Montréal et qui a été annulé pour cause de pandémie. Je te suis reconnaissant pour m'avoir ouvert les portes de la Chaire une deuxième fois. C'est bien grâce à votre engagement à tous les deux que ce projet a pu être mis sur pied.

Je tiens à remercier à nouveau Marta et Marcello, ainsi qu'Emmanuel Château-Dutier en leur qualité de directrice et directeurs de recherche, pour leur confiance, leur bienveillance et leur soutien indéfectible tout au long de ce parcours doctoral.

Merci aux membres du jury, Bruno Bachimont, Susan Kovacs, Camille Prime-Claverie et Samuel Szoniecky d'avoir évalué mon travail.

Tous les projets sur lesquels j'ai pu travailler et qui sont mentionnés dans cette thèse sont animés par des personnes passionnantes qui ont largement contribué à les enrichir. Merci à Hélène Beauchef, Maitané Lenoir, Guillaume Grossetie, Thomas Parisot, David Larlet, Servanne Monjour, Nicolas Sauret, Yves Marcoux, Edith Cannet, Dominique Roux, Michael Sinatra, Louis Gabrysiak et Timothée Guicherd.

Une première mention spéciale est adressée à Margot Mellet, Antoine Fauchié, Isabelle Burlet, Enzo Poggio, Alix Chagué, Giulia Ferretti et Mathilde Verstraete pour tous nos moments partagés à Montréal. On s'est bien amusé ! Que ce soit entre les murs de l'université, durant les colloques et les hackathons ou en vadrouille dans la ville et sur le fleuve.

Une deuxième mention spéciale est adressée à Robert Alessi, que je remercie d'avoir partagé son enthousiasme contagieux sur les technologies libres, ainsi que pour m'avoir initié à Libreboot et accompagné dans mon apprentissage d'OpenBSD et de LaTeX.

Merci à Aurélie Sansen pour cette précieuse relecture de toute la bibliographie dans le dernier kilomètre avant le dépôt de la thèse.

Nous avons bricolé et fabriqué de superbes environnements d'écriture numérique avec les membres de la Chaire de recherche du Canada. L'énergie déployée par chacun dans cette équipe crée un formidable écosystème de recherche. Merci à Margot Mellet, Antoine Fauchié, Louis-Olivier Brassard, Giulia Ferretti, Mathilde Verstraete, Alix Chagué, Eugénie Matthey-Jonais, François Maltais-Tremblay, Adrien Savard-Arseneault, Clara Grometto, Victor Chaix, Camille Germain, Lena Krause, Emmanuelle Lescouet, Yann Audin, Arilys Jia, Halima Malek, Luiz Capelo et Enrico Agostini Marchese.

Dans un autre environnement tout aussi propice à la recherche, un grand merci à tous les membres de l'équipe de Nanterre, avec une mention spéciale pour Romain Vindevoghel, Fabrizio Defilippi. C'est avec vous que j'ai passé la plus grande partie de mon temps dans les locaux de Nanterre. Je remercie également Lucas Fritz, Mathilde Vassor, Irene De Togni, Andreas Verner, Jorge Sosa, Arthur Lacour, Giuseppe Amapani, Alizé Sibella, Julien Gaillard, Victoria Laurent, Antonin Segault, Dario Compagno, Céline Morin, Camille Prime-Claverie, Eleni Mouratidou, Jeanne Ferrari-Giovanangeli, Zhao Alexandre Huang, Elie Petit et Cécile Payeur.

Un grand merci, enfin, à mes proches et à ma famille pour leur soutien et leurs encouragements durant ces dernières années : Clarisse, qui a vécu la rédaction mot après mot, Alix, Raphaël, Bruno, Florence, David et Fabienne.

Résumé

Depuis plus de quatre décennies, nos pratiques d'écriture numérique demeurent largement influencées par une culture héritée de l'imprimé et des dispositifs qui l'ont précédée. Cette continuité interroge les significations de l'écriture numérique et ses conditions de production, notamment dans un environnement conçu pour le calcul.

Cette thèse s'intéresse aux environnements d'écriture numérique et à leur rôle dans la production de sens. Loin d'être de simples outils techniques ou des interfaces graphiques, ces environnements reposent sur des modèles textuels – comme les formats DOCX et ODT – qui incarnent des théories du texte et définissent formellement le sens des éléments textuels mobilisés lorsque nous écrivons.

Cette recherche s'appuie sur les méthodes des *Science and Technology Studies* afin de constituer une biographie des environnements étudiés et d'analyser les pratiques d'écriture à partir de documents produits dans deux contextes éditoriaux en sciences humaines et sociales. L'étude porte plus particulièrement sur Microsoft Word, LibreOffice et Stylo, ainsi que sur Lodel/OpenEdition et le Pressoir qui servent à transformer ces documents pour les publier.

À partir de l'observation des documents produits par ces environnements, que l'on recoupe avec leur documentation technique, cette thèse montre qu'il y a des variations du sens produit par le dispositif d'écriture selon les modèles textuels mobilisés. Cette thèse montre également que la conception graphique des éléments textuels n'est pas uniquement située sur nos écrans, au niveau de l'interface ou du traitement de l'affichage. Il s'agit en effet d'une conception plus profonde de nos modes d'écriture que l'on retrouve dans la définition même des éléments décrits dans les spécifications des formats, que ce soit des formats de texte brut ou des formats de traitement de texte.

Mot-clefs :
Modèle textuel, Écriture numérique, Éditorialisation, Néomatérialisme, MSWord, Stylo, LibreOffice, Lodel, le Pressoir.

Abstract

For over 40 years, our digital writing practices have been mostly influenced by a culture that came from print and the devices that came before it. This makes us ask questions about what digital writing means and how it is produced, especially in a computing environment.

This thesis looks at how digital writing tools are used to create meaning. These tools are more than just technical tools or graphical interfaces. They are based on textual models, such as DOCX and ODT formats. These models are theories of text and formally define the meaning of the textual elements we use when we write.

This research uses methods from Science and Technology Studies to create a biography of the environments studied and to analyse writing practices based on documents produced in two different editorial contexts in the humanities and social sciences. The study looks in more detail at the Microsoft Word, LibreOffice and Stylo environments, as well as on environments for changing these documents for publication, Lodel/OpenEdition and le Pressoir.

After looking closely at the documents from these environments and comparing them with their technical documents, this thesis shows that the meaning of the writing system changes depending on the text model used. This thesis also shows that the way text looks is not only on our screens, at the level of the interface or display processing. It is actually a more complex design of our writing modes that can be found in the definition of the elements described in the format specifications, whether plain text formats or word processing formats.

Keywords :
Textual modeling, Digital textualities, Editorialization, Neomaterialism, MSWord, Stylo, LibreOffice, Lodel, le Pressoir.

Sommaire

	Remerciements	iii
	Résumé	v
	Abstract	vii
	Sommaire	ix
	Chronologie des théories et technologies mobilisées	xi
	Introduction	1
I	Écriture et production du sens	19
II	Écriture numérique et production du sens	59
III	Écrire avec des logiciels *What You See Is What You Get*	99
IV	Écrire avec Stylo	141
V	Les transformations éditoriales avec Lodel et le Pressoir	185
	Conclusion	231
	Bibliographie	243
	Annexe 1 : Entretien avec Stéphane Pouyllau	259
	Glossaire	263
	Index des personnalités	267
	Liste des acronymes	271
	Table des extraits de code	275
	Table des figures	277
	Table des matières	279
	Colophon	281

Chronologie des théories et technologies mobilisées

Avant propos

Cette partie préliminaire à la thèse présente une chronologie des principaux éléments en lien avec les environnements d'écriture numérique abordés tout au long du manuscrit. Cette chronologie est affichée sous la forme d'un tableau.

Le tableau 1 regroupe les éléments en quatre catégories : théories, *hardware*, *software*, ainsi que les normes et formats. Comme nous l'illustrons par la suite, l'informatique connaît une évolution rapide entre les années 1970 et 2025. Tous les aspects mentionnés évoluent, soit à la suite de nouvelles mises à jour successives, soit parce qu'ils sont supplantés par de nouvelles normes ou technologies, ce qui influe sur les capacités des environnements d'écriture numérique. Il nous a semblé pertinent de confronter ces évolutions aux théories sur lesquelles nous nous appuyons pour étayer notre raisonnement, afin d'en proposer une contextualisation à partir d'une vue d'ensemble.

La période couverte par cette chronologie s'étend de 1878 à 2025. Toutefois, notre attention se porte principalement sur les soixante dernières années, décennies durant lesquelles se sont déroulées la plupart des activités directement liées à l'activité d'écriture sur un ordinateur. Les quatre-vingts premières années du tableau visent surtout à signaler quelques événements historiques clés pour comprendre les environnements d'écriture, sans toutefois représenter la majorité des éléments durant cette période.

Ce tableau remplit ainsi un double objectif : d'une part, fournir une vue d'ensemble des éléments mobilisés dans ce manuscrit, servant d'outil d'accompagnement à la lecture auquel se référer, et d'autre part, fournir des jalons temporels permettant de mieux situer les théories et les marqueurs historiques évoqués tout au long de la thèse.

Tableau chronologique

Table 1 : Chronologie des théories et technologies mobilisées

Année	Théorie	*Hardware*	*Software*	Normes et formats
1878		Remington II, première machine à écrire comportant la touche SHIFT dans son clavier		
1934	Documentation (Otlet)			
1935				
1936	Machine de Turing			
1937				
1938				
1939				
1940				
1941				
1942				
1943				
1944		Harvard Mark I (ASCC)		
1945	Architecture de von Neumann	Ordinateur EDVAC		
1946				
1947				
1948				
1949				
1950				
1951	Documentation (Briet)			

Année	Théorie	*Hardware*	*Software*	Normes et formats
1952				
1953				
1954				
1955				
1956				
1957				
1958				
1959		Commercialisation de la machine à écrire Valentine (Olivetti)		
1960				
1961				
1962				
1963				Approbation de la norme d'encodage ASCII par l'ASA
1964	Théorie des médias (McLuhan)			
1965		Commercialisation du P101, premier ordinateur personnel (Olivetti)		
1966	Intermédialité (Higgins)			
1967				
1968				
1969				

Année	Théorie	*Hardware*	*Software*	Normes et formats
1970		Obtention du brevet pour la souris (Engelbart)		
1971	Développement du concept de *Document Type Definition* (DTD) (Goldfarb)	Commercialisation du processeur Intel 4004		
1972				
1973		Xerox Alto, premier ordinateur personnel avec une interface graphique		Première apparition publique du langage de balisage GML
1974		Commercialisation du processeur Intel 8080	Bravo, précurseur des logiciels de traitement de texte WYSIWYG développé pour l'Alto	Publication des protocoles Internet TCP/IP
1975		Altair 8800 (basé sur le processeur Intel 8080)		

Année	Théorie	*Hardware*	*Software*	Normes et formats
1976		Commercialisation du processeur Intel 80C85 / Commercialisation du processeur Zilog Z80 (les logiciels sont compatibles avec le Intel 8080)	Electric Pencil, publication du premier du traitement de texte pour l'Altair 8800	
1977	La raison graphique (Goody)	TRS-80 Model I (processeur Zilog Z80)		
1978			Commercialisation du premier WYSIWYG WordStar (langage assembleur Intel 8080) / Première version de TeX publiée par Donald Knuth / Adaptation du code d'Electric Pencil au TRS-80 Model I	
1979			EasyWriter, traitement de texte pour Apple / Commercialisation du traitement de texte WordPerfect	
1980				

Année	Théorie	*Hardware*	*Software*	Normes et formats
1981				Publication de la version étendue de la norme d'encodage ASCII
1982			Nota Bene, un traitement de texte pour l'écriture scientifique	
1983		Commercialisation de l'ordinateur personnel M10 incorporant le processeur Intel 80C85 (Olivetti)	Première version de MSWord / Premières interfaces graphiques chez Apple	
1984	*Literate Programming* (Knuth)		Commercialisation de WordStar 2000 (successeur de WordStar) / Publication de LaTeX (par Leslie Lamport)	
1985			Publication de la première version de l'éditeur de texte Emacs / Première version de StarOffice (ancêtre de LibreOffice)	Création du format BibTeX
1986				Le langage de balisage SGML devient une norme internationale (ISO)

Année	Théorie	*Hardware*	*Software*	Normes et formats
1987		Commercialisation de la machine à écrire électronique ETP 55 avec son nouveau design (Olivetti)		Début de la TEI (recommandations en SGML)
1988				
1989				Création du World Wide Web (Tim Berners-Lee)
1990	Définition du modèle OHCO (DeRose et al.)		Commercialisation de Windows 3.0 avec son interface graphique	
1991	Le logiciel n'existe pas (Kittler)		Première version de l'éditeur de texte Vim	Première publication de la norme d'encodage Unicode
1992				
1993				Première publication du format PDF
1994				
1995				HTML 2.0 / Création du JavaScript
1996				Finalisation de la version 1.0 du protocole HTTP
1997	Document (Buckland)			
1998	Énonciation éditoriale (Souchier)			Première publication de XML

Année	Théorie	*Hardware*	*Software*	Normes et formats
1999	Architexte (Souchier)		Ouverture du code d'OpenOffice (rachat de StarOffice par SUN Microsystem) / Début du projet Métopes (chaîne de transformation XML)	
2000	La raison computationnelle (Bachimont)		Création du CMS Lodel	
2001			Publication de la première version du CMS SPIP	Création du format YAML
2002				Introduction de XML dans la TEI
2003			FCKEditor, premier éditeur de texte pour le Web	
2004	Éditorialisation (Pédauque) / Intermédialité montréalaise			Création du format Markdown
2005	Posthumanisme (Hayles)		Première version du traitement de texte Pages (Apple)	Approbation de la norme ODF par l'OASIS / Publication de la première version du système de contrôle des versions Git

Année	Théorie	*Hardware*	*Software*	Normes et formats
2006	Document numérique (Pédauque)		Ajout du sous-programme Management Engine dans le BIOS de la firme Intel / Première version de Pandoc (MacFarlane)	Approbation de la norme internationale ODF par l'ISO / Approbation de la norme OOXML par l'ECMA
2007	Réalisme agentiel (Barad)		Publication de la première version de l'éditeur de texte pour le Web CodeMirror	Publication du SGBD MongoDB / Publication de la première version du format ePUB
2008	Éditorialisation (Marcello Vitali-Rosati)			Approbation de la norme internationale OOXML par l'ISO / Approbation de la version 1.7 du format PDF en tant que norme internationale par l'ISO
2009				
2010	Dispositif et théorie du support (Bachimont)			
2011			Poursuite du projet OpenOffice sous le nom LibreOffice par The Document Foundation / Application d'annotation Hypothes.is	

Année	Théorie	*Hardware*	*Software*	Normes et formats
2012	Écriture distribuée (Paveau)	Lenovo Thinkpad X230 (sur lequel j'ai écrit la thèse)		
2013	Architecture des informations (Broudoux *et al.*)	Lenovo Thinkpad T440P (sur lequel j'ai écrit la thèse)	Publication de la première version de la librairie JavaScript React	
2014				
2015			Publication de l'éditeur Monaco (avec Visual Studio Code)	Première publication de GraphQL
2016				
2017			Début du projet Stylo	Intégration du schéma XML COMMONS Publishing à Métopes et à Lodel 2
2018			Premières ébauches du générateur de site statique le Pressoir	
2019				
2020		Stylo est déployé sur les serveurs de la TGIR Humanum	Première version du paquet LaTeX ekdosis (Alessi)	
2021			Version 1.0 du logiciel de collation de variants ChrysoCollate (Moureau)	
2022				
2023				

Année	Théorie	*Hardware*	*Software*	Normes et formats
2024			Publication du Pressoir sous la forme de librairie Python	
2025			Mise en ligne de l'outil de traitement de texte collaboratif Docs par le gouvernement français	

Introduction

Situer la recherche

Cette thèse porte sur l'étude des environnements d'écriture numérique. Il s'agit d'une notion fondamentale de nos modes de communication contemporains, puisqu'ils constituent les cadres à travers lesquels nous pouvons écrire en contexte numérique. En effet, nous considérons que ces environnements numériques, ainsi que les médiations qui leur sont associées, définissent formellement ce qu'il est possible d'écrire ou non dans ces espaces et, par conséquent, le sens produit par l'écriture numérique.

Afin de traiter cet objet, *Environnements d'écriture numérique et production du sens* est une thèse réalisée à la croisée de deux disciplines, les sciences de l'information et de la communication et la littérature, ainsi que d'un champ de recherche, les humanités numériques. Cette triple perspective se traduit par une inscription en cotutelle entre, d'une part, l'Université Paris Nanterre en sciences de l'information et de la communication et, d'autre part, l'Université de Montréal en littérature avec une option en humanités numériques. Ce rappel n'est pas anodin, car il permet de mieux situer l'épistémologie qui caractérise notre recherche.

La théorie de l'éditorialisation a constitué le socle théorique à partir duquel nous avons établi des liens entre ces différentes perspectives. Nous nous positionnons à l'articulation de deux courants associés à cette notion, tous deux issus des travaux menés par le collectif RTP-DOC dès 2004. Le premier provient du domaine de la documentation, avec des contributions de Bruno Bachimont, Évelyne Broudoux, Ghislaine Chartron, Madjid Ihadjadene, Carole Lipsyc, Louise Merzeau, Cécile Payeur et Manuel Zacklad. Le second, sur lequel nous nous appuyons également, se situe à la croisée de la philosophie et de la littérature : ce courant a été initié à partir de 2008 par Marcello Vitali-Rosati, en collaboration notamment avec Louise Merzeau, Servanne Monjour, Nicolas Sauret, Michael Sinatra, Gérard Wormser et bien d'autres, lors d'un séminaire consacré à cette thématique.

Cette théorie de l'éditorialisation a été essentielle à notre démarche, car c'est grâce à ce point d'entrée entre la littérature et les sciences de l'information et

de la communication que nous avons convoqué et lié de multiples concepts ou approches provenant de ces deux disciplines. Du côté des sciences de l'information et la communication, nous nous sommes appuyés sur les notions de document, de dispositif, d'énonciation éditoriale, d'indexation des informations, d'organisation des connaissances, ou encore de certains aspects de l'édition numérique. Tandis que d'un autre côté, la littérature nous a apporté la théorie des médias, l'intermédialité montréalaise, le posthumanisme et surtout, une approche méthodologique développée au fil des années par la Chaire de Recherche du Canada sur les Écritures Numériques (CRCEN) qui consiste en l'application de l'analyse textuelle traditionnellement réservée aux œuvres littéraires aux scripts de code informatique et plus largement aux artefacts numériques.

La dernière perspective mobilisée dans cette recherche est celle des humanités numériques. Si les définitions de ce champ de recherche sont multiples et différentes d'une discipline à une autre, d'une école à une autre, ou même selon les continents, nous nous sommes pour notre part largement inspiré de la vision qu'en offrent Stephen Ramsay et Geoffrey Rockwell que l'on peut résumer par ce fragment de phrase devenu presque une modalité de recherche : « considérer les artefacts numériques en tant que théories »[1] (Ramsay et Rockwell 2012 : 79). Selon leur perspective, les artefacts numériques ou les prototypes d'artefacts, leur code informatique ainsi que leur documentation sont des objets théoriques en eux-mêmes, sans qu'il soit nécessaire de leur adjoindre des éléments discursifs supplémentaires pour les expliciter. La théorie est déjà là, car elle n'est ni pensée en amont des développements de ces artefacts, ni en aval. Elle se matérialise avec l'artefact numérique et lui est indissociable.

Cette vision des humanités numériques a été pratiquée tout au long de cette recherche, que ce soit à Montréal ou à Nanterre. En effet, j'ai eu l'occasion de m'impliquer dans plusieurs projets de recherche, notamment Stylo et le Pressoir pendant plus de trois années ou encore les projets Webmémoires[2] et l'édition de l'ouvrage *Contribution numérique : cultures et savoirs* avec une chaîne éditoriale expérimentale. Nous retrouvons non seulement ces différents projets en tant que cas d'étude de notre recherche, mais aussi en filigrane de notre discours. Les questions que nous nous sommes posés quant aux différents développements ou obstacles que nous avons rencontrés sont tout autant d'éléments qui ont alimenté notre recherche, nos analyses et nos conclusions. Pour le dire autrement, c'est

1. Sauf mention contraire, toutes les citations étrangères sont des traductions personnelles.

2. Webmémoires est le nom d'un projet de recherche développé en collaboration avec le DataLab de la Bibliothèque National de France (BNF) et soutenu par le Labex *Les passés dans le présent*. Ce projet visait à étudier comment les archives du Web jouent un rôle fondamental dans la construction de la mémoire collective. Nous nous sommes appuyés sur les archives du Web du dépôt légal de la BNF, en particulier sur les phénomènes liés à la pandémie de Covid-19.

à partir des artefacts numériques que cette recherche débute et c'est à travers eux que nous avons pu lier les approches théoriques que nous mobilisons dans ce manuscrit.

Frictions dans l'écriture

Environnements d'écriture numérique et production du sens examine les environnements d'écriture numérique et les médiations qui les traversent durant la production de l'écriture et du document. L'écriture y est envisagée comme un dispositif technique distribué entre des éléments techniques, culturels, sociaux et humains. Parmi ces médiations, nous nous concentrons sur les modèles textuels implémentés dans les environnements numériques. Un modèle textuel est une représentation formelle et explicite du texte, structurée et hiérarchisée, qui formalise une théorie du texte pour permettre sa manipulation technique et sémantique dans les environnements d'écriture numérique. En mobilisant la théorie des médias développée par l'École de Toronto, nous avançons l'hypothèse que ces modèles textuels et les théories implicites qu'ils véhiculent participent directement à la production du sens dans l'écriture numérique.

Le volume du contenu de cette thèse au format TEX, sans les images, représente environ 800 kilobytes, tandis que son équivalent au format DOCX vaut pour un peu plus du double, une fois que l'archive est décompressée. En ce qui concerne le volume des fichiers au format *Portable Document Format* (PDF) produits respectivement à partir de ces deux formats, nous avons constaté qu'ils sont similaires, soit légèrement inférieur à 2 megabytes dans les deux cas[3]. Ce dernier volume représente environ 0.0034% de l'espace disponible sur le disque dur de mon ordinateur. C'est un poids dérisoire si l'on considère les capacités de calcul et de stockage des ordinateurs employés pour produire ce document. Cette différence de poids entre les fichiers sources ne semble pas significative au vu de l'espace libre sur mon disque dur. D'ailleurs, nous n'en faisons pas l'économie dans nos dispositifs d'écriture. Cette question ne se pose plus, car nos ordinateurs modernes, dotés de grandes capacités de stockage et de traitement, sont à même de gérer beaucoup plus de fichiers, sans nécessiter l'ajout d'une mémoire externe ou d'un autre organe.

Pourtant, cette différence de taille que l'on constate entre les deux fichiers implique nécessairement différentes modalités d'écriture entre les deux environnements. Écrire en LATEX, nous en conviendrons, n'est donc pas la même action

3. Ce volume est celui des fichiers PDF sans les images. Ces dernières ont un volume conséquent puisqu'elles sont au format JPG avec un profil colorimétrique prévu pour l'imprimé. Si l'on comptabilise les images dans le volume des fichiers PDF, celui-ci s'élève à un peu moins de 54 megabytes.

qu'écrire en DOCX. Dès lors, un simple coup d'oeil aux fichiers, notamment les éléments structurels du texte, nous permet de trouver immédiatement les 800 kilobytes de différence entre nos deux formats. En effet, la structure du texte au format DOCX est aussi lourde que le texte lui-même. Si nous la comparons au fichier au format TEX, la structure du DOCX vaut pour à peu près le volume total du fichier.

Ramené à la taille du fichier issu du traitement de texte, le poids de la structure du texte en DOCX semble alors très lourd en comparaison de celui de la structure en LATEX. Cependant, si l'on ne s'en tient qu'aux fonctionnalités similaires des deux environnements et que nous observons les documents qu'ils produisent, en omettant de regarder les différences typographiques ou encore les pages préliminaires et liminaires produites par LATEX, nous pouvons avancer que les structures graphiques des deux fichiers PDF semblent passablement similaires. La hiérarchie des titres a l'air identique dans les deux fichiers, ainsi que certains éléments structurels tels que les emphases, les blocs de citation, les notes de bas de page et les références bibliographiques.

Au vu de ce faible écart entre les deux structures graphiques, malgré la médiocre qualité typographique du rendu PDF produit avec le fichier DOCX[4], nous pourrions nous demander pourquoi la structure du fichier DOCX est-elle aussi lourde comparée à celle du fichier LATEX. C'est d'ailleurs une question similaire qui a motivé cette recherche à ses débuts. Cette question pose cependant le problème de travers, car elle soulève un enjeu technique qui nous intéresse peu en sciences de l'information et de la communication et en littérature. On pourrait même envisager qu'il n'y a pas d'enjeu, au-delà d'une forme d'élégance, à réduire la taille des fichiers à une forme plus légère tout en produisant une structure graphique qui soit semblable. Néanmoins, si ce n'est pas une question de volume, mais une question d'organisation des connaissances et de production du sens qui est en jeu dans cette différence de 800 kilobytes, nous pourrions nous demander s'il y a des différences de signification entre les deux architectures. En suivant cette direction, nous pourrions également présupposer que l'architecture DOCX est plus riche que celle de LATEX puisqu'elle est bien plus lourde et plus verbeuse.

Voilà qu'apparaît notre problème : les différences de structure des éléments textuels selon l'environnement d'écriture employé laissent sous-entendre une différence de contenu et donc de sens.

4. La médiocrité typographique produite par le DOCX n'est pas qu'un avis personnel ou subjectif. Aucune chaîne éditoriale ne publie des fichiers produits avec du traitement de texte. Ils sont nécessairement transformés soit via des outils propriétaires comme ceux de la suite Adobe, ou leurs équivalents libres tels que Scribus, ou encore à l'aide de *Content Management System* (CMS) pour une publication sur le Web.

Ce premier constat réactualisé avec ce manuscrit est doublé d'une autre expérience qui précède le début du doctorat. Elle en constitue d'ailleurs les prémisses. À ce moment-là, je travaillais en tant que chargé d'édition pour le compte de l'École Universitaire de Recherche ArTeC (EUR ArTeC). Les missions éditoriales de l'EUR ArTeC concernaient deux collections de livres produits à partir de documents issus de logiciels de traitement de texte et transformés avec le logiciel InDesign, ainsi qu'*Hybrid*, une revue numérique diffusée sur la plateforme OpenEdition. La chaîne éditoriale de la revue reposait essentiellement sur des fichiers de traitement de texte et le CMS Lodel pour transformer les documents en pages web.

L'observation des fichiers envoyés par les auteurs des livres ou d'un article de la revue a permis de mettre en lumière une pratique d'écriture pour le moins singulière. Dans certains documents, la structuration des éléments textuels, tels que les titres, ne semblait pas être le fruit de l'application d'un style conforme à la structure de l'objet souhaité. Par exemple, le choix d'un niveau de titre n'était pas la méthode employée pour structurer cet élément. Au contraire, il apparaissait que la structuration de ce titre était visuelle. Elle était obtenue par l'application d'effets graphiques, comme une emphase en gras couplée à un changement de la taille de la police, et cela, pour exprimer qu'un élément était un titre et non un paragraphe. Ce qui comptait pour les auteurs était donc que l'apparence des éléments textuels soit conforme à des structures graphiques que l'on rencontre habituellement dans les objets imprimés.

Cette expérience n'est pas isolée. À partir de 2022, à l'occasion de mon implication dans les projets de recherche Stylo et le Pressoir, j'ai été au contact de multiples auteurs et utilisateurs, que ce soit lors d'ateliers animés sur ces technologies ou lors d'accompagnement à la rédaction et à l'édition de textes pour des livres, pour des numéros de revue, pour des travaux étudiants, et d'autres objets. À certaines de ces occasions, la même expérience que celle décrite durant mon activité à l'EUR ArTeC s'est reproduite. La structure des informations textuelles ne suivait pas les bonnes pratiques de structuration des contenus. Elle suivait plutôt une structuration graphique et visuelle des informations. Lors d'ateliers d'initiation à l'éditeur de texte Stylo, une fois les premiers pas en Markdown effectués, les questions posées par les participants étaient souvent d'ordre graphique et ne concernaient pas la structuration sémantique du texte : comment puis-je mettre en page le contenu ? Est-ce que je peux afficher les titres plus gros ?

Du point de vue du responsable éditorial, lorsque l'on est confronté à un document mal formaté, c'est-à-dire à un document structuré graphiquement et non pas selon les syntaxes utilisées dans chaque environnement d'écriture,

on considère alors trop rapidement ou à tort que l'auteur de ce texte manque de littératie numérique et qu'il serait en quelque sorte un illettré numérique (Bouchardon et Cailleau 2018). Du côté des usagers, la première réaction face à un environnement qui ne permet pas d'écrire « comme dans MSWord » relève d'expressions négatives du type « l'interface n'est pas ergonomique », mais aussi « ce n'est pas intuitif » ou, mieux encore, « c'est plus facile d'écrire dans MSWord ».

Pour continuer dans cette direction, nous avons noté à plusieurs reprises lors d'ateliers d'écriture dans des environnements de texte brut que la plupart des rejets de cette forme d'écriture provenait du fait qu'il n'y avait pas de rendu graphique immédiat du texte saisi. D'ailleurs, Overleaf[5], la plateforme d'écriture collaborative en LaTeX, s'est emparée de cet enjeu et propose un aperçu au format PDF en direct du texte saisi en LaTeX. D'autres éditeurs de texte comme Notion ou Neovim proposent une interprétation directe du Markdown plutôt que de garder la structure du texte visible[6]. On remarque alors qu'il y a un attachement féroce aux rendus graphiques à l'écran, qu'ils soient de bonne ou de piètre qualité.

Pour autant, cette friction entre structuration graphique et structuration textuelle est-elle vraiment réductible à ces deux impasses ? C'est-à-dire soit à un manque dans les développements de la plateforme, soit à un manque de littératie numérique chez les usagers. La conclusion trop hâtive de cette situation ne repose finalement que sur une forme d'incompétence des uns ou des autres, conclusion à laquelle nous n'adhérons pas.

Il nous semble alors que les deux constats que nous venons d'évoquer se rejoignent et peuvent nous aider à nous sortir de ces impasses. Puisque nous partons du présupposé qu'il ne s'agit pas d'une quelconque forme d'incompétence chez des développeurs ou des utilisateurs, nous pouvons émettre l'idée que quelque chose se joue au niveau des 800 kilobytes de notre exemple précédent[7], c'est-à-dire au niveau des éléments structurels de nos textes numériques. Que se passe-t-il dans ces structures que l'on ne questionne jamais ?

Ces éléments structurels, que l'on rencontre dans tous les environnements d'écriture numérique, sont tour à tour invisibilisés ou discrétisés, comme dans les environnements de traitement de texte par exemple, mais pas uniquement, et doivent néanmoins jouer un rôle clef dans les situations inconfortables décrites

5. Overleaf est un éditeur LaTeX collaboratif en ligne orienté pour l'écriture scientifique et technique. Voir la plateforme à l'adresse suivante : `https://www.overleaf.com/`, consultée le 20 août 2025.

6. Les interprétations du texte brut dans Neovim ou Notion ne correspondent pas à une prévisualisation d'un document imprimé. Il s'agit juste d'un confort graphique à l'écran. Ce ne sont pas des applications du même type que MSWord ou LibreOffice.

7. Nous prenons ici l'exemple de la structure proposée par MSWord comme illustration de notre propos, toutefois l'enjeu est valable pour absolument tous les types d'environnement d'écriture numérique, comme nous le verrons plus loin dans la thèse.

précédemment. Si ce n'est pas le cas, à quoi bon leur octroyer un volume aussi grand, voire plus grand, que celui du texte qu'il structure ?

Le problème initial de cette recherche n'est pas nouveau, il est même partagé par plusieurs individus et communautés. Par exemple, du côté de la CRCEN, on peut trouver cette friction de l'écriture numérique dans un billet de blog publié en 2015. Celle-ci est signifiée par le constat qu'il n'y a pas d'environnement d'écriture adapté à l'écriture en Sciences Humaines et Sociales (SHS) (Vitali-Rosati 2015*a*). Autrement dit, les membres de la CRCEN estiment que l'architecture de 800 kilobytes, qu'elle provienne du format DOCX ou d'un autre environnement, ne répond pas aux exigences de l'écriture en SHS. En effet, les éléments de cette architecture ne permettent pas d'exprimer des besoins spécifiques à cette forme d'écriture qui ne seraient pas graphiques.

Voilà matière à enrichir notre problème. De façon similaire à la nôtre, ce constat réalisé par la CRCEN présuppose qu'il existe un enjeu structurel important dans l'écriture numérique et que les environnements d'écriture les plus populaires ne sont pas en mesure de répondre à toutes les formes d'écriture. Il y a donc un problème à l'intérieur de nos environnements d'écriture numérique où se joue cette friction que l'on observe entre un attachement à une représentation graphique de l'information et une incapacité à formaliser certains objets au-delà de cette représentation.

Mise en place de la problématique

Ces frictions que l'on rencontre dans les écritures numériques nous amènent à nous intéresser aux environnements qui permettent de les produire. Ceux-ci désignent des dispositifs (Bachimont 2010*b* ; Larroche 2018 : 18) de médiations techniques implémentés à l'intérieur d'un ordinateur. En règle générale, un ordinateur contemporain peut être défini comme l'assemblage de deux catégories d'éléments : d'une part, des composants électroniques rangés sous l'appellation *hardware*, et d'autre part, des *software* permettant d'interagir avec ce matériel et que l'on empile sous la forme de couches d'abstraction matérielle. Chaque couche d'abstraction matérielle a pour fonction de masquer la complexité de la couche située en dessous, tout en offrant à la couche supérieure une interface standardisée pour accéder aux ressources matérielles. Le système d'exploitation illustre bien ce principe : il constitue une couche intermédiaire qui permet aux logiciels applicatifs de manipuler le processeur, la mémoire ou les périphériques sans avoir à gérer directement leurs spécificités techniques. Grâce à cette organisation en strates, on peut faire évoluer ou remplacer certains composants matériels sans remettre

en cause l'ensemble du système, tant que l'interface fournie par chaque couche reste stable.

Lorsque l'on fait référence à un environnement d'écriture numérique et à toutes ses composantes, on le nomme généralement par le nom de son logiciel, par exemple le logiciel de traitement de texte *Pages* installé uniquement sur les ordinateurs Macintosh vendus par la firme Apple. Ce logiciel convoque alors l'ensemble des éléments qui le composent ; ce sont des normes, des formats, des protocoles, des licences, etc., ainsi que toute la partie *hardware* que nous venons d'évoquer.

Que l'on utilise des logiciels de traitement de texte, comme Microsoft Word, LibreOffice, GoogleDoc, ou encore des éditeurs de texte comme Vim, Ed ou Stylo, mais aussi des outils de conversion des documents tels que Pandoc ou des feuilles de transformation en langage *eXtensible Stylesheet Language* (XSL) pour transformer un document en un artefact publiable, chaque environnement d'écriture présente des caractéristiques qui lui sont propres et les différencient des autres environnements. Ces caractéristiques s'expriment notamment par des fonctionnalités sémantiques qui permettent de modéliser des contenus plus ou moins précisément selon les orientations de l'environnement employé. Par exemple, le cas de la description d'un titre dans un document est significative de ces fonctionnalités sémantiques. Un environnement d'écriture numérique est donc défini par toutes les composantes matérielles et logicielles nécessaires à la production d'un document numérique et de son contenu.

Dans un environnement d'écriture tel que LibreOffice, si l'on souhaite ajouter un titre à un document, la méthode la plus simple consiste à saisir une chaîne de caractères au tout début du document et d'y appliquer un style du type « Heading 1 » pour obtenir à la fois un rendu graphique conforme à ce que l'on attend d'un titre (taille de police plus grande et texte centré dans la page par exemple) et un balisage cohérent dans le format *Open Document Text* (ODT). Prenons par exemple la chaîne suivante : « Pour comprendre les médias ». Si celle-ci est traitée comme cela est mentionné ci-dessus, le lecteur averti comprendra que la chaîne de caractères « Pour comprendre les médias » est le titre du document. Il s'agit là d'une forme de littératie (Le Deuff 2012).

La littératie est la capacité à lire et à écrire dans un cadre socioculturel particulier et dépasse l'alphabétisation – la capacité à lire et écrire un ensemble de signes organisés. Le fait d'interpréter la chaîne de caractères « Pour comprendre les médias » en tant que titre de cet exemple relève d'un cadre socioculturel et donc d'une littératie. Si l'on réplique cet exemple dans un autre environnement, l'éditeur de texte Stylo, la déclaration du titre du document y est plus explicite et plus formelle, car elle ne dépend pas de l'application d'un style. Un espace de l'éditeur

de texte est réservé à cette énonciation sous l'apparence d'un formulaire composé d'un champ selon le principe de *clé : valeur*, comme dans un dictionnaire. Ainsi, dans l'éditeur de texte Stylo, le titre du document est décrit comme suit : `title: Pour comprendre les médias`. Cet exemple a priori anodin met en évidence des différences de pratiques entre plusieurs environnements d'écriture numérique. La culture numérique, ainsi que la littératie associée (Doueihi 2011), se fonde alors dans cette possibilité technique de décrire un objet d'une multitude de façons, qu'elles soient explicites, formelles ou ambiguës.

Depuis les années 1970, l'histoire des environnements d'écriture numérique met en exergue les volontés à l'œuvre dans l'élaboration et la conception de ces environnements. Malgré une compétition acharnée pour obtenir le monopole de ce marché dès l'avènement des ordinateurs personnels, puis des premières interfaces graphiques (Bergin 2006*a*,*b*), il existe aujourd'hui une pléthore d'environnements d'écriture différents que l'on peut trouver sur le marché des logiciels. Certains se veulent spécialisés, par exemple les éditeurs de texte orientés pour écrire du code informatique (Notepad++, VSCode et son pendant *open source* VSCodium, PyCharm si l'on veut utiliser un éditeur spécialisé pour un langage de programmation, etc.), ou encore des environnements avec une orientation académique comme Stylo, ainsi que bien d'autres spécialisations. Au contraire, d'autres environnements ont une approche plus généraliste et tentent de répondre à un vaste ensemble de besoins, c'est par exemple le cas des suites de bureautique dont on connaît au moins deux environnements, le premier sous une forme propriétaire avec la suite office de Microsoft et le second sous une forme libre avec LibreOffice. En outre, les différences que l'on observe entre tous ces environnements ne relèvent pas uniquement de l'ergonomie et du design, mais aussi de pratiques d'écriture différentes. Ces pratiques d'écriture englobent, d'une part, les préconisations de l'environnement d'écriture, que l'on retrouve implémentées à travers des éléments techniques tels que des formats, des fonctionnalités sémantiques, une architecture des informations, etc. ; et, d'autre part, les usages qui en sont faits, c'est-à-dire ceux que l'on observe dans les documents une fois qu'ils sont produits. « [L]es pratiques [d'écriture] et ses "techniques intellectuelles" sont intimement liés aux supports d'inscription et aux formes éditoriales qui les transmettent », nous dit Nicolas Sauret (2020 : 16). En ce sens, il n'y aurait pas une littératie numérique qui sous-entendrait une vision unifiée du numérique, à l'image d'une culture globale, mais plutôt des littératies numériques en concordance avec cette pluralité d'environnements et de pratiques.

Puisque les pratiques d'écriture varient selon les environnements dans lesquelles elles s'inscrivent, nous nous demandons de quelle manière ces environnements produisent-ils du sens dans l'acte d'écriture. Pour le formuler autrement,

notre postulat théorique issu de la théorie des médias permet de stipuler que les différences que l'on peut observer d'un environnement d'écriture à l'autre génèrent des variations dans les médiations techniques qui produisent l'écriture. Dès lors, cette problématique relève de ces médiations variées et du sens qu'elles produisent dans le document numérique.

Une partie de ces différentes médiations se cristallise sous une notion : le modèle textuel. Nous définissons ce dernier en tant que formalisation d'une théorie du texte, à l'instar de DeRose *et al.* (1990). En ce sens, un modèle textuel est implémenté dans un environnement d'écriture à travers des formats, des protocoles ainsi que des fonctionnalités. Loin d'être une notion éthérée et sans substance, le modèle textuel se matérialise non seulement à travers les objets que nous venons de mentionner, mais également à l'intérieur des documents numériques. Si nous reprenons la définition du document numérique donnée par Pédauque, soit un composite formé d'une structure et d'un ensemble de données (Pédauque 2006 : 40), le modèle textuel se matérialise sous la forme d'éléments structurels qui définissent le sens octroyé aux données qu'ils contiennent. Ces théories du texte traversent ainsi le processus de production de l'écriture.

Dès lors, nous pouvons considérer que les environnements d'écriture numérique produisent les contraintes de l'écriture, c'est-à-dire ce qu'il est possible d'écrire et ce qu'il n'est pas possible d'écrire. Ces contraintes suscitent ou induisent des pratiques d'écriture caractéristiques de l'environnement duquel elles proviennent. Au regard de cela, nous émettons l'hypothèse que ces environnements et notamment les modèles textuels qui y sont implémentés sont des coupures agentielles (Barad 2023) qui orientent l'écriture, qu'ils n'en sont pas séparables et que, par conséquent, en interagissant avec eux, ils participent à produire le sens d'un document et de son contenu.

Méthodologie

Pour répondre à cette problématique et réaliser cette analyse dispositive (Kovacs 2023), nous nous appuyons sur des méthodes employées par les *Science and Techonology Studies* (STS).

Tout d'abord, nous avons mené une étude ethnographique à travers plus de trois années d'observation participante au sein de la CRCEN à l'Université de Montréal, de janvier 2022 à mars 2025, dont deux années dédiées aux terrains que nous abordons par la suite. Au cours de cette période, la CRCEN a conçu et développé un ensemble de fonctionnalités pour l'éditeur de texte Stylo et le générateur de site statique le Pressoir. Ces deux environnements ont été utilisés pour produire des publications scientifiques.

Stylo a notamment été utilisé par la revue *Humanités numériques* pour la rédaction et l'édition d'un article, tandis que le Pressoir a été mobilisé par la maison d'édition les Ateliers de sens public comme chaîne éditoriale pour la production d'un livre. Du côté de la revue *Humanités numériques*, l'emploi de l'éditeur de texte Stylo dans leur chaîne éditoriale relevait d'une première expérimentation. Mon rôle consistait à assurer la médiation entre la CRCEN et la revue pour que toutes les étapes éditoriales se déroulent sans retarder le calendrier de publication. Tandis que du côté des Ateliers de sens public, qui utilise le Pressoir depuis plusieurs années, mon rôle était double. D'une part, en tant que codirecteur de l'ouvrage, je prenais part aux décisions éditoriales et assurais le suivi auprès des auteurs. D'autre part, pour la CRCEN, il me fallait assurer l'adaptation du Pressoir et l'édition de l'ouvrage en fonction de ces choix éditoriaux aux côtés d'Hélène Beauchef (éditrice) et David Larlet (développeur).

Au-delà de ces deux cas, mon implication dans le développement de ces outils à la CRCEN m'a donné un accès privilégié aux acteurs de leur conception, ce qui m'a permis d'identifier les éléments de production de sens intégrés à ces dispositifs d'écriture. Cette proximité avec les terrains sur une période aussi étendue génère une certaine difficulté à prendre de la distance vis-à-vis de ces derniers et peut générer quelques biais, notamment du fait de mon attachement aux technologies libres et *open source*.

En complément à cette méthode ethnographique, nous mobilisons la méthode *Biography of Artifacts and Practices* (BOAP) (Hyysalo et al. 2019 ; Silvast et Virtanen 2023). Cette méthode consiste à réaliser une biographie des technologies analysées à partir de divers éléments tels que des événements historiques significatifs, qu'il s'agisse, dans notre cas, d'événements liés à la technologie analysée ou des événements plus globaux liés à l'histoire des environnements d'écriture numérique. Nous mobilisons également différentes versions des documentations techniques des technologies analysées, représentant alors les théories du texte implémentées sous la forme de modèle textuel par les concepteurs de ces technologies, afin d'observer leurs évolutions dans le temps.

Nous nous appuyons également sur des documents produits par les environnements d'écriture numérique pour appréhender les usages de ces technologies. Pour ce faire, nous mobilisons une démarche de rétro-ingénierie des connaissances (Chikofsky et Cross 1990 ; Quintas et Demaid 2000) permettant d'analyser et de reconstruire *a posteriori* les étapes de la production du sens, « afin d'inférer les intentions et les projets des créateurs d'une technologie, quand ceux-ci ne sont pas accessibles car perdus [...] ou protégés par le secret industriel » (Severo 2013 : 53), comme c'est le cas de certains environnements que nous étudions.

L'intérêt de cette approche des technologies est de mettre en évidence la tension qu'il existe entre la modélisation de l'écriture réalisée par les concepteurs de ces technologies et la manière dont les utilisateurs se l'approprient.

Un des points d'attention de la méthode BOAP est que les événements doivent être considérés comme étant simultanément constitués par des modèles plus larges : le contexte de toute situation est compris comme étant composé d'éléments constitutifs à des rythmes variés, comme cela est présenté dans différentes théories telles que la cognition distribuée d'Hutchins (Hyysalo et al. 2019) que nous détaillons dans le premier chapitre. La cognition distribuée est un outil conceptuel pour décrire et analyser un événement, un moment, dans sa pluralité de processus tels qu'ils sont distribués dans des individus, des artefacts et des temporalités dans lesquelles ces événements se déroulent. Elle offre ainsi un cadre d'étude des relations entre les représentations internes (propres aux individus) et externes (véhiculées par les artefacts et outils) du traitement de l'information, notamment en examinant leur circulation entre ces différents acteurs cognitifs. De plus, elle intègre la dimension temporelle de l'événement, en mettant en évidence à la fois le rôle de l'expérience passée des acteurs dans l'anticipation des actions présentes et futures, mais aussi l'ordre dans lequel se déroulent les différences séquences de l'événement analysé par le biais de l'observation des transformations des informations qui circulent sur les médias mobilisés par cette cognition (Hutchins 1995).

Cette approche distribuée de l'écriture numérique et du sens qu'elle produit, à travers des techniques, des technologies et des pratiques, n'est pas sans rappeler la définition que donne Bachimont au concept de dispositif. Selon lui, le « dispositif correspond à une organisation spatiale d'éléments telle que cette configuration détermine un déroulement temporel » (Bachimont 2010*b* : 22). Pour le dire autrement, toujours selon les mots de Bachimont, « [le dispositif] construit dans le présent ce qui n'était que prévision pour le futur. Par conséquent, le propre du dispositif est de rendre actuel ce qui est prévu, de rendre présent au terme de son déroulement ce qui était à venir » (Bachimont 2010*b* : 44). Ce dispositif est donc caractérisé par une configuration spatiale d'acteurs qui modélise le déroulement des séquences d'un événement.

Pour donner à comprendre le dispositif, Bachimont lui octroie des formes de cohérence pour expliquer son fonctionnement : la cohérence interne, la cohérence concrète et la cohérence externe (Bachimont 2010*b* : 50). Ces trois cohérences correspondent à trois phases du dispositif : la formalisation, la modélisation et l'interprétation. La première est celle de l'invention scientifique, formelle par nécessité puiqu'elle doit être répétable et reproductible pour respecter les conditions de production la recherche. Toutefois, cette formalisation déroge à

une répétabilité sans faille lors de sa sortie du laboratoire, au moment de sa modélisation (par les ingénieurs), lorsque l'on entre dans la cohérence concrète du dispositif qui rencontre des contraintes physiques pour exister ; puis, elle y déroge à nouveau lors de sa troisième cohérence, externe, lorsque le modèle se confronte à une réalité sociale et à des interprétations par ses usagers. Cette dernière cohérence montre ainsi l'écart avec la « norme établie » et module le sens initialement modélisé (Bachimont 2010*b* : 60).

Nous mobilisons le dispositif de Bachimont comme un outil permettant d'effectuer un va-et-vient entre une phase de modélisation et une phase d'interprétation, mais aussi entre le laboratoire, espace dans lequel le modèle formel est inventé, et l'espace *au dehors* qui correspond à une autre configuration spatiale, comme le souligne Hutchins avec le titre de son ouvrage *Cognition in the wild* (Hutchins 1995 : xiii).

En nous appuyant sur cette définition, nous considérons que l'application de la méthode BOAP aux environnements d'écriture numérique permet d'étudier et de considérer l'écriture en tant que dispositif.

Pour compléter la méthode BOAP, nous ajoutons une étude comparée de plusieurs dispositifs afin de mettre en exergue les différents sens produits par chacun d'entre eux. Ce type d'approche a notamment été adopté par Kembellec et Scopsi dans leur étude des gestionnaires de références bibliographiques (Kembellec et Scopsi 2012), ou encore Roig et Ribera lorsqu'ils ont comparé certaines caractéristiques des modèles de document *Office Open XML* et *Open Document Format* (Roig et Ribera 2021). Nous ajoutons également un entretien semi-directif[8] mené en janvier 2024 avec Stéphane Pouyllau, responsable de l'Huma-num Lab et de la collaboration avec la CRCEN pour le compte d'Huma-num dans le cadre des développements de l'éditeur de texte Stylo, un de nos terrains.

Néanmoins, tous les outils de l'environnement d'écriture ne sont pas comparables entre eux puisqu'ils n'interviennent pas au même moment du dispositif d'écriture. Aussi, nous configurons ce dispositif en deux phases : celle de production et celle de publication de l'écriture. La première est relative à toute l'étape de rédaction et de production d'un texte et d'un document, tandis que la deuxième fait référence aux transformations des documents produits pour qu'ils soient adaptés à un format de publication. Le processus d'écriture est ainsi étendu à la quasi-totalité d'une chaîne éditoriale que l'on peut définir comme :

> des ensembles techniques qui rassemblent les outils et les rouages nécessaires à la production de fichiers informatiques, pour l'impression de documents paginés ou pour la mise à disposition de livres

8. La transcription de l'entretien se trouve dans l'annexe V.5

> numériques dans des formats divers et sur des plateformes multiples (Fauchié 2024 : 25).

Par soucis de cohérence tout au long de la thèse, nous rattachons donc certains outils à la phase de production de l'écriture et d'autres à la phase de publication.

L'application de cette méthodologie est réalisée sur cinq environnements d'écriture numérique, dont trois appartiennent à la phase de production et deux à la phase de publication. Pour les trois premiers environnements, notre choix s'est d'abord porté sur deux logiciels *What You See Is What You Get* (WYSIWYG) emblématiques, Microsoft Word (MSWord) ainsi que LibreOffice. Ces deux environnements sont des incontournables lorsqu'il s'agit d'écrire avec un ordinateur. MSWord est l'un des logiciels de traitement de texte les plus vieux encore sur le marché depuis sa première version lancée en 1983. Grâce à ce logiciel, Microsoft a détenu plus de 90% de parts de marché du traitement de texte à partir de 1994 (Kirschenbaum 2016), et ce, jusqu'à ce qu'OpenOffice (devenu par la suite LibreOffice) adopte un format de fichier ouvert reconnu comme norme internationale en 2006 (Yami et al. 2015). Il est par la suite devenu le concurrent libre de MSWord. Depuis le début des années 2010, LibreOffice est la suite de bureautique nativement installé sur les distributions Linux les plus populaires et certaines distributions BSD. Pour ces raisons, il nous a semblé évident d'étudier ces deux environnements.

Le troisième environnement de production de l'écriture numérique est un éditeur *What You See Is What You Mean* (WYSIWYM) : Stylo. À la différence des deux environnements précédents, Stylo ne fait pas l'objet d'une distribution massive. Cet outil a été développé dans le cadre d'un projet de recherche sur les écritures numériques mené par la CRCEN sous la direction de Marcello Vitali-Rosati (2020). Dans le cadre de l'installation logicielle, il est à noter que les solutions bureautiques LibreOffice et MSWord sont installées localement sur les ordinateurs, tandis que Stylo est accessible à distance via le Web. Par ailleurs, il convient de préciser que cet outil a été conçu pour un type d'écriture spécifique, à savoir celui des SHS, contrairement aux traitements de texte qui se veulent plus généralistes. Les propriétés de Stylo en font un choix intéressant à mettre en perspective avec les deux environnements précédents. Leur comparaison donnera à voir les différences de production du sens dans les documents issus de ces trois environnements.

Les deux derniers environnements que nous avons sélectionnés appartiennent à la phase de publication de l'écriture. Ces outils permettent de transformer les documents issus de la phase de production en objets publiables. Il s'agit du système de gestion de contenus (CMS) Lodel et du générateur de site statique le Pressoir. Lodel est le CMS conçu pour la gestion des contenus publiés sur la plateforme

de diffusion OpenEdition à partir du début des années 2000, notamment ceux de plus de 650 revues. Lodel prend principalement en charge des documents au format DOCX, mais également des documents produits avec Stylo, en passant par la chaîne de traitement Métopes développées à l'université Caen Normandie.

Tout comme Stylo, le Pressoir est un outil également développé par la CRCEN. La démarche conceptuelle derrière ce projet est de produire des livres dans plusieurs formats, notamment des livres web augmentés non homothétiques avec leur pendant imprimés. Pour réaliser les transformations nécessaires à la création de tels objets, le Pressoir s'appuie sur le logiciel de conversion Pandoc, un convertisseur très populaire depuis presque vingt ans pour convertir des fichiers formatés avec des langages de balisage. Les particularités de Lodel et du Pressoir en font des terrains complémentaires à ceux de la phase de production. Ils nous permettent ainsi d'observer plusieurs dispositifs d'écriture depuis les débuts de la production d'un texte, jusqu'à sa publication.

Afin de traverser ces cinq terrains, nous avons convenu de deux cas d'étude. Le premier cas d'étude est celui de la rédaction et de l'édition d'un article écrit dans Stylo et publié dans la revue *Humanités numériques* sur la plateforme OpenEdition Journals via le CMS Lodel. Le second cas d'étude est celui de la rédaction des chapitres du livre *Contribution numériques : cultures et savoirs* dans les logiciels de traitement de texte MSWord et LibreOffice, puis transformés à l'aide du Pressoir pour produire le livre en plusieurs formats.

Que ce soit l'article publié sur OpenEdition ou le livre produit avec le Pressoir, les écritures que l'on y étudie ne relèvent que d'un seul type : l'écriture scientifique, et plus précisément l'écriture en SHS. Néanmoins, ce ne sont pas les usages de ces différents environnements que nous étudions. Malgré le fait que notre constat initial provient de ces usages, nous nous concentrons principalement sur les documents numériques produits par ces derniers. Ce n'est qu'à partir de ces documents et de leur contenu que nous analysons les modalités de production du sens de chaque environnement d'écriture numérique.

Plan de la thèse

L'argumentation de cette thèse est développée à travers cinq chapitres. Les deux premiers présentent une revue de la littérature autour des notions d'écriture et d'écriture numérique, tandis que les trois chapitres suivants sont consacrés à l'étude des environnements d'écriture numérique.

Dans le premier chapitre intitulé *Écriture et production du sens*, nous analysons le dispositif qu'est l'écriture à travers la théorie des médias et une approche matérialiste, en soulignant son caractère distribué dans tous les éléments de son

environnement, notamment les éléments techniques. Nous nous appuyons à la fois sur les travaux de l'école de Toronto depuis McLuhan, puis ceux de Kittler (2015 ; 2018), ainsi que ceux de l'école de l'intermédialité montréalaise, et sur la notion d'énonciation éditoriale telle qu'elle a été développée en sciences de l'information et de la communication (Jeanneret et Souchier 2005 ; Souchier 1998) pour montrer que l'écriture dépend des médiations techniques. Cependant, ces perspectives restent marquées par un certain déterminisme, séparant individu et support d'écriture. Ce paradigme est remis en question par le posthumanisme et le nouveau matérialisme, notamment avec Barad (2007 ; 2023), qui propose une causalité inspirée de la physique quantique, où le message et le médium ne se distinguent qu'après qu'il y ait eu intra-action. Ainsi, nous adoptons cette perspective pour décrire l'écriture comme un produit des médiations techniques spécifiques, impossible à dissocier de son environnement matériel.

Dans le deuxième chapitre, intitulé *Écriture numérique et production du sens*, nous nous penchons sur ce que le numérique apporte de spécifique à l'acte d'écrire. Nous analysons l'écriture numérique à partir des caractéristiques propres aux ordinateurs, décrites dans les sciences de l'information et de la communication comme la calculabilité, la variabilité et l'exécutabilité. La calculabilité correspond à la capacité de l'écriture numérique à être quantifiée et traitée par la machine (Crozat et al. 2011) ; la variabilité renvoie à la possibilité de modifier dynamiquement les valeurs stockées en mémoire (Bouchardon 2014) ; l'exécutabilité désigne, quant à elle, la faculté d'un texte à déclencher des actions via du code informatique, c'est-à-dire des écritures « capables [...] de lire et d'écrire par elles-mêmes » (Kittler 2015 : 30 ; Sauret 2020 : 101). Ces propriétés fondamentales structurent l'écriture numérique à tous les niveaux, des couches matérielles les plus basses aux couches logicielles les plus hautes. Pourtant, comme nous le voyons dans la section II.3, les interfaces graphiques des environnements d'écriture numérique véhiculent une métaphore, celle de la page, qui en devient un objet skeuomorphique par sa fonction de garde-fou. En effet, la page permet à la fois de préserver des repères issus de la culture de l'imprimé dans la représentation de l'écriture, ainsi que de masquer les caractéristiques distinctives de l'écriture numérique. En s'affranchissant de cette page dans la section II.4, nous revenons sur les notions de document numérique et de modèle textuel pour montrer comment s'incarnent ces caractéristiques dans le processus d'écriture numérique et dans la production du sens qui en découle.

Dans les troisième, quatrième et cinquième chapitres, nous nous intéresserons aux couches les plus hautes des environnements d'écriture numérique à travers nos deux cas d'étude. Chacun de ces cas est scindé selon les deux phases de production et de publication de l'écriture numérique. Les chapitres III et IV

concernent la phase de production, tandis que le chapitre V traite de la phase de publication. Les environnements mobilisés dans chacune de ces phases sont analysés selon la méthodologie précédemment décrite.

Dans le troisième chapitre, *Écrire avec des logiciels* What You See Is What You Get, nous nous focalisons sur la comparaison de deux environnements de traitement de texte utilisés pour écrire les sources du livre *Contribution numérique* : MSWord et LibreOffice. Pour comprendre les choix de développement nous menant aux fonctionnalités sémantiques présentes dans ces deux environnements, nous traversons d'abord l'histoire des interfaces graphiques et du mode WYSIWYG prédominant dans les environnements d'écriture numérique depuis les années 1980. Ceci nous permettra de mieux appréhender les contenus des sources du livre *Contribution numérique* envoyées par les auteurs, de saisir et de comparer le sens produit dans ces documents.

Lors du quatrième chapitre, *Écrire avec Stylo*, nous analysons l'éditeur de texte Stylo à travers son utilisation pour la rédaction d'un article scientifique publié dans la revue *Humanités numériques*. Contrairement aux logiciels de traitement de texte étudiés dans le chapitre III, Stylo est un éditeur WYSIWYM développé pour répondre aux besoins d'écriture en sciences humaines et sociales. À cette occasion, nous pourrons nous demander ce que sont ces besoins d'écriture et de quelle manière ils sont implémentés, puis nous pourrons observer le sens qu'ils produisent dans cet environnement d'écriture numérique. Au-delà de cette analyse, nous comparons également le document produit dans l'environnement Stylo avec ceux issus de MSWord et LibreOffice étudiés dans le chapitre III afin de mettre en lumière les différentes matérialités émergentes dans ces documents et les différents sens que chaque environnement génère.

Le cinquième chapitre aborde la transformation des textes sources en artefacts publiables, élargissant l'étude des environnements d'écriture à l'ensemble des étapes et opérations nécessaires à la publication, qu'elle soit imprimée ou numérique. Nous montrons que les transformations éditoriales ne se limitent pas à la mise en page d'un document, mais modifient la structure du document et l'inscrivent dans des réseaux d'objets, un processus appelé éditorialisation (Broudoux 2022 ; Merzeau 2013*a* ; Vitali-Rosati 2016). Transformer un document implique d'adapter son modèle textuel à celui du format cible, ce qui peut enrichir, appauvrir ou altérer le sens du texte. Le sens du document continue ainsi d'évoluer au-delà de la simple saisie, à travers un ensemble de transformations. Pour illustrer cette dynamique, nous analysons les transformations des documents étudiés dans les chapitres précédents : l'article de la revue *Humanités numériques*, passé de Stylo au Web sur la plateforme OpenEdition via Lodel, et le livre *Contribution numérique*, converti à partir de MSWord et LibreOffice

vers les formats *Electronic Publication* (ePUB), *HyperText Markup Language* (HTML) et PDF. La comparaison des documents avant et après transformation met en évidence ces variations de sens.

Enfin, la conclusion met en perspective l'ensemble de nos analyses pour souligner que, malgré certaines équivalences apparentes entre les environnements d'écriture numérique étudiés, chacun produit un sens qui lui est spécifique. Nous y proposons également quelques pistes de réflexion sur de nouvelles modalités de production du sens pour de futurs environnements d'écriture numérique.

Chapitre premier

Écriture et production du sens

I.1 Introduction

L'écriture est un sujet controversé qui fait toujours couler beaucoup d'encre à son sujet. La littérature y est très abondante et, à notre regret, nous ne pouvons pas en faire le tour. Si l'on s'en tient à un nombre restreint de disciplines académiques issues des sciences humaines et sociales, définir l'écriture relève tout autant de la philosophie contemporaine, que de l'anthropologie, de l'ethnologie ou des sciences de l'information et de la communication. On pense notamment aux travaux de Leroi-Gourhan pour qui l'écriture mêle le geste et la parole (Leroi-Gourhan 1964), ou ceux, plus récents, d'Ingold sur l'histoire des lignes (Ingold 2013), ou encore les travaux en littérature de Christin sur l'origine de l'écriture (Christin 1999), ainsi qu'à ceux de Souchier sur la lettrure (Souchier 2012) et ceux de Paveau sur la distribution de l'écriture (2015).

Nous constatons qu'il s'agit là d'un sujet quasiment inépuisable et en constante reconfiguration selon les époques et les techniques auxquelles nous recourrons pour justement pouvoir écrire. Cette dimension technique est sans doute celle qui nous intéresse le plus dans le cadre de notre recherche. Mobiliser la technique, c'est également considérer la place des supports dans l'écriture. Le support devient alors un élément indissociable de l'histoire de l'écriture puisqu'il en détermine les configurations, comme l'ont démontré Goody avec la raison graphique (1979) et Bachimont à sa suite avec la théorie du support (2010*b*) et la raison computationnelle (2000). Cette dimension technique de l'écriture sera notre fil d'Ariane tout au long de ce chapitre.

Afin de l'initier, nous proposons une définition assez large de l'écriture, de manière que nous puissions disposer d'un objet commun auquel se référer sans pour autant prendre parti dans les multiples controverses qu'elle suscite et qui animent les travaux que nous venons de mentionner. Ainsi, nous définissons l'écriture comme un mode d'expression et une « fonction de communication » au sein d'une société (Christin 1999 : 30).

Saisie sous cet angle, et puisque l'histoire de l'écriture n'est pas notre sujet, l'écriture échappe notamment à la controverse de son origine, à savoir si elle provient du paradigme de la trace, étant soit comprise comme le signe verbal transposé sur un support soit comme la marque laissée par un corps, ou si elle provient du paradigme du signe selon son sens étymologique d'« événement inaugural [qui] participe d'une révélation » tant qu'il s'inscrit dans un « système » tel que la disposition des entrailles d'une bête sacrifiée lors d'une cérémonie (Christin 1999 : 32). Circonscrire l'écriture selon des modalités d'expression et de communication, c'est avant tout mettre en avant une propriété qui peut caractériser toutes les formes d'écriture, même numérique. Lorsque l'écriture est convoquée, elle fait ainsi appel à deux actions : d'abord l'inscription, puis la lecture (Pédauque 2006).

Qu'elles soient réalisées en temps réel ou en temps différé, ces actions laissent nécessairement des traces ou des indices dans la matière, car l'écriture est avant tout un phénomène matériel. Sans cette matérialité, l'écriture n'a plus lieu d'être.

En ce sens, les agents mobilisés à travers ce phénomène sont multiples et relèvent de toutes les entités nécessaires à l'émergence de l'écriture. Autrement dit, l'écriture n'est pas seulement une affaire d'émetteur et de récepteur (ou d'écrivain et de lecteur, quitte à ce que ces deux fonctions soient assumées par la même personne), c'est un phénomène avec lequel les outils, les techniques, les matériaux physiques sont primordiaux et irréductibles. Comprise de cette manière, l'écriture devient un dispositif technique dans le sens que lui donne Bachimont, c'est-à-dire « une organisation matérielle et spatiale capable de produire et de déterminer un devenir » (Bachimont 2010*b* : 42). Nous ne comprenons pas la définition évoquée par Bachimont dans un sens déterministe et nous revenons sur cette notion dans la section I.3 de ce chapitre. À ce stade, retenons que cette détermination n'est pas un phénomène linéaire prédictible. Observée en tant que dispositif, l'écriture apparaît sous une nouvelle perspective : les éléments qui construisent l'écriture sont disséminés dans diverses entités comme peuvent l'être le choix de la langue employée, le support physique sur lequel on réalise une inscription ou encore le mode de transport (s'il y en a un) de ce dernier. Cette liste est loin d'être exhaustive. Elle illustre simplement le fait que chacun de ces exemples est un élément du dispositif et qu'il participe donc à la production de l'écriture.

À partir de cette définition de l'écriture, la problématique qui nous anime questionne les modalités de production du sens de l'écriture, notamment quant à la place et au rôle de l'environnement technique dans cette production. Parmi ces agentivités cognitives participantes de ce dispositif, est-ce que la technique produit-elle du sens ?

Notre hypothèse est que les dispositifs d'écriture forment un ensemble plus ou moins complexe de médiations, selon la technique employée. Que l'on écrive avec un crayon sur une feuille de papier, avec une craie sur une ardoise ou avec une machine à écrire, il y a toujours des médiations (techniques) entre le geste d'écriture et son incarnation matérielle dans un document. En contraignant matériellement l'écriture, ces techniques contraignent ce qu'il est possible d'écrire et ce qui ne l'est pas. Ces médiations techniques deviennent porteuses du sens produit durant l'action d'écriture.

La vérification de cette hypothèse est réalisée en deux sections. Dans la première, nous y testons l'idée que l'écriture est un phénomène distribué. En nous appuyant sur les travaux de distribution de l'écriture que Paveau a initialement réalisés pour l'écriture numérique, et que nous employons plus largement pour toutes écritures, nous émettons l'hypothèse que l'écriture est un acte qui dépasse le simple geste d'inscrire un signe sur un support, et que ce geste incarne plutôt un ensemble de dynamiques culturelles, techniques, technologiques et sociales. De Leroi-Gourhan à Goody, en passant par Hutchins, nous nous intéresserons à la fois aux effets provoqués par l'usage de l'écriture, notamment en termes de possibilités de structuration des informations que la parole ne connaît pas, ce que Goody nomme la raison graphique, puis, en nous appuyant sur les travaux d'Hutchins, sur sa distribution au sein des éléments constituants un dispositif. Au même titre qu'Hutchins identifie une distribution de la cognition dans ce dispositif, nous y voyons une distribution de l'écriture organisée selon des protocoles propres à cet environnement. Nous illustrons cette distribution de l'écriture à travers deux exemples : les *Lettres à Lucilius* rédigées par Sénèque et les *Pensées pour moi-même* de Marc Aurèle. En nous appuyant sur Hadot, nous mobilisons ces textes anciens pour la raison qu'ils permettent d'observer les effets de l'écriture sur les modalités de production de la pensée et peuvent être mis en vis-à-vis avec des modalités orales de cette production. Malgré le fait que l'origine de l'écriture remonte à encore plus loin dans le temps, la pratique de l'écriture de soi réalisée par ces stoïciens permet d'observer les caractéristiques de l'écriture, ainsi que les effets d'un tel dispositif sur la production du sens.

Tandis que la section I.2 de ce chapitre nous permet de commencer à lever la dichotomie entre inscription et support, nous nous intéressons dans la section I.3 aux médiations productrices de l'écriture, sous l'angle de la théorie des médias telle qu'elle est pensée depuis l'école de Toronto. Plutôt que de conceptualiser l'inscription d'un côté, telle qu'elle serait pensée par un auteur, et le support de l'autre, prêt à recevoir cette inscription, nous envisageons l'écriture comme un tout, c'est-à-dire comme un environnement dans lequel chaque entité agit

sur l'autre. Les travaux de McLuhan, de Kittler et de l'école de l'intermédialité montréalaise nous ont été précieux pour envisager l'écriture sous un tel prisme. Néanmoins, les deux premières théories présentent un aspect critique de déterminisme technologique que la troisième dépasse implicitement, et qui nous empêche de mener à bien cette réconciliation. En nous appuyant sur le nouveau matérialisme et le posthumanisme, nous montrons qu'une autre forme de causalité agentielle est possible entre ces entités agissantes dans l'écriture. Ce faisant, il devient possible de considérer le support comme partie intégrante de l'écriture, comme une dynamique, et non plus comme un élément neutre et inerte qui en serait séparé.

I.2 L'écriture est distribuée

À partir de la définition de l'écriture donnée en introduction, nous proposons de considérer l'écriture comme un phénomène distribué à travers tous les éléments qui en composent le dispositif. Cette proposition n'est pas anodine puisqu'elle écarte totalement la définition de l'écriture perçue comme une production uniquement humaine, c'est-à-dire produite par un seul centre de contrôle, la cognition humaine (et le geste technique qui l'accompagnerait). Le fait de réfuter cette conception anthropocentrée de l'écriture n'a rien de nouveau, bien au contraire. Depuis longtemps déjà, l'écriture n'est pas considérée en tant que fait humain.

Prenons pour exemple le *Phèdre* de Platon, car ce texte abonde déjà dans ce sens d'une écriture qui n'est pas d'origine humaine. Socrate, le personnage, révèle à Phèdre, ainsi qu'au lecteur, l'origine divine de l'écriture. Celle-ci aurait pour père le dieu Theuth (274*c*–*d*). C'est Theuth qui, en premier, aurait découvert les arcanes de l'écriture. L'écriture est donc ici présentée en tant qu'art découvert par une divinité et non par l'humanité. Ce n'est qu'ensuite que Theuth aurait transmis cet art au roi d'Égypte Thamous. Ce texte dont le sujet porte plutôt sur la transition entre oralité et scripturalité, ne comporte aucun doute sur la condition extérieure à l'humain de l'écriture.

Dans un contexte plus contemporain, Ong, en s'appuyant sur cette origine divine décrite dans le *Phèdre*, poursuit dans cette direction et va jusqu'à qualifier l'écriture d'inhumaine, puisqu'elle « elle prétend établir en dehors de l'esprit ce qui ne peut être en réalité que dans l'esprit ». Il ajoute également que pour Socrate, le personnage, « elle est une chose, un produit manufacturé » séparé de sa source vivante et de l'esprit d'où émerge l'idée avant qu'elle ne soit inscrite sur son support (Ong 2014 : 98). Pour Ong, l'écriture est une technique, puisqu'elle mobilise toujours des outils, que ce soit une feuille ou un pinceau, qu'elle soit numérique ou non, et implique, comme toutes les techniques, « des transformations

intérieures de la conscience ». Le fait de qualifier l'écriture d'inhumaine renvoie alors à cette possibilité que l'écriture, en tant qu'élément extérieur à l'humain, participe à la construction de la pensée et de l'individu.

Afin de mieux comprendre ce qu'impliquerait la participation de ces éléments extérieurs à l'être humain dans la production du sens, nous nous tournons vers plusieurs théories, ou plutôt vers un ensemble d'intrications théoriques provenant de plusieurs disciplines et qui intègrent le non-humain dans leur approche. Les théories à ce sujet, même si elles ne sont pas majoritaires, foisonnent depuis plusieurs siècles. Leroi-Gourhan, dont l'une de ses thèses est justement que la technique façonne les êtres humains et la pensée, cite *La Création de l'homme* de Grégoire de Nysse (379 ap. JC.) dans son ouvrage *Le Geste et la parole* :

> Ainsi, c'est grâce à cette organisation que l'esprit, comme un musicien, produit en nous le langage et que nous devenons capables de parler. Ce privilège, jamais sans doute nous ne l'aurions, si nos lèvres devaient assurer, pour les besoins du corps, la charge pesante et pénible de la nourriture. Mais les mains ont pris sur elles cette charge et ont libéré la bouche pour le service de la parole (Leroi-Gourhan 1964 : 59).

En s'appuyant sur Grégoire de Nysse, Leroi-Gourhan soutient que le développement du cerveau – le siège de l'intelligence humaine – est une conséquence biologique de l'évolution des êtres humains et non sa propriété principale, contrairement aux idées évolutionnistes largement répandues durant le XIXe siècle (Leroi-Gourhan 1964 : 30). Dès lors, il échappe ainsi au :

> déterminisme aboutissant à des formes vivantes de plus en plus adaptées aux mobiles de l'exploitation de la matière [dont] pour nombre d'évolutionnistes, [...] le fait significatif est le développement toujours plus grand du cerveau et de ses dépendances nerveuses (Leroi-Gourhan 1964 : 109-10).

Pour le dire autrement, Leroi-Gourhan émet l'hypothèse que les développements du cerveau sont liés à « l'adaptation des structures corporelles » aux environnements dans lesquels le corps se situe. La cognition ou la pensée seraient alors des conséquences de ces adaptations techniques et non l'inverse. La pensée est quelque chose qui vient après les changements techniques et non pas avant, ce qui veut dire que la technique joue un rôle dans le fonctionnement de la pensée. Dès lors, nous pouvons nous demander si les techniques d'écriture n'ont pas, à leur tour, libéré quelque chose dans la pensée, comme ce fut par exemple le cas de l'empereur Marc Aurèle que nous abordons plus loin.

Cette question de ce que peut libérer l'écriture dans la pensée est une brèche dans laquelle se sont engouffrées les sciences de l'information et de la communication, notamment à partir de Leroi-Gourhan. Parmi tous les travaux à ce sujet dans cette discipline, un article attire particulièrement notre attention. Il s'agit de celui rédigé par Candel *et al.* en 2019 à l'intérieur duquel ils nous proposent un tour d'horizon de l'épistémologie de l'écriture développée par la revue *Communications et langage* depuis ses débuts (Candel et al. 2019). Cet article est composé par une série d'entrées ordonnées par ordre alphabétique, à la façon d'un lexique. Ces entrées, commençant toutes par la lettre « e », couvrent l'ensemble du processus éditorial de la revue, en débutant par l'entrée « écriture » et en terminant par l'entrée « évaluation ».

À partir de la perspective anthropologique proposée par Leroi-Gourhan, l'écriture est d'emblée considérée en tant que dispositif de communication au sein d'une société et dépasse le simple « tracé, manuel ou mécanique, de caractères alphanumériques qui la définit usuellement » afin d'être étendu à un dispositif où l'écriture est couplée à la lecture (Candel et al. 2019 : 109). C'est-à-dire que dans ce dispositif, l'action de lecture joue un rôle majeur dans la production du sens, puisque le lecteur interprète et s'approprie le texte.

De plus, une autre fonction est insérée entre l'écriture et la lecture. En conceptualisant l'écriture à partir du processus éditorial d'une revue scientifique, comme cela est indiqué à l'entrée « ÉDITEUR » (Candel et al. 2019 : 113) de cet article-lexique, les auteurs lui confèrent une multiplicité d'actions qui contribuent à complexifier sa polysémie. La première de ces actions est la fonction éditoriale assurée par la revue, qui intervient en transformant le texte initial afin de l'adapter et de le préparer pour sa publication. Cette médiation éditoriale prépare ainsi le texte à être présenté à son lectorat. La seconde fonction est donc celle de la lecture, qui, en recevant et interprétant ce texte transformé, complète le processus. Ainsi, l'écriture ne se limite pas à sa production initiale, mais s'inscrit dans une dynamique englobant la transformation éditoriale et la réception par les lecteurs. Ces deux fonctions participent alors activement à la production du sens.

Cet ensemble de fonctions qui gravite autour de la production initiale, le « tracé », est rangé sous la notion qu'est l'énonciation éditoriale. Cette énonciation éditoriale n'est pas réduite « au métier d'éditeur, mais concerne un mode spécifique d'intervention sur l'ordre et l'image du texte » (Jeanneret et Souchier 2005 : 4). Plus précisément, Souchier la définit comme suit :

> [Ce concept] annonce une théorie de l'énonciation polyphonique du texte produite ou proférée par toute instance susceptible d'intervenir dans la conception, la réalisation ou la production du livre, et plus généralement de l'écrit. Au-delà, il intéresse tout support associant

> texte, image et son, notamment les écrans informatiques – étant entendu que tout texte est vu aussi bien que lu (Souchier 1998 : 141).

L'énonciation éditoriale désigne alors tous ces processus invisibilisés de transformation du texte pour le rendre accessible aux lecteurs. Ces transformations permettent au texte de suivre une série de conventions éditoriales et graphiques qui « [donnent] à lire » le texte (Candel et al. 2019 : 114). La visée de cette énonciation éditoriale est donc la lecture. D'ailleurs, l'énonciation éditoriale est l'une des notions fondamentales à l'éditorialisation, comme nous le verrons dans le chapitre V. Pour l'instant, nous pouvons retenir que l'éditorialisation diffère de l'énonciation éditoriale par sa visée qui consiste en l'inscription d'un texte dans un réseau d'objets, comme le souligne Merzeau (2013*a*), Broudoux (2022) et Vitali-Rosati (2016).

L'action de lecture est ambivalente dans ce dispositif, car elle est à la fois convoquée comme une finalité, mais aussi comme une nécessité à la production de l'écriture et du sens. Selon l'énonciation éditoriale, pour amener le texte à sa finalité qu'est la lecture, il est nécessaire de lire pour écrire, c'est-à-dire de maîtriser l'environnement technique de production de l'écriture. Pour désigner cette capacité jointe de lecture et d'écriture qui s'entremêlent, Souchier fait appel au concept de « lettrure » (Souchier 2012). La lettrure est un doublet datant du XIIe siècle et tombé en désuétude au XVe siècle avec l'apparition de l'imprimerie et de l'humanisme. Souchier remet en circulation ce terme afin de traduire l'anglicisme littératie (*literacy*), pour lequel il n'existe pas réellement d'équivalent en français, car il désigne précisément la capacité à lire et à écrire dans un contexte social particulier. Souchier en profite aussi pour lier la notion de lettrure à un geste fondateur de la culture numérique, qui requiert justement cette compétence de lecture et d'écriture pour agir dans cet environnement.

Ainsi, les sciences de l'information et de la communication, notamment à travers les notions que sont l'énonciation éditoriale et la lettrure, nous montrent que le dispositif d'écriture s'étend au-delà d'un *logos* et d'un tracé sur une surface. C'est un dispositif bien plus complexe qui met en œuvre des dynamiques qui dépassent l'individu et l'acte de saisi du texte, et cela quel que soit le niveau de complexité technique pour l'effectuer. Ce dispositif mobilise tout un ensemble de conventions, à travers des formats, des normes typographiques, des protocoles de circulation de ces objets et tout un ensemble d'acteurs pour produire l'écriture. L'écriture et son sens ne sont pas uniquement produits par le tracé initial, mais par l'ensemble des fonctions appelées par cette énonciation éditoriale et qui font intégralement parties du dispositif d'écriture, car sans elles, l'écriture ne saurait être produite.

Il y a donc dans le dispositif d'écriture des éléments nécessaires à sa production et qui pourtant dépassent l'écrivain. L'écriture est, d'une certaine manière, extérieure à l'individu. Elle est externalisée, si l'on reprend le terme proposé par Marie-Anne Paveau. Pour aborder cette externalisation, Paveau développe une approche non-anthropocentrée de la production du sens, notamment :

> [une] théorie du discours retravaillée par la cognition sociale dans sa version distribuée : l'intelligence est distribuée dans les environnements, humains comme non humains, et l'objet d'analyse est le système, et non seulement les énoncés ou les locuteurs, qui n'en constituent qu'une partie. Il s'agit donc d'une conception du discours comme intégré dans l'ensemble de l'environnement humain et non humain, et non distingué de cet environnement : je souhaite éviter l'approche « égocéphalocentrée » (expression de C. Brassac qu'il emprunte à J.-P. Kaufmann, et qui désigne une approche internaliste centré sur le sujet-locuteur) et logocentrée. (Paveau 2012 : § 4).

Cette théorie rejoint en un sens le propos de Leroi-Gourhan, excepté le fait qu'elle ne se limite pas seulement à la technique, mais concerne également l'ensemble de l'environnement humain et non-humain. Cette perspective d'une « cognition socialement partagée [et] distribuée » ouvre la possibilité d'une cognition externe à la fois sociale, mais aussi culturelle et plus seulement internaliste (à l'intérieur du cerveau) (Paveau 2015 : § 6).

Pour construire cette réflexion, Paveau s'appuie avant tout sur la cognition distribuée développée par Edwin Hutchins dans les années 1990 (Paveau 2015). Le texte d'Hutchins s'intitule *Cognition in the Wild* et a été publié en 1995[1]. La publication de cet ouvrage majeur fait suite à une recherche ethnographique menée pendant plusieurs années auprès de l'*US Navy*. C'est grâce à une pratique antérieure de la navigation et surtout à l'observation des manœuvres de navigation réalisées par l'équipage de *l'USS Palau*, un bâtiment amphibie de transport d'hélicoptères, qu'Hutchins développe cette notion de « cognition distribuée » (Hutchins 1995).

Dans ce texte, Hutchins nous explique que manœuvrer un bâtiment de plusieurs milliers de tonnes et de presque 200 mètres de long n'est pas une action réalisable par une seule personne. Pour accomplir cela, un navire a besoin de tout un équipage coordonné, que l'on peut répartir en plusieurs organes nécessaires à la navigation. Tout d'abord, plusieurs membres d'équipage doivent être affectés

1. *Cognition in the wild* d'Hutchins n'est pas à confondre avec *La pensée sauvage* de Lévi-Strauss. Chez Hutchins, cette pensée n'est pas à mettre en opposition avec une pensée domestiquée. La pensée sauvage réfère ici à une pensée hors laboratoire, une pensée observée dans son « habitat naturel », entourée d'une culture, de pratiques, de techniques et d'aléas.

directement aux commandes de la navigation, un organe de prise de décision de la route à suivre ou de la manœuvre à effectuer. Ensuite, d'autres marins doivent s'occuper de la propulsion du bateau, il faut donc des mécaniciens parés à modifier l'état de la propulsion en tout temps. Un navire de cette taille est également doté d'organes sensoriels, des sonars, des radars, des radios, des jumelles, de sondes, etc., afin de lui permettre d'appréhender le monde extérieur. Pour utiliser ces outils, d'autres marins doivent également être affectés à ces postes afin de collationner et analyser les informations captées par ces senseurs. Tout cela ne représente que les grandes lignes de la navigation d'un tel bâtiment, la liste pourrait largement être allongée par d'autres services tels que l'aéronautique (selon les manœuvres, il peut être nécessaire d'ajouter cette fonction puisque le *Palau* est un porte-hélicoptères), la logistique (à la fois en termes de transport, mais aussi en termes de support de l'équipage, il faut bien que les personnels affectés à la navigation mangent, dorment, puissent prendre soin d'eux, etc.), l'artillerie, la santé, etc. Toutes ces personnes et ces matériels sont rigoureusement organisés selon des procédures très précises pour que chaque manœuvre puisse être réalisée correctement ou que l'équipage puisse répondre dans les plus brefs délais à une avarie ou à une situation exceptionnelle.

Ces procédures sont encadrées par des codes sociaux et une chaîne hiérarchique qu'impose le monde militaire. Lorsque le commandant de l'unité donne un ordre, celui-ci se répercute auprès du chef de quart en poste à la navigation, qui va ensuite traduire cet ordre par une multitude d'actions et d'ordres qu'il va déléguer aux différents services que nous venons de mentionner. Par exemple, au moment d'entrer dans un port, le chef de quart va probablement demander à la salle des machines de réduire l'allure à une vitesse appropriée, puis il va demander au barreur de modifier la route de navigation pour positionner le bâtiment dans le chenal de navigation, puis il demandera aux manœuvriers de se préparer sur les plages avant et arrière du navire pour préparer l'accostage et l'amarrage à quai. La réalisation de ces actions ne dépend pas de la réflexion d'une seule personne – le chef de quart en l'occurrence – mais de tout un équipage pour lequel chaque personne compte, car chaque action a son importance pour le bon déroulement de la procédure. Puisque le chef de quart ne peut pas tout penser par lui-même, parce que la charge est trop importante, il la distribue avec le reste de l'équipage.

Ces actions sont rendues possibles par la présence d'outils de mesure et de calcul présents à bord, les radars, les sonars, les GPS, le sextant ou la table traçante. Par calcul – ou computation –, Hutchins se réfère à la « propagation d'un état à travers les supports » (1995 : 118), c'est-à-dire à la captation d'un événement par un senseur que l'équipage va traduire par une valeur, puis un graphe, puis un ordre et enfin une action pour l'appliquer. La particularité de

cette valeur mesurée est qu'elle ne peut l'être qu'au moyen d'un outil : sans cet outil et sans écriture, la mesure ne peut pas être réalisée et la propagation de l'état correspondant ne se fera pas. Les décisions qui en découlent seraient alors certainement différentes sans cette information.

La pensée d'Hutchins peut être alors résumée par cette citation :

> Je propose une notion plus large de la cognition parce que je veux préserver un concept de cognition en tant que computation, et je veux que le type de computation qu'est la cognition soit applicable aux événements qui impliquent l'interaction des humains avec des artefacts et avec d'autres humains, ainsi qu'aux événements qui sont exclusivement internes aux personnes individuelles (Hutchins 1995 : 118).

En s'appuyant sur la théorie de la cognition distribuée proposée par Hutchins, Paveau externalise également le processus d'écriture, qui n'est alors plus considéré comme un produit logocentré. L'écriture devient ainsi un produit des agents du dispositif, qu'ils soient humains ou non humains. Selon l'analyse de Paveau, la technologie sous-jacente à ce dispositif peut être qualifiée de « discursive ». Ainsi, la technologie discursive est définie comme un « dispositif au sein duquel la production discursive est intrinsèquement liée à des outils technologiques » (Paveau 2015 : § 9). Dans son analyse, Paveau opère une transposition de la cognition distribuée à l'écriture distribuée, en se référant à la notion de « technologie intellectuelle », concept développé notamment par Goody.

Jack Goody est un anthropologue britannique ayant exercé dans la seconde moitié du XXe siècle. Ses travaux sur l'écriture ont largement marqué la recherche en Occident, que ce soit en Amérique du Nord chez Ong, chez McLuhan ou, quelques décennies plus tard avec l'école de l'intermédialité montréalaise, ou encore de l'autre côté de l'Atlantique, en Europe, avec notamment les recherches en sciences de l'information et de la communication ou les travaux très critiques d'Anne-Marie Christin sur la *déraison graphique* (Christin 2009).

Goody est notamment connu pour ses recherches sur l'écriture et ses développements de la raison graphique dans lesquels il traite justement de la production du sens. En s'appuyant sur les différences entre les modes de communication oral et écrit, ainsi que sur ses expériences de terrain au sein de différentes communautés à travers le monde, Goody analyse les effets de l'écriture sur les « modes de pensée » (Goody 1979 : 31).

Contrairement à Hutchins, Goody est influencé par Lévi-Strauss et se positionne directement dans ses pas en accolant le sous-titre *La domestication de la pensée sauvage* à son ouvrage *La raison graphique*. La thèse de Goody est que cette domestication de la pensée est un processus rendu possible par l'introduction

de nouvelles techniques de communication dans nos sociétés telles que l'écriture, l'alphabet et l'imprimerie (Goody 1979 : 107) dont l'une des conséquences est l'accumulation de textes critiques permettant l'avènement de la pensée scientifique (1979 : 101). À ce titre, Goody s'appuie sur les travaux de Havelock pour étayer sa thèse, tout comme l'a également fait Ong. Il convoque notamment la phase d'appropriation de l'écriture durant les époques classique et hellénistique, ainsi que les modifications de la cognition engendrées par cette appropriation (Goody 1979 : 86-7).

L'écriture occupe une place prépondérante dans la thèse de Goody. C'est grâce à l'écriture, ainsi qu'aux représentations graphiques que cette technique rend possibles, que l'on accède à d'autres formes de structuration des connaissances et, par extension, de cognition et de production du sens. En s'appuyant sur trois formes écrites, le tableau, la liste et la formule, Goody démontre que la structuration spatiale des connaissances influence nos modes de pensée. À tel point que ces techniques de structuration nous empêchent de penser selon d'autres modalités, comme de nous projeter dans un système de pensée purement oral.

Pour le formuler autrement, Goody prolonge la conception courante de l'écriture en tant que « redoublement matériel » du discours oral d'une conception graphique de la disposition des informations dans l'espace écrit (Goody 1979 : 144). Qu'il s'agisse du tableau, de la liste ou de la formule, ces agencements permettent d'organiser les informations et de créer des liens logiques entre elles. Ces arrangements graphiques permettent alors de hiérarchiser les informations selon une structure et un sens de lecture. Par exemple, dans une liste, les informations sont ordonnées verticalement, tandis que dans un tableau les éléments sont ordonnés à la fois verticalement, mais aussi horizontalement sous la forme de colonnes et de lignes. L'écriture n'est plus seulement une affaire de langage et devient également une affaire de disposition graphique de l'information. En disposant les informations selon des structures graphiques, les rapports entre les éléments changent. Le texte n'est plus donné à lire de façon linéaire (de gauche à droite et de haut en bas dans notre cas) car il faut suivre l'arborescence de la structure pour comprendre les liens hiérarchiques entre les éléments qui la composent. Une lecture linéaire d'une liste ou d'un tableau n'aura aucun sens si elle ne suit pas la structure des informations. Sans cela, elle ne comportera qu'une suite d'écrits sans aucun lien entre les différents éléments.

La raison graphique démontre alors que ces formes de structuration de l'information ne sont rendues possibles que parce que l'écriture le permet. À l'oral, il n'était pas envisageable de structurer les informations de cette manière graphique. La création de nouvelles relations entre les éléments de ces structures

graphiques produisent de nouvelles formes de connaissances, de nouvelles formes de cognition et produisent donc un nouveau sens.

La raison graphique a également été un appui théorique pour développer une autre forme de « cognition externe » et de « technologie intellectuelle » chez Paveau. Au tournant du numérique, les sciences de l'information et de la communication se sont emparées de cette raison graphique et l'ont réadaptée au regard de l'écriture numérique. Bachimont a ainsi proposé la notion de « raison computationnelle » en s'appuyant à la fois sur la raison graphique, mais aussi sur le principe de « technologie cognitive » dans le sens d'« une technologie symbolique permettant de manipuler des unités signifiantes » (Bachimont 2000 : 12). À l'endroit où la raison graphique permet de produire du sens par les médiations qu'offre l'écriture, comme la structuration des contenus en liste ou en tableau sur un support, la raison computationnelle exprime la possibilité de produire du sens sur la base des calculs que réalise l'informatique pour traiter l'information. En s'appuyant sur le calcul et non plus sur une raison graphique, l'écriture numérique permet d'autres agencements des informations et des connaissances de ceux produits par une écriture manuscrite, et donne ainsi lieu à de nouvelles formes de cognition et de sens[2].

Cette brève introduction à la raison computationnelle met en évidence le fait que l'écriture ne mobilise pas qu'une seule forme de cognition externe que l'on pourrait généraliser à toutes les formes d'écriture. L'écriture n'est donc pas une technique que l'on pourrait réduire à un seul et unique mode de pensée, ce que démontrent les différences entre la raison graphique et la raison computationnelle qui n'impliquent pas la même organisation des informations, ni la même production du sens.

Au-delà d'une modalité d'écriture générique – écriture manuscrite, écriture numérique, etc – il est nécessaire de connaître le rôle des différents agents mobilisés dans une forme d'écriture pour comprendre leur implication dans la production du sens. C'est ainsi que, dans les travaux de Paveau, cette conception de l'écriture distribuée dans des « technologies discursives » implique une « approche écologique » de l'écriture, soit la nécessité de prendre en considération la totalité de l'environnement duquel elle émerge, ainsi que le sens qu'elle produit, puisqu'« il n'y a pas de rupture entre ce qui est humain et ce qui ne l'est pas, entre l'esprit et la matière, le linguistique et l'extralinguistique, le discours et le contexte » (Paveau 2013 : 3).

2. Nous reviendrons plus en détail sur l'écriture numérique dans le chapitre II. À ce stade, la partie qui nous intéresse concerne les techniques d'écriture et la production du sens ainsi que le fait qu'il y ait une forme de cognition distribuée dans la technique.

L'écriture ne doit pas être réduite à la capacité d'inscrire et d'organiser des signes dans un espace, mais doit être considérée dans tous ces paramètres, c'est-à-dire tout le dispositif : l'instrument pour tracer, la composition de la matière employée pour l'inscription, les protocoles employés, le cadre social et le cadre culturel, etc.

Lorsque Ong et Paveau qualifient l'écriture de non-humaine ou d'inhumaine, ils nous renvoient à cette cognition distribuée, située dans des « interactions sociales, culturelles, techniques et technologiques » (Vitali-Rosati 2020*b* : § 20) et non dans un *logos*, à partir de laquelle l'écriture n'est plus dissociable de son environnement.

Afin d'illustrer au mieux cette conception de l'écriture distribuée, nous avons choisi d'étudier la pratique de l'écriture de soi que réalisent les stoïciens durant l'époque hellénistique et l'Empire romain. Cette lecture des écrits stoïciens, notamment ceux de Sénèque et de Marc Aurèle, repose principalement sur les travaux de Pierre Hadot, spécialiste de la philosophie antique et des exercices spirituels qu'il y associe.

Si nous faisons un détour par le stoïcisme, c'est parce qu'il permet d'apercevoir les changements qu'apporte l'écriture dans une société basée principalement sur la transmission orale des informations (l'écriture étant alors essentiellement utilisée par la magistrature), notamment en nous intéressant plus précisément à la pratique de l'écriture de soi.

Hadot établit un parallèle entre cette pratique de l'écriture de soi et l'examen de conscience, un exercice spirituel nommé *prosochè* (Hadot 2002 : 89). Cette démarche implique la rédaction systématique des actions accomplies durant une période définie, telle qu'une journée, et ce, où le philosophe se raconte lui-même. C'est ce que font, par exemple, Sénèque dans les *Lettres à Lucilius* ou Marc Aurèle dans les *Pensées pour moi-même*[3].

En réalisant un léger anachronisme, cette pratique de l'écriture de soi pourrait aisément être confondue avec une écriture diaristique ou se rapprocher d'un récit autobiographique. Cette confusion est souvent faite notamment pour des ouvrages tels que les *Confessions* de Saint Augustin, ou *Les Confessions* de Rousseau, ou encore les *Méditations* de Descartes. Elles peuvent effectivement être lues comme un récit autobiographique ou alors comme la réalisation d'une *askesis*, un exercice où l'auteur utilise l'écriture pour exercer une tension entre un récit de lui-même et des réflexions philosophiques. Le succès de cette méthode qu'est l'écriture perdure pendant plusieurs siècles tandis que l'époque hellénistique est révolue,

3. Le texte de Marc Aurèle revient à plusieurs occasions dans ce chapitre. Pour éviter d'avoir à répéter systématiquement le titre à chaque occurrence, nous le désignons par la version abrégée suivante : *Pensées*.

comme en témoignent les exemples précédents ainsi que les écrits d'Athanase d'Alexandrie dans la *Vie d'Antoine* vers l'an 360 (soit environ 40 ans avant les *Confessions* d'Augustin).

Ainsi, l'examen de conscience, dont la finalité est la maîtrise de soi, peut être réalisé par une série d'étapes dont la première est l'introspection. Cette introspection est accomplie grâce à une mise en récit de soi via un *medium*, l'écriture, et génère alors une deuxième étape, celle de l'extériorisation de soi. Dans le cadre de cet exercice, l'écriture dépasse la simple condition de technique grâce à laquelle une information peut être transmise et devient la condition *sine qua non* de l'accès à l'autre.

À titre d'exemple, dans la lettre 83[4] de sa correspondance avec Lucilius[5], Sénèque écrit ceci : « Sans doute l'homme devrait toujours se conduire comme s'il avait des témoins, toujours penser comme si quelqu'un pouvait lire au fond de son cœur ». Dans cette lettre, antérieure de trois siècles au texte d'Athanase d'Alexandrie, Sénèque soumet à Lucilius une méthode très similaire que celle racontée dans le récit de Saint Antoine de Padoue. Cette méthode s'incarne à travers la lettre, alors employée à la fois comme medium de cette correspondance épistolaire et de cette leçon philosophique.

Sénèque nous indique dès le début de la lettre qu'il s'agit de l'exercice de l'examen de soi : « Je vais donc me mettre à m'observer, et, pour plus de sûreté, je ferai le soir la revue de ma journée ». Si nous considérons qu'il s'agit bien là de la réalisation d'un exercice spirituel, nous pouvons en déduire que cette lettre comporte un double enjeu. Le premier est explicite : Sénèque fait une démonstration à Lucilius comme un maître peut le faire avec son élève. Le second est la réalisation de l'exercice pour Sénèque lui-même. En réalisant cet exercice dans le cadre d'une leçon qu'il dispense, Sénèque en profite pour appliquer cette méthode et écrire son examen de conscience qu'il va pouvoir livrer à Lucilius qui, en l'occurrence, incarne l'autre. L'emploi de la conjugaison au futur dans la lettre donne à penser que Sénèque prémédite les actions et mouvements qu'il va réaliser dans la journée. Nous pouvons supposer qu'il fait en sorte que ses actions soient vertueuses afin qu'il n'y ait rien dont il puisse avoir honte, car il sait que Lucilius sera témoin de son récit.

Néanmoins, il ne s'agit pas uniquement de se livrer à autrui au sens d'un destinataire, comme cela peut paraître être le cas dans cette correspondance.

4. La lettre 83 est disponible sur cette page web : `https://agoraclass.fltr.ucl.ac.be/concordances/sen_luciliusX/lecture/3.htm`, visitée le 29 octobre 2024.

5. La correspondance entre les deux protagonistes s'étale sur deux années, entre l'an 62 et l'an 64, et l'on décompte pas moins de 124 lettres rédigées par Sénèque à l'intention de Lucilius. Elles ont été regroupées sous la forme d'un recueil intitulé *Lettres à Lucilius*. Ce recueil est connu pour aborder une large variété de sujets ayant pour objectif de former Lucilius au stoïcisme.

D'ailleurs ce n'est pas le regard que l'autre peut porter sur soi qui importe en matière de stoïcisme. Qu'il s'agisse de Sénèque, ou plus tard d'Antoine, leurs méthodes convoquent un autre qui est soit « public », soit « témoin ». L'autre ainsi convoqué dans ce mouvement d'extériorisation est avant tout un autre social et politique que l'écrivain projette sur son texte lorsqu'il écrit. Finalement, le stoïcien se doit d'être irréprochable et sa conduite doit correspondre à l'image attendue d'un philosophe dans l'école, mais aussi et surtout dans la cité (grecque). Il ne dépend pas du regard que peuvent porter les citoyens sur lui, mais plutôt d'un système de valeurs qui le détermine en tant que philosophe. La question de la maîtrise de soi et de l'examen de conscience est donc fondamentalement éthique. En ce sens, en appliquant cette méthode, l'écriture produit Sénèque en tant que philosophe et non l'inverse.

Un exemple différent de cette pratique de l'écriture de soi sont les *Pensées* de Marc Aurèle. Ces *Pensées* revêtent un caractère un peu particulier, étant donné qu'il ne s'agit pas d'un texte conçu comme un ouvrage, ou un livre, mais plutôt comme une série de notes désorganisées et désordonnées. Cette autre pratique relève de la fonction primaire de l'écriture relative à l'oubli et qui consiste à décharger la mémoire de l'écrivain, comme nous le rappelle Platon (*Phèdre* 275*a*).

Ce format de notes porte un nom spécifique : ces notes sont appelées *hypomnémata*. Hadot, en citant Aulu-Gelle, nous rappelle qu'il s'agit là du sens étymologique d'*hypomnéma*, d'être justement un aide-mémoire (Hadot 1992 : 47). Initialement, ces notes ne sont pas destinées au public, elles sont plutôt gardées pour soi. Toutefois, elles peuvent selon les circonstances faire l'objet d'une publication, comme ce fut le cas pour les *Pensées* de Marc Aurèle, dont la publication de la première édition est réalisée plusieurs après son décès (Hadot 1992 : 36).

Les Pensées laissées par Marc Aurèle sont aujourd'hui considérées comme une œuvre majeure de la philosophie antique, dont Hadot était un spécialiste. Son ouvrage *La Citadelle intérieure : introduction aux Pensées de Marc Aurèle* est révélateur quant à la place de l'écriture dans la vie philosophique de l'empereur romain. Selon Hadot, les *Pensées* sont des *hypomnémata* particulières par rapport à celles des autres philosophes. En plus d'être des notes qu'il rédige pour lui-même, Marc Aurèle les emploie à des fins de réalisation d'un exercice spirituel, comme le fait Sénèque lors de sa correspondance avec Lucilius. Les *Pensées* sont donc la trace de cette pratique de l'écriture quotidienne, sous forme de notes où l'auteur s'y raconte. Marc Aurèle les écrit au jour le jour sans qu'il n'y ait, *a priori*, de suite logique entre ces notes, car elles n'ont pas pour objectif d'être publiées par la suite. D'ailleurs, elles seront recomposées ensemble plus tard, lors de leur publication.

Un fait intéressant est que ces notes ont été soigneusement rédigées par l'empereur lui-même selon la méthode stoïcienne. Par son statut, Marc Aurèle avait accès à tout un ensemble de services dont des scribes pour occuper le rôle de secrétaire. Pourtant, Hadot nous confirme que les *Pensées* ont été rédigées de la main de l'auteur et en grec qui plus est, tandis que durant le IIe siècle, la langue de l'Empire romain était le latin (Hadot 1992 : 67).

Cette information nous permet de mettre à jour la deuxième fonctionnalité de l'écriture que l'on doit retenir de cette pratique de l'écriture philosophique. Marc Aurèle écrit pour se réapproprier les dogmes stoïciens en suivant les consignes laissées par Épictète. Marc Aurèle le dit lui-même : « Les principes vivent. Comment pourraient-ils mourir, à moins que les idées qui leur correspondent ne s'éteignent ? Or celles-ci, il dépend de toi de les raviver sans cesse » (*Pensées* VII, 2).

Les consignes laissées par Épictète sont très similaires à cette note de Marc Aurèle. Hadot en cite deux (1992 : 65) : « Voilà ce que doivent méditer les philosophes, voilà ce qu'ils doivent *écrire* tous les jours, qui doit être leur matière d'exercice (I, 1, 25) ». « Ces principes, il faut que tu les aies sous la main (*procheira*), la nuit et le jour, il faut les *écrire*, il faut les lire (III, 24, 103) ».

Grâce à ce que nous disent Marc Aurèle et Épictète, on devine que l'écriture est nécessaire à la pratique de la philosophie en ce qu'elle permet de réactiver quelque chose chez le philosophe que la lecture ne permet pas car « les pages écrites sont déjà mortes » (Hadot 1992 : 66). Écrire permet de se rappeler les dogmes stoïciens et aussi de se les approprier par la reformulation constante. La répétition par la réécriture permet à Marc Aurèle de formuler différemment ses *Pensées* jusqu'à ce qu'il les ait assimilées et qu'elles soient siennes, jusqu'à ce qu'il vive en tant que philosophe stoïcien, c'est-à-dire que ses actes soient en accord avec les dogmes qu'il répète sans cesse – c'est-à-dire jusqu'à sa mort puisque l'état de philosophe relève du divin, il est inatteignable.

C'est d'ailleurs pour cette raison que Marc Aurèle rédige lui-même ses *Pensées* alors qu'il pourrait les faire écrire par un scribe. C'est également pour cette raison qu'il écrit en grec et non en latin : les concepts majeurs du stoïcisme ont été fondés durant l'époque hellénistique, en grec donc, et n'ont pas été traduits en latin. Marc Aurèle manipule le grec pour exprimer et réactiver quelque chose en lui que le latin ne peut réaliser.

> En écrivant ses *Pensées*, Marc Aurèle pratique donc des exercices spirituels stoïciens, c'est-à-dire qu'il utilise une technique, un procédé, l'écriture, pour s'influencer lui-même, pour transformer son discours intérieur par la méditation des dogmes et des règles de vie du stoïcisme. Exercice d'écriture au jour le jour, toujours renouvelé,

> toujours repris, toujours à reprendre, puisque le vrai philosophe est celui qui a conscience de ne pas encore avoir atteint la sagesse (Hadot 1992 : 67).

Les exercices spirituels, que ce soit à travers Sénèque ou Marc Aurèle, nous montrent que l'écriture est une technique qui affecte la pensée. L'écriture a alors une double fonction, la première est un aide-mémoire tandis que la seconde déplace la stricte répétition identique d'un dogme (d'une idée) pour passer à l'action de reformulation. Cette deuxième fonctionnalité découle de la première, même si ce n'est pas un fait évident de prime abord. Si la reformulation est rendue possible, c'est bien parce que l'écriture décharge l'auteur du devoir de mémoire, puisque celle-ci est dorénavant inscrite sur un support. En libérant la pensée de cette charge, l'écriture permet à Marc Aurèle de formuler ses *Pensées* d'une nouvelle façon.

Ces deux fonctions de l'écriture ne sont pas dissociables. La seconde n'est envisageable que parce que la première libère justement la mémoire, comme nous le démontrent les spécialistes de cette transition chez les Grecs entre oralité et scripturalité.

Dans son ouvrage *Preface to Plato*, Havelock nous informe que, du temps où il n'y avait pas de livres, la poésie était le moyen par lequel les informations étaient transmises (Havelock 1967 : 44). Les poètes s'exerçaient à répéter les mêmes formules clichées d'un poème à l'autre pour pouvoir transmettre de génération en génération les mémoires et les sagesses de leurs prédécesseurs, Homère étant le poète par excellence de cette forme de transmission. La poésie orale a alors une fonction de mémoire et de fait délimite la formulation de la pensée (et de la création de poème) pour éviter que les savoirs ne se perdent (Ong 2014 : 44).

Tandis qu'à partir de l'époque de Platon, les Grecs intériorisent l'écriture et se déchargent de la fonction mémorielle : les savoirs et les connaissances sont préservées dans les supports écrits. Ainsi, « le nouveau moyen de stocker du savoir ne consistait plus en formules mnémoniques mais en texte écrit, ce qui, en libérant l'esprit, le disposait à d'autres réflexions plus originales, plus abstraites » (Ong 2014 : 44).

C'est précisément parce que l'écriture permet de décharger l'esprit de la fonction mémorielle, qu'elle permet à Marc Aurèle de se réapproprier les dogmes du stoïcisme non pas par la répétition, mais par la reformulation. En la sollicitant, les philosophes affectent leur façon de penser et surtout leur façon de vivre.

Cette pratique de l'écriture de soi nous montre qu'il y a dans l'écriture quelque chose qui permet de produire un sens différent d'un autre mode d'expression, oral dans notre exemple. Lorsque Marc Aurèle écrit, il le fait sur un matériau donné, certainement des tablettes (*pugillares*) ou des feuillets (*schedae*) sur

lesquelles l'écriture de notes peut être rapide, mais l'espace est limité et la perennité incertaine (Hadot 1992 : 48). Ces deux supports ont la particularité d'être relativement petits. Les *schedae* sont des feuillets volants, non rattachés entre eux, à l'image des fiches contemporaines (Ronconi 2022) tandis que les *pugillares* sont des petites tablettes composées de cire et qui peuvent tenir dans la main (d'où leur nom).

De plus, Marc Aurèle emploie également le grec et non le latin pour inscrire des concepts qui n'existent que dans cette langue, alors qu'il pourrait aussi bien proposer des traductions de son cru. Plus que le choix de la langue, Marc Aurèle fait le choix d'employer les notions qu'il a apprises dans un contexte social précis : celui de l'école stoïcienne. En respectant ainsi les traditions et les dogmes de l'école stoïcienne, il revendique une appartenance identitaire à cette communauté, identité qui d'ailleurs persistera dans le temps puisque, lorsque nous évoquons Marc Aurèle aujourd'hui, nous convoquons à la fois l'image de l'empereur et celle du philosophe stoïcien. Tous ces choix que réalise Marc Aurèle définissent les contraintes de l'écriture et la manière dont il va pouvoir écrire. Ce sont ces contraintes qui, en un sens, déterminent l'écriture et le sens qu'elle produit.

En conséquence, au même titre que la main libère la parole chez Leroi-Gourhan, l'ajout d'une nouvelle technique, l'écriture, décharge Marc Aurèle d'une fonction mémorielle, que la technique occupe par la suite, au profit d'une nouvelle, dans ce cas-ci la possibilité laissée à Marc Aurèle de reformuler les préceptes de l'école stoïcienne pour les faire siens.

La possibilité de cette nouvelle fonction n'est pas juste rendue possible parce que l'écriture décharge Marc Aurèle d'un devoir de mémoire et de transmission de ces connaissances. En se référant aux divers travaux sur la distribution de la cognition, nous avons remarqué que l'écriture n'est pas un phénomène neutre ne servant qu'à déposer la pensée sur un support. L'écriture est une technique avec sa propre agentivité, qu'induisent les techniques et technologies employées pour la produire. Pour reprendre les mots de Bachimont, l'écriture produit du sens non pas parce « qu'[elle] possède une quelconque pertinence cognitive, mais parce qu'[elle] constitue une médiation technique » (Bachimont 2000 : 12).

Les reformulations de Marc Aurèle sont alors non seulement réalisables parce que l'écriture décharge Marc Aurèle de son devoir de mémoire de la parole, mais surtout parce que l'écriture est une médiation technique offrant la possibilité d'opérer de nouveaux agencements des connaissances et donc de production du sens. Les reformulations de Marc Aurèle ne sont pas également faites au hasard ou selon son humeur. Elles sont élaborées en suivant les règles édictées par le stoïcisme, c'est-à-dire selon un ensemble de règles établies pour développer une pensée stoïcienne et en respecter les dogmes (une culture).

Cet ensemble de règles que suit Marc Aurèle pour écrire, nous donne l'occasion d'introduire la notion de protocole que l'on adosse au dispositif d'écriture. En effet, Marc Aurèle, en suivant ces règles, suit des protocoles très particuliers pour écrire.

Les protocoles foisonnent dans nos quotidiens. Nous les suivons pas à pas pour un certain nombre de nos activités, volontairement ou involontairement. Par exemple, Servanne Monjour et Nicolas Sauret décrivent l'usage du protocole Git dans le contexte de l'écriture littéraire (Monjour et Sauret 2021).

Git est un système de gestion de versions distribué conçu pour faciliter l'écriture et la modification collaborative de fichiers numériques. Il permet de suivre précisément les changements apportés aux fichiers au fil du temps, en conservant un historique complet des différentes versions. Chaque contributeur travaille sur sa propre copie locale du projet, où il peut effectuer des ajouts, des modifications ou des suppressions. Ces évolutions sont enregistrées sous forme de *commits* qui documentent les changements effectués. Git organise ces *commits* de manière chronologique et hiérarchique, ce qui permet de reconstituer facilement l'historique complet du projet. Lors du travail collaboratif, Git gère la synchronisation des modifications entre les contributeurs, facilite la création de branches pour expérimenter sans impacter la version principale, et propose des outils pour résoudre les éventuels conflits lorsque plusieurs personnes modifient les mêmes parties d'un fichier.

En tant qu'ensemble de règles permettant la création et l'organisation des versions de fichiers, Monjour et Sauret considèrent Git comme un protocole, car c'est cet ensemble de règles qui régit l'ensemble de l'organisation collective de l'écriture. La « gittérature », ce néologisme forgée à partir de la fusion des mots « git » et « littérature », traduit l'intégration de Git au dispositif d'écriture littéraire. À travers l'exemple de la Zone Autonome à Poétiser (ZAP) Rimbaud, les auteurs illustrent leur propos avec un dispositif reposant intégralement sur ce protocole pour produire les documents donnés à lire. Ce projet propose « d'écrire progressivement par-dessus le poème original de Rimbaud un texte collectif, de manière à faire émerger un nouvel objet littéraire, "cyberpoétique" » (Monjour et Sauret 2021 : 7). Dans ce dispositif, Git est utilisé à travers une interface, l'application Gitlab. Grâce à cette dernière, les auteurs qui participent à la ZAP peuvent accéder aux contenus et proposer des modifications.

> Techniquement parlant, les participants doivent « cloner » le dépôt sur leur machine, réécrire des parties de textes sur une « branche » personnelle, proposée ensuite à la publication via une « proposition de révision » (*merge request*) qui pourra être intégrée à la branche maîtresse maintenue par les éditeurs. Sur le serveur, le registre

> accumule les versions qui, en fusionnant entre elles, matérialisent en direct le hack progressif du texte de Rimbaud. Par ailleurs, le dépôt héberge une série de fichiers HTML, CSS et JS qui éditorialisent à la volée le poème collectif en un site web constamment mis-à-jour selon le principe du déploiement continu de la plateforme Gitlab. La création éditoriale des éditeurs a consisté à distinguer sur la page web de ce palimpseste les parties originales et les parties réécrites par deux couleurs distinctes (Monjour et Sauret 2021 : 8).

Dans cet exemple, c'est bien le protocole Git qui permet aux différents auteurs et éditeurs de s'organiser entre eux et surtout d'organiser l'écriture et de produire les documents de la ZAP Rimbaud.

En suivant ce premier exemple, nous pouvons considérer qu'un protocole regroupe donc un ensemble de méthodes que l'on emploie dans un environnement donné pour que chaque acteur présent dans cet environnement soit à même d'interpréter l'autre. De la même façon, tout acte de communication nécessite un protocole pour que l'autre à qui le message est adressé puisse comprendre la signification de ce message. « Sans protocole partagé, il n'y a pas de réseau », écrivait Galloway (2004 : 12).

En s'appuyant sur les travaux de Deleuze et Foucault, Galloway démontre que les protocoles sont nécessaires dans les environnements de communication distribuée (en opposition aux environnements centralisés) afin que chaque nœud du réseau soit à même de lire et d'écrire le message sans avoir à passer par un nœud central. Galloway applique cette définition non seulement à Internet, avec les protocoles *Transmission Control Protocol* (TCP) et *Internet Protocol* (IP), mais aussi au Web et aux protocoles mis en place par Tim Berners-Lee pour le Web : *HyperText Transfert Protocol* (HTTP), HTML et *Uniform Resource Identifier* (URI) (Galloway 2004 : 137). Ces trois protocoles sont au centre du Web et c'est bien à travers eux que le Web peut exister.

> Ce qui était souvent difficile à comprendre, c'est qu'il n'y avait rien d'autre que les URI, HTTP et HTML. Il n'y avait pas d'ordinateur central « contrôlant » le Web, pas de réseau unique sur lequel ces protocoles fonctionnaient, pas même d'organisation quelque part qui « gérait » le web. Le Web n'était pas une « chose » physique qui existait dans un certain « lieu ». C'était un « espace » dans lequel l'information pouvait exister (Tim Berners-Lee (1999 : 36) cité dans (Galloway 2004 : 138)).

Ces trois protocoles que mentionnent Galloway et Berners-Lee permettent de définir la structuration du texte (HTML), les adresses des contenus (URI), et

les méthodes de circulation entre un client et un serveur (HTTP). Avec ces trois éléments, c'est toute une organisation des connaissances qui se met en place à travers le Web.

Toutefois, un protocole seul, isolé de tout environnement dans lequel il est implémenté, n'est pas opérant. Si le Web n'est pas représenté par une adresse géolocalisable, il n'est pas pour autant dématérialisé en dehors de toute infrastructure. Pour que le Web existe, les protocoles s'appuient sur une infrastructure matérielle, des serveurs et des clients, des câbles sous-marins, mais aussi sur toute une pile d'autres protocoles qui lui sont extérieurs, comme c'est le cas des protocoles TCP et IP d'Internet. L'application d'un protocole est en ce sens directement liée aux caractéristiques du *medium* employé, comme en témoigne les évolutions du Web que l'on connaît depuis ses débuts.

En d'autres termes, il existe une différence entre le protocole en tant que tel, c'est-à-dire en tant qu'ensemble de règles de communication, et son application effective par les entités physiques présentes dans l'environnement où ce protocole est appliqué. Le protocole intervient alors dans le dispositif d'écriture comme un ensemble de règles permettant l'agencement ou « l'organisation spatiale des éléments » (Bachimont 2010*b* : 22) à l'intérieur de ce dernier. Ainsi, le passage du protocole en tant qu'abstraction à sa mise en œuvre concrète permet de comprendre comment les modalités d'écriture se déploient dans des contextes précis.

Finalement, la distribution de l'écriture à travers les interactions sociales, culturelles, techniques et technologiques d'un environnement est réalisée par l'application d'un ou plusieurs protocoles que chaque acteur est en capacité d'interpréter en toute autonomie. En reprenant le cas de Platon et du *Phèdre*, nous avons vu qu'il y écrit une critique de l'écriture. Ce que la démarche de Marc Aurèle nous a appris s'applique également à l'écriture du *Phèdre*. Si l'on suit notre raisonnement, Platon n'a pu émettre toute cette critique de l'écriture uniquement parce qu'il était en capacité de l'écrire. C'est l'écriture elle-même qui permet à Platon d'en faire la critique. Sans elle, il n'aurait pu élaborer ce raisonnement critique, nous disent Ong et Goody (Ong 2014 : 48 ; Goody 1979 : 87). Cette situation apparaît ironique puisque la critique de cette technique ne peut être faîte qu'en l'employant. La technique, en tant que *medium*, apporterait donc quelque chose d'irréductible à la pensée et à la production du sens.

Toutefois, la technique n'est pas à considérer en tant qu'abstraction universelle, puisque selon les technologies employées dans la réalisation de cette technique, les modalités de l'environnement se voient modifiées et transforment par la même occasion les entités entre lesquelles l'écriture se retrouve distribuée. C'est ce que nous avons vu avec Hutchins, ou encore la distinction qu'il peut y

avoir entre une raison graphique dans un environnement manuscrit et une raison computationnelle dans un environnement informatique.

Cette écriture distribuée, à travers des technologies discursives, serait liée a des médias qui, par leurs propriétés, déterminent 1) les caractéristiques de la distribution de la cognition, 2) les modalités de propagation de l'information entre les supports dans les différentes étapes de la cognition et 3) les modalités de la production du sens.

I.3 Le support n'existe pas

Dans cette dernière section du chapitre, nous traitons de la place du support physique sur lequel l'écriture est effectuée. Jusqu'à ce stade, nous avons abordé l'écriture en tant que technique détenant une part active de la cognition à l'œuvre dans la production du sens. Pour ce faire, nous avons laissé de côté un élément majeur de notre démonstration, pourtant abordé en filigrane tout au long de ce chapitre : le support sur lequel l'information est inscrite. Ce support a été évoqué avec les écrits de Marc Aurèle sur les *pugillares* ou encore avec la navigation lorsqu'un marin inscrit la position du navire sur la table traçante. Toutes ces formes d'écriture nécessitent un objet tangible sur lequel l'écriture va être inscrite. Quel que soit l'exemple, l'écriture apparaît toujours matérielle et non pas diffuse dans quelque chose de non tangible. La matérialité de l'écriture est la propriété que nous pouvons retenir de la distribution de l'écriture dans tout l'environnement qui la produit. Sans ces éléments physiques, il n'y a pas d'écriture, et la pensée ne peut pas en être décorrélée. Ce postulat n'est pas sans rappeler la théorie du support proposée par Bachimont, pour qui « les propriétés matérielles du support d'inscription conditionnent l'intelligibilité de l'inscription » (Bachimont 2010*b* : 121).

Toutefois, nous nous détachons de cette appellation de théorie du support du fait de notre perception différente de la notion de « support », puisque pour nous, comme le titre de cette section l'indique, le support n'existe pas.

Avant d'aller plus avant dans cette section, il nous est nécessaire de préciser ce que nous entendons par matérialité. La matérialité n'est pas uniquement comprise dans le sens d'un objet physique avec des propriétés physiques. Le papier sur lequel écrire une lettre pourrait être une feuille au format A4 (210*297mm) avec un grammage de 80g/m² (soit 5g pour une seule feuille), que l'on plie ensuite en 3 pour la faire entrer dans une enveloppe. Ce que nous venons de décrire sont des propriétés physiques du support de l'écriture. Nous avons là les propriétés du support avant que l'on commence à écrire. La matérialité est différente de ces propriétés, elle concerne l'objet final, une fois que la lettre est

rédigée et que le texte y est inscrit. Pour bien distinguer la dimension physique de la dimension matérielle, Hayles propose de considérer la physicalité en tant qu'ensemble des propriétés physiques d'un artefact tandis que la matérialité est une propriété émergente de l'écriture (Hayles 2005 : 3). La matérialité serait donc une production de l'écriture, au croisement des propriétés physiques d'un objet et de leur part cognitive dans la distribution de l'écriture. Pour Hayles, la matérialité ne produit pas du sens, elle est le sens de l'écriture, elle est cette « matière qui compte pour la pensée humaine »[6] (Hayles 2005 : 3). En nous appuyant sur l'écriture distribuée, nous pouvons défendre l'idée selon laquelle le sens donné à notre lettre précédente est contingenté aux caractéristiques du papier sur lequel elle est écrite, puis aux protocoles de transmission. Ces éléments participent à l'émergence de la matérialité de l'écriture, en tant qu'ils participent à la production du sens. Dorénavant, lorsque nous nous référerons aux propriétés physiques d'un artefact, nous emploierons le terme de physicalité, et lorsque nous nous référerons au sens produit par cette écriture distribuée, nous emploierons le terme de matérialité.

Cette distinction entre physicalité et matérialité nous amène à considérer la notion de support. Le support est l'élément qui se retrouve à l'interface entre cette physicalité et cette matérialité. Au moment de l'écriture, c'est-à-dire au moment où un objet est en passe de devenir un document, une preuve d'un fait (Briet 1951), qu'il s'agisse d'une stèle, d'un post-it, d'une feuille A4, d'un fichier au format TXT, un objet, un support accepte une inscription et se transforme en un autre objet : un document. En jouant ce rôle de médiateur entre deux états, le support devient un *medium*.

L'emploie du terme *medium* à cet endroit n'est pas un fait anodin. Nous le mobilisons en nous positionnant depuis la théorie des médias telle qu'elle a été formulée depuis McLuhan avec l'école de Toronto dans les années 1960 (dans laquelle participe Ong) puis à travers différents courants entre l'Amérique du Nord et l'Europe (principalement en France et en Allemagne).

Ce terme, *medium*, n'est pas aisé à circonscrire, car il est déjà ambivalent depuis son étymologie latine. En latin, le *medium* signifie le milieu ou le centre[7]. Krämer nous rappelle que ce milieu que l'étymologie désigne peut être envisagé sous trois perspectives. La première est spatiale, le milieu occupe une position intermédiaire au centre de l'espace. La deuxième est fonctionnelle : elle renvoie à la fonction de médiation du milieu, qui sert d'intermédiaire entre deux entités. Enfin, la troisième est d'ordre formel et renvoie à la neutralisation et à l'état

6. Cette assertion sera reprise sous la forme de *matter matter* par Barad, que l'on peut traduire par « c'est la matière qui compte » (Barad 2007).

7. Voir le dictionnaire français-latin Gaffiot : `https://gaffiot.fr/#medium`, page visitée le 20 novembre 2024.

neutre que peut représenter le milieu. Dans cette troisième signification, le milieu devient un point d'équilibre entre des forces opposées (Krämer 2016 : 209). Le *medium* est donc ce qui se trouve au milieu, ou plutôt entre des entités ou des états. Au prisme de son étymologie, le terme *medium* n'est pas une notion réductible à un catalogue d'objets, malgré le fait que nous nous en servions couramment comme un mot-valise pour désigner tour à tour des moyens de communication classiques, comme l'écriture ou la parole, des moyens de communication de masse, comme la radio ou la télévision, des artefacts et des objets, comme le papier, la pierre ou encore la silice, ou encore des systèmes techniques tels que l'ordinateur ou la machine à écrire, des techniques, comme l'imprimerie, etc. Cette liste pourrait continuer d'être allongée sans jamais être exhaustive, car elle serait toujours obsolète. En effet, selon la théorie de la communication de Shannon, le *medium* concerne tout ce qui se trouve entre un émetteur et un récepteur (Shannon 1948). À titre d'exemple, dans son introduction à la *Théorie des médias*, Mersch comptabilise pas moins de 26 médias différents mentionnés dans l'ouvrage *Pour comprendre les médias* de McLuhan (Mersch 2018 : 9), tous différents les uns des autres et couvrant un large spectre de choses, allant de l'électricité à l'artillerie militaire. Puisque c'est un objet instable, la question de ce qu'est un *medium* n'est alors pas opérationnelle, précise Friesen en introduction d'un ouvrage sur les développements de la théorie des médias entre l'Amérique du Nord et l'Europe (selon une orientation germanophone) (Friesen 2018 : 3), car elle nous pousse à l'essentialiser et biaise notre compréhension d'un phénomène dans lequel il intervient. À défaut de le définir, nous pouvons le qualifier en tant qu'« ensemble des processus relatifs à l'échange et à la transmission ainsi que les dispositifs techniques et ceux que l'on peut caractériser comme organe d'accumulation de données » (Mersch 2018 : 12). L'intérêt que nous pouvons trouver à cette définition proposée par Mersch est la mise en avant des propriétés médiales – au sens de médiations – du *medium*. Mersch, ainsi que Friesen et de bien d'autres théoriciens des médias, étudient plutôt les médiations puisque ce sont elles qui, finalement, construisent les relations.

> Que signifie au juste : médiation ? Là où nous sommes confrontés à une extériorité – un dehors ou un « autre » – et afin de pouvoir la percevoir, la comprendre et lui conférer du sens, il nous faut un milieu au double sens que peut prendre ce terme, un moyen ou un intermédiaire. Aussi, la médiation peut prendre les formes les plus variées : une atmosphère, un discours, un procédé technique, mais aussi des espaces et des temps ou encore un agencement des choses, un ensemble d'architectures ou de programmes, pour ne donner que quelques exemples. L'important, c'est de glisser entre cette altérité et

> le lieu de l'humain ce minuscule opérateur, pourtant si fondamental (Mersch 2018 : 13).

Tandis que Mersch, dans cette citation, nous donne une définition de la médiation qui englobe les deux premières étymologies – spatiale et fonctionnelle – du terme *medium*, il faut se tourner vers Krämer pour y adjoindre la troisième. Krämer développe une approche du *medium* sur le modèle du messager mourant. Krämer convoque la métaphore du messager qui porte les messages et les informations jusqu'à son destinataire avant de disparaître, tel le coureur de marathon qui, en armure, meurt lors de la délivrance du message de victoire des athéniens sur les perses (Krämer 2016 : 210). Selon cette perspective, la figure du messager mourant illustre la disparition du *medium* derrière le message qu'il transporte et se neutralise lui-même pour mettre le message en avant. À la fonction d'espace intermédiaire que joue le *medium*, il faut alors lui ajouter une caractéristique discrète : ce n'est pas un élément directement visible. Si nous employons tour à tour les termes *medium* et médiation, presque comme des synonymes, c'est parce que ce ne sont pas des éléments séparables l'un de l'autre. La définition de la médiation que nous venons de donner englobe ainsi les trois étymologies du mot *medium*. Selon ce prisme, nous considérons que puisque les médiations sont définies par les caractéristiques du *medium*, ce sont elles qui déterminent ce dernier. Pour cette raison, l'étude des médiations est une forme d'étude des médias.

L'étude des médiations est incontournable depuis les années 1960, notamment depuis la publication des travaux de McLuhan et de sa célèbre assertion « le message, c'est le medium ». La signification de cette phrase ne donne pas simplement son importance au medium en dépit du message, mais considère que le contenu d'un *medium* est toujours un autre *medium* (McLuhan 1977 : 22-3). Ainsi, « le contenu de l'écriture, c'est la parole, tout comme le mot écrit est le contenu de l'imprimé et l'imprimé, celui du télégraphe » (McLuhan 1977 : 38). En prenant l'exemple de la lumière électrique, McLuhan montre qu'il s'agit là d'un *medium* pur, c'est-à-dire un contenant auquel personne n'a adjoint de message. Pourtant, ce *medium* permet de réaliser des actions – une opération chirurgicale par exemple, ou d'écrire alors qu'il fait nuit – alors que sans cette lumière électrique, ces actions ne seraient pas réalisables. Cette phrase a donc un double niveau d'interprétation. Premièrement, elle démontre la thèse de McLuhan selon laquelle le *medium* et les médiations qu'il génère produisent le sens de l'événement au centre duquel ils se trouvent. Un autre exemple que celui de la lumière électrique serait celui de la voie ferrée ou de la route. En tant que support du transport ferroviaire ou routier, ces médias modifient notre conception de l'espace, par exemple la ville, et des déplacements à l'intérieur de celui-ci : les

architectures évoluent en fonction de cette technique, tandis que dans cette interprétation de l'architecture des villes modernes, nous ne nous préoccupons pas des trains, des voitures, ni même de leurs contenus. Ce n'est pas tant le contenu qu'il pourrait y avoir qui intéresse McLuhan, mais les modifications qu'engendrent le *medium* dans le milieu dans lequel il se trouve : il y aurait, pour l'auteur, une co-construction entre le milieu et le *medium*.

Deuxièmement, le fait que le contenu d'un *medium* soit un autre *medium* constitue les prémices de ce qui est énoncé en tant que remédiation par McLuhan, puis repris par Bolter et Grusin à la fin des années 1990 (Bolter et Grusin 1998). Porter une attention particulière à la remédiation équivaut à vouloir observer les relations entre les médias.

L'étude de ces relations entre les médias est le principal sujet d'un courant de la théorie des médias nommé intermédialité. L'histoire de l'intermédialité débute aux États-Unis d'Amérique avec Dick Higgins, un artiste affilié au collectif Fluxus, qui dans le cadre de ses projets, emploi le terme *intermedia* en 1966 (Higgins 1966). Dans un article consacré à ce sujet, « Statement on Intermedia », Higgins développe l'idée que les artistes, pour faire œuvre, ne se cantonnent plus à un seul champ artistique, mais s'emparent de plusieurs champs, donc de plusieurs médias, jusqu'à un point de rupture où finalement, l'évocation d'un *medium* pour désigner une œuvre n'est plus qu'« un point de référence » ou un « repère » que l'on emploie en tant qu'« outil critique » pour analyser cette œuvre. Ce qui fonde « l'approche intermédiale », c'est cette capacité à « souligner la dialectique entre les médias » (Higgins 1966 : § 4).

De cette intermédialité naissante émergent différents courants de pensée parmi lesquels nous nous intéressons particulièrement au creuset montréalais qui se développe depuis un peu plus d'une vingtaine années autour de plusieurs disciplines académiques artistiques et littéraires. En 2014, à l'occasion des dix ans de la revue *Intermédialités* que porte cette école de l'intermédialité montréalaise, Philippe Despoix résume « l'intermédialité non comme un objet mais comme un réseau complexe de relations, matérielles, techniques, sociales et sémiologiques, permettant de dessiner *a posteriori* des formes de médiation culturelle de fait toujours hybrides » (Despoix 2012 : 7 ; Tadier 2021 : 10).

Une expérience menée par André Habib, professeur en histoire de l'art et études cinématographiques à l'université de Montréal, illustre parfaitement le propos de Despoix. Lors de cette expérience, Habib commence son cours de cinéma expérimental par la projection d'un film d'une dizaine de minutes, sans aucune image sur la pellicule (Habib 2021). Dans la salle de cours plongée dans le noir, les étudiants observent alors la projection de la pellicule, sur laquelle s'accumule poussières et rayures pendant toute la durée de l'expérience.

> La première année, je m'étais assuré de faire traîner au sol l'amorce de pellicule transparente afin qu'elle accumule un peu de poussière et des marques d'usure. Sur l'écran, au fond de la salle, le faisceau du projecteur fait doucement vibrer un carré de lumière laiteux, un peu caillé, moucheté de ponctuations hasardeuses et de fines rayures qui se sont gravées sur la matière plastique du film. Parfois, un poil ou un grain de poussière s'invite et danse un temps dans l'embrasure de l'image. Le vrombissement régulier du projecteur emplit l'espace. Au moment où les griffes du projecteur transportent dans son mécanisme la section de la pellicule où se situe la collure, on entend clairement un petit déclic sonore, une façon pour le spectateur ou la spectatrice qui tendrait l'oreille de marquer le point de jonction de la bande (Habib 2021 : 95-6).

Cette réadaptation de *Zen for film* de Nam June Paik (œuvre présentée pour la première fois en 1964 dans le cadre d'un Festival Fluxus) que propose Habib durant son séminaire incarne l'approche intermédiale de l'école montréalaise. D'une certaine manière, ce film sans image rejoint le propos de McLuhan, puisque l'expérience repose sur les relations entre le *medium* et son milieu, sans avoir *a priori* de contenu, comme cela était le cas pour l'exemple de l'électricité. Tout le travail de Habib repose alors sur la présence de la poussière dans la salle et sur la projection de cette poussière à l'écran. À chaque tour que la pellicule fait sur le projecteur, ce sont de nouvelles poussières qui s'accumulent sur la bande avant d'être projetées. Ces poussières sont souvent faites « de pellicule (petite peau étymologiquement), de peau desquamée, de pilosités (poil, cheveu), de fines particules échues de la nature ou de l'activité des individus, des machines, des appareils, de l'industrie » (Habib 2021 : 110). Elles sont le résultat des actions du milieu dans lequel se trouve la bande. Ce sont les personnes présentes dans la salle, ainsi que les équipements qui, par leurs actions et leurs interactions, génèrent cette poussière que capte la bande avant d'être projetée à l'écran. À ce propos, Habib nous indique quelques caractéristiques techniques des équipements employés pour cette expérimentation, « projecteur 16mm Eiki au Xenon », « boucle de 12 pieds (un peu plus de 3m) », pour caractériser les comportements de ces objets techniques dans le milieu (Habib 2021 : 95). Par exemple, selon la longueur de la bande, la durée du film n'est plus la même et l'accumulation de poussière serait différente. L'expérience de Habib montre que même sans image, le *medium* se construit et véhicule du sens de par son interaction avec le milieu. Le *medium* n'est alors pas quelque chose que l'on définit en amont, mais bien quelque chose que l'on détermine *a posteriori*. Pour continuer dans cette direction, Elsa Tadier, en rapprochant l'intermédialité montréalaise des sciences de l'information et

de la communication, décrit la méthode de « penser par cas » (Tadier 2021 : 16) employée par les membres de l'école montréalaise : on ne généralise pas le traitement du *medium* au risque de l'essentialiser. De *Zen for film* à *Zen for Zoom*, Habib boucle la boucle de son expérience par la description d'une réadaptation de son expérience en pleine période de pandémie et de confinement, alors en distanciel, via le logiciel de vidéoconférence Zoom. La remédiation de son dispositif n'a pas le résultat escompté puisque, malgré de nouveaux effets tels qu'un « clignotement » dû à la « différence de battement d'image », « personne n'avait été sensible ou n'avait perçu le jeu des poussières » (Habib 2021 : 112). Dans ce dispositif, la technique numérique n'a pas bien rendu ces parasites et a altéré l'expérience que Habib souhaitait proposer à ses étudiants, et cela parce que la matérialité du numérique est profondément différente du dispositif initialement proposé à ces étudiants. Malgré l'intention initiale de Habib, cette remédiation nous montre à quel point le *medium* utilisé modifie en profondeur le sens produit dans le phénomène.

Un autre exemple flagrant de ce phénomène de remédiation est celui de la technique en philologie de collation des variants dans les différents témoins d'un manuscrit de référence. Une des méthodes consiste d'abord à découper le texte de référence en unités significatives (un mot ou un groupe de mots) que l'on dispose sur une première ligne, puis à collationner les variations observées dans les manuscrits témoins que l'on dispose sur d'autres lignes, une par témoin[8]. Ce travail de collation permet de recueillir et d'élaborer un appareil critique en prévision d'une édition critique du texte de référence. Avec du papier, ce travail est réalisé au moyen d'un collage entre des feuilles pour obtenir un accordéon que l'on peut déplier et remplir selon l'endroit où l'on se trouve dans le texte et selon la ligne du témoin concerné par la variation. Appliquées à un environnement informatique, les premières tentatives de reproduction de cette méthode sont réalisées dans des tableurs du type de ceux que proposent les logiciels Microsoft Excel ou LibreOffice. Un changement clair est opéré dans les tableurs : le fonctionnement par ligne est transformé par un fonctionnement en colonne, mais le principe reste le même. Toutefois, la possibilité d'insérer des formules dans les entrées du tableur permet de modifier une partie du processus et le rapport que le philologue entretient avec cette méthode. Une autre remédiation de cette méthode philologique, toujours dans l'environnement informatique, prend la forme du logiciel ChrysoCollate développé par Sébastien Moureau, professeur à l'Université catholique de Louvain[9]. ChrysoCollate est un logiciel développé

8. Pour cette pratique, il faut imaginer qu'après collage, le contenu d'un manuscrit contient sur une seule et longue ligne.

9. Voir le site web du logiciel, `https://cental.uclouvain.be/chrysocollate/`, consulté le 21 novembre 2024.

pour la collation de variants selon la méthode que nous venons de décrire. Cet environnement bénéficie de fonctionnalités avancées pour aider le philologue dans l'application de cette méthode (comme des fonctionnalités de déplacement et de recherche dans les manuscrits ou des modes de saisie adaptés aux types de contenus à ajouter) mais ne révolutionne absolument pas la méthode telle qu'elle est employée depuis plusieurs siècles. À travers ces différents environnements, on observe qu'il y a alors une remédiation de la méthode de collation sans que le changement d'environnement n'opère de rupture dans le processus. Toutefois, chaque nouvel environnement apporte des nouveautés dans l'appréhension de cette méthode, et des nouveaux usages pour produire un sens différent des autres méthodes, au même titre que la réadaptation de *Zen for film* et sa remédiation en *Zen for zoom* d'Habib ne produisent pas le même sens.

Finalement, ce que McLuhan et le courant de l'intermédialité montréalaise nous apprennent, c'est qu'il n'y a pas de rupture nette entre les médias, ni de révolution lorsqu'un nouveau *medium* est employé, d'ailleurs. Au contraire, il y a toujours une continuité entre une tradition et sa remédiation dans une nouvelle technique. Ainsi, la remédiation porte l'héritage des dispositifs qui l'ont précédé sans toutefois préserver exactement le même sens puisque celui-ci est produit par les actions dans le milieu, dont la technique fait entièrement partie, comme le montre la remédiation *Zen for Zoom* montée par Habib, ou le logiciel ChrysoCollate de Sébastien Moureau,

Une autre interprétation des travaux de McLuhan est faite par Friedrich Kittler de l'autre côté de l'Atlantique, en Allemagne, et sera dans les années 1980 un élément fondateur du courant germanophone de la théorie des médias contre lequel se positionne l'intermédialité montréalaise. Kittler est dans une double posture vis-à-vis de McLuhan. Si de prime abord, il semble rejeter le titre *Pour comprendre les médias*, puisque « comprendre les média [*sic*] demeure impossible » (Kittler 2018 : 31), Kittler reprend pourtant à son compte les deux thèses de McLuhan que nous venons d'énoncer, à savoir « *the medium is the message* », ainsi que la deuxième selon laquelle le contenu d'un *medium* serait toujours un autre *medium* (Mersch 2018 : 198) tant que cette thèse décrit des « systèmes composites partiels » (Kittler 2018 : 36), c'est-à-dire des systèmes où persistent des incompatibilités entre les médias.

Néanmoins, tous les systèmes ne sont pas partiels. Pour Kittler, les systèmes partiels concernent principalement les systèmes continus et analogiques. Tandis que les systèmes numériques, avec l'informatique et les supports optiques/électroniques de l'information, sont considérés en tant que « support composite total » (2018 : 36). Ce support composite total apparaît pour l'auteur comme une rupture avec la remédiation de McLuhan qui va annexer le concept de *medium*, puisque

tous les médias remédiés reposeront sur la même structure binaire de l'information. À l'intérieur des systèmes composites partiels, le signal de la télévision n'est pas compatible avec celui d'une radio – la réciproque est vraie également – ni avec celui du service postal. Ces médias optique, acoustique et écrit, sont techniquement différenciés et intègrent chacun une fonctionnalité et un usage particulier. McLuhan nomme cette fonctionnalité « extension » de l'être humain tandis que Kittler considère que cette différenciation, que l'on peut dater de 1880 environ, « rend celui que l'on appelle être humain fabriquable » (Kittler 2018 : 57). Ainsi, la thèse de Kittler diffère de celle de McLuhan à propos des médiations comme productrice de sens et propose de placer la production du sens dans des effets techniques que déterminent des technologies de l'information : la machine détermine l'être humain. Ces deux approches du *medium* ne sont pas opposées, ni irréconciliables, mais bien au contraire complémentaires, car elles offrent deux perspectives différentes d'un même objet. Là où la méthode de McLuhan est anthopocentrée, avec l'humain augmenté par les médias (McLuhan et Nevitt 1973), la méthode de Kittler est foncièrement axée sur les machines, sur leur fabrication et leur fonctionnement afin de déterminer leur contribution à la production du sens (Kittler 2015 : 14).

Une critique formulée à l'encontre de cette perspective matérialiste des médias et des technologies de l'information est sa dimension déterministe. Que ce soit chez McLuhan ou chez Kittler, les deux convoquent une forme de causalité traditionnelle d'une cause à l'origine d'un effet. Pourtant, le déterminisme que l'on impute à McLuhan n'est pas si bien fondé puisque lui-même s'en défend, comme le souligne Jean-François Vallée (2011). McLuhan, en s'appuyant sur les travaux d'un pair de l'école de Toronto, Harold Innis, nous rappelle que cette forme causale traditionnelle existe depuis que les grecs ont inventé le concept de Nature (McLuhan et Nevitt 1973). Il considère que cette Nature grecque est un territoire abstrait dont s'empare les philosophes au moyen de figures conceptuelles et, surtout, de l'écriture (Innis 1964). De ce constat, Innis a démontré que la capacité des grecs à séparer la pensée de l'être était due à l'écriture alphabétique. Ainsi, le processus de passage entre oralité et scripturalité aurait non seulement provoqué cette séparation, mais également produit deux événements : un divorce entre « sujet » et « objet » (McLuhan et Nevitt 1973 : 3) et l'émergence de la causalité (Mersch 2018 : 104).

Autrement dit, lorsque l'on connaît la cause, il devient possible de prédire quel en sera l'effet. Ce déterminisme technologique est une critique gênante à l'égard des thèses de McLuhan et de Kittler, puisque si on leur applique ce raisonnement, il deviendrait possible de prédire le sens que produirait une machine ou une technique – un *medium* – avant même que celle-ci soit mise en action.

Le problème fondamental de cette perspective est que si chaque phénomène est prévisible, il devient possible de le répéter et d'obtenir le même résultat autant de fois que l'on réitère l'expérience. Dans ce cas, l'action d'écrire avec un stylo bille au milieu d'un parc peut être répétée pour produire un texte identique à chaque itération. Toutefois, lors d'une des itérations de l'expérience, un événement inattendu pourrait survenir : le stylo pourrait par exemple être à court d'encre ou la météo devenir pluvieuse et détremper le papier. Le principe causal n'est plus suffisant pour expliquer les effets obtenus dans les deux cas imprévus. McLuhan se défend de ce principe de causalité et s'oriente plutôt vers une « causalité formelle » (qui s'intéresse aux formes) avec pour structure centrale de nos sociétés modernes l'électricité (Vallée 2011 : § 18). Il ne faut pas interpréter l'électricité comme le nœud au centre de la causalité, l'électricité n'est la cause de rien, mais comme une structure permettant de renverser le principe de causalité du fait de la circulation de l'information à la vitesse de la lumière. À cette vitesse, « de nouvelles perceptions directes de l'existence dépassent les concepts rigides de la Nature » (McLuhan et Nevitt 1973 : 18). Les perceptions sont l'élément central de cette phrase de McLuhan, car elles sont juxtaposées par l'auteur aux concepts. Les concepts, issus de la Nature abstraite grecque, permettent d'identifier les causes, sans eux il n'est pas possible de les nommer, tandis que les percepts permettent quant à eux d'identifier et de percevoir les effets à partir de nos sens. McLuhan nous renvoie ici à la dichotomie de la pensée et de l'être évoquée précédemment. Or les conceptions ou les perceptions ne sont jamais totales, elles ne sont toujours que partielles, car il y a toujours des phénomènes que nous ne prenons pas en compte. De plus, avec l'avènement de l'ère électrique, l'auteur déplace le monde abstrait de la pensée de la Nature abstraite vers le réseau électrique. L'information ainsi véhiculée à la vitesse de la lumière, se propage partout autour du globe et engendre à la fois de nouvelles formes de conception, puisque le *medium* façonne la pensée, et de nouvelles formes de perception. De cette manière, McLuhan renverse le principe de causalité selon le paradigme de la Nature grecque pour celui de l'électricité.

Malgré cela, le dépassement du déterminisme technologique n'apparaît pas comme flagrant. Pourtant, si Kittler défend également l'idée que la pensée réside dans des technologies de l'information à l'heure où l'être humain n'écrit plus (Kittler 2015 : 30), il faut se tourner vers le posthumanisme et le nouveau matérialisme avec les travaux de Hayles, d'abord, puis ceux de Barad, pour que se mette en place une autre approche causale. Inspirée par les travaux de Kittler, Hayles propose une solide défense à ses travaux accompagnée d'une nouvelle forme de causalité (Hayles 2005 : 7). Là où McLuhan est resté orienté sur l'être humain, Hayles décide de passer ce pas et de prendre appui sur le posthumanisme

pour étayer son propos. Le posthumanisme est une tentative de dépassement de certains dualismes (par exemple nature/culture) et de la non-différenciation entre humain et non-humain, ainsi que de la prise en compte du non-humain dans la production du sens, comme c'était le cas avec l'approche de Paveau sur l'écriture distribuée. Le posthumanisme se distingue ainsi du transhumanisme, et ne concerne pas le dépassement de la condition humaine par une extension technologique. Karen Barad en donne la définition suivante :

> Par « posthumaniste », j'entends la reconnaissance cruciale du fait que les non-humains jouent un rôle important dans les pratiques naturelles/culturelles, y compris les pratiques sociales quotidiennes, les pratiques scientifiques et les pratiques qui n'incluent pas les humains. Mais au-delà de cela, mon utilisation du terme « posthumanisme » marque un refus de considérer la distinction entre « humain » et « non humain » comme allant de soi, et de fonder les analyses sur cet ensemble de catégories présumées fixes et inhérentes. Un tel câblage exclut une enquête généalogique sur les pratiques par lesquelles les « humains » et les « non-humains » sont délimités et constitués de manière différentielle. Un récit performatif posthumaniste digne de ce nom doit également éviter de cimenter la dichotomie nature-culture dans ses fondations, permettant ainsi une analyse généalogique de la manière dont ces distinctions cruciales sont matériellement et discursivement produites (Barad 2007 : 32).

My Mother was a Computer est le titre d'un ouvrage fondateur quant à la redéfinition du principe de causalité (Hayles 2005). Dans cet ouvrage, l'auteure développe une thèse similaire à celles de McLuhan et Kittler. La proposition de Hayles consiste en un remplacement de Mère Nature par un Ordinateur Universel (Hayles 2005 : 3). À l'ère numérique, cet Ordinateur Universel est un référentiel pour interagir dans le monde : le monde devient le résultat de calculs. De la même façon que Bachimont a déplacé la raison graphique vers une raison computationnelle, Hayles propose un changement de perspective et de « vision du monde » (*worldview*) basé sur le calcul en tant qu'élément fondateur d'une modélisation du monde (Hayles 2005 : 17). Au-delà de la dimension numérique et des « intermédiations » que l'auteure développe entre oralité, scripturalité et code informatique, le fait qui nous intéresse est ce changement de paradigme et de conception du monde lors de la venue d'un nouveau *medium*. Si, pour Innis, la Nature et son abstraction sont la vision du monde rendue possible par l'écriture, Hayles nous dit qu'avec l'émergence des technologies numériques, ce paradigme s'effondre pour laisser la place à une nouvelle vision du monde composée d'une matrice, à partir de laquelle nous élaborons le sens des choses. À la différence des

travaux qui l'ont précédés, la proposition d'Hayles s'inscrit dans une perspective posthumaine où le non-humain est pris en compte au même titre que l'humain dans l'analyse des événements et la production du sens. Pour le dire autrement, il n'y a pas de pré-détermination ni de différentiation entre l'humain et le non-humain dans le modèle de Hayles. Comme le mentionne Barad, ce renversement causal est un pré-requis nécessaire pour analyser un événement (par généalogie). Sans remplir ce pré-requis, Barad affirme que l'enquête généalogique est alors exclue de l'analyse du sens produit lors d'une interaction, car les éléments en jeu seraient d'ores et déjà déterminés.

Si la première intuition que nous aurions pu considérer serait qu'un support technique, avec ses particularités, serait l'élément dont les caractéristiques façonnent l'écriture, nos hypothèses tomberaient alors dans une forme de déterminisme technologique, car il y aurait un support pré-déterminé et prêt à accueillir l'écriture. Cette intuition est erronée si l'on suit d'abord les critiques faites à l'encontre de McLuhan et de Kittler, puis les approches de Barad et de Hayles. L'intermédialité montréalaise abonde également dans cette direction, comme en témoigne l'ouvrage intitulé *Media do not exist* (Vitali-Rosati et Larrue 2019 : 39). Selon Larrue et Vitali-Rosati, considérer le support en amont d'une interaction, le pré-déterminer, c'est prendre le risque de décontextualiser le support de son histoire et de l'essentialiser, c'est-à-dire de ne pas prendre en considération les remédiations des médias dans l'analyse généalogique, étant donné que l'intermédialité montréalaise applique une méthode très similaire à celle de l'analyse généalogique à partir de cas concrets. Que ce soit pour Hayles ou pour Larrue et Vitali-Rosati, cette méthode généalogique implique d'analyser *a posteriori* les médiations d'une interaction pour en observer les « conjonctures » (Hayles 2005 : 2 ; Vitali-Rosati et Larrue 2019).

Afin de se débarrasser de tout risque d'essentialisation du *medium* et d'une réduction d'un événement à une seule médiation – celle par laquelle on désigne l'action principale qui nous intéresse, par exemple écrire – Larrue et Vitali-Rosati proposent de mettre ce terme de médiation au pluriel. Les médiations réfèrent alors à tous les éléments qui peuvent entrer en action dans un événement. Lorsque l'on écrit sur une tablette de cire, il n'y alors plus seulement une médiation qui engloberait toute l'action en cours, mais des médiations dont chacune renvoie aux états des caractéristiques des éléments présents dans cette action à un instant donné : par exemple, l'état de la cire selon si la température ambiante est élevée ou non, l'état de la pointe du stylet, etc.

À cet instant précis où se déroule l'action, ces médiations se lient alors en un mouvement et s'incarnent à travers une forme matérielle, le *medium*. Cette

liaison, Larrue et Vitali-Rosati la nomme conjoncture. Ainsi, ils définissent les « conjonctures médiatrices » comme suit :

> Il n'y a pas de médiateur ; mais un ensemble d'éléments en jeu au moment de l'action et ces éléments (forces, agents humains et non humains, aspects contextuels...) se croisent, se confondent, s'opposent, se rencontrent et se dédoublent dans un mouvement continu (Vitali-Rosati et Larrue 2019 : 52).

Le fait qu'un *medium* soit toujours la remédiation d'un autre *medium*, si l'on considère le support en tant que *medium* de quelque chose, c'est-à-dire en tant qu'ensemble de conjonctures médiatrices qui le produisent, évoque la possibilité qu'un support ne contienne jamais directement un contenu, mais contienne plutôt un autre support qu'il remédie (et ceci n'est de fait pas limité à une seule remédiation) (Vitali-Rosati et Larrue 2019 : 39).

Le problème posé par l'essentialisation du *medium* est évacué par les conjonctures médiatrices. Néanmoins, elles ne règlent qu'implicitement la question du déterminisme technologique. Pour asseoir plus solidement notre raisonnement et résoudre cette problématique du déterminisme, nous nous tournons vers les travaux de Barad et mobilisons la notion de « réalisme agentiel » (Barad 2023).

Karen Barad est à la fois philosophe et physicienne. En s'appuyant sur les littératures et les théories féministes et queer, notamment Haraway, ainsi que sur la physique théorique, Barad développe une épistémologie des savoirs situés au sein de laquelle le réalisme agentiel trouve une place importante. Le réalisme agentiel est le nom donné à l'approche philosophique de Barad. Pour construire cette approche, Barad s'appuie sur une interprétation de la philosophysique de Bohr pour qui la physique (quantique) était inséparable de la philosophie. Pour Bohr, les concepts théoriques ne sont pas de nature conceptuelle, ce sont plutôt des agencements physiques spécifiques. Il démontre que dans le cas d'une particule, les concepts de position et de quantité de mouvement relèvent d'un agencement physique particulier, en l'occurrence de la disposition d'outils de mesure particuliers dans l'environnement pour appréhender ces concepts. C'est-à-dire que ces concepts n'ont de valeur que lorsqu'ils sont agencés avec des outils de mesure permettant d'effectuer cette mesure. En outre, Bohr démontre que le principe d'incertitude de Heisenberg est due à l'impossibilité de déterminer simultanément ces deux concepts / mesures, car l'une relève d'un état fixe et l'autre d'un état mobile (et nécessitent différents outils). En ce sens, le dispositif de mesure agit dans cet agencement et produit une exclusion d'un autre état qu'il repousse à l'extérieur (Barad 2023 : 48). Le fait qu'une mesure ne puisse être effectuée, et correspondre à un instant T à un état donné, uniquement lorsqu'un dispositif est inséré dans un phénomène montre, pour Barad, l'inséparabilité

entre l'observateur et l'observé. L'observateur n'est pas un élément neutre et externe à un phénomène, bien au contraire, il intra-agit avec le phénomène. Pour Barad, la notion d'intra-action est fondamentale et désigne une autre forme de relation que la notion plus courante d'interaction. Plutôt que d'envisager des relations entre des entités dont chacune serait prédéterminée, il faut envisager les intra-actions comme des agencements permettant l'émergence des propriétés, comme c'est le cas de la position ou de la quantité de mouvement d'une particule.

> C'est à travers des intra-actions agentielles spécifiques que les contours et les propriétés des « composantes » des phénomènes émergents, se déterminent, autrement dit s'instancient, et que des concepts incarnés particuliers deviennent significatifs (Barad 2023 : 49).

Ce phénomène d'émergence des propriétés, Barad le nomme « matérialisation ». Nous pouvons ici faire le rapprochement avec la matérialité chez Hayles, qui serait également un produit pour Barad, et serait également un phénomène. Plus qu'un produit que l'on pourrait uniquement observer après le phénomène, il s'agit d'un processus qui se construit tout au long du phénomène, selon les conjonctures médiatrices en jeu à l'intérieur du phénomène – et par extension, excluant une partie de celui-ci. C'est cette double action locale d'extériorisation et de mise en exergue d'une forme intérieure à un phénomène qui s'appelle matérialisation et donne au phénomène sa matérialité. Les intra-actions conduisent à la matérialisation des forces matérialo-discursives, elles « comprennent toutes les forces que l'on classe habituellement comme "sociales", "culturelles", "psychiques", "économiques", "naturelles", "physiques", "biologiques", "géopolitiques" et "géologiques" » (Barad 2023 : 39-40).

En conceptualisant ce réalisme agentiel, Barad pose également la question de la causalité, car puisqu'il n'y a plus de prédétermination des phénomènes, la causalité cartésienne ne fait plus sens. Le phénomène causal de Barad repose sur cette séparation entre une intériorité et une extériorité que nous venons de mentionner. Ce que Barad nous explique est que l'extériorité dont il est question n'est pas un extérieur absolu puisqu'il fait partie du phénomène, c'est un extérieur relatif que provoque une intra-action sous la forme d'une coupure ou d'une sélection à l'intérieur du phénomène. La coupure réalisée à l'intérieur du phénomène écarte alors une partie du phénomène à l'extérieur de son champ d'action, mais pas à l'extérieur du phénomène lui-même. C'est cette coupure qui réalise une distinction entre un sujet et un objet, ou entre une cause et un effet. Une intra-action causale n'est donc pas un phénomène déterminant, mais un phénomène contraignant, puisqu'une autre intra-action causale provoquerait une autre matérialisation (Barad 2023 : 70). Dans le réalisme agentiel, la séparation

de la cause et de l'effet devient un phénomène à l'intérieur du phénomène sous la forme d'une intra-action qui contraint la matérialisation.

L'étude du cas de *Pfiesteria piscicida* (le tueur de poisson) réalisée par Astrid Schrader est un bon exemple pour expliquer cette causalité agentielle (Schrader 2010). *Pfiesteria piscicida* est le nom donné à un micro-organisme unicellulaire de la famille des dinoflagellés, *piscicida* signifie littéralement tueur de poisson. Parmi les quelque 1500 espèces recensées appartenant à cette famille, il n'y a eu qu'une seule espèce tueuse de poisson. L'histoire de *Pfiesteria piscicida* débute dans les années 1990 lorsque cette espèce prolifère dans l'estuaire de la Caroline du Nord (tout à l'Est des États-Unis) et devient la cause de la mort de milliers de poissons. La particularité de cette histoire est que le phénomène du micro-organisme tueur de poissons est resté un mystère scientifique pendant plus d'une décennie, sans que les scientifiques parviennent réellement à trouver un accord sur la cause qui rendrait toxique le dinoflagellé pour les poissons. L'histoire de ce micro-organisme relève *a priori* d'études en biologie marine et en écologie, pourtant, c'est depuis la sociologie qu'Astrid Schrader apporte un nouvel éclairage scientifique à cette énigme en 2010, en s'appuyant entre autres sur le réalisme agentiel de Barad.

La controverse autour de cette affaire concerne l'incertitude quant à la toxicité ou la non-toxicité du *Pfiesteria piscicida* envers les poissons. La principale difficulté autour du *Pfiesteria* réside dans la reproduction des conditions environnementales permettant d'amener le dinoflagellé à cet état de prédateur de poisson. De manière globale, ces organismes monocellulaires gisent sur la couche de sédiments au fond de l'eau. Ils mangent, se reproduisent, de façon sexuée ou asexuée et le cycle continue. Dans le cas du tueur de poisson, les scientifiques considèrent que des éléments comme le mauvais traitement des excréments issus de l'élevage animal entraîne un dérèglement de l'environnement du *Pfiesteria* et le pousse à tenir ce comportement. Le problème n'est pas si facilement résolu, car le cycle de vie de cette espèce est particulier puisqu'il comporte pas moins de 24 stades différents. Cependant, ce cycle de vie n'en est pas réellement un car ce n'est pas un cycle linéaire comme ce serait le cas la plupart des êtres vivants. Les stades ou les morphes du *Pfiesteria* répondent plus à des conditions environnementales qu'à un écoulement temporel. L'affaire peut encore être corsée puisqu'un morphe peut correspondre à plusieurs environnements différents, augmentant ainsi la difficulté de trouver un modèle de métamorphose cohérent. Pour le dire autrement, le fait qu'il n'y ait pas de modèle particulier que l'on pourrait associer aux évolutions du dinoflagellé nous indique qu'il n'y a pas de « détermination mutuelle entre ces organismes et un environnement particulier ». Dans ces circonstances, « il n'est pas possible de dire quel morphe appartient "naturellement" au cycle de

vie du *Pfiesteria* et quelles transformations sont induites par l'environnement » (Schrader 2010 : 282). Pour terminer le portrait du *Pfiesteria*, selon les états dans lequel il se trouve, ce micro-organisme se comporte plutôt comme un animal ou plutôt comme un végétal. En somme, c'est un organisme dont l'identité n'est pas et ne peut pas être prédéterminée. Pour ces raisons, il s'agit d'un organisme difficilement classable dans un règne puisqu'il peut appartenir à plusieurs règnes à la fois.

La thèse soutenue par Schrader est que l'incertitude est une caractéristique de ce micro-organisme et qu'en ce sens, il n'est donc pas possible de déterminer les propriétés exactes de cet organisme hors de son environnement. En considérant ces éléments, Schrader ramène le mystère du *Pfiesteria* à la théorie du réalisme agentiel et considère que ce dinoflagellé ne peut pas préexister à son environnement. Ce n'est que lorsqu'il est observé dans son environnement qu'il devient possible d'appréhender à la fois son comportement, ses métamorphoses et les propriétés de l'environnement dans lequel il se trouve. Les intra-actions ou les conjonctures médiatrices ne sont pas seulement des actions dirigées dans le sens des métamorphoses du *Pfiesteria*, mais elles sont des phénomènes agissant sur la totalité du milieu : le *Pfiesteria* n'évolue pas dans un milieu immuable, mais se co-construit avec son environnement. Le réalisme agentiel permet de formaliser l'approche du *medium* que proposait McLuhan lorsqu'il considérait que le *medium* et l'environnement se co-construisent et ne sont pas dissociables.

En appliquant le même raisonnement au cas de Marc Aurèle et de l'écriture de ses *Pensées*, les choix d'employer la tablette de cire et le grec pour inscrire le texte se dissolvent dans le phénomène qu'est l'écriture. Cette tablette de cire et le texte en grec ne sont plus des éléments qui préexistent l'écriture. Il n'y a pas un support qui serait là, au préalable, et attendrait sagement qu'un Marc Aurèle y déverse ses *Pensées* déjà formulées dans son esprit. Bien au contraire, la fonction de support émerge pendant la matérialisation de l'écriture sur celui-ci. Chaque action de sélection des éléments composant cette écriture provoque alors une coupure agentielle du phénomène et exclue par conséquent les autres phénomènes possibles. En ce sens, Marc Aurèle (et la culture qu'il incarne) opère une contrainte dans le processus d'écriture qui va déterminer ce qu'il peut ou ne peut pas écrire, puisque, par ce processus, il exclut ce qu'il ne peut pas écrire. Par exemple, le choix du grec lui permet d'exprimer les concepts du stoïcisme. Cependant, il ne peut pas (ou alors de manière détournée) exprimer en grec des concepts créés seulement en latin.

En nous autorisant un léger anachronisme, il est possible de prendre pour exemple la notion qu'est l'intime, qui provient du latin *intimus* et signifie littéralement l'intérieur de l'intérieur. L'*intimus* nous intéresse pour deux raisons :

premièrement ce concept n'a pas d'équivalent en grec ancien et, deuxièmement, il est plutôt associé à Augustin et à ses *Confessions*. Des traces de cette notion ont été retrouvées chez Sénèque (en latin) mais son emploi est légèrement différent du sens que lui donne Augustin plusieurs siècles plus tard (et plusieurs siècles après Marc Aurèle). Si le concept d'intime n'existe pas en grec (ni au sens où nous le concevons), Marc Aurèle ne peut alors pas l'exprimer ni même le conceptualiser, car le grec ne le permet pas. De fait, les *Pensées* ne peuvent pas être considérées en tant que journal intime, parce que Marc Aurèle ne pouvait exprimer quoi que ce soit qui se rapproche d'une intimité. En ce sens, le choix d'utiliser le grec opère cette coupure dans les possibilités d'expression qui s'offrent à lui.

Un autre exemple serait celui du choix de la tablette de cire pour écrire. Une des difficultés rapportées par les philologues sur ce texte touche à sa structuration même. Les scribes qui ont copié le texte dans les siècles qui ont suivi son écriture, puis par la suite les philologues spécialistes de Marc Aurèle, n'ont jamais réellement statué sur la structure même des *Pensées* en tant que livre. La tablette de cire n'est pas un objet pensé pour une écriture pérenne, ni même pour structurer le texte en profondeur : il n'y a pas la place pour effectuer cette structuration, et cela, sans considérer le fait que la cire est un matériau fragile. Regrouper de petites tablettes entre elles pour former des chapitres, des parties, voire des volumes semble alors inconcevable. Ce phénomène se ressent dans la construction des *Pensées* qui ne suivent pas une structure véritablement explicite et claire pour le lecteur.

Finalement, la théorie des médias, l'école de l'intermédialité montréalaise, le posthumanisme et le réalisme agentiel nous montrent que le support n'est pas un élément que l'on doit traiter au préalable de l'écriture, puisqu'il en est matériellement le produit. Ce n'est qu'une fois que nous avons le document écrit entre les mains que le support devient effectivement un support. En ce sens, il n'y a plus lieu de questionner la physicalité du support puisqu'elle est déjà matérielle.

Toutefois, cette approche rend possible l'étude de la production du sens par la reconstruction généalogique des différentes phases de matérialisation des conjonctures médiatrices selon les coupures agentielles qu'elles réalisent dans un phénomène particulier, que ces médiations soient le fait d'humains ou de non-humains. Ce faisant, il devient possible de comprendre le sens produit par un phénomène, dans notre cas l'écriture.

I.4 Conclusion

La problématique de ce chapitre porte sur la manière dont l'environnement technique participe à la production du sens de l'écriture. Nous avons vu que chaque

technique d'écriture (tablette de cire, machine à écrire, carte de navigation, etc.) institue un ensemble de médiations qui influent à la fois sur l'écriture et sur le sens que le dispositif produit. À travers l'énonciation éditoriale, il est apparu que l'écriture ne résulte pas uniquement des capacités cognitives de l'auteur, mais qu'elle est un phénomène distribué, impliquant des dynamiques culturelles, sociales et techniques, comme le montre l'exemple de Marc Aurèle.

Ce premier chapitre s'inscrit dans une perspective qui emprunte à la fois au courant matérialiste de la théorie des médias et à l'intermédialité montréalaise, lesquelles considèrent que l'importance du *medium* réside dans les médiations qu'il rend possibles. Cette approche, enrichie par le réalisme agentiel de Barad, invite à s'intéresser aux relations entre toutes les entités présentes dans l'environnement d'écriture, car ce sont ces médiations qui ouvrent et limitent les possibilités de la production du sens. Ainsi, étudier l'écriture, c'est analyser, suivant une démarche généalogique depuis le document produit, les relations entre les entités du dispositif d'écriture et les contraintes qu'elles font émerger.

Étant donné que les environnements d'écriture et les médiations qu'ils impliquent se complexifient au fur et à mesure des développements technologiques, notamment pour les environnements numériques sur lesquels nous allons nous concentrer par la suite, nous mobilisons de la documentation technique dans la suite de cette recherche pour être au plus proche du comportement de certaines médiations à l'œuvre dans ces environnements.

Afin de respecter l'approche que nous venons d'énoncer, l'étude de ces environnements est réalisée à partir d'une approche empirique et qualitative de plusieurs types de documents tels que des articles de revue ou encore des chapitres d'un livre. Or, nous l'avons vu, l'étude des relations entre les médias est devenue une spécialité de l'école de l'intermédialité montréalaise, dont la méthode est parfois très similaire à celle des sciences de l'information et de la communication. Aussi, nous réaffirmons notre méthode d'analyse des environnements d'écriture au croisement de ces deux approches de la recherche.

Au préalable de ces analyses, nous abordons dans le chapitre suivant les différentes couches des environnements informatiques, des couches matérielles des ordinateurs à la production de documents numériques, en passant par les couches logicielles et les modèles textuels. Cette exploration des strates techniques posera les bases nécessaires pour comprendre, dans les chapitres ultérieurs, le fonctionnement de cinq environnements d'écriture numérique et la façon dont ils contribuent à la production du sens des écritures numériques.

Chapitre II

Écriture numérique et production du sens

II.1 Introduction

Ce deuxième chapitre est axé sur les techniques et les environnements d'écriture numérique au sens le plus large. Écrire sur un ordinateur mobilise un ensemble complexe de médiations allant de l'alimentation électrique nécessaire au fonctionnement de la machine jusqu'à la production d'un document, en passant par tous les composants du *hardware*, les différentes couches d'abstraction matérielle qui permettent d'oublier ce dernier, les normes d'encodage des caractères, les formats, les protocoles de circulation des informations, les modèles textuels et les logiciels, avec leur rendu graphique affiché à l'écran.

Parmi tout ce fonctionnement, notre intérêt se porte sur les médiations qui produisent l'écriture dans un environnement numérique. Comme nous l'avons vu dans le chapitre I, le sens de l'écriture est le produit de l'ensemble des médiations à l'œuvre dans un environnement donné. En ce sens, puisque la production du sens est contingenté aux conjonctures médiatrices d'un environnement (Vitali-Rosati et Larrue 2019) , la problématique de ce chapitre est relative aux conjonctures médiatrices présentes dans les environnements d'écriture numérique.

Si l'écriture est le produit de médiations techniques, chaque type d'écriture va convoquer un ensemble plus ou moins complexe de médiations : inscrire des lettres sur du papier au moyen d'un stylo ou d'une machine à écrire, ou encore les tailler directement dans de la pierre, mobilisent des médiations techniques différentes. L'écriture numérique n'échappe pas à cette règle. Toutefois, ce type d'écriture convoque un ensemble de médiations beaucoup plus complexe que les exemples susmentionnés, d'où émerge une littérature dense sur le sujet depuis plusieurs décennies afin de la définir.

L'écriture numérique hérite ainsi des caractéristiques de l'écriture telle qu'énoncée dans le chapitre I, notamment la distribution de celle-ci à travers son environnement (Paveau 2015). Du fait des particularités de son environnement, l'écriture numérique se dote en prime de plusieurs caractéristiques sur lesquelles nous reviendrons dans ce chapitre : la variabilité (Bouchardon 2014), la calculabilité (Crozat 2016) et l'exécutabilité (Kittler 2015 ; Sauret 2020 : 101). Grâce à ces propriétés, les caractères saisis sur les touches d'un clavier suivent des instructions, des programmes, pour descendre les couches informatiques et devenir des données calculables. En interprétant ou en compilant ces instructions, les données font l'objet de calculs avant d'être à nouveau transformées en caractères, en textes, puis en documents.

À ces propriétés, Crozat *et al.* ajoutent différents niveaux de « concrétisation » de l'écriture en correspondance avec les couches des systèmes informatiques : le « niveau théorético-idéal » relatif à la couche la plus basse, celle de la représentation binaire de l'écriture et des calculs, le « niveau techno-applicatif, niveau de la manifestation » correspond à la mobilisation des contenus via des formats et des fonctions, le « niveau sémio-rhétorique » est celui de l'interaction avec les utilisateurs et, enfin, le « niveau socio-politique » est destiné à observer les « petites et grandes implications de la littératie numérique sur l'organisation des sociétés » (Crozat et al. 2011 : 32). L'enjeu de la littératie numérique traverse alors ces différents niveaux et ces nouvelles caractéristiques, puisqu'elles sont attribuées à une dimension technique de l'écriture, elles agissent sur nos modes de pensée et nos capacités à écrire, ce que soulève Bachimont avec la raison computationnelle (Bachimont 2000) .

Cette approche de l'écriture numérique relève plus du « lettré du numérique » que de « l'alphabétisé du numérique », nous disent Bouchardon et Cailleau qui distinguent une fracture entre ces deux postures (Bouchardon et Cailleau 2018 : 6). Selon eux, les alphabétisés du numérique le pratiquent sans pour autant avoir une culture du numérique, c'est-à-dire sans avoir une compréhension fine de la sémantique et de la rhétorique associée à ce que l'écriture d'un objet en environnement numérique implique. Or, pour réaliser cette approche en tant que lettré du numérique, Bouchardon et Cailleau préconisent d'observer l'écriture numérique non plus depuis l'environnement, mais depuis le milieu car, du fait de sa polysémie, le milieu désigne tout autant l'environnement que l'entre, c'est-à-dire à la fois une intériorité et une extériorité, tandis que l'environnement ne désigne que des éléments extérieurs. Cette approche de l'écriture numérique rejoint le prisme de l'intermédialité montréalaise que nous avons décidé d'employer pour observer l'écriture et les différentes médiations qui la constituent[1]. En ce sens,

1. Voir la section I.3.

nous adoptons l'approche du lettré du numérique que Bouchardon et Cailleau proposent dans leur article (Bouchardon et Cailleau 2018).

Du fait des caractéristiques propres à l'écriture numérique, nous posons la problématique suivante : comment les environnements d'écriture numérique participent-ils aux processus de production du sens ? Notre hypothèse est que l'écriture numérique, du fait de sa matérialité et de la représentation du monde sous forme de chiffres qu'elle incarne, entraîne des médiations spécifiques du savoir – dont une raison computationnelle – qui diffèrent profondément de la raison graphique engendrée par d'autres formes d'écriture plus traditionnelles. Ce changement de paradigme de l'écriture à l'écriture numérique se traduit notamment par une structuration de l'information formelle et explicite adossée à un ensemble de protocoles (des formats, des modèles de documents, etc.) décrivant les comportements des informations de façon non ambiguë.

Pour répondre à cette problématique et confirmer notre hypothèse, nous structurons ce chapitre en trois sections. Dans la première, nous revenons sur les conditions matérielles de l'écriture numérique contemporaine. L'acception courante de l'écriture numérique désigne très largement toutes les formes d'écriture sur support électronique, notamment parce que ces écritures sont enregistrées dans la mémoire de l'artefact sous forme de chiffres. À partir de la conception théorique d'un ordinateur telle qu'elle a été réalisée par Turing (1936), et à son implémentation matérielle développée par von Neumann (1945), nous proposons une exploration de la structure matérielle de traitement de l'information d'un ordinateur. Nous nous penchons également sur la façon dont les logiciels font abstraction de cette composante matérielle en se positionnant comme des médiateurs entre l'utilisateur et le système informatique.

L'empilement successif de couches d'abstraction matérielle jusqu'à l'interface graphique proposée à l'utilisateur nous mène au propos de la deuxième section dans laquelle nous passons au travers de ces représentations de l'information. Cet empilement de médiations commence à être instauré dans les années 1980, lorsque les recherches menées sur les interfaces graphiques par le *Palo Alto Research Center* (PARC) sont implémentées dans les ordinateurs et rendent ces derniers accessibles au grand public. Les interfaces véhiculent des métaphores et convoquent ainsi des objets hérités d'autres dispositifs techniques, comme c'est le cas de la page que l'on retrouve dans des objets imprimés, afin de rendre les données traitées par l'ordinateur intelligibles. Or ces métaphores tendent à camoufler la conception du texte telle qu'elle est modélisée dans les environnements d'écriture. En ce sens, notre proposition consiste à dépasser cette surface à l'écran et à s'intéresser aux modèles textuels et à la description formelle des éléments textuels contenus dans un document numérique.

Enfin, la dernière section de ce deuxième chapitre explore le document numérique depuis cette perspective qu'est le modèle textuel. Les modèles textuels sont pris en tant que théorie du texte, en s'appuyant notamment sur le modèle *Ordered Hierarchy Content Object* (OHCO) théorisé par DeRose *et al.* (1990) et les premières conceptions des langages de balisages qui y sont adossés, telles que *Standard Generalized Markup Language* (SGML), HTML et *eXtensible Markup Language* (XML). La structure hiérarchique présentée dans ce type de modèle textuel est verticale, et consiste en une succession d'éléments que l'on y imbrique à l'intérieur via l'utilisation de balises (*tags*). Nous défendons l'idée que les éléments structurels du modèle textuel définissent ce que l'on peut ou non inscrire dans un document numérique et, par conséquent, le sens accordé à ce qu'ils contiennent, car ces modèles textuels sont à la fois interprétables par les utilisateurs, ainsi que par des programmes et par extension, le système informatique jusque dans ces couches les plus basses.

II.2 Écrire dans un environnement numérique

Nous partons du présupposé que lorsque nous évoquons les mots environnement d'écriture numérique, ceux-ci sont synonymes d'un environnement d'écriture informatique et désignent la même chose. En conséquence, lorsqu'il s'agit de convoquer l'écriture numérique, nous pensons tout de suite à un ordinateur, aux claviers, aux écrans et aux pointeurs qui clignotent dans des éditeurs de texte ou dans les champs des formulaires en ligne. Avec le numérique ubiquitaire, ces pratiques d'écriture sont ancrées dans nos habitudes au point de ne plus les remettre en question (*Angles morts du numérique ubiquitaire : Glossaire critique et amoureux* 2023). Les dispositifs d'écriture analogique sont ainsi renvoyés à l'état de vestiges archaïques, comme peuvent l'être les machines à écrire, tandis qu'elles ont été fabriquées méticuleusement par des designers et des ingénieurs. Elles ont par ailleurs fait la fierté et la renommée de certaines entreprises comme Olivetti en Italie juste avant que les ordinateurs personnels n'arrivent sur le marché. Aujourd'hui ces machines sont complètement désuètes et inutilisées depuis presque une trentaine d'années. Elles sont maintenant exposées dans des musées – entre autres au MoMA et au Centre Pompidou – et sont intégrées dans des collections permanentes ou exhibées lors des expositions en lien avec les designers qui les ont conçues[2].

2. C'est par exemple le cas de la machine à écrire Valentine conçue en 1968 par le célèbre designer Ettore Sottsass, machine qui est devenue le produit emblématique de l'entreprise Olivetti lors de sa commercialisation en 1969. Comme nous le verrons plus loin, lors des mêmes années aux États-Unis, le président Johnson déclara qu'à l'échelle fédérale les ordinateurs devaient être compatibles avec la norme *American Standard Code for Information Interchange* (ASCII).

Pourtant, les derniers modèles fabriqués par ces entreprises l'ont été dans les années 1980 et 1990, comme c'est le cas du modèle ETP 55 Portable[3] où sont intégrés des composants électroniques et propose un nouveau design pour suivre le marché des ordinateurs. Les constructeurs ont opéré un changement de paradigme de l'analogique vers le numérique dès les années 1960 et ont suivi les innovations technologiques apportées par la miniaturisation des composants électroniques pour l'informatique. Pour preuve, en 1965, la firme Olivetti commercialise le tout premier ordinateur personnel, l'ordinateur P101, qui ressemble fortement à l'ancêtre des calculatrices programmables. Puis, en 1983, Perry A. King et Antonio Macchi Cassia conçoivent le premier ordinateur personnel portable d'Olivetti avec le modèle M10 en adaptant un clavier à un écran à cristaux liquide. Cet ordinateur, équipé du processeur Intel 80C85 en 8-bits, pouvait également se connecter à tout un ensemble de périphériques comme des imprimantes.

Il faut se rappeler qu'au début des années 1980, il n'est pas encore certain que l'ordinateur personnel (avec sa tour et son écran à tube cathodique) deviendra l'outil d'écriture par excellence. À cette époque, les machines à écrire ont encore quelques avantages sur les plans esthétique, financier et social puisque ce sont des objets que l'on retrouve encore présents dans les bureaux professionnels ou personnels.

La fin des années 1970 et les années 1980 marquent un tournant décisif pour l'ordinateur personnel avec l'apparition des interfaces graphiques, des logiciels de traitement de texte et la bataille qui sévit durant toute cette période pour en avoir le monopole. Matthew Kirschenbaum et Thomas Bergin détaillent dans leurs travaux cette course au développement de logiciels durant cette décennie, dans le but d'obtenir un monopole sur le marché de la production documentaire avec un ordinateur[4] (Bergin 2006*a*,*b* ; Kirschenbaum 2016). Avant l'engouement pour les interfaces graphiques et les gestionnaires de fenêtres – 1983 et 1984 avec l'entreprise Apple qui s'est largement inspirée des interfaces graphiques développées par le Xerox PARC dans les années 1970 – la seule interface affichée à l'écran était un terminal, un écran noir où clignote un curseur. Dans cette interface, la navigation s'y faisait au moyen de commandes. C'est ce que nous retrouvons avec les premiers logiciels de traitement de texte comme *Electric Pencil*. Cependant, ils ne permettent pas une gestion fine de la mise en page et ne fonctionnent pas sur tous les modèles d'ordinateurs présents sur le marché[5].

3. Cette machine a été conçue par Mario Bellini pour Olivetti en 1987, voir `https://www.moma.org/collection/works/3641`, site consulté le 21 février 2024.

4. Nous verrons les débuts des logiciels de traitement de texte plus en détails dans le chapitre III.

5. Un autre logiciel comme TEX développé en 1978 par Donald Knuth tente de résoudre ce problème de la mise en page selon une approche WYSIWYM, que Knuth nomme *Literate programming* (Knuth 1984*a*) alors que la tendance est plutôt aux interfaces WYSIWYG.

Ainsi, écrire sur un ordinateur (et ses dérivés la tablette et le téléphone) paraît aujourd'hui être une évidence alors que de lourds efforts ont été déployés à une époque où cette évidence était incertaine.

L'écriture numérique est ainsi à distinguer de l'écriture dans un environnement numérique : un ordinateur, Internet, le Web, une calculatrice ou une machine à écrire de la dernière génération. L'écriture numérique, en tant qu'abstraction, désigne la représentation du monde sous forme de chiffres ; elle se distingue de l'écriture réalisée dans un environnement numérique (ordinateur, web, calculatrice, etc.). Lorsqu'elle se fait au travers d'un *medium*, elle s'incarne matériellement, mais cette incarnation ne permet pas d'en épuiser toute la portée ni de la limiter à un seul support. En somme, cette représentation numérique du monde n'est pas nouvelle et ce n'est pas l'ordinateur qui l'a apporté. À notre connaissance, son origine remonte d'abord aux prémisses de l'écriture et aux développements des mathématiques, puis aux développements des premiers systèmes monétaires à partir du VIIe siècle avant notre ère, nous dirait Clarisse Herrenschmidt (2023 : 268) .

Dorénavant, lorsque nous ferons référence à l'écriture numérique, nous parlerons d'une écriture numérique dans un dispositif informatique.

Lorsqu'il s'agit d'observer l'écriture numérique à la loupe, on remarque qu'elle se distingue d'une écriture plus traditionnelle par le revêtement de trois caractéristiques que sont la calculabilité (Crozat 2016) , la variabilité (Bouchardon 2014) et l'exécutabilité (Kittler 2015 ; Chun 2023 : 39).

La première caractéristique est d'ordre computationnel : l'écriture devient calculable et peut donc faire l'objet d'instructions. Pour réaliser cette action, on procède à une équivalence où chaque caractère que l'on peut inscrire dans cet environnement à son pendant unique sous forme de bits. Lorsque chaque caractère peut être identifié en tant que nombre, il devient possible d'implémenter ce modèle dans une machine et de lui demander, grâce à des instructions, d'appliquer des calculs.

L'exemple idéal pour illustrer cette caractéristique n'est rien de moins que la machine imaginée par Alan Turing, qu'il présente en 1936 dans son article « On Computable Numbers, with an Application to the Entscheidungsproblem » dans la section *Computing machines* (1936). Ce que Turing décrit n'est pas une machine physique, mais un modèle théorique, une machine abstraite fondamentale pour les développements futurs de l'informatique. Cette machine est constituée de plusieurs éléments :

— un ruban (« *tape* ») divisé en sections (appelées « *square* ») dont chacune peut porter un symbole (0 ou 1, car cette machine est dans un système binaire)

— un organe de lecture (« *scan* ») pour lire les symboles un à un (« *scanned square and scanned symbol* ») et un organe d'écriture pour modifier un symbole ou en écrire un nouveau si la section est vide
— une mémoire pour se rappeler des sections déjà scannées (« *remember some of the symbols which it has been "seen" (scanned) previously* »)
— des instructions pour se déplacer sur le ruban, soit d'une case vers la gauche soit d'une case vers la droite, lire et écrire (« *scan and print* ») ou modifier la case scannée et se re-déplacer avant de s'arrêter.

Théoriquement, le ruban sur lequel la machine exécute ses programmes est infini vers la gauche et la droite, et cela, afin de permettre l'exécution des instructions les plus complexes. La machine de Turing ne s'intéresse pas aux résultats des instructions ni à leur signification, d'où résulte une forme d'automatisation de l'écriture. L'espace de la machine, aussi vaste soit-il, n'est composé que de séries de 0 et de 1 ainsi que de différents états, renvoyant à des instructions et permettant ainsi à la machine de modifier son propre espace. Cette capacité de modification peut être associée à la deuxième caractéristique de l'écriture numérique que Bouchardon nomme la variabilité [6].

Le passage du signe à l'unité atomique et discrète qu'est le chiffre signifie un changement de représentation du monde (au sens que Hayles donne au terme *worldview* (2005)) : le monde – ou l'espace – n'est alors plus signifié par des mots ou des concepts, mais le devient par des chiffres. Comme McLuhan nous le rappelle dans son ouvrage *Pour comprendre les médias* (1977 : 142), les alphabets phonétiques (contrairement à d'autres alphabets comme ceux composés de pictogrammes) sont asémantiques. Si toutefois les alphabets peuvent être liés à une culture d'où ils émergent, l'abstraction nécessaire pour représenter le monde sous forme de chiffres détacherait a priori cette vision de tout sens. En dehors de tout modèle mathématique abstrait, et cela quel que soit le langage ou la base utilisée pour l'écrire, `3`, `trois`, `three`, `III`, `0011`, `zéro zéro un un`, un chiffre ne signifie pas grand-chose s'il n'est pas associé à un discours ou à un système de valeurs particulier, par exemple le système métrique (Herrenschmidt 2023 : 12). En échange de cette perte de signification, l'écriture numérique y gagne cette particularité d'être calculable et mesurable.

L'écriture numérique se distingue également des autres types d'écriture par une troisième caractéristique. Il s'agit de l'exécutabilité. Il s'agit de la faculté d'un texte à déclencher des actions via du code informatique. Autrement, les écritures deviennent « capables [...] de lire et d'écrire par elles-mêmes » (Kittler 2015 : 30 ; Sauret 2020 : 101).

6. Nous reviendrons sur cette caractéristique, la variabilité, dans le troisième chapitre de la thèse.

Par exemple, lorsque l'on saisit une ligne de commande sous la forme de lettres dans un terminal, par exemple `pwd`, ou que l'on clique sur le nom d'un programme dont l'extension est `.exe`, les lettres composant la commande `pwd` ne signifient plus seulement *print working directory*, mais sont exécutées par l'ordinateur, tout comme le sont les lettres du nom du programme en `.exe`. Le sens accordé à ces caractères correspond aux instructions qui leur sont données pour que l'ordinateur puisse les exécuter. De cette manière, lorsqu'on appuie sur une touche du clavier, par exemple la lettre `a`, elle ne s'inscrit pas dans l'écran : l'instruction d'inscrire un caractère dans la mémoire de l'ordinateur est d'abord donnée à la machine, ensuite celle de l'afficher à l'écran au moyen d'un logiciel particulier (Kittler 2015 : 33). Néanmoins, le fait d'appuyer sur une touche du clavier lorsque l'ordinateur est sous tension ne suffit pas pour déclencher cette instruction : si aucun environnement dédié à l'écriture n'est déjà exécuté au préalable, le fait d'enfoncer une touche du clavier ne déclenchera aucune réaction de la part de la machine et l'écriture n'aura pas lieu. Par contre, lorsque l'on se situe dans un environnement où cette réaction est attendue, comme un éditeur de texte, la frappe d'une touche déclenchera un événement et le logiciel pourra générer l'instruction correspondant à l'action d'écrire.

Ces trois caractéristiques de l'écriture numérique ne sont pas uniquement des propriétés qui s'ajoutent à l'existant et, d'une certaine manière, rendraient l'écriture plus complexe. L'écriture, nous l'avons évoqué, peut être ramenée aux actions d'inscription dans la matière et de lecture. Or, la calculabilité, la variabilité et l'exécutabilité abondent dans le sens d'une écriture inhumaine puisque l'inscription et la lecture des caractères et/ou traces sur le support numérique sont des actions réalisées par la machine et ne le sont plus par l'être humain, comme le souligne Kittler (2015 : 30). Kittler poursuit sa réflexion plus loin jusqu'à soutenir, de manière provocatrice, que l'humain n'écrit plus et qu'à l'ère du numérique, c'est la machine qui écrit. À défaut de prendre cette provocation au pied de la lettre, elle ouvre la perspective d'une machine qui participe et contribue à l'écriture et, ce faisant, participerait à la production d'une épistémologie du texte et du document.

Seulement, la *machine* ou l'*ordinateur* sont des appellations vagues qui ne rendent pas très explicite les éléments qu'elles désignent, ni ceux qui sont impliqués dans cette action d'écriture et dans cette relation entre humain et machine.

La représentation d'un ordinateur est souvent associée à un couple *hardware* / *software*. La partie matérielle concerne tous les composants électroniques (carte mère, mémoires, périphériques, etc.), alors que la partie logicielle englobe tous les programmes permettant d'interagir avec la partie matérielle, comme le *Basic*

Input Output System (BIOS), le système d'exploitation ou encore un logiciel de traitement de texte comme LibreOffice.

Ce couple composé du matériel et du logiciel range l'ordinateur dans la catégorie des appareils programmables. La plupart de nos appareils du quotidien ne sont pas programmables : ils exécutent ce pour quoi ils sont conçus et ne font rien d'autre. Dans le cas d'un ordinateur ou d'un téléphone intelligent, ou de tout autre appareil programmable, leur conception prévoit qu'ils soient manipulables : ils n'ont pas de fonction précise, néanmoins ils sont capables de répondre à plusieurs fonctions tant qu'ils en ont les instructions. Un ordinateur qui n'a aucune instruction ne pourra rien faire une fois alimenté. C'est là que les logiciels interviennent : ils définissent un usage d'un ordinateur en manipulant des informations de façon à exécuter une suite d'instructions formelles.

Pour fonctionner, un ordinateur n'a besoin que des éléments suivants : une alimentation électrique, un processeur, une mémoire vive, des entrées et sorties et une carte mère auxquels viennent s'ajouter un certain nombre de périphériques (écrans, souris, clavier, etc.), des extensions pour prendre en charge une partie des calculs que l'on peut appeler des cartes filles (carte son, carte graphique) et des mémoires de stockage (disques durs). L'intérêt de ce type d'architecture matérielle se situe dans le fait que le jeu d'instructions et les données se trouvent dans une même mémoire, ou du moins qu'elles suivent la même logique de stockage (et d'encodage), ce qui permet de les manipuler l'un comme l'autre sensiblement de la même façon. Cette architecture est communément appelée l'architecture de von Neumann et fait référence à l'architecture de l'ordinateur EDVAC décrite par John von Neumann en 1945 (von Neumann 1945 : 1; Godfrey et Hendry 1993).

Parmi les différents composants d'un ordinateur, le processeur (ou microprocesseur) est le calculateur central : il manipule et traite toutes les données. Chaque modèle de processeur à une architecture qui lui est propre, ce qui veut dire que chacun traite les informations différemment (même si le résultat obtenu est identique). Un processeur est un assemblage de multiples types de circuits dont l'élément le plus petit est le transistor. L'évolution des processeurs a suivi la loi Moore jusqu'au début des années 2020[7], date à partir de laquelle nous arrivons à la limite physique de la miniaturisation d'un transistor.

7. La première loi de Moore est relative à l'évolution des processeurs dans le temps et stipule que le nombre de transistors présents dans les processeurs doublera tous les dix-huit mois pour un coût constant.

Le premier processeur commercialisé, le processeur Intel 4004, l'a été en 1971[8]. Il s'agissait d'un processeur 4-bits comportant pas moins de 2300 transistors. Lors de la commercialisation de cet objet s'opère un changement radical dans la conception des ordinateurs puisque, dès lors, du fait de la miniaturisation de ce composant, les ordinateurs deviennent accessibles au grand public. En suivant la première loi de Moore, les microprocesseurs ont continué à évoluer jusqu'à atteindre le nombre de plusieurs milliards de transistors par processeur, démultipliant ainsi leur capacité de traitement des informations.

Cette miniaturisation est rendue possible par la gravure des transistors dans des disques de silice (*wafer*) plutôt que l'usage plus coûteux et instable de relais et de tubes électroniques. Un transistor est un composant électronique dont le rôle est de laisser passer ou non le courant grâce aux propriétés du semi-conducteur à partir duquel il est fabriqué. En fonction de la valeur du courant qui lui est appliqué, le résultat associé à ce courant sera `0` ou `1`. Ce transistor est l'élément physique qui permet de composer les portes logiques (NOT, AND, OR, XOR, NAND, NOR) et de traiter les données. Parmi tous les traitements possibles, certains nécessitent de garder en mémoire des résultats intermédiaires pour aboutir. Ils sont alors stockés dans la mémoire vive en attendant d'être réutilisés.

Toutes ces informations traitées, qu'elles soient transformées ou mémorisées, proviennent de ce que l'on nomme des *entrées*. Ce sont ces entrées qui encodent les informations en chiffres. Une fois traitées, ou lorsqu'elles sont appelées par un programme, ces données transitent par les *sorties*. Elles font la transformation inverse et décodent le plus souvent les chiffres en caractères humainement interprétables.

L'encodage et le décodage des caractères accompagne toute l'histoire de l'informatique (et du numérique). Aux prémices de l'informatique, chaque matériel comportait ses propres programmes et tables d'encodage, rendant rarement possible la transposition des données d'un matériel à un autre par équivalence. Dans la plupart des cas, les données ne pouvaient pas circuler entre les différents modèles d'ordinateur, ou alors au moyen de transformations fastidieuses, rendant ainsi les traitements réalisés sur les données enfermées dans des silos.

La norme ASCII fait son apparition dans les années 1960 pour résoudre l'enjeu d'interopérabilité de l'encodage des données. Soumise à l'*American Standards Association* (ASA) (d'abord ASA puis *American National Standards Institute* (ANSI)) en 1961 par l'un de ses inventeurs, Bob Bemer, puis approuvée en 1963, l'ASCII permet d'encoder 128 caractères sur 7 bits, soit l'ensemble des

8. Voir la page web correspondante sur le site de l'entreprise Intel, consultée le 16 février 2024 : `https://www.intel.fr/content/www/fr/fr/history/museum-story-of-intel-4004.html`.

du clavier américain sans aucun accent[9]. Néanmoins, ce n'est pas parce qu'un encodage est reconnu en tant que norme que son usage est effectif à l'instant même de sa reconnaissance. Il faut attendre 1968 que le président des États-Unis d'Amérique Johnson demande à ce que l'ASCII devienne la norme fédérale d'encodage des informations afin de réduire les incompatibilités au sein des réseaux de télécommunication pour qu'elle commence à se répandre. Dès 1969, tous les ordinateurs achetés par le gouvernement des États-Unis étaient compatibles avec la norme ASCII.

Du côté des ordinateurs personnels, il faudra attendre le début des années 1980 pour que cette norme se répande grâce, entre autres, à son implémentation dans les ordinateurs construits par IBM. La norme X3.4:1986 en vigueur aujourd'hui, a été déposée auprès de l'ANSI en 1986. C'est à partir de cette norme que d'autres ont été développées et restent compatibles ASCII, comme c'est par exemple le cas de la norme Unicode, publiée en 1991, qui est la plus répandue de nos jours puisqu'elle encode le plus de caractères afin de prendre en charge le plus de langues possibles. Si ASCII contient 128 points de code, et sa version étendue 256, le standard Unicode permet d'en encoder plus de 149000 sur une vingtaine de bits par point de code dans sa version 15.1 publiée en 2023. Dans le but de préserver cette compatibilité entre les normes, il est d'usage d'encoder les 128 premiers caractères de façon identique à la norme ASCII (au moins pour l'anglais de base). Sous sa forme la plus répandue, UTF-8, Unicode est un encodage à taille variable grâce auquel on encode les caractères de un à quatre octets (soit de 8 bits à 32 bits). Les 128 premiers caractères de l'ASCII sont encodés sur un octet, tandis que les autres points de codes sont codés sur plusieurs octets.

Pour pouvoir utiliser ces tables d'encodage et stocker des données dans la mémoire d'un ordinateur, les utilisateurs ont besoin d'une interface les rendant accessibles et manipulables. Ces interfaces peuvent être rangées sous l'appellation de logiciel. Il est intéressant d'introduire les logiciels et leur fonctionnement à partir du matériel composant l'ordinateur et plus particulièrement à partir de la carte mère. Les fournisseurs de carte mère incorporent généralement dans leur carte une première couche d'abstraction matérielle, un BIOS[10], flashé dans la mémoire morte de l'ordinateur et programmé pour s'exécuter lors de la mise sous tension de ce dernier. Ce que l'on appelle *couche d'abstraction matérielle* en informatique représente la couche logicielle qui se trouve entre la partie matérielle et le système d'exploitation. Comme son nom l'indique, la fonction principale de cette couche est de permettre la manipulation du matériel tout

9. Il convient de noter que la version étendue de la norme ASCII, qui inclut notamment les accents, a été publiée en 1981.

10. Le BIOS désigne littéralement un *Système élémentaire d'entrée sortie*.

en faisant abstraction de celui-ci. Le BIOS, ce tout premier jeu d'instructions qu'un ordinateur exécute, est un programme propriétaire chargé d'initialiser la séquence d'amorçage (*boot*) de l'ordinateur, de trouver le système d'exploitation, les périphériques (*a minima* le clavier et l'écran) et d'opérer quelques vérifications de bon fonctionnement des composants comme c'est le cas de l'horloge temps réel qui fonctionne en tout temps, même lorsque l'ordinateur est éteint, et rythme la totalité des cycles des autres circuits.

Hormis quelques rares initiatives telles que Libreboot[11] et Coreboot[12], des logiciels libres et *open source* chargés de remplacer partiellement le BIOS propriétaire, la majorité des cartes mères sont liées à leur BIOS du fait de l'ajout par Intel, à partir de 2006, d'un sous-programme nommé *Management Engine* (ME) qui est accompagné d'un ensemble de modules comme *Boot Guard* et *Secure Boot* dont l'objectif est de veiller à ce qu'il n'y ait pas de corruption du système d'amorçage de l'ordinateur[13].

Ces programmes ont sans cesse été améliorés depuis leur introduction en 2006 et, aujourd'hui, ils empêchent toute modification de cette couche logicielle, la plus basse d'un ordinateur, si celle-ci n'est pas vérifiée et validée (avec un système de clés cryptées) par la firme propriétaire/fabricante. Il y aurait donc, au plus bas niveau d'abstraction matérielle dans un ordinateur, une imposition d'une vision de la machine aux utilisateurs, réalisée par les quelques sociétés qui détiennent le monopole de la production de ce composant.

Le BIOS est donc l'interface entre l'utilisateur et la machine qui nous permet de manipuler les différentes entrées et sorties du système, donc de gérer les périphériques, fonction que le système d'exploitation peut également réaliser une fois que la phase d'amorçage est terminée. Le système d'exploitation (*Operating System* (OS)), est un niveau d'abstraction supplémentaire et se retrouve à l'interface entre les applications logicielles et la couche matérielle. Un OS est composé d'un ensemble de programmes permettant la bonne gestion des ressources de l'ordinateur : mémoires, calculs, périphériques, registres, etc. Chaque OS a un fonctionnement qui lui est propre : l'architecture des informations – l'arborescence des dossiers, l'indexation des documents et des fichiers binaires change selon l'OS utilisé –, l'ordonnancement des tâches pour le processeur ou encore l'allocation de la mémoire, etc. Même si cela n'a pas toujours été le cas, les applications logicielles sont maintenant installées à l'intérieur des systèmes

11. Voir le site web de Libreboot : `https://libreboot.org/`, consulté le 03 avril 2024.
12. Voir le site web de Coreboot : `https://www.coreboot.org/`, consulté le 03 avril 2024.
13. Des informations sur ce sujet sont disponibles à cette adresse sur le site web de Libreboot : `https://libreboot.org/faq.html#what-systems-are-compatible-with-libreboot` ou dans la documentation des matériels d'Intel : `https://www.intel.com/content/www/us/en/search.html?ws=idsa-default#q=boot%20guard&sort=relevancy&f:@tabfilter=%5BDevelopers%5D`, consultés le 03 avril 2024.

d'exploitation et prêtes à être exécutées à partir de ce niveau d'abstraction de la couche matérielle. Le passage par un système d'exploitation permet aux logiciels de ne plus dépendre d'un modèle particulier du *hardware* et d'en faire justement abstraction, les rendant ainsi opérables sur différentes machines.

Ce tour d'horizon des particularités de l'écriture numérique et de l'agencement entre logiciel et matériel dans l'ordinateur nous montre que la conception de la machine ne permet pas à un auteur d'y inscrire des caractères dans sa mémoire, ni de pouvoir les consulter directement puisqu'elle lui est inaccessible à moins qu'un intermédiaire ne servent d'interface. La médiation entre une machine et un humain se fait au moyen d'un langage compréhensible par les deux parties, que l'on assemble sous la forme d'instructions qui, une fois empaquetées, forment un logiciel.

Pour symboliser la médiation du matériel par le truchement du logiciel à l'interface de l'humain et de la machine, l'entreprise Microsoft emploie la métaphore de la fenêtre (*window(s)*) à travers laquelle l'usager accède à l'espace numérique, et donc l'ordinateur. Pourtant, il ne faut pas s'y méprendre, quelle que soit la fenêtre logicielle, elle ne permet d'accéder qu'à un certain nombre fini d'instructions. Alors qu'en tant qu'appareil programmable qui ne se préoccupe pas de la signification du traitement des informations ni des résultats obtenus, l'ordinateur semble être un environnement beaucoup plus vaste que ce que cette fenêtre ne nous laisse croire (Turing 1936). Plutôt qu'une fenêtre comme ouverture ou passage vers le numérique, il serait plus juste de considérer cette fenêtre comme une vision du monde parmi d'autres. Cette vision du monde n'est pas seulement une vision particulière que l'humain a de la machine, car dans ce cas, nous serions dans un paradigme anthropocentré et utilitariste de la machine. En nous déplaçant de l'autre côté de la fenêtre, on se rend compte que la vision que porte la machine sur le monde est différente de la nôtre : la machine incarne une autre vision du monde sous forme de matrice, où chaque élément qu'elle perçoit l'est sous forme binaire. Le monde n'est alors plus que chiffres, calculs et distances, comme c'est le cas dans la proposition de Hayles lorsqu'elle remplace Mère Nature par une Matrice (Hayles 2005).

Un début de relation s'instaure entre l'humain et la machine grâce à l'entremise du logiciel. À travers cette interface, lorsque l'on touche une lettre du bout du doigt, la machine devient alors accessible et l'impulsion (électrique) que cette action génère se transforme en une lettre à l'écran. Pour autant, cette accessibilité est-elle un synonyme de mise en visibilité ? Le fait que « ça marche » rendrait-il le texte produit plus visible ?

Au contraire, c'est le rôle de l'interface graphique et des métaphores qu'elle véhicule que de cacher et d'invisibiliser le fonctionnement même de la machine

(Jeanneret 2011). Ce faisant l'interface graphique invisibilise les médiations qui interviennent dans l'écriture numérique et le processus de production du sens. Ainsi, le logiciel aurait une double fonctionnalité : la première est une médiation qui ouvre le dialogue avec la machine, tandis que la seconde en fait abstraction et la cache, ce qui a pour effet de rendre la machine quasiment invisible à l'utilisateur. Cependant, que découvrons-nous lorsque nous retirons ce voile devant la fenêtre ? Là se dévoile un vaste environnement constitué de formats, des protocoles et leurs flux électriques d'informations et de documents, parfois temporaires, voyageant d'une couche à une autre, prenant forme et se transformant pour suivre un cheminement prédéfini dans les composants de l'ordinateur jusqu'à la création d'un document final que l'utilisateur récupère. Chacune de ces fenêtres offre finalement une vision particulière d'un document, un modèle épistémologique qui lui est propre (Vitali-Rosati 2018*a*) et une manière de produire du sens.

II.3 La page est un doudou

L'interaction entre un humain et une machine consiste en une série d'instructions que l'utilisateur donne à la machine qui, ensuite, les interprète ou les compile avant de les exécuter . Le mécanisme sous-jacent à ce que l'on considère communément comme l'écriture numérique – frapper une touche du clavier et voir la lettre s'afficher à l'écran – s'avère être plus complexe. Le moment de la frappe n'est plus le moment où le caractère que l'on voit figurer sur la touche du clavier est inscrit dans le disque dur, il s'agit plutôt du moment où une instruction est donnée à l'ordinateur qui ensuite se charge d'inscrire le caractère correspondant sur le disque dur. Si l'on se trouve dans le cas de figure de la saisie d'un texte dans un éditeur de texte, l'instruction suivante, selon les logiciels et les actions souhaitées, consiste à afficher à l'écran le caractère encodé dans la mémoire de l'ordinateur.

Pour réaliser cette suite d'actions, Yves Jeanneret et Emmanuël Souchier partent du constat qu'il n'est pas possible d'écrire un texte numérique sans qu'un autre texte soit déjà présent pour réaliser cette action. Ce texte particulier qui préexiste toute activité numérique est nommé architexte (Jeanneret et Souchier 2005 ; Souchier et Jeanneret 1999).

L'architexte est un concept d'abord employé par Gérard Genette et désigne « l'ensemble des catégories générales, ou transcendantes – types de discours, modes d'énonciations, genre littéraires, etc. –, dont relève chaque texte singulier » (Genette 1979 : Quatrième de couverture).

En 2019, dans leur ouvrage intitulé *Le numérique comme écriture*, Gomez Mejia, Candel et Souchier résument l'architexte numérique comme :

> Initialement défini comme « écriture d'écriture » puis comme un « dispositif d'écriture écrit », l'architexte s'avère être un point de passage obligé pour toute activité numérique. Il n'y a effectivement pas d'écriture à l'écran sans un architexte qui la rend possible, l'accompagne et la formate. Pour la première fois de son histoire, l'homme a donc recours à des « dispositifs d'écriture écrits » spécifiques pour pouvoir pratiquer une activité d'écriture (E. Souchier, 1998, 2013). Or, précisément en ce qu'ils sont « eux-mêmes écrits », les architextes « sont des textes lisibles et interprétables. Porteurs et prescripteurs d'une écriture à venir, ils anticipent de ce fait une figure de l'auteur » (É. Candel, G. Gomez Mejia, 2013) et relèvent donc de « l'énonciation éditoriale » (E. Souchier, 1998) (Souchier, Candel et al. 2019 : 302).

Globalement, l'architexte incarne le cadre dans lequel les agents peuvent écrire. Il permet de faire la distinction entre un gabarit, entendu comme l'espace proposé par les éditeurs de logiciels ou applications pour écrire, et le texte saisi par l'utilisateur, c'est-à-dire le texte qui remplit le gabarit. Cet architexte, ce cadre, est régi par des règles qui définissent comment l'on peut écrire sur un support numérique, mais également comment les caractères à inscrire doivent être formatés.

Nous l'avons vu, l'architexte se positionne en tant que médiateur entre un auteur et la machine qu'il emploie pour écrire. Jusqu'à présent, la définition de l'architexte englobe largement tous les écrits qui permettent d'écrire à l'écran. Gomez-Mejia, Souchier et Candel précisent ce que sont ces méta-écritures et en dressent une typologie composée de quatre cadres d'écrits d'écran : le matériel, le système, le logiciel et le document.

En régissant ainsi les procédés de saisi du texte à tous les niveaux, un rapport de force semble s'instaurer entre les instances éditrices des architextes (que ce soit des collectifs, des institutions ou des entreprises) et les usagers (Jeanneret et Souchier 2005). Dans le cas d'un logiciel de traitement de texte lorsque, par exemple, Microsoft propose une modification de la police utilisée par défaut dans une version actualisée du logiciel Microsoft Word, Microsoft change également les manières d'écrire de tous les individus à travers le monde qui utilisent ce logiciel (et qui ont installé la mise à jour).

Si l'on s'arrête à la vision superficielle du texte, comme le propose Goody avec la raison graphique (Goody 1979), on ne voit que les modifications d'affichage des éléments graphiques, mais nous oublions celles qui sont invisibles et cachées derrière la page.

Certes, les interfaces d'écriture sont présentées sous la forme de gabarits que l'on doit remplir, comme on peut le faire avec des logiciels de création de diapositives, dont chacune est découpée en sections contenant tour à tour des images, des titres ou du texte. Dans cet exemple, nous avons affaire à une construction visuelle du document : un emplacement pour le titre de la diapositive, un autre pour le texte, un autre pour une image ou pour un graphique, etc. À ce sujet, Edward Tufte, dans un article sur l'utilisation du logiciel PowerPoint, démontre les effets du logiciel sur la forme des présentations et des informations qu'elles contiennent. La thèse qu'il y défend est que ce logiciel, en 2003, « [...] perturbe, domine et banalise systématiquement le contenu » notamment parce qu'il « facilite activement la réalisation de présentation légère » (Tufte 2003 : 22). À travers son analyse des usages de PowerPoint, Tufte nous montre qu'il ne s'agit pas d'un manque de fonctionnalité pour enrichir des supports de présentation, que l'auteur qualifie de pauvres, mais que le logiciel lui-même induit ce type de présentation avec des *templates* préfabriqués, des réalisations de graphiques automatisées ou d'autres fonctionnalités similaires qui appauvrissent les présentations parce que leur fonctionnement est calqué sur un modèle de présentation marketing qui n'est pas adapté aux sciences. Il ne s'agit plus seulement de remplir des gabarits préfabriqués, mais également de penser les formes que peuvent prendre l'information, ce que Tufte nomme « *The Cognitive Style of PowerPoint* ».

En changeant de paradigme, de la raison graphique pour celui de la raison computationnelle, l'assujettissement à ces architextes dépasse cette surcouche graphique et concerne également toutes les sous-couches (in)visibles de structuration du texte, mais aussi tout le processus d'inscription du document dans la mémoire, ainsi que les protocoles et méthodes qui permettent d'accéder à ces données. Comme nous l'avons vu précédemment, ce n'est pas l'image du texte affichée à l'écran qui est sauvegardée, mais bien une suite de caractères binaires dont l'écriture intermédiaire est composée de chiffres et de lettres.

Pourtant, on constate un paradoxe entre le nom d'un logiciel comme Pages, un traitement de texte disponible sous MacOS convoquant la métaphore de la page comme imaginaire en y enfermant les utilisateurs, et le rôle de guide qu'il doit remplir dans le traitement des informations. Dans ce cas-ci, le nom du logiciel ne réfère ni à son fonctionnement ni à son utilité. Tandis que dans les années 1980, lors de la genèse des logiciels de traitements de texte, les lettres `WP` appelaient le logiciel WordPerfect[14] et insistaient sur le mot comme élément central du logiciel, car c'est bien lui qui faisait l'objet des traitements informatiques[15], la démarche

14. Cet acronyme correspondait à la commande pour exécuter le logiciel depuis un terminal, alors qu'aujourd'hui il réfère plutôt au logiciel WordPress.

15. La plupart des autres concurrents employaient également le mot *word* dans le nom de leur logiciel : WordStar, MSWord, etc.

d'Apple en 2005 nous montre un changement de perspective : on passe du mot à la page. L'attention est portée à un autre endroit, sur une page que génère *Pages* et qui n'existe pas dans d'autres environnements. La page créée dans cet espace n'est pas reproductible ailleurs même si le document qui en résulte est ouvert, à un autre moment, par le biais d'un autre logiciel. La page de Pages devient un espace délimité qui n'existe sous cette forme qu'à cet endroit. Depuis vingt ans que cet outil est nativement disponible sur les ordinateurs de chez Apple, la compatibilité avec d'autres formats et/ou logiciels augmente tardivement, en témoigne les arguments de communication mis en avant sur la page web du logiciel[16], mais compatible ne veut pas dire identique. En plus de n'être accessible que sous MacOS, cette page ne l'est également que sous Pages : cette formulation courante laisse entendre que l'utilisateur devient alors sujet de son environnement d'écriture, nous dit Kittler (2015) .

Cette posture kittlerienne, que l'on peut qualifier d'essentialiste, pose les fondations des travaux de Hayles (2005), du posthumanisme, et du nouveau matérialisme, courants dans lesquels s'inscrivent en outre les travaux de Barad (2007 ; 2023) et ceux de Vitali-Rosati (2021)[17].

Pourtant, leur approche du rapport entre humain et machine est radicalement différente de celle de Kittler. Alors que Kittler identifie la machine et l'utilisateur par une série de propriétés ou définitions *avant* leur interaction, quasiment de manière décisive, les posthumanistes choisissent de ne pas déterminer les agents préalablement à l'environnement, mais comme résultats de l'agencement de plusieurs dynamiques dans un milieu donné. C'est en ce sens que sont mobilisées et développées les notions de *worldview* chez Hayles, où Mère Nature devient une Matrice (*My Mother was a Computer*), l'intra-action à la place d'interaction puisque les agents ne sont pas prédéterminés chez Barad et enfin l'éditorialisation chez Vitali-Rosati qui propose une ontologie de la médiation (métaontologie) selon laquelle le *medium* n'existe pas, on y retrouve la provocation de Kittler, et que toutes ces dynamiques, ces intra-actions, sont des médiations dont la matérialité, dans un agencement donné, produit du sens (Vitali-Rosati et Larrue 2019).

Ainsi, l'assujettissement de l'humain aux logiciels que nous avons mentionnés, que Kittler critique vivement dans ses travaux, n'a plus de raison d'être dans cette perspective non-essentialiste offerte par l'éditorialisation, puisque ces entités sont uniquement déterminées lorsqu'il y a intra-action. Les relations entre les agents ne peuvent plus être présupposées et leur détermination est réalisée depuis un

16. Voir le site web, consulté le 21 mars 2024 : `https://www.apple.com/pages/compatibility/`

17. Cette partie a été traitée dans la section I.3

référentiel quasiment unique si l'on considère que les paramètres de cet environnement sont variables et que la probabilité d'obtention de conditions strictement identiques est quasi nulle. Depuis cette perspective où l'on considère les différents agents comme des productions de leur agencement dans un environnement, il devient intéressant d'observer leur relation tout au long de ce processus pour comprendre comment ils s'affectent les uns les autres.

Pour réaliser cela, le dépassement de l'interface et des métaphores affichées à l'écran devient dès lors un acte symbolique nécessaire pour se soustraire à une vision essentialiste des actions de lecture et d'écriture. Pour effectuer ce changement de perspective, nous devons d'abord nous débarrasser d'un élément central entre l'humain et la machine : la page.

Dans un environnement numérique, la page devient skeuomorphique, c'est-à-dire qu'elle devient une métaphore matérielle instanciée dans une technologie pour que l'on ne perde pas nos repères en la manipulant. Même si elle peut s'avérer utile dans un usage alphabétisé, pour reprendre le terme de Bouchardon et Cailleau (2018), Gessler attire notre attention sur les problèmes engendrés par l'usage des skeuomorphes à l'intérieur des systèmes informatiques. En effet, ces métaphores matérielles génèrent des biais de perception et de compréhension de ce que l'on produit dans ce milieu informatique qui diffère d'une autre vision du monde non numérique (Gessler 1998).

Pourtant, la métaphore de la page hante de manière récurrente nos usages de l'ordinateur : on la retrouve dans les logiciels de traitement de textes dont Pages fait partie, dans les livres numériques ou encore dans le Web pour lequel chaque *Uniform Resource Locator* (URL) correspond à l'adresse d'une page web.

Cet objet qu'est la page a été instauré dans l'ordinateur uniquement pour reproduire une habitude et créer un lien fictif entre les visions du monde de l'imprimé et de l'informatique. Cet artefact produit une forme de réconfort auprès de l'utilisateur pour que le monde informatique lui semble plus tangible, qu'il ait quelque chose auquel se raccrocher, d'où sa déclinaison dans des espaces différents qui ne ressemblent plus du tout à des pages de livres ou des feuilles (par exemple la feuille aux formats A4, lettre US, ou le livre au format poche). La page affichée à l'écran n'existe qu'à cet endroit, il ne s'agit que d'un rendu graphique qui ne fait pas partie de l'écriture (au sens du texte saisi) : elle est un « simulacre » (Herrenschmidt 2023 : 465).

Malgré tous les efforts effectués depuis son instauration à l'écran, la page affichée n'est jamais la page imprimée, car aussi précis que soient les détails

typographiques que l'on peut y ajuster, elle ne reflétera jamais le grain, l'épaisseur, l'odeur ou toute autre caractéristique physique du papier[18].

La critique énoncée à l'endroit de la page ne doit pas être réduite à une apologie d'un mode sans page. Elle consiste à montrer qu'à vouloir préserver une habitude pour ne pas effrayer l'utilisateur, la page fait écran devant l'ordinateur, et cache la machine qui ne devient plus qu'un simple mécanisme au lieu d'être un agent de l'énonciation éditoriale.

Cette peur de l'informatique pourrait relever essentiellement de l'angoisse de l'arrachement d'une valeur qui définie l'être humain et devienne une caractéristique d'une autre entité, ne permettant plus de définir l'humain en regard de ce que lui seul est capable de faire.

Kittler, à ce propos, nous rappelle qu'historiquement les caractéristiques qui définissent l'être humain sont souvent le symbole du pouvoir et désigne plutôt les hommes alors qu'à l'instant même où cette caractéristique est déchue de son statut de marqueur de puissance, ce sont les femmes qui en héritent. Dans le cas de l'écriture – dactylographie et sténographie –, réservée pendant longtemps à la gent masculine, les femmes s'en emparent dès 1881, au moment même où les ventes de la machine à écrire Remington II explosent alors que chute le pourcentage d'hommes dans ce domaine (Kittler 2018 : 306). La Remington Model II de 1878 comporte une particularité, il s'agit de la première machine à écrire comportant une touche SHIFT pour avoir les hauts et les bas de casse sur le même clavier. Pourtant, malgré cette nouvelle fonctionnalité, les ventes ne se développèrent pas dès cette date. En 1881, l'entreprise modifie sa stratégie de vente et cible les femmes qui n'ont pas de travail. En parallèle, l'Association chrétienne de jeunes femmes de New York commence à former des jeunes femmes à la dactylographie, fait qui a été ensuite reproduit en Europe dû à son succès (Kittler 2018 : 322). Il y aurait donc une peur de perdre non seulement une caractéristique de l'humanité, mais surtout une caractéristique de la masculinité.

Interagir avec une machine demande une certaine rigueur : qu'il s'agisse de structurer un document ou de lui donner une série d'instructions (du code), une machine ne peut interpréter l'ambiguïté ou l'implicite culturel. Cela voudrait dire qu'aucun échange humain-ordinateur ne peut reposer sur des conventions culturelles implicites de lecture et que l'instruction donnée n'a, en elle-même, aucun sens. Dès lors, comment pouvons-nous admettre que quelque chose qui n'a pas de sens puisse en générer ?

18. Le terme *physique* est à distinguer de *matériel*. Nous l'employons dans le même sens que Hayles (2015). Physique renvoie aux caractéristiques d'un objet tandis que la dimension matérielle est une propriété émergente. Elle émerge de l'action d'inscription sur l'objet physique.

Le dépassement de la page et de l'écran est une proposition pour poser un regard *lettré*, et non plus seulement alphabétisé, ou plutôt un regard non anthropocentré sur cette question de l'écriture numérique. Cela afin de laisser de côté les modalités de définition de l'être humain en acceptant son caractère inhumain. En dépassant la page, il ne s'agit plus de poser la question de l'auteur de l'écriture, en admettant que ce soit bien la machine qui écrit, mais de se demander comment cette nouvelle fonction (intra)agit entre les agents d'un système d'informations. Que se passe-t-il lorsque cet ordinateur devient un agent actif qui écrit et transmet des informations entre, d'une part, l'instructeur (la personne qui donne des instructions) et les documents issus du traitement de ces instructions ? Dans cette configuration s'opère alors un changement radical du statut de l'ordinateur. D'abord à l'état de médiateur puis de support de l'écriture, l'ordinateur passe maintenant au statut d'entité agissante au sein d'un système d'informations.

Traverser la page pour atteindre les couches inférieures nous amène à faire escale sur la couche logicielle. D'ailleurs, rappelons que le logiciel a un statut intéressant : on le considère souvent comme un médiateur, un agent qui permet la communication et l'interaction humain-machine, pourtant ce n'est pas le cas de toutes les recherches. Kittler et sa très célèbre provocation « le logiciel n'existe pas » (2015 : 29), nous rappelle que ces écritures (qui nous permettent d'écrire), sont stockées et traitées par la machine exactement de la même façon que n'importe quelle écriture numérique.

Ces textes numériques (logiciels et documents) sont situés au même niveau hiérarchique dans l'architecture du système d'exploitation et sont traités de manière identique par le processeur. La définition de Souchier, qui qualifie les logiciels de « dispositifs d'écriture écrits » (Souchier, Candel et al. 2019 : 302), souligne cette égalité entre les écritures numériques : en effet, le logiciel partage la même nature que les textes que nous écrivons à l'intérieur de son environnement.

Toutefois, une distinction persiste. Si le texte peut être remédié dans un autre format – et être imprimé par exemple –, le logiciel quant à lui ne peut exister que dans son environnement numérique. Son code source peut lui aussi faire l'objet d'une remédiation (Bolter et Grusin 1998) mais il sera dénaturé, car sa fonction principale est d'être exécuté pour organiser le traitement des informations dans un ordinateur. D'ailleurs, Herrenschmidt nous rappelle que le terme de logiciel a été forgé à partir de la contraction du mot « logique » avec le mot « matériel » (Herrenschmidt 2023 : 474), pour justement montrer à la fois l'opposition du logiciel avec l'aspect matériel (*hardware*) et marquer leur complémentarité : l'ordinateur (*hardware*) serait très peu accessible (voire inaccessible) sans logiciel, et le logiciel n'existe pas en dehors de l'ordinateur.

Lorsque l'on définit le logiciel en opposition au *hardware*, comme nous l'avons énoncé précédemment, nous les positionnons tous deux au même niveau hiérarchique. Ils en deviennent ainsi faussement comparables et le logiciel se voit doté d'une propriété immatérielle, ce qui a pour effet d'être en totale contradiction avec ce que nous venons d'énoncer sur la place du logiciel aux côtés de n'importe quel texte à l'intérieur de la mémoire de l'ordinateur. En suivant les pensées d'Herrenschmidt et de Kittler, s'ils affirment que le logiciel n'existe pas, c'est parce qu'il n'est pas une entité propre, mais est une partie de l'ordinateur. Il se constitue en un organe de communication interne et externe à l'ordinateur.

La relation entre un logiciel avec le reste de son écosystème peut être qualifiée de fragile et nécessite une attention particulière puisque l'agencement du logiciel avec le reste de l'environnement ne peut se faire qu'au prix d'une compatibilité entre les divers éléments qui le composent – même si les systèmes d'exploitation récents lissent et camouflent cet aspect-là. Pour fonctionner, le logiciel doit être compatible avec plusieurs composants de l'ordinateur. Les premiers composants sont physiques : est-ce que l'ordinateur a une carte graphique, quel type de processeur ou la quantité de mémoire vive, etc.

Ce phénomène, connu du temps des premiers logiciels comme *WordPerfect* (Bergin 2006*a* ; Kirschenbaum 2016 ; Kittler 2015) est aujourd'hui moins fréquent, notamment parce que 1) les logiciels à installer sont disponibles pour beaucoup de *hardware*, et 2) parce que le développement des téléphones intelligents depuis une vingtaine d'années a donné naissance à un nouveau format d'application : les *progressive web app*. Ces applications, basées sur les technologies du Web pour fonctionner, sont donc exécutables sur plus de supports puisqu'elles sont agnostiques vis-à-vis du système d'exploitation[19].

L'environnement matériel est donc une première condition pour faire fonctionner un logiciel. Le deuxième composant est le système d'exploitation (et sa version). En fonction du système d'exploitation – et de sa version – un logiciel pourra y être installé à l'intérieur. Ce deuxième paramètre ne doit pas être sous-estimé, car l'écosystème des logiciels fonctionne sur la base d'un système réticulaire : les programmes ne sont pas développés *from scratch*, ils s'appuient souvent sur d'autres briques logicielles qui elles-mêmes s'appuient sur d'autres briques logicielles. Chacune d'entre elles dépend d'une version particulière de l'autre. Si une version venait à être mise à jour sans vérification préalable, alors le château de cartes pourrait s'effondrer et le logiciel ne plus fonctionner.

D'ailleurs, une pratique courante en développement informatique consiste à créer un environnement virtuel – une bulle – à l'intérieur même de son ordinateur pour y installer des versions sélectionnées de dépendances logicielles afin

19. Le Web, en tant que couche d'abstraction matérielle, est étudié dans le chapitre IV.4.

qu'elles ne soient pas victimes d'un effet de bord dû à une mise à jour d'un autre programme (et d'autres dépendances).

Le logiciel est un langage de haut niveau qui permet de manipuler des données jusqu'au plus bas niveau de l'ordinateur, au niveau des entrées et des sorties. Toutes ces manipulations sont exécutées en appelant des instructions dans ce réseau de dépendances et de logiciels pour que les données puissent descendre les couches de l'ordinateur et être transformées jusqu'à atteindre leur espace de stockage dans la mémoire morte.

Le nom d'un logiciel comme MSWord ou LibreOffice désigne plus que les vagues notions que peuvent être leur fonctionnalité principale, dans ces cas-ci l'écriture d'un texte. Un logiciel évoque la totalité des instructions nécessaires à la manipulation des informations (ce que nous appelons des fonctionnalités), mais aussi l'intégralité de l'écosystème dans lequel le logiciel fonctionne. À titre d'exemple, les logiciels obsolètes que l'on ne peut plus faire fonctionner dans les environnements récents doivent faire l'objet d'une émulation pour être exécuté : les conditions permettant leur bon fonctionnement sont simulées à l'identique pour reproduire la machine sur laquelle le logiciel a été initialement conçu. Ce phénomène est récurrent dans plusieurs domaines tels que les archives du web ou les jeux vidéos.

Par exemple, le logiciel *Electric Pencil*, premier logiciel qualifié de logiciel de traitement de texte (*word processor*) en 1978 illustre bien ce propos. Si l'on souhaitait en faire l'installation sur un ordinateur fabriqué en 2025, ce ne serait pas une chose très aisée puisqu'il a été développé pour un ordinateur datant de son époque, le TRS-80 Model I. Pour réaliser cette installation, il faudrait connaître les spécificités de cet ordinateur et les simuler pour qu'*Electric Pencil* puisse être exécuté correctement. C'est en ce sens qu'*Electric Pencil* embarque avec lui tous les agencements et l'histoire du *medium* pour lequel il a été développé. Le nom d'un logiciel désigne alors l'ensemble des éléments nécessaires à son exécution, *hardware* et *software*, soit un environnement d'écriture numérique.

À l'instar de McLuhan (1977), l'on pourrait percevoir les environnements d'écriture numérique comme des espaces construits – des architectures de l'information soignées – avec une topologie qui leur est propre et à travers laquelle chaque suite d'instructions forme une route que des unités sémiotiques empruntent pour y être transformées en unités calculables. Cette architecture de l'information a pour « priorité des questions de sens, et notamment de modélisation du contenu » (Broudoux, Chartron et al. 2013 : 4). Le sens est ainsi modélisé par l'environnement d'écriture.

D'après cette perspective matérialiste, le message, soit les unités sémiotiques qui circulent dans cette architecture, importe finalement peu. L'élément qui

compte c'est le *medium*, c'est-à-dire l'environnement d'écriture lui-même. Si l'on considère l'environnement d'écriture numérique comme une architecture des informations, nous pouvons reprendre la définition proposée par Broudoux *et al.*, qui identifient la notion d'architecture des informations par l'assemblage de trois composantes : des technologies de l'information, une organisation des connaissances et l'expérience utilisateur (Broudoux, Chartron et al. 2013 : 9). À travers cette définition, nous retrouvons bien certains éléments évoqués jusqu'à maintenant : le *hardware* et le *software* comme technologie de l'information et comme modalité d'intégration de l'humain, via l'expérience utilisateur, dans cette architecture.

L'humain est donc une composante de l'environnement et de l'architecture des informations, il ne lui est pas extérieur. Cette assertion ne fait pas référence à une intentionnalité que l'humain pourrait apporter dans l'environnement. Si l'on reste dans notre perspective matérialiste, en intégrant l'humain à l'environnement d'écriture, celui-ci devient une partie de l'architecture des informations, il devient un élément de construction de cet espace qui permet d'écrire. Autrement dit, l'humain, au même titre que le *hardware* et le *software*, provoque des médiations productrices de sens à l'intérieur de l'environnement d'écriture numérique.

La dernière composante de l'architecture des informations, l'organisation des connaissances, réfère à la modélisation du contenu et à sa « gestion » (Broudoux, Chartron et al. 2013 : 9). Dans son acceptation la plus large, ce terme est « très fortement lié à la tradition bibliothéconomique d'indexation et de classification des contenus documentaires » (Hudon et Hadi 2012 : 1). Toutefois, dans le cadre de cette étude, nous proposons de l'aborder sous l'angle spécifique de l'environnement d'écriture numérique.

Dans ce contexte particulier, l'organisation des connaissances s'opère principalement à travers deux dimensions : la manipulation des documents et les modèles textuels.

II.4 Document numérique et modèle textuel

La page maintenant levée, l'écriture numérique se dévoile à nous dans un agencement de formats, de protocoles de communication et de normes d'encodage. La partie concernant les logiciels est mise de côté, pour l'instant, puisque nous y reviendrons dans les chapitres III, IV et V. Dans cette dernière section, nous manipulons des notions plus englobantes telles que texte ou encore document. Le prisme sous lequel nous les observons est certainement inhabituel, puisque nous les considérons depuis un point de vue empirique : celui du modèle textuel.

Définissons ce qu'est un modèle avant de poursuivre plus en aval de ce développement. Le modèle, pour reprendre les mots de Bachimont, est :

> toute représentation d'un objet ou d'une situation permettant de les étudier sans avoir à convoquer le réel dont le modèle est la représentation. Autrement dit, le modèle fournit un substitut du réel que l'on peut directement expérimenter sans avoir à se confronter au réel lui-même (Bachimont 2022*c* : 6).

Selon cette définition, le modèle est une abstraction d'un réel « borné et délimité », elle est une réduction d'une situation ou d'un objet qui doit rester intelligible. La « Carte de l'Empire » de Borges illustre bien cette particularité du modèle (Borges 1994 : 107) . Dans cette fiction, des géographes ont créé une « Carte de l'Empire » à l'échelle 1:1. Autrement dit, pour représenter la totalité de l'Empire, la carte créée se superpose à l'Empire lui-même et devient inutilisable, car elle perd tout ce que l'objet carte peut offrir à ses lecteurs.

En réduisant un modèle à un ensemble de particularités du réel, celui-ci prescrit et articule une ou des théories, que Bachimont considère en tant que « la forme la plus générale qu'adopte une compréhension du monde ou d'une partie de ce dernier » (Bachimont 2022*c* : 5), pour justement rendre compte de cette forme réduite du réel. Pour reproduire le réel, l'articulation des différentes théories entre elles doit être calculable : en somme le modèle agit comme une machine. Le modèle peut donc être numérique (au sens de calcul). Néanmoins, ce propos peut être relativisé : les modèles ne s'imposent pas totalement aux données qu'ils traitent pour représenter le réel et peuvent être réorganisés, adaptés et ajustés à partir d'une abstraction réalisée sur ces mêmes données.

En ce sens, les modèles textuels incarnent des théories du texte, ils en sont la représentation formelle nécessaire à l'écriture numérique (Pierazzo 2016 : 37), puisque sans eux, l'écriture serait difficilement calculable. Par exemple, c'est le cas des langages de balisage dérivés de SGML[20]. Chacun d'entre eux incarne une vision et une théorie du texte. Ainsi, les modèles textuels sont des éléments structurels des documents et de leur contenu, car ils réalisent une coupure agentielle dans le processus de construction du document en ce sens qu'ils injectent un biais dans ce dernier en imposant une architecture de l'information et des concepts qui permettent de l'interpréter. Le texte numérique est donc affaire de modèles.

Avant d'approfondir cette notion de modélisation du texte, il convient de définir ce qu'est un document – ainsi qu'un document numérique – pour mieux

20. *Standard Generalized Markup Language* (SGML) est le langage de balisage développé par Charles Goldfarb dans les années 1980. Ce langage est devenu une norme internationale en 1986 et a été utilisé comme référence pour les développements du HTML et du XML.

situer l'intervention du modèle textuel dans le processus de construction de ce dernier.

Le document

La notion de document renvoie à toute une histoire de la documentation. Héritière de la bibliographie, la documentation désignait, lors d'une phase de documentarisation (Salaün 2004 ; Pédauque 2007 : 15) au tournant du xxe siècle, un ensemble d'informations – communément appelées documents – accompagné d'un système d'organisation et de recherche des connaissances (Briet 1951 ; Otlet 1934) .

Michael Buckland, professeur émérite à Berkeley, réalise à partir des années 1990 une revue de la littérature sur la notion de document. Il passe notamment au travers des textes de Paul Otlet, Suzanne Briet, Donker Duyvis ou encore Shiyali Ranganathan pour construire cette revue de la littérature. Nous nous appuyons notamment sur les travaux de Buckland pour donner une définition de la notion de document.

Le document est un objet délicat à définir tant il revêt de formes différentes. Ceci est certainement lié au fait que l'emploi du terme document désigne une multitude d'objets que nous manipulons au quotidien, dans différents lieux, que ce soit au travail, à l'école ou à l'université, dans des administrations publiques ou encore chez soi. Un document peut très bien prendre l'apparence d'un formulaire administratif ou d'un livre bien rangé sur une étagère d'une bibliothèque, ou encore d'une photographie stockée dans les locaux des Archives nationales. En ne désignant pas un objet précis, mais plutôt des informations organisées dans un dispositif, le terme de document devient un mot-valise renvoyant à tous les objets qui peuvent figurer dans ces dispositifs.

Paul Otlet et Suzanne Briet, deux figures importantes de la documentation durant le xxe siècle, s'appuient sur l'étymologie du document pour en donner une définition plus précise. Le document provient ainsi de *documentum*, qui signifie exemple, modèle, leçon, enseignement, démonstration. Cette définition est intéressante puisqu'elle ne fait pas référence à un support de l'information, comme un livre, une feuille, un catalogue, mais correspond à un usage.

Que ce soit chez Otlet ou Briet, les deux visions du document ont en commun de mobiliser une action humaine. Pour la première, il s'agit de l'action que réalise l'auteur du document, tandis que pour la seconde, il s'agit de la conception et de la création d'une architecture dans laquelle les connaissances sont organisées et se matérialisent sous la forme de document. Quelle que soit la perspective

à travers laquelle on entrevoit le document, celui-ci s'avère toujours être une production (Cotte 2004).

Otlet défini de son côté le document comme ceci :

> 1. Livre (Biblion ou Document ou Gramme) est le terme conventionnel employé ici pour exprimer toute espèce de documents. Il comprend non seulement le livre proprement dit, manuscrit ou imprimé, mais les revues, les journaux, les écrits et reproductions graphiques de toute espèce, dessins, gravures, cartes schémas, diagrammes, photographies, etc. La Documentation au sens large du terme comprend : Livre, éléments servant à indiquer ou reproduire une pensée envisagée sous n'importe quelle forme. 2. Le Livre ainsi entendu présente un double aspect : a) il est au premier chef une oeuvre de l'homme, le résultat de son travail intellectuel ; b) mais, multiplié à de nombreux exemplaires, il se présente aussi comme l'un des multiples objets créés par la civilisation et susceptible d'agir sur elle ; c'est le propre de tout objet ayant caractère corporel et agencé techniquement (Otlet 1934 : 9).

Si l'on continue notre lecture d'Otlet, ce dernier nous donne les éléments constitutifs d'un document, à savoir :

> Qu'est-ce qui dans le Livre lui est propre, qu'est-ce qui est proprement bibliographique ? On a déjà dit la distinction entre : a) la Réalité objective, b) la Pensée subjective ou l'état de conscience provoqué ou le moi par la réalité, c) la Pensée objective qui est l'effort de la réflexion combinée et collective sur ces données premières jusqu'à la science impersonnelle et totale, d) la Langue, instrument collectif de l'expression de la Pensée. Collection totale, tout livre contient ces quatre éléments associés concrètement en lui-même et que, par abstraction seulement, il est possible de dissocier et d'étudier à part. Ce qui est propre au livre, c'est le cinquième élément : la pensée désormais fixée par l'écriture des mots ou l'image des choses, signes visibles, fixés sur un support matériel (Otlet 1934 : 10).

Aussi, il est possible de résumer la pensée d'Otlet par cette dernière citation :

> La définition la plus générale qu'on puisse donner du Livre et du Document est celle-ci : un support d'une certaine matière et dimension, éventuellement certain pliage ou enroulement sur lequel sont portés des signes représentatifs de certaines données intellectuelles (Otlet 1934 : 43).

Les travaux d'Otlet font finalement apparaître de nouveaux éléments dans la définition du document que l'étymologie ne contient pas. Il y aurait, d'abord, la pensée de l'auteur, puis un support pour l'accueillir. Cette approche du document va à l'encontre de l'approche que nous développons, selon laquelle le sens de l'écriture émerge justement du milieu d'écriture, sans que le support ou la pensée soient des agents distincts.

Pourtant, si l'on suit la direction d'Otlet, Donker Duyvis, qui succède à Otlet au sein de la Fédération internationale de documentation, abonde dans la même direction et ajoute qu'il faut prêter une attention toute particulière à ce que le support, loin d'être neutre dans cette affaire, ne déforme pas la pensée de l'auteur :

> Un document est le dépositaire d'une pensée exprimée. Par conséquent, son contenu a un caractère spirituel. Le danger que l'unification brutale de la forme extérieure se répercute sur le contenu en le rendant sans caractère et impersonnel n'est pas illusoire.... En normalisant la forme et la disposition des documents, il est nécessaire de limiter cette activité à ce qui n'affecte pas le contenu spirituel et qui sert à éliminer une variété vraiment irrationnelle (Donker Duyvis traduit par (Voorhoeve 1964) et cité dans (Buckland 1997 : 806)).

La vision portée par Otlet et Donker Duyvis sur le document n'est pas du tout matérialiste et porte au contraire une vision du document comme une production de la pensée.

La recherche que réalise Buckland sur le document ne rapporte pas seulement les travaux d'Otlet et de ses successeurs. Il y fait également mention d'une autre approche, celle de Briet, bibliothécaire en France et largement impliquée dans les différents organismes en lien avec la documentation.

Dans son essai *Qu'est-ce que la documentation* publié en 1951, Suzanne Briet apporte, quant à elle, un autre regard sur le document. Briet semble en désaccord avec Otlet, qu'elle qualifie péjorativement de « mage », ainsi que son projet d'un Répertoire bibliographique universel qu'elle considère comme une « chimère [...] qui n'offrait pas un intérêt comparable au plus localisé des catalogues collectifs » (Briet 1951 : § 12).

La définition que donne Briet au document trouve ces racines dans l'étymologie latine (« enseignement ou preuve » [21]), qu'elle complète par une citation d'une « bibliographe soucieuse de clarté » : « un document est une preuve à l'appui d'un fait » (Briet 1951 : § 1).

21. Briet cite notamment les dictionnaires Richelet et le Littré pour construire son propos. Nous retrouvons ces deux objets en ligne aux adresses suivantes : `https://gallica.bnf.fr/ark:/12148/bpt6k58593308/f306.item` pour le Richelet, et `https://www.littre.org/definition/document` pour le Littré.

Pour préciser plus encore la définition du document, Briet mentionne la définition officielle donnée par l'Union française des organismes de documentation : « toute base de connaissance fixée matériellement et susceptible d'être utilisée pour consultation, étude ou preuve » (Briet 1951 : § 2).

En confrontant ces définitions au regard de la linguistique et de la philosophie, Briet en propose une plus abstraite : « tout indice concret ou symbolique, conservé ou enregistré, aux fins de représenter, de reconstituer ou de prouver un phénomène physique ou intellectuel » (Briet 1951 : § 3).

À partir de cette définition, Briet mobilise l'exemple, aujourd'hui très célèbre, de l'antilope. « Un animal vivant est-il un document ? Non », nous dit-elle (Briet 1951 : § 4). Du moins cela dépend du contexte dans lequel cet animal se trouve. Dans le cas d'une antilope, si cette dernière est dans son environnement naturel, il ne s'agit que d'une antilope. Par contre, si l'antilope est capturée et mise en cage dans un zoo, elle est alors cataloguée et devient un document vivant. Lorsque arrivera l'heure de son trépas, l'antilope sera alors empaillée pour conservation, puis exposée dans un musée. Elle deviendra alors un nouveau document. De plus, si des photographies de cette antilope sont prises, ou que des articles de presse ou savant sont rédigés, ou encore que des enregistrements sonores ou vidéos sont réalisés, ils formeront alors un ensemble de documents secondaires du principal qui est l'antilope (Briet 1951 : § 6). Le fait d'enfermer l'antilope dans une cage n'est alors pas suffisant pour la transformer en document : il est nécessaire qu'elle soit accompagnée « d'une production de discours » (Cotte 2004 : 35).

À la différence d'Otlet, le document de Briet ne correspond pas à l'impression de la volonté d'un auteur sur un support. Sa vision du document est profondément matérialiste, en ce sens qu'elle relève d'une preuve matérielle agencée dans une base de connaissances. La preuve fait alors partie d'un catalogue, d'où l'on peut extraire des informations depuis une unité de ce catalogue : le document.

Buckland s'intéresse également aux changements qu'apporte le numérique à la nature du document, notamment à partir de la conception du document proposée par Ranganathan :

> Mais ce ne sont pas des documents, parce qu'il ne s'agit pas d'enregistrements sur des matériaux aptes à être manipulés ou conservés. Les statues, les pièces de porcelaine et les objets exposés dans un musée ont été mentionnés parce qu'ils véhiculent une pensée exprimée d'une manière ou d'une autre. Mais aucun de ces objets n'est un document, puisqu'il ne s'agit pas d'un enregistrement sur une surface plus ou moins plane ((Ranganathan 1963) cité par (Buckland 1997 : 807)).

Cette conception du document manipulable sur une surface plane est très limitée, et ne permet pas d'y inclure beaucoup d'autres supports que ceux utilisés

plus communément pour l'écriture. Pourtant, l'informatique perturbe cette notion de document avec sa représentation du monde numérique, car les conditions matérielles évoquées par Ranganathan deviennent obsolètes devant la matérialité de l'infrastructure numérique.

Toutefois, nous considérons que le passage de l'imprimé au numérique ne se manifeste pas en tant que rupture (Ghitalla 2000) nette entre ces deux paradigmes, mais se manifeste plutôt en tant que continuité (Doueihi 2011). La transition s'effectue sur plusieurs décennies et la pratique du document papier reste présente dans les usages. D'ailleurs, comme nous le voyons dans la section III.2 du chapitre III, la production de documents numériques avec des logiciels de traitement de texte dans les années 1980 était orientée vers l'impression et l'obtention de « documents parfaits » (Kirschenbaum 2016 : 36). De plus l'engouement pour les interfaces graphiques et le mode WYSIWYG encouragent cette pratique puisque la page imprimée se retrouve affichée à l'écran sous une forme numérique.

Seulement, cette page affichée à l'écran n'est ni un objet pérenne, ni un objet manipulable. Elle n'est qu'une projection instable et éphémère de ce à quoi elle pourrait ressembler une fois imprimée, en dehors de son environnement numérique.

En 1994, David Levy traite de la variabilité des écritures numériques. Dans un article intitulé « Fixed or fluid ? », il critique la distinction faite entre un document papier *fixe*, étant compris comme un document stable et durable, et un document numérique (qui n'en serait pas un) fluide ou instable. Sa thèse consiste à considérer que tous les types de documents, et cela, sans considération pour le *medium*, peuvent être fixe ou variable. Il nous rappelle également qu'un document numérique a sa propre matérialité, sous la forme de bits dans le support de la mémoire numérique. Pour le dire autrement, Levy considère qu'un document n'est pas défini par l'état *fixe* du support tel qu'il caractérisait les documents pré-numériques, mais qu'au contraire, il est nécessaire de prendre en compte la matérialité du numérique dans l'établissement de ce dernier (Levy 1994).

Au début des années 2000, un réseau composé de plus de 150 scientifiques francophones, le RTP-DOC, a pris le nom collectif de Roger T. Pédauque et travaille pendant plusieurs années à fournir une théorie fondatrice du document numérique. Pour Pédauque, si le début du XXe siècle marque une période de *documentarisation* à travers le « rôle [joué par le] document-papier imprimé dans l'émergence des sociétés modernes » (Pédauque 2007 : 15), la globalisation de l'informatique et la mise en réseau des informations, Internet et ses applications comme les courriels, le Web, etc., marquent quant à elles une nouvelle documentarisation, ce que Pédauque nomme la *redocumentarisation* (Pédauque 2007 :

16). À partir de ce postulat, Pédauque travaille sur les mutations engendrées par ce document numérique, notamment celles de la figure qu'est l'auteur et de l'auctorialité qui lui est associée (Pédauque 2006).

Pour ce collectif, le document, en son sens le plus large, est étudié selon trois dimensions : la forme, le signe et le médium.

— Le *document comme forme* ; sous cette rubrique, nous rangerons les approches qui analysent le document comme un objet, matériel ou immatériel et qui en étudient la structure pour mieux l'analyser, l'utiliser ou le manipuler.

— Le *document comme signe* ; pour ces chercheurs le document est perçu avant tout comme porteur de sens et doté d'une intentionnalité ; ainsi le document est indissociable du sujet en contexte qui le construit ou le reconstruit et lui donne sens ; en même temps, il est pris dans un système documentaire ou un système de connaissances.

— Le *document comme médium* ; cette dimension enfin pose la question du statut du document dans les relations sociales ; le document est une trace, construite ou retrouvée, d'une communication qui s'est affranchie de l'espace et du temps ; en même temps, il est un élément de systèmes identitaires et un vecteur de pouvoir (Pédauque 2006 : 32).

À partir de ces trois dimensions, Pédauque définit ainsi le document au moyen d'une équation relativement générale (Pédauque 2006 : 36) :

$$document \quad traditionnel = support + inscription \tag{II.1}$$

Son évolution, le document numérique, est quant à elle définie par (Pédauque 2006 : 40) :

$$document \quad numérique = structure + données \tag{II.2}$$

La transition décrite par Pédauque du support vers la structure et de l'inscription à la donnée relève de ce glissement du fixe au fluide souligné par Levy en 1994 et tient des particularités de l'écriture numérique, notamment la variabilité. Cette caractéristique est la raison pour laquelle le document numérique est considéré comme fluide et instable, car son contenu peut théoriquement être modifié à l'infini – dans la limite accordée par les matériaux sur lesquels il est inscrit.

Cette inscription numérique est composée, selon Pédauque, de deux éléments principaux : une structure et des données. Si les données correspondent au

contenu du document, la structure est quant à elle plus complexe à définir. En effet, cette dernière fait référence à toutes les couches logicielles qui permettent d'accéder au contenu d'un document depuis le niveau le plus bas d'un ordinateur jusqu'à l'affichage à l'écran, ainsi qu'à la structure des informations à l'intérieur de celui-ci (Pédauque 2003). Cette structuration des contenus repose entre autres sur du balisage du texte, tel que SGML et ses différentes applications.

L'emploi du terme donnée, que l'on distingue du contenu, fait ici référence à l'élément le plus petit que l'on trouve à l'intérieur du contenu du document, c'est-à-dire les informations décrites par la structure. Ces données sont les éléments à partir desquels l'ordinateur va pouvoir effectuer des calculs. Autrement dit, si la structure d'un contenu est mal définie – dans sa globalité, pas seulement les balises – alors les données qu'il engendre ne sont pas calculables, ou alors font l'objet de calculs très pauvres ; pire encore, le résultat produit pourrait être corrompu et donner un sens complètement différent au document qui dès lors n'assurerait plus son rôle de preuve (Bachimont 2022*a*).

La donnée, en tant qu'unité de calcul, appartient plus à la dimension calculatoire du document numérique qu'à la dimension herméneutique (Bachimont 2017 ; 2022*b*). Du côté du calcul, nous l'avons vu précédemment avec la machine de Turing, l'ordinateur ne se préoccupe pas du sens des données qu'il traite, ni du sens qu'il produit grâce à elle. Pour autant, cela ne veut pas dire que le document n'a aucun sens, mais qu'il n'en a pas *a priori*. Le document et le sens qu'il transporte sont des productions issues de ces calculs (Cotte 2004). En ce sens, le document numérique est composé d'un ensemble de données, que l'on structure au moyen de normes d'encodage, de formats, de logiciels pour les interpréter et les visualiser, etc.

Bachimont propose de considérer que le numérique apporte ainsi un nouveau régime documentaire qu'il résume selon quatre niveaux :

— Au premier niveau, celui du code, le numérique s'interprète de manière formelle en algorithme, modèles mathématiques et logiques, où la manipulation binaire aveugle se comprend comme un calcul et une preuve opérant sur une logique symbolique. C'est le niveau des théoriciens du calcul, de l'informatique théorique, et de l'algorithmique.

— Au second niveau, celui des formats, le code s'interprète selon des conventions permettant de le mettre en correspondance avec des domaines du réel : le binaire correspondra à des nombres, à des caractères, à des pixels, à des points d'impression, etc. A ce niveau, technique et logique, le code n'est plus seulement une réalité mathématique et formelle permettant d'interpréter

> ce qui se passe dans les registres binaires, mais devient une manipulation du réel où l'on voit des écrans afficher des contenus, des textes se manipuler, des sons se produire.
>
> — Au troisième niveau, celui des applications, les formats se mobilisent dans des systèmes logiciels s'intégrant dans des logiques métiers, des usages, des pratiques. Un système est conçu pour permettre un usage.
>
> [...]
>
> A ces trois niveaux, il convient d'en ajouter un, ou de le penser de manière sous-jacente, c'est celui de la concrétisation matérielle et physique de la machine informatique, qui définira une architecture, un code machine natif (un microprocesseur), qui sera la base sous-tendant toutes les autres opérations (Bachimont 2022*a* : 16-7).

Chaque étape du processus de structuration des contenus permet ainsi de définir les données et les calculs que l'on peut réaliser sur ces dernières. Si la norme d'encodage correspond au niveau le plus bas, c'est-à-dire le niveau le plus éloigné du contenu où l'information est déjà formatée[22] ; le balisage du texte et les modèles textuels que ces balises véhiculent peuvent être associés aux couches les plus proches de l'utilisateur. Ces couches proches de l'humain permettent de manipuler et structurer les éléments textuels et les informations de manière très précise, autrement dit de formater ces dernières.

À l'intérieur d'un environnement d'écriture numérique, le modèle textuel devient un intermédiaire supplémentaire entre l'humain et la machine, il se situe au même niveau que les formats et les protocoles de communication. Les choix de modélisation des éléments textuels influencent non seulement la structure des documents et leur lisibilité, mais aussi les possibilités d'interaction, d'analyse automatisée et de traitement computationnel.

En ce sens, un modèle textuel peut ainsi imposer des contraintes formelles sur la manière dont le contenu est saisi, organisé et rendu exploitable par les systèmes informatiques, tout en ouvrant la voie à de nouvelles formes expressives et à l'émergence de pratiques scripturales, comme l'écriture collaborative et l'hypertextualité. De cette façon, la conception du modèle textuel participe pleinement à la constitution de l'environnement d'écriture, en conditionnant les usages, les représentations et l'expérience de l'utilisateur.

22. On distingue l'action de formater et d'encoder. Le formatage correspond à la structuration des informations selon un format, tandis que l'encodage correspond à la transformation des caractères en chaîne binaire selon une norme d'encodage.

Les modèles textuels

Dans le cadre de cette recherche, le *modèle textuel* est mobilisé dans le sens que DeRose *et al.* leur confèrent en (1990).

Les modèles textuels s'appuient sur le principe OHCO que décrivent DeRose *et al.* dans leur article *What is text, really ?*. OHCO est le principe de structuration des éléments d'un texte au moyen de balises descriptives infratextuelles. Ainsi, les balises permettent de décrire formellement et explicitement le contenu d'un document.

Pour les auteurs de cet article, un ensemble de balises, ou plus largement un ensemble de moyens de description formelle et explicite des informations contenues dans un texte, devient un modèle textuel. Par l'exploration de ce modèle, il devient dès lors possible de connaître la signification des concepts mobilisés dans la structuration des contenus et d'identifier à quelle sémantique et rhétorique renvoie le modèle étudié.

DeRose *et al.* se réfèrent à un modèle OHCO pour illustrer ce principe : il s'agit du standard SGML, à la fois un langage de balisage pour décrire des éléments dans un texte, et à la fois un métalangage permettant de créer son propre langage de balisage. Des langages comme HTML ou encore XML sont des exemples d'application de SGML.

SGML est le successeur de *Generalized Markup Language* (GML), un système de balisage généralisé développé par Charles Goldfarb et son équipe dans les années 1960 pour le compte de l'entreprise IBM dont la première apparition publique date de 1973 (Goldfarb 1996). Pour l'anecdote, les trois lettres de l'acronyme représentent à la fois le nom du langage, mais également les initiales de ses concepteurs : Charles Goldfarb, Ed Mosher et Ray Lorie (Goldfarb 1997). L'intérêt de GML est de produire un système qui permette à la fois d'indexer des informations et de composer des pages destinées à l'impression. GML avait pour ambition de dépasser ce que proposait le langage de composition de page *Script*, orienté pour l'impression, qui définissait de façon assez pauvre les éléments d'un texte.

Un fait marginal mérite d'être signalé : Goldfarb est juriste de formation, tout comme l'était Henri La Fontaine quelques décennies auparavant. Or, il se trouve que La Fontaine a participé à l'élaboration de la Classification Décimale Universelle (CDU) aux côtés de Paul Otlet à la fin du XIXe siècle. Il s'avère particulièrement intéressant de constater que les systèmes de classification et d'indexation des informations et des documents, devenus aujourd'hui des normes internationales, trouvent leurs racines dans le droit, et certainement dans l'écriture des textes législatifs.

L'idée de baliser le texte que Goldfarb s'emploie à développer sous la forme de GML est tirée des travaux de Stanley Rice, un designer graphique qui a travaillé sur la structure éditoriale des textes dès les années 1960. Nous retrouvons des traces des travaux de Rice dans plusieurs textes fondateurs des langages de balisages, que ce soit chez Goldfarb ou encore chez DeRose (1999) sans toutefois trouver les textes qui y sont cités. Néanmoins, d'autres textes ultérieurs à la publication de GML nous donnent à voir le regard que Rice portait sur l'édition à cette période (Rice 1980), notamment sur les structures éditoriales des textes qu'il employait (Rice 1978).

> Rice a proposé un langage de structure de texte de haut niveau qui décrirait les parties du texte à l'aide de codes mnémoniques (10). Le programme de composition identifierait les codes comme des appels à des formats stockés ; le programme de recherche les utiliserait pour la classification.
>
> *Stanley Rice, alors designer de livres à New York et aujourd'hui consultant en édition en Californie, est à l'origine de mon inspiration pour le GML. La première phrase de la phrase précédente était son idée, la seconde était la mienne* (Goldfarb 1997 : 7).

L'approche de Rice pour qualifier les éléments textuels est *visuelle* (Rice 1978 : vii). À partir d'un regard expert sur le texte, un designer ou un typographe peut qualifier des éléments du texte selon ce qu'ils sont censés modéliser : une note de bas de page, une liste ordonnée ou non ordonnée, un titre ou un paragraphe, etc. Dans cette méthode, Rice recommande l'emploi de balises (Rice 1978 : viii) pour qualifier les éléments textuels. Par exemple, des listes à puces dans le corps de texte seront annotées avec la balise LT (Rice 1978 : 25). L'inspiration de la modélisation des textes au moyen de langages de balisage provient donc de la composition typographique et de l'univers du livre imprimé[23].

Rice pense non seulement une méthode d'annotation des contenus à l'aide de balises pour modéliser le texte à mettre en page, mais il ajoute à celle-ci des spécifications pour permettre aux dactylographes et aux éditeurs d'utiliser le même modèle textuel pour la composition d'un ouvrage (Rice 1978 : 199). Cette conception d'un document comprenant les spécifications d'un modèle textuel s'apparente au concept de *Document Type Definition* (DTD) implémenté dans GML et SGML. Bien que cet ouvrage soit postérieur à la première publication de GML, nous envisageons la possibilité que les modèles textuels énoncés par Rice lui soit antérieur, comme le stipule Goldfarb (1996) quelques années plus tard.

23. Ce fait est également valide pour le travail sur TEX réalisé par Knuth.

Le concept de DTD naît en 1971 (Goldfarb 1996). C'est Edward Mosher qui le développe pour la production des manuels relatifs à la collection intitulée *Telecommunications Access Method* d'IBM. La DTD est une spécification qui contient toutes les déclarations des balises que l'on peut rencontrer dans un document dont le texte est décrit selon cette dernière. Cette spécification peut également être utilisée pour valider un document, c'est-à-dire vérifier qu'il est bien formé et conforme à la spécification. C'est un concept qui est largement repris pour les développements d'HTML, dont les spécifications se trouvent sur le site web du W3C[24] et pour XML où chaque document fait référence à un schéma XML, un modèle textuel particulier déclaré en en-tête de document, par exemple dans la balise `<!DOCTYPE>` des documents au format HTML.

Cette notion de DTD couplée à un balisage du texte normalisé font le succès de SGML qui devient une norme internationale reconnue par l'*International Organization for Standardization* (ISO) en 1986. Le domaine d'application de SGML s'étend alors à tous types de « documents traités par tout éditeur ou traitement de texte » (ISO 8879:1986 1986 : 5). SGML est utilisé par des acteurs de tous les milieux, que ce soit « l'industrie de l'aéronautique et les compagnies aériennes (Boeing, Airbus, Rolls Royce, Lufthansa, etc.) » ; « l'industrie pharmaceutique » ; le milieu de l'édition et du livre avec « l'association des éditeurs allemands (*Börsenverein des Deutschen Buchhandels*) », la « *British Library* », « le Syndicat national de l'édition et le Cercle de la librairie », « l'Office des Publications de la Communauté Européenne (FORMEX) », la « *Text Encoding Initiative* », « la plupart des manuels techniques, avec p. ex. DTD DocBook ou autres, utilisées par IBM, HP, OSF, O'Reilly, etc. », des « agences de presse », « l'Office des Publications de l'ISO (Genève) et le HMSO pour les brevets (Angleterre) » ; mais aussi par des entreprises de services informatiques pour « les modules sgml entrée/sortie sur des systèmes de traitement de texte ou de base de données (Frame, Interleaf, Microsoft, Oracle, Wordperfect) » (Goossens 1995 : 29).

L'engouement pour cette technologie tient du fait qu'avec SGML, les documents produits sont d'une meilleure qualité, avec des données indexées dans les documents via des déclarations formelles (les balises) (Chaudiron 2007) que l'on peut ensuite parser et réutiliser dans d'autres applications tel que c'est le cas de l'hypertexte avec le Web, ou encore des bases de données (DeRose et al. 1990 ; Goossens 1995) .

Avec SGML, c'est une multitude de modèles textuels qui sont créés, comme en témoignent les développements du langage de balisage HTML et du Web ou encore du format XML. La création de documents via un langage normalisé

24. Voir par exemple la DTD d'XHTML1 à cette adresse : `https://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd`, page consultée le 15 avril 2025.

n'irrigue pas seulement l'industrie et le monde des entreprises, mais touche également la sphère scientifique, qui s'empare également de cette technologie pour développer ses propres modèles textuels.

Comme le soulignent Coombs *et al* (1987) , les possibilités de modélisation du texte sur ordinateur au début des années 1980 sont relativement pauvres comparés aux besoins de la recherche : insertion de notes de bas de page, de références bibliographiques ou d'appareils critiques, etc. Les premiers ordinateurs personnels paraissent même moins bien dotés que les machines à écrire. L'arrivée des langages de balisage ouvre alors de nouvelles perspectives en termes de modèle textuel et d'écriture spécialisée pour la recherche en milieu numérique.

Tandis que le modèle proposé par le W3C à travers le langage HTML montre des limites du fait de l'imposition d'un ensemble fini de balises, et de l'impossibilité d'ajout de nouvelles balises dans cet ensemble, un autre format, le XML, est alors publié par ce même consortium en 1998[25] et tente de réunir les avantages connus de SGML et de HTML (DeRose 1999). Entre autres, XML combine la syntaxe plus légère du HTML et la capacité à créer ses propres jeux de balises, soit à partir d'une DTD, soit à partir d'un schéma XML[26]. Pour autant, le modèle textuel associé à HTML est loin d'être abandonné puisque, comme nous le verrons dans le chapitre IV, le Web est un environnement particulièrement intéressant pour publier la recherche et rendre les publications accessibles aux publics. Il s'agit d'un moyen de diffusion incontournable des écritures numériques. Aussi, le modèle textuel lié au format HTML se retrouve étendu à partir du début des années 2000 par des langages de balisages légers censés permettre une écriture plus fluide et moins verbeuse (destinée au Web), comme c'est le cas avec le format Markdown.

La modélisation du texte selon le principe OHCO rencontre cependant des limites, nous disent DeRose et Pierazzo, notamment lorsqu'il s'agit de modéliser certains éléments dont la structure n'est pas en adéquation avec l'application des balises dans le texte (DeRose 1999 ; Pierazzo 2016 : 62).

XML illustre parfaitement ce propos, et la *Text Encoding Initiative* (TEI) encore plus. Si l'on est capable de définir des éléments structurels pouvant rendre compte de la plupart des éléments d'un texte[27], le modèle est pourtant contingenté

25. Voir la page web de la publication des spécifications d'XML : `https://www.w3.org/TR/1998/REC-xml-19980210`, page consultée le 16 avril 2025.

26. Que l'on utilise un schéma XML ou une DTD, le principe reste le même : ces deux objets permettent de vérifier que le document soit *bien formé* et surtout *valide*, c'est-à-dire conforme à ce que les règles de structuration présentes dans ces documents stipulent.

27. Pierazzo ne serait pas d'accord avec ce point particulier, notamment dans le cadre d'édition diplomatique où le modèle textuel ne rend pas forcément compte de la matérialité du document original (Pierazzo 2016 : 52 & 63). Selon l'approche de Pierazzo, le document original est matériel tandis que le document numérique relève plutôt de l'immatériel. En choisissant une approche

à l'emboîtement des balises entre elles. En effet, le principe même du modèle OHCO repose sur une hiérarchie stricte des éléments contenus les uns dans les autres. Une balise ouvrante doit avoir une balise fermante correspondante et les balises doivent s'imbriquer sans chevauchement.

Par exemple, dans un livre, un paragraphe est un sous-élément d'un chapitre : ainsi, le paragraphe ne peut pas être à cheval entre deux chapitres. Cette phrase peut être traduite par des balises fictives pour illustrer l'exemple :

Extrait de code II.1 : Exemple d'imbrication de balises OHCO

```
<chapitre>
  <paragraphe>
    Ceci est un paragraphe
  </paragraphe>
</chapitre>
```

Cette structuration représente en un sens la structuration telle que nous la connaissons dans le livre imprimé. La règle de non-chevauchement des balises semble logique dans cette situation, puisque le paragraphe ne peut pas sortir de l'élément hiérarchique supérieur.

À notre connaissance, aucun texte imprimé ou numérique (selon le modèle OHCO) n'est structuré comme dans le bloc II.2 :

Extrait de code II.2 : Exemple de mauvaise imbrication de balises OHCO

```
<chapitre>
  <paragraphe>
    Ceci est le début d'un paragraphe.
 </chapitre>
    Ceci est la fin du paragraphe.
  </paragraphe>
```

Cette approche est représentée par des balises ouvrantes et fermantes délimitant les éléments à structurer dans le texte, avec pour règle d'écriture le fait qu'une balise ouverte à l'intérieur d'un autre élément ne peut pas être fermée à l'extérieur de celui-ci. De cette manière, le paragraphe ne peut pas débuter dans un chapitre et finir dans le suivant. Si cette règle paraît évidente pour le paragraphe et le chapitre, elle ne l'est plus forcément lorsqu'il s'agit de structurer d'autres éléments tels que les figures de style dans un poème composé en vers.

La modélisation d'un tel texte nécessiterait de baliser chaque vers du poème. Néanmoins, si l'on décide de baliser les figures de style contenues dans ce poème, nous pourrions rencontrer une limite du modèle OHCO : la disposition d'une figure de style dans un poème n'est pas limitée à un seul vers, elle peut s'étendre sur plusieurs vers. Or, puisque le modèle OHCO est établi sur cette hiérarchie

matérialiste, nous considérons plutôt que le document numérique et le document original ont deux matérialités distinctes.

stricte entre les éléments, il n'est pas possible de structurer les figures de style débordant sur plusieurs vers d'un poème en une seule balise, parce qu'il n'y a pas de relation hiérarchique entre le vers et la figure de style. La figure de style n'existe donc pas dans ce cas de figure particulier, car le modèle ne peut pas la représenter formellement et explicitement puisque cet élément structurel n'est pas défini.

Le modèle textuel OHCO, comme toutes les modélisations, présente des limites et ne permet pas de modéliser tous les textes, c'est-à-dire qu'il ne permet pas de structurer et donc de donner du sens à tous les éléments que l'on peut rencontrer dans un texte. Cependant, en raison de son caractère ubiquitaire dans les environnements d'écriture examinés dans les chapitres suivants, notre analyse se concentre sur ce type de modèle textuel présentant une hiérarchie d'objets structurés, sans considération pour les autres types de modèles textuels[28].

II.5 Conclusion

Tandis que le premier chapitre de cette thèse traite de l'écriture en son sens le plus large, le chapitre II est quant à lui orienté sur l'écriture numérique en milieu informatique. Dans le chapitre I, nous avons démontré que les médiations à l'œuvre dans l'écriture, comprise en tant que dispositif technique, produisent le sens octroyé à ce qui est écrit. En poursuivant dans la même direction, ce deuxième chapitre concerne les médiations propres à l'environnement numérique et au sens qu'elles produisent.

La traversée des différentes couches qui assurent le bon fonctionnement des environnements d'écriture numérique nous a permis de constater qu'elles sont toutes indispensables. Il n'est donc pas possible d'en exclure certaines ni de se focaliser sur une seule pour comprendre pleinement les médiations qui s'y déroulent. Les environnements d'écriture numérique sont définis conjointement par toutes les couches abordées dans ce chapitre, depuis les différents composants du *hardware*, comme le processeur, et les diverses couches d'abstraction matérielle jusqu'au logiciel qui affiche l'interface graphique à l'écran. Un environnement d'écriture numérique n'est pas réductible à la couche la plus proche de l'utilisateur, souvent le logiciel ou l'application employée, car en l'absence des autres éléments d'un environnement d'écriture numérique, cette couche la plus haute ne saurait ni fonctionner ni permettre l'écriture. De plus, l'utilisateur, généralement abordé comme élément externe à l'environnement d'écriture, en est également un élément

28. Parmi les autres types de modèles textuels, on retrouve par exemple les modèles probabilistes utilisés par les algorithmes d'intelligence artificielle qui, quant à eux, ne sont pas créés à partir de théories du texte, mais à partir de *patterns* dégagés de la masse de données traitées pour les créer (Bachimont 2022*c*).

indispensable, puisqu'il est responsable de plusieurs conjonctures médiatrices qui s'y déroulent. L'action de l'utilisateur ne se réduit pas non plus à l'appui sur des touches, il opère aussi une série de choix, notamment ceux concernant la sélection des éléments technologiques présents dans son environnement d'écriture. Dans une perspective non-anthropocentrée et distribuée de l'écriture, cette dissociation entre l'utilisateur et l'environnement d'écriture numérique n'est donc pas pertinente. L'environnement d'écriture numérique inclut à la fois les éléments techniques de l'ordinateur, la dimension humaine qui participe au processus ainsi que les interactions entre les deux. Toutefois, malgré cette présence de l'humain dans toutes les strates du dispositif, nous choisissons ici d'en faire abstraction pour concentrer notre attention sur les autres médiations qui se déploient au sein de l'environnement d'écriture. Dans cette perspective, le document numérique occupe une place privilégiée, car il constitue un point d'observation central de ces différentes médiations.

Le document numérique tient une place particulière dans l'environnement d'écriture numérique : il y est à la fois positionné en entrée et en sortie. Lorsqu'il est à l'intérieur de l'environnement d'écriture, le document est décomposé en éléments calculables, qu'ils soient textuels ou structurels, grâce auxquels il devient possible d'écrire. Ensuite, l'environnement d'écriture recompose le document numérique au moment de l'export lorsqu'il le réinsère dans une collection ou autre forme d'organisation des connaissances.

À l'intérieur de l'environnement d'écriture, le document décomposé n'est alors qu'une suite d'unités calculables se positionnant au gré des médiations dans l'environnement au sein d'une architecture de l'information. La page ainsi affichée à l'écran de certains environnements d'écriture, notamment dans les logiciels WYSIWYG comme nous le verrons dans le prochain chapitre, est ambiguë et trompeuse quant à ce sujet. Cette fausse page laisse sous-entendre qu'il y a bien un document dans l'environnement d'écriture numérique qui serait homothétique de son pendant imprimé. Par sa présence, la page invisibilise les médiations dans l'environnement d'écriture numérique qui matérialisent le document numérique.

En écartant la page affichée dans l'interface à l'écran, puisque nous considérons qu'il s'agit là d'un simulacre voué à rassurer les usagers par la reproduction d'un environnement d'écriture connu, nous avons décidé de focaliser notre attention sur les modèles textuels présents dans les environnements d'écriture.

Les modèles textuels incarnent des théories du texte et définissent les éléments textuels mobilisables dans un document. En suivant la définition de DeRose *et al.* (1990), le texte est structuré par une hiérarchie d'objets, c'est-à-dire une succession d'objets parents et enfants qui permet d'imbriquer tous les éléments d'un texte en suivant une architecture verticale. Cette modélisation du texte

correspond à ce que Goldfarb a implémenté dans les langages de balisages GML et SGML et que l'on retrouve par la suite dans les successeurs de SGML : d'abord HTML, puis XML.

Pour chacun de ces formats, la structure utilisée pour écrire se traduit par un ensemble de règles, que l'on nomme DTD ou schéma selon le format, qui explicitent formellement les éléments structurels que l'on peut employer, leur place hiérarchique dans le modèle, ainsi que leur signification. Grâce à ce principe, il est d'abord possible de vérifier si un document respecte le modèle textuel auquel il fait référence. Il est également possible de faire circuler le document créé auprès d'autres acteurs – machines ou humains – qui n'auront pas de problème pour l'interpréter, puisque le modèle a pour rôle de lever toute ambiguïté quant au sens des éléments textuels présents dans le document. Ce fonctionnement permet au Web, par exemple, de fonctionner, et aux différents navigateurs d'interpréter le contenu des documents HTML (en considérant qu'il peut y avoir de légères variations d'un navigateur à un autre).

Notre intérêt se porte alors sur ces modèles textuels parce qu'ils se trouvent être présents dans la majorité des environnements d'écriture numérique que nous utilisons quotidiennement, comme nous le voyons dans les chapitres III, IV, V. Même si elle est parfois cachée par l'interface graphique, il s'agit d'une technologie incontournable des pratiques d'écriture contemporaine.

Dans le cadre de l'écriture numérique, la modélisation du texte constitue un élément fondamental, car elle induit une forme de calcul des éléments textuels contenus dans un document numérique, puisque ces calculs participent à la production du sens d'un texte. C'est pourquoi il apparaît essentiel de les observer. En ce sens, dans les prochains chapitres, nous étudions et comparons des modèles textuels variés afin de montrer les différents sens qu'ils peuvent produire.

Chapitre III

Écrire avec des logiciels *What You See Is What You Get*

III.1 Introduction

Lors des deux premiers chapitres de cette thèse, nous avons abordé l'écriture en son sens le plus large et montré que l'écriture, en tant que phénomène distribué producteur de sens, est avant tout une affaire de médiations (Vitali-Rosati et Larrue 2019). L'écriture est ainsi distribuée à la fois dans toute une infrastructure sociale (Denis et Pontille 2012), mais également dans toute une infrastructure technique (Paveau 2015), puisque c'est grâce aux protocoles de communication (Galloway 2004), aux formats (Mourat 2020) et à toute une infrastructure matérielle que les informations circulent dans un dispositif de communication (McLuhan 1967 ; Shannon 1948).

Dans cette thèse, nous focalisons notre attention sur la dimension sociotechnique de ces médiations, à l'instar de la proposition d'Hutchins dans sa version distribuée de la cognition que nous avons abordée dans le premier chapitre (Hutchins 1995).

Nous nous sommes ensuite intéressés aux couches les plus basses de l'infrastructure numérique avec la partie matérielle comme point de départ, ainsi qu'à la modélisation du texte dans les couches les plus hautes du numérique. Cet aperçu de l'histoire du *hardware* et de son fonctionnement nous a permis de montrer que l'abstraction nécessaire à l'interopérabilité des modes d'écriture en environnement numérique n'est pas un fait acquis depuis les débuts de l'informatique. Au contraire, plusieurs décennies de progrès et de perfectionnements ont été nécessaires pour atteindre cette forme presque agnostique de l'écriture vis-à-vis du matériel, un fait frappant qui démultiplie les possibilités de communication en faisant abstraction du *hardware*, comme si sa dimension matérielle devenait insignifiante. L'essor des *software* dans les années 1970 et 1980, suite à la miniaturisation des composants électroniques des ordinateurs,

notamment les micro-processeurs, engendre un empilement de couches entre la partie matérielle et l'utilisateur. En s'éloignant de plus en plus de la machine et de son langage binaire, chaque couche occasionne un niveau supplémentaire d'abstraction matérielle jusqu'au point où l'on ne considère plus le choix d'un matériel comme relativement important dans l'exécution d'un programme.

Un élément capital commun à toutes ces couches est le formatage des informations : quel que soit le niveau où l'on se trouve entre l'humain et la machine, les informations doivent être formatées pour être interprétées, de la sémantique formelle dans un format à l'encodage en ASCII ou Unicode pour devenir calculable. Que l'on soit proche de l'utilisateur ou de la machine, il est nécessaire que ces deux parties puissent comprendre l'information qui leur est donnée à lire[1]. Si aux deux extrêmes de ce dispositif, nous retrouvons le langage naturel et le langage machine, l'ajout de médiations au moyen des logiciels permet d'explorer des étapes entre ces deux pôles et donne lieu à des documents à la fois traitables par la machine et compréhensibles par l'être humain.

Afin de rendre cela possible, il est nécessaire de désambiguïser le langage naturel pour décrire formellement et explicitement un élément structurel d'un document. C'est à cet endroit qu'interviennent les normes et les formats, notamment dans le cas de l'écriture. Cet espace entre le langage naturel et le langage machine où le texte transite d'une forme graphique (affichée à l'écran, que le texte soit stylé ou structuré avec des balises) à un formatage, et ceci jusqu'à l'étape d'encodage, relève du champ herméneutique, c'est-à-dire d'un espace où les caractères ont encore une signification. Au-delà de l'encodage, lorsque les caractères deviennent des chiffres, ils quittent cette dimension herméneutique pour celle du calcul et perdent leur sens (Bachimont 2022*b* ; Turing 1936). La dimension herméneutique dans sa phase formelle devient d'une importance capitale, car une granularité plus fine dans la définition des informations permet d'effectuer une mise en données plus précise des documents, ce qui ouvre la voie à d'autres calculs et à de nouveaux sens produits lorsque les résultats de ces calculs, par la suite, se matérialisent en documents interprétables par les utilisateurs[2] (Bachimont 2017).

Sans cette couche herméneutique formatée, comment définissons-nous un titre ? De quel niveau s'agit-il ? Un titre en est-il un parce que la taille de la police est plus élevée par rapport à celle d'un paragraphe ? La structuration des

1. Comme nous l'avons vu dans le deuxième chapitre, l'emploi du terme lecture est à double sens. Ce mot désigne les actions de lire et d'interpréter dont les êtres humains sont capables, tandis que la lecture du côté des ordinateurs relève plus de l'action de *scan* d'une tête de lecture qui lit une donnée pour effectuer des calculs, comme le désignait Turing dans sa machine (Turing 1936).

2. La phase de calcul correspond à l'état binaire de l'information, illisible pour les êtres humains.

contenus dans ces couches intermédiaires apparaît alors comme un enjeu crucial, et cela, dès la fin des années 1970 avec les premiers logiciels de traitement de texte tel qu'*Electric Pencil*.

Les premiers logiciels de traitement de texte comportaient leur propre format et leur propre syntaxe de structuration des contenus. Ce faisant, les documents ne pouvaient pas être portés d'un logiciel à un autre, voire d'une version d'un logiciel à une autre. La structuration des contenus fera l'objet de plusieurs spécifications et de formats différents depuis cette période et jusqu'à nos jours, donnant ainsi naissance à des standards comme SGML, HTML ou XML. Chaque format est adossé à un ensemble de règles dont la finalité est d'expliciter le comportement des éléments structurels d'un document [3]. En un sens, il apparaît alors fondamental pour chaque environnement d'écriture numérique de se doter de la capacité de traiter une ou plusieurs modélisations du texte, phénomène que nous abordons dans ce chapitre de la thèse et les suivants.

Notre intérêt se porte ainsi sur les couches les plus hautes des environnements d'écritures numériques, celles au plus proche de l'utilisateur. Puisqu'il est difficile d'être exhaustif dans une analyse de tous les environnements d'écriture numérique et de toutes les modélisations du texte possibles – ils étaient quelque 300 environnements rien qu'au début des années 1980 (Bergin 2006*a*) –, nous nous focalisons sur un nombre très restreint de ces environnements pour n'évoquer que les plus pertinents dans le cadre de cette étude.

Cette étude est caractérisée par une double comparaison, d'abord une comparaison entre les représentations graphiques et formelles des documents, puis une comparaison entre deux environnements d'écriture, MSWord et LibreOffice. La comparaison entre MSWord et LibreOffice est réalisée selon une méthode similaire à celle menée par Roig et Ribera (2021) sur la comparaison entre les normes *Office Open XML* (OOXML) et *Open Document Format* (ODF) qui correspondent aux modèles textuels mobilisés dans ces environnements d'écriture [4]. Nous ne pouvons cependant pas étudier tout l'historique des modèles textuels de ces environnements puisqu'une partie de ces éléments ont été ou sont encore propriétaires.

3. Nous n'évoquons ici que les formats liés à du texte, mais tous les objets qui peuvent être numérisés font l'objet d'un encodage dans des formats.

4. Il ressort de l'étude menée par Roig et Ribera un manque de clarté dans la norme de Microsoft, à la fois en termes de structuration des informations qu'en termes de technologies mobilisées pour décrire certains éléments complexes tels que les graphiques, tandis que la norme ODF mobilise des standards partagés par d'autres environnements. Par exemple, pour insérer des graphiques dans un document, ODF s'appuie sur le format standard *Scalable Vector Graphics* (SVG), alors que Microsoft a intégré sa propre technologie *DrawingML* pour réaliser la même chose.

Ce que nous désignons comme des représentations graphiques et formelles des documents correspond tout d'abord à la forme du texte tel qu'il est affiché dans l'interface de l'environnement d'écriture (représentation graphique), puis au texte tel qu'il est formaté en texte brut et dont les éléments apparaissent comme formellement structurés (représentation formelle). Si toutefois ces deux modes de représentation sont graphiques, au sens où ils sont affichés à l'écran et reposent sur la mobilisation de la vue pour les interpréter, nous qualifions les représentations de graphique et formelle pour bien distinguer ces deux facettes d'un même document.

Afin de représenter un des modes d'écriture numérique les plus répandus, nous nous focalisons sur deux environnements *What You See Is What You Get* (WYSIWYG) encore très prisés, il s'agit du logiciel propriétaire Microsoft Word (MSWord) et de son pendant libre, d'abord OpenOffice, devenu ensuite LibreOffice au début des années 2010. Conformément à notre découpage de la chaîne éditoriale réalisé en introduction , ces deux logiciels font partie de la phase de production de l'écriture.

Un environnement WYSIWYG se distingue d'un environnement *What You See Is What You Mean* (WYSIWYM). Dans un environnement WYSIWYG, les informations sont affichées à l'écran de manière à prévisualiser le rendu final tel qu'il apparaîtra à l'impression, grâce à l'application de styles graphiques facilitant cette visualisation. En revanche, dans les environnements WYSIWYM, comme nous le verrons au chapitre IV, le texte est présenté dans son apparence la plus brute : le contenu y est décorrélé de sa mise en forme graphique.

Historiquement, MSWord est le logiciel de traitement de texte avec le plus grand monopole sur l'écriture numérique, détenant plus de 90% des parts du marché à partir de 1994 et cela pendant un peu plus d'une décennie (Kirschenbaum 2016 : 235). Ce logiciel est également particulièrement intéressant pour notre étude, car il a traversé la quasi-totalité de l'histoire des logiciels de traitement de texte depuis sa mise sur le marché en 1983 et illustre les différentes problématiques que ces logiciels ont rencontrées tout au long de leurs développements (Bergin 2006*a*). D'un autre côté, OpenOffice devient un concurrent sérieux à MSWord à partir de 2005 lorsque le schéma qu'ils utilisent pour décrire la structure des informations contenues dans les documents produits par leurs logiciels devient une norme internationale interopérable (Yami et al. 2015).

Ces logiciels, similaires en apparence, car le principe d'écriture repose dans les deux cas sur l'application de styles graphiques aux contenus du document, s'appuient sur des modèles textuels différents pour structurer les informations que contiennent les documents. Le modèle textuel de MSWord est *WordprocessingML*, une application de la norme OOXML développée par Microsoft, tandis que le

modèle textuel d'OpenOffice est ODF, une autre norme dédiée au traitement de texte, développée par l'entreprise Sun Microsystems au début des années 2000.

Bien que les technologies de construction des documents diffèrent d'un dispositif à l'autre, le sens produit par ces deux environnements est-il identique ?

Pour répondre à cette problématique, ce chapitre se décompose en trois sections. La première analyse l'histoire des logiciels de traitement de texte, parmi lesquels MSWord et LibreOffice s'inscrivent. Cet examen historique est fondamental pour appréhender les choix de conception de ces environnements d'écriture numérique, dans la mesure où les exigences et contraintes associées à l'écriture numérique dans les années 1980 diffèrent de celles d'aujourd'hui. Ces deux logiciels conservent cet héritage dans leurs versions contemporaines, les premières ayant été publiées respectivement en 1983 et 1985.

La deuxième section se concentre sur un aspect particulier de l'histoire des logiciels de traitement de texte : l'intégration du format XML au sein des formats DOCX et ODT pour décrire les éléments textuels présents dans ces logiciels. Cette évolution constitue une étape majeure de leur histoire, car les modèles textuels sous-jacents à ces formats deviennent des normes internationales utilisables pour la conception de n'importe quel logiciel de traitement de texte. Ces modèles ont pour fonction de définir formellement les éléments textuels que l'on peut saisir dans ces environnements. Ce jalon historique est essentiel pour comprendre la manière dont le sens est produit dans ces environnements d'écriture numérique.

Enfin, dans la dernière section, nous menons une étude comparative des chapitres de l'ouvrage collectif *Contribution numérique : cultures et savoir*, rédigés dans les environnements MSWord et LibreOffice par les auteurs. En analysant ces documents sous leur représentation XML, c'est-à-dire selon une représentation formelle des informations, nous étudions les contenus tels qu'ils ont été produits par les différents environnements d'écriture. Cette lecture permet d'accéder à la fois aux textes et aux éléments structurels indispensables à l'établissement des documents. Pour en saisir la signification, nous nous appuyons sur les documentations des normes associées à ces environnements, *Office Open XML* pour MSWord et *Open Document Format*, et sur les définitions qu'elles contiennent.

III.2 Histoire du traitement de texte

La compréhension des choix de modélisation du texte dans un logiciel de traitement de texte nécessite de rappeler l'histoire des logiciels étudiés, mais aussi celle du traitement de texte en général. Alors que l'histoire de MSWord s'inscrit dans la course aux développements des logiciels de traitement de texte que l'on observe tout au long des années 1980, celle d'OpenOffice commence

bien plus tardivement, en l'an 2000, lorsque Sun Microsystems, l'entreprise qui détient les droits de ce logiciel, ouvre l'accès au code source du logiciel, un an après l'acquisition de la suite logicielle StarOffice en 1999, un projet propriétaire initialement développé par l'entreprise Star Division en 1985.

Malgré le fait qu'aujourd'hui nous ne remettons plus en cause la suprématie de ce logiciel, MSWord n'apparaît pas comme une solution tout à fait innovante en 1983 et s'inscrit, au contraire, dans toute une bataille entre logiciels dont l'objectif ultime est le monopole du traitement de texte. Bien avant la fin des années 1970, c'est-à-dire avant que ces logiciels ne fassent leur apparition, écrire un texte avec un ordinateur n'est pas quelque chose que l'on pratique quotidiennement, c'est d'ailleurs quelque chose que l'on pourrait qualifier d'impensable. Les ordinateurs, même lorsqu'ils commencent à être miniaturisés, sont d'abord conçus et utilisés en tant que machines à calculer et non pas comme des machines à écrire, comme c'était déjà le cas de leurs prédécesseurs de plus grande envergure, tel que l'un des premiers ordinateurs, baptisé le Harvard Mark I dont le nom officiel donné par IBM est *Automatic Sequence Controlled Calculator* (ASCC)[5]. D'ailleurs, les machines à écrire sont toujours en vogue à cette période[6] et c'est bien sur ces dernières que les informaticiens rédigent des documents, que ce soit de la documentation technique ou autre papier que l'on doit faire circuler ou publier.

Avec la multiplication des ordinateurs personnels et la diversité croissante des modèles sur le marché, les éditeurs de logiciels sont confrontés à la nécessité de consacrer beaucoup de temps à la rédaction de la documentation destinée aux utilisateurs, en raison de l'adaptation requise pour les différentes configurations matérielles possibles. Ainsi, les éditeurs de logiciels rédigent le code source sur une machine, l'ordinateur, tandis que la documentation est produite sur une autre, la machine à écrire. Or, la machine à écrire traditionnelle présente un inconvénient majeur, contrairement à l'ordinateur : elle n'offre pas de moyen d'éditer le texte une fois qu'il est tapé. Créer une documentation pour un logiciel devient très vite une activité fastidieuse, car il faut adapter la documentation pour chaque combinaison de matériel disponible, et selon tout un ensemble de caractéristiques du *hardware* comme le processeur utilisé ou encore la quantité de mémoire disponible dans l'ordinateur. Les technologies de reproduction des documents créés à l'aide de machines à écrire – l'utilisation du papier carbone combinée à une polycopieuse – ne sont plus adaptées dès qu'une modification du

5. Voir la page web : `https://history-computer.com/technology/the-history-of-harvard-mark-1/` consultée le 16 décembre 2024.

6. Nous avons vu dans le deuxième chapitre que des fabricants comme Olivetti fabriquent de nouveaux modèles à la fin des années 1960.

contenu de la documentation est nécessaire[7]. Dans un ordinateur, au contraire, les données sont accessibles en écriture. On peut donc les éditer, mais on peut également les dupliquer. Grâce à ces fonctionnalités, il est possible de maintenir plusieurs versions d'une documentation d'un logiciel, chacune étant adaptée à une configuration matérielle différente.

C'est ainsi qu'est développée l'idée de pouvoir écrire la documentation d'un logiciel dans le même environnement que son code source (Bergin 2006*a*). C'est dans cette situation que Michael Schrayer développe en 1976 le tout premier logiciel de traitement de texte grand public : *Electric Pencil*[8].

Malgré le fait que les fonctionnalités d'*Electric Pencil* paraissent limitées et minimales pour tout éditeur de texte contemporain, elles influenceront pourtant les logiciels qui suivirent, WordStar, WordPerfect, etc. Comme nous pouvons le lire dans la documentation du logiciel, nous retrouvons parmi ces fonctionnalités l'affichage du texte à l'écran, l'insertion de texte à n'importe quel endroit d'un document, les fonctions de recherche et de remplacement, la numérotation automatique des pages et, bien évidemment, la fonctionnalité d'impression du document (Schrayer 1977). À l'époque d'*Electric Pencil*, le traitement de texte est encore dans le paradigme *What You See Is What You Mean* (WYSIWYM) et repose sur la saisie de commandes et de fonctions propres à ce logiciel pour structurer les informations en vue d'obtenir un texte mis en page lors de l'impression. Dans cet environnement, l'utilisateur n'a pas à se soucier des marges sur les côtés ou du placement du texte dans la page, c'est *Electric Pencil* qui s'en occupe.

Donald Knuth, professeur en informatique à l'Université de Stanford, rencontre une autre problématique durant les années 1970. Insatisfait des épreuves que sa maison d'édition lui envoie pour la seconde édition de *The Art of Computer Programming* (Knuth 1997), il décide de développer TEX, un système de composition typographique, et publie une première version de celui-ci en 1978. Il s'agit là d'un des environnements d'écriture connaissant la plus grande longévité de l'histoire de ces environnements. À l'instar d'*Electric Pencil*, TEX fonctionne sur un mode WYSIWYM, sans autre interface graphique qu'un éditeur de texte. La composition du texte repose sur l'emploi de commandes que l'on écrit en texte brut à l'intérieur du document – de façon analogue au balisage d'un texte –, puis le programme TEX est en charge d'interpréter ces commandes et de compiler

7. L'abandon de la machine à écrire pour l'ordinateur pose des questions, car les fabricants de machine à écrire ont adapté leurs produits aux technologies de l'informatique, comme nous l'avons vu dans le chapitre II.2 avec le modèle M10 de chez Olivetti qui intègre un micro-processeur et un écran à cristaux liquide. Les machines à écrire des années 1980 peuvent être considérées comme des ordinateurs. Leur abandon n'est alors pas expliqué dans cette thèse.

8. Voir la documentation d'*Electric Pencil* publiée en 1977 : `https://www.sol20.org/manuals/pencil.pdf` (Schrayer 1977), consultée le 18 décembre 2024.

le document à imprimer à partir de ces dernières. Ces commandes peuvent être divisées en deux types. Il y a d'abord les fonctions primitives, initialement au nombre de 300, puis Donald Knuth y ajoute des macros que l'on appelle *plain TEX*. Toutes ces fonctions sont décrites dans le TEXBook, le manuel de référence écrit par Knuth. C'est un incontournable pour qui souhaite en savoir plus la conception d'un document dans TEX. Il est disponible en version numérique sur le dépôt du *Comprehensive TEX Archive Network* (CTAN) [9] (Knuth 1984*b*). Ce sont ces macros qui sont couramment utilisées par les usagers lorsqu'ils écrivent dans l'environnement TEX (et LATEX). Par la suite, la communauté a développé des milliers de paquets que tout utilisateur de LATEX peut charger dans son document selon ses besoins. On compte à ce jour 6694 paquets disponibles sur le CTAN, développés par 3033 contributeurs [10].

Que l'on soit dans *Electric Pencil*, TEX, ou encore d'autres environnements de cette même époque tels que EasyWriter [11], la conception du document numérique diffère quelque peu de la conception du document imprimé. Le document numérique comporte un ensemble de commandes qu'un logiciel interprète alors pour effectuer les calculs de mise page du texte en vue de son impression. Sans aperçu à l'écran, il est donc nécessaire d'attendre que le document soit imprimé pour valider ou non la mise en page. Même si l'objectif est l'obtention d'un document imprimé, écrire dans ce type d'environnement relève d'une raison fonctionnelle, où l'auteur déclare dans son document des fonctions et des commandes que la machine doit appliquer, et non d'une raison purement graphique.

L'approche WYSIWYG, qui offre cette possibilité d'afficher un aperçu de ce à quoi ressemblera le document après impression, apparaît sur le marché des logiciels en 1978 avec le logiciel WordStar, considéré par son concepteur comme le tout premier logiciel de ce type (Rubinstein 2006). WordStar était l'un des logiciels les plus populaires au début des années 1980, d'ailleurs, il est toujours utilisé grâce à des émulateurs MS-DOS, comme en témoigne George R. R. Martin, le célèbre auteur de la saga *Le Trône de Fer* (Kirschenbaum 2016 : 9). WordStar a été vendu à plusieurs millions d'exemplaires partout dans le monde. Si WordStar n'avait pas perdu des parts de marché à cause d'une grave erreur de conception, MicroPro International, l'entreprise qui détenait les droits du logiciel WordStar, aurait peut-être pu développer le concept de pack office que l'on connaît chez Microsoft : « Dès le départ, mon concept était celui d'une suite d'applications de

9. Voir la page web `https://mirrors.ircam.fr/pub/CTAN/systems/knuth/dist/tex/texbook.tex`, consultée le 18 décembre 2024.

10. Voir la page du CTAN : `https://ctan.org/`, consultée le 11 janvier 2025.

11. EasyWriter est le logiciel développé pour Apple. Il a été très mal noté du fait de sa non-portabilité sur d'autres ordinateurs qui n'étaient pas issus de la firme Apple puisqu'ils utilisaient un format de support qui n'était pas compatible avec les autres systèmes (Bergin 2006*a*).

type *front office*. Elle s'appelait Starburst, et contenait CalcStar, InfoStar, et WordStar : le tableur, la base de données et le traitement de texte » (Rubinstein 2006 : 71).

Cette interface, aussi rudimentaire fut-elle, était malgré tout la première interface à montrer des éléments à l'écran comme les sauts de page, le gras et l'italique[12]. En fournissant ce type de fonctionnalité, WordStar avait un atout majeur par rapport à ses concurrents puisqu'il permettait de faire de l'ordinateur un objet fonctionnel et utilisable immédiatement sans qu'aucune compétence en ingénierie ou en informatique ne soit requise (Rubinstein 2006). Toutefois, le déclin de WordStar est tout aussi fulgurant que sa montée en puissance. Puisque le logiciel initial a été programmé directement en assembleur pour le processeur Intel 8080, le portage sur le système Unix plus moderne a été fastidieux (Bergin 2006*a*). Au lieu de s'appuyer sur l'existant et de créer un nouveau logiciel compatible avec l'ancien, WordStar 2000, le logiciel créé à la suite de WordStar, ne permettait pas de traiter des documents créés avec WordStar (et inversement), causant ainsi la chute de WordStar à partir de 1984 pour laisser la place à WordPerfect et à MSWord (Campbell-Kelly 2001 : 6), qui seront leader du marché des logiciels de traitement de texte jusqu'en 1992, lorsque le système d'exploitation Windows deviendra le plus populaire et permettra à Microsoft de dominer ce marché avec son logiciel *Word for Windows*. Intégré depuis 1989 à Windows dans sa version 1.0, *Word for Windows* reçoit à cette date de bonnes critiques, selon les propos de Bergin (Bergin 2006*b*). Deux ans plus tard, en 1991, lorsque Word 2.0 est intégré à Windows, le logiciel reçoit également de très bonnes critiques, alors que les premières versions des logiciels WordPerfect et WordStar sont également disponibles pour Windows. Ces derniers recevront pourtant de mauvaises critiques, car ils ne profitent pas pleinement de l'environnement de Windows, contrairement à MSWord.

MSWord a fait sa première apparition sur le marché public en 1983. En 1981, Microsoft embauche Charles Simonyi, un ancien développeur du PARC de l'entreprise Xerox, pour développer MSWord. Historiquement, le PARC est un centre de recherche très célèbre pour les développements des logiciels de la micro-informatique et des interfaces graphiques. Le PARC a largement inspiré la conception des environnements numériques mis au point par les entreprises que nous considérons aujourd'hui comme des mastodontes de ce milieu, Apple, Microsoft, Adobe, etc.

Les travaux de Simonyi, tout comme la plupart des recherches effectuées chez Xerox PARC, ont largement été influencés par les travaux de Licklider, l'ancien

12. Selon les propos de Rubinstein : « *WordStar was the first program that could show on-screen where the page breaks would fall* » (Rubinstein 2006 : 69)

directeur de l'ARPA de 1962 à 1964, notamment son article « Man Computer Symbiosis » (Licklider 1960), tel qu'en témoigne Bob Taylor dans l'ouvrage *Dream Machine* de Waldrop, qui est d'abord directeur de l'ARPA de 1965 à 1969 après Licklider, puis fondateur du Xerox PARC :

> Je ne pense pas qu'Ivan, ou moi, ou quiconque a occupé ce poste à l'ARPA depuis, ait eu la même vision que Lick. La plupart des avancées significatives dans la technologie informatique – y compris le travail effectué par mon groupe au Xerox PARC – étaient simplement des extrapolations de la vision de Lick. Il ne s'agissait pas vraiment de nouvelles visions propres. C'est donc lui qui est à l'origine de tout (Waldrop 2001 : 470).

Dans cet article fondateur « Man Computer Symbiosis », Licklider décrit ainsi sa vision du futur de l'informatique et des relations humain-machine. Pour Licklider, la relation entre un ordinateur électronique et un être humain est une promesse d'une évolution des modes de pensée. Dans cette relation symbiotique, l'ordinateur devient une aide à la prise de décision dans des systèmes complexes que l'humain ne pourrait réaliser sans son concours. De la même manière, Licklider ne croit pas au dépassement de l'intelligence artificielle sur l'intelligence humaine. Au contraire, il estime qu'avant d'en arriver à une telle situation, les machines et les humains vont « travailler ensemble dans une association intime » pour développer des pensées complexes (Licklider 1960 : 5). La notion de symbiose que convoque l'auteur positionne la machine comme une entité bénéficiaire de cette relation, et pas seulement comme un outil permettant d'étendre les capacités de ce dernier via l'automatisation. Ce bénéfice, que ce soit pour l'humain ou la machine, se trouve justement au niveau de cette pensée. En 1960, lorsque Licklider écrit ces lignes, les ordinateurs ne sont pas encore à la hauteur de cette vision. Aussi, après avoir introduit cette dernière, il dresse une liste des points d'amélioration nécessaires pour l'atteindre : la mémoire numérique, les langages de programmation, l'équipement de gestion des *inputs* et *outputs* tels que la reconnaissance vocale, notamment.

Au début des années 1970, le PARC développe un ordinateur personnel, l'Alto, lequel a fait l'objet des premières implémentations d'interface graphique. C'est dans ce contexte que Simonyi a effectué ses travaux de doctorat au Xerox PARC, développant notamment le logiciel de traitement de texte Bravo dans le cadre de ses expérimentations. Bravo a été directement implémenté sur l'ordinateur Alto, tirant parti de ses capacités graphiques. Par la suite, Simonyi a contribué à la mise sur le marché de l'ordinateur personnel Alto, après deux années de développement (Swaine et Freiberger 2000 : 326).

Dans un entretien organisé par le Computer History Museum en 2008 avec Simonyi, celui-ci atteste l'emploi du terme WYSIWYG en 1975 avec la version 3.1 du logiciel Bravo (Simonyi 2008 : 21), que l'on peut alors considérer comme le tout premier logiciel de traitement de texte WYSIWYG. Ce fait est d'ailleurs contesté par Rubinstein, comme nous l'avons vu précédemment avec son logiciel WordStar. Pourtant, Bergin semble avoir trouvé des traces de cet acronyme dans les compte-rendus du projet Bravo (Bergin 2006*b*). En complément, nous pouvons lire dans la documentation du logiciel Bravo datant de 1979 :

> Si vous comparez la copie papier de votre document avec l'écran de Bravo, vous verrez que, bien que le texte soit identique, la copie papier comporte plus de mots sur chaque ligne, de sorte que les deux versions sont très différentes. Pour voir à l'écran un fac-similé presque exact de la copie papier, vous pouvez passer la commande `Look hardcopy (note the lower-case h)`. Vous êtes maintenant en mode copie papier à l'écran. Jusqu'à nouvel ordre, Bravo représentera la version imprimée de votre document aussi fidèlement que possible, en positionnant chaque caractère à l'écran à un demi-point d'écran (environ 0,007 pouce) de sa position dans la copie papier finale (Lampson et al. 1979 : 38).

Cette partie de la documentation nous laisse penser que le mode *hardcopy* est ce qui se rapproche le plus d'un mode WYSIWYG où l'affichage du document à l'écran est quasiment conforme au document imprimé, bien que le rapport d'échelle entre l'écran et le papier ne soit pas de un.

Si WordStar et Bravo sont les premiers outils de traitement de texte WYSIWYG, ils ne sont pas encore dans une interface graphique avec des icônes sur lesquels cliquer pour déclencher une action. C'est également le PARC qui implémente cette fonctionnalité dans certains de leurs logiciels. À ce propos, implémentation ne veut pas dire invention, car le PARC n'a pas tout inventé. Beaucoup de recherches ont été menées dans d'autres centres de recherche et ont été implémentées dans les ordinateurs plus ou moins pendant les mêmes années. Le PARC réutilise notamment les recherches réalisées à l'Université de Stanford, par exemple la souris qui a été développée par Douglad Engelbart et ses équipes de recherche dans les années 1960. La demande de brevet pour cette technologie a d'ailleurs été déposée en 1967 et obtenue en 1970, comme en atteste le document publié par l'office des brevets des États-Unis (Engelbart 1970). Si le PARC récupère ces technologies, c'est parce qu'une partie des membres travaillant avec Engelbart sont ensuite allés travailler au PARC. Le PARC implémente ainsi une série d'avancées en informatique telles que : les interfaces graphiques, les

icône-boutons, etc., mais ces développements ne seront pas vraiment disponibles sur le marché des logiciels durant les années 1970.

Au sujet des développements réalisés par le PARC, on peut lire dans l'interview de Simonyi précédemment citée un passage racontant une discussion avec Steve Jobs datant de 1983 :

> « Oh maintenant je dois vous montrer les joyaux de la couronne, c'est top, top, top, top, top, secret » et puis il nous montre ou nous dessine en fait, je ne pense pas que c'était même dans le, il nous dessine juste un menu déroulant, et vous savez, nous étions assis là et nous avons dit, « Oh ouais, Steve, c'est très bien. » Bien sûr, nous savions tout sur les menus déroulants. Je veux dire que j'avais vu des menus déroulants pendant dix ans à ce moment-là. Je suis sûr que ça a dû être choquant pour tout le monde, mais pour nous, vous savez, vous avez un peu oublié ça, et il a tendance à oublier que nous venons tous des mêmes racines, et que certains d'entre nous étaient plus proches des racines que d'autres, mais en même temps, vous savez, il dit toujours à Bill que Bill a volé Mac et alors Bill dit, « Non, non, non, nous avons tous les deux volé la même chose, vous savez, nous avons tous les deux volé le truc de Xerox » (Simonyi 2008 : 34).

Grâce à Simonyi, MSWord hérite des réalisations du PARC pour ses développements, notamment des résultats obtenus avec le logiciel de traitement de texte Bravo et les recherches sur les interfaces graphiques.

Pourtant, MSWord n'est pas le logiciel le plus utilisé dès sa sortie, il faudra attendre presque 10 années pour que ce soit le cas, dès que le logiciel sera intégré au système d'exploitation Windows 3.0 en 1992 (et à son interface graphique). Comme nous l'avons vu, WordStar domine le marché à cette époque-là, suivi par WordPerfect[13] jusqu'à ce que MSWord remporte cette course au monopole en 1992 (Bergin 2006*b*). L'engouement observé à cette époque pour les traitements de texte et le mode WYSIWYG ne tient pas seulement du fait que l'ordinateur permette d'écrire : plus que sa capacité à écrire, l'ordinateur permet d'obtenir un document qualifié de « parfait ». Dans un passage de son ouvrage sur le traitement

13. Nous passons très rapidement sur le logiciel WordPerfect, car il n'apporte rien de plus significatif à cette étude. Néanmoins, nous avons noté que Kittler l'utilise comme exemple dans sa présentation intitulée *Le logiciel n'existe pas* publiée en 1992 qu'il rédige avec ce logiciel pour définir la caractéristique exécutable des mots dans l'espace numérique : « Le sigle WP fait en effet ce qu'il dit. Contrairement non seulement au mot "WordPerfect", mais aussi aux mots vides de la vieille Europe tels que "Esprit" ou "Mot", les fichiers informatiques exécutables incluent tous les sous-programmes et les données nécessaires à leur réalisation. L'acte d'écriture consistant à frapper les touches W et P puis "Entrée" d'une console AT ne rend certes pas le *mot parfait* mais lance l'exécution de WordPerfect » (Kittler 2015 : 34). Dans ce cas-ci WordPerfect incarne cette caractéristique exécutable où le mot ne renvoie plus à une définition des mots le composant, mais à un ensemble de programme qu'il appelle.

de texte que cite Kirschenbaum (2016 : 36), McWilliams résume cette idée de la perfection du document comme suit : « Une fois que tout ce traitement des mots a eu lieu, l'ordinateur de traitement de texte imprime autant de copies que vous le souhaitez, avec des lettres parfaites » (McWilliams 1983 : 23).

Il y a donc ici un lien très étroit entre l'écriture sur ordinateur et le document que l'on obtient après impression. Ce qualificatif « parfait », fait référence à une forme de « dématérialisation de l'acte d'écriture » car le document, une fois imprimé, ne comporte plus les traces de son écriture, comme s'il n'avait pas d'historique des modifications et qu'il avait directement émergé de l'esprit de l'auteur en une seule itération (Kirschenbaum 2016 : 36). Cette perfection du document issu du traitement de texte est à mettre en perspective avec le document obtenu avec une machine à écrire. Quel que soit le dispositif utilisé pour écrire – qu'il s'agisse d'une machine à écrire, d'un ordinateur ou d'un stylo –, des erreurs ou coquilles peuvent facilement se glisser dans un document. Toutefois, en dehors de l'ordinateur, les processus de réécriture et de correction des erreurs dans les autres dispositifs laissent souvent des traces visibles dans le document final. Par exemple, une erreur corrigée en bas de page peut laisser une marque discrète sur un document produit avec une machine à écrire, car réécrire toute une page était fréquemment jugé trop chronophage. En revanche, si les erreurs se situent en haut de page, il était parfois nécessaire de réécrire entièrement le document sur une nouvelle feuille (Kirschenbaum 2016 : 34). Et encore, cela ne tient pas compte des conséquences que ces coquilles pouvaient avoir sur les copies carbones du document original. La correction ou la réécriture d'un document a alors un coût économique très élevé, à la fois en temps de travail, mais aussi en fournitures.

Écrire sur un ordinateur avec un logiciel de traitement de texte a pour effet de diminuer ces coûts et d'obtenir, depuis ce point de vue, des documents « parfaits ». La perfection du document numérique a alors un double sens : elle est à la fois économique et gratifiante pour l'auteur. Dans le deux cas de figure, ce qualificatif désigne la copie imprimée du document et pas sa version numérique. La version numérique n'est alors qu'un moyen d'obtenir une version papier (*hardcopy*) du document. En ce sens, la modélisation du texte et de sa structure n'a que peu d'importance puisque, l'élément qui doit être parfait est le document imprimé.

Néanmoins, alors que dans les premières décennies de l'histoire des traitements de texte, le document parfait correspondait au document imprimé, cette assertion est-elle toujours valide pour les logiciels contemporains ?

III.3 XML et le traitement de texte

Des années 1970 jusqu'aux années 2000, l'écriture avec des logiciels de traitement de texte a été dominée par des formats propriétaires, limitant ainsi l'échange et la pérennité des documents numériques entre différents environnements. L'un des exemples les plus marquants est le format DOC, utilisé dans le logiciel MSWord, qui était inaccessible en dehors de l'écosystème Microsoft. Ce verrouillage posait un problème majeur d'interopérabilité, notamment pour les institutions et les entreprises qui devaient échanger des documents avec différents acteurs.

Face à la demande croissante d'interopérabilité et sous la pression des directives nationales et supranationales émises entre 2007 et 2009 (Yami et al. 2015), les entreprises éditrices des logiciels de traitement de texte ont amorcé un changement stratégique. L'essor d'initiatives visant à standardiser les formats de documents, telles que le Référentiel Général de l'Interopérabilité (RGI) en France initié par l'ordonnance 2005-1516 du 8 décembre 2005 et publié en 2009, qui recommande l'utilisation des normes ODF ou OOXML (*Référentiel général d'interopérabilité (RGI)* 2009 : 61), a conduit à une remise en question du modèle propriétaire.

En ce sens, l'entreprise Sun Microsystems, après avoir acquis la suite logicielle StarOffice en 1999, a joué un rôle central dans le développement de la suite bureautique OpenOffice, notamment du fait de l'ouverture du code source du logiciel en l'an 2000. En 2005, l'entreprise a soumis une proposition de norme à l'*Organization for the Advancement of Structured Information Standards* (OASIS) afin d'établir un format de document ouvert. Cette demande a abouti à l'adoption de la norme ODF, approuvée par l'OASIS en mai 2005[14]. Peu après, l'OASIS a soumis cette norme à l'ISO, qui l'a officiellement reconnue en décembre 2006 sous la référence ISO/IEC 26300:2006 (2006), faisant ainsi d'ODF une norme internationale[15].

En réaction à cette nouvelle norme, Microsoft a introduit une norme concurrente, OOXML, qui a été soumise à l'ECMA en 2006 avant d'être approuvée comme norme ISO en 2008 sous la référence ISO/IEC 29500-1:2008 (2008)[16]. Cette norme, tout en restant sous l'influence de Microsoft, permet une plus grande

14. L'approbation de la norme ODF par l'OASIS est publiée sur le site web de l'OASIS, voir `https://www.oasis-open.org/standards/`, page consultée le 06 janvier 2024.

15. L'approbation de la norme ODF par l'ISO est visible sur le site web de l'ISO, voir `https://www.iso.org/standard/43485.html`, page consultée le 06 janvier 2024.

16. Voir la page web : `https://ecma-international.org/publications-and-standards/standards/ecma-376/`, consultée le 06 janvier 2025.

accessibilité de l'environnement d'écriture proposé par Microsoft tout en s'appuyant sur un standard documenté. Son application dans Microsoft Word sous la forme *WordprocessingML*, un dérivé du format XML, est implémentée avec le format DOCX en 2008 et marque une rupture avec le format historique DOC.

La transition de Microsoft vers un format plus ouvert n'a pas été uniquement motivée par des considérations techniques, mais aussi par des enjeux concurrentiels. En parallèle, la suite bureautique OpenOffice avait adopté l'ODF comme standard en 2005, constituant ainsi une alternative crédible aux solutions propriétaires de Microsoft. La possibilité d'échanger des documents entre différentes plateformes sans contrainte de logiciel devenait un critère déterminant pour les administrations et les entreprises. De plus, avec 90% des parts du marché détenues depuis 1994 par le format DOC (Yami et al. 2015 ; Kirschenbaum 2016 : 235), Microsoft risquait de perdre sa position dominante si les documents produits avec leur outil devenaient incompatibles avec les exigences des nouvelles normes d'interopérabilité. L'adoption d'OOXML a ainsi permis à Microsoft de maintenir sa présence sur le marché tout en se conformant aux directives internationales.

Parmi les différentes possibilités d'évolution de MSWord, Microsoft a fait le choix de créer sa propre norme pour se distinguer de leur concurrent plutôt que d'utiliser la même norme qu'eux, *Open Document Format*. Pour comprendre le choix que fait Microsoft, Yami, Chappert et Mione proposent en 2015 une étude des stratégies de compétition et de coopération mises en œuvre par l'entreprise Microsoft auprès du comité de l'AFNOR – représentant de la position française – dans le processus de normalisation d'*Open Office XML* auprès de l'ISO (Yami et al. 2015).

En 2007, une fois que l'ECMA a validé le format comme standard officiel, elle le soumet ensuite à l'ISO pour élaborer une norme internationale par « voie express ». Yami *et al.* nous montrent que le processus de normalisation de ce format n'est pas une affaire banale et, qu'au contraire, elle a fait l'objet de discussions, d'échanges et de votes (pas toujours positif) avant que OOXML soit adoubée en tant que norme ISO. Au-delà de la création d'une norme concurrente, l'ISO a soulevé la contradiction dans l'élaboration d'une nouvelle norme internationale répondant au même besoin que la norme ODF créée quelques mois auparavant. Dans l'article de Yami *et al.*, nous trouvons des extraits de courriels envoyés au moment de l'élaboration de documents pour les enquêtes probatoires dont le contenu est clairement positionné contre l'établissement de cette norme :

> (...) Ne t'inquiète pas, ceci a bien été vu et dit hier. Pour info, la réunion d'hier, dont j'étais, a été assez houleuse, Microsoft contestant systématiquement tous les points et tentant de faire rejeter un

> maximum de commentaires. Plus surprenant, le Président de la commission a tenté à plusieurs reprises d'éliminer des commentaires techniques pertinents qui n'étaient peut-être pas bien rédigés (les règles étant pour le moins floues, pour ma part, si les règles avaient été plus claires j'aurais bâti autrement ma contribution) sous prétexte que lui ne les comprenait pas. Je souhaite bien du plaisir à ceux qui vont travailler aujourd'hui, car vu la situation d'hier, il est clair qu'il n'y a pas consensus, et qu'il parait assez évident que MS devrait retravailler complètement son brouillon, et l'alléger de plein de fatras. Problème, si ceci est fait, il ne resterait sans doute qu'un clone bien proche d'ODF ! (...) (Courriel 127) (Yami et al. 2015 : 345).

Lors d'un premier vote le 4 septembre 2007, l'assemblée formée à l'ISO, composée de membres et délégations provenant de 104 pays différents, vote pour ne pas retenir ce format en tant que norme ISO car :

> l'approbation exige une majorité des 2/3 (66,66 %) des votes exprimés par les membres participants de l'ISO/IEC JTC 1. Les votes négatifs ne doivent pas dépasser le quart (25 %) du nombre total des votes exprimés par les comités membres nationaux. Aucun de ces critères n'a été rempli, avec 53 % de votes positifs des membres participants de l'ISO/IEC JTC 1 et 26 % de votes nationaux négatifs[17].

Cinq mois plus tard, du 25 au 29 février 2008, les délégations nationales de 37 pays se sont réunis une deuxième fois pour analyser les 3500 commentaires émis en 2006 sur ce dossier[18]. Sur ces cinq journées de travail, il est mentionné que le comité n'a pas pu passer en revue la totalité des commentaires, car leur nombre était trop élevé pour un traitement dans un laps de temps aussi court.

> Lorsqu'il est apparu qu'il serait impossible de passer toutes les observations en revue, il a été convenu, après discussion, d'une procédure de vote pour statuer sur le reste des modifications proposées. Au total 43 résolutions, impliquant des dispositions ou des groupes de dispositions, ont été acceptées, la plupart d'entre elles à l'unanimité, quelques-unes par voie de consensus, quatre seulement à la majorité simple ; quatre ont été refusées[19].

17. Voir la page web de l'ISO : `https://www.iso.org/fr/news/2007/09/Ref1070.html`, consultée le 06 janvier 2025.

18. Le nombre de ces commentaires sur le dossier ISO 29500 a été ramené à 1100 après une première phase de tri effectuée par des experts, notamment pour éliminer les doublons, nous indique le compte-rendu de la réunion d'analyse des votes sur ce projet, voir `https://www.iso.org/fr/news/2008/02/Ref1114.html`, consultée le 31 janvier 2025.

19. Voir la page web de l'ISO : `https://www.iso.org/fr/news/2008/03/Ref1117.html`, consultée le 07 janvier 2025.

Suite à cette deuxième réunion, 87 délégations ayant voté en septembre 2007 ont alors bénéficié d'un mois supplémentaire (jusqu'à mars 2008) pour modifier leur vote initial et ainsi changer l'issue négative de création d'une norme concurrente à ODF[20]. Le 2 avril 2008, un billet est publié sur le site de l'ISO et confirme l'approbation de la norme 29500 en tant que norme internationale « avec 75% de votes positifs des membres participants du JTC 1 et 14% de votes négatifs sur le total des votes des comités membres nationaux »[21].

Depuis l'introduction de cette norme internationale, Microsoft s'appuie sur celle-ci à travers *WordprocessingML*, son application pour gérer la structuration des informations dans le logiciel de traitement de texte MSWord.

Ce processus de normalisation aura mobilisé des centaines de personnes provenant du monde entier, qu'elles soient expertes du domaine ou qu'elles soient des concurrentes. Malgré les avis divergents, la norme soumise par l'ECMA devient une norme internationale sur laquelle des applications de traitement de texte (et des feuilles de calcul, etc.) peuvent être construites. Qu'en est-il de tout ce « fatras » évoqué par les membres de la délégation durant l'élaboration du dossier, ou encore des questions sur les différences entre *Open Document Format* et *Office Open XML* ?

La réalisation du schéma pseudo-XML de Microsoft s'étale sur plusieurs années dès la fin des années 1990, et non pas à la hâte en 2005, lorsque la norme ODF est acceptée par OASIS. Microsoft connaît très bien cette technologie. D'ailleurs, bien qu'il ne soit pas nommé dans les premiers brouillons des spécifications de XML en 1996, le nom de Jean Paoli, travaillant alors pour Microsoft, apparaît dès août 1997 parmi les éditeurs de ces documents[22]. Dans ce même document, apparaît également le nom de Jon Bosak travaillant pour Sun Microsystems[23], l'entreprise qui a développé la suite OpenOffice avant qu'elle ne soit rachetée en 2008 par Oracle.

> XML a été développé par un groupe de travail XML (connu à l'origine sous le nom de SGML Editorial Review Board) formé sous les auspices du World Wide Web Consortium (W3C) en 1996 et présidé par Jon Bosak de Sun Microsystems [...] Cette spécification a été

20. Voir la page web de l'ISO : `https://www.iso.org/fr/news/2008/02/Ref1114.html`, consultée le 07 janvier 2025.

21. Voir la page web de l'ISO : `https://www.iso.org/fr/news/2008/04/Ref1123.html`, consultée le 07 janvier 2025.

22. Voir les spécifications XML sur le site web du W3C : `https://www.w3.org/TR/WD-xml-970807.html`, consulté le 07 janvier 2025.

23. L'acronyme SUN du nom de Sun Microsystems créé en 1982 signifie *Stanford University Network*. L'Université de Stanford se situe à Palo Alto, non loin du Xerox PARC. C'est également dans cette université qu'a travaillé le mathématicien et informaticien Donald Knuth, auteur de TEX, ainsi que Douglas Engelbart, l'inventeur de la souris. Bosak, pour sa part, n'a pas été étudiant à Stanford.

> préparée et approuvée pour publication par le W3C XML Working Group (WG). L'approbation de cette spécification ne signifie pas nécessairement que tous les membres du WG ont voté pour cette approbation. Au moment où cette spécification a été approuvée, le WG était composé des membres suivants : Jon Bosak, Sun (Chair) ; James Clark (Technical Lead) ; Tim Bray, Textuality and Netscape (XML Co-editor) ; Jean Paoli, Microsoft (XML Co-editor) ; C. M. Sperberg-McQueen, U. of Ill. (XML Co-editor) ; Steve DeRose, INSO ; Dave Hollander, HP ; Eliot Kimber, Highland ; Tom Magliery, NCSA ; Eve Maler, ArborText ; Murray Maloney, Grif ; Peter Sharpe, SoftQuad [24].

Microsoft était donc impliqué dans le développement de la technologie XML depuis ses débuts et a d'ailleurs commencé son intégration à la suite Office dès la version de 2003 [25]. Dans la note de remise aux fabricants datant du mois d'août 2003, nous pouvons y lire que Microsoft se prépare déjà à rendre son environnement interopérable avec d'autres logiciels grâce au XML :

> La prise en charge de XML et d'autres normes industrielles permet une communication transparente entre les programmes Microsoft Office et les applications d'éditeurs de logiciels indépendants, et pose les bases de l'ajout de données en temps réel et de services Web pour une valeur commerciale encore plus grande. Les améliorations apportées à Outlook, Exchange, Windows® SharePoint Services et SharePoint Portal Server offrent de nouvelles fonctionnalités qui permettent aux individus, aux équipes et aux organisations de collaborer le plus efficacement possible [26].

Quelques semaines plus tard, en octobre 2003, lorsque la nouvelle version de la suite Office est rendue accessible au public, Steve Ballmer, le directeur exécutif de Microsoft à cette époque, s'exprime publiquement lors de l'événement de lancement de la nouvelle version de Microsoft Office [27] dans lequel il prévoit clairement que le futur du pack Office réside dans l'interconnexion des logiciels entre eux, avec pour objectif d'améliorer la productivité des utilisateurs :

---

24. Voir le site web du W3C : `https://www.w3.org/TR/WD-xml-970807.html`, consulté le 07 janvier 2025.
25. Voir la note de remise aux fabricants sur le site web de Microsoft datant du 19 août 2003 : `https://news.microsoft.com/2003/08/19/core-microsoft-office-system-products-are-complete-released-to-manufacturers/`, consulté le 07 janvier 2025.
26. Voir le § 7 de la note de remise aux fabricants datant du mois d'août 2003 mentionnée précédemment.
27. Le discours de Steve Ballmer a été retranscrit et est accessible sur le site web de Microsoft : `https://news.microsoft.com/speeches/steve-ballmer-speech-transcript-microsoft-office-system-launch/`, consulté le 06 août 2025.

> De notre point de vue, il ne s'agit pas d'un tableur et d'un traitement de texte. C'est peut-être là que les gens pensent qu'il a évolué. Nous considérons Microsoft Office comme le fondement de l'ensemble définitif d'outils que nous devons fournir pour améliorer la productivité des travailleurs de l'information. [...] Nous avons apporté quelques améliorations en matière de collaboration. Nous avons étendu leurs capacités Internet. Nous avons facilité leur déploiement. Mais c'est vraiment la première génération depuis longtemps où nous avons, je pense, fait avancer de manière significative la frontière de la productivité pour le travailleur de l'information. Le système Microsoft Office 2003 n'est pas seulement un traitement de texte et un tableur. En fait, il ne s'agit même pas d'une simple application côté client. Il s'agit d'une application côté client qui peut parler et exploiter intelligemment l'infrastructure du serveur, les services qui fonctionnent dans le nuage Internet et les solutions qui ont été développées et écrites par nos partenaires ainsi que par les clients finaux qui aident à améliorer l'expérience globale de la productivité pour les gens (Ballmer 2003 : § 25 & § 29-30).

Durant les années 1970 et 1980, l'ordinateur et les premières générations de logiciels de traitement de texte ont remplacé la machine à écrire, puis, pendant les années 1990, les interfaces graphiques se sont imposées et ont remplacé les interfaces textuelles. Au tournant du nouveau millénaire, Ballmer met l'accent lors de son discours sur l'interopérabilité des données entre les logiciels de l'environnement numérique pour favoriser la collaboration et fluidifier la circulation des informations. Parmi plusieurs technologies employées, cette interopérabilité est principalement rendue possible grâce à l'implémentation du format XML.

Ballmer ajoute même que « XML est la clé. XML permettra à toute application de commencer à être exploitée intelligemment par d'autres applications XML de la bonne manière » (Ballmer 2003 : § 129).

En prenant ceci en considération, est-il possible que le futur de Microsoft Word en 2006 / 2007 ne repose que sur du « fatras » qualifié de « brouillon » comme cela a été indiqué dans les échanges de courriels lors du traitement de la demande de création d'une norme ISO pour la modélisation d'OOXML (Yami et al. 2015) ?

Pour répondre à cette question, nous émettons ici une hypothèse que nous ne pourrons vérifier faute d'accès au code source et à la documentation des modèles textuels de MSWord antérieurs à *WordprocessingML*. Notre hypothèse est que l'entreprise Microsoft s'est vue contrainte par la nécessité de produire un schéma qui assure la préservation des documents produits sous MSWord avec le

format DOC. Tandis que du côté du concurrent direct de MSWord, OpenOffice, cette difficulté ne se pose pas puisque le format XML a été employé comme pivot de la structuration des éléments dès le début. Le passage à la norme ODF assure alors, pour Sun Microsytems, la rétrocompatibilité avec les anciens documents produits avec OpenOffice. Il y a donc une urgence pour Microsoft de produire une norme équivalente à celle de la concurrence qui puisse assurer que les documents au format DOC soient toujours interprétables avec la nouvelle structure implémentée dans les nouvelles versions du logiciel. En un sens cette préservation des documents est déjà assurée puisque Microsoft a implémenté XML dans MSWord dès 2003. Seulement cette implémentation n'est pas conforme à la norme ODF en vigueur deux ans plus tard. Pour étayer notre hypothèse, nous nous appuyons sur le fait que les auteurs de la proposition de la norme justifient leur démarche auprès de l'ISO par l'un de leurs objectifs qui « est d'assurer la conservation à long terme de documents créés durant les deux dernières décennies par des programmes qui deviennent incompatibles avec les nouveautés continues dans le domaine des technologies de l'information »[28]. Cet argument, nous le retrouvons également dans le référentiel général d'interopérabilité publié en 2009 par le gouvernement français, où une observation indique que « le format Office Open XML [...] supporte nativement une partie des formats binaires bureautiques existants » (*Référentiel général d'interopérabilité (RGI)* 2009 : 61), sous-entendu le format propriétaire DOC qui détenait depuis 1994, et jusqu'à l'arrivée de la norme ODF, la majorité des parts de marché du traitement de texte.

Si Microsoft n'avait pas assuré de continuité entre DOC et DOCX de 2003 à 2007, tous les documents créés avec MSWord pendant les années qui ont précédé cette période charnière n'auraient plus été interprétables par les logiciels de traitement de texte qui lui ont succédé (à partir de la version de 2008 de MSWord). Comme nous l'avons vu précédemment, au début des années 1980, MicroPro International, l'entreprise de Seymour Rubinstein, a commis une erreur tragique pour l'entreprise et ses logiciels en ne garantissant pas la continuité entre le logiciel WordStar et son successeur WordStar 2000. Pour Rubinstein, cette action leur a coûté leurs parts du marché des logiciels de traitement de texte et a précipité la fin de WordStar. À partir de cet antécédent, nous supposons que Microsoft n'a pas voulu reproduire l'erreur commise par MicroPro International.

Si nous revenons légèrement en arrière dans le temps, le succès initial de MSWord (et plus largement celui de Microsoft) à la fin des années 1980 et au début des années 1990 repose principalement sur l'intégration de composants

28. Voir la page de l'ISO : `https://www.iso.org/fr/news/2007/09/Ref1070.html`, consultée le 07 janvier 2025.

graphiques dans l'interface, et sur le mode WYSIWYG de son logiciel de traitement de texte, ce même mode qui permettait de produire des « documents parfaits ». La structuration des informations était alors axée sur un paradigme graphique et imprimé, centré sur l'objet page, et non sur une structuration textuelle fine des informations dans le document en vue de son partage. Cependant, avec la modélisation du texte et des informations en XML, les documents produits avec des logiciels de traitement de texte ne doivent plus seulement être imprimés, mais également être interprétés voire transformés par d'autres logiciels de l'environnement numérique.

III.4 Du texte traité au texte brut

Cette section du chapitre est consacrée à l'analyse des chapitres de l'ouvrage collectif *Contribution numérique* rédigés dans MSWord et LibreOffice.

Information

L'ouvrage *Contribution numérique : cultures et savoirs* restitue les interventions présentées lors du forum éponyme, organisé en juillet 2022. L'ouvrage a été publié en 2024 aux Ateliers de sens public, une maison d'édition dont le projet éditorial est de proposer des monographies en accès libre sous plusieurs formats : PDF, ePUB, HTML augmenté et imprimé.

Ce livre est construit sur la base de 13 sources : 12 chapitres transmis dans des formats de logiciels de traitement de texte dont 8 au format DOCX et 4 au format ODT. Le dernier document a été envoyé au format Markdown et a été écarté de l'étude.

Pour rédiger les chapitres de cet ouvrage, les auteurs ont reçu pour instruction de fournir un document au format DOCX ou ODT, comportant un titre, la liste des auteurs, un choix de thématique parmi une liste de trois thématiques dans laquelle le texte s'inscrit (ces thématiques correspondent aux trois axes traités dans le livre), un résumé, une liste de mot-clefs, et le contenu principal du document. D'autres recommandations s'appliquait au texte principal, à la fois en termes de structuration, avec un seul niveau de titre souhaité, qu'en termes de volume avec 20000 caractères espaces comprises environ.

Nous analysons plusieurs éléments de production du sens à l'intérieur de ces documents, que nous organisons en trois parties : d'abord les éléments « titre » et « paragraphe », puis les éléments « commentaire » et « annotation », et enfin l'élément « résumé ». À travers la comparaison de ces éléments décrits dans les

documents DOCX et ODT, nous mettrons en lumière les différents sens produits par ces environnements d'écriture numérique.

L'accès à la structure des éléments textuels en XML contenus dans les fichiers DOCX et ODT se fait en les décompressant [29]. Lorsque l'on observe les structures en *WordprocessingML* et ODF des chapitres de l'ouvrage décompressés, l'une des premières différences que l'on peut noter concerne la conception de ce qu'est un *document* dans ces archives.

Comme nous l'avons vu dans le chapitre II.4, le document réfère à un fait, il est une preuve, et il n'est pas forcément constitué d'un seul objet (Briet 1951). L'objet livre illustre bien ce propos : si un livre comme *Contribution numérique* peut être considéré comme un seul document, il peut également l'être comme constitué de tout un ensemble de documents composés de chapitres, de pages, etc., où chacun de ces éléments, isolé de leur objet, est lui-même un document. Par exemple, un extrait du livre peut être polycopié et devenir un support de travail durant un cours : c'est un document isolé de son origine le livre.

Le même phénomène peut être appliqué à un document numérique : le document n'est pas obligatoirement formé d'un seul fichier. C'est d'ailleurs ce que laisse supposer le principe de l'archive XML présentée sous la forme de DOCX et ODT. Lorsque l'on décompresse ces archives documentaires, on remarque qu'elles contiennent plusieurs fichiers différents qui, mis tous ensemble, forment un document de traitement de texte que l'on désigne par son format. Seulement, si nous envisageons le document produit par les logiciels de traitement de texte sous cette perspective, nous ne prenons pas en considération le sens que leur donnent les normes auxquelles ils sont associés. Or, l'une des finalités de ces normes est justement de définir ce qu'est un document de traitement de texte. Afin de vérifier les définitions du document que contiennent les schémas ODF et *WordprocessingML*, un premier indicateur consiste à lire le nœud racine du document.

Ce nœud racine nous renvoie aux schémas qui décrivent les éléments et leurs règles d'emploi dans les documents de traitement de texte, ainsi que leur valeur sémantique. Ces règles nous indiquent les structures que l'on peut employer dans le document, leurs règles d'écriture, ainsi que leur définition. Par exemple, l'une des premières règles stipule que toutes les balises doivent être contenues dans un nœud racine qui marque le début et la fin du document [30].

29. Pour décompresser un document au format DOCX ou ODT, il suffit de modifier l'extension du fichier en remplaçant `.docx` par `.zip` et de décompresser l'archive. À l'intérieur de ce dossier, nous y trouvons une bonne douzaine de documents au format XML contenant soit les notes de bas de page, soit les commentaires, soit le contenu du document, etc.

30. Nous attirons l'attention sur le fait que selon les versions des logiciels utilisés par les auteurs de *Contribution numérique*, la structuration des informations n'est pas réalisée selon la même version des schémas de *WordprocessingML* et d'ODF, puisque ces normes évoluent depuis

Du côté des documents DOCX, le schéma utilisé pour décrire les éléments est *WordprocessingML*, l'application de la norme *Office Open XML* créée par Microsoft. Pour les documents au format ODT, c'est la norme *Open Document Format* qui est directement utilisée pour structurer les éléments d'un document. Ces schémas nous montrent qu'en deçà de l'écran et de la page affichée, le texte est modélisé selon le principe OHCO présenté par DeRose *et al.*, que nous connaissons déjà avec les modèles SGML, HTML et XML (DeRose et al. 1990) [31]. L'application de ce principe au texte se traduit, dans notre cas, par la structuration hiérarchique des éléments du texte que l'on représente avec des balises.

Afin de savoir précisément à quelle version du schéma le document fait référence, il nous faut lire les déclarations des schémas employés parmi les attributs du nœud racine du document : `<office:document-content>` dans ODT et `<w:document>` dans DOCX.

Les déclarations des schémas sont très longues, car il est nécessaire de spécifier chaque schéma (et sa version) pour chaque type d'objet que l'on peut rencontrer dans ces documents : du texte, des images, des dessins, des graphiques, des tableaux, des images en 3D, etc.

Dans un document DOCX, les déclarations des schémas s'étendent d'une dizaine à une trentaine d'attributs [32] ajoutés à l'élément racine du document, contre en moyenne une vingtaine dans ODT.

L'attribut qui nous intéresse particulièrement pour cette étude est celui qui concerne le texte. Pour le document au format ODT il s'agit de l'attribut suivant :

Extrait de code III.1 : Espace de nom d'un fichier au format ODT

```
1  xmlns:text="urn:oasis:names:tc:opendocument:xmlns:text:1.0"
```

Pour le document au format DOCX l'attribut qui nous intéresse est :

Extrait de code III.2 : Espace de nom d'un fichier DOCX

```
1  xmlns:w="http://schemas.openxmlformats.org/wordprocessingml/
2  2006/main"
```

Dans les deux cas, les schémas mobilisés renvoient à la première documentation respectivement publiée pour chacune des normes [33].

leur première publication. Lorsque cela est nécessaire, nous spécifions l'emploi de la version de la norme à laquelle nous faisons référence. Si la mention de la version n'est pas précisée, c'est que nous faisons référence à la première version publiée de la norme.

31. Voir le chapitre II.4.

32. Un attribut est une valeur supplémentaire que l'on affecte à un nœud pour modifier son comportement.

33. La norme ODF est plus stable que la norme OOXML et contient moins de versions différentes. De plus, il est parfois difficile de s'y retrouver dans la norme OOXML, car la documentation n'est pas segmentée de la même façon d'une version à l'autre. Par exemple, la partie 4 de la première version se retrouve dans la partie 1 de la version de 2012, ce qui ne rend pas la comparaison des versions très aisée.

Au-delà des attributs affectés à cet élément, le nœud racine en lui-même est un élément fondamental de la structuration des contenus, car en plus de marquer le début et la fin du document, il est le premier élément auquel le lecteur, humain ou machine, se confronte pour connaître le type de contenu du document. Il a donc pour rôle de décrire le contenu des éléments qu'il contient. Par exemple, dans le cas d'un document au format HTML, la première balise `<html>` permet d'identifier immédiatement ce qu'elle contient, en l'occurrence du contenu HTML. En suivant ce principe pour les documents DOCX et ODT, le nœud racine du document principal décrit les contenus que l'on y trouve. Pour ODT, la racine est `<office:document-content>`, que l'on peut lire comme l'élément parent de tout le contenu du document. Si l'on se déplace dans un autre fichier de l'archive ODT, par exemple `meta.xml`, la racine de ce fichier est `<office:document-meta>` et désigne donc les métadonnées du document. Il en va de même pour les deux autres fichiers contenant les styles et les paramètres avec pour racine respectivement `<office:document-styles>` et `<office:document-settings>`. Chaque fichier apparaît alors comme un fragment documentaire et illustre le fonctionnement de l'archive comme un document composé de plusieurs sous-documents.

Dans *WordprocessingML*, le concept de document est mobilisé différemment, car il décrit explicitement le conteneur des éléments textuels du document principal et pas les autres éléments de l'archive. Si dans l'archive ODT le fichier dans lequel est stocké le texte est nommé `content.xml`, il est nommé `document.xml` pour l'archive DOCX. À l'intérieur de ce fichier, l'élément racine est `<w:document>`. Tout le contenu de ce fichier est alors pris entre les balises ouvrantes et fermantes nommées `document`. Lorsque l'on navigue dans les autres fichiers de l'archive DOCX, les racines décrivent les éléments que ces fichiers contiennent comme des commentaires, `<w:comments>` ou des notes de fin de texte, `<w:endnotes>`. Dans le schéma *WordprocessingML*, le concept de document est alors uniquement utilisé pour décrire le contenu textuel affiché dans l'interface et dans la page, et pas l'ensemble des éléments qui constituent l'objet DOCX. Cette notion fait donc uniquement référence à la dimension graphique de l'objet, comme si l'on pouvait se saisir de la page à l'écran et oublier le reste du document. Or ce fichier `document.xml` n'est pas un élément autonome. Isolé du reste de son archive, il ne permet pas de reconstituer le document dans sa totalité car il lui manque toutes les autres informations que contient l'archive pour être exécuté correctement. Ainsi, en désignant comme document les éléments visibles que l'on retrouve dans la page, *WordprocessingML* fait référence au document en tant qu'objet imprimé et pas en tant que document numérique. Cette description du document affirme en un sens l'ancrage de MSWord dans le mode WYSIWYG et dans la raison graphique.

Éléments « titre » et « paragraphe »

La deuxième différence que l'on note est la conception du texte, notamment celle des éléments principaux qui le composent, les titres et les paragraphes.

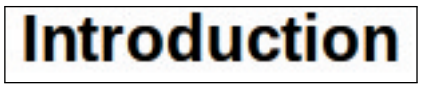


Figure III.1 : Capture d'écran d'un titre dans un fichier DOCX de *Contribution numérique*

La figure III.1, ci-dessus, est une capture d'un fragment de texte extrait d'un document au format DOCX.

Quel sens est accordé à ce fragment de texte ? Si l'on se réfère uniquement à ce qui est affiché à l'écran, c'est-à-dire depuis le paradigme de la page et de la raison graphique de Goody (1979), le sens de cet extrait de texte est limpide : il s'agit du mot *introduction* qu'un lecteur pourra interpréter comme le titre au tout début d'un texte signifiant que le contenu est introduit par le texte qui suit ce titre.

Pour autant, est-ce que cette phrase correspond à ce qui est inscrit sur le document ? Si l'on change de représentation pour celle en texte brut en décompressant le fichier au format DOCX nous observons un tout autre texte.

Les douze caractères compris dans l'image de l'extrait représentent dix-sept lignes au format *WordprocessingML* pour décrire l'élément structurel qu'est le paragraphe `<w:p>` (ECMA-376 2006 : 34).

Extrait de code III.3 : Extrait d'un paragraphe en *WordprocessingML*

```
<w:p>
    <w:pPr>
        <w:pStyle w:val="Normal" />
        <w:rPr>
            <w:b />
            <w:b />
            <w:bCs />
        </w:rPr>
    </w:pPr>
    <w:r>
        <w:rPr>
            <w:b />
            <w:bCs />
        </w:rPr>
        <w:t>Introduction</w:t>
    </w:r>
</w:p>
```

En naviguant dans la documentation de *WordprocessingML*, nous pouvons lire et interpréter le fragment de document inscrit ci-dessus. Tout d'abord, il y a le

`w:` que l'on retrouve dans toutes les balises de *WordprocessingML*. Il correspond à l'espace de nom du modèle de document. Ensuite viennent trois caractères centraux du modèle que l'on désigne par des lettres minuscules. Ces éléments en minuscule dans les balises de *WordprocessingML* ont tous une signification : le `p` signifie un paragraphe, le `r` signifie une séquence à exécuter et le `t` correspond à une plage de texte.

La lecture de ce mot, *Introduction*, se fait alors comme suit. La première balise `<w:p>` renvoie à un élément de type `paragraphe`. Au premier niveau à l'intérieur de celle-ci se trouve une balise `<w:pPr>` contenant les propriétés de ce paragraphe. La première balise enfant `<w:pStyle w:val="Normal" />` de cette déclaration des propriétés indique que le style appliqué au paragraphe a pour valeur `Normal`, ce qui indique le niveau de texte d'un paragraphe[34] dans le modèle de document de *WordprocessingML*. Ensuite, cet extrait présente trois balises qui nous indiquent des styles à appliquer au paragraphe, deux fois `<w:b/>` pour appliquer du **gras** (ECMA-376 2006 : 167) et une fois `<w:bCs />` pour appliquer du **gras** aux scripts complexes (ECMA-376 2006 : 168). Ces propriétés ne sont pas directement exécutées puisqu'elles sont déclarées dans les propriétés du paragraphe et non pas celles du texte. Pour la partie relative au texte inscrit par l'auteur, il faut se rendre au niveau de la deuxième balise enfant du paragraphe, `<w:r>`, une balise contenant une séquence à exécuter. Cette balise contient deux balises enfants, la première étant la balise `<w:rPr>`. `<w:rPr>` exécute les propriétés décrites dans les balises qu'elle contient, `<w:bCs />` et `<w:b />`, et les appliques à la balise contenant le texte, `<w:t>`.

Cet extrait nous permet de comprendre que la notion de texte dans *WordprocessingML* correspond au texte affiché à l'écran et non pas à l'ensemble des caractères que l'on peut trouver dans le document décompressé. Ces caractères que l'on affiche à l'écran sont alors désignés de façon très générique par le concept de texte (`<w:t>`), qui lui-même est obligatoirement inséré dans un paragraphe (`<w:p>`). Dans *WordprocessingML*, ceci s'applique à tous les niveaux de structuration du texte affiché à l'écran. Sans paragraphe, le texte n'existe pas dans ce document. La déclaration du niveau hiérarchique du paragraphe dans le document se fait quant à elle dans un attribut du nœud paragraphe auquel on affecte une valeur correspondant au niveau souhaité par la sélection d'un style graphique.

Cette lecture très linéaire de la modélisation de cet extrait est perturbante à plusieurs égards. D'abord, la notion de paragraphe ne réfère pas à quelque

34. Attention, comme nous le verrons plus loin, la notion de paragraphe comporte un double sens dans MSWord : elle désigne soit le paragraphe en tant que conteneur dans le modèle textuel, soit un paragraphe dans son sens le plus courant, c'est-à-dire un élément dans la hiérarchie d'un texte. Dans ce cas-ci, la valeur « Normal » appliquée à l'élément paragraphe de MSWord renvoie au paragraphe comme élément hiérarchique dans le texte.

chose de visible, mais plutôt à une forme de conteneur ou de bloc et n'est plus en rapport avec le paragraphe tel que nous le manipulons usuellement, comme un ensemble de phrases délimité, par exemple, par un alinéa et un saut de ligne. Dans la première version d'OOXML publiée par l'ECMA en 2006, le paragraphe y est défini comme

> l'unité la plus élémentaire de contenu au niveau du bloc dans un document *WordprocessingML*, les paragraphes sont stockés à l'aide de l'élément `p`. Un paragraphe définit une division distincte du contenu qui commence sur une nouvelle ligne (ECMA-376 2006 : 4).

Le paragraphe `<w:p>` est donc associé à la notion de ligne, voir en est quasiment un synonyme puisque chaque nouvelle ligne est un nouveau paragraphe.

L'extrait du document *WordprocessingML* contenant le mot « Introduction » est donc désigné comme un paragraphe dans le document. Dans *WordprocessingML*, l'affectation d'une valeur hiérarchique de l'élément paragraphe est déplacée dans un attribut de style. Dans ce cas-ci la valeur de cet attribut est `Normal`, ce qui correspond donc à un paragraphe dans son sens usuel. Le fait de lire que cet extrait est structuré comme un paragraphe stylé en **gras** ne correspond pas à la lecture que nous faisons de cet élément dans la page. Au contraire, nous nous attendons plutôt à voir un niveau hiérarchique de titre pour structurer ce mot « Introduction ». Il y a donc une différence entre la valeur affectée à l'attribut utilisé pour décrire le niveau hiérarchique du paragraphe et le résultat graphique obtenu. Cette tension entre structuration textuelle et rendu graphique n'est autre que le reflet de cette friction entre une raison graphique et une raison formelle.

Ainsi, la structuration des éléments les plus élémentaires d'un texte, comme un titre, un paragraphe ou une emphase, est déjà complexe en raison de la différence entre les concepts tels qu'ils sont définis dans le schéma *WordprocessingML*, de la définition personnelle que l'on peut leur attribuer et de l'écart créé entre la représentation graphique et la représentation formelle du document.

Dans ce cas-ci, le concept de paragraphe et sa valeur hiérarchique y sont distincts, tandis que dans d'autres modélisations d'un document, comme en HTML, l'élément paragraphe `<p>` contient à la fois le texte et son niveau hiérarchique. En se référant à la documentation la plus récente du paragraphe dans *WordprocessingML*[35], le paragraphe correspond à un espace dans lequel tout un ensemble d'informations peuvent y être contenues. Ces informations peuvent être de quatre types différents : des propriétés, des annotations (des commentaires, des révisions, des marque-pages,), des éléments personnalisés et, enfin, des éléments

35. Voir la documentation en ligne : `https://learn.microsoft.com/en-us/dotnet/api/documentformat.openxml.wordprocessing.paragraph?view=openxml-3.0.1`, consultée le 22 décembre 2024.

exécutables comme du texte, des hyperliens, des champs, etc. Le sens donné au concept de paragraphe est donc ambiguë dans *WordprocessingML*, puisque sa valeur sémantique ainsi que sa valeur hiérarchique ne correspondent pas à ce que l'on observe dans d'autres modèles textuels.

Si l'on reprend notre extrait de texte en *WordprocessingML*, le mot « Introduction » structuré en tant que *paragraphe* n'est pas une erreur de la part de l'auteur puisque le concept de *titre* n'existe pas en tant qu'élément balise dans ce modèle, comme on le retrouve par exemple en HTML (avec les balises sur 6 niveaux allant de `<h1>` à `<h6>`), mais est une propriété stylistique de l'élément paragraphe. Que l'auteur saisisse un paragraphe `"Normal"` ou qu'il affecte un style particulier pour déclarer un titre, ce n'est que la valeur affectée aux propriétés du paragraphe qui change : pour déclarer un titre de niveau 2, la propriété du paragraphe `<w:pStyle w:val="Normal" />` serait remplacée par `<w:pStyle w:val="Titre2" />`. Dans tous les cas, pour cette modélisation du document, un titre doit être interprété comme un paragraphe. Le paragraphe n'est alors plus un délimiteur d'un ensemble d'éléments de bas niveau, comme des phrases ou des objets comme des emphases ou des hyperliens, que l'on associerait à un élément plus important dans la hiérarchie des informations (comme un titre), mais comme un conteneur parent de toute une modélisation de la structure des informations textuelles dans laquelle peuvent s'entremêler des commentaires avec du texte ou des éléments de révision.

Du côté d'ODT, un extrait de texte équivalent à celui que nous venons d'observer dans DOCX se présente comme suit dans l'interface :

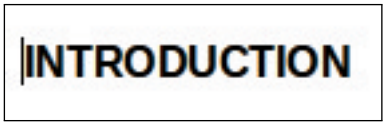


Figure III.2 : Capture d'écran d'un titre dans un fichier ODT de *Contribution numérique*

Hormis l'affichage du style, il n'y a pas de différence majeure entre le rendu ODT et celui de DOCX pour ce qui concerne la partie graphique, comme on peut le voir dans la figure III.2. Cependant, lorsque l'on observe le texte brut, les deux fichiers ne présentent pas la même structure.

Extrait de code III.4 : Structure d'un titre en ODT

```
1 <text:h text:style-name="Heading_20_2"
    text:outline-level="2">INTRODUCTION</text:h>
```

En comparaison avec la structure observée dans le document DOCX, la déclaration d'un titre en ODT tient en une seule ligne (contre 17 dans le fichier au

format DOCX). Le contenu de cette ligne doit être lu comme ceci : au début de la balise, l'élément `text` désigne l'espace de nom dans lequel l'élément se trouve, du texte dans ce cas-ci (ISO/IEC 26300:2006 2006 : 32). L'élément `text:h` que l'on trouve juste après l'espace de nom réfère quant à lui à l'objet que la balise décrit. Ici, le `h` signifie *heading* soit un titre (ISO/IEC 26300:2006 2006 : 66). Pour connaître le niveau de titre d'un élément `text:h`, il faut lire l'attribut `text:outline-level` qui doit être égal à un nombre entier pour déterminer son niveau dans la hiérarchie du document. Dans cet exemple, il s'agit d'un titre de niveau `2` (ISO/IEC 26300:2006 2006 : 66). Enfin, le dernier élément de cette balise, `text:style-name`, désigne le nom du style à appliquer à la chaîne de caractères contenue entre les balises d'un élément textuel (ISO/IEC 26300:2006 2006 : 67).

Si l'on compare avec la structure que l'on trouve dans le DOCX où tous les éléments textuels sont des paragraphes, on remarque que dans ODT, la balise d'un titre `<text:h>` est différente de celle d'un paragraphe `<text:p>`, pourtant la notion de paragraphe désigne dans ODT tout aussi bien un élément que l'autre :

> Les éléments `<text:h>` et `<text:p>` représentent respectivement les titres et les paragraphes et sont collectivement considérés comme des **éléments de paragraphe**. Tout le contenu textuel d'un fichier OpenDocument doit être contenu dans l'un ou l'autre de ces éléments (ISO/IEC 26300:2006 2006 : 66).

Pour ces deux éléments structurels, un titre et un paragraphe, la différence majeure qu'il y a entre ODT et DOCX réside dans la définition du paragraphe, car il ne désigne pas tout à fait le même objet dans les deux schémas. Du côté de *WordprocessingML*, il n'y a pas de distinction entre les éléments textuels, quel que soit le niveau, ce sont tous des paragraphes, hormis sous la forme de styles à appliquer au contenu de la balise. Tandis que du côté de la norme ODF, les paragraphes et les titres sont des éléments distincts et formellement modélisés, même si ces éléments sont plus couramment désignés dans la documentation de la norme sous l'appellation paragraphe. La valeur sémantique du paragraphe telle qu'elle est définie dans ODF s'approche plus de la notion de paragraphe utilisée dans d'autres modèles, par exemple HTML. Toutefois, elle s'en distingue par l'indifférenciation des niveaux de titre au niveau du nom de l'élément `h`. Cette information se trouve dans une propriété que l'on ajoute à la balise.

Éléments « commentaire » et « annotation »

La notion de paragraphe nous mène à un autre objet textuel essentiel à l'écriture : l'annotation ou le commentaire. Dans la documentation de *WordprocessingML*, nous avons vu que le paragraphe n'est pas seulement un élément renvoyant au texte du document, mais qu'il réfère également à d'autres objets comme le commentaire (ECMA-376 2012 : 235).

Le commentaire est un objet que nous avons mobilisé durant le processus éditorial de *Contribution numérique*, notamment pour proposer des modifications, des révisions ou demander aux auteurs des précisions sur certains passages de leur texte. C'est un élément important du processus éditorial grâce auquel il est possible d'ouvrir une discussion sur un objet ciblé dans le document. Dans les documents de travail que nous avons échangé avec les auteurs, les commentaires n'étaient pas structurés de la même manière selon si le document était au format DOCX ou ODT.

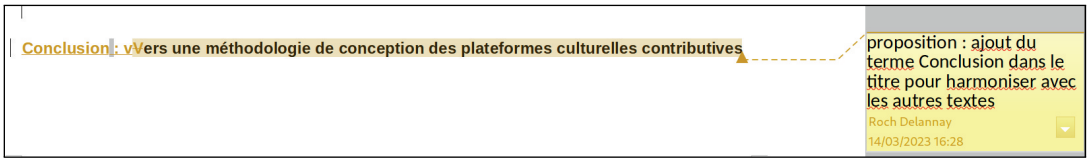


Figure III.3 : Capture d'écran d'un commentaire dans un fichier DOCX

Le premier commentaire que nous présentons est un commentaire issu d'un fichier au format DOCX. Il s'agit d'une proposition de révision en vue d'harmoniser le titre de la conclusion avec ceux des autres chapitres. Dans la figure III.3, nous avons affaire à un paragraphe avec pour valeur « titre de niveau 3 » et comportant un commentaire dans lequel nous avons proposé une légère modification du titre en ajoutant le mot « Conclusion : » au début du titre et en modifiant le « V » majuscule pour une minuscule[36].

Extrait de code III.5 : Proposition de révision dans un document au format DOCX

```
<w:p>
    <w:pPr>
        <w:pStyle w:val="Heading3" />
        <w:rPr>
            <w:b />
            <w:b />
            <w:bCs />
        </w:rPr>
    </w:pPr>
```

36. Les commentaires ont été apportés aux fichiers DOCX depuis LibreOffice. Travaillant uniquement avec des systèmes d'exploitation Unix, je préfère travailler avec des logiciels libres plutôt que propriétaires.

```
10      <w:ins w:id="0" w:author="Roch Delannay"
           w:date="2023-03-17T10:20:21Z">
11          <w:r>
12              <w:rPr></w:rPr>
13              <w:t xml:space="preserve">Conclusion: </w:t>
14          </w:r>
15      </w:ins>
16      <w:ins w:id="1" w:author="Roch Delannay"
           w:date="2023-03-17T10:42:40Z">
17          <w:r>
18              <w:rPr></w:rPr>
19              <w:t>v</w:t>
20          </w:r>
21      </w:ins>
22      <w:del w:id="2" w:author="Roch Delannay"
           w:date="2023-03-17T10:42:40Z">
23          <w:commentRangeStart w:id="0" />
24          <w:r>
25              <w:rPr>
26                  <w:b />
27                  <w:bCs />
28                  </w:rPr>
29                  <w:delText>V</w:delText>
30          </w:r>
31      </w:del>
32          <w:r>
33              <w:rPr>
34                  <w:b />
35                  <w:bCs />
36              </w:rPr>
37              <w:t>ers une méthodologie de conception des
                   plateformes culturelles contributives</w:t>
38          </w:r>
39          <w:commentRangeEnd w:id="0" />
40          <w:r>
41              <w:commentReference w:id="0" />
42          </w:r>
43          <w:r>
44              <w:rPr>
45                  <w:b />
46                  <w:bCs />
47              </w:rPr>
48          </w:r>
49  </w:p>
```

Ce commentaire, qui est en réalité une proposition de révision, est divisé en deux parties associées entre elles par un identifiant unique `w:id="0"`. Si la première partie du commentaire est présente dans le document principal avec le texte, la deuxième ne s'y trouve pas. Elle est en dehors du document principal, dans un autre document au format XML présent dans l'archive décompressée nommé `comments.xml`, sous la forme suivante :

Extrait de code III.6 : Un commentaire dans le fichier `comments.xml` de l'archive DOCX

```
<w:comment w:id="0" w:author="Roch Delannay"
   w:date="2023-03-14T16:28:48Z" w:initials="">
    <w:p>
        <w:r>
            <w:rPr>
                <w:rFonts w:ascii="Calibri"
                   w:hAnsi="Calibri" w:eastAsia="Tahoma"
                   w:cs="Tahoma" />
                <w:sz w:val="24" />
                <w:szCs w:val="24" />
                <w:lang w:val="en-US" w:eastAsia="en-US"
                   w:bidi="en-US" />
            </w:rPr>
            <w:t>proposition : ajout du terme Conclusion
               dans le titre pour harmoniser avec les autres
               textes</w:t>
        </w:r>
    </w:p>
</w:comment>
```

Dans *WordprocessingML*, le commentaire est une notion ambiguë puisque, pour une même action dans l'interface, on la retrouve dans deux documents différents : à la fois dans le document principal comme une insertion dans un paragraphe en tant que nœud enfant de l'élément paragraphe, et à la fois dans un autre document dédié `comments.xml` en tant que conteneur `<w:comment>` de l'élément paragraphe `<w:p>` qui contient le texte du commentaire (ECMA-376 : 2006 : 29). Ces deux parties du commentaire, celle dans le document principal `document.xml` et celle dans le document secondaire `comments.xml` sont reliées par un identifiant qui a pour rôle de structurer la relation entre les fragments du commentaire que l'on retrouve dans les différents fichiers XML (ECMA-376 2012 : 20). Le commentaire est donc une notion qui déborde du document principal et qui comporte des éléments qui seront imprimés et d'autres non. Si la notion de WYSIWYG nous montre dans son interface ce que l'on va obtenir lors de l'impression en insérant dans la page ce qui doit être imprimé et à l'extérieur de la page ce qui ne doit pas l'être, la définition de l'élément *commentaire* dans *WordprocessingML* n'est pas aussi franche puisqu'elle déborde dans ces deux facettes du document, à la fois comme élément que l'on va imprimer et comme élément informationnel pour l'utilisateur dans son interface.

En apparence, l'affichage du commentaire dans l'interface se présente de la même manière pour un fichier ODT ou un fichier DOCX, voir les figures III.3 et III.4. Pourtant, la norme ODF propose une autre structuration du commentaire

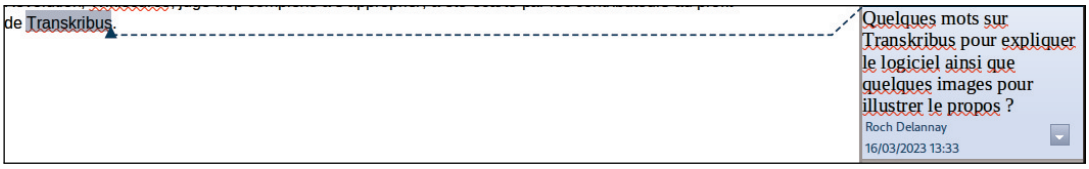


Figure III.4 : Capture d'écran d'un commentaire dans un fichier ODT

dans le document. C'est un élément structuré seulement dans le document principal, il n'est pas contenu dans un document secondaire comme c'est le cas dans l'archive DOCX. Cette modélisation est intéressante puisque ce commentaire est structuré à l'intérieur de la balise `<office:document-content>`, alors que ce n'est pas un élément que l'on veut voir imprimé dans la page au-delà d'une version de travail.

À l'intérieur du document principal, `content.xml`, le commentaire dans sa version brute se présente comme ceci[37] :

Extrait de code III.7 : Commentaire dans l'archive ODT

```
1  <text:p text:style-name="P26">
2      <office:annotation
          office:name="__Annotation__130_3346307396"
          loext:resolved="false">
3          <dc:creator>Roch Delannay</dc:creator>
4          <dc:date>2023-03-16T13:33:44</dc:date>
5          <text:p>
6              <text:span text:style-name="T66">Quelques mots
                  sur Transkribus pour expliquer le logiciel
                  ainsi que quelques images pour illustrer le
                  propos ?</text:span>
7          </text:p>
8      </office:annotation>
9      <text:span text:style-name="T6">Transkribus</text:span>
10     <office:annotation-end
          office:name="__Annotation__130_3346307396" />
11 </text:p>
```

La notion de commentaire utilisée dans l'interface d'édition est en elle-même est un abus de langage dans le cas de la norme ODF, puisque c'est un concept qui n'existe pas directement dans cette modélisation. Cependant, on retrouve l'équivalent du commentaire de *WordprocessingML* avec le concept d'annotation. Une annotation est définie dans la documentation comme un élément « contenu dans les éléments `<text:p>` et dans `<text:list>` » (ISO/IEC 26300:2006 2006 : 435).

37. Nous avons tronqué le contenu du nœud parent `<text:p>` du commentaire pour limiter l'affichage aux éléments qui nous intéressent.

À l'instar de son équivalent en *WordprocessingML*, l'annotation d'ODF est un élément enfant du paragraphe. Dans ODF, l'annotation ne peut pas être créée en dehors d'un paragraphe (un titre étant entendu comme un paragraphe) ou d'une liste. Plus qu'à un titre, un paragraphe ou une liste, cette définition nous indique que l'annotation ne peut être rattachée qu'à un élément considéré comme du texte par ODF. C'est d'ailleurs ce qu'applique également *WordprocessingML*. Cela veut dire qu'une annotation ne peut pas être rattachée à une image, à un graphique ou à tout autre élément qui n'est pas du texte.

Le sens donné à cet objet est intéressant, car que l'on soit dans un fichier DOCX ou dans un fichier ODT, il n'est possible de commenter ou d'annoter que le texte et pas les autres éléments que le schéma (ODF ou OOXML) permet d'insérer dans le document. Cela signifie que seul le texte est considéré comme un élément que l'on peut mettre en discussion. Les graphiques, les images ou les autres objets que nous pouvons insérer dans ce type de document ne sont alors pas considérés par ces schémas comme des éléments à commenter ou à annoter, car d'un point de vue formel, la discussion ne peut pas être inscrite et rattachée à ces nœuds du document.

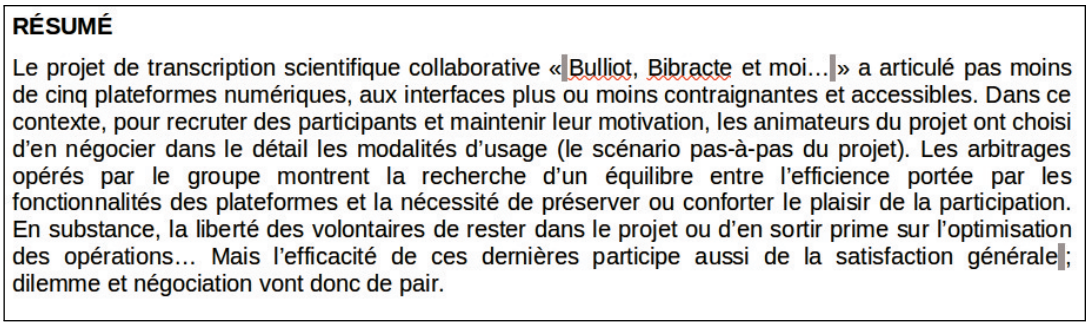
Ces deux exemples de modélisation des éléments textuels du document de traitement de texte, le paragraphe et le commentaire, mettent en exergue la vision que portent les normes qui les structurent et le sens qu'elles leur confèrent. Nous avons vu deux exemples différents de la modélisation des textes dans des fichiers DOCX et ODT. Le premier exemple témoigne, pour la norme *WordprocessingML*, d'une confusion entre la structuration graphique des informations et l'affectation d'une valeur sémantique et hiérarchique à ces dernières dans le texte. Le second exemple révèle quant à lui des ambiguïtés structurelles des informations liées à l'utilisation du commentaire. L'action de commenter ou d'annoter peut être à la fois éparse dans plusieurs espaces du document alors qu'il ne s'agit en réalité que d'une seule action du côté utilisateur, et à la fois limitée aux seuls éléments textuels du document, ne couvrant ainsi qu'une portion des éléments que l'on peut insérer dans ces modélisations. Nous allons aborder un troisième exemple, celui-ci met en évidence la non-structuration de certains éléments, puisque ces notions n'existent ni dans le modèle *WordprocessingML*, ni dans le modèle ODF.

Élément « résumé »

Toute modélisation du texte est confrontée à une limite : les modèles définissent ce que l'on peut décrire ou non, sans jamais être totalement exhaustif (c'est aussi pour cette raison que l'on crée des modèles différents). Or, par exemple, la notion de résumé ou d'*abstract*, si importante dans le monde éditorial, n'existe pas

dans le schéma de MSWord ni dans celui utilisé par LibreOffice, à moins qu'un utilisateur ne crée un style graphique correspondant. Néanmoins, le style appliqué à un paragraphe ne changera pas son niveau hiérarchique dans le document, ni sa valeur sémantique, puisque les styles ne modifient que les caractéristiques graphiques que l'on affiche à l'écran dans l'interface du logiciel de traitement de texte. Ainsi, puisque le résumé est une notion absente des modèles de texte ; un auteur ne pourra donc pas structurer cette information pour décrire formellement qu'un ensemble de caractères est un résumé. Il y a donc une rupture entre la raison graphique et la raison formelle.

Dans le cas de l'ouvrage *Contribution numérique*, le résumé fait partie des éléments demandés aux auteurs. Chaque auteur devait fournir un document contenant un ensemble de métadonnées dont un résumé. Dans un des textes au format DOCX, le résumé est déclaré comme suit :


RÉSUMÉ

Le projet de transcription scientifique collaborative « Bulliot, Bibracte et moi… » a articulé pas moins de cinq plateformes numériques, aux interfaces plus ou moins contraignantes et accessibles. Dans ce contexte, pour recruter des participants et maintenir leur motivation, les animateurs du projet ont choisi d'en négocier dans le détail les modalités d'usage (le scénario pas-à-pas du projet). Les arbitrages opérés par le groupe montrent la recherche d'un équilibre entre l'efficience portée par les fonctionnalités des plateformes et la nécessité de préserver ou conforter le plaisir de la participation. En substance, la liberté des volontaires de rester dans le projet ou d'en sortir prime sur l'optimisation des opérations… Mais l'efficacité de ces dernières participe aussi de la satisfaction générale ; dilemme et négociation vont donc de pair.

Figure III.5 : Capture d'écran d'un résumé dans un fichier DOCX

En lisant le texte contenu dans la figure III.5 depuis l'interface du logiciel de traitement de texte, nous pouvons associer le paragraphe au titre et comprendre qu'il s'agit bien là d'un résumé. Cependant, si nous observons la structure des informations dans le document, on remarque qu'il n'y a pas d'élément « résumé », mais seulement des paragraphes, un avec une valeur `titre de niveau 2` et l'autre comportant du texte. L'extrait ci-dessous a été tronqué pour mettre en évidence seulement les balises `<w:p>`.

Extrait de code III.8 : Résumé dans l'archive DOCX

```
1 <w:p w14:paraId="5A04E37F" w14:textId="77777777"
     w:rsidR="00A96967" w:rsidRDefault="004D40F9">
2      <w:pPr>
3          <w:pStyle w:val="Titre2" />
4      </w:pPr>
5      <w:r>
6          <w:t xml:space="preserve">Résumé </w:t>
7      </w:r>
8 </w:p>
```

```
<w:p w14:paraId="0898529C" w14:textId="142DF421"
   w:rsidR="00A96967" w:rsidRDefault="004D40F9">
    <w:r>
        <w:t>Le projet de transcription scientifique
           collaborative «
        </w:t>
    </w:r>
```

Cette structuration du résumé et des autres métadonnées est valable pour absolument tous les textes de *Contribution numérique*. Puisque l'élément `résumé` n'est pas descriptible dans les modèles *WordprocessingML* et ODF, les auteurs s'en remettent à une raison graphique pour écrire ces éléments. Sans cela, il ne serait pas possible d'écrire un résumé. Toutefois, selon le référentiel formel, le résumé n'a pas de définition, mais est bricolé avec celle de deux paragraphes, un titre et du texte, rédigés au début du document. Si l'on s'appuie strictement sur ce référentiel, alors le résumé n'existe pas.

Analyse des différences observées

À travers ces trois exemples de description sémantique et hiérarchique des contenus, nous constatons une distinction entre l'interprétation graphique du texte et son interprétation structurelle. C'est précisément à ce niveau que se manifeste une tension entre le texte imprimé et le texte formaté, notamment à travers plusieurs formes d'ambiguïté : la confusion entre un titre et un paragraphe dans le premier exemple, l'ambivalence du commentaire et de l'annotation, ou encore la difficulté de représenter des éléments structurels non descriptibles, comme illustré avec le résumé dans le troisième exemple.

Ce phénomène est récurrent, et pas seulement pour la structuration des éléments les plus simples comme un titre, un paragraphe ou un commentaire ; c'est également le cas pour des éléments plus complexes tels que des appareils critiques pour l'écriture en philologie [38]. De plus, pour les éléments descriptibles dans le modèle de MSWord, leur structuration est verbeuse et ambiguë si on ne la considère que depuis le texte brut, contrairement à ODF. Ce fait rendrait d'ailleurs la tâche d'écrire un document au format DOCX en dehors d'une interface WYSIWYG très ardue (Anonymous 2017) [39]. Néanmoins, si l'on reste dans la perspective de

38. Un autre environnement de traitement de texte a été développé en 1982 pour répondre exclusivement aux besoins académiques. Il s'agit du logiciel *Nota Bene*. Ce logiciel connaît encore un grand succès auprès de la communauté américaine et comprend des éléments dans son modèle que l'on ne retrouve pas dans MSWord, notamment les appareils critiques. Voir le site web : `https://www.notabene.com/index.html`, consulté le 22 décembre 2024.

39. À ce sujet, un article du portail web Open Source Observatory de la Commission européenne, mis en place pour partager des ressources et connaissances sur les questions d'interopérabilité en Europe, nous indique qu'aucun autre logiciel, à leur connaissance, n'utilise la modélisation du texte employée pour le format DOCX.

créer un document à imprimer, la lecture de la structure des éléments devient secondaire au profit de l'interprétation graphique du document, puisque, historiquement, la définition des éléments structurels de *WordprocessingML* et d'ODF se rapproche finalement plus des définitions des objets issus de l'imprimé.

Dans ce chapitre, nous nous sommes interrogés sur le sens produit dans les environnements MSWord et LibreOffice, et plus particulièrement sur leurs formats DOCX et ODT, à travers l'analyse des textes de l'ouvrage *Contribution numériques : cultures et savoirs*. Ces formats permettent de créer des documents dans un environnement de traitement de texte, c'est-à-dire un environnement dans lequel on applique des styles graphiques aux contenus du document. En ce sens, les formats DOCX et ODT s'inscrivent dans des environnements dits WYSIWYG, où le document affiché à l'écran simule le document que l'on obtient à l'impression. L'application de styles aux éléments du document devient l'action de structuration des contenus et cette structuration se traduit quant à elle par l'application de normes de formatage du texte pour que celui-ci devienne interprétable par la machine. Chaque format est régi par une norme, le format DOCX est lié à la norme *Office Open XML*, tandis que le format ODT est lié à la norme *Open Document Format*.

Dans le cadre de cette étude, une analyse comparative des documents produits dans ces environnements a été conduite. Cette analyse a permis d'observer les différences inscrites dans les documents selon les normes ODF et OOXML. Une attention particulière a ensuite été portée sur le rapport entre, d'une part, l'affichage des textes dans la page à l'écran et, d'autre part, la structuration des informations dans le document sous sa forme de texte brut.

L'analyse des documents montre que la structuration graphique des contenus en mode WYSIWYG, que ce soit au format DOCX ou ODT, est coincée dans la page que les formats tentent de représenter et qu'elle se limite à rendre compte d'une description visuelle ou graphique des éléments qui la composent.

Ce fait peut être expliqué par l'objectif initial de ces logiciels durant les années 1970 et 1980. Historiquement, les logiciels de traitement de texte étaient principalement utilisés pour produire des documents à imprimer. Toutefois, un changement s'opère au tournant des années 2000, puisque les documents produits avec du traitement de texte doivent pouvoir circuler – être lu et édité – dans l'espace numérique et gagner en interopérabilité. Pour répondre à cette demande, Sun Microsystems et Microsoft, les éditeurs d'OpenOffice et de Microsoft Word, intègrent la technologie XML sous la forme des normes que l'on vient de mentionner à l'intérieur de leur environnement d'écriture.

Si d'un côté, Sun Microsystems intègre nativement XML à OpenOffice, la firme Microsoft s'est quant à elle retrouvée dans la situation où devaient coexister

à la fois la vision du document telle qu'elle existait sous le format DOC ainsi que la vision du document sous le format DOCX dans un même schéma afin de garantir à la fois la préservation et l'interopérabilité des documents produits avec son logiciel depuis les années 1980.

Le résultat de la conjugaison de ces deux visions du document se concrétise alors dans la norme *Office Open XML* et son application *WordprocessingML* pour le logiciel MSWord. Pour réaliser cette transition, Microsoft ne se serait alors pas appuyé sur une définition des concepts mobilisés dans son schéma proche d'autres standards, comme le HTML dont s'inspire Sun Microsystems pour la norme ODF (Roig et Ribera 2021), mais sur une représentation de ces concepts qui correspond à la vision que Microsoft porte sur le document depuis ses débuts : une vision graphique du document imprimé.

Les éléments structurels sont alors définis pour rendre compte et décrire ces mêmes éléments sous leur forme visuelle. L'exemple du paragraphe illustre cette hypothèse. Précédemment, nous avons vu que la notion de paragraphe dans *WordprocessingML* ne ressemblait pas à ce que l'on attend usuellement d'un tel objet.

Extrait de code III.9 : Extrait d'un paragraphe en *WordprocessingML*

```
<w:p>
    <w:pPr>
        <w:pStyle w:val="Normal" />
        <w:rPr>
            <w:b />
            <w:b />
            <w:bCs />
        </w:rPr>
    </w:pPr>
    <w:r>
        <w:rPr>
            <w:b />
            <w:bCs />
        </w:rPr>
        <w:t>Introduction</w:t>
    </w:r>
</w:p>
```

Le paragraphe[40] n'est plus une subdivision délimitée par des marqueurs visuels tels qu'un alinéa ou un saut de ligne, mais est un conteneur hors de ce type de structuration qui délimite un espace dans la page de MSWord que l'on peut associer à une ligne informatique, et qui peut représenter une ou plusieurs

40. Le sens donné par MSWord au bloc de code signifie que le texte (`<w:t>`) "Introduction" est affiché (`<w:r>`) en gras `<w:b />`, `<w:bCs />`) dans un conteneur de type paragraphe (`<w:p>`) avec pour propriétés (`<w:pPr>`) une valeur "Normal" (`<w:pStyle w:val="Normal" />`) et des styles en gras (`<w:b />`, `<w:bCs />`).

lignes dans la page imprimée. Le paragraphe de *WordprocessingML*, en tant qu'incarnation textuelle d'un bloc contenant tous les éléments nécessaires à l'affichage du texte dans la page, peut alors prendre plusieurs valeurs comme un `"titre de niveau 1"` ou la valeur `"Normal"` qui, elles, dépendent du style que l'auteur applique au paragraphe en question.

Microsoft a ainsi donné un autre sens et une autre matérialité au concept de *paragraphe* puisqu'il s'agit de l'unité documentaire dans le modèle textuel de MSWord dont le sens est co-produit par cette définition et l'ajout d'un style graphique qui décrit la représentation graphique de cet élément (un titre, un paragraphe, un commentaire, etc.).

En comparant les éléments graphiques affichés sur la page et la structure brute des informations en XML pour ODF ou en *WordprocessingML*, nous avons la possibilité d'examiner à la fois ce que le modèle décrit, ainsi que d'observer si la représentation graphique associée correspond à la définition du modèle. Comme nous le voyons pour certains éléments, par exemple les métadonnées, les modèles textuels ne rendent pas compte de tout ce qu'il est possible d'inscrire dans une page : certains éléments comme les métadonnées d'un chapitre d'ouvrage n'y sont pas formellement décrits et ne peuvent donc pas exister dans le document. Cependant, puisque les auteurs sont dans une interface WYSIWYG, ils ont recours à une structuration uniquement graphique de ces éléments qui ne peuvent pas être décrits par le modèle et s'en remettent à des styles graphiques pour, ensuite, interpréter les contenus.

III.5 Conclusion

En conclusion, nous constatons que là où d'autres environnements d'écriture sont plus stricts et ne permettent pas d'y inscrire des informations non décrites dans le modèle textuel, par exemple la TEI en XML ; les environnements WYSIWYG MSWord et LibreOffice se trouvent être plus permissifs, car ils permettent aux auteurs de ne plus s'en remettre au modèle textuel pour écrire, mais uniquement à une structuration graphique des informations dans le texte. Cette capacité que donne le WYSIWYG aux auteurs de jongler entre les paradigmes graphiques et formels selon ce que le modèle permet ou non d'écrire donne à la fois de la souplesse en termes d'écriture, si l'on indique au lecteur de s'en remettre à une littératie de la page imprimée, mais réduit également la capacité d'expression à ce que des styles graphiques peuvent exprimer.

Ce recours à une structuration uniquement graphique des contenus crée une rupture entre la raison graphique et la raison formelle, puisque la description des éléments dans le texte brut ne correspond plus à la signification graphique de ce

dernier. Cette rupture agit comme un effet de bord du mode d'écriture WYSIWYG qui, d'une part, ne change pas le résultat obtenu lors d'une impression directe du document, mais, d'autre part, peut s'avérer être une source d'erreur si le document est traité par un autre environnement numérique.

En se positionnant dans le paradigme WYSIWYG, MSWord et LibreOffice se tiennent ainsi à la frontière entre un régime graphique et régime formel : d'un côté se tient le texte affiché à l'écran, tel qu'il pourrait se présenter sur une page, et de l'autre se trouve le texte modélisé, formaté dans sa forme prête à être calculée. Il y a une friction entre ces deux paradigmes que l'environnement tente de résoudre par la structuration graphique des informations. Nous nommons cette friction en reprenant un terme énoncé par Julien Dehut, il s'agit de la « pensée traitement de texte » (Dehut 2023 : 305).

Hérité du modèle de l'imprimerie et de la machine à écrire, la page à l'écran se retrouve alors au centre de cet environnement et intervient comme une tentative de médiation entre les régimes graphique et formel. En calquant son modèle sur celui de l'imprimé, les logiciels de traitement de texte sont des environnements où l'on peut alors choisir de décrire et de structurer des éléments dans un document soit selon un sens graphique en prévision d'une impression directe, soit selon un sens plus formel et explicite en appliquant des styles graphiques aux contenus. Cette interface graphique qui a fait le succès de MSWord dans les années 1980 est restée au cœur des usages et du sens que produisent aujourd'hui les logiciels de traitement de texte. Ce n'est pas l'interface en tant que représentation graphique qui compte mais ce qu'elle fait : elle permet aux utilisateurs d'osciller entre une forme et un format (Bachimont 2022*b*) selon les besoins, c'est-à-dire de produire du sens qui réfère soit à une littératie de l'imprimé soit à une littératie numérique. Cet effet est la « pensée traitement de texte » (Dehut 2023 : 305).

En conséquence, la production du sens dans ces environnements d'écriture dépend du positionnement de l'auteur et de la littératie qu'il convoque, numérique ou imprimé. La tentative du WYSIWYG de concilier ces deux modes d'écriture dans un seul environnement s'avère alors fonctionnelle puisque le logiciel le rend réalisable, mais laisse également la possibilité aux utilisateurs de ne se référer qu'à un seul de ces paradigmes. Le choix de l'un ou l'autre de ces modes d'écriture n'étant pas explicite rend alors confus la lecture si l'on ne sait pas à quelle littératie le document renvoie, surtout lorsque la lecture est réalisée par un autre programme de l'environnement numérique, comme cela est prévu pour remplir la condition d'interopérabilité des fichiers de traitement de texte. Lorsque le document produit se réfère au paradigme graphique de la page, la structuration des éléments n'est plus forcément explicite et peut occasionner des erreurs d'interprétation – qui n'en sont pas du point de vue computationnel – que

seul une action humaine pourra corriger ou rectifier en restructurant les éléments concernés.

Dans le chapitre suivant, nous étudions un environnement d'écriture WYSIWYM, à savoir l'éditeur de texte Stylo. Nous en profitons pour comparer les éléments observés dans MSWord et LibreOffice avec ceux présents dans Stylo, afin de mettre en évidence les différences de production du sens entre ces environnements d'écriture numérique.

Chapitre IV

Écrire avec Stylo

IV.1 Introduction

Dans le quatrième chapitre de cette thèse, nous poursuivons notre exploration de la phase de production de l'écriture numérique, un sujet que nous avons abordé en détail dans le chapitre précédent. Ce chapitre se concentre sur un environnement *What You See Is What You Mean* (WYSIWYM) (Fauchié 2024 : 315 ; Kembellec 2019), c'est-à-dire un environnement centré sur le fond et non la forme, accessible exclusivement via un navigateur connecté au Web, l'éditeur de texte Stylo. L'objectif de ce chapitre est d'analyser les modalités de production du sens dans cet environnement d'écriture numérique. Cette démarche permet de mettre en évidence les différences entre cet environnement et les environnements WYSIWYG étudiés auparavant.

Information

Stylo est un éditeur de texte sémantique en ligne développé pour l'édition savante en sciences humaines et sociales (SHS) et en lettres. Il s'agit tout autant d'un projet de recherche que d'un outil d'écriture et d'édition. C'est un outil libre et *open source* conçu en de 2017 par la CRCEN (Vitali-Rosati, Sauret et al. 2020), et soutenu depuis 2020 par la Très Grande Infrastructure de Recherche (TGIR) Huma-num. Guillaume Grossetie et Thomas Parisot, tous deux développeurs, maintiennent et développent l'infrastructure technique de Stylo en collaboration avec Maïtané Lenoir, designer UX, ainsi qu'avec la CRCEN depuis plusieurs années.

L'objectif de Stylo est de proposer une alternative au flux d'édition numérique des revues savantes en SHS. En tant qu'éditeur de texte sémantique WYSIWYM, il vise à proposer une chaîne de publication académique alternative (Kembellec

2020), tout en invitant à une réflexion théorique et pratique sur nos façons d'écrire et d'éditer.

« Le projet Stylo est né de la volonté de donner aux auteurs la maîtrise de leurs données scientifiques et de l'ensemble de la chaîne éditoriale (Debouy 2022 : § 4). » Cet objectif est réalisé à travers plusieurs fonctionnalités fondatrices ou plus récentes qui s'inscrivent dans le domaine des technologies de l'édition numérique (J. Blanc et Haute 2018) : balisage du texte pour une structure sémantique fine, import de données bibliographiques structurées depuis l'application Zotero, mot-clés contrôlés depuis plusieurs ontologies, prévisualisation avec la possibilité d'annoter avec Hypothesis, génération de plusieurs formats (HTML, PDF, XML, XML TEI, ODT ou DOCX, PDF), export respectant les standards de l'édition scientifique, par exemple le format XML compatible avec le schéma COMMONS Publishing commun à Métopes et OpenEdition, fonctions avancées de rechercher-remplacer, édition collaborative simultanée, accès aux données via une *Application Programming Interface* (API) GraphQL, etc (Delannay, Fauchié et al. 2023).

Au cœur de Stylo ce sont donc les formats de balisage Markdown, de sérialisation de données *YAML Ain't Markup Language* (YAML) ou encore de structuration de références bibliographiques BibTeX qui offrent la possibilité de produire plusieurs formats de sortie depuis une source unique. Il existe un terme dans le jargon éditorial pour définir cette capacité d'une chaîne éditoriale à produire plusieurs objets publiables à partir d'une seule source. Il s'agit de la notion de *single source publishing* (Fauchié 2024). Stylo suit ce principe. En s'appuyant sur Pandoc, un outil de conversion de documents désigné comme le couteau suisse de l'édition, le module d'export de Stylo génère les formats de sortie susmentionnés.

Le choix d'étudier Stylo comme terrain pour cette recherche découle de plusieurs raisons. Tout d'abord, il s'agit d'un éditeur moderne construit avec les technologies du Web les plus récentes. Que ce soit à travers des environnements tels que Stylo, GoogleDoc, Hedgedoc ou encore Framapad, les environnements d'écriture en ligne (Web) suscitent un certain engouement auprès des utilisateurs, notamment pour leur capacité à offrir un espace de travail collaboratif en temps réel leur permettant d'écrire à plusieurs dans cet espace. La deuxième raison qui fait de Stylo un terrain opportun est l'accessibilité de son code source. Contrairement à d'autres éditeurs propriétaires comme l'est GoogleDoc, la totalité du code source de Stylo est disponible en ligne, ce qui est indispensable pour notre étude. Enfin, le fait d'être impliqué dans les développements de Stylo depuis plus de trois années m'offre une position privilégiée pour étudier cet éditeur, puisque j'ai accès aux différentes phases de test de développement, me permettant

ainsi d'observer le comportement des nouvelles fonctionnalités et de participer à leur modification. Depuis cette position, j'ai également un accès direct à la communauté d'utilisateurs, s'élevant à un peu plus de 6000 comptes utilisateurs fin 2023 pour plus de 40000 documents différents.

À partir de ces éléments contextuels, et afin de traiter notre problématique en réalisant une biographie de Stylo, nous avons structuré ce chapitre en trois sections. La première est consacrée à une brève histoire de Stylo et à son inscription dans les environnements d'écriture sur le Web. Contrairement aux logiciels que nous avons cité dans le chapitre III.2 et que l'on voit installés sur les ordinateurs dès la fin des années 1970 et le début des années 1980, il faut attendre le début des années 2000 et les prémices du Web 2.0 pour réellement écrire dans un éditeur web. Des premiers éditeurs créés à partir de balises HTML `<textarea>` survitaminées, jusqu'aux structures de données conçues pour éviter les conflits (*Conflict-Free Replicated Data Type* (CRDT)) et permettre l'écriture collaborative synchrone, Stylo hérite de ces avancées technologiques et s'inscrit dans cette histoire. Dans cette perspective, il apparaît opportun de rappeler succinctement certains aspects de l'histoire du Web afin d'appréhender les choix et les technologies mises en œuvre dans Stylo.

La seconde section concerne l'exploration de la modélisation des informations en texte brut, en considérant les formats spécifiques employés dans le cadre de Stylo. Nous profitons de cette section pour comparer certains des éléments structurels modélisés dans ces formats en texte brut avec ceux étudiés dans le chapitre III. La traversée des différents formats de texte brut est cette fois réalisée en s'appuyant sur l'écriture de l'article dans Stylo qui est paru dans le numéro 8 de la revue *Humanités Numériques* publié en 2023. Notre choix s'est porté sur cette revue parce qu'elle a intégré Stylo à l'occasion de ce numéro dans son flux de travail, ce qui nous offrait un cadre idéal pour échanger régulièrement sur les questions d'écriture et d'utilisation de Stylo avec les membres de l'équipe éditoriale. Nous avons choisi cet article parce qu'il a été le premier à intégrer Stylo dans le processus de la revue.

Enfin, dans la dernière section, nous nous concentrons sur la circulation des informations dans tout l'environnement d'écriture qu'est Stylo. Puisqu'il s'agit d'un environnement web, le texte, ainsi que tout un ensemble d'informations, doivent pouvoir être échangés et circuler entre *a minima* le navigateur de l'utilisateur et le serveur d'Huma-num sur lequel les données sont stockées. Ceci implique des manipulations et des transformations à l'échelle du document propre à cette surcouche qu'est le Web et que nous ne trouvons pas dans d'autres circonstances. Ainsi, c'est à travers les différentes techniques, technologies et protocoles

mobilisés dans ce dispositif que nous étudions le processus de matérialisation du document dans Stylo et le sens que celui-ci produit.

IV.2 Brève histoire de Stylo

L'histoire de Stylo, ainsi que les choix de conception qui ont été réalisés par le CRCEN s'inscrit dans l'histoire, plus large, du Web. En effet, Stylo est un éditeur de texte qui n'est exécuté qu'en environnement web. Stylo est hébergé sur un des serveurs d'Huma-num à Lyon et est rendu accessible en ligne depuis une URL : `stylo.huma-num.fr`[1]. Son interface, ainsi que les protocoles de circulation des informations entre les différents modules qui le composent, s'appuient intégralement sur les technologies du Web : les langages HTML, *Cascade Style Sheets* (CSS), JavaScript (JS), ainsi que le protocole HTTP, etc. Ce faisant, certaines particularités de l'histoire du Web deviennent significatives quant à la production de l'écriture dans Stylo, puisque ce sont elles qui permettent au document de se matérialiser. Il apparaît dès lors opportun de rappeler ces différentes spécificités du Web afin d'appréhender le sens produit durant l'écriture d'un article dans Stylo.

À la différence d'un espace fermé comme peut l'être un espace informatique local coupé du réseau, le Web est un environnement ouvert et accessible pour celles et ceux qui ont un accès à Internet. Plus largement, le Web est une application d'Internet permettant le transfert d'informations selon un réseau décentralisé. Son architecture repose sur deux éléments fondamentaux : un protocole d'adressage des ressources (IP) et un protocole de transfert des informations (TCP).

Alain Mille dresse l'histoire depuis les débuts d'Internet dans les années 1960 à partir du réseau filaire ARPAnet développé par le département de la défense américaine (Mille 2014). À ce stade précoce, comme le souligne Mille, il manque une brique technologique pour que naisse l'Internet : un protocole de transfert des documents. Le premier protocole a vu le jour en 1969[2] et a fait l'objet de la première *Request for comments* (RFC)[3] avant de trouver une forme plus aboutie dans le protocole TCP en 1974 – décrit par Vincent Cerf et Bob Kahn – et permet,

1. Il ne s'agit là que de l'instance principale de Stylo. Il est tout à fait possible de déployer Stylo sur un autre serveur et sous un autre nom de domaine.

2. Il n'y a pas de corrélation directe avec l'utilisation de la norme ASCII par les institutions américaines à cette date, néanmoins on remarque qu'il y a un engouement pour l'informatique à la fin des années 1960.

3. On peut retrouver tout le contenu de cette RFC sur cette page web : `https://www.rfc-editor.org/rfc/rfc3.html`, page consultée le 25 septembre 2024. Les RFC sont des documents numérotés, dans ce cas-ci, il s'agit de la RFC 3, et vont par ordre croissant. Une modification d'un document numéroté fait l'objet d'un nouveau numéro et rend obsolète la version antérieure. Comme on peut le voir sur la page web, la RFC 3 est rendue obsolète par la RFC numéro 10, puis la 16, etc.

avec sa distribution sous forme de paquets, la naissance d'Internet. Ce n'est qu'en 1990, à l'Organisation Européenne pour la Recherche Nucléaire (CERN), que Tim Berners-Lee participe à la conception du Web – et du *World Wide Web* – pour pallier le problème d'échanges de documents numériques rencontré dans cette institution grâce au développement du langage de balisage HTML. Le Web vient donc répondre à un besoin, celui de la compatibilité des informations et de leur interopérabilité dans une structure. En créant un environnement spécifique composé de normes de structuration des informations interprétables par un logiciel, le navigateur, le Web devient agnostique et ne dépend plus de la même couche d'abstraction matérielle qu'un environnement local. L'ordinateur devient un terminal, un client à partir duquel on peut se connecter au réseau et accéder aux informations qui y circulent.

C'est ainsi que sur le Web, l'action de stockage des données est généralement séparée de l'espace d'affichage dans le navigateur. Les données sont habituellement stockées sur un serveur, voire dans une base de données sur ce serveur. Il y aurait donc au moins deux parties différentes pour écrire dans un espace web, la partie client – ce qui est affiché dans le navigateur – et la partie serveur où les informations sont stockées et organisées. Ce fonctionnement, devenu trivial en 2025, nous intéresse particulièrement dans notre contexte parce qu'il conditionne le bon déroulé de l'écriture numérique en ligne sous réserve que deux facteurs soient présents : une connexion au réseau et un protocole de communication entre le client et le serveur.

Le Web est donc conçu comme un système hypertexte décentralisé permettant en tout premier lieu d'accéder à des documents pour les consulter, c'est-à-dire qu'il s'agissait au départ d'un espace de lecture. À ses débuts, le Web ne permet pas vraiment d'écrire. Le fait de pouvoir écrire dans une page web est une fonctionnalité apparue dans un second temps, entre 1990 et 1995, notamment avec l'ajout de l'élément `<form>` en HTML que l'on voit apparaître dans la RFC 1866 publiée en novembre 1995 et contenant les spécifications de HTML 2.0. Cette première forme d'écriture s'apparente plus à de la saisie d'informations dans des formulaires qu'à la rédaction d'un manuscrit ou d'un article.

Pendant les premières années du Web, celui-ci n'était pas vraiment dynamique, parce qu'il n'y avait pas de standard pour le faire, et dépendait donc des éditeurs de navigateur. Chaque navigateur implémentait les fonctionnalités dont il avait besoin sans qu'il n'y ait de portage des fonctionnalités d'un navigateur à un autre. Il faut attendre la première publication du langage de programmation JavaScript en décembre 1995, le langage le plus utilisé par les navigateurs aujourd'hui, pour que tous les navigateurs reposent sur la même base avec les trois langages pivots du Web : HTML, CSS et JS.

La RFC 1866 précise la définition de l'élément `<form>`[4]. Grâce à cet élément, il devient possible d'envoyer des données au serveur depuis le client (le navigateur) via, par exemple, une requête HTTP. Cette fonctionnalité permet donc d'écrire du texte dans des zones très ciblées d'une page web, et ce texte peut ensuite être envoyé sur un serveur. Si le formulaire prend en général comme entrée des chaînes de caractères relativement courtes, même lorsque le type d'entrée du formulaire est `type="text"`, la RFC 1866 recommande l'emploi de la balise HTML `<textarea>` pour des chaînes plus longues.

HTTP est un protocole de communication conçu par Tim Berners-Lee et le W3C en même temps que les autres technologies du Web au début des années 1990. La première spécification d'HTTP date de 1996 avec la RFC 1945[5]. Ce protocole a été pensé comme un moyen de communication entre des agents du Web, clients ou serveurs, mais également comme une passerelle avec d'autres protocoles d'Internet, par exemple *Simple Mail Transfer Protocol* (SMTP) ou *File Transfer Protocol* (FTP). Pour la circulation des informations, ou plutôt des requêtes, entre un client et un serveur, le protocole HTTP prévoit neuf méthodes pour couvrir tous les cas d'usage. Huit de ces méthodes sont définies dans les RFC 9110 (la dernière version publiée en juin 2022)[6], et la neuvième, `PATCH`, fait l'objet d'une RFC séparée, la RFC 5789.

Parmi ces méthodes de circulation des informations entre un serveur et un client, le protocole HTTP en comporte deux bien connues : `GET` et `POST`. Chacune de ces méthodes a un comportement différent. Lors de la transmission des informations par la méthode `GET`, toutes les informations sont insérées dans l'URL ce qui 1) les rend visibles (et vulnérables) et 2) impose une limite du nombre de caractères (aux alentours de 2000 au maximum) au risque de déclencher une erreur 414 dans le navigateur (URL trop longue)[7]. Pour ces raisons, la méthode `GET` n'est pas utilisée dans Stylo. Tandis qu'en utilisant la méthode `POST` pour envoyer ou récupérer des informations, celles-ci ne seront ni visibles ni limitées en longueur.

La méthode `POST` est définie dans le paragraphe 9.3.3 de la RFC 9110 comme suit :

4. Voir la page web : `https://www.rfc-editor.org/rfc/rfc1866#section-8`, consultée le 28 février 2025.

5. La RFC 1945 est consultable à l'URL suivante : `https://datatracker.ietf.org/doc/html/rfc1945`, page consultée le 13 mars 2025.

6. Pour plus de détail sur les méthodes du protocole HTTP, se rendre à cette adresse : `https://www.rfc-editor.org/rfc/rfc9110#name-method-definitions`, page consultée le 13 mars 2025.

7. Pour plus d'informations sur l'erreur 414, voir la RFC 9110 à l'adresse suivant : `https://www.rfc-editor.org/rfc/rfc9110#name-414-uri-too-long`, page consultée le 13 mars 2025.

> La méthode `POST` demande à la ressource cible de traiter la représentation incluse dans la demande selon sa propre sémantique. Par exemple, la méthode `POST` est utilisée pour les usages suivants (parmi d'autres) : Fournir les blocs de données, comme les champs d'un formulaire HTML, à un traitement de données ; Publier un message sur un tableau d'affichage, un groupe d'échange, une liste de diffusion, un blog ou un groupe d'articles similaire ; Créer une nouvelle ressource qui n'a pas encore été identifiée par le serveur d'origine ; et Ajouter des données à la (aux) représentation(s) existante(s) d'une ressource.

À travers cette brève définition, on remarque que l'usage principal de la méthode `POST` est plutôt relatif à l'envoi d'informations, qu'elles soient nouvelles ou mises à jour. Le comportement de `POST` fait toutefois débat, notamment quant à son usage pour l'envoi de certaines informations puisque, comme cela est indiqué dans sa définition, `POST` laisse le soin au serveur (la ressource cible) de traiter les données contenues dans son message selon sa propre sémantique. En somme, contrairement à d'autres méthodes comme `PUT`, `POST` n'est pas idempotente[8], ce qui pourrait entraîner des différences de résultat lors de l'exécution d'une requête. Par exemple, la duplication d'une requête en cas de problème de connexion. Au contraire, une méthode idempotente comme `PUT` ou `DELETE` aurait le même effet du côté du serveur et cela quel que soit le nombre de fois que cette requête lui aura été envoyée : une ressource supprimée ne le sera qu'une fois.

À ce stade, nous savons saisir du texte dans une page web et l'envoyer sur un serveur distant pour le stocker grâce au protocole de communication HTTP. Cependant, il s'agit juste de prendre en charge des textes, mais pas forcément de pouvoir les styliser comme dans MSWord ou LibreOffice, ni de pouvoir les structurer selon la syntaxe du format Markdown ou celle d'un autre format. Le texte est aussi brut que possible sans réelle structure sémantique. Le Web n'est donc pas encore un environnement d'écriture pour des textes complexes, car il n'y a pas de modèle textuel adapté à cette complexité.

Il faut attendre 2003 pour qu'apparaisse une technologie permettant de structurer des informations dans un navigateur. Le logiciel qui réalisait ceci s'appelait FCKEditor. Il est le premier de sa catégorie, que l'on connaît sous l'appellation éditeur de texte enrichi en ligne (*rich text editor*). FCKEditor est une librairie JavaScript permettant de transformer et d'améliorer l'élément HTML `<textarea>` d'un formulaire en éditeur de texte WYSIWYG[9]. Le principe de cet outil est de permettre d'éditer des contenus structurés par l'application de styles

8. L'idempotence signifie qu'une opération a le même effet et cela quel que soit le nombre d'applications de cette opération.

9. Voir le site web : `https://www.rfc-editor.org/rfc/rfc1866#section-8`

graphiques, afin de créer des contenus à destination du Web. Par exemple, il devient possible de créer des CMS qui s'appuient sur cette technologie, ou de produire des contenus destinés à des blogs, des forums, des réseaux sociaux et autres plateformes en ligne.

À la suite de FCKEditor, et grâce à l'amélioration des spécifications d'HTML (depuis l'arrivée de HTML5), d'autres initiatives ont vu le jour comme CodeMirror[10], ou alors l'éditeur de texte Monaco. Ces deux éditeurs de code ont été initialement conçus pour éditer du code informatique. Monaco est par exemple utilisé à l'intérieur de l'éditeur de code VSCode (et sa version *open source* VSCodium) de Microsoft, ou encore dans Stylo. Stylo s'appuyait initialement sur CodeMirror pour permettre l'écriture dans cet environnement ; toutefois, lorsque la fonctionnalité d'écriture collaborative synchrone a été implémentée en 2023, l'utilisation de Monaco semblait alors plus stable. Ce dernier a remplacé CodeMirror dans Stylo pour cette raison.

Plus que du code informatique, ces outils permettent d'écrire du texte dans une page web, et cela sans trop de limite de caractères. Si CodeMirror s'appuie sur le même principe de fonctionnement que FCKEditor (devenu ensuite CKEditor), en modifiant le comportement d'un élément `<textarea>` dans le HTML grâce à du JavaScript, Monaco utilise quant à lui une toute autre méthode.

L'approche de Monaco ne consiste pas en la manipulation d'un élément de la page HTML, mais en la manipulation du *Document Object Model* (DOM) de la page HTML manipulée par le navigateur, qui représente directement toute la structure de cette page web, c'est-à-dire l'architecture des éléments HTML[11]. Lorsque l'on écrit dans Monaco, le texte n'est pas uniquement ajouté à la page HTML, mais modifie la structure même du document et de toute la page. Cette modification ne prend effet que dans le document tel qu'il est exécuté par le client dans le navigateur et ne change en rien la structure HTML du fichier stocké sur le serveur.

Ces deux approches, celle de la modification du comportement de la balise `<textarea>` et celle de la modification du DOM, sont les deux approches les plus répandues parmi les éditeurs de texte en environnement web. Dans un cas comme dans l'autre, l'inscription du texte se fait au sein d'un document HTML, que l'on soit en mode WYSIWYG ou en mode WYSIWYM. Écrire dans un navigateur, c'est donc écrire selon le modèle de texte que comprend le Web pour que ces informations puissent être interprétées dans la page HTML, mais également pour qu'elles puissent circuler sur ce réseau d'informations.

10. Voir le page web de CodeMirror : `https://codemirror.net/`, consultée le 28 février 2025.
11. Voir le site web dédié à l'éditeur Monaco : `https://microsoft.github.io/monaco-editor/`, page consultée le 25 septembre 2024.

Le Web n'est donc pas un environnement trivial lorsque l'on parle d'écriture. En tant que couche d'abstraction matérielle supplémentaire entre l'humain et la machine, le Web incarne une modélisation du texte avec sa propre conception des documents, leur interprétation, leur circulation à l'intérieur de cette couche, et un formatage spécifique des informations.

L'ajout de cette couche significative supplémentaire, par rapport aux cas d'étude du chapitre précédent où les logiciels étaient installés « en local », accroît la complexité du sens accordé à ces documents, notamment parce qu'ils ne sont plus uniquement destinés à rester au sein de cette couche, et doivent être interprétés dans d'autres espaces du numérique, comme c'est le cas de l'article de la revue *Humanités numériques* produit dans Stylo, puis importé sur la plateforme OpenEdition Journals via le CMS Lodel[12]. Ainsi, les documents produits avec Stylo ne sont pas seulement interprétés dans une page web : ils sont généralement exportés pour être publiés dans des revues, ou pour produire des mémoires, des thèses ou encore des billets de blog, par exemple.

Utiliser le Web pour rendre accessible une application comme Stylo c'est également donner un accès rapide à cet environnement d'écriture aux utilisateurs, et cela quel que soit le système d'exploitation installé sur leur ordinateur, car il n'y a aucun logiciel à installer sur sa machine pour accéder à Stylo. Les seuls pré-requis sont un accès à Internet et un navigateur. Il s'agit là aussi d'un des enjeux initiaux du projet de recherche.

En effet, si l'amorce initiale du partenariat avec Huma-num date de 2017, et plus précisément d'un échange entre les deux parties qui s'est tenu lors du congrès DH2017, comme me le mentionnait Stéphane Pouyllau (Huma-num) durant un entretien mené en janvier 2024[13], l'équipe de la CRCEN avait déjà entamé plusieurs réflexions sur le sujet plusieurs années auparavant. Un billet de blog posté en 2015 par Marcello Vitali-Rosati témoigne de ces réflexions (Vitali-Rosati 2015*a*).

Le premier constat qui y est écrit relate le fait qu'il n'y a pas d'environnement d'écriture adapté aux besoins de l'écriture savante, hormis des solutions mobilisant un paratexte très verbeux (LaTeX, XML). En parallèle, les solutions de traitement de texte présentent également des limites en termes de modélisation de l'écriture, mais également en termes de transformations pour que le texte soit publié en format numérique. En effet, comme nous l'avons vu dans le chapitre III, des solutions comme LaTeX ou de traitement de texte sont orientés pour de l'impression. Leur finalité est donc graphique et ne se prête pas à une conversion

12. Nous revenons plus en détail sur cette partie du traitement éditorial de l'article rédigé dans Stylo dans le chapitre V.

13. La retranscription de l'entretien se trouve en annexe, voir V.5.

dans les formats requis pour les plateformes de diffusion telles qu'Érudit ou OpenEdition[14]. Le point principal abordé par l'auteur révèle alors un angle mort des environnements d'écriture : ils ne sont pas prévus pour écrire à des fins de diffusion numérique.

Ce constat peut être étendu aux environnements d'écriture en ligne dérivés de FCKEditor. Ils sont principalement utilisés pour écrire sur des plateformes en ligne, mais leur usage ne concerne pas l'écriture savante. Chaque environnement d'écriture répond à certains besoins particuliers. Par exemple, les réseaux sociaux sont orientés sur le partage de contenus brefs, ainsi que de contenus multimédias. L'espace de saisie des contenus n'est pas adapté à l'écriture d'un article scientifique, car les éléments textuels nécessaires à ce type d'écriture n'y sont pas modélisés.

De plus, il faut également noter qu'un éditeur de texte uniquement en Markdown ne répond pas plus à ce besoin, car la problématique soulevée dans ce billet de blog ne concerne pas l'usage d'un format ou d'un autre. Il ne s'agit pas de savoir si DOCX, ODT, HTML ou Markdown est mieux que les autres formats pour écrire, mais d'établir un environnement d'écriture qui répond à des besoins spécifiques, c'est-à-dire qui modélise les informations dont l'auteur a besoin.

Pour échapper à cette question du format employé, l'auteur souligne la nécessité de pouvoir convertir le document produit en différents formats, et cela afin de respecter les modalités de réception des documents savants émises par les éditeurs, les diffuseurs, les laboratoires de recherche, etc. En somme, cette vision d'un environnement d'écriture peut être résumée par le fait de ne pas imposer ni prescrire un mode de production de l'écrit, ou plutôt d'en proposer un qui puisse être compatible avec d'autres modélisations considérées comme des standards.

Trois années plus tard, tandis que les premières versions du prototype de Stylo sont déployées sur le Web, l'expression de cette problématique est exprimée à nouveau, dans un autre billet du même auteur, en l'associant ce coup-ci à ces premiers prototypes (Vitali-Rosati 2018*b*). Reprenant la vision précédente, le point clef de ce billet se situe dans la formulation suivante : « l'éditeur doit être compatible avec d'autres formes d'écriture ». Cela veut dire que la structuration des informations dans ce prototype doit pouvoir être transposée à différents cas de figure : du traitement de texte, donc du XML, du Web (HTML), du PDF, etc. Pour réaliser cela et répondre aux besoins de l'écriture savante, c'est-à-dire renseigner des métadonnées riches, baliser des éléments infratextuels complexes et donner cette souplesse de conversion à différentes formes d'écriture, trois

14. Nous abordons les plateformes de diffusion dans le chapitre V.3.

formats de texte brut ont été implémentés dans Stylo[15], le Markdown, le YAML et le BibTeX.

Le choix du Markdown comme format d'écriture n'est pas anodin. Dans les années 2010, on remarque un engouement pour ce format lorsqu'il s'agit d'écritures destinées à être diffusées dans un environnement numérique. En 2014, Nikola Sander en fait l'éloge sur son blog personnel[16] pour sortir du carcan de MSWord et de LaTeX, qu'elle considère inadaptés à ses besoins. Cet emploi du Markdown n'est pas marginal ou isolé à cette époque, c'est un phénomène qui se répand et autour duquel s'agrège des communautés de pratiques. Sander et l'équipe de la CRCEN en référence beaucoup.

L'intérêt porté à ce format, comme nous le voyons dans la section suivante, est qu'il a été pensé comme une version allégée du HTML, l'idée étant de s'en servir pour pouvoir convertir facilement les contenus en HTML. Si le Markdown n'a pas de schéma précis sur lequel s'aligner, c'est en quelque sorte parce qu'il s'aligne sur le format HTML et sa DTD.

Ce trait caractéristique du Markdown est énoncé dans le billet de blog de Vitali-Rosati :

> Stylo permet d'utiliser une sémantique ouverte et customisable et de créer facilement des structures de balisage du document personnalisées, sans être lié à un schéma prédéfini - comme dans le cas du xml (ce qui justifie le choix du html comme format pivot.)

Néanmoins, cette phrase n'est pas totalement juste, parce qu'elle omet de préciser que le Markdown isolé n'est pas capable de cette prouesse qu'est la personnalisation sémantique des contenus, car il n'est qu'une simplification du format HTML. Pour réaliser cela, il est nécessaire de s'abstraire du format et de la modélisation qu'il propose à un autre niveau de représentation du texte et des éléments sémantiques qu'il comporte. C'est là le rôle d'une technologie tierce implémentée dans Stylo (et que l'on retrouve également chez Sander) : il s'agit du logiciel de conversion Pandoc[17].

Pandoc[18] permet de convertir des fichiers d'un format de balisage vers un autre format de balisage. Puisque la majorité des documents sont structurés en suivant le principe OHCO, que ce soit avec du XML pour le traitement de texte ou pour la TEI, ou encore pour les documents publiés sur le Web avec le format HTML, l'emploi du Markdown comme format léger d'écriture, que l'on couple à

15. Nous revenons plus en détails sur ces formats dans la section IV.3.

16. Voir son blog à l'URL suivante : `https://web.archive.org/web/20150216192428/http://nikolasander.com/writing-in-markdown/`, consultée le 17 juin 2025.

17. Pandoc est un incontournable pour transformer des documents. Il a été développé et maintenu en Haskell par son créateur John MacFarlane depuis 2006.

18. Nous revenons plus en détail sur le logiciel Pandoc dans la section V.2.

Pandoc permet ainsi de remplir un des objectifs de Stylo, celui de ne pas imposer de forme d'écriture. Le tour de force de Pandoc consiste à interpréter la syntaxe Markdown qui permet de déclarer des classes à des éléments textuels, c'est-à-dire des attributs de type `class` en HTML, puis de donner la possibilité aux utilisateurs de convertir ces classes en un autre élément. Par exemple, un paragraphe en Markdown pourrait être affublé de la classe `epigraphe` pour signifier que ce paragraphe-là n'est pas comme les autres : il s'agit d'une épigraphe placée en en-tête du document. Il n'a ainsi plus la même valeur sémantique que les autres paragraphes. Grâce à cette fonctionnalité, il est possible de personnaliser la structuration sémantique des éléments textuels en Markdown et de la convertir pour correspondre à des besoins très particuliers adaptés à ces autres formes d'écriture savante. En ce sens, l'assertion de Vitali-Rosati n'est pas tout à fait juste, car il y a bien un modèle qui ne porte pas le nom de schéma auquel est rattaché Stylo : il s'agit du format Markdown (et de la DTD du format HTML) étendu aux capacités de traitement du logiciel de conversion Pandoc.

Cette première approche de Stylo est toutefois trop réductrice, car elle ne prend pas en considération la matérialité du Web. En effet, la combinaison des technologies et des formats présentée ci-dessus pourrait très bien être réalisée sur un ordinateur en local, sans passer par l'installation d'un logiciel supplémentaire. Quelques lignes de commande et un éditeur de texte seraient amplement suffisants pour reproduire un modèle textuel très similaire. Néanmoins, Stylo n'est pas installé en local, il est accessible depuis le Web, fait qui sous-entend qu'il y a un serveur sur lequel les données sont organisées et conservées, ainsi qu'un protocole de communication entre le client web appelé depuis un navigateur et cette base de données.

L'architecture de Stylo, et le modèle textuel qu'il propose, repose donc à la fois sur les formats de texte brut que sont Markdown, YAML et BibTEX, mais bénéficie également de toute cette architecture qui constitue le Web. On y retrouve les parties serveur et client, avec le client qui permet d'afficher l'interface de l'éditeur et d'accéder aux données stockées sur le serveur. Du côté du serveur, en plus de la base de données, on trouve un autre service : le module d'export qui permet quant à lui d'extraire les documents contenus dans Stylo hors de cet environnement au moyen du logiciel de conversion Pandoc. L'architecture logicielle de Stylo peut donc être scindée en trois parties (voir la figure IV.1).

Tout d'abord, nous retrouvons la base de données où sont stockées toutes les informations et données de Stylo : les comptes utilisateurs, les articles, les espaces de travail, les corpus, etc. Cette base de données est réalisée avec MongoDB, un système de gestion de base de données non relationnelles développé en 2007 et

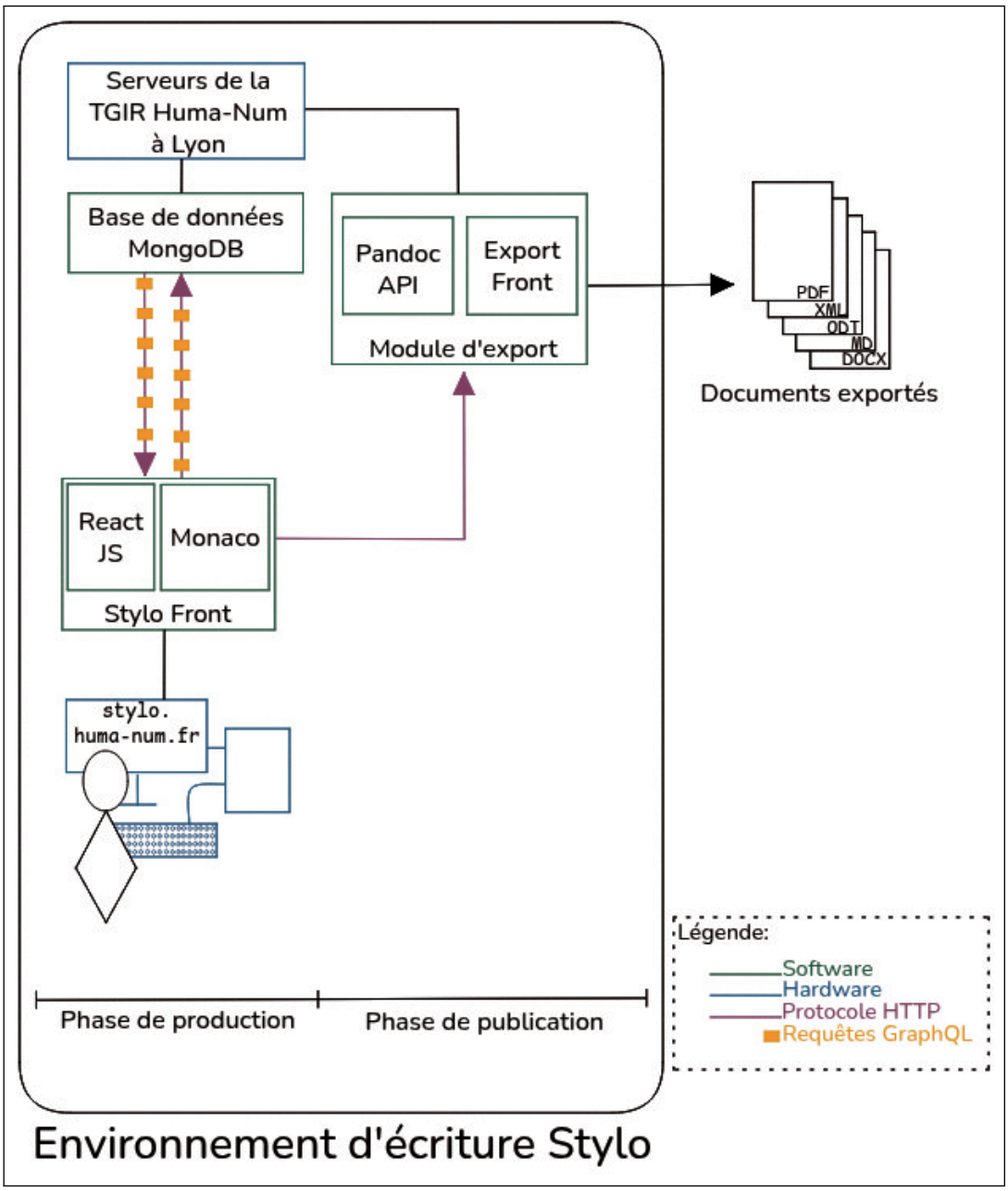


Figure IV.1 : Schéma de l'environnement Stylo

s'appuyant sur des documents structurés au format *JavaScript Object Notation* (JSON).

Le deuxième bloc de Stylo est le module d'export qui permet de transformer les informations saisies et visibles dans l'éditeur en de multiples documents. Tout ce module est développé et maintenu avec le langage de programmation Python par David Larlet. Cette brique technologique est articulée autour du logiciel de transformation et de conversion Pandoc déployée sur un serveur et rendue accessible via une autre API[19] fabriquée à partir du framework FastAPI[20]. Le module d'export intégré à Stylo[21] permet de convertir et de transformer les textes sources en une multitude d'artefacts, selon les capacités de transformation et de conversion du logiciel Pandoc auquel il est rattaché. La phase de publication inscrite sur le schéma IV.1 est représentée finie à l'intérieur de l'environnement Stylo, mais rien n'empêche d'effectuer d'autres transformations des documents exportés avec Stylo avant de les publier.

Le dernier bloc de Stylo concerne l'interface que les utilisateurs voient affichée sur leur écran. Étant donné que Stylo est accessible via un navigateur web, l'interface a été conçue avec les technologies de cet environnement. On retrouve des objets en HTML, en CSS et en JS. Le *framework* React, une surcouche à Javascript *open source* développée par Facebook (aujourd'hui Meta) en 2013, a été employé pour construire les différents composants de l'interface et intégrer de nombreuses librairies telles que *i18n* qui permet d'implémenter le multilinguisme dans l'interface et de changer la langue affichée à l'écran en un seul clic. L'éditeur de texte, pièce maîtresse de l'interface Stylo, s'appuie dans ses versions les plus récentes sur la technologie Monaco développée par Microsoft et rendue disponible sous licence MIT. C'est grâce à ce composant qu'il a été possible d'intégrer l'écriture collaborative synchrone dans Stylo.

Cette brève histoire de Stylo nous permet d'introduire les sections suivantes, qui portent sur l'étude du modèle textuel de cet environnement d'écriture utilisé lors de l'écriture de l'article publié dans la revue *Humanités numériques*.

IV.3 Modélisation des informations en texte brut

Cette section a pour objet l'étude des caractéristiques des formats de texte brut mobilisés dans Stylo. Notre approche s'articule principalement autour d'éléments

19. La pandoc-api est accessible à cet *endpoint* : `https://pandoc-api.stylo.huma-num.fr/`, site consulté le 25 septembre 2024.

20. FastAPI est disponible à cette adresse : `https://fastapi.tiangolo.com/`, consulté le 24 février 2024.

21. On peut trouver le module d'export à cette URL : `https://export.stylo.huma-num.fr/`, site consulté le 25 septembre 2024.

communs à ceux qui ont été analysés dans le chapitre précédent portant sur les logiciels de traitement de texte. À travers cette démarche, nous visons à mettre en lumière la polysémie de ces éléments selon les divers modèles textuels auxquels ils sont rattachés.

Lorsqu'il est question d'environnements d'écriture WYSIWYM, c'est-à-dire des environnements où le fond et la forme sont séparés en deux étapes distinctes de la production du document, la notion de texte brut est convoquée. Cette forme de texte, que l'on manipule et que l'on formate uniquement avec des caractères, est utilisée pour produire un document. Dans cette configuration, les éléments de structuration sont visibles et ne sont pas dissimulés sous un rendu graphique. En texte brut, les éléments structurels varient selon les formats employés. Ils présentent l'avantage d'être lisible à la fois par les humains et les machines. Ainsi, le texte brut consiste en une autre approche que celle du texte qui a été mis en page et structuré graphiquement dans un mode WYSIWYG.

Un fichier en texte brut, *plain text* en anglais, ou en « format texte » comme le nomme Perret dans sa thèse de doctorat, peut être largement défini comme « un fichier dont les données exprimées en binaire (0 et 1) se traduisent par des caractères textuels d'après une norme de codage – c'est-à-dire ici une convention qui établit quelles successions de bits correspondent à quels caractères » (Perret 2022 : 154).

Par exemple, le *Linux Information Project* (LINFO), a choisi de restreindre la définition du texte brut à une seule norme de codage, puisqu'il ne considère le texte brut que depuis certains caractères de la norme ASCII :

> Le *texte brut* désigne toute *chaîne de caractères* (c'est-à-dire une séquence finie de *caractères*) composée entièrement de *caractères imprimables* (c'est-à-dire de caractères lisibles par l'homme) et, éventuellement, de quelques types spécifiques de *caractères de contrôle* (par exemple, des caractères indiquant une tabulation ou le début d'une nouvelle ligne) [...]
>
> Le terme texte brut fait généralement référence à un texte composé entièrement de caractères imprimables ASCII et de quelques-uns de ses caractères de contrôle. ASCII, acronyme de *American standard code for information interchange*, est basé sur les caractères utilisés pour écrire la langue anglaise telle qu'elle est utilisée aux États-Unis. Il s'agit de la norme de facto pour le *codage des caractères* (c'est-à-dire la représentation des caractères par des nombres) utilisé par les ordinateurs et les équipements de communication pour représenter le texte, et il (ou une extension compatible de celui-ci) est utilisé sur la

> plupart des ordinateurs, y compris la quasi-totalité des ordinateurs personnels et des stations de travail.
>
> Le terme *caractères imprimables* désigne les 96 caractères ASCII (y compris le caractère espace, qui occupe un seul espace) qui sont effectivement lisibles par l'homme lorsqu'ils sont affichés sur un écran d'ordinateur ou imprimés sur du papier[22].

Si des communautés restreignent le texte brut à certaines normes de codages, comme le fait le LINFO, nous lui préférons la définition que donne Unicode dans ses spécifications : « Le texte brut est une séquence pure de code de caractères ; le texte brut en Unicode est donc une séquence de code de caractères Unicode (*The Unicode Standard : Core Specification* 2024 : 54) ».

Cette spécification Unicode nous apporte également une subtilité concernant le texte brut. Le texte brut ne désigne pas sans distinction toutes les séquences de code de caractères, car il doit être distingué des caractères qui structurent le contenu du contenu lui-même. En suivant cette distinction, le contenu est nommé texte brut, tandis que les caractères qui le structurent sont rangés sous l'appellation de texte enrichi, *rich text* an anglais (*The Unicode Standard : Core Specification* 2024 : 55).

La mention de ce texte enrichi apparaît également dans les RFC 2646[23] et 1523[24] relatives au texte brut contenu dans les courriels[25]. Dans ce cas également, le texte enrichi représente les balises (en ASCII) utilisée pour structurer certains éléments des courriels. Par exemple, le chevron « > » est utilisée dans les courriels pour marquer et structurer une citation.

Ainsi, nous employons les notions de *texte brut* et de *texte enrichi* pour désigner, à l'instar de la spécification Unicode, les contenus et leurs caractères structurant. Parmi les exemples de texte enrichi, on retrouve le SGML et toutes ses applications (HTML, XML, OOXML, ODF) ; mais aussi des langages de balisage léger comme Markdown, Asciidoc, reStructeredText ou encore un système de composition typographique comme LaTeX.

22. Voir la page web : `https://www.linfo.org/plain_text.html`, consultée le 28 février 2025.

23. Voir la page web : `https://www.rfc-editor.org/rfc/rfc2646#ref-RICH`, consultée le 28 février 2025.

24. Voir la page web : `https://www.rfc-editor.org/rfc/rfc1523`, consultée le 28 février 2025.

25. Historiquement, les courriels ont toujours été rédigés en texte brut, même si aujourd'hui la plupart des clients de messagerie utilisent par défaut le HTML comme format. Ceci permet notamment d'intégrer des éléments de mise en page dans les mails, ainsi que d'y attacher certains médias dans le corps du texte comme les images. Pour ce qui est du texte brut dans les courriels, en plus des spécifications du texte brut, on retrouve également une *Netiquette* dans laquelle sont expliquées les règles de bienséance lorsque l'on communique par courriel. Cette *Netiquette* a aussi fait l'objet d'une RFC en 1995, la RFC1855. Voir la page web : `https://www.rfc1855.net/#chapter-2-1-1` .

La distinction entre le texte brut et le texte enrichi tient du fait que le texte brut est un standard public lisible par tous les ordinateurs, puisqu'il est défini par sa norme de codage, tandis que le texte enrichi dépend non seulement du type d'encodage, mais également d'une implémentation spécifique à travers un format, qu'il soit libre ou propriétaire.

Comme nous l'avons déjà mentionné, il y a trois formats centraux de texte enrichi dans l'éditeur de texte Stylo : le Markdown pour le corps du texte, le YAML pour les métadonnées et le BibTEX pour les références bibliographiques. Chacun de ces formats a sa propre histoire et ses propres spécifications. Pour mieux appréhender la structuration des informations dans Stylo, nous commencerons par contextualiser ces formats et l'histoire à laquelle ils appartiennent, puis nous analyserons les mêmes éléments de production du sens que ceux étudiés dans le chapitre précédent afin d'effectuer notre comparaison. Par souci de concision, le format BibTEX ne sera pas abordé dans ce chapitre, mais sera traité dans le suivant.

Historique des formats utilisés dans Stylo

Mardown est un langage de balisage léger créé en 2004 par John Gruber[26] avec l'aide d'Aaron Swartz. Ses créateurs l'ont pensé comme une version plus légère du format HTML, d'où son nom *Markdown* pour signifier cette concision des balises, à l'inverse des formats se terminant par les lettres *ML* (*Markup language*) et souvent plus verbeux. Sa syntaxe légère permet de structurer et de décrire sémantiquement le texte. Il a été pensé pour pouvoir être converti facilement vers d'autres formats comme HTML, ou plus tard XML ainsi que d'autres formats de texte brut, comme en témoigne le convertisseur Pandoc et les possibilités de conversion qu'il offre.

Markdown se distingue des autres langages de balisages légers, car il est déclinable en différentes variantes (ou saveurs). Chacune d'entre elles comporte ses propres spécificités dans la syntaxe Markdown et la distingue des autres. Parmi les plus populaires, nous retrouvons :

— CommonMark[27]
— GitHub Flavored Markdown (GFM)[28]

26. Voir son site web : `https://daringfireball.net/projects/markdown/`, consulté le 31 mars 2024.
27. Les spécificités de CommonMark sont disponibles sur le site web dédié à cette saveur : `https://commonmark.org/`, consulté le 31 mars 2024.
28. Les spécificités de GFM sont disponibles sur cette page web : `https://github.github.com/gfm/`, consultée le 31 mars 2024.

— MultiMarkdown[29]
— Pandoc[30]
— Quarto[31]

Cette capacité à être déclinable et adaptable distingue fortement Markdown des autres langages de balisage. En effet, puisque chaque saveur contient des éléments personnalisés de structuration des contenus – des balises –, il est important de connaître la saveur que l'on doit utiliser dans un environnement au risque de se retrouver avec des balises qui ne sont pas interprétées et qui par extension, n'ont aucune signification pour cet environnement. C'est-à-dire que selon l'environnement d'écriture dans lequel on se trouve, et selon la saveur de Markdown qui est utilisée dans cet environnement, le texte enrichi risque de n'être que du texte brut sans aucune valeur hiérarchique ni sémantique puisque ces caractères ne sont pas interprétés par l'environnement d'écriture.

Extrait de code IV.1 : Baliser une vidéo en Quarto Markdown

```
1 {{< video https://www.youtube.com/embed/wo9vZccmqwc >}}
```

Par exemple, la saveur Quarto Markdown utilise la structure ci-dessus pour insérer une vidéo dans un texte. Cependant, ce marquage ne sera interprété que lorsque Quarto transformera le document Markdown en un autre document ; or dans Stylo cette ligne serait traitée comme un paragraphe et ne serait pas transformée parce que Stylo ne connaît pas cette structure puisque la saveur Quarto de Markdown n'y est pas prise en charge.

À ce propos, aucune saveur spécifique n'a été implémentée dans Stylo, dans le sens où Stylo n'utilise pas sa propre saveur de Markdown, et laisse aux utilisateurs le soin d'employer celle qui leur convient le mieux. Néanmoins, lorsque les sources sont transformées par le module d'export, les utilisateurs doivent respecter les préconisations du logiciel Pandoc puisque c'est ce dernier qui réalise les transformations et conversions des documents (l'export des fichiers sources n'est pas concerné). Les saveurs les plus couramment utilisées avec Pandoc sont CommonMark et GitHub Flavored Markdown[32].

Autrement dit, Stylo n'impose pas de variante de Markdown si l'on s'en sert comme éditeur de texte sans la nécessité d'utiliser le module d'export. Dès qu'une

29. Les spécificités de MultiMarkdown sont disponibles sur cette page web : `https://rawgit.com/fletcher/MultiMarkdown-6-Syntax-Guide/master/index.html`, consultée le 31 mars 2024.
30. Les spécificités de Pandoc sont disponibles sur cette page web : `https://pandoc.org/MANUAL.html#pandocs-markdown`, consultée le 31 mars 2024.
31. Les spécificités de Quarto sont disponibles sur cette page web : `https://quarto.org/docs/authoring/markdown-basics.html`, consultée le 31 mars 2024.
32. Outre celle qui porte son nom, Pandoc prend en charge d'autres variantes de Markdown comme cela est indiqué dans la documentation à ce sujet : `https://pandoc.org/MANUAL.html#markdown-variants`, page consultée le 25 septembre 2024.

chaîne éditoriale s'appuie sur ce module, comme c'est le cas pour la chaîne Stylo utilisant l'export XML conforme au schéma de COMMONS Publishing, il devient essentiel d'employer les saveurs de Markdown que traite Pandoc pour que les transformations et conversions se fassent sans erreurs.

Pour conclure sur la présentation du langage de balisage Markdown, sa possible déclinaison en diverses saveurs fait de ce langage un avantage et un inconvénient. C'est un avantage pour sa plasticité et son adaptabilité aux besoins d'une communauté ou d'un projet. Cependant, si les adaptations réalisées le sont dans une niche, soit parce que la communauté qui en définit les règles comporte trop peu de membres, soit parce qu'il n'y a qu'un seul environnement qui traite cette saveur, le Markdown perd sa caractéristique interopérable et contraint les usagers à bricoler des équivalences entre les transformations pour préserver la richesse sémantique des contenus.

Le deuxième format que nous étudions est le YAML qui est utilisé dans Stylo pour sérialiser des métadonnées. Dans sa version originale de 2001, l'acronyme du format avait pour signification *Yet Another Markup Language* puis se transforme à l'occasion de la publication de sa version 1.1 en *YAML Ain't Markup Language*[33]. YAML est un langage de sérialisation de données pour tous les langages de programmation. Un usage récurrent qui en est fait consiste à utiliser YAML pour créer des fichiers de configuration. Dans notre cas, YAML est utilisé pour enregistrer les métadonnées associées à un article Stylo. Le principe de YAML est très facile à assimiler puisqu'il repose sur le même fonctionnement qu'un dictionnaire et suit la structure `clef:valeur`. Tant que ce principe est respecté, chacun est en mesure de créer de toute pièce son document YAML et de choisir les `clefs` et les `valeurs` qui leur sont affectées. C'est ensuite l'environnement dans lequel ce format est utilisé qui va parser le contenu en suivant l'architecture des informations dans le fichier YAML.

Dans Stylo, un ensemble de `clefs` ont été choisies lors des développements de l'interface et les utilisateurs n'ont plus qu'à remplir un formulaire pour déclarer les `valeurs` qui seront associées aux différentes `clefs`[34].

33. Les spécifications des différentes versions sont disponibles en ligne à l'adresse `https://yaml.org/spec/`.

34. Le comportement par défaut de la saisie des métadonnées se fait dans un formulaire, soit dans une surcouche graphique qui évite aux utilisateurs d'apprendre la syntaxe YAML. Néanmoins, un mode en texte brut a été ajouté pour laisser aux utilisateurs la possibilité de saisir directement leurs métadonnées en texte brut, et de créer leurs propres métadonnées s'ils le souhaitent.

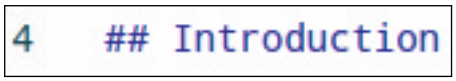


Figure IV.2 : Exemple d'un titre de niveau 2 en Markdown dans Stylo

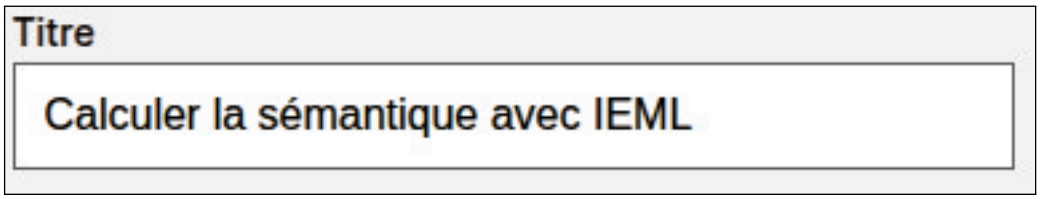


Figure IV.3 : Exemple d'un titre en YAML dans Stylo

Éléments « titre » et « paragraphe »

Dans le cadre de la comparaison entre le format Markdown utilisé dans Stylo et les formats étudiés dans le chapitre précédent, à savoir les formats ODT et DOCX, des éléments structurels équivalents doivent être observés. Afin d'effectuer cette comparaison, nous observons principalement dans Stylo le texte enrichi qui donne une modélisation du texte brut. Puisque le format Markdown a été pensé à partir du modèle textuel du format HTML, nous nous attendons à y retrouver peu ou prou un modèle similaire.

Les premiers éléments observés dans le chapitre III étaient le titre et le paragraphe. Une première distinction se manifeste aussitôt entre nos environnements WYSIWYM et WYSIWYG : en Markdown, il n'existe aucune ambiguïté entre ces deux éléments textuels. La structure étant explicitement indiquée dans le corps du texte, le titre et le paragraphe sont nettement séparés. Afin de rester cohérent avec la structure du chapitre précédent, nous les traiterons l'un après l'autre en commençant par le titre.

Pour indiquer un titre selon la syntaxe du format Markdown, il suffit d'ajouter le caractère # devant la chaîne de caractères concernée. Comme en HTML, les titres peuvent être répartis sur six niveaux : il suffit de répéter le caractère # autant de fois que le niveau de titre désiré.

Un titre de niveau 2 en Markkown dans l'article de la revue *Humanités numériques* ressemble donc à ce que l'on voit sur la figure IV.2.

Toutefois, Stylo comporte une particularité dans son modèle textuel. Si le Markdown permet de saisir des titres via sa syntaxe, l'environnement Stylo permet également de saisir un titre dans les métadonnées en YAML, comme on peut le voir dans la figure IV.3.

L'environnement Stylo distingue ainsi deux types de titre. Le premier type se trouve dans le corps de texte et le second dans les métadonnées. Nous remarquons que le titre en Markdown est un titre de niveau 2 et qu'il n'y a pas d'élément qui

lui serait supérieur dans le corps de texte, c'est-à-dire qu'il n'y a pas de titre de niveau 1 saisi en Markdown. Ainsi, le corps de l'article n'est structuré qu'avec des éléments de niveau 2 ou inférieur. Cette écriture peut sembler illogique dans la mesure où les bonnes pratiques de structuration d'un texte recommandent une hiérarchie stricte à l'intérieur de laquelle les niveaux de 1 à 6 doivent être respectés sans qu'il n'y ait d'omission.

Une autre pratique consiste à ne saisir qu'un seul titre de niveau 1 dans un document en Markdown. On considère qu'il s'agit là du titre du document [35]. Le reste du contenu est alors structuré à partir du niveau 2. Stylo propose aux utilisateurs de respecter cette dernière pratique en déplaçant le titre de niveau 1 à l'intérieur du formulaire de métadonnées YAML, plutôt que de le garder dans le corps de texte, comme on le voit dans la figure IV.3. En suivant cette forme d'écriture, la structuration du corps de texte en Markdown peut débuter au niveau 2 [36], puisque le niveau 1 est représenté par la métadonnée en YAML. Il y a là un glissement sémantique assez fort, car le titre du document n'est pas modélisé par un titre de niveau 1 en Markdown, mais par une entrée dans le formulaire de métadonnées. Dans ce cas de figure, le titre du document devient une métadonnée et n'est plus considéré comme un élément du corps de texte. Cette modélisation particulière des titres dans Stylo lie le Markdown au YAML, car si ce dernier venait à manquer, le contenu n'aurait plus de titre de niveau 1 (ou équivalent).

Le deuxième élément que nous étudions est le paragraphe. En Markdown, la structure d'un paragraphe n'est pas délimitée par une balise, contrairement à leur définition dans les normes OOXML, ODF ou même HTML. Il l'est seulement par du texte brut rédigé sur une ou plusieurs lignes et sans aucune balise en début de ligne. Les deux lignes autour d'un paragraphe, la précédente et la suivante, sont laissées vides pour séparer les différents paragraphes d'un texte ou séparer un paragraphe d'un autre élément dans la structure (un titre, une liste, une citation, etc.).

Dans la figure IV.4 , aucun caractère n'est employé pour signifier que le paragraphe en est un : c'est la ligne vide qui est le marqueur de cet élément structurel. Si dans les formats DOCX et ODT le paragraphe est associé à la ligne, puisque chaque nouvelle ligne déclenchée par un retour chariot correspond à un nouveau paragraphe, il l'est également dans Stylo avec le Markdown. Néanmoins, une différence majeure subsiste puisque l'espace et le retour chariot seul ne sont pas interprétés en Markdown. Pour marquer la fin d'un paragraphe dans ce

35. Plusieurs tutoriels d'utilisation du Markdown recommandent cet usage des niveaux de titre, voir par exemple le tutoriel suivant : `https://www.markdownlang.com/fr/basic/headings.html`, consulté 09 août 2025.

36. Il n'y a aucune prescription, le document peut commencer par un titre de niveau 1, il ne sera juste pas considéré comme le titre du document lors des différentes transformations dans Stylo.

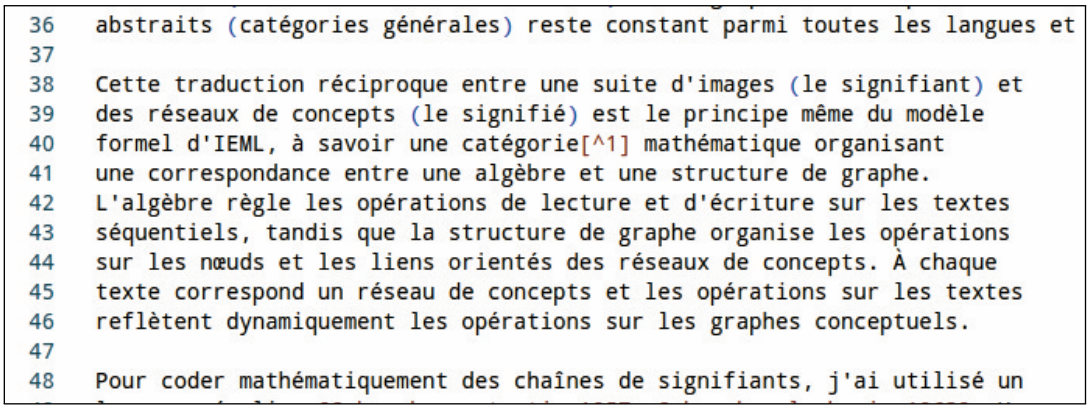


Figure IV.4 : Un paragraphe de l'article saisi dans Stylo

format, il faut nécessairement une ligne vide[37]. En suivant ce fonctionnement, si à la fin d'une phrase, ou lorsqu'une ligne atteint entre 80 et 100 caractères, on trouve un simple retour chariot à la suite du dernier caractère de la ligne, puis du texte brut à la ligne suivante, sans constater de ligne vide, le texte est alors considéré comme appartenant toujours au même paragraphe. Il n'y a pas de nouveau paragraphe créé lors du retour chariot, comme on peut le voir dans l'extrait de code IV.2.

Extrait de code IV.2 : Le paragraphe Markdown dans Stylo transformé au format HTML

```
<p>Cette traduction réciproque entre une suite d'images
(le signifiant) et des réseaux de concepts (le signifié)
est le principe même du modèle formel d'IEML, à savoir
une catégorie<a href="#fn2" class="footnote-ref"
id="fnref2" role="doc-noteref"><sup>2</sup></a>
mathématique organisant une correspondance entre une
algèbre et une structure de graphe. L'algèbre règle les
opérations de lecture et ’décriture sur les textes
séquentiels, tandis que la structure de graphe organise
les opérations sur les œnuds et les liens orientés des
réseaux de concepts. À chaque texte correspond un réseau
de concepts et les opérations sur les textes reflètent
dynamiquement les opérations sur les graphes
conceptuels.</p>
```

En Markdown (voir la figure IV.4), le paragraphe se présente comme un objet visuel connu, en ce sens que sa structure est composée d'un bloc de phrases ou de texte brut, caractérisé par des espaces vides tout autour, et ce, indépendamment du nombre de lignes qui le composent. Cette approche rappelle la conception

37. Deux espaces l'un derrière l'autre ou un espace auquel on ajoute un retour chariot sont des combinaisons qui fonctionnent pour obtenir le même résultat, toutefois ce ne sont pas des bonnes pratiques, car elles rendent la lecture plus difficile et risquent de provoquer des erreurs d'interprétation.

traditionnelle des paragraphes dans les documents imprimés, où ils sont également délimités par des espaces vides, conférant ainsi une organisation visuelle similaire à celle que l'on retrouve dans ces documents imprimés alors que ce n'était pas le cas dans MSWord et LibreOffice. Si le formalisme prévaut dans ce mode d'écriture, il y a pourtant une part graphique qui persiste et se rend visible à l'écran dans la forme que prend le paragraphe en Markdown.

Éléments « commentaire » et « annotation »

Si le paragraphe est un des éléments basiques de la syntaxe Markdown, le commentaire est un objet plus complexe. Dans Stylo, le commentaire[38] prend la forme d'une annotation que l'on saisit grâce à l'application Hypothesis[39] intégré à l'environnement d'écriture.

Hypothesis est une application web permettant d'annoter les contenus des sites web. En créant un compte utilisateur sur Hypothesis, il devient possible d'annoter les contenus et de retrouver toutes les annotations sur son compte personnel, car les annotations sont physiquement stockées sur les serveurs d'Hypothesis. Cette fonctionnalité, étant liée à des contenus web, a été implémentée dans la prévisualisation HTML des articles rédigés dans Stylo. La prévisualisation permet de transformer le contenu saisi dans l'éditeur de texte en HTML, de le mettre en forme avec une feuille de styles au format CSS, puis de le servir à une URL. Les annotations ainsi saisies dans la prévisualisation d'un article sont liées à un document Stylo grâce à cette URL.



Figure IV.5 : Une annotation dans la prévisualisation HTML de l'article

38. Il est également possible de saisir un commentaire directement en Markdown. Néanmoins, il ne s'agit pas de l'équivalent du commentaire dans MSWord et LibreOffice. En conséquence, nous ne le traitons pas. La syntaxe d'un commentaire est identique à celle du langage HTML : `<!--Ceci est un commentaire HTML et Markdown.-->`. Si, lors de la lecture, nous comprenons qu'il s'agit d'un commentaire dans le texte, il en est différemment du point de vue formel et computationnel. Le commentaire HTML hérite du commentaire que l'on trouve dans la plupart des langages de programmation : c'est une chaîne de caractère qui n'est pas interprétée. Autrement dit, lorsqu'une ligne est commentée, elle n'a pas la signification d'un commentaire dans le sens d'une observation ou d'une remarque, mais d'une chaîne de caractère qui n'est pas interprétée, comme cela est stipulé dans la RFC 1866. La RFC 1866 spécifie bien que la totalité du commentaire est ignorée : « The entire comment declaration is ignored », voir la page web `https://www.rfc-editor.org/rfc/rfc1866` à la section 3.2.5, page consultée le 03 avril 2025.

39. Voir la page web du projet : `https://web.hypothes.is/`, consultée le 08 août 2025.

En étant ainsi liée à une URL et à Hypothesis, l'annotation n'est pas un objet modélisé dans le document de Stylo, puisqu'elle lui est externe, mais est plutôt modélisé dans l'environnement même de Stylo. L'annotation a en un sens un pouvoir très fort sur le document. C'est à travers elle, comme on le voit dans la figure IV.5 , que l'on peut proposer des révisions, des commentaires, des remarques ou des observations qui, *in fine*, auront un impact sur le document.

Néanmoins, ce choix de technologie n'est pas optimal et pose plusieurs problèmes. Il pose tout d'abord des problèmes en termes d'expérience utilisateur, car Hypothesis est intégré dans un onglet du navigateur dont le contenu est statique, c'est-à-dire qu'il ne peut pas être modifié. Si de prime abord cette solution semble tout à fait adaptée pour effectuer de l'*open peer review*, tel que c'est par exemple l'usage qu'en fait la revue *Sens public*, ce type de commentaire n'est toutefois pas adapté à un travail éditorial, car il sous-entend un nombre conséquent de va-et-vient entre l'onglet d'édition et l'onglet de consultation / saisie des annotations dans Hypothesis.

De plus, la modélisation textuelle de ces annotations reste très superficielle, car le modèle ne permet pas de qualifier le type d'annotation saisie. Pour ce faire, la seule façon de typer le commentaire est de lui adjoindre une étiquette dont la valeur est librement affectée (c'est une chaîne de caractères), ce qui nécessite d'être très rigoureux dans leur saisie, surtout lorsque plusieurs acteurs interviennent sur le texte.

Malgré le fait qu'il y ait des similitudes entre le commentaire présent dans les traitements de texte et l'annotation dans Hypothesis, l'annotation dans Stylo rompt également avec la tradition éditoriale de relecture et d'annotation des contenus.

Des éditeurs, ou des utilisateurs, nous ont en ce sens fait remarquer à plusieurs reprises qu'il n'était pas possible d'annoter du contenu directement dans l'éditeur de texte de Stylo. L'ajout de cette fonctionnalité n'est pas quelque chose de trivial, en effet, il ne s'agit pas d'ajouter un espace de saisie libre que l'on nommerait commentaire ou annotation à l'éditeur de texte. Cet enjeu soulève la problématique de modéliser l'annotation, telle que les éditeurs la définissent, au sein d'un modèle textuel numérique, chose qui à notre connaissance n'a pas été réalisé.

Actuellement, le modèle textuel d'Hypothesis (dans Stylo), ne fait donc pas la distinction entre un commentaire ou une annotation saisie par un pair, ou saisie par un éditeur. Il serait donc à l'avantage de Stylo de pouvoir distinguer et préciser formellement cette typologie dans son modèle textuel afin d'enrichir sémantiquement cette dimension du paratexte d'un article.

Pour conclure sur l'annotation dans Stylo avec Hypothesis, celle-ci est alors modélisée d'une manière différente des modélisations dans les formats DOCX

et ODT, puisque dans ces deux environnements, le commentaire et l'annotation sont des objets directement intégrés dans le modèle textuel. L'intégration de l'annotation à l'intérieur des modèles textuels et des documents en DOCX et ODT la rend portable, alors que celle de Stylo est liée à son environnement : lorsque le document est exporté en dehors de cet environnement, il perd les annotations qui lui sont associées.

Le dernier élément que nous avons observé dans le chapitre précédent était le résumé, et plus largement l'ensemble des métadonnées d'un document.

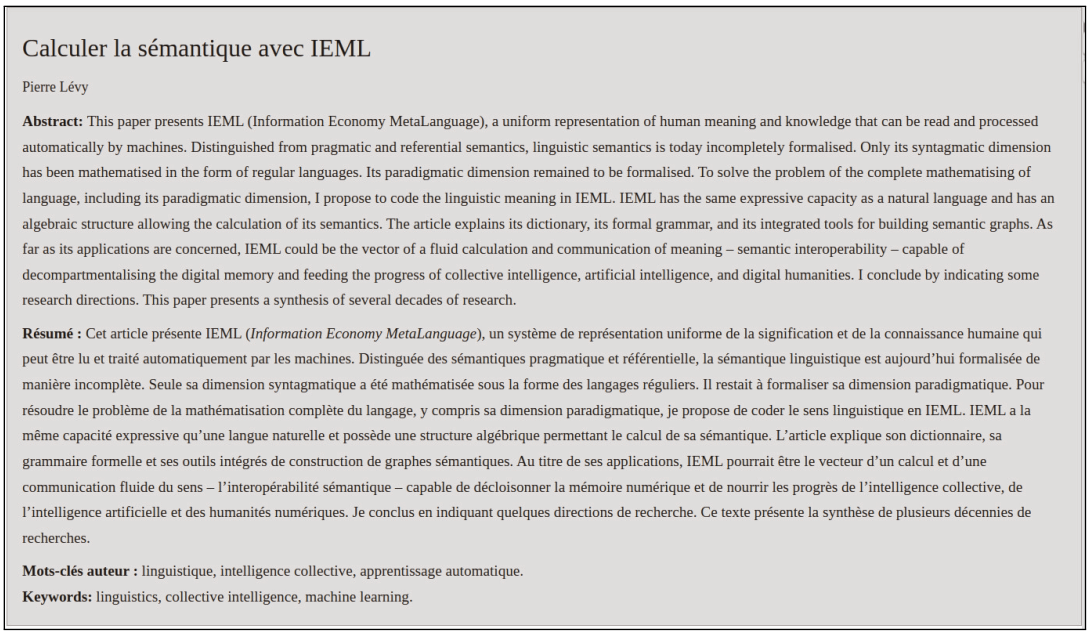
Élément « résumé »

Si nous reprenons l'exemple du résumé, comme ce fut le cas dans le chapitre précédent, celui de l'article d'*Humanités numériques* est déclaré comme suit dans Stylo :

Extrait de code IV.3 : Extrait des métadonnées en YAML dans Stylo

```
abstract: >-
Cet article présente IEML (*Information Economy
MetaLanguage*), un système de représentation uniforme de la
signification et de la connaissance humaine qui peut être
lu et traité automatiquement par les machines. Distinguée
des sémantiques pragmatique et référentielle, la sémantique
linguistique est aujourd'hui formalisée de manière
incomplète. Seule sa dimension syntagmatique a été
mathématisée sous la forme des langages réguliers. Il
restait à formaliser sadimension paradigmatique. Pour
résoudre le problème de la mathématisation complète du
langage, y compris sa dimension paradigmatique, je propose
de coder le sens linguistique en IEML. IEML a la même
capacité expressive qu'une langue naturelle et possède une
structure algébrique permettant le calcul de sa sémantique.
L'article explique son dictionnaire, sa grammaire formelle
et ses outils intégrés de construction de graphes
sémantiques. Au titre de ses applications, IEML pourrait
être le vecteur d'un calcul et d'une communication fluide
du sens -- l'interopérabilité sémantique -- capable de
décloisonner la mémoire numérique et de nourrir les progrès
de l'intelligence collective, de l'intelligence
artificielle et des humanités numériques. Je conclus en
indiquant quelques directions de recherche. Ce texte
présente la synthèse de plusieurs décennies de recherches.
```

Le résumé est un élément déclaré dans un formulaire[40] juxtaposé à l'éditeur en Markdown. C'est un élément que l'on trouve dans un fichier séparé et qui est formellement décrit par l'affectation d'une chaîne de caractères à la `clef` YAML *abstract*. Le formalisme dont il est question avec ce format est relativement souple, car le choix de la `clef` ne repose pas sur un standard établi, mais sur un choix éditorial réalisé lors des développements de Stylo. En conséquence, cette `clef` n'a de valeur sémantique que pour cet environnement. De ce fait, c'est un élément qui devient calculable et transformable par Stylo.



Calculer la sémantique avec IEML

Pierre Lévy

Abstract: This paper presents IEML (Information Economy MetaLanguage), a uniform representation of human meaning and knowledge that can be read and processed automatically by machines. Distinguished from pragmatic and referential semantics, linguistic semantics is today incompletely formalised. Only its syntagmatic dimension has been mathematised in the form of regular languages. Its paradigmatic dimension remained to be formalised. To solve the problem of the complete mathematising of language, including its paradigmatic dimension, I propose to code the linguistic meaning in IEML. IEML has the same expressive capacity as a natural language and has an algebraic structure allowing the calculation of its semantics. The article explains its dictionary, its formal grammar, and its integrated tools for building semantic graphs. As far as its applications are concerned, IEML could be the vector of a fluid calculation and communication of meaning – semantic interoperability – capable of decompartmentalising the digital memory and feeding the progress of collective intelligence, artificial intelligence, and digital humanities. I conclude by indicating some research directions. This paper presents a synthesis of several decades of research.

Résumé : Cet article présente IEML (*Information Economy MetaLanguage*), un système de représentation uniforme de la signification et de la connaissance humaine qui peut être lu et traité automatiquement par les machines. Distinguée des sémantiques pragmatique et référentielle, la sémantique linguistique est aujourd'hui formalisée de manière incomplète. Seule sa dimension syntagmatique a été mathématisée sous la forme des langages réguliers. Il restait à formaliser sa dimension paradigmatique. Pour résoudre le problème de la mathématisation complète du langage, y compris sa dimension paradigmatique, je propose de coder le sens linguistique en IEML. IEML a la même capacité expressive qu'une langue naturelle et possède une structure algébrique permettant le calcul de sa sémantique. L'article explique son dictionnaire, sa grammaire formelle et ses outils intégrés de construction de graphes sémantiques. Au titre de ses applications, IEML pourrait être le vecteur d'un calcul et d'une communication fluide du sens – l'interopérabilité sémantique – capable de décloisonner la mémoire numérique et de nourrir les progrès de l'intelligence collective, de l'intelligence artificielle et des humanités numériques. Je conclus en indiquant quelques directions de recherche. Ce texte présente la synthèse de plusieurs décennies de recherches.

Mots-clés auteur : linguistique, intelligence collective, apprentissage automatique.
Keywords: linguistics, collective intelligence, machine learning.

FIGURE IV.6 : Résumé de l'article dans la prévisualisation HTML

Comme on peut le voir dans la prévisualisation HTML dont un fragment est montré dans la figure IV.6, le résumé déclaré dans le YAML est transformé en différents éléments HTML visibles et invisibles de cette page. Pour ce qui relève du visible, le résumé est disposé dans l'encart en en-tête de la page web affichée à l'écran, dans une division HTML qui a pour propriété `description` et pour classe

40. La plupart des métadonnées dans Stylo sont donc saisies dans ce formulaire présent dans l'interface d'écriture. Cependant, ce n'est pas le seul endroit où des métadonnées en YAML peuvent être renseignées. Une autre pratique d'écriture en Markdown, que l'on rencontre avec Pandoc, mais également avec des générateurs de site statique, consiste à intégrer en en-tête de document des métadonnées sérialisées en YAML dans ce que l'on appelle un *frontmatter*. Cette pratique semble étrange de prime abord, car c'est un fichier en Markdown qui contient d'abord des informations au format YAML puis des informations structurées en Markdown, ce qui n'est pas immédiatement intuitif du fait des mélanges entre les syntaxes. Pourtant, c'est une pratique répandue car elle permet à chaque utilisateur de renseigner ses métadonnées selon une architecture personnelle. La saisie des métadonnées dans Stylo peut donc se faire de deux manières différentes, soit via le formulaire, soit via le *frontmatter*. Dans un cas comme dans l'autre, les métadonnées sont associées à une `clef` qui structure explicitement la métadonnée concernée.

`resume`. Le résumé est également présent dans l'en-tête `<head>` de la page HTML, là où les éléments décrits ne sont pas directement visibles sur la page. On le retrouve notamment dans plusieurs métadonnées de cette page web, qualifiée par des balises `<meta>` et dont les propriétés sont décrites grâce à des vocabulaires standards comme le *Dublin Core*. Ces informations sont principalement adressées à des robots ou à des parseurs.

Tandis que la déclaration des métadonnées ne reposait que sur le paradigme graphique dans les environnements de traitement de texte, une déclaration plus formelle dans Stylo nous fait rentrer dans un autre paradigme, celui du calcul, puisque cette forme d'écriture d'un résumé composé d'une chaîne de caractères en texte brut que l'on affecte à une `clef` YAML permet de rendre le résumé calculable et de l'enrichir en l'adossant à des vocabulaires contrôlés affiliés au web sémantique.

Au-delà de Stylo, l'utilisation de YAML est toutefois controversée. Contrairement à d'autres langages de structuration de données dont le comportement est pérenne, comme le standard JSON utilisé pour la première fois en 2001 (ECMA-404 : 2017 : iii), YAML 1.0 subit des modifications régulières depuis 2004 avec une version 1.1 en 2005 puis une version 1.2 en 2009 et une dernière mise à jour en 2021 avec la version 1.2.2. Alors que la stabilité offerte par des formats tels que JSON garantit une certaine pérennité aux applications, malgré une modification mineure en 2005[41], YAML a fait le choix d'évoluer et de s'adapter aux besoins des communautés.

Cependant, comme le mentionne Ruud van Asseldonk sur son blog[42], ces mises à jour peuvent générer des complications lorsque les fichiers YAML doivent passer d'un environnement à un autre alors que les versions de YAML utilisées sont différentes. Par exemple, Pandoc intègre en juillet 2018 la version 1.2 de YAML[43] où nous pouvons y lire :

« Mise à jour du manuel pour les valeurs YAML "vraies". Maintenant que nous utilisons HsYAML et YAML 1.2, les valeurs true valides sont true, True, TRUE. NOTE! y, yes, on, ne sont plus des valeurs vraies. »

Le changement de version génère une modification de comportement des valeurs `y, yes, on!`, qui signifiaient le booléen `true` dans la version 1.1 et ne sont plus que des chaînes de caractères à partir de la version 1.2. Or, tous les parseurs de YAML n'ont pas fait cette mise à jour. Par exemple, la très répandue librairie

41. En 2005, Douglas Crockford décide de supprimer la saisie des commentaires à l'intérieur des documents JSON. Voir l'histoire de JSON `https://www.youtube.com/watch?v=-C-JoyNuQJs&t=965s`, consultée le 24 septembre 2024

42. Voir le blog de Ruud van Asseldonk : `https://ruudvanasseldonk.com/2023/01/11/the-yaml-document-from-hell`, page consultée le 31 mars 2024.

43. Voir la page des *releases* de Pandoc : `https://pandoc.org/releases.html#pandoc-2.2.2-2018-07-16`, consultée le 31 mars 2024.

Python PyYaml, dont la dernière mise à jour remonte à juillet 2023 [44], s'appuie toujours sur la version 1.1 de YAML. En somme, si un document doit passer d'un environnement utilisant la version 1.1 ou la version 1.2, les informations structurées ne seront pas traitées de la même manière.

Nous sommes en droit de nous demander pourquoi YAML reste aussi populaire malgré ces inconvénients ? Ruud van Asseldonk apporte plusieurs réponses à cette question. La première est que YAML fait partie des plus anciens langages de sérialisation de données et répondait alors à un besoin de toute une génération de développeurs, ensuite, il permet l'écriture de commentaires à l'intérieur des documents, tout comme les commentaires en Markdown, c'est-à-dire du texte qui sera ignoré par le parseur, alors que JSON ne le permet pas [45]. Des alternatives comme le langage *Tom's Obvious, Minimal Language* (TOML) [46] ont vu le jour dans les années 2010 (2013 pour le TOML) pour tenter de pallier les problèmes susmentionnés. Le langage TOML est par exemple utilisé pour le fichier de configuration du paquet Python « Pressoir » afin de déclarer différents paramètres, par exemple de mise en page, parsés par le Pressoir et utilisés pour générer des livres au format HTML.

Limites des formats d'écriture

Étant strictement définis par des règles, les formats dépassent une simple manière de saisir une donnée. À travers ces formats et les modes de lectures que l'on peut y adosser, les informations saisies se voient dotées de comportements et peuvent modifier le sens produit, comme le montre les différences de signification entre les informations saisies dans les environnements de traitement de texte MSWord et LibreOffice et celles saisies dans Stylo en YAML et en Markdown.

Le choix des formats dans lesquels les utilisateurs peuvent saisir leurs textes et leurs données n'est pas anodin. Qu'il soit ancien, récent, verbeux ou léger, permissif ou rigide, le format d'écriture conditionne en grande partie ce que l'on a le droit d'écrire ou non. En ce sens la décision de ce qui peut être saisi est déjà prise avant qu'un texte soit frappé sur le clavier. Tout comme c'était le cas avec DOCX et ODT dans MSWord et LibreOffice, le Markdown dans Stylo ne permet pas à un philologue de saisir explicitement un appareil critique. C'est une

44. Voir la page web de la librairie : `https://pypi.org/project/PyYAML/`, consultée le 31 mars 2024.

45. Dans la vidéo précédente de l'histoire de JSON, Douglas Crockford fait une erreur lorsqu'il compare YAML à JSON pour la saisie des commentaires. L'utilisation de commentaires en YAML existe depuis la version de travail des recommandations du 16 juin 2001 avec l'intégration du caractère # pour signaler un commentaire signalé dans les changements apportés le 26 mai 2001, voir la page `https://yaml.org/spec/history/2001-06-16.html` consultée le 24 septembre 2024.

46. Voir le site web du langage TOML : `https://toml.io/en/`, consulté le 31 mars 2024.

syntaxe qui n'existe pas. Dans ce cas-ci, puisque l'appareil critique n'existe pas en Markdown, il ne peut pas exister dans Stylo.

Ainsi, lorsque nous écrivons dans un environnement numérique – qu'il s'agisse de Stylo, MSWord, LibreOffice, etc. – il existe un risque que celui-ci ne réponde pas pleinement aux besoins ou à l'intention de l'auteur. Une friction peut alors survenir entre les formats imposés par l'environnement et la formalisation des éléments textuels, en raison d'un modèle textuel incomplet ne définissant pas les éléments sous-jacents à ces besoins.

IV.4 La circulation des informations dans Stylo

Nous l'avons déjà énoncé dans la section IV.2, le modèle textuel de Stylo ne peut pas être réduit uniquement aux formats en texte brut. C'est un environnement d'écriture web, et en ce sens, le texte brut doit être manipulé dans cet espace pour produire un document. Les informations doivent circuler d'un module à un autre, entre l'application visible à son URL et la base de données stockée sur les serveurs de la TGIR Huma-num. Pour ce faire, elle nécessite un ensemble d'étapes et de micro-transformations qui adaptent le texte brut aux protocoles de communication entre client et serveur. De plus, d'autres applications web et externes à Stylo sont intégrées à cet espace et peuvent parfois intervenir dans le modèle textuel. Nous en avons déjà eu un aperçu avec les annotations dans l'application Hypothesis. À travers cette dernière section du chapitre, nous allons donc étudier les différentes étapes de matérialisation du document à l'intérieur de Stylo à partir de la saisie du texte brut en Markdown, YAML et BibTeX.

Le Web, en tant que couche intermédiaire supplémentaire entre l'utilisateur et la machine par rapport aux dispositifs d'écriture précédents, implique un ensemble de médiations qui lui sont propres. L'agencement de ces médiations dans cet environnement qu'est Stylo est alors la condition de matérialisation du document, puisque, si l'une d'entre elles venait à disparaître, la matérialisation échouerait et le document ne pourrait être produit. Ainsi, le Web, et plus précisément l'ensemble des médiations qui relèvent de ce dispositif dans Stylo, est une couche technique active de la production du sens dans Stylo. Les manipulations des éléments textuels effectuées par l'API GraphQL ou encore de la méthode `POST` du protocole HTTP, ainsi que par le composant de l'éditeur Monaco et la manipulation du DOM, montrent que la matérialisation du document dans Stylo s'effectue d'une tout autre manière que dans MSWord et LibreOffice.

Communication client-serveur

La figure IV.1 nous montrait un schéma des différents modules composant Stylo. À l'intérieur de ce dernier, nous avons vu qu'il y avait trois modules principaux dans Stylo : la base de données, l'interface web et le module d'export. Nous laissons de côté le module d'export, que nous abordons dans le chapitre suivant, dans la section V.3.

Les deux blocs restants, la base de données et l'interface web, ne sont pas en communication directe, puisque les champs de saisie du texte dans la partie HTML ne permettent pas d'écrire directement dans la base de données. En conséquence, un canal de communication devait être établi entre ces deux objets pour que les données puissent être accessibles à la fois en lecture, pour l'affichage dans la page web, et en écriture pour ajouter, modifier ou supprimer des éléments. Pour mettre en œuvre cette communication, une API utilisant le langage de requête GraphQL a été mise en place et rendue accessible via le protocole HTTP [47], la surcouche du protocole internet utilisée pour le web. Le langage de requête et de manipulation des données GraphQL a été développé par Facebook à partir de 2012 puis publié en *open source* en 2015.

L'une des particularités d'une API GraphQL, contrairement à une API *Representational State Transfer* (REST) par exemple, est qu'elle sert l'ensemble des données à une seule adresse (*endpoint*) alors que plus généralement, les données sont accessibles à des URL très précises, ce qui a pour effet de rendre explicite la structuration des données dans la base. En ne servant les données qu'à une seule adresse, l'API s'échappe de la contrainte de la structuration des données et contourne les problèmes récurrents d'*over-fetching* ou d'*under-fetching* que l'on peut rencontrer dans certaines applications [48]. L'API GraphQL est donc agnostique à l'égard de la forme de la base de données, MongoDB dans ce cas-ci. Par contre, la définition de requêtes adressables à la base de données doit être déclarée pour que l'on puisse faire circuler les informations entre le serveur et le client. Pour effectuer cela, GraphQL à son propre langage de description de schéma (*Schema Definition Language* (SDL)). Ce schéma représente la structure des informations et la façon de les appeler via des requêtes.

Ce qui est intéressant avec ce type de structuration des informations est qu'il est sous forme de graphe d'objets, c'est-à-dire qu'un objet comme un article est défini par ces liens à d'autres d'objets, que ce soit les éléments du contenu de

47. L'*endpoint* de l'API GraphQL de Stylo est accessible ici : `https://stylo.huma-num.fr/graphql`, page consultée le 25 septembre 2025.

48. Ces deux problèmes désignent soit une récupération trop importante de données et nécessite un tri après récupération des données sur le serveur, soit un manque de données pallié par un deuxième appel ou plus au serveur pour compléter le besoin.

l'article ou alors des `corpus`, des `workspaces`[49], d'autres utilisateurs de Stylo, etc. Chacun de ces objets auquel un article est lié est lui-même lié à d'autres objets. Un `workspace` est par exemple lié à tout un ensemble d'articles et d'utilisateurs. Nous revenons sur cette notion un peu plus tard. Le point essentiel à retenir est que ces liens qui connectent les objets définissent les relations entre eux dans la base de données et forment ainsi un graphe. Cette modélisation de la base de données est décrite dans un schéma accessible sur la plateforme Github où se trouve le code source de Stylo[50].

Par exemple dans Stylo, le champ dans la base de données qui modélise ce qu'est un utilisateur (`user`) dans cet environnement contient notamment les informations suivantes :

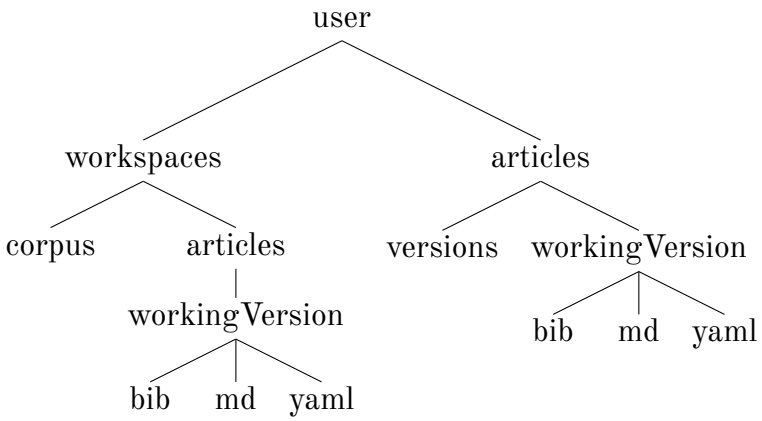


Figure IV.7 : Échantillon de l'API GraphQL de Stylo

Cette modélisation met en évidence le fait que l'objet `articles` est à la fois lié à l'utilisateur, mais également aux espaces de travail (`workspaces`). Toutefois, selon si on accède aux articles contenus dans un espace de travail ou ceux liés à un utilisateur, la requête n'obtiendra pas la même réponse.

La modélisation IV.7 est une réduction du graphe de l'API et ne représente qu'un tout petit fragment de toutes les informations que l'on y retrouve. Par exemple, l'entrée `articles` de l'API GraphQL de Stylo est liée à deux autres objets dans l'échantillon présenté dans la figure IV.7, tandis qu'elle est définie par une trentaine d'objets (ainsi que tous leurs sous-objets) dans le modèle de données, dont certaines ne contiennent que des informations à afficher, comme pour le titre, tandis que d'autres correspondent à des actions à effectuer, comme `startCollaborativeSession`.

49. Les `corpus` et les `workspaces` sont des objets présents dans le graphe de l'API GraphQL de Stylo et correspondent également à deux fonctionnalités sur lesquelles nous revenons plus loin dans cette section.

50. La modélisation du schéma GraphQL de Stylo est accessible sur le dépôt Github à l'adresse suivante :`https://github.com/EcrituresNumeriques/stylo/blob/master/graphql/models/user.js`, page consultée le 25 septembre 2024.

Une requête où l'on ne récupère que quelques informations d'un `article` pourrait ressembler à ceci :

Extrait de code IV.4 : Requête GraphQL getMyArticle()

```
query getMyArticles ($articleId: ID!) {
  article(article: $articleId) {
    title
    owner {
      firstName
      lastName
    }
    createdAt
    versions {
      createdAt
    }
  }
}
```

La requête IV.4 renverrait comme réponse :

Extrait de code IV.5 : Réponse de la requête GraphQL getMyArticle()

```
{
  "data": {
    "article": {
      "title": "Calculer la sémantique avec IEML",
      "owner": {
        "firstName": "Pierre",
        "lastName": "Levy"
      },
      "createdAt": "2022-12-18T21:28:36.802Z",
    }
  }
}
```

Cette requête IV.4 cible l'article ayant pour identifiant la valeur affectée à la variable *articleId*, et récupère le titre de l'article, le prénom et le nom de son propriétaire, ainsi la date de création.

Cet exemple montre qu'il y a une certaine économie de l'information implémentée dans le fonctionnement même de GraphQL pour n'aller chercher que les informations nécessaires pour une requête particulière et non pas l'ensemble des informations que l'entrée *article* contient, pour peu que la requête en elle-même soit bien rédigée. D'ailleurs, il s'agit là d'un des écueils potentiels de GraphQL : des requêtes mal formulées peuvent aller à l'encontre de ce principe.

Dans Stylo, chaque fonctionnalité, chaque bouton (ou presque) qui réalise une action de lecture ou d'écriture est lié à une requête GraphQL et au schéma de données correspondant. Chacune de ces actions suit en conséquence une modalité d'inscription dans la base de données se conformant à l'architecture des informations implémentée lors des développements de Stylo. Ce faisant,

chacun de ces boutons est une amorce à l'organisation des informations et à la matérialisation du document, puisqu'ils organisent la manière dont sont traitées les informations. Par exemple, il est toujours possible d'utiliser le champ dédié au corps de texte pour indexer des métadonnées, seulement Stylo n'est pas prévu pour traiter en tant que métadonnées les informations contenues dans l'espace réservé au texte. Elles seront alors traitées comme du texte et perdront leur valeur sémantique.

Structurer les données grâce à cette technologie ne se résume pas à générer un schéma pour construire le graphe de données. Il faut également penser et définir les objets qui y sont contenus. En réduisant l'article aux informations mentionnées dans la modélisation IV.7, nous voyons déjà apparaître des éléments tels que les espaces de travail, les versions, la version de travail, le contenu lui-même en Markdown, YAML et BibTEX. Chacun des éléments définis dans la structure des données est un élément producteur de sens de l'environnement d'écriture Stylo.

Par exemple, l'espace de travail est un concept implémenté en mai 2023 dans Stylo [51]. Cette notion a été créée en réponse à un besoin des utilisateurs. Le besoin émit par la communauté de Stylo était celui de partager les informations avec d'autres utilisateurs, afin de travailler collaborativement sur les documents. Jusqu'à ce que ces espaces de travail soient mis en ligne, le partage des informations était réalisé au moyen de deux fonctionnalités de partage : le partage d'un article entre utilisateur et le partage de compte, ce dernier donnant accès à tous les documents d'un utilisateur à un autre utilisateur. Ce fonctionnement couplé à l'augmentation du nombre de partages et du nombre d'articles sur chaque compte est devenu inopérant, ce qui a entraîné une somme de problèmes sur la plateforme, notamment une importante consommation des ressources du serveur [52]. Cet exemple montre que l'écriture, selon les modalités d'écriture et la modélisation du texte, n'est pas une action différenciée de l'infrastructure : il s'agit d'un même dispositif au sein duquel la matérialité de l'écriture peut également s'exprimer de cette manière.

L'espace de travail est donc une réponse à ces problématiques. Il a été pensé comme une entité de Stylo à laquelle rattacher des utilisateurs, comme les membres d'une revue, d'un laboratoire, d'une équipe de recherche, etc. Ensuite, chaque utilisateur peut décider d'associer un ou plusieurs de ses articles à l'espace

51. L'historique de l'implémentation de l'espace de travail dans Stylo est disponible sur Github : `https://github.com/EcrituresNumeriques/stylo/pull/865`, consulté le 19 juin 2025.

52. Les problèmes rencontrés sur Stylo sont généralement documentés dans des tickets sur Github, comme en témoigne ces deux tickets : `https://github.com/EcrituresNumeriques/stylo/issues/615` et `https://github.com/EcrituresNumeriques/stylo/issues/835`, consultés le 19 juin 2025.

de travail, donnant ainsi accès à ces ressources à l'ensemble des membres que l'espace de travail contient. Ce fonctionnement permet alors de réduire le nombre de ressources consommées du côté du serveur, tout en répondant à ce besoin de partage des documents émis par la communauté de Stylo. Néanmoins, le statut de cette entité peut porter à confusion. Par exemple, prenons le cas d'une revue utilisant un espace de travail afin d'agréger tous leurs articles au même endroit dans Stylo, pour ensuite pouvoir les trier en numéro via la fonctionnalité de corpus. Tous les membres de la revue, ainsi que les auteurs, auraient alors accès à l'espace de travail. Dans Stylo, le créateur d'un article en est le propriétaire. Lorsqu'un utilisateur partage un document à un espace de travail, cette propriété n'est pas transférée à l'espace de travail, elle reste à l'utilisateur. Admettons qu'un auteur décide de partager un article avec l'espace de travail d'une revue le temps d'éditer le texte jusqu'à sa publication. Une fois publié, l'auteur décide de retirer son article de cet espace de travail. La revue perdrait alors l'accès au document source qui a été employé pour produire le document publié, ce qui n'est peut-être pas souhaitable. L'espace de travail a donc un statut très particulier, car il n'est pas la revue lui-même ni un espace de stockage pérenne des données. Il s'agit juste d'un espace de travail collaboratif permettant de fluidifier les échanges entre différents acteurs. Il y aurait donc d'autres stratégies à mettre en place si une revue souhaitait garder un accès pérenne aux documents sources.

À l'occasion de cet exemple sur les espaces de travail, nous avons également évoqué l'objet corpus [53]. Il s'agit là aussi d'un objet un peu particulier de Stylo. Historiquement, Stylo a été pensé pour rédiger et éditer des articles scientifiques. En ce sens, l'unité documentaire à l'intérieur de Stylo était cet article de revue. Toutefois, le besoin de pouvoir manipuler d'autres types de document s'est avérée nécessaire : des thèses, des mémoires, des livres, des billets de blog, des numéros de revue, etc. Chacun de ces documents se situe à une échelle documentaire différente, d'où résulte une incompatibilité avec cette unité documentaire qu'est l'article. De plus, ces nouveaux objets ne requièrent pas les mêmes jeux de métadonnées : un livre n'est pas décrit avec les mêmes informations qu'une thèse ou qu'un billet de blog. Cette situation nous a amené à penser le corpus en un sens le plus générique possible afin qu'il puisse répondre à ces différentes formes de document. Ainsi, la définition d'un document nouvellement créé dans Stylo se complexifie selon le type de corpus auquel ce dernier est rattaché. Ce n'est pas un élément obligatoire, toutefois, lorsqu'il est renseigné, il rend la nature du document explicite.

53. Les corpus ont été implémentés dans Stylo en février 2023 : `https://github.com/EcrituresNumeriques/stylo/pull/796`, consulté le 19 juin 2025.

Tous ces objets que nous venons de décrire sont définis à l'intérieur de l'API GraphQL. Pourtant, la circulation des informations entre le client et le serveur ne repose pas uniquement sur GraphQL, qui permet de structurer les données et de construire les requêtes liées aux données de Stylo par le biais de l'API. De la même façon que l'on n'envoie pas une lettre à quelqu'un sans service postal, les requêtes GraphQL ne peuvent pas circuler entre la base de données et le client sans un protocole pour les acheminer. Pour être livrée à destination, la carte postale doit respecter plusieurs conditions : elle doit comporter l'adresse du destinataire, être affranchie et enfin être déposée sur le réseau du service postal (soit dans un bureau de poste, soit dans une boîte postale). Si l'on ne respecte pas ce protocole et que l'on décide plutôt de jeter cette carte postale par la fenêtre, il est fort probable que celle-ci n'arrive jamais à destination. Le procédé est similaire pour les requêtes formulées sur les données de Stylo. Pour émettre ou recevoir une requête GraphQL, le client a besoin d'un protocole, soit un ensemble de règles pour formater et véhiculer la requête sur le réseau. Pour Stylo, le protocole employé est HTTP. Ce protocole induit un comportement particulier de notre requête puisque selon la méthode employée, les informations ne seront pas traitées de la même manière, comme nous l'avons vu dans la section IV.2 de ce chapitre.

Pourtant, l'une des particularités de l'API GraphQL est sa dimension agnostique par rapport au protocole de communication des informations employé, que ce soit HTTP ou des WebSockets ou autre. Malgré la capacité de GraphQL à être utilisable avec toutes les méthodes d'HTTP [54], une bonne pratique appliquée par la communauté GraphQL est l'emploi du protocole HTTP couplé à la méthode `POST` pour tous types de requêtes GraphQL (quel que soit le type de requête : une `query`, une `mutation` ou encore une `subscription`).

Malgré l'aspect agnostique de GraphQL, la forme textuelle des requêtes implique en elle-même un choix particulier de transmission des informations avec ce qu'il comporte comme avantages et inconvénients. En respect de ces bonnes pratiques, la méthode `POST` a été préférée dans Stylo, que ce soit pour récupérer ou pour envoyer des données entre client et serveur. Les informations sont alors formatées en JSON pour respecter les spécifications de la méthode `POST`. Ainsi, le document prend une nouvelle forme dans l'écosystème Stylo.

Extrait de code IV.6 : Requête HTTP contenant la requête GraphQL de mise à jour de la version de travail

```
1 {"query":"query updateWorkingVersion(articleId: ID!,
     \$content:
2 WorkingVersionInput!)
```

54. Voir `https://graphql.org/learn/serving-over-http/`, consulté le 24 février 2024.

```
3  {\n
4  article(article: \$articleId) {\n
5  updateWorkingVersion(content: \$content) {\n
6  updatedAT\n
7  }\n
8  }\n
9  }",
10 "variables":{"userId":"61d62.....",
11 "articleId":"65e0e38129637c0012ef7a",
12 "content":{"md":"Ajout du texte pour la requête \gls{http}
      'POST'"}}}
```

Dans l'exemple de requête IV.6, un bref extrait de Markdown a été ajouté à un article créé dans le but de mettre en évidence l'inscription du texte et son formatage dans Stylo à l'instant précis où le texte à l'écran est indexé dans la base de données. On remarque que la structure JSON de ce document contient une requête nommée *updateWorkingVersion*, ainsi qu'un ensemble d'objets comme *content* auquel le contenu en Markdown de notre article Stylo a été affecté. La structure sémantique et hiérarchique du Markdown n'a plus d'importance dans ce fragment de document. Le Markdown est réduit à l'état de donnée pour être indexé dans la base de données de Stylo. Ce qui importe dans la structuration des informations de ce document n'est pas le texte, mais tout un ensemble d'autres informations qui permettent au protocole de communication (la méthode HTTP `POST`) de faire circuler les informations jusqu'à la base de données, grâce, notamment, à l'identifiant de l'article.

L'élément principal de ce document JSON se situe dans le nom de la requête GraphQL : *query updateWorkingVersion*. Qu'est-ce donc que cet objet ? Précédemment, nous avons vu que parmi la liste des éléments constitutifs d'un article dans Stylo se trouvait un objet nommé `workingVersion`. Si le terme *update* dans la requête GraphQL fait référence à l'action de mise à jour, la notion de *working version*, version de travail, renvoie quant à elle à une toute autre notion introduite dans Stylo, et qui lui est indispensable, car c'est un élément essentiel à l'écriture collaborative asynchrone dans Stylo. Grâce à ce concept, Stylo ouvre la possibilité d'intégrer d'autres acteurs au processus d'écriture, permettant ainsi de modifier tant les contenus que le sens produit.

La version de travail est un concept qui introduit la notion de *version*, et fait référence aux versions des documents des systèmes de contrôle de version tels que Git, le plus répandu, son alternative Got pour la communauté OpenBSD, que l'on peut grouper avec d'autres logiciels comme Mercurial ou Darcs pour les systèmes distribués ou encore Subversion dans le cas d'un système de contrôle des versions centralisé.

Ces protocoles sont généralement utilisés lors de collaborations sur des projets de développement informatique, mais également pour des projets de littérature ou de *gittérature*[55], comme le nomme Servanne Monjour et Nicolas Sauret (2021).

Contrairement à un mode de collaboration synchrone, comme peut l'être le mode d'écriture collaboratif que l'on connaît essentiellement avec des outils comme GoogleDoc, HedgeDoc, EtherPad, etc., l'utilisation d'un système de contrôle des versions permet la collaboration asynchrone, dans un temps différé, sur la base de la mise à jour de l'arbre des versions.

Un système de contrôle des versions s'appuie sur un protocole pour créer une arborescence des différentes versions d'un document afin de préserver à la fois l'historique et une sauvegarde des documents. Ces protocoles permettent de suivre les modifications réalisées dans un document à partir d'états précis que l'utilisateur juge suffisamment stables pour être versionnés. Les versions sont donc des états déclarés par les utilisateurs. Dans le cas de Git, cette déclaration est signifiée par un *commit*, une étiquette que l'on associe au document lors de la création de la nouvelle version.

Dans Stylo, ce principe a été simplifié et implémenté sous la forme d'un gestionnaire linéaire des versions (il n'y a pas d'intégration d'un système composé de multiples branches, de commandes plus complexes de fusion de différents états du document, etc.), avec une version de travail, celle sur laquelle l'utilisateur écrit le texte, et un historique des versions antérieures à cette version de travail. La création d'une nouvelle version est un événement déclaratif auquel l'auteur associe une étiquette, tandis que les modifications de la version de travail sont automatiquement enregistrées dans la base de données.

Pour l'article de la revue *Humanités numériques*, ce sont 63 versions du document qui ont été créées dans Stylo. La plupart de ces versions concernent des modifications très précises à propos de corrections éditoriales et typographiques, ou encore de l'intégration des révisions dans le texte, toutes réalisées par l'équipe éditoriale de la revue *Humanités Numériques*. Ce texte connaît donc 64 états différents dans lesquels il est possible de naviguer : les 63 versions contenant l'historique de toutes les modifications et la version de travail que l'on voit affichée dans l'éditeur de texte Monaco.

L'implémentation de la notion de version, et de surcroît celle de version de travail, n'est pas juste un nom donné à une entrée de la base de données. C'est un élément structurel intégré au processus de matérialisation du document dans Stylo. En ce sens, le document dans Stylo ne comporte pas uniquement le texte modélisé dans la version de travail, mais contient cette version de travail et toutes

55. Ce néologisme est abordé dans le chapitre I.2 de la thèse.

les versions antérieures du document qui ont été déclarées par le ou les auteurs et les contributeurs.

Représentation du document dans le composant Monaco

Nous notons toutefois qu'il manque un élément pour compléter le circuit emprunté par les données dans Stylo, car les différentes phases de la matérialisation du document que nous avons abordé jusqu'à maintenant sont techniquement incomplètes. Ces phases que nous avons décrites, la saisie en texte brut et la modélisation GraphQL sous-entendent que la requête HTTP `POST` envoyée au serveur s'effectue en continu entre le client et le serveur ; ce qui n'est pas le cas. Une dernière phase documentaire se glisse à l'interstice de l'interface graphique et de la requête HTTP envoyée au serveur.

Ainsi, la création d'un document dans Stylo et la mise à jour des informations qu'il contient n'est pas synonyme d'une inscription continue dans la base de données que fournirait le canal HTTP. Une phase latente se situe au niveau de l'interface web, entre le moment où l'utilisateur frappe les touches de son clavier et le moment où la base de données est mise à jour. Cette phase est rendue visible par l'affichage d'un message au-dessus de l'éditeur de texte. Lorsque aucune touche du clavier n'est enfoncée pendant une seconde[56], le message « Last saved... » est remplacé par « *saving* » : la copie de travail vient d'être enregistrée dans la base de données MongoDB grâce à la requête GraphQL `updateWorkingCopy()`. Dans ce laps de temps, entre la frappe des mots au clavier et l'envoi de la requête au serveur, qu'advient-il du texte ?

Comme cela est mentionné précédemment, l'espace d'écriture de Stylo est un espace web. Pour y accéder, nous avons besoin d'un logiciel particulier – un navigateur ou un fureteur – capable d'interpréter du HTML, du CSS et d'exécuter du JavaScript. Lorsque l'on écrit dans Stylo – et de surcroit dans le composant Monaco –, le texte saisi doit être manipulable et interprétable par le navigateur pour pouvoir être envoyé sur le serveur.

Le rôle de Monaco est crucial à cet endroit, puisque le texte saisi n'est pas en HTML, mais en Markdown, et ne peut donc pas être directement interprété par le navigateur, qui lui n'est pas doté de cette capacité sans ce composant Monaco. Ainsi, lorsque l'utilisateur frappe sur une touche de son clavier, à cet instant précis, ce caractère n'est inscrit sur aucun support en dehors du rendu

56. Ce paramètre est configuré sur 1000 millisecondes dans le code source de Stylo à cette adresse : `https://github.com/EcrituresNumeriques/stylo/blob/1ce71a8e8b5f80be97678e4876ed27477b3a65d2/front/src/components/Write/Write.jsx#L144`, page consultée le 19 mars 2025.

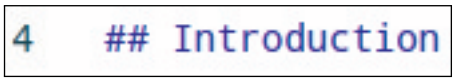


Figure IV.8 : Exemple de texte saisi en Markdown dans Stylo

visuel affiché à l'écran : il s'agit du premier fragment du document dans Stylo. Le document est partiel et n'existe encore que dans le navigateur de l'auteur.

Ce premier fragment de document combine deux types de contenus : le texte brut et le texte enrichi. À cela s'ajoute un autre composant, l'éditeur Monaco qui interprète les contenus pour les afficher à l'écran.

L'éditeur Monaco s'appuie sur des *modèles* pour fonctionner, avec lesquels les utilisateurs interagissent : le texte saisi est transformé selon le modèle utilisé par Monaco en un document interprétable par le navigateur [57]. Ensuite, chaque modèle est rattaché à une URI (que l'on peut identifier dans Stylo avec l'identifiant des articles présent dans l'URL). C'est notamment grâce à cette identification unique que Monaco peut manipuler le DOM du navigateur pour créer le document et son rendu graphique à partir du texte brut saisi dans l'éditeur. Pour que le texte au format Markdown soit interprétable dans un navigateur, Monaco manipule le DOM pour modifier la représentation du document afin qu'elle soit compatible avec son environnement.

Le DOM est une représentation abstraite d'un document HTML exécutée dans le navigateur. Tous les éléments structurés à l'intérieur de ce document deviennent des objets, des nœuds manipulables avec un langage de programmation tel que JavaScript. C'est grâce à ce procédé qu'une page web est rendue dynamique : le contenu de la page web peut être modifié au gré des usages fait dans la page web sans jamais modifier le document HTML présent sur le serveur (et appelé via son URL). Puisque la construction du DOM dépend du navigateur employé, nous pouvons en déduire que ce document sera différent selon le navigateur utilisé ou selon les différentes versions d'un même logiciel. Pour accéder à ce DOM, il suffit d'ouvrir les outils de développements du navigateur et d'inspecter le contenu de la page HTML.

La figure IV.8 et l'extrait de code IV.7 montrent le texte saisi à l'écran et ce qui est inscrit dans le DOM.

Extrait de code IV.7 : Affichage du DOM pour le texte de la figure IV.8

```
1 <div style="top:57px;height:19px" class="view-line">
2   <span>
3     <span class="mtk6">##\ Introduction</span>
4   </span>
```

57. La documentation disponible sur le répertoire GitHub de Monaco explique ce concept, voir `https://github.com/microsoft/monaco-editor?tab=readme-ov-file#models`, page consultée le 25 septembre 2024.

```
5 </div>
```

L'état du texte inscrit dans le DOM généré par Monaco est différent de celui qui apparaît à l'écran, dans l'interface de Stylo. Le Markdown se retrouve encapsulé dans des balises attribuées par l'éditeur Monaco tandis que la syntaxe Markdown se retrouve à l'état de chaîne de caractères : la balise Markdown utilisée pour signifier un titre de niveau 2 (`##`) n'a alors plus de valeur sémantique. Le DOM interprète la structure HTML des informations contenues dans Monaco pour les afficher de façon à ce que l'utilisateur puisse avoir un rendu graphique du texte qu'il a saisi, comme la colorisation syntaxique des balises. Néanmoins, pour Stylo, le texte saisi n'a en lui-même aucun sens et il ne doit pas en avoir puisque c'est cette particularité qui rend l'écriture calculable.

Rapport de l'auteur avec Stylo

Ces différents aspects techniques de la circulation des informations dans la surcouche web de l'environnement d'écriture ajoutent un ensemble de médiations dans le processus de matérialisation du document dans Stylo.

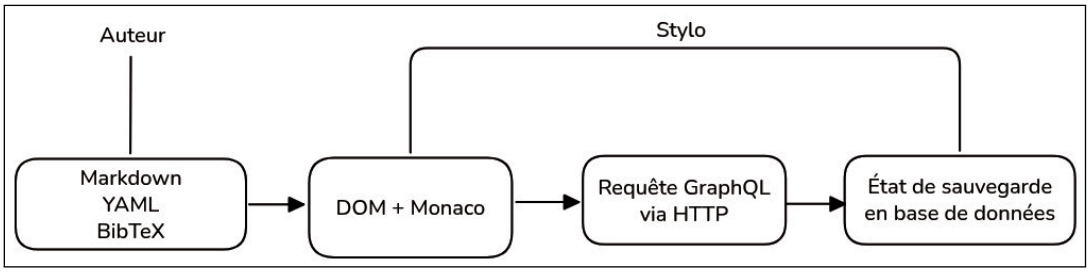


Figure IV.9 : Schéma des étapes de matérialisation du document dans Stylo

Nous pouvons résumer ces différentes médiations sous la forme de quatre étapes, comme cela est mentionné dans IV.9.

Le texte brut et enrichi sont tout d'abord saisis par l'utilisateur en Markdown, YAML et BibTeX dans le composant Monaco. Ensuite, Monaco manipule le texte saisi dans le DOM. Puis ces éléments sont envoyés sur le serveur au moyen d'une requête GraphQL formatée en JSON, qui est elle-même contenue dans une requête HTTP utilisant la méthode `POST` comme modalité de circulation de l'information, avant d'être stockée dans une base de données en texte brut.

Chacun de ces états a une signification particulière. Le premier état est la projection d'une structure de l'information, tandis que le deuxième en permet l'interprétation et l'affichage par le navigateur, la troisième est une représentation formatée pour circuler entre un client et un serveur et enfin, la quatrième, est à

l'état de stockage, prête à être appelée pour réaliser le chemin en sens inverse ou être exportée.

Ce récapitulatif illustre le fonctionnement de l'écriture dans Stylo, et montre que cet acte ne consiste pas seulement en une simple saisie du texte à l'écran, mais en la mobilisation de tous les agents de ce dispositif technique. L'impression d'écrire en Markdown, YAML et BibTEX n'est alors que la surface de cette écriture numérique, puisque Stylo écrit de son côté un texte bien différent.

Écrire dans Stylo nécessite ainsi une multitude de médiations, dont la plupart se déroulent hors de toute action de la part de l'auteur à l'intérieur de l'environnement d'écriture. Stylo rédige et inscrit du contenu dans la mémoire numérique pour produire du sens. L'exécutabilité de l'écriture numérique prend ici toute son ampleur, puisque cet environnement est responsable de l'écriture de trois états documentaires sur les quatre que nous avons mentionnés.

En ce sens, l'article de la revue *Humanités numériques* ne peut exister uniquement parce qu'il est produit par ces différentes médiations durant la matérialisation du document. Ce sont elles qui dictent ce qu'il est possible d'inscrire ou non dans le document, et donc le sens accordé au texte. Ainsi, le document qui en résulte est à la fois composé par les multiples versions contenant tous les états du Markdown, YAML et BibTEX ; mais aussi par la totalité de ces phases documentaires éphémères nécessaires à l'établissement d'un document dans l'espace web qu'est Stylo.

IV.5 Conclusion

Lors de ce chapitre, nous avons procédé à une analyse de la production du sens dans l'éditeur de texte Stylo. À la différence des environnements WYSIWYG étudiés précédemment, Stylo se présente comme un éditeur de texte WYSIWYM accessible uniquement depuis le Web. Ainsi, Stylo se distingue en premier lieu de MSWord et LibreOffice par une écriture en texte brut doublé d'un texte enrichi, puis, en deuxième lieu, par l'ajout d'une couche intermédiaire supplémentaire entre l'usager et la machine : le Web.

Stylo n'a pas un format propre à cet environnement, par exemple un format de type .stylo ; contrairement aux archives ODT et DOCX qui renferment quant à elles plusieurs fichiers en XML et *WordprocessingML*. Sans ce format unique, le document source de Stylo se retrouve alors divisé en trois parties au minimum : le Markdown, le BibTEX et le YAML.

Au-delà d'une volonté de transparence vis-à-vis des utilisateurs, le fait de ne pas créer un format dédié à cet environnement d'écriture provoque la dispersion du document dans de multiples fragments à l'intérieur Stylo. Le document s'éparpille

dans différents espaces, et ne correspond plus à la forme habituelle d'un document qui serait associé à un seul fichier. De plus, le document n'est pas stocké dans un espace local d'un ordinateur personnel, mais l'est sur un serveur distant auquel nous accédons via le Web. Le Web, avec ses protocoles et canaux de communication, fragmente encore plus le document dans Stylo, que ce soit par le biais de l'interface dans le navigateur, de la modélisation du texte en HTML, ou encore dans ses multiples formes documentaires, prises lorsque les informations transitent jusqu'au serveur par l'API GraphQL et le protocole HTTP.

Ces différentes formes (Bachimont 2022*b*) que prend le texte sont plus que de simples représentations. Chacune à une signification et un usage qui lui est propre. Par exemple, la forme en Markdown brut ne peut pas circuler en l'état avec le protocole HTTP, il lui manque toute une série d'informations et une transformation vers un autre format, le JSON, pour employer ce canal de communication : transformation dont s'occupe Stylo. Nous notons que parmi ces quatre formes documentaires produites pour écrire, une seule l'est par l'auteur tandis que les autres formes sont écrites par Stylo. Or, le document dans Stylo n'est pas uniquement constitué par le texte saisi, mais bien par l'ensemble de ces formes documentaires et des médiations qu'elles sous-entendent.

Dans cette matérialisation, le Markdown, le YAML et le BibTEX n'ont finalement que peu de signification dans la constitution du document. Ce sont plutôt les concepts, les formats, les protocoles, les librairies de code, le *hardware*, qui, agencés en tant qu'environnement, crée le dispositif à partir duquel le document émerge. Cet agencement est relativement fragile : si une des briques technologiques venait à ne plus fonctionner, c'est l'ensemble du processus de matérialisation du document qui devient inopérant. Le cas échéant, le document ne peut plus être produit et l'écriture non plus. Phénomène que l'on caractérise par le terme de *bogue* ou de *bug*.

Le processus de formatage et d'encodage des textes ne se limite donc pas à l'affichage du texte à l'écran dans une seule forme. Pour être inscrit dans la mémoire numérique, le texte est transformé en données numériques. Ces dernières sont alors segmentées, fragmentées et (ré)organisées au gré des fonctionnalités implémentées dans Stylo, et des calculs que l'environnement réalise pour (re)construire un document et le sens qu'il transporte.

Ainsi, le modèle textuel implémenté dans Stylo ne se limite pas à la modélisation des éléments du texte comme cela était défini par le modèle OHCO, mais introduit d'autres notions ou concepts dans le processus de construction du document à travers différentes médiations techniques, tel que c'est le cas du concept de version de travail utilisé pour mettre à jour la base de données et créer un historique des versions du texte. Au vu de ces modalités de production

de l'écriture, il devient évident que le Web, à travers les techniques qui lui sont empruntées dans Stylo, est une surcouche significative du document et qu'elle n'est ni abstraite ni neutre dans ce processus.

Conformément à ce que nous verrons dans le prochain chapitre, ce modèle de production de l'écriture de Stylo ne dépend pas uniquement de cette modélisation interne à Stylo, car les documents seraient voués à être enfermés dans cet environnement. Pour en sortir, il dépend également de la modélisation textuelle du module d'export et surtout de la représentation du document que la pierre angulaire de ce module porte : Pandoc. C'est parce qu'il y a une relation entre la production dans Stylo et sa transformation via Pandoc que la syntaxe de Stylo a pu être modifié afin de proposer un balisage sémantique répondant aux besoins de l'écriture savante. En ce sens, la modélisation textuelle de l'export joue également un rôle dans la constitution du modèle textuel de l'espace de production de l'écriture. Finalement, le choix du Markdown est plutôt insignifiant tant les modifications apportées au Markdown ne ressemblent plus à du Markdown mais à un langage de balisage augmenté et compatible avec Pandoc. C'est-à-dire que dans une certaine mesure, le Markdown en lui-même n'a aucune importance, car il ne signifie pas ces objets particuliers sans Pandoc.

Comme nous l'avons vu, cette forme d'écriture en suivant les formats de texte brut ne correspond d'ailleurs qu'à une forme sur les quatre que le document prend tout au long du processus de production. Ces formats à la syntaxe légère ne sont donc là que pour initier le processus d'écriture et pas pour réellement modéliser le texte. C'est Stylo qui se charge de cette modélisation en vue de l'export de ces contenus via Pandoc. En somme le modèle textuel de Stylo est complètement lié à Pandoc puisque l'agencement des formats de texte brut dans Stylo est pensé selon la structure du document que produit Pandoc durant l'export (même s'il est tout à fait possible de se passer du module d'export).

Toutes ces dynamiques éditorialisent et constituent les traces d'une épistémologie du document, et cela, en amont des transformations opérées par le reste de la chaîne éditoriale. Autrement dit, écrire dans l'environnement Stylo produit quelque chose qui ne serait pas identique dans un autre environnement, car les dynamiques observées seraient affectées par d'autres facteurs et produiraient ainsi un autre document. Le choix de l'environnement d'écriture constitue en conséquence un choix politique puisque cet environnement agit et produit une matérialité singulière.

Lors du prochain chapitre, nous nous concentrons sur les transformations des documents produits dans les environnements d'écriture numérique Stylo, MSWord et LibreOffice afin que ces derniers puissent être publiés.

Chapitre V

Les transformations éditoriales avec Lodel et le Pressoir

V.1 Introduction

La production d'un document numérique en suivant un modèle textuel ne garanti pas que celui-ci puisse être directement diffusé et distribué dans l'espace public, c'est-à-dire publié. Dans certains cas de figure, par exemple pour les textes que nous avons étudiés dans les chapitres III et IV, il a été nécessaire de les transformer afin de réaliser cette action de publication. Ces différentes transformations, d'un document initial permettant d'obtenir un document publiable, forment une chaîne de production de l'écrit dans laquelle s'inscrit l'acte d'éditer (Sauret 2020 : 46) . Le texte est alors édité afin qu'il corresponde à une politique éditoriale propre à une maison d'édition, à une revue, à une collection, à une identité éditoriale ou encore à des choix esthétiques.

L'emploi du terme transformation est essentiel : il reflète d'une part les modifications typographiques apportées au document et, d'autre part, les modifications de la structure même du document. En effet, en contexte numérique, ce processus éditorial exige bien souvent un changement du format employé par l'auteur et, par extension, du modèle textuel utilisé pour structurer le texte. La transformation qui s'opère en éditant un document est donc une transformation au niveau de la théorie du texte qui structure le contenu d'un document.

Selon une perspective relativement générale, l'acte d'éditer un texte désigne de prime abord la composition typographique, c'est-à-dire une « opération manufacturière » visant à produire un artefact à travers la mise en forme d'un texte, par exemple un livre (Mailloux 1980 : 30). Les débuts de l'édition sont alors souvent associés à l'impression et aux débuts de l'imprimerie au XVe siècle, celle mettant en œuvre des caractères mobiles en plomb ou en bois. Par la suite, l'édition devient un terme plus englobant et désigne un ensemble d'actions sur le texte qui dépasse cette composition typographique, comme peuvent l'être la relecture ou encore la

diffusion, que Fauchié définit en tant qu'activité économique de production du savoir (2024 : 100). Bien qu'il implique différents savoir-faire et spécialités, ce mode de production qu'est l'édition s'incarne à travers une figure : l'éditeur.

L'éditeur, que ce terme désigne soit une personne morale – par exemple une maison d'édition – soit une personne physique, est une figure qui se cristallise et se traduit par un métier à part entière à partir du XIXe siècle, lorsque les éditeurs deviennent « la nouvelle puissance sur la chaîne des métiers du livre » (Mollier 2015 : 260). Selon Vitali-Rosati et Sinatra, l'édition, et par extension l'éditeur, est caractérisée par trois fonctions principales : « le choix des contenus, la légitimation de ces contenus et leur diffusion » (Sinatra et Vitali-Rosati 2014 : § 3). À travers ces fonctions, l'éditeur endosse « la (double) responsabilité matérielle et morale d'une œuvre » (Poirier et Genêt 2014 : § 6). La fonction principale d'un éditeur se situe à l'endroit de cette responsabilité (Mollier 2015 : 267), qui fait office d'autorité (Broudoux 2022 : 61), et permet à un auteur ainsi qu'à un livre d'exister par son entremise. Sans cette prise de responsabilité de la part de l'éditeur, le livre reste un manuscrit et l'auteur un écrivain.

Alors que les technologies numériques occupent désormais une place centrale dans notre accès à l'information et au savoir, elles ont profondément reconfiguré l'univers de l'édition et les dispositifs de légitimation des contenus par l'apport de nouveaux moyens de réalisation des fonctions éditoriales, qui dès lors ne se cristallisent plus forcément sous la figure de l'éditeur (au sens de la maison d'édition), mais sont éparpillées sous d'autres instances numériques (Epron et Vitali-Rosati 2018).

Dans ce contexte, la structuration des contenus devient un enjeu majeur : elle permet d'anticiper leur circulation, leur indexation et leur visibilité, tout en conditionnant leur découvrabilité, leur pérennité et leur inscription dynamique au sein de réseaux d'objets. Un tel agencement relève d'un processus d'éditorialisation, par lequel l'objet publié s'intègre dans des espaces numériques.

Tandis que nous avons abordé l'écriture dans sa phase de production dans les chapitres précédents, nous nous intéressons dans celui-ci à sa phase de préparation à la publication, durant laquelle la fonction éditoriale intervient dans le texte pour le restructurer sémantiquement à l'aide d'un modèle textuel adapté aux enjeux de la publication de ce texte.

Afin d'observer ces transformations du texte, nous nous appuyons sur deux objets éditoriaux différents que nous avons étudiés précédemment : l'ouvrage *Contribution numérique*, ainsi que l'article publié dans la revue *Humanités numériques*. Contrairement à la méthode employée dans les chapitres précédents, il n'est pas question dans ce chapitre de comparer ces deux objets, car les processus de production de ces artefacts sont distincts puisque l'un est un livre, tandis

que l'autre est un article de revue. Néanmoins, nous continuons d'appliquer la méthodologie décrite en introduction de la thèse à ces cas d'étude afin de mettre en évidence les médiations techniques qui permettent d'effectuer ces transformations. À défaut d'effectuer une comparaison, ces deux cas d'étude sont complémentaires dans le sens où ils permettent de montrer qu'avec des technologies sensiblement similaires, les choix éditoriaux et les modèles textuels qui y sont associés produisent des documents complètement différents.

Pour démontrer cela, ce chapitre est articulé en trois sections. La première est axée sur la notion d'éditorialisation. Il s'agit d'un concept majeur pour comprendre ces transformations éditoriales et le passage d'un modèle textuel à un autre. Dans cette première section, nous revenons sur l'histoire de ce concept et les différents courants qu'il emprunte depuis le début des années 2000 (au moins pour les courants francophones).

La deuxième section s'attache à analyser les transformations subies par l'article rédigé dans l'éditeur de texte Stylo avant sa publication en 2023 dans le numéro 8 de la revue *Humanités numériques*. En suivant les différentes étapes, depuis la rédaction dans Stylo jusqu'à l'intégration dans Lodel, le système de gestion de contenus de la plateforme OpenEdition Journals, nous observons, à chaque étape et pour chaque transformation du processus éditorial, les modifications de sens que celles-ci provoquent, jusqu'à la mise en ligne finale de l'article.

Enfin, la dernière section porte sur le Pressoir, la chaîne éditoriale multimodale utilisée pour produire les différentes versions du livre *Contribution numérique : cultures et savoirs*. Ce cas d'étude met en évidence les frictions qui surgissent lorsque le modèle textuel de la phase de production de l'écriture ne correspond pas aux modélisations requises pour les transformations. Les chapitres du livre passent ainsi par plusieurs représentations – modèle OHCO, arbre de syntaxe abstraite, puis modélisation mathématique pour le produire PDF – montrant comment le Pressoir articule ces visions pour générer les livrables finaux.

V.2 Écriture et éditorialisation

La publication d'un document évoque l'univers de l'édition, notamment en raison de la mise en œuvre de tout un processus impliquant la composition typographique, l'impression des documents, leur distribution et la communication autour de ces derniers, dans le but de conférer une certaine visibilité à cette publication auprès des publics cibles. Or, cette conception traditionnelle de la publication se trouve reconfigurée par les technologies de l'information numérique, nous dit Broudoux (2022 : 58).

Dans l'espace numérique, il existe plusieurs voies pour rendre public un texte que l'on écrit. Il est tout à fait possible de procéder d'une manière analogue à celle de la publication des textes imprimés, c'est-à-dire de publier un texte à l'aide d'un tiers qui aurait le rôle de l'éditeur, une maison d'édition par exemple. Toutefois, grâce à Internet et au Web notamment, la publication d'un texte peut être réalisée sans passer par ces éditeurs : les contenus peuvent être générés par des individus et rendus accessibles directement depuis un site web ou une plateforme, par exemple. Parmi ces procédés, la fonction éditoriale est souvent déplacée du côté de l'auteur, ce qui soulève des interrogations quant à la responsabilité matérielle et morale d'une œuvre qui incombait auparavant à l'éditeur et à son statut lorsqu'elle est publiée sans son intervention (Broudoux, Bootz et al. 2005).

Dans ce contexte où la maison d'édition est absente, il est pertinent de s'interroger sur la responsabilité de la publication et de la diffusion de contenus dans l'espace numérique. C'est à cette problématique qu'entend répondre la notion d'*éditorialisation*, convoquée (en France) notamment par les sciences de l'information et de la communication au début des années 2000 (Broudoux 2022 : 95).

L'éditorialisation est une notion utilisée en français depuis le début du XXIe siècle. Il s'agit d'un anglicisme dérivé du verbe *editorialize* qui signifie « exprimer une opinion sous la forme d'un éditorial »[1]. Si elle est parfois utilisée en ces termes en français, sa signification est principalement concentrée sur les formes de production des connaissances liées au numérique. Pour Vitali-Rosati, cette « resémantisation de ce concept » n'est pas en contradiction avec la signification anglaise, et « s'avère même utile », car elle « conserve un lien étroit avec la notion d'opinion en ce qu'il se réfère à la production de contenus qui expriment une sorte d'opinion » (Vitali-Rosati 2016 : § 2-3).

Les premières occurrences du terme éditorialisation apparaissent dans les travaux de Pédauque, d'abord en 2005 dans le *Rapport final de l'action spécifique « Document et Organisation »*, puis dans une sous-section d'un chapitre intitulé « Interactions document organisation » présent dans l'ouvrage *La redocumentarisation du monde*.

Dans le rapport de l'action spécifique, Marie-France Peyrelong et Brigitte Guyot mobilisent l'éditorialisation[2] car ce terme :

1. Voir le dictionnaire Merriam Webster à l'URL suivante : `https://www.merriam-webster.com/dictionary/editorialize`, consultée le 19 mai 2025.

2. Nous notons que la toute première occurrence que nous avons trouvée du terme éditorialisation remonte à 2004. Cette occurrence apparaît dans une version de pré-publication d'un article rédigé par Brigitte Guyot pour la revue *Hermès*, mais n'a pas été retenue dans la version publiée (2004).

> semble plus adéquat que celui de travail d'édition, parce qu'il signifie un processus collectif qui engage des acteurs multiples en tous points d'une organisation. Il peut se définir comme un processus d'organisation, de stabilisations et de traductions successives, plus ou moins temporaires, qui joue sur la forme, le statut du document mais aussi sur sa gestion et sa circulation. Il trans-forme, réalise des changements de statut et de forme, organise des circulations circulation entre collectifs différents (passage d'une situation à d'autres) (M. Blanc et al. 2005 : 84).

Concernant la seconde occurrence dans l'ouvrage sur la redocumentarisation, l'éditorialisation y est mobilisée dans la continuité de la première occurrence et rend compte du processus éditorial dans lequel est engagé un document. L'emploi de ce concept, plutôt que celui d'édition, permet de prendre de la distance vis-à-vis de ce dernier, et cela afin de pouvoir les comparer. En effet, Pédauque constate

> un élargissement et une recomposition des rôles traditionnels : celui d'auteur (qui n'est plus unique ni identifié formellement) et d'éditeur, avec des fonctions nouvelles comme celles de contributeur, de valideur, d'administrateur, qui ne sont pas nécessairement assignés au même acteur (Pédauque 2007 : 172).

Selon cette première définition, l'éditorialisation permet de considérer les étapes éditoriales qui ne sont pas prises en charge par le processus d'édition non numérique.

Si Pédauque verbalise cette problématique, que reprennent des chercheurs en SIC tels que Payeur et Zacklad (2007), nous retrouvons l'éditorialisation quelques mois plus tard la même année, dans un chapitre rédigé par Bachimont.

> L'idée centrale de cet article est que l'indexation fine du contenu rendue possible pour le numérique introduit un rapport nouveau au contenu et au document. Alors que selon l'indexation traditionnelle l'enjeu est de retrouver le ou les documents contenant l'information recherchée, l'indexation fine du contenu permet de ne retrouver que les segments concernés par la recherche d'information et de paramétrer l'usage de ces segments. Si le document demeure présent dans les résultats de la recherche dans une indexation et recherche traditionnelles, il n'en est pas de même dans une recherche profitant de l'indexation fine du contenu, dans la mesure où les segments peuvent se désolidariser du contenu dont ils sont issus, perdant ainsi leur origine et nature documentaires. Devenant des ressources, ces segments sont remobilisés pour la production d'autres contenus dont

> ils constituent les composants. La finalité n'est plus de retrouver des documents mais d'en produire de nouveaux à l'aide des ressources retrouvées. On passe ainsi de l'indexation pour la recherche à l'indexation pour la publication. Comme cette dernière s'effectue selon des règles et des normes, on parlera plutôt d'éditorialisation, pour souligner le fait que les segments indexés sont enrôlés dans des processus éditoriaux en vue de nouvelles publications (Bachimont 2007*c* : 314-5).

En expliquant le passage d'un document non numérique à un document numérique, Bachimont illustre cette phase que Pédauque nomme redocumentarisation et les évolutions qu'elle implique. Ce chapitre de Bachimont précise les travaux de Pédauque en traitant de l'organisation des informations à travers l'indexation, la fragmentation et la recombinaison des contenus dans l'espace numérique.

Cette première couleur donnée à l'éditorialisation est orientée sur le processus numérique de production et de diffusion d'un document. Broudoux (2022 : 96) résume ce processus en reprenant les différentes phases proposées par Lipsyc et Ihadjadene (2013) lorsqu'ils insèrent l'étape d'architecture de l'information au sein de l'éditorialisation :

- En amont, la phase de conceptualisation de métamodèles pouvant créer une chaîne éditoriale complète et la phase de modélisation de ses constituants comme les types de documents, les modèles éditoriaux,
- la phase de documentation comprend la collecte, le recyclage et la phase de marquage par des métadonnées qui traitent, assemblent et enrichissent les documents,
- la phase d'édition qui est l'étape de fabrication proprement dite où s'inscrivent les genres éditoriaux,
- la phase de médiatisation constituée par les techniques de communication et de diffusion automatisée,
- la phase des usages et de la réception.

L'éditorialisation ainsi définie s'avère être un processus fini dans le temps, avec un début marqué par l'intention d'un auteur, et les différentes étapes qui mènent à sa diffusion et à sa réception par des lecteurs.

Crozat abonde également dans le sens d'un processus en proposant la notion de *rééditorialisation*.

> Le terme de rééditorialisation est un néologisme qui a émergé dans le domaine du document numérique pour désigner le processus consistant à reconstruire un nouveau document à partir d'archives. La

> construction de ce mot tente une première synthèse entre les concepts d'édition au sens de publication d'une œuvre, d'éditorialisation au sens d'expression d'un point de vue propre, de réédition au sens de nouvelle proposition de lecture. Elle tente une seconde synthèse entre les fonctions d'éditeur, celui qui met en forme et diffuse, et d'auteur, celui qui écrit, fonctions qui tendent à se mêler dans le contexte du numérique. La rééditorialisation est donc la publication d'une œuvre originale dans son point de vue, sa forme, sa scénarisation, à partir de contenus qui ne le sont pas tous (Crozat 2012 : 7).

Ce néologisme, proposé par Crozat, signifie sensiblement ce que les sciences de l'information et de la communication nomment éditorialisation. D'ailleurs, l'emploi de ce néologisme est controversé et le concept d'éditorialisation lui est parfois préféré, comme c'est le cas de Lipsyc et Ihadjadene (2013).

Cette vision du processus fini ne fait pas l'unanimité au sein de la communauté qui emploie le concept d'éditorialisation. En effet, si l'on s'oriente de l'autre côté de l'Atlantique, au Québec notamment, Vitali-Rosati fait un autre usage de ce concept depuis la philosophie.

À partir de 2008, il coorganise un séminaire à cheval entre la France et la Québec avec Gérard Wormser, Louise Merzeau, Carole Dely, Nicolas Sauret et Michael Sinatra intitulé *Écritures numériques et éditorialisation* pour « étudier l'impact du numérique sur la circulation du savoir » (Vitali-Rosati 2015*b*). Le séminaire se poursuit jusqu'en 2018[3] et donne lieu à diverses publications, par exemple celles de Louise Merzeau (2013*a*,*b* ; 2016 ; 2017) ou celles de Marcello Vitali-Rosati (2014 ; 2016 ; 2018*a* ; 2020*a* ; 2021) (Vitali-Rosati et Monjour 2017).

Les réflexions menées à l'occasion de ce séminaire et des différentes recherches de chacun, qu'il s'agisse des *Entretiens du Nouveau Monde Industriel (ENMI)* ou d'un *road trip* sur l'autoroute transcanadienne, aboutissent à une autre définition de l'éditorialisation qui n'est plus pensée comme un processus fini, mais comme un processus continu et ouvert.

Selon cette perspective, l'éditorialisation n'est plus seulement un processus d'édition numérique augmenté, mais un « tressage » de médiations qui « exploite la fragmentation des contenus » propre au numérique pour organiser et structurer les connaissances « de manière à faciliter leur mémorisation, leur évocation et leur exploitation » (Merzeau 2013*a* : 114).

3. Les différents programmes du séminaire sont encore disponibles sur le site web de l'Institut de recherche et d'innovation : `https://www.iri.centrepompidou.fr/evenement/nouvelles-formes-editorialisation/`, page consultée le 21 mai 2025.

Le dispositif mis en place durant les ENMI en 2012 illustre bien ce propos. Pour cet événement, une série de plateformes a été mise à disposition des publics et des acteurs concernés. Ces plateformes se composaient, premièrement, d'un dispositif destiné à la collecte ou à la production de données, tant en amont qu'en aval de l'événement. Deuxièmement, un autre dispositif a été spécifiquement dédié à la couverture en direct de l'événement. Même si elles étaient autonomes, ces différentes applications étaient reliées entre elles, ce qui permettait de faire circuler les informations et de les rendre accessibles sous différentes formes en ligne, à travers un glossaire, des annotations collectives, une représentation du colloque sous forme de graphe ou encore une carte heuristique qui spatialise les cheminements argumentatifs de chaque conférence (Merzeau 2013*a*).

À une échelle locale, cet événement dissipe les notions d'auteur et d'éditeur à travers tout le dispositif. Les contenus produits le sont tout autant par les intervenants, que par les participants qui se saisissent des plateformes pour accéder et/ou ajouter des informations. En ce sens, les plateformes ne sont pas de simples supports de l'événement, puisqu'elles fragmentent et réindexent les contenus, elles les manipulent, puis les réagencent selon une structure ou un modèle d'affichage, ce qui caractérise l'éditorialisation.

En outre, la configuration de l'événement se distingue par son caractère non conforme à un processus éditorial délimité. En effet, il est difficile d'en déterminer le point de départ ou d'en prévoir la clôture. Par exemple, dans le cadre de la préparation de l'événement, une série de décisions préliminaires ont été nécessaires, notamment celle concernant la diffusion des contenus agrégés sur des plateformes web. Cette étape de conception pourrait-elle être interprétée comme le point de départ de l'éditorialisation de l'événement ? Cela est peut-être le cas selon une perspective éditoriale, mais elle omet de mentionner le fait que les intervenants et les participants ont, systématiquement ou occasionnellement, eu recours au numérique pour préparer leurs interventions ou accéder à des contenus tiers en dehors des plateformes mises à disposition, que ce soit en amont, durant ou après l'événement. Il est légitime de se demander dans quelle mesure ces actions peuvent être considérées comme relevant du champ de l'éditorialisation.

Si l'on regarde en aval de l'événement, le choix d'employer les technologies du Web ne permet pas de préserver les productions de l'événement de manière pérenne. À ce jour, le site web de l'événement n'est plus disponible en ligne à son URL `http://enmi12.org`. Pour y accéder, il est nécessaire de se rendre sur les archives du Web, celles disponibles sur la plateforme *Internet Archive*, par exemple. Lorsque l'on consulte ces archives du Web via leur *Wayback machine*, les contenus de l'événement qui étaient stockés en base de données ne sont plus disponibles, car les contenus dynamiques générés à la volée sont plus difficiles

à archiver[4]. L'information accessible dans ces archives est alors incomplète et morcelée. Le fait qu'elles soient partielles n'est pas un problème, c'est même significatif des choix techniques qui ont été réalisés. Ceci résulte à la fois des choix de modélisation et d'organisation des informations en 2012 pendant l'événement, mais également des processus d'archivage du Web.

Décrite de cette manière, l'éditorialisation se dilue entre différents acteurs et technologies numériques à plusieurs échelles spatio-temporelles : ce n'est plus un processus que l'on peut délimiter comme on le ferait d'une chaîne d'édition (Vitali-Rosati 2016). « L'éditorialisation désigne la manière dont les outils, les pratiques émergentes et les structures déterminées par les outils engendrent une relation différente au contenu lui-même » (Vitali-Rosati 2018*a* : 63). L'éditorialisation bifurque d'une chaîne de traitement de l'information, et de l'action de sélection des informations via un système technique d'organisation des connaissances, à la manière dont cette action est façonnée par les caractéristiques techniques et culturelles de l'environnement numérique (Vitali-Rosati 2018*a* : 64). Dans ce cas-ci, l'éditorialisation n'est plus caractérisée par l'intention d'un auteur : il n'y a pas d'intention et s'il y en avait une, son importance serait mineure dans la réalisation de tout ce processus.

Pour circonscrire cet aspect de l'éditorialisation, Vitali-Rosati en donne la définition suivante :

> L'éditorialisation est l'ensemble des dynamiques qui constituent l'espace numérique et qui permettent à partir de cette constitution l'émergence des sens. Ces dynamiques sont le résultat de forces et d'actions différentes qui déterminent après-coup l'apparition et l'identification d'objets particuliers (personnes, communautés, algorithmes, plateformes...) (Vitali-Rosati 2021 : § 55).

Ainsi, ce n'est pas tant la production d'un contenu qui est concernée par l'éditorialisation que l'ensemble des médiations nécessaires à la production d'un phénomène. Si cette production nécessite des médiations techniques, ces médiations, à leur tour, mobilisent d'autres dynamiques et d'autres agencements d'objets particuliers pour émerger. Ainsi, cette production du sens émerge d'un agencement complexe de dynamiques, que l'on peut considérer comme des conjonctures médiatrices où chaque médiation agit en tant que coupure agentielle de la production dudit phénomène[5]. Pour cette raison, les médiations et ce qu'elles produisent ne peuvent être déterminées *a priori*, car il ne s'agit pas d'une construction que l'on peut déterminer ; elles le sont par contre *après-coup*, puisque c'est une fois

4. Voir le site web des ENMI : `https://web.archive.org/web/20131022193437/http://enmi12.org/`, site consulté le 21 mai 2025.

5. Le chapitre I.3 contient plus d'éléments sur les coupures agentielles.

la coupure agentielle effectuée que l'on peut identifier des éléments, des objets ou des propriétés significatives.

Ces deux visions de l'éditorialisation, l'une en tant que processus fini et restreint marqué initialement par l'intention d'un auteur, et l'autre en tant que processus ouvert, diffèrent également à l'endroit de la question de l'autorité posée initialement par ce concept au début des années 2000. En effet, nous avons vu que l'emploi du concept éditorialisation vient du fait qu'en environnement numérique, les fonctions d'auteur et d'éditeur, telle qu'elles étaient conçues dans un environnement pré-numérique, se retrouvent perturbées. Cette problématique imprègne les deux définitions que nous venons de donner de l'éditorialisation. Aussi, chacune d'entre elles y répond depuis son prisme.

Une autorité se manifeste en tant que relation entre deux agents, l'un qui détient ce pouvoir et l'autre qui s'y soumet. Nous rencontrons cette autorité sous différentes formes dans nos sociétés occidentales. Elle s'incarne à travers différentes figures, que ce soit l'autorité familiale (les parents, les aînés), l'autorité religieuse, l'autorité étatique ou encore l'autorité conférée à un auteur.

La notion d'auteur moderne s'articule autour de deux attributs fondamentaux : la création d'une œuvre et la reconnaissance juridique de sa propriété (Hesse 1990 ; Rose 1994 ; Sapiro 2014). On peut considérer que cette notion émerge, telle que nous la connaissons dans la modernité, au XVIIIe siècle. Une de ses fonctions consiste à garantir à la fois l'originalité des contenus et leur fiabilité. Cette dernière forme d'autorité est souvent accordée par une reconnaissance de compétences et de connaissances dans un champ d'expertise donné : l'autorité d'un spécialiste de neurochirurgie ne s'applique pas à la microbiologie des sols.

Puisque les autorités traditionnelles de la publication imprimée (éditeurs, diffuseurs, producteurs, etc.) sont remises en question par l'environnement numérique, il relève de l'éditorialisation d'éclairer ces mutations.

À ce propos, Évelyne Broudoux a publié son habilitation à diriger les recherches sur ce sujet (Broudoux 2022). Si de prime abord, elle note que le numérique, et plus spécifiquement le Web, permet à des auteurs de s'autopublier « sans passer par l'assentiment d'un médiateur qui leur offrirait une légitimation reconnaissable sur le web » avec, par exemple, la publication sur des blogs ; Broudoux constate également l'existence d'autres modèles de publication dont la légitimation des contenus est réalisée *a posteriori*, lorsque le contenu est déjà publié, puis annoté par d'autres contributeurs, comme c'est par exemple le cas avec la plateforme Wikipédia (Broudoux 2022 : 78). Ce type de publication collective prend également d'autres formes, comme pour les annotations de textes savants avec l'application Hypothesis, ce que Broudoux considère en tant que « pratiques érudites d'écrilecture » (2022 : 113). Dans ce cas, les contributions par

annotations permettent soit d'amorcer une discussion sur le texte en vue d'une publication ultérieure dans une revue, soit d'ouvrir une discussion relevant de la relecture par les pairs (Sauret 2020). Il y aurait donc une redistribution collective et collaborative de l'autorité aux niveaux de l'ensemble des contributeurs, des plateformes qui permettent ce type de publication et des calculs qui y sont effectués (Broudoux 2022 : 185). Cette perspective de l'éditorialisation en tant que processus fini préserve ainsi une forme d'autorité autour de la légitimation des contenus.

D'un autre côté, la définition de l'éditorialisation en tant que processus ouvert envisage la problématique de l'autorité sous une autre perspective. Tout d'abord, le numérique et le non-numérique ne sont pas à distinguer : il s'agit du même espace dans lequel nous agissons et au sein duquel le numérique est omniprésent. Cela veut dire que l'autorité telle qu'elle était définie à l'époque pré-numérique ne s'applique plus aujourd'hui, elle a été réorganisée au prisme de cette configuration numérique et comporte désormais une autre signification (Vitali-Rosati 2016).

En effet, selon Vitali-Rosati, puisque l'autorité de l'espace numérique est « déterminée par l'éditorialisation » (Vitali-Rosati 2016 : § 69), sa réorganisation dépasse le simple transfert d'autorité d'un agent à un autre et concerne directement les éléments du ressort d'une autorité : l'originalité et la fiabilité des contenus. Ce ne sont plus ces éléments que l'autorité légitime, car la dimension processuelle de l'éditorialisation la rend caduque.

L'espace numérique, et plus particulièrement le Web, fonctionne de manière réticulaire. Un contenu ou une ressource est rendue accessible sur le Web en étant connectée à tout un ensemble d'autres ressources, par exemple par le biais d'hyperliens. En ce sens, ladite ressource n'est ni indépendante, ni entière, elle s'inscrit plutôt dans ce réseau d'informations : ce n'est plus un tout isolé que l'on peut adjoindre à un auteur ou un éditeur, car cette ressource dépend du réseau duquel elle fait partie pour exister (du moteur de recherche jusqu'à l'hyperlien). Cependant, ce réseau de ressources est un objet instable qui est en perpétuelle métamorphose, caractéristique qui fait de l'éditorialisation un processus ouvert. En ce sens, l'autorité conférée par l'éditorialisation est relative au maintien de l'ensemble des relations d'une ressource avec le reste de son réseau, car sans cela, elle risque de disparaître. L'autorité ne légitime donc plus un objet, mais un processus (Vitali-Rosati 2016 ; 2018*a* : 78).

Ces deux points de vue sur l'éditorialisation nous montrent que ce processus a un rapport avec la phase de publication de l'écriture numérique (telle que nous l'envisageons dans la structure de cette thèse), mais également avec sa phase de production que nous avons abordé dans les chapitres précédents.

L'éditorialisation est donc le nom qui cristallise cette idée selon laquelle un ensemble de dynamiques, impliquant le numérique, produit du sens : une publication scientifique ne commence pas au moment de l'écriture, elle émerge de discussions, des projets de recherche et des résultats que l'on obtient, du comment on les obtient, surtout lorsqu'ils impliquent des moulinettes numériques [6] pour les produire (Bord 2025). Ces dynamiques sont des intrications de médiations entre elles, des coupures agentielles qui impliquent des médiations techniques, mais aussi d'autres formes d'interaction, comme la lecture. En cela, il est impossible d'établir un point initial à l'éditorialisation.

Le numérique, comme le montrent Pédauque, Broudoux et Vitali-Rosati, reconfigure nos manières d'écrire et de publier. Ce qu'ajoute Vitali-Rosati est que cette reconfiguration concerne non seulement les modalités d'écriture, mais surtout les modes de pensées. Le numérique n'entre pas en jeu lorsque l'on frappe sur une touche du clavier, mais dès lors que nos modes de vie reposent sur le numérique en tant qu'infrastructure : l'accès à l'information a changé, leur mode d'indexation également, etc. Tous ces changements sont apportés par ce medium numérique – ce qui réactualise la célèbre assertion de McLuhan *the medium is the message*) – et participe de cette raison computationnelle décrite par Bachimont (2000). Lorsque l'on veut écrire un texte, la question du choix de l'environnement d'écriture, que ce soit Microsoft Word, LibreOffice ou Stylo, n'est pas qu'une question de libre ou de propriétaire, elle insinue que ces environnements, en tant que médiations dans la production d'une pensée, jouent un rôle majeur.

Les formats XML, HTML, DOCX, ODT, Markdown, YAML, BibTEX, etc ; les protocoles de communication que ce soit par courriel ou via le Web, et toutes autres médiations techniques, définissent non seulement la théorie du texte rédigé dans l'environnement d'écriture, mais définissent surtout une coupure agentielle, matérielle, de la théorie de l'éditorialisation en train de se dérouler.

Dès que le choix d'un environnement d'écriture est réalisé, il opère une coupure agentielle dans le processus de production de l'objet et génère par cette action de nouvelles médiations qui inscrivent l'objet produit dans un réseau de relations avec d'autres objets.

En nous référant à l'éditeur de texte Stylo, étudié au chapitre IV, nous avons pu constater que, sur l'ensemble des formes de texte requises pour produire un document, trois états sur les quatre sont écrits par l'environnement d'écriture, tandis qu'un seul l'est par l'auteur. Ces trois états ne sont pas uniquement nécessaires à l'édition d'un document, ils inscrivent déjà le texte dans un réseau

6. Une moulinette est un outil *ad hoc* que l'on met en place pour répondre à une tâche, un traitement de données particulier. Il ne s'agit presque jamais d'outils mis à disposition en ligne, et encore moins d'outils réellement stables. Ces outils moulinent et brassent des données, parfois d'une façon qui n'est pas optimisée, mais qui fonctionne quand même.

d'objets dynamiques différent de celui dans lequel le texte pourrait être inscrit s'il était rédigé dans MSWord ou dans LibreOffice.

C'est en cela que l'éditorialisation produit la matérialité de l'écriture. Sans cette différence entre les environnements d'écriture, il n'y aurait que peu d'intérêt à créer autant d'environnements d'écriture différents (plusieurs centaines dès les années 1980) au-delà de l'esthétique de l'interface affichée à l'écran. Ainsi, le choix d'un environnement d'écriture et du modèle textuel qu'il incarne devient primordial du fait de ces relations qu'il crée avec d'autres objets et des réseaux d'objets auxquels il accède.

Il est toutefois important de souligner que le processus d'éditorialisation ne se limite pas à la seule production d'un texte. Dans le cadre de sa publication, notamment sur le Web, il est impératif que ce texte subisse une ou plusieurs transformations pour devenir publiable, c'est-à-dire qu'il soit mis en lien avec un réseau d'objets et d'informations. Ces transformations ont pour objectif d'inscrire le texte dans un processus de médiation et de l'intégrer à un réseau d'objets partiellement contrôlés. Cela implique la sélection de certains éléments ou objets avec lesquels le texte sera mis en relation. D'un point de vue technique, ces transformations passent par des conversions d'un format à un autre et recourent ainsi à d'autres modèles textuels pour y parvenir.

Afin d'observer cette inscription d'une production écrite dans un réseau d'objets, il apparaît ainsi nécessaire de réduire notre approche de ces transformations, et par extension de l'éditorialisation, à l'observation des modèles textuels qui y sont mobilisés. Cette démarche nous permet d'apporter des éléments de comparaison avec ceux étudiés durant la phase de production de l'écriture dans les chapitres précédents.

Dans un souci méthodologique, nous poursuivons les études de cas présentées dans les chapitres III et IV. Ainsi, la phase de publication de l'écriture de ces textes ne constitue pas le point de départ de l'éditorialisation. Ce processus débute avant même la phase de production de l'écriture. Les analyses précédentes durant la phase de production de ces textes serviront de point de départ pour la compréhension des transformations opérées sur ces derniers. En conséquence, ces transformations sont également abordées depuis les modèles textuels qu'elles incarnent et les médiations techniques qui permettent de passer du modèle utilisé durant la phase de production à celui nécessaire à la phase de publication.

V.3 Les transformations dans *Lodel*

La première chaîne de transformation étudiée est *Lodel*, un système de gestion de contenu (*Content Management System* (CMS)) conçu à l'origine pour offrir

une solution de numérisation aux revues existantes au format papier avant d'être élargi aux contenus nativement numérique ainsi qu'à d'autres types de contenus, tels que des informations de colloques, des livres, des blogs, etc. Cette chaîne a été employée par la revue *Humanités numériques* pour publier leurs articles sur la plateforme OpenEdition Journals.

Lodel est un projet créé en l'an 2000, lorsque OpenEdition, encore sous l'appellation Revues.org[7] commence les développements en *open source* de ce CMS « spécialisé dans l'édition de textes longs et complexes dans un environnement éditorial très structuré »[8] afin de répondre aux enjeux de publication et de diffusion numérique des textes savants.

Au-delà de l'administration et de la gestion d'un site web, la particularité de Lodel est de fournir un environnement comportant un modèle textuel adapté aux besoins de l'écriture scientifique comportant : « les notes de bas de page, la structure du texte, les divers jeux de caractères correspondant à des langues non latines, les signes diacritiques, les petites capitales, les espaces insécables, etc »[9].

L'intérêt de créer une telle plateforme est donc justifié par la nécessité d'employer une modélisation du texte pour l'écriture scientifique diffusée sur le Web qui, jusqu'alors, n'existait pas en France à cette échelle. En prenant ceci en compte, l'avantage principal d'un tel outil pour OpenEdition nous semble assez clair : offrir aux acteurs de la recherche une plateforme à l'intérieur de laquelle les documents respectent les conventions de l'écriture scientifique.

Information

OpenEdition est un portail de ressources électroniques en sciences humaines et sociales créé par le Centre pour l'édition électronique ouverte (Cléo) en 2011. Ce portail comprend quatre plateformes : OpenEdition Journals (anciennement Revues.org), Calenda, Hypotheses.org et OpenEdition Books. OpenEdition est une plateforme de diffusion qui agrège, en 2025, plus de 15000 livres, plus de 650 revues, un peu plus de 5000 carnets et près de 60000 événements. Dans le cadre de cette thèse, nous nous intéressons à la plateforme OpenEdition Journals.

Avoir son propre modèle, maîtrisé du début à la fin de la chaîne de transformation, offre également un autre avantage, plus technique cette fois.

7. Voir la page d'accueil de la plateforme sur *Internet archive* : `https://web.archive.org/web/20000816033833/http://www.revues.org/`, consultée le 26 mai 2025.

8. Voir la page d'accueil du site web de Lodel : `https://lodel.hypotheses.org/`, consultée le 27 mai 2025.

9. Voir également la page d'accueil du site web de Lodel : `https://lodel.hypotheses.org/`, consultée le 27 mai 2025.

Puisque OpenEdition manipule un ensemble conséquent de données, il est dans son intérêt d'indexer les informations selon une modélisation qui lui est propre et d'employer une architecture des documents cohérente à tous les niveaux, de l'échelle la plus grande, celle de la plateforme dans son intégralité, à une gestion fine des informations contenues dans les documents.

L'apparition des plateformes de diffusion est relativement récente dans l'histoire de la publication savante et des revues. Lorsque le Web devient plus accessible dans les années 1990, les premières plateformes de diffusion apparaissent durant cette décennie sur plusieurs continents : il y a la plateforme SciELO en Amérique latine (1998) qui promeut un accès ouvert à la recherche et aux publications scientifiques[10] ; mais aussi la plateforme Érudit au Québec, qui a été mise en ligne 1998 également[11] et défend rapidement une politique du libre accès ; ou encore la plateforme JSTOR lancée en 1995 aux États-Unis[12].

En France, deux plateformes de diffusion voient le jour entre la fin des années 1990 et le début des années 2000. Il y a tout d'abord Revues.org (qui devient OpenEdition quelques années plus tard), dont les contenus sont gérés avec le CMS Lodel. Ensuite, en 2003, un partenariat entre l'Université Lumière Lyon 2, la Maison de l'Orient et de la Méditerranée Jean Pouilloux de Lyon et de l'université Nice Sophia Antipolis est à l'origine de la plateforme de diffusion Persée, qui a été mise en ligne en 2005[13]. Ces deux plateformes de diffusion divergent cependant dans leurs objectifs. D'un côté, OpenEdition se donne comme objectif de fournir une chaîne éditoriale numérique complète et fournit ainsi plus qu'un service de diffusion centralisée. Elle offre, notamment via Lodel, un outil de gestion des contenus que ce soit des articles de revue, des livres ou des événements. Tandis que la plateforme Persée est orientée sur la numérisation, l'enrichissement et la diffusion de collections de documents déjà existant. Ces plateformes ne revêtent donc pas les mêmes enjeux, ni les mêmes besoins en termes d'infrastructure numérique.

Au-delà de leurs missions respectives, toutes ces plateformes de diffusion perturbent la division des tâches telle qu'elle était construite dans l'édition avant leur arrivée, souligne Mounier.

10. Voir la page web de SciELO : `https://www.scielo.org/en/about-scielo/program-publication-model-and-scielo-network/`, consultée le 27 mai 2025.

11. Voir l'historique d'Érudit en ligne : `https://apropos.erudit.org/ligne-du-temps/`, consulté le 27 mai 2025.

12. Voir leur site web : `https://about.jstor.org/mission-history/`, consulté le 27 mai 2025.

13. Voir le site web de Persée : `https://info.persee.fr/historique/`, consulté le 10 juin 2025.

« Elle est à la fois le distributeur, le diffuseur et le libraire et joue également le rôle d'imprimeur dans certains cas, lorsque la plateforme génère ses propres formats de fichiers (Mounier 2020 : § 5). »

Initialement, ce système de publication permet de rendre accessible et visible la production des connaissances, afin que toute la communauté érudite puisse en bénéficier[14]. Cette mise en accessibilité ne se joue pas sur la face visible à l'écran, c'est-à-dire où l'information se trouve à travers un portail, une plateforme, mais elle se joue dans la transformation de l'article en un ensemble de données (Sauret 2020 : 152), car le numérique permet de reconfigurer ces informations sous forme de fragments, ce qui permet de les indexer plus finement et de modifier notre rapport à celles-ci (Sauret 2020 : 105).

Afin que les articles restent accessibles, il devient nécessaire de les inscrire dans un réseau d'objets bien référencés, ce à quoi tente de répondre Lodel qui est principalement employé pour publier des articles sur les plateformes d'OpenEdition.

Au début des années 2000, le concept à la base de cette chaîne Lodel est fondé sur le postulat selon lequel les auteurs ont recours à un logiciel de traitement de texte pour rédiger leurs documents, qu'il s'agisse d'articles ou de chapitres d'ouvrages. Une fois les textes rédigés, relus, corrigés et enfin prêts à être publiés, ils sont transformés en fichier XML TEI pour être importés dans le CMS Lodel[15].

Les différentes manipulations se font avec le format XML, car il est le format pivot de Lodel. Lodel emploie la technologie XML avec son propre schéma issu de la TEI[16] pour structurer les contenus des documents. Un deuxième schéma, que nous n'abordons pas, est également employé pour l'import et la gestion des collections de documents via leurs métadonnées (livres, numéros de revue, etc.) dans les plateformes OpenEdition Books et Journals : il s'agit du schéma METS. Sans entrer dans les détails de ces schémas, leur configuration montre déjà la structure employée pour indexer et organiser un maximum d'informations sur la plateforme OpenEdition.

En ce qui concerne l'indexation des contenus à l'intérieur du CMS Lodel, ceux-ci le sont dans une base de données MySQL. Les métadonnées des documents, qui contiennent des informations descriptives sur le contenu, sont intégrées dans cette base de données sous forme d'entrées distinctes. En revanche, le texte intégral des articles est stocké dans une seule entrée texte, qui contient le texte

14. Nous ne prenons pas en considération les modèles économiques et les politiques d'accès aux publications, car d'autres travaux traitent déjà de cette question.

15. Aujourd'hui, il existe deux outils pour effectuer cette transformation de DOCX vers XML : soit en utilisant un module développé par le Cléo, nommé OTX, que l'on intègre à Lodel, soit via un autre ensemble de *méthodes et outils pour l'édition structurée* (Métopes).

16. Voir le répertoire public sur la plateforme GitHub : `https://github.com/OpenEdition/tei.openedition`, consultée le 29 mai 2025.

en langage XML. Le texte en XML peut ainsi être transformé en *Extensible HyperText Markup Language* (XHTML) pour l'affichage dans une page web ou utilisé pour parser les contenus et retrouver les informations textuelles qui y sont indexées.

La transformation de l'article de la revue *Humanités numériques* en XML a été réalisée au moyen du module d'export de Stylo. Ce premier document a ensuite été validé par les outils de la chaîne Métopes avant son intégration dans Lodel. Cette chaîne de traitements étant expérimentale, le comité scientifique de la revue précise que quelques modifications du fichier XML ont été apportées à la main pour finaliser le document avant son import dans Lodel (Bardiot et al. 2023). Ces corrections de dernière minute sont notamment dues au fait que le schéma employé par le module d'export de Stylo pour produire le document XML est le schéma *Commons Publishing* et non le schéma *OpenEdition TEI* mentionné précédemment.

Dans le chapitre précédent, nous avons étudié les médiations techniques nécessaires à la production d'un document dans Stylo. Toutefois, ces médiations ne permettent pas à elles seules de produire un document lisibles en dehors de cet environnement. Afin que cela soit le cas, il est nécessaire d'employer le module d'export de Stylo (au moins pour exporter les sources dans les formats de texte brut). Ce module est un objet séparé de l'application Stylo[17], conformément à la structure de Stylo présentée dans le chapitre précédent. L'objectif de dissocier ces deux modules est de rendre les deux parties indépendantes l'une de l'autre, notamment pour la maintenance du code source de ces outils.

Ce module d'export[18], programmé avec le langage Python, est dépendant d'une autre brique logicielle pour fonctionner : le convertisseur Pandoc, créé en 2006 par John MacFarlane. Afin d'éviter de mélanger toutes les médiations techniques au risque de complexifier notre discours, nous abordons les transformations réalisées par Pandoc dans la section suivante de ce chapitre. Pour l'instant gardons à l'esprit que Pandoc est nécessaire pour transformer les sources au format Markdown, YAML et BibTeX en un seul fichier au format XML TEI[19].

Ce fichier TEI n'est que temporaire. L'utilisateur qui émet la requête d'export ne voit pas ce document temporaire, puisqu'il est immédiatement transformé par le module d'export de Stylo en un document XML conforme au schéma XML *Commons Publishing* fourni par l'infrastructure de recherche Métopes, grâce

17. Le code source du module d'export de Stylo est hébergé sur l'instance gitlab hébergée par la TGIR Huma-num : `https://gitlab.huma-num.fr/ecrinum/stylo/stylo-export`, consultée le 29 mai 2025.

18. Voir le schéma de Stylo dans la figure IV.1.

19. La définition de la fonction du module d'export qui appelle Pandoc se trouve ici : `https://gitlab.huma-num.fr/ecrinum/stylo/stylo-export/-/blob/main/styloexport/pandocapi.py?ref_type=heads#L175`, page consultée le 29 mai 2025.

à une feuille de transformations écrite en XSL. La récupération d'une archive au format ZIP contenant le fichier transformé par l'utilisateur marque la fin de l'export du document de l'environnement Stylo.

Le schéma *Commons Publishing* utilisé pour réaliser cette transformation a été élaboré par plusieurs infrastructures dont Métopes, OpenEdition, le Centic dans le cadre du projet I-FAIR-IR, et financé par le Fonds National pour la Science Ouverte (FNSO). Ce schéma doit notamment être employé pour la nouvelle version 2.0 de Lodel en cours de développement. Lodel 2.0 a été déployé en 2024 pour la plateforme OpenEdition Book, mais ne l'a pas été pour OpenEdition Journals (Team 2024). En conséquence, le document produit par Stylo est rétrogradé par les outils Métopes en une version valide pour Lodel 1, et cela jusqu'à ce que Lodel 2 soit déployé sur la plateforme OpenEdition Journals.

Afin d'aboutir à la forme documentaire qui puisse être importé par Lodel, et transformé en page web XHTML sur OpenEdition Journals, l'article de la revue *Humanités numériques* doit donc faire l'objet de deux transformations et passer à travers trois formes différentes à l'intérieur du module d'export de Stylo. La première forme, que nous n'étudions pas, est une forme temporaire créé par Pandoc : il s'agit de son arbre de la syntaxe abstraite (*Abstract Syntax Tree* (AST))[20]. Cette première forme ne sort pas du giron de ce logiciel. Ensuite, le document exporté doit à nouveau être transformé pour être compatible avec la version 1 de Lodel.

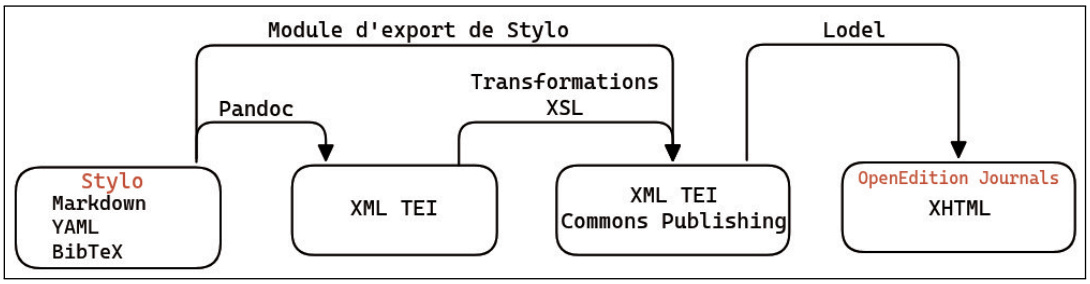


Figure V.1 : Les transformations de Stylo vers Lodel

La première transformation permet donc de convertir les sources produites dans Stylo, le Markdown, le YAML et le BibTEX en XML TEI via le convertisseur Pandoc. Sans compter le modèle textuel implémenté dans Pandoc, la première modélisation que nous étudions est celle de ce fichier en TEI.

Un fichier en XML TEI est construit de la même façon que la plupart des fichiers au format XML. Il est scindé en trois parties ; la première définie le *namespace* dans lequel se situe le document, que l'on traduit par espace de nom,

20. Nous voyons l'*Abstract Syntax Tree* (AST) de Pandoc plus en détail dans la section V.4. Un AST est une structure arborescente représentant la structure sémantique d'un document.

la deuxième est ce que l'on appelle un *teiHeader*, un en-tête de fichier, et la troisième contient le corps du fichier dans une balise *text*.

La particularité de cette première transformation provient du fait que les contenus jusqu'alors distingués par trois formats différents (Markdown, YAML et BibTEX) vont être réunis en un seul fichier.

En premier lieu, nous observons l'espace de nom de notre nouveau document. L'espace de nom est un élément très important dans un document XML, car il lève toute ambiguïté sur la signification des balises que le document contient, voire peut apporter des précisions supplémentaires lorsqu'un schéma est déclaré en attribut de cet espace de nom.

Dans le document produit par le module d'export de Stylo, l'espace de nom est le suivant :

Extrait de code V.1 : Espace de nom de la TEI

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <TEI xmlns="http://www.tei-c.org/ns/1.0" xml:lang="fr">
```

À la deuxième ligne se trouve l'élément racine `<TEI>` du document, avec pour espace de nom XML (*xmlns* pour *XML namespace*) celui de la TEI dont la valeur qui lui est affectée est l'URL du site web du consortium de la TEI. Ce lien a une signification très intéressante, car le document qui se situe à cette adresse fait autorité sur notre document transformé : c'est lui qui définit le sens de chacune des balises employées dans le document transformé avec le module d'export de Stylo.

Le site web de la TEI contient également toutes les directives qui définissent les significations des balises employées. Nous nous appuyons sur ces directives pour étudier le glissement sémantique entre les sources issues de Stylo et cette première transformation en TEI. À l'intérieur de cette balise, nous remarquons également l'absence d'un schéma, qui apparaîtrait avec l'attribut *change*. L'absence d'un schéma signifie qu'il n'est pas possible de vérifier qu'un document est bien valide, mais n'empêche pas de vérifier que le document soit bien formé et respecte la syntaxe de la TEI.

Ensuite, la deuxième partie du fichier, l'en-tête, contient toutes les métadonnées du document. Cette partie est prévue pour indexer tous les descripteurs d'un document, ses auteurs, la date de publication, le titre, etc. La particularité de cet en-tête *teiHeader* produit avec l'export de Stylo se situe dans l'usage de la balise `<xenodata>`, puisqu'elle agit en tant que conteneur pour y insérer des métadonnées ne faisant pas partie de la TEI[21]. Ce conteneur est employé pour transposer les métadonnées sérialisées en YAML dans l'éditeur de texte Stylo

21. Voir les directives de la TEI : `https://www.tei-c.org/release/doc/tei-p5-doc/en/html/ref-xenoData.html`, page consultée le 30 mai 2025.

directement dans le fichier TEI, en remplaçant les clés YAML par des balises XML.

Voici un extrait du YAML issu de Stylo :

Extrait de code V.2 : Métadonnées en YAML dans Stylo

```
1  authors:
2  - affiliations: >-
3      Membre de la Société Royale du Canada Chercheur
          associé à l'Université de
4      Montréal (CRC sur les écritures numériques et Centre
          de Recherche
5      Interuniversitaire sur les Humanités Numériques).
          Montréal, Canada.
6    email: pierre.levy@mac.com
7    forename: Pierre
8    orcid: 0000-0003-1599-221X
9    surname: Lévy
```

En voici un extrait du contenu de la balise `<xenodata>` dont le total fait 146 lignes :

Extrait de code V.3 : Métadonnées dans la balise xenodata de la TEI

```
1  <xenoData>
2  <stylo-metadata xmlns="http://stylo.huma-num.fr/ns/1.0">
3  <title_f>Calculer la sémantique avec IEML</title_f>
4  <title>Calculer la sémantique avec IEML</title>
5  <subtitle_f></subtitle_f>
6  <author>
7    <affiliations>Membre de la Société Royale du Canada
       Chercheur associé
8  à 'lUniversité de Montréal (CRC sur les écritures numériques
     et Centre
9  de Recherche Interuniversitaire sur les Humanités
     Numériques). Montréal,
10 Canada.</affiliations>
11   <email>pierre.levy@mac.com</email>
12   <forname>Pierre</forname>
13   <orcid>0000-0003-1599-221X</orcid>
14   <surname>Lévy</surname>
15  </author>
```

Cet élément `<xenodata>` est très intéressant parce qu'il permet d'insérer dans le modèle textuel de la TEI des éléments qui lui sont extérieurs et, dans ce cas-ci, qui ne sont définis nulle part ailleurs : ces métadonnées n'ont pas de signification en dehors de l'environnement Stylo, elles lui sont contingentées, car c'est une structure propre à cet environnement et à la modélisation des métadonnées jugées nécessaires à l'écriture en sciences humaines et sociales par l'équipe qui a développé Stylo. Autrement dit, cette première phase de transformation combine deux modèles textuels différents : celui de Stylo, et celui de la TEI.

Enfin, la dernière partie `<text>` « contient un seul texte quelconque, simple ou composite, par exemple un poème ou une pièce de théâtre, un recueil d'essais, un roman, un dictionnaire ou un échantillon de corpus »[22]. Ramenée à notre article, cette partie contient tout le Markdown ainsi que les références bibliographiques. Il est assez intéressant que le corps du texte soit inscrit dans cette balise, et pas dans une balise `<body>`, comme c'est le cas en HTML. Nous faisons cette comparaison parce que le format Markdown a été pensé à partir du format HTML et de son modèle textuel. Cependant, en TEI, le `<body>`, est un sous-élément structurel de `<text>` que l'on retrouve au même niveau que d'autres éléments tels que `<front>`, `<back>` et `<group>`. Rappelons-nous, la TEI signifie *Text Encoding Initiative*. Cette modélisation est pensée pour indexer, encoder et décrire du texte, d'où le fait que le corps du texte ne soit pas dans une balise `<body>`. Les sous-éléments structurels rappellent quant à eux la structure d'un document imprimé ou manuscrit, avec des contenus préliminaires et des contenus liminaires, comme des pages de titre ou des appendices.

Cette modélisation n'existe pas en Markdown, ni dans Stylo. Par exemple, les éléments pris en charge par la balise `<front>` en TEI se retrouvent dans les métadonnées en YAML dans Stylo, puis dans l'élément `<xenodata>` en en-tête de fichier et non pas dans cet élément structurel `<front>`.

Dans le document, nous voyons bien que nous passons directement de l'en-tête au `<body>`, l'élément `<text>` n'étant là presque que par obligation puisqu'il n'y a ni `<front>`, ni `<group>`, ni `<back>`. L'élément `<text>` est présent car sans lui le document ne serait pas bien formé.

Extrait de code V.4 : Structure TEI après une première transformation par Pandoc

```
1 </teiHeader>
2 <text>
3   <body>
4     <div type="level1" xml:id="introduction">
```

Cette forme d'écriture intermédiaire ne signifie pas que le document serait « mal » structuré. Au contraire, il montre qu'à cette étape de la transformation, le modèle du texte change et se trouve à mi-chemin entre deux modèles : une structure plutôt pensée pour une écriture en environnement web, et une structure pensée pour le texte, voire le texte imprimé.

Un dernier élément attire notre attention dans ce document, il s'agit des références bibliographiques contenues dans le document. Initialement, ces références sont structurées selon le format BibTeX.

22. Pour plus de détails sur cette balise `<text>`, voir la page web des directives à ce sujet'`https://www.tei-c.org/release/doc/tei-p5-doc/fr/html/ref-text.html`, consultée le 31 mai 2025.

BibTEX est un format créé en 1985 par Leslie Lamport et Oren Patashnik pour la gestion des références bibliographiques dans l'environnement LATEX (créé au début des années 1980 par Leslie Lamport sur la base du travail de Donald Knuth avec TEX). Ce format permet de décrire, de stocker et de trier des références bibliographiques dans des fichiers en texte brut. Grâce à celui-ci, il est possible de créer des bases de données de références bibliographiques. Avec BibTEX, il est possible d'indexer dix-sept types de documents différents (article de revue, livre, brevet, billet de blog, etc.), ce qui couvre une large partie des besoins de citation. Les informations de chacune des ressources bibliographiques sont rédigées selon le même principe qu'un dictionnaire : `clef= "valeur"`.

La structure des données en BibTEX ressemble à ceci :

Extrait de code V.5 : Référence bibliographique en BibTEX dans Stylo

```
@book{austin_how_1962,
  address = {Oxford},
  title = {How to {Do} {Things} {With} {Words}},
  publisher = {Oxford University Press},
  author = {Austin, Langshaw, John},
  year = {1962},
}
```

Dans le document en TEI, ces références sont transformées pour intégrer le `<body>` en fin de document pour former la bibliographie. La transformation donne comme résultat ceci :

Extrait de code V.6 : Référence bibliographique transformée en XML TEI

```
  <p xml:id="ref-austin_how_1962">Austin, J., Langshaw.
     (1962).  <hi
  rendition="simple:italic">How to Do Things With
     Words</hi>. Oxford
  University Press.</p>
```

Lors de cette transformation, le phénomène que l'on observe est une mise à plat des références bibliographiques. C'est-à-dire que l'on passe d'éléments textuels structurés à un paragraphe sans aucune structure, en dehors d'un rendu visuel en italique pour le titre du livre, comme on l'afficherait dans une page ou sur un écran. On observe alors une perte sémantique relativement forte : d'une part, l'intérêt d'être dans un environnement numérique disparaît, car cet objet devient mal indexé ; d'autre part, en raison de cette mauvaise indexation, cet objet est inscrit en dur directement dans le texte de l'article, sans pouvoir être rattaché à d'autres objets ou inscrit dans un réseau d'informations plus vaste.

Cette perte sémantique n'est pas un problème de modélisation de la TEI qui prévoit des éléments pour les références bibliographiques, par exemple l'élément

`<bibl>`[23]. Malgré le fait que cet objet soit décrit par les directives de la TEI comme « faiblement structuré », il a *a minima* la capacité de structurer les informations de la référence bibliographique ci-dessus.

La mise à plat des informations bibliographiques structurées provient du convertisseur Pandoc, couplé à l'utilisation du module de Citeproc[24] utilisé pour générer des citations et des références bibliographiques suivant des styles décrits selon le format *Citation Style Language* (CSL) désiré. À ce jour, les choix techniques de structuration du document en TEI dans ces deux outils réalisés par MacFarlane et la communauté de Pandoc ne permettent pas de préserver des données bibliographiques structurées. Ce problème a été reporté à leur attention à plusieurs reprises, notamment dans des tickets[25] sur la plateforme Github où le code source de ces outils est hébergé.

Le point important de cette transformation des données bibliographiques est que Pandoc impose sa vision du document à cet endroit, qui n'est ni celle de Stylo, ni celle de la TEI. C'est-à-dire qu'il ne s'agit pas simplement d'une conversion, ou d'une correspondance approximative entre deux modèles signifiant sensiblement la même chose ; il y a un changement profond du sens de ces informations, avec d'une part la mise en plat des données sans structure et d'autre part la suppression de certaines données présentes dans le BibTeX qui n'apparaissent plus dans le style bibliographique choisi.

Cette première transformation nous amène à l'obtention d'un document intermédiaire en XML TEI. Néanmoins, celui-ci ne peut pas être importé en l'état dans Lodel, car il ne respecte pas le schéma XML développé par OpenEdition : le document en l'état n'est donc pas valide. Pour remédier à cela, il est prévu dans le module d'export de Stylo que ce document en TEI soit à nouveau transformé. La transformation n'est cependant plus réalisée par Pandoc, mais par une feuille de transformation XSL[26], comme cela figure sur le schéma V.1.

Cette feuille de transformation, comme son nom l'indique, permet de sélectionner des nœuds dans un document, et de leur appliquer des modifications et donc de les transformer. Cette technique permet de circuler dans un document

23. Voir la page web à propos de l'élément `<bibl>` à l'url suivante : `https://tei-c.org/release/doc/tei-p5-doc/fr/html/ref-bibl.html`, consultée le 31 mai 2025.

24. Le répertoire Github de Citeproc se trouve à cette URL : `https://github.com/jgm/citeproc/tree/master`, consulté le 02 juin 2025.

25. Voir par exemple un ticket sur le dépôt de Citeproc : `https://github.com/jgm/citeproc/issues/132`. Ou encore un ticket sur le dépôt de Pandoc : `https://github.com/jgm/pandoc/issues/8790` ; pages consultées le 02 juin 2025.

26. Cette feuille de transformation est accessible depuis cette URL : `https://gitlab.huma-num.fr/ecrinum/stylo/stylo-export/-/blob/main/templates/generique/template-metopes.xsl?ref_type=heads`, consultée le 02 juin 2025. Elle a été réalisée par Yves Marcoux, professeur honoraire à l'Université de Montréal et spécialiste des technologies XML.

structuré sur la base des chemins (*paths)* de chaque nœud et de les modifier selon les besoins, par exemple pour leur ajouter des attributs ou pour changer l'intitulé du nœud afin qu'il corresponde à un schéma donné. Afin de préserver une certaine cohérence dans notre propos, nous nous appuyons principalement sur les éléments post-transformés, et nous faisons référence à la ligne concernée par ladite transformation dans la feuille de style XSL en note de bas de page.

La première transformation [27] que l'on remarque dans le document est réalisée sur la racine de celui-ci :

Extrait de code V.7 : Espace de nom de Commons publishing

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <TEI xmlns="http://www.tei-c.org/ns/1.0"
3       change="commons_edition"
4       xml:lang="fr">
```

Si l'espace de nom est toujours celui de la TEI, il est tout de même restreint à une version réduite qui est celle du schéma Commons Publishing évoqué précédemment. Ceci est rendu lisible grâce à l'ajout de l'attribut *change* à la racine et à sa valeur *common_edition* qui fait référence à ce schéma [28]. L'autorité principale derrière ce document reste bien la TEI à ceci près qu'elle ne peut pas être utilisée dans sa totalité et qu'une nouvelle autorité s'insère dans ce processus d'éditorialisation : il s'agit du document de Commons Publishing.

La structure des métadonnées a changé en conséquence, puisqu'elles ne peuvent plus être contenues dans un élément `<xenodata>` que Lodel ne saurait parser et traiter. Elles ont été complètement transformées afin de respecter le schéma de Lodel, avec une partie des métadonnées dans l'en-tête du document `<teiHeader>` et une autre partie dans la sous-structure `<front>` du corps de texte du document.

Par exemple, pour l'auteur de l'article, nous retrouvons certains éléments dans l'en-tête du document [29] :

Extrait de code V.8 : En-tête du document TEI Commons publishing

```
1  <teiHeader>
2       <fileDesc>
3          <titleStmt>
```

27. La transformation de la racine du document est réalisée par ce bloc de code : https://gitlab.huma-num.fr/ecrinum/stylo/stylo-export/-/blob/main/templates/generique/template-metopes.xsl?ref_type=heads#L58, consultée le 02 juin 2025.

28. Voir l'explication du schéma Commons Publishing en version HTML à l'adresse suivante : https://tei-commons-publishing.readthedocs.io/fr/latest/odd_doc.html#TEI.TEI, consultée le 02 juin 2025.

29. Cette transformation est réalisée par ce bloc de code : https://gitlab.huma-num.fr/ecrinum/stylo/stylo-export/-/blob/main/templates/generique/template-metopes.xsl?ref_type=heads#L228, page consulté le 02 juin 2025.

```
4              <title type="main">Calculer la sémantique avec
                  IEML</title>
5              <author role="aut">
6                 <persName>
7                    <forename>Pierre</forename>
8                    <surname>Lévy</surname>
9                 </persName>
10                <affiliation>
11                   <orgName>Membre de la Société Royale du
                        Canada Chercheur associé
12 à l'Université de Montréal (CRC sur les écritures numériques
      et Centre
13 de Recherche Interuniversitaire sur les Humanités
      Numériques). Montréal,
14 Canada.</orgName>
15                </affiliation>
16                <email>pierre.levy@mac.com</email>
17                <idno type="ORCID">0000-0003-1599-221X</idno>
18             </author>
```

D'autres éléments des métadonnées apparaissent également dans le `<front>` du document, car elles sont amenées à apparaître dans la page web produite par Lodel. En voici un échantillon :

Extrait de code V.9 : Échantillon du front du document TEI Commons publishing

```
1  <front>
2           <div type="titlePage">
3              <p rend="title-main">Calculer la sémantique avec
                  IEML</p>
4              <p rend="author-aut">Pierre Lévy</p>
5              <p rend="authority_affiliation">Membre de la
                  Société Royale du Canada Chercheur associé
6  à l'Université de Montréal (CRC sur les écritures numériques
      et Centre
7  de Recherche Interuniversitaire sur les Humanités
      Numériques). Montréal,
8  Canada.</p>
9              <p rend="authority_mail">pierre.levy@mac.com</p>
10          </div>
```

Une distinction fondamentale réside entre les métadonnées de l'en-tête et celles intégrées au corps du texte. En effet, celles dans le corps de texte sont insérées dans une structure plus légère et composée de paragraphes, tandis que celles de l'en-tête sont plus verbeuses et structurées plus en profondeur, car leur rôle diffère : tandis que l'enjeu du corps de texte réside principalement dans l'affichage des informations, celui de l'en-tête relève plutôt de la bonne indexation des données.

Une fois ces transformations terminées, lorsque l'on compare ces métadonnées à celles saisies dans Stylo vues dans le bloc de code V.2, puis celles transformées

en TEI, observées dans le bloc de code V.3 ; nous repérons immédiatement les changements de structure et la disparition du modèle textuel initialement utilisé durant la phase de production de l'écriture de l'article dans Stylo.

En effet, le modèle textuel utilisé durant la phase de publication de l'écriture, celui associé au schéma XML de Commons Publishing, supplante totalement celui de la phase de production dans Stylo après les deux phases de transformation. L'auteur de l'article, par exemple, n'est plus défini par un ensemble de `cle:valeur` sérialisées au format YAML, mais l'est par le nœud `<author>` [30] que l'on trouve dans l'en-tête du document transformé.

Enfin, nous retrouvons les références bibliographiques en fin de document. Nous les abordons rapidement à titre de comparaison, car il n'y a pas de changement de structure majeur après celui de la première transformation. L'élément paragraphe qui encapsule chaque référence après la première transformation (voir le bloc de code V.6) est simplement transformé en élément `<bibl>` sans que les références soient plus structurées que cela [31].

Ces transformations sont des étapes nécessaires à l'insertion d'un objet, l'article scientifique, dans un écosystème : la revue *Humanités numériques* hébergée et diffusée sur la plateforme OpenEdition par le biais du CMS Lodel. Elles permettent notamment à Lodel de générer le document au format XHTML publié [32] sur le site en ligne de la revue (voir la figure V.1). Il s'agit là d'une dernière transformation que Lodel effectue afin que le document puisse être interprété par le navigateur (et les robots qui circulent sur le Web). Dans ce dernier document, nous pouvons observer que Lodel ajoute un certain nombre d'informations dans le document XHTML pour justement le situer dans tout l'écosystème en ligne, dans la revue et la plateforme OpenEdition, puisque c'est ce document *publié* qui sert de référence, notamment pour le citer.

Par exemple, nous retrouvons des éléments tels que le nom de la revue, des identifiants comme l'*International Standard Serial Number* (ISSN) ou le *Digital Object Identifier* (DOI), implémentés dans les métadonnées du document et décrits selon le vocabulaire contrôlé *Dublin core* :

Extrait de code V.10 : Les métadonnées publiées dans l'article en ligne sur OpenEdition Journals

30. Voir la documentation de Commons Publishing à propos de l'auteur : `https://tei-commons-publishing.readthedocs.io/fr/latest/usage_oeb.html#auteurs-contributeurs`, consultée le 02 juin 2025.

31. Voir la document du schéma Commons Publishing : `https://tei-commons-publishing.readthedocs.io/fr/latest/usage_oeb.html#references-bibliographiques`, consultée le 02 juin 2025.

32. La source de l'article en XHTML est visible à cette URL : `view-source:https://journals.openedition.org/revuehn/3836`, consultée le 03 juin 2025.

```
1 <meta name="citation_journal_title" content="Humanités
     numériques" />
2 <meta name="DC.identifier" scheme="DOI"
     content="10.4000/revuehn.3836 />
3 <meta name="DC.identifier" scheme="ISSN" content="2736-2337"
     />
```

À travers ces différentes phases de transformation, le document produit dans Stylo mue et délaisse son modèle textuel pour celui de Lodel et d'OpenEdition. Ce processus, qui s'inscrit dans le cadre plus large de l'éditorialisation, ne saurait être réduit à un simple remplacement ou écrasement d'un modèle textuel par un nouveau. En effet, ces transformations ne sont rendues possibles que parce que les informations et les données sont déjà présentes dans Stylo. Sans cela, les transformations seraient caduques et généreraient un document vide.

L'éditorialisation observée à travers ces quelques médiations techniques ne reflète que la partie émergée de l'iceberg, là où les traces sont visibles dans les documents. En modifiant le modèle textuel sous-jacent à l'article, ces médiations permettent de créer une structure d'objets que Lodel va pouvoir indexer. L'indicissible de ce processus, pourtant nécessaire, relève de toutes les discussions et échanges qu'il y a pu avoir entre l'infrastructure de recherche Métopes, la CRCEN et les membres de la revue *Humanités numériques* (comité scientifique et éditrices), sur la mise en place de cette chaîne de transformation : des vidéo-conférences, des réunions en présence (à Montréal et à Paris) avec des supports de présentation projetés sur écran, des échanges par courriels, ou encore une demande de financement non aboutie, celle du projet Péristyles soumis au FNSO en 2023. Ce sont l'ensemble de ces médiations qui permettent d'inscrire l'article de la revue *Humanités numériques* dans cet écosystème sur la plateforme OpenEdition Journals.

V.4 Les transformations dans le Pressoir

La deuxième chaîne de transformation que nous étudions se nomme le Pressoir[33]. Il est intéressant de la mettre en perspective avec Lodel, car il s'agit d'une chaîne de transformation pensée pour produire des livres numériques. Ainsi, nous pouvons réaliser l'étude d'un second objet éditorial à la suite de l'article

33. Dans le contexte de notre étude, nous ne nous référerons pas à la dernière version du Pressoir, qui est plus aboutie, car ce n'est pas celle-ci qui a été employée pour fabriquer l'ouvrage *Contribution numérique: cultures et savoirs*, mais une version antérieure datant de la publication du livre. Les informations sur le livre et ses différentes versions sont publiées sur le site web des Ateliers de sens public `https://ateliers.sens-public.org/contribution-numerique/index.html`, consultée le 03 juin 2025. Nous pouvons retrouver la version utilisée pour construire l'ouvrage à cette adresse : `https://pypi.org/project/pressoir-cli/2.0.1/`, consultée le 03 juin 2025.

produit avec Lodel. En plus du type d'objet éditorial créé, cette chaîne se distingue également par le fait qu'elle est pensée pour être la plus générique possible et modulable selon les besoins à l'usage, contrairement à Lodel qui est très orienté pour la plateforme OpenEdition.

Information

Le Pressoir est un générateur de sites statiques qui peut être utilisé pour créer des livres augmentés, des supports de cours, des documentations et toutes sortes de publications numériques. C'est est un projet amorcé en 2018 par la CRCEN. Il a connu de nombreux développements jusqu'à devenir, en 2024, un librairie Python. La documentation du projet est disponible en ligne : `pressoir.org`.

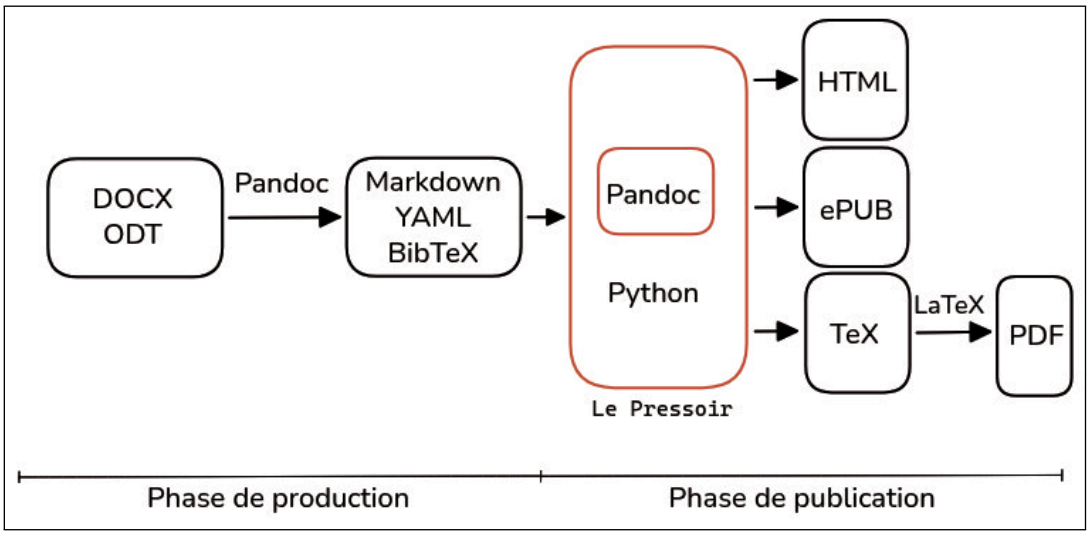


Figure V.2 : Les transformations avec le Pressoir

Le Pressoir est une chaîne de transformation un peu particulière, car, contrairement à Lodel qui ne permet de produire qu'un seul objet, elle permet de produire plusieurs artefacts à partir d'une même source, avec pour cible principale les livres web augmentés. C'est une chaîne de transformation qui respecte le principe de *single source publishing* (Fauchié et Audin 2023). Grâce à ce principe, le livre numérique est produit en plusieurs formats, chacun étant conçu pour s'insérer dans un environnement précis : deux versions HTML différentes, l'une pour être diffusée sur le Web, l'autre est au format ePUB ; une version PDF et une version imprimée. Chacune de ces versions répond à des enjeux d'écriture différents que le Pressoir prend en charge au moment de la transformation des sources, qui dès lors sont censées répondre aux enjeux posés par ces différents formats. Ainsi, comme le montre la figure V.2, le Pressoir est une réponse à la problématique

d'une chaîne éditoriale multimodale dans le domaine des technologies de l'édition numérique (J. Blanc et Haute 2018).

Histoire du Pressoir et de son modèle textuel

L'idée de produire une chaîne de transformation multimodale remonte à 2014, lorsque le premier ouvrage d'une nouvelle collection des Presses de l'Université de Montréal (PUM) est paru : *Pratiques de l'édition numérique*, sous la direction de Michael Sinatra et Marcello Vitali-Rosati (2014). Ce livre traite à la fois des humanités numériques et des défis éditoriaux liés à la production de contenus numériques. Il est disponible dans une version imprimée classique dans un format de poche, et dans quelques formats numériques : ePUB et PDF, mais aussi dans une version web et enrichie mise en ligne avec le CMS SPIP (Système de publication pour l'Internet). Le choix de SPIP est réalisé pour trois raisons principales : c'est un CMS léger, facile à utiliser et relativement flexible au début des années 2010 (créé en 2001), et parce que SPIP est issu d'une certaine culture numérique et a été utilisé et développé autant par des universitaires que par des journalistes. SPIP dispose de son propre langage de balisage pour l'écriture et présente des fonctionnalités similaires à celles du célèbre WordPress (Fauchié, Delannay et al. 2023).

Les éditeurs de la collection des PUM souhaitaient trouver un moyen de partager le contenu enrichi, et la collection a été rendue disponible dans un modèle hybride en libre accès : la version web (et enrichie) peut être lue gratuitement en ligne, les fichiers PDF et ePUB peuvent être téléchargés gratuitement, et la version imprimée peut être achetée. Ce livre a été l'opportunité de construire un nouveau processus de publication numérique avec pour problématique la construction d'une chaîne éditoriale pour une collection hybride comportant des artefacts au format divers.

Au début des années 2010, d'autres équipes de recherche s'intéressent à la question du livre augmenté. Plusieurs initiatives voient le jour telles que Scalar [34], un projet développé depuis 2013 par l'*Alliance for Networking Visual Culture* (ANVC) en association avec Vectors et IML, ou encore Manifold, un autre projet développé par les partenaires CUNY *Graduate Center's Digital Scholarship Lab*, les Presses de l'Université du Minnesota et *Cast Iron Coding* depuis 2012 [35].

Que ce soit SPIP, la première solution déployée en 2014 pour mettre en ligne l'ouvrage *Pratiques de l'édition numérique*, Manifold, Scalar ou bien d'autres outils de génération de livres augmentés, le problème principal restait le même,

34. Voir le répertoire GitHub du projet `https://github.com/anvc/scalar`, consulté le 03 juin 2025.

35. Voir le site web du projet : `https://manifoldapp.org/`, page consultée le 03 juin 2025.

car ces chaînes éditoriales ne permettaient pas de produire une pluralité d'objets à partir d'une source unique, et encore moins d'adapter les contenus au type de média produit (imprimé, web, etc.). Une autre limite importante de ces plateformes est que la modélisation du texte est très figée, car elle est implémentée dans une base de données relationnelle gérée par des CMS. En ce sens, s'approprier le modèle textuel afin de le personnaliser est très difficile. Pour ces raisons, la CRCEN s'est tournée vers d'autres solutions plus légères avec un fort intérêt pour le texte brut et les langages de balisages légers. C'est ainsi que les développements du Pressoir ont commencé (Fauchié, Delannay et al. 2023).

Le choix d'employer des langages de balisage léger s'inscrit dans la continuité de celui réalisé pour les technologies à la base de l'éditeur de Stylo que nous avons abordé dans le chapitre IV. Il y a, d'une part, un engouement pour ces technologies dans les années 2010 du fait de leur implémentation dans les générateurs de site statique (Jekyll, Hugo, 11ty, etc.) et des logiciels de conversion comme Pandoc. Dès lors, beaucoup de chercheurs utilisent ce type de technologie pour, par exemple, créer leur site web personnel, et s'affranchissent ainsi de technologies plus lourdes nécessitant le déploiement d'une base de données. D'autre part, il s'agissait d'inscrire le Pressoir dans le même mouvement que l'éditeur de texte Stylo, ce qui a donné la possibilité de créer une passerelle entre ces deux outils. Depuis le début de l'année 2025, il est également possible de récupérer directement avec le Pressoir des contenus qui ont été saisis dans Stylo.

En 2018, le Pressoir, qui ne porte pas encore ce nom, est d'abord pensé à partir d'un script Bash. Un script Bash est un fichier texte contenant une série de commandes et d'instructions destinées à être exécutées par l'interpréteur Bash (*Bourne Again Shell*). On en retrouve les vestiges sur le dépôt gitlab de la CRCEN [36] où se trouvent les sources de l'ouvrage de David Lankes, *Exigeons de meilleures bibliothèques*, publié cette année-là à la maison d'édition les Ateliers de sens public. Ce script Bash est très intéressant, parce qu'il nous montre qu'à ce moment-là de l'histoire du Pressoir, cet outil de transformation n'a pas encore de modèle textuel.

Les technologies fondamentalement nécessaires à la fabrication du livre de Lankes sont des fichiers en texte brut, avec les textes et les contenus additionnels en Markdown, les métadonnées en YAML et les références bibliographiques en BibTeX ; auxquels nous ajoutons le convertisseur Pandoc pour transformer les sources en HTML ou au format LaTeX, afin de transformer ce dernier document en fichier PDF. Le script Bash pourrait se contenter de répéter les appels à

36. Le script permettant de construire le livre se trouve à cette adresse : `https://gitlab.huma-num.fr/ateliers-sp/lankes/-/blob/master/build.sh?ref_type=heads`, page consultée le 03 juin 2025.

Pandoc pour convertir les fichiers et les déplacer dans différents dossiers selon l'objet créé.

Néanmoins, la volonté des éditeurs des Ateliers de sens public est de créer « une complémentarité entre édition papier et édition numérique » [37]. Les deux objets, papier et numérique, ne sont alors plus pensés dans un rapport homothétique où l'un et l'autre contiendraient exactement les mêmes contenus. Pour répondre à cet enjeu, les Ateliers de sens public ont pris plusieurs décisions éditoriales [38].

Leur choix a été de proposer une version papier de l'ouvrage qui contient un discours linéaire, sans élément qui pourrait perturber cette linéarité, comme des images par exemples. La version web augmentée, quant à elle, se démarque de la version papier par sa tabularité, avec, par exemple, la prise en charge de contenus multimédias qui viennent s'agréger au texte en tant que contenus additionnels. On peut y trouver des images, des vidéos, des hyperliens, des sons, des cadres dans lesquels sont référencés d'autres sites web, du texte additionnel, pour ne citer que les exemples les plus courants. Les deux versions du livre, linéaire et tabulaire, sont imaginées pour être complémentaires et proposent ainsi deux lectures différentes.

En plus de devoir modéliser le texte et l'appareil savant d'un livre pour un environnement web, l'enjeu principal du Pressoir consiste en la modélisation de ces contenus additionnels, que l'on pourrait trouver insolites lorsqu'ils sont présents dans un livre. Les questions que cela pose ne relèvent pas seulement de la technique. Ce sont surtout des choix éditoriaux qu'il convient de faire : quel format pour quel objet ? Est-il pérenne ? Est-ce que cette pérennité a la même durée de vie qu'un livre ? Que se passe-t-il lorsque les contenus additionnels ne fonctionnent plus dans l'environnement web ? Étant donné que les technologies évoluent, de nouvelles sont publiées tandis que d'autres deviennent obsolètes, les réponses à ces questions ne peuvent pas être figées ; elles sont reposées pour chaque ouvrage.

La première étape de la modélisation textuelle de ces contenus additionnels consistait à trouver un système pour les noter en Markdown, puisque c'est ce format qui a été retenu pour écrire les textes. Les choix de la syntaxe de ces contenus additionnels faits en 2018 sont toujours valables en 2025, à quelques modifications mineures près. La saisie de ces contenus se fait en deux étapes : un marqueur est ajouté dans le fil du texte, à l'endroit où l'on veut ajouter un

37. Voir le fichier README.md dans le répertoire Gitlab du livre de Lankes : `https://gitlab.huma-num.fr/ateliers-sp/lankes/-/blob/master/README.md?ref_type=heads&plain=1#L7`, consulté le 03 juin 2025.

38. Ces décisions sont très similaires que celles prises, par les mêmes acteurs, pour la collection Parcours numériques aux Presses de l'Université de Montréal.

contenu additionnel, puis les informations du contenu sont saisies dans un fichier séparé et dédié.

Dans le texte de Lankes [39], la syntaxe des contenus additionnels créée uniquement pour le Pressoir se note comme suit :

Extrait de code V.11 : Marqueur du contenu additionnel dans le Markdown

```
!contenuadd(./videoLankesChapter1)(video)
```

Dans le fichier dédié à la description de tous les contenus additionnels [40], la syntaxe est la suivante :

Extrait de code V.12 : Description du contenu additionnel en YAML et Markdown

```
## videoLankesChapter1

---
title: >-
  Déconstruction du chapitre 1\ : « Le printemps
     arabe : exigeons 'lexceptionnel » (2016,
     12min28s)
credits: >-
  R. David Lankes
keywords: video,vimeo,anglais,lankes
lang: en
type: video
link: https://vimeo.com/202536052
link-archive: https://web.archive.org/web/20190910141424/
              https://vimeo.com/202536052
embed: https://player.vimeo.com/video/202536052
zotero:
date: 2016
date-publication: 2019-10-28
source: editeur
priority: lowpriority
position: main
---

Dans cette vidéo, R. David Lankes déconstruit le texte du
chapitre 1 « Le printemps arabe : exigeons
l'exceptionnel » de l'ouvrage *Exigeons de meilleures
bibliothèques* et traite de l'utilisation de ce texte au
sein d'une conversation plus large sur les bibliothèques
et leurs services.
```

39. Voir la source dans le répertoire Gitlab du livre : https://gitlab.huma-num.fr/ateliers-sp/lankes/-/blob/master/textes/chapitre1/chapitre1.md?ref_type=heads&plain=1#L66, consultée le 04 juin 2025.

40. Voir la source dans le répertoire du livre : https://gitlab.huma-num.fr/ateliers-sp/lankes/-/blob/master/textes/additionnels.md?ref_type=heads&plain=1#L144, consultée le 04 juin 2025.

Le moyen de créer un lien entre ces deux objets, placés dans des fichiers différents, repose sur la saisie rigoureuse du titre de niveau 2 du contenu additionnel que l'on reporte dans le texte, comme cela est montré dans le bloc de code V.11. Dans l'exemple ci-dessus, l'élément qui sert d'identifiant est : `videoLankesChapter1`.

Comme nous le mentionnons précédemment, cette syntaxe n'a pas beaucoup évolué depuis 2018. Ceci s'explique certainement parce que la modélisation sous-jacente à celle-ci suffit pour signifier ces contenus additionnels. La documentation du Pressoir [41] nous montre que le principal changement apporté au fonctionnement des contenus additionnels est structurel : au lieu de créer un seul fichier à la racine du livre qui décrit tous les contenus additionnels, un fichier est dorénavant créé pour chaque chapitre, comme cela a été appliqué pour le livre *Contribution numérique* [42].

Ces contenus additionnels ne doivent pas être réduits à de simples objets multimédias insérés au milieu d'un livre. L'exemple ci-dessus illustre l'organisation des informations selon une structure créée à leur intention. En effet, elle requiert d'abord un titre de niveau deux en Markdown, puis l'usage de la syntaxe généralement employée pour créer un *frontmatter* YAML dans un fichier en Markdown pour sérialiser une série d'informations censées décrire l'objet inséré dans le texte. Lorsqu'une vidéo hébergée sur la plateforme Viméo est insérée en tant que contenu additionnel, tel que c'est le cas dans l'exemple V.12, on n'y trouve pas uniquement un hyperlien menant à cette ressource, mais bien tout un ensemble d'éléments descriptifs qui contextualisent le contenu additionnel dans le livre et le positionne au sein des autres informations. Les descripteurs ainsi employés dans cet espace définissent littéralement le contenu additionnel du livre.

Nous notons également que l'insertion de ces trois caractères --- pour délimiter les descripteurs YAML de chaque contenu additionnel dans un fichier détourne leur fonction principale de délimiteur du *frontmatter*. Il ne s'agit plus du *frontmatter* habituel que l'on écrit une seule fois en en-tête d'un fichier pour spécifier des métadonnées, mais d'un objet récurrent que l'on mobilise autant de fois qu'il y a de contenus additionnels dans un chapitre. Dans l'environnement du Pressoir, et pour la description des contenus additionnels, la délimitation YAML --- change de sens et marque dorénavant les limites d'un jeu de métadonnées relatives à un contenu additionnel.

41. Voir la documentation en ligne : `https://ecrinum.gitpages.huma-num.fr/pressoir/chapitre4.html#FormulaireYaml`, consultée le 04 juin 2025.

42. La structure du livre *Contribution numérique* est visible sur le dépôt Gitlab des sources : `https://gitlab.huma-num.fr/ateliers-sp/contributionnum/-/tree/main/textes?ref_type=heads`, consultée le 04 juin 2025.

Afin de transformer ces contenus un peu particuliers en éléments HTML, le script Bash de 2018 utilise massivement la commande `sed`[43] (*stream editor for filtering and transforming text*). C'est donc à coups de `sed`, et de `grep` (*print lines that match patterns*), que le Pressoir de 2018 manipule les éléments textuels pour les transformer.

La manipulation des éléments textuels au moyen de ces expressions régulières, donc au moyen de *pattern* présents dans le texte montre que le Pressoir ne s'appuie pas sur un modèle textuel. Il n'y a pas de théorie du texte sur laquelle s'appuyer, mais uniquement des motifs dans le texte à partir desquels le Pressoir se repère pour effectuer les transformations. En 2024, lorsque le Pressoir est une librairie Python, il ne s'appuie plus sur ces commandes pour transformer les éléments textuels, mais sur un parser. Un parser est un analyseur syntaxique qui analyse une séquence d'éléments en suivant un ensemble de règles syntaxiques définies par une grammaire formelle. Les motifs utilisés par les expressions régulières ont été implémentés dans le parser du Pressoir. C'est de cette manière que le modèle textuel du Pressoir a été créé.

Si le reste du script Bash mobilise des commandes similaires pour construire les éléments de navigation entre les pages HTML du livre, ce sont les dernières lignes du script qui attirent notre attention[44]. Ces dernières lignes appellent un autre logiciel, BaseX[45], un *framework* utilisé pour créer des bases de données en XML, afin de produire les index à la fin du livre web augmenté. C'est-à-dire que le prototype du Pressoir en Bash touche ici à sa limite, et nécessite la dépendance à un autre logiciel pour créer une fonctionnalité. Cette dépendance à BaseX est l'une des raisons pour lesquelles le Pressoir est passé d'un script *Bash* au langage de programmation Python utilisé dans les dernières versions. Ce changement permet notamment de s'émanciper de cette dépendance.

Les transformations des chapitres de *Contribution numérique*

Le Pressoir et ses fonctionnalités maintenant présentées, nous pouvons passer aux transformations des textes étudiés dans le chapitre III. Ceux-ci ont été majoritairement fournis par les auteurs dans des formats de traitement de texte, ODT et DOCX. Néanmoins, comme nous venons de le voir, le Pressoir n'utilise pas ces formats en entrée, avant transformation des sources en objet publiable.

43. Voir les commandes `sed` dans le script Bash : `https://gitlab.huma-num.fr/ateliers-sp/lankes/-/blob/master/build.sh?ref_type=heads#L58`, page consultée le 04 juin 2025.

44. Voir le script à partir de la ligne 132 : `https://gitlab.huma-num.fr/ateliers-sp/lankes/-/blob/master/build.sh?ref_type=heads#L132`, consulté le 04 juin 2025.

45. La documentation de BaseX est disponible sur le site web suivant : `https://basex.org/`, consulté le 04 juin 2025.

Il a donc été nécessaire de transformer les fichiers envoyés par les auteurs en fichiers aux formats Markdown, YAML et BibTEX.

Cette conversion a été réalisée manuellement par Hélène Beauchef et moi-même à l'aide du logiciel Pandoc. La commande Pandoc employée [46] pour tous les fichiers était la suivante :

Extrait de code V.13 : Commande Pandoc DOCX ou ODT vers Markdown

```
pandoc -s nom-du-fichier.extension -t markdown -o
    nom-du-fichier.md
```

Les résultats obtenus avec cette commande ne permettaient pas d'utiliser le Pressoir en l'état pour transformer les documents en artefacts publiables. Plusieurs étapes de réécriture ont été indispensables. Elles comprennent notamment la restructuration du texte, la séparation des métadonnées et du corps du texte, l'insertion des contenus additionnels et l'organisation des textes selon l'arborescence prise en charge par le Pressoir. À cela s'ajoute la création d'un fichier de configuration au format YAML qui contient les métadonnées du livre [47] utilisées à la fois pour enrichir les documents finaux, mais aussi pour configurer certains éléments tels que la table des matières.

Au-delà d'une simple conversion du traitement de texte vers le texte brut, on comprend tout de suite la nécessité de cette manipulation, car les modèles textuels des traitements de texte MSWord et LibreOffice ne permettent pas d'insérer des contenus additionnels dans le texte dans le sens que leur confère le Pressoir, puisque ces objets n'existent pas dans les modèles textuels des traitements de texte.

La première étape de transformation correspond donc à une phase préliminaire de réécriture des textes afin que les modèles textuels des sources et du Pressoir concordent. Sans cette étape, le livre ne saurait être produit avec le Pressoir. L'étape suivante concerne les transformations essentielles à la publication du document, au même titre que l'article de la revue *Humanités numériques* a été transformé par le module d'export de Stylo, puis par Lodel pour être diffusé sur la plateforme OpenEdition.

La pierre angulaire du Pressoir est le convertisseur Pandoc. C'est grâce à cet outil que le Pressoir peut transformer les sources en artefacts publiables. Pandoc

46. Les fichiers envoyés par les auteurs et les résultats bruts de cette première transformation ont été archivés dans un dépôt gitlab séparé. Il est accessible à l'adresse suivante : `https://gitlab.huma-num.fr/ateliers-sp/contributionnum-archives/-/tree/main/sources-avril-2023?ref_type=heads`, consulté le 04 juin 2025.

47. Voir la page suivante disponible sur le répertoire Gitlab du livre : `https://gitlab.huma-num.fr/ateliers-sp/contributionnum/-/blob/main/textes/garde/livre.yaml?ref_type=heads`, consultée le 04 juin 2025.



Figure V.3 : Schéma du fonctionnement de Pandoc

est un outil *open source* largement utilisé dans d'autres chaînes de transformations, notamment académiques (Dürkop 2020 ; Krewinkel et Winkler 2017). C'est un outil incontournable pour convertir un fichier dont le contenu est structuré selon un format vers un autre format, une autre structure.

Toutefois, il y a beaucoup de formats différents, et de structures différentes, alors comment Pandoc fait-il pour effectuer et maintenir toutes ces conversions ? Si de prime abord le logiciel semble fonctionner sur la base d'équivalences et de correspondances entre les structures présentes dans les différents formats traités par Pandoc, comme le serait la syntaxe Markdown avec un titre de niveau 2 `##` et la syntaxe HTML `<h2>`, il en est tout autrement, car cela impliquerait de maintenir ces équivalences entre tous les formats. Cette tâche se révélerait très chronophage puisqu'elle sous-entendrait que pour chaque modification d'une structure d'un format, il faudrait effectuer des vérifications auprès de tous les autres formats.

Afin d'éviter ce fonctionnement, MacFarlane, l'auteur de Pandoc, a mis au point une structure de données interne au logiciel qui permette de représenter la syntaxe abstraite d'un texte. Cette technologie se nomme *Abstract Syntax Tree* (AST) ou arbre de la syntaxe abstraite. C'est une technologie répandue lorsqu'il s'agit de représenter des textes sans s'embarrasser de toute la syntaxe d'une langue. Cette forme de représentation permet de ne sélectionner qu'un ensemble d'éléments jugés essentiels d'une structure linguistique que l'on organise sous une forme arborescente (Jones 2003). Concrètement, lorsque l'on demande à Pandoc de convertir un fichier, celui-ci interprète le contenu d'un document, qu'il représente sous une forme arborescente, puis transforme cette arborescence selon le format souhaité. Dans la documentation de Pandoc, ce fonctionnement est résumé par le petit schéma [48] de la figure V.3.

Pandoc **lit** le texte avec des programmes appelés *reader*, qui lui permettent de décomposer les différents éléments présents en entrée, puis **écrit** grâce à des *writer* pour générer le texte structuré selon le format de sortie demandé. Ce qu'il y a entre ces deux étapes, l'AST, est une nouvelle forme significative que prend le document et le texte, car selon ce que *lit* Pandoc, l'AST prendra une certaine forme et Pandoc écrira selon elle le texte en sortie. Il y a donc à l'endroit

48. Ce schéma est disponible sur le site web de Pandoc : `https://pandoc.org/using-the-pandoc-api.html#pandocs-architecture`, consulté le 04 juin 2025.

de cet AST une conception et une théorie du texte différente de celle que nous avons rencontrée avec le modèle OHCO. Pandoc ne fonctionne donc pas grâce à des correspondances entre les formats, il fonctionne par l'interprétation de ces structures à travers sa propre représentation du texte.

Cette modélisation du texte sous la forme d'un AST est accessible dans Pandoc. Pour cela, il suffit de lui demander de générer le document selon le format de sortie *native*.

Si l'on prend un des textes du livre *Contribution numérique*, et qu'on lui applique ce paramètre de sortie [49] pour obtenir sa représentation selon cet AST, voici un échantillon de ce que renvoie Pandoc :

Extrait de code V.14 : Échantillon de l'AST de Pandoc

```
[ Header
    2 ( "introduction" , [] , [] ) [ Str "Introduction" ]
, Para
    [ Span
        ( ""
        , [ "conceptEX" ]
        , [ ( "idsp" , "Histoires de nature - Changing
            natures" )
          , ( "idreference" ,
              "https://www.changing-natures.org/fr/" )
          , ( "urlarchive"
          , "https://web.archive.org/web/20231031101225/
          https://www.changing-natures.org/fr"
            )
          ]
        )
        [ Link
            ( ""
            , []
            , [ ( "link-archive"
            , "https://web.archive.org/web/20231031101225/
            https://www.changing-natures.org/fr/"
                )
              ]
            )
            [ Str "Histoires"
            , Space
            , Str "de"
            , Space
            , Str "nature."
            , Space
            , Str "Une"
            , Space
            , Str "collection"
            , Space
            , Str "participative"
```

49. La commande utilisée est : `pandoc --from markdown --to native texte.md`.

```
35              , Space
36              , Str "sur"
37              , Space
38              , Str "les"
39              , Space
40              , Str "changements"
41              , Space
42              , Str "environnementaux"
43              ]
44              ( "https://www.changing-natures.org/fr/" , "" )
45          ]
46      , Space
47      , Str "(en"
48      , Space
49      , Str "allemand"
50      , Space
51      , Emph
52          [ Str "Natur"
53          , Space
54          , Str "der"
55          , Space
56          , Str "Dinge,"
57          , Space
58          , Str "eine"
59          , Space
60          , Str "partizipative"
61          , Space
62          , Str "Sammlung"
63          , Space
64          , Str "des"
65          , Space
66          , Str "Anthropoz\228ns"
67          ]
```

Cette représentation découpe le texte élément par élément : il n'y a plus de phrase, plus de syntaxe ni d'élément sémantique comme des auteurs. À la place, nous trouvons une pluralité d'éléments divisés entre eux et organisés en arborescence. Une chaîne de caractères `Str`, un espace `Space` ou une emphase `Emph` sont ici placés sous l'élément paragraphe `Para`. Pandoc représente ainsi le document selon des entités structurelles pour lesquelles le texte lui même ne devient qu'une valeur affectée à cette entité. C'est grâce à cette modélisation intermédiaire que Pandoc réalise les conversions d'un format à un autre.

Les actions de lecture et d'écriture réalisées par Pandoc ne sont pas des vases clos. Il est tout à fait possible, dès lors que l'on maîtrise le langage de programmation Python, Lua ou Haskell, d'écrire des filtres[50] pour prescrire un autre comportement que le comportement par défaut de Pandoc. C'est en

50. Les filtres sont documentés sur le site web de Pandoc : `https://pandoc.org/filters.html#summary`, page consultée le 04 juin 2025.

s'appuyant sur cette modularité de l'AST de Pandoc, et bien d'autres spécificités, qu'une équipe de l'Université de Washington a créé la chaîne de transformations *Living papers*. Cette chaîne est destinée aux articles scientifiques et permet l'intégration d'objets dynamiques à l'intérieur de ces derniers, comme du code à exécuter (Heer et al. 2023).

Si toutefois, le Pressoir ne fait pas usage de ces filtres, il s'appuie tout de même sur ce modèle textuel de l'AST pour produire les différents formats des livres. On observe donc un glissement entre la modélisation du texte selon le principe OHCO que l'on trouve en entrée et en sortie du Pressoir, avec une hiérarchie d'objets ordonnés, et une modélisation intermédiaire sous la forme d'un arbre syntaxique lorsque le document est traité par Pandoc. Le rôle du Pressoir consiste alors à jongler entre ces différentes modélisations, ces différentes théories du texte, pour produire les artefacts désirés.

Un des formats échappe toutefois à ce schéma de production des documents, puisqu'il nécessite une étape intermédiaire supplémentaire pour être produit. Il s'agit du format PDF que Pandoc ne peut pas compiler sans passer par un environnement tiers. Dans le cas du Pressoir, celui-ci fait appel à LaTeX.

LaTeX, comme nous l'avons mentionné dans le chapitre III.2, est un logiciel de composition typographique basé sur des commandes que l'on écrit directement dans le texte (brut) et que le logiciel se charge d'interpréter. Ces commandes encapsulent un fragment de texte à traiter. Par exemple la commande pour mettre en emphase un mot sera notée comme ceci : `\emph{un mot}`. Malgré le fait que le texte traité soit entouré par des accolades, cela ne signifie pas que la commande ait une valeur sémantique. Le fonctionnement des commandes n'est pas du tout similaire à un langage de balisage comme HTML. Ces commandes sont des éléments interprétés par LaTeX pour compiler et produire un document. Si LaTeX prescrit un comportement par défaut à une commande, il est néanmoins tout à fait possible de la réécrire et d'obtenir un comportement totalement différent de celui par défaut. En ce sens, un document LaTeX sans son préambule (ni la version de LaTeX utilisée) n'a quasiment que peu de valeur au-delà du fait que les commandes sont inscrites dans le texte : avec les mêmes commandes passées à un texte, selon le préambule que l'on rédige, nous obtenons des documents complètement différents.

C'est en ce sens que nous avons parlé de raison fonctionnelle dans le chapitre III.2, car les commandes utilisées dans LaTeX se rapprochent plus de la fonction mathématique que de la balise sémantique. Ainsi, le modèle textuel de LaTeX ne relève ni de la modélisation OHCO, ni de celle de l'AST que l'on trouve à l'intérieur Pandoc, mais relèverait plutôt d'une modélisation mathématique du

texte. Il s'agit là d'une troisième théorie du texte convoquée dans cette chaîne de transformation qu'est le Pressoir.

Si l'on reprend l'exemple précédent dans l'AST de Pandoc, le texte transformé par le convertisseur au format TEX ressemble à ceci :

Extrait de code V.15 : Extrait de l'introduction d'un chapitre du livre transformé au format TEX par Pandoc

```
1 \hypertarget{introduction}{%
2 \section*{Introduction}\label{introduction}}
3 \phantomsection\addcontentsline{toc}{section}{Introduction}
4
5 {\href{https://www.changing-natures.org/fr/}{Histoires de
     nature. Une
6 collection participative sur les changements
     environnementaux}} (en
7 allemand \emph{Natur der Dinge, eine partizipative Sammlung
     des
8   Anthropozäns}
```

Nous retrouvons bien les différents éléments vus dans l'AST de Pandoc, avec un titre, puis un paragraphe contenant un hyperlien et une emphase contenant une chaîne de caractères en allemand. Toutefois, nous notons quelques différences. Tout d'abord, le titre se voit doté de caractéristiques supplémentaires telles que les commandes `\hypertarget{}`, `\label{}` et `\phantomsection{}` qui permettent d'identifier le titre de la section dans diverses strates du document en y créant une ancre. De plus, le titre perd son niveau sémantique, qui était un titre de niveau 2, et est remplacé par la commande `\section*{}`.

Lorsque l'on crée un document de la classe livre (*book*) dans l'environnement LATEX, celui-ci comporte différents niveaux hiérarchiques : le titre du document, des parties (optionnelles), des chapitres, des sections, des sous-sections. Si l'on part du principe que le titre du document correspond à un titre de niveau 1, et que nous n'avons pas de partie dans la version imprimée du livre *Contribution numérique*, il serait logique de s'attendre à ce que les titres des chapitres correspondent à des titres de niveau 2. À ce compte-là, les sections seraient équivalentes à des titres de niveau 3, comme c'est par exemple le cas pour cette thèse.

Or, dans les extraits que l'on vient de voir, l'AST de Pandoc dans l'extrait V.14, et le contenu Markdown correspondant [51], affichent bien que le titre de la section est un titre de niveau 2. Nous observons donc qu'un décalage sémantique se forme au cours de la transformation puisque la notion de section, par défaut un titre de niveau 3, est créé à partir d'un titre de niveau 2.

51. L'extrait de la source en Markdown est disponible en ligne : `https://gitlab.huma-num.fr/ateliers-sp/contributionnum/-/blob/main/textes/chapitre10/chapitre10.md?ref_type=heads&plain=1`, page consultée le 06 juin 2025.

Cette action est réalisée par un paramètre passé à Pandoc par le Pressoir lors de la transformation du Markdown en TEX : il s'agit du paramètre `--top-level-division=chapter`[52] qui déclare le plus au niveau de division (le titre de niveau 1) comme étant le chapitre. Cette opération où l'on modifie le comportement de la commande LATEX `\section{}` permet de garder une structure textuelle cohérente entre les différents formats produits par le Pressoir. Si l'on observe la version web augmentée, chaque chapitre fait l'objet d'une page différente et pour chacune de ces pages, le titre de niveau 1 est le titre du chapitre. Puisque le titre du chapitre est de niveau 1, il semble cohérent que la division suivante dans l'organisation du texte soit de niveau 2 [53]. Afin de préserver cette organisation et ne pas créer une rupture dans le principe de *single source publishing*, il est nécessaire de créer ce décalage sémantique pour que le titre de niveau 2 dans le Markdown puisse devenir une division de niveau 3 dans la structure du livre imprimé (et de la version PDF).

Cet exemple nous montre bien que les commandes LATEX passées au texte n'ont pas de valeur sémantique à proprement parler, et qu'il est nécessaire de se référer au reste de l'environnement d'écriture pour en faire émerger le sens.

Ceci conclut les différentes étapes de transformation opérées par le Pressoir. Au moyen de ces transformations, que ce soit via Pandoc ou LATEX, le Pressoir est en capacité de produire les différents artefacts à publier.

Chacun de ces artefacts répond à une modélisation du texte et présente ainsi son lot de différences. Si les différences entre le PDF et la version web augmentée en HTML semblent évidentes, celles entre les artefacts dans les formats ePUB et la version web augmentée pourraient ne pas l'être, étant donné que ces formats mobilisent sensiblement les mêmes technologies pour être produits, puisque nous y retrouvons les formats HTML et CSS dans les deux cas.

Pourtant, dès que l'on observe les en-têtes des documents, l'on remarque que les espaces de noms diffèrent, celui de l'ePUB renvoie à deux espaces de nom, d'abord celui d'XHTML, qui est un espace de nom relativement générique puisqu'il fait référence à plusieurs spécifications (HTML5, XHTML1.0, XHTML1.1, etc.) ; puis celui de la spécification d'ePUB3.

Extrait de code V.16 : Extrait de l'en-tête de la version ePUB

52. Le paramètre `--top-level-division=chapter` est passé à la commande Pandoc située dans le script Python suivant : `https://gitlab.huma-num.fr/ateliers-sp/contributionnum/-/blob/main/generateTex.py?ref_type=heads#L227`, consulté le 05 juin 2025.

53. On peut observer la structure du livre augmenté en ligne. Pour cela il suffit d'ouvrir les outils de développement et d'inspecter les éléments HTML ou d'appuyer sur le raccourci clavier `ctrl+u` pour ouvrir le code source dans un nouvel onglet. L'URL de ce chapitre est la suivante `https://ateliers.sens-public.org/contribution-numerique/chapitre10.html`, consultée le 05 juin 2025.

```
1 <html xmlns="http://www.w3.org/1999/xhtml"
     xmlns:epub="http://www.idpf.org/2007/ops" xml:lang="fr">
2 <head>
3   <meta charset="utf-8" />
4   <meta name="generator" content="pandoc" />
5   <title>ch001.xhtml</title>
6   <style>
7   </style>
8   <link rel="stylesheet" type="text/css"
       href="../styles/stylesheet1.css" />
9 </head>
```

On remarque également que le nombre de métadonnées est très restreint dans l'ePUB. Tandis que si l'on compare cet en-tête à la version web augmentée[54], les métadonnées sont beaucoup plus précises. On y trouve plusieurs vocabulaires contrôlés tels que le *Dublin core*, le *Publishing Requirements for Industry Standard Metadata* (PRISM) ou encore l'*Open Graph* (OG). La version web augmentée pourrait très bien se passer de toutes ces descriptions pour « fonctionner ». Si l'on souhaitait faire un objet très minimaliste à n'afficher que sur quelques écrans, l'en-tête HTML pourrait être réduite à son minimum et afficher sensiblement les contenus de la même manière. Toutefois, une différence fondamentale s'imposerait entre la version augmentée et la version minimaliste (fictive) : elles ne s'inscriraient pas sur le Web de la même façon.

Les algorithmes de référencement, les robots de Google et autres moteurs de recherche, les outils de *scrapping* ou de *crawling*, les plugins de récupération des informations bibliographiques comme Zotero ne détecteraient pas chaque chapitre de ce livre augmenté et n'accéderaient pas aux documents dans les mêmes conditions et, de fait, notre rapport à ces objets en tant qu'usager en serait modifié.

Malgré le fait que les technologies mobilisées soient similaires, les différences que l'on observe entre l'objet ePUB et HTML relèvent en conséquence de cette disparité qu'il y a entre les milieux dans lesquels ils sont publiés et les réseaux d'objets dans lesquels ils s'inscrivent. C'est en ce sens que les transformations des sources du livre *Contribution numérique* réalisées par le Pressoir intègre plus largement le processus d'éditorialisation.

Cette analyse met en évidence que la chaîne éditoriale le Pressoir combine plusieurs modèles textuels complexes afin de produire divers artefacts publiables à partir d'une source unique, comme pour le livre *Contribution numérique*.

En suivant la figure V.4 qui résume notre analyse de la production du sens dans le Pressoir, nous pouvons observer que la modélisation des éléments textuels

54. Par exemple, en visitant les sources de l'introduction du livre à cette URL : `view-source:https://ateliers.sens-public.org/contribution-numerique/introduction.html`, consultée le 05 juin 2025.

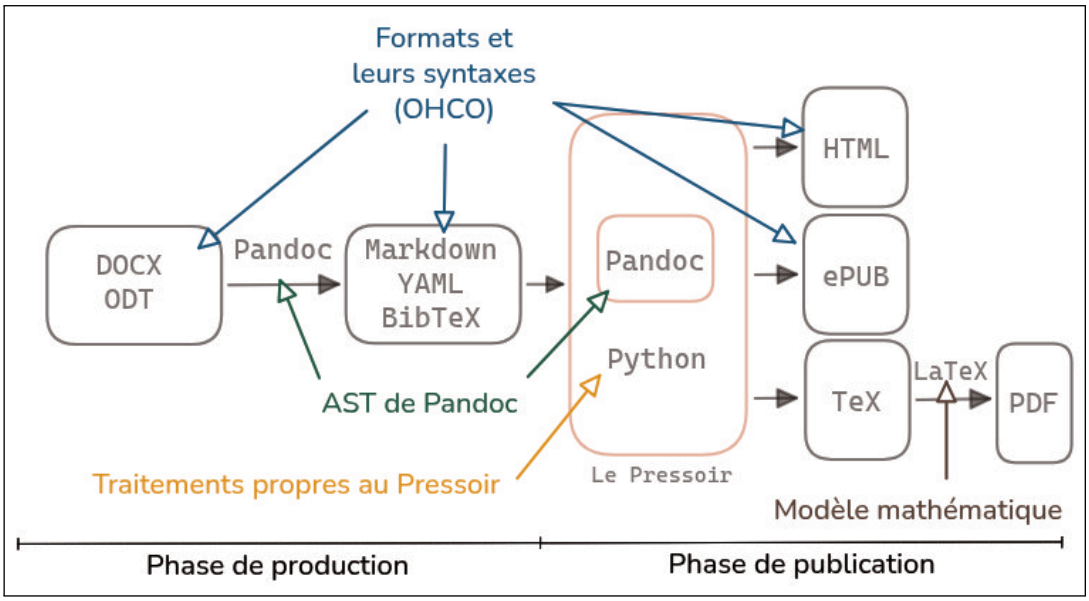


Figure V.4 : Schéma de la production du sens dans le Pressoir

repose d'abord sur les documents et leurs formats issus de la phase de production de l'écriture, ainsi que sur les règles syntaxiques associées, ce que l'on considère comme étant la source à transformer. Elle s'appuie ensuite sur leurs parsers, enrichis de traitements spécifiques propres au Pressoir, pour générer les sorties finales. Pour réaliser ces transformations, le Pressoir utilise la modélisation en arbre syntaxique abstrait (AST) de Pandoc, puis la représentation mathématique de LaTeX pour produire le fichier PDF. Les autres formats sont, quant à eux, générés directement par Pandoc. En suivant l'ensemble de ces étapes, chaque document acquiert une structure spécifique, adaptée pour s'intégrer de manière cohérente au sein d'un réseau d'objets hors de l'environnement d'écriture.

V.5 Conclusion

À travers ce chapitre, nous avons vu que l'écriture numérique est une action qui s'étend bien au-delà de la saisie d'un texte dans un environnement d'écriture, que ce soit Stylo ou un logiciel de traitement de texte. Pour être publié, un document doit nécessairement être transformé et adapté à l'espace dans lequel il va s'inscrire.

L'édition connaissait déjà ce processus avec la transformation d'un manuscrit en un livre, par exemple. Le livre peut alors être inséré dans une collection, dans un rayon de bibliothèque, etc. L'inscription d'un livre dans un réseau d'objets est un phénomène maîtrisé et contrôlé à l'intérieur de la chaîne éditoriale pour laquelle les différents rôles et et leurs actions sont délimités.

Lorsqu'il s'agit d'un objet numérique, que ce soit un article ou un livre, nous avons vu dans la première section de ce chapitre V.2 que la chaîne éditoriale traditionnelle n'était plus adaptée à l'espace numérique et qu'une reconfiguration des différents rôles qu'elle comportait a été nécessaire. C'est à cette question qu'entend répondre la notion d'éditorialisation. L'exemple de la plateforme de diffusion OpenEdition étudiée dans la section V.3 illustre bien cette redistribution, puisqu'elle combine à elle seule les rôles de distributeur, de diffuseur, de libraire et parfois d'imprimeur, lorsque la plateforme génère ses propres fichiers.

Seulement, l'acte de publication ne consiste pas uniquement en la mise à disposition d'un document à travers une interface, que ce soit le Web, un lecteur PDF ou une liseuse d'e-book ; elle consiste en l'inscription de l'objet à publier au sein d'un réseau d'objets plus vaste. Ce réseau d'objet peut se traduire par un lien direct, l'hyperlien, et relier ainsi deux objets entre eux. Toutefois, il s'avère que ce réseau d'objets est bien plus complexe que de simples liens directs, notamment lorsqu'on évoque le Web, qu'il s'agisse d'un livre augmenté produit avec le Pressoir ou d'un article de la revue *Humanités numériques* publié avec le CMS Lodel, puisque d'autres entités vont accéder à l'objet publié pour l'indexer, le référencer et le mettre en relation avec d'autres objets circulant sur le Web. Ces processus ouverts de circulation de l'information ne dépendent pas de la chaîne éditoriale, mais de la configuration de l'espace. En ce sens, le processus d'éditorialisation s'ouvre à l'ensemble des médiations permettant d'inscrire de façon dynamique un objet éditorial dans un réseau d'entités plus vaste. En cela, l'éditorialisation dépasse le processus d'édition numérique jusqu'à s'en détacher.

L'écriture numérique, initialement envisagée selon une division en deux phases distinctes – la production, puis la publication – s'inscrit, en ce sens, dans un processus d'édition et d'éditorialisation. L'écriture numérique n'est plus seulement distribuée dans un environnement technique, mais l'est également dans un environnement éditorial. Ainsi, elle n'est alors plus contingentée à un seul modèle textuel, celui que l'auteur du texte choisi, mais l'est également à celui du modèle textuel éditorial en vue de la publication dudit texte. Le texte et son sens sont alors étroitement liés à une politique éditoriale.

Le passage de la phase de production à la phase de publication sous-entend des transformations à l'échelle du document : l'article de la revue *Humanités numériques* rédigé dans Stylo, ou les chapitres de l'ouvrage *Contribution numérique* ne sont publiables en l'état, car ils ne pourraient s'inscrire que dans des réseaux d'objets très limités. Pour ce qui est de l'article sur Stylo, si l'on préservait le document sans rien transformer, il ne pourrait pas du tout être importé sur la plateforme OpenEdition. Il en va de même pour les chapitres du livre issus des logiciels de traitement de texte.

Dès lors que ces transformations sont requises, les environnements d'écriture numérique sont étendus à des services de conversion pour manipuler et modifier les documents afin que la structure des informations qu'ils contiennent soit alignée sur le modèle de publication attendu. C'est à ce moment-là que des outils comme le service d'export de Stylo, Lodel ou encore le Pressoir entrent en scène et traitent les différents documents que nous avons étudiés. Leur rôle est d'adapter le modèle textuel initial au modèle textuel de la publication. Cette étape d'adaptation ne saurait être réduite à une simple conversion. En effet, il s'agit d'une réécriture complète du document puisque, bien souvent, lorsqu'il est nécessaire de transformer un document, il y a un changement d'autorité auquel le document doit se référer, que ce soit des formats, des schémas, des espaces de nom, etc. Ces changements, ces transformations, en se référant à un nouveau modèle textuel, redéfinissent ainsi la signification accordée à chacun des éléments qui compose le document. De cette manière, ces transformations permettent au document de s'inscrire véritablement au sein d'un réseau d'objets/entités numériques et en changent ainsi profondément la signification.

Les transformations ainsi appliquées aux documents font intégralement partie des dynamiques qui constituent l'écriture numérique.

Plus que le contenu du document, voué à cette variabilité propre à l'écriture numérique, c'est au processus lui-même qu'il convient d'accorder de l'importance, en ce sens qu'il est à l'origine de cette variabilité et des variations que connaît le contenu. Chaque transformation, et plus largement chaque étape de l'écriture numérique dans ses phases de production et de publication, correspond à une, voire à des coupures agentielles mobilisant leur lot de conjonctures médiatrices et de transformations du sens conféré au document. À chacune de ces coupures agentielles, le document numérique change de sens, d'abord du fait des changements internes de son contenu, comme nous avons pu l'observer avec les documents étudiés durant ce chapitre ; puis, surtout, car en se transformant, il accède ainsi à un nouvel environnement et se retrouve indexé au milieu d'un nouveau réseau d'informations, soit d'une nouvelle organisation des connaissances.

Conclusion

La présente recherche, qui arrive à son terme, avait pour objectif de fournir une réponse à la problématique du sens produit par les environnements d'écriture numérique. Cette problématique trouve son origine dans plusieurs constats. Dans un premier temps, il a été observé que la taille d'un fichier pouvait fluctuer en fonction de la structure des éléments textuels qu'il contient. Cette différence est particulièrement notable pour les fichiers générés par des logiciels de traitement de texte, qui sont souvent plus volumineux que ceux créés par des éditeurs de texte brut. Le second constat a été réalisé lorsque nous avons été confrontés à l'attachement profond de certains auteurs à la dimension graphique de l'écriture. De plus, nous avons également observé une forme de rejet de certains environnements qui séparent le fond de la forme et ne se préoccupent pas de cette dernière. Enfin, dans un billet de blog publié en 2015 par la CRCEN (2015*a*), le dernier constat sur lequel nous nous appuyons mentionne le fait qu'il n'existe pas d'environnement d'écriture adapté pour les SHS, c'est-à-dire des environnements où l'on peut saisir des éléments textuels spécifiques aux SHS et qui profitent de toute l'étendue des propriétés de l'espace numérique. Par cela, nous entendons la possibilité de produire des documents qui puissent intégrer des éléments textuels propres à l'appareil scientifique que l'on retrouve dans les publications scientifiques afin de répondre aux besoins des chercheurs en termes de production et de publication de leurs textes.

À la suite de ces observations, notre hypothèse était la suivante : un ensemble de médiations sont à l'œuvre dans les environnements d'écriture numérique et produisent le sens accordé aux documents qui en résultent. Parmi ces médiations, nous avons remarqué que des théories du texte sont implémentées dans les environnements d'écriture numérique sous la forme de modèle textuel. Ces théories du texte définissent alors des notions, des concepts, des fonctionnalités grâce auxquelles il devient possible d'écrire et de produire du sens. Autrement dit, le sens produit par l'écriture numérique diffère selon l'environnement employé parce que ce dernier porte et applique une théorie du texte à l'écriture. Ces théories du texte ne sont pas désincarnées et abstraites. Les concepts qui y sont définis infiltrent alors l'écriture et participent à la production du sens, car ce sont eux qui organisent, structurent et appliquent leur définition aux caractères saisis dans

l'environnement d'écriture numérique. En ce sens, les environnements d'écriture numérique ne sont pas neutres, bien au contraire, ils deviennent des éléments significatifs du dispositif d'écriture à travers cette théorie qu'il incarne et qui modèle le texte.

Cette hypothèse a été constituée à partir d'une approche intermédiale, néomatérialiste et posthumaine de l'écriture ; puis elle a été confrontée à cinq environnements d'écriture, trois environnements de production et deux environnements de publication. Afin de comprendre les modalités de production du sens de ces environnements, nous avons étudié un ensemble de documents produits par ces derniers.

Au moyen de ces approches théoriques, nous avons défini l'écriture en tant que dispositif technique distribué parmi les agents qui le composent. La notion de dispositif est convoquée depuis la perspective que nous en donne Bachimont (2010*b* : 22), c'est-à-dire en tant qu'« organisation spatiale d'éléments telle que cette configuration détermine un déroulement temporel ». Ainsi perçu, le processus réalisé au sein d'un dispositif technique est modélisé en plusieurs étapes, selon l'organisation de ses éléments. Nous ajoutons à cette définition du dispositif la thèse relative à la technique proposée par Leroi-Gourhan. Selon cette dernière, la cognition et la pensée seraient des conséquences de la technique, et non l'inverse (Leroi-Gourhan 1964). Appliquée au dispositif technique, cette thèse nous laisse alors supposer que c'est le dispositif qui produit l'écriture et la pensée. Bachimont abonde également dans ce sens lorsqu'il signale qu'« une pensée n'est donc pas un idéal flottant dans l'éther de la conscience, mais un processus dynamique de réinscriptrion » (Bachimont 2010*b* : 125). Les travaux de Jeanneret, de Souchier et de Paveau, respectivement sur l'énonciation éditoriale et la distribution de l'écriture numérique, nous amènent alors à postuler que l'écriture est distribuée dans tous les éléments qui constituent ce dispositif. Le dispositif n'est ainsi plus observé à une échelle globale, mais à celle de chaque agent qui le compose, car c'est à ce niveau que se déroulent les médiations permettant la circulation des informations en son sein.

Nous avons traversé cinq environnements d'écriture (MSWord, LibreOffice, Stylo, Lodel et le Pressoir) à travers deux cas d'étude : un livre publié aux Ateliers de sens public, *Contribution numérique : cultures et savoirs*, et un article publié dans la revue *Humanités numériques* sur la pleteforme OpenEdition Journals. Le premier mettait en oeuvre un dispositif s'appuyant sur MSWord, LibreOffice et le Pressoir, tandis que le second s'appuyait sur l'éditeur de texte Stylo et le CMS Lodel.

À travers ces cas d'étude, nous avons remarqué que les environnements d'écriture numérique ne mobilisent jamais qu'une seule théorie du texte. Le fait

de prendre en compte toutes les étapes de la production à la publication des écrits nécessitent en conséquence de mobiliser plusieurs théories du texte. Ce phénomène est d'autant plus accru lorsqu'il s'agit de dispositif pensé selon le principe de *single source publishing*, comme c'est le cas de l'environnement Stylo. Contrairement à ce que l'on pourrait envisager, l'enjeu de ce type de dispositif d'écriture et d'édition n'est pas technique, il est théorique, dans le sens où la production d'un ou plusieurs documents est contingenté à la capacité d'aligner plusieurs théories du texte entre elles pour qu'un texte puisse être transformé et franchir toutes les étapes jusqu'à sa publication.

Nous avons pu observer une continuité entre les environnements de production et de publication, avec des traces qui persistent dans le texte malgré les changements de modèles appliqués lors des transformations. Toutefois, le sens accordé à ces traces fluctue légèrement du fait de la variation dans les définitions des éléments textuels selon les modèles auxquels elles sont soumises. En finalité, le choix de tous les environnements mobilisés dans la production de l'écriture compte, car ils mobilisent des médiations différentes, offrant, selon leurs modèles textuels, des significations différentes aux caractères saisis.

Selon cette perspective, les modèles textuels sont des médiations majeures du processus de matérialisation du document au sein de l'environnement d'écriture numérique, puisque le sens et la matérialité de cette écriture émergent de leur application dans l'environnement. Leur action dans l'environnement permet d'écarter et d'extérioriser certaines propriétés de l'environnement, tandis que d'autres sont mises en avant. Les modèles textuels et les théories qu'ils véhiculent contraignent ainsi l'écriture et le sens produit à leur définition.

Toutefois, l'écriture ne peut pas être réduite à cette seule médiation. Cette réduction, adoptée dans le cadre de cette recherche, laisse de côté d'autres médiations qui, elles aussi, contribuent à produire la matérialité de l'écriture et à lui donner son épaisseur. Les écritures non créatives de Goldsmith sont un contre-exemple parfait à la production du sens issu de la formalisation des éléments textuels à travers un modèle riche (Goldsmith 2018). Bien au contraire, Goldsmith joue avec cette rigidité des environnements d'écriture et des formats pour proposer de nouvelles formes d'écriture artistiques et poétiques qui ne respectent pas nécessairement les modèles textuels implémentés dans les environnements qu'il emploie. Par exemple, lorsqu'il insère des chaînes de caractères à l'intérieur d'un fichier d'une image, il met à la fois en évidence le fait que tout objet numérique est une forme d'écriture, et à la fois le fait que le format de l'image n'est pas adapté à la saisie de texte, même si l'environnement permet ce geste. Le modèle textuel sous-jacent est alors dans l'incapacité de rendre compte d'une signification, ou plutôt, il rend compte de son incapacité à signifier, à travers sa théorie du texte,

le sens de ce que Goldsmith a saisi dans le fichier. Un nouveau sens inattendu émerge alors de cette conjoncture médiatrice.

À partir de ce contre-exemple, nous arrivons aux limites de notre recherche et des méthodes que nous avons appliquées pour la mener.

La première limite que nous rencontrons est méthodologique. Nous avons fait abstraction de plusieurs paramètres nécessaires à l'écriture afin de bien observer ce que produisent les environnements d'écriture numérique. Il y a, notamment, une partie de la dimension humaine de ce dispositif, à travers les pratiques d'écriture des utilisateurs, auxquelles nous ne nous sommes pas confrontés de manière directe. Cette dimension n'est toutefois pas totalement absente, puisque les cas d'étude sont basés sur l'observation indirecte des pratiques d'écriture des utilisateurs à partir des documents qu'ils ont produits. Ce choix s'explique d'une part par l'abondance de recherches existantes sur les pratiques d'écriture, et d'autre part, parce qu'il permet de révéler des aspects de l'environnement d'écriture qui sont parfois invisibles à travers l'analyse des pratiques ou qui échappent à la perception humaine. Il ne s'agit donc pas d'oublier cette dimension de l'écriture, seulement de la mettre de côté pour observer en détail le fonctionnement des environnements d'écriture numérique.

Une des pistes possibles pour compléter nos résultats serait d'accorder plus d'attention aux pratiques des utilisateurs (Kovacs 2023) en s'inspirant des recherches menées par Nicolas Sauret avec les acteurs des revues scientifiques (2020) et de l'étendre à la totalité du dispositif d'écriture numérique. Nous pourrions notamment mener des entretiens auprès des auteurs et des éditeurs qui utilisent ces dispositifs pour analyser leurs usages au regard de nos connaissances des environnements d'écritures numériques concernés.

Une autre limite que nous rencontrons concerne l'évolution du réseau dans lequel s'inscrivent les documents une fois qu'ils sont publiés. Nous avons vu que le processus d'éditorialisation était un processus continu qui ne prenait pas fin au moment de la publication d'un artefact. Cependant, nous avons arrêté notre analyse au moment de la publication, puisque nous nous sommes concentrés sur la production du sens à partir des environnements d'écriture numérique durant les phases de production et de préparation à la publication de l'écriture. L'arrimage à un réseau d'objets, ainsi que l'évolution de ce réseau, agissent également comme un ensemble de médiations au sein du dispositif.

Il nous semble alors pertinent de mener une étude sur ces réseaux d'objets pour compléter notre recherche. Cette étude pourrait par ailleurs être réalisée à partir de la modélisation du texte dans les documents publiés et dans les documents connexes du réseau d'objets auquel la publication serait associée, notamment pour les objets publiés sur le Web. Afin d'obtenir un premier réseau d'objets, il

serait pertinent d'utiliser des outils d'extraction de données du Web. Ces derniers permettraient de mettre en évidence les connexions entre les différents objets à partir des hyperliens contenus dans les documents. Cette première vue d'ensemble du réseau d'objets peut être complétée par des méthodes de fouille de texte et ainsi permettre une analyse fine des contenus.

Un fait intéressant et en contradiction avec l'étude que nous avons mené dans cette thèse, est que les méthodes de pré-traitement du texte préconisent souvent de se focaliser uniquement sur le contenu du document et pas sur sa structuration. En ce sens, et surtout lorsque la quantité de texte à traiter est massive, la phase de pré-traitement du texte, celle où l'on nettoie le texte, consiste à supprimer la structuration des contenus des documents, en appliquant une fonction de type *strip*, pour ne garder que ce que l'on estime être ce que l'utilisateur déclare. C'est-à-dire que la matérialité de l'écriture numérique n'est pas prise en compte dans ce type d'analyse.

Lors d'une recherche connexe menée sur les archives du Web de la BNF sur la pandémie de Covid-19, portée par Marta Severo et Sarah Gensburger, nous avons appliqué ce type d'analyse et de pré-traitement aux contenus web du corpus établi (Delannay, Severo et al. 2024). L'une des particularités du contenu des documents relevait du fait que, malgré l'usage répandu du CMS WordPress pour créer des sites web, chacun comportait une structure différente : on pouvait observer des variations de structuration d'un site à l'autre. Le pré-traitement consistait alors à supprimer tous ces éléments d'architecture de l'information pour ne garder que certaines chaînes de caractères à l'intérieur des balises.

Une contradiction apparaît entre d'un côté cette matérialité du modèle textuel qui participe à produire le sens de l'écriture, et sa suppression totale lorsque l'on analyse les contenus *a posteriori*. Puisque l'un de nos résultats de recherche révèle que les éléments structurels d'un texte sont essentiels à la production du sens, il nous paraît pertinent d'envisager comme objet d'étude ce qui, jusqu'à présent, était considéré comme du bruit à éliminer.

Au regard de l'ensemble de nos résultats et des limites exposées, nous pouvons affirmer que tous les éléments composant un environnement d'écriture numérique, y compris ceux qui paraissent insignifiants, contribuent à la matérialité des écritures numériques et au sens produit par le dispositif.

Les couches d'abstraction matérielle ne sont pas abstraites

Une notion a traversé toute la thèse en filigrane et semble maintenant devoir faire l'objet d'une critique. En effet, à chaque couche que nous avons ajouté dans le dispositif d'écriture numérique, nous avons mobilisé un concept, celui de couche d'abstraction matérielle. Ces couches d'abstraction matérielle sont généralement employées en tant que jargon technique pour signifier que l'ajout d'une surcouche logicielle permet de faire abstraction de la couche précédente, jusqu'à la couche la plus basse, celle de la machine.

La couche matérielle, le *hardware* est alors effacé par le BIOS, puis par le système d'exploitation, puis par Internet et enfin par le Web. En faisant ainsi complètement abstraction de la machine, cette notion véhicule deux idées. Il y a d'abord celle qui sous-entend que le numérique n'est pas une affaire matérielle, et qu'au contraire le logiciel, ou du moins dans sa forme la plus élevée sur cette pile de logiciels, peut faire abstraction de toutes les couches positionnées en dessous de cette dernière, et qu'en les camouflant ainsi, elles ne seraient plus significatives.

Ensuite, ce camouflage des couches inférieures oriente le regard uniquement sur la couche logicielle supérieure, puisque celles qui lui sont inférieures n'importent plus. En ce sens, il y a un fort risque d'essentialiser la couche supérieure et de ne plus considérer le reste de l'infrastructure numérique dans son ensemble, car c'est bien cet ensemble de médiations présent à tous les niveaux de l'infrastructure numérique qui rend possible la matérialisation d'un phénomène comme l'écriture.

Cette perspective du numérique et du logiciel en tant qu'abstraction de la matière apparaît alors en contradiction avec notre approche matérialiste de l'écriture. Chacun des cas d'étude que nous avons mobilisés dans cette thèse nous renvoie au contraire à cette matérialité du numérique et aux coupures agentielles que ces couches logicielles appliquent au dispositif d'écriture.

Comme nous avons pu le voir à l'occasion des différents cas d'étude, la manifestation de la distribution de l'écriture à travers les médiations du dispositif ne se joue pas uniquement dans la couche logicielle supérieure, mais bien dans l'ensemble du dispositif, jusqu'à la norme d'encodage utilisée, ainsi qu'aux différents composants matériels de l'ordinateur utilisé. L'histoire des logiciels de traitement de texte[55], notamment celle de la transition entre WordStar et WordStar 2000 racontée par Seymour Rubinstein (2006), nous a montré la nécessité

55. Voir la section III.2.

d'employer des logiciels compatibles avec différents modèles et générations d'ordinateurs au risque que les médiations nécessaires au déroulement de l'écriture ne puissent avoir lieu, brisant ainsi la capacité d'écrire en environnement numérique. D'ailleurs dans cet exemple frappant, les documents produits avec WordStar étaient incompatibles avec son successeur WordStar 2000, empêchant ainsi toute capacité à lire et écrire des documents dont les modélisations textuelles sous-jacentes étaient elles-mêmes incompatibles. Il y avait là des environnements d'écriture numérique qui, à leur niveau le plus bas, c'est-à-dire au niveau du microprocesseur, ne pouvaient être transposés de l'un à l'autre.

Les couches d'abstraction matérielle interviennent alors munies de leurs standards pour permettre cette transposition entre les différents environnements d'écriture numérique. Au-delà des normes d'encodage telles que ASCII et Unicode, cette problématique de la transposition entre différents environnements s'est également posée dans les couches les plus hautes de l'infrastructure numérique, notamment dans le courant des années 2000 pour les logiciels de traitement de texte. À cette période apparaissent deux standards XML reconnus par l'ISO en tant que normes internationales pour structurer les informations dans les logiciels de traitement de texte : il s'agit des normes OOXML créée par Microsoft et ODF créée par Sun Microsystems. Durant les années qui ont précédé l'application de ces normes, principalement les années 1990, l'une des raisons qui a donné la possibilité à Microsoft d'obtenir un monopole avec son modèle textuel (MSWord) était justement dû au fait qu'il n'y avait pas cette possibilité de transposition entre différents environnements (Yami et al. 2015). L'arrivée de ces normes (essentiellement la norme ODF) a donc été un levier à la création de nouveaux modèles textuels interopérables pour les environnements de traitement de texte, évitant ainsi l'écueil d'une vision monolithique du texte à travers l'implémentation d'une seule théorie. Toutefois, ce propos est à modérer puisque un effet de lissage de l'écriture numérique apparaît lors de l'implémentation de ces normes dans les environnements d'écriture, car il n'est alors plus possible de modéliser des éléments textuels en dehors de ce que ces normes définissent. Il s'agit là d'une des limites à la reconnaissance d'un modèle en tant que norme internationale, puisque par définition, un modèle est une réduction et ne peut pas tout exprimer.

À un niveau plus élevé que le microprocesseur dans la multitude de couches d'abstraction matérielle, les normes XML implémentées dans MSWord et LibreOffice nous montrent bien que les couches intermédiaires ne sont pas insignifiantes. Malgré le fait qu'elles soient invisibles pour l'usager, ces couches sont des éléments nécessaires à l'environnement d'écriture numérique, puisqu'elles ouvrent la possibilité d'ajouter de nouvelles médiations dans le dispositif d'écriture.

Si l'on continue encore notre exploration de la matérialité des couches intermédiaires, le cas de l'éditeur de texte Stylo est révélateur à cet égard. Nous l'avons vu à l'occasion de la section IV.4, Stylo comporte une couche d'abstraction supplémentaire par rapport aux logiciels de traitement de texte installés directement à l'intérieur du système d'exploitation. Il s'agit du Web. En tant que réseau décentralisé, le Web implique une fragmentation des informations contenues dans un document entre les différents composants de l'application sur le réseau. Les caractéristiques du Web ouvrent alors à d'autres modèles textuels, c'est-à-dire à d'autres modalités d'écriture, comme l'écriture collaborative synchrone où plusieurs utilisateurs séparés géographiquement se retrouvent à écrire en même temps sur le même document. En changeant ainsi le rapport au document, cette nouvelle couche modifie également la production de l'écriture. Cependant, cette surcouche qu'est le Web n'écrase pas pour autant les autres couches et n'en fait pas abstraction : les formats et leurs spécifications, ainsi que les normes d'encodage sont par exemple indispensables à la transformation des documents via le module d'export et le logiciel de conversion Pandoc, qui s'appuient sur ces parties du modèle textuel pour convertir les documents en d'autres formats.

À travers l'étude des modèles textuels implémentés dans ces environnements, qu'ils soient WYSIWYG, WYSIWYM, en local sous un système d'exploitation libre ou propriétaire, ou encore sur le Web, les couches d'abstraction matérielle, de la plus proche du microprocesseur jusqu'à celle au niveau de l'utilisateur, apparaissent comme des éléments constitutifs de la matérialité de l'écriture numérique. Ce sont des couches matérielles qui ne sont en rien abstraites ou négligeables du processus de production de l'écriture. Ces couches sont intercalées à l'intérieur des dispositifs techniques d'écriture et réalisent ainsi une coupure agentielle à l'intérieur de ce processus. À chaque nouvelle coupure agentielle, ce sont de nouvelles conjonctures médiatrices qui, de fait, entraînent des médiations et de nouvelles productions textuelles.

Vers d'autres théories du texte

Certaines des théories du texte abordées durant cette recherche nous montrent finalement une vision très pauvre de ce qu'elles définissent en tant que texte. Hormis certaines modélisations dans Stylo ou le Pressoir qui montrent quelques avancées, ainsi que d'autres modèles textuels que nous n'avons pas traités, par exemple ceux proposés par la TEI dont l'orientation concerne plutôt l'indexation des informations, les modèles textuels implémentés dans la plupart des environnements d'écriture ne désignent bien souvent que des éléments communs du

texte : des titres, des paragraphes, des notes de bas de page, etc. Chacun de ces éléments est en quelque sorte identifiable au premier coup d'oeil, et préserve dans ses formes numérisées une part de son héritage issu de l'imprimé.

Les formats de structuration les plus courants perpétuent ainsi l'ambivalence entre une structuration graphique et une structuration sémantique des contenus, comme le montre le balisage du paragraphe en Markdown qui, pour du texte brut, repose sur un agencement de balises discrètes – les lignes vides autour du bloc de texte – et redéfinissent le paragraphe selon une perspective graphique et imprimée.

Dans une certaine mesure, ces formats que nous manipulons au quotidien ont été établis, pour la majorité, à l'époque où la numérisation des contenus était essentielle pour rendre ces contenus plus accessibles, via le Web par exemple. Au regard de notre recherche, il ne serait alors pas incongru d'avancer le fait que ces formats que nous utilisons ont été partiellement pensé pour rendre compte des contenus des documents non numérisés, et relater ainsi les modes d'accès non numérique à ces informations via leur mise en page. Les modèles textuels, et donc les théories du texte sous-jacentes, héritent en ce sens de cette culture de l'imprimé. Nous pourrions même aller plus loin puisque, comme nous l'avons vu dans le chapitre II.4, la plupart des modèles textuels en vigueur s'appuient principalement sur le modèle OHCO de DeRose *et al.* créé à partir du langage SGML (DeRose et al. 1990). Or ce modèle SGML, pensé par Goldfarb, a été créé à partir des travaux d'un typographe, Stanley Rice, donc à partir d'une représentation et d'une théorie graphique du texte (Goldfarb 1997 ; Rice 1978). Il y a donc bien une continuité et non une rupture entre les modèles textuels imprimés et numériques. Pour reprendre le terme de McLuhan, nous avons affaire à une remédiation de la modélisation textuelle de l'imprimé à l'intérieur de la modélisation textuelle numérique.

Ainsi, l'attachement des utilisateurs au rendu graphique des documents provient d'une certaine manière de l'ambiguïté résiduelle que l'on retrouve dans les modèles textuels qui définissent les différents éléments du modèle sur une base graphique.

La question de l'écriture numérique ne concerne plus la distinction entre une littératie numérique d'une littératie de l'imprimé, puisque ces deux visions existent intrinsèquement dans les environnements numériques que nous employons. Les deux sont alors nécessaires pour écrire en environnement numérique.

Pour autant, est-il souhaitable de préserver cette ambivalence dans nos modes d'écriture numérique, notamment lorsque les artefacts sont diffusés dans des espaces numériques ? En s'en affranchissant, ne serait-il pas intéressant de structurer d'autres éléments propres à un type d'écriture donné ? À propos de l'écriture

scientifique, pour continuer avec le même type d'écriture que nous avons étudié tout au long de notre recherche, sommes-nous en mesure de modéliser une hypothèse, une problématique, une introduction ou une conclusion ? Que cette proposition soit pertinente ou non, elle soulève la question de la possibilité de modifier la modélisation d'un paragraphe et de distinguer différents paragraphes au sein d'un texte, à l'inverse des paragraphes indifférenciés de nos environnements actuels. Dans un tel exemple, la notion de paragraphe ne serait alors plus définie en référence à un objet imprimé et à une structure graphique, mais par le type d'informations qu'il contient.

Les modélisations effectuées pour les projets Stylo et le Pressoir pourraient servir de fondation à cette recherche. En poursuivant dans cette direction, il serait alors intéressant de recenser les pratiques et les besoins en écriture de certaines disciplines, puis de les modéliser. Plusieurs applications seraient alors envisageables à partir des résultats obtenus. Un premier exemple serait le développement d'environnements d'écriture très spécifiques, plus que ne l'est Stylo qui est finalement très généraliste au regard de notre proposition, allant de la production à la publication (voire à l'archivage). Une autre piste serait d'envisager la création de protocoles de versionnement, d'échange et de circulation des textes numériques sur la base de modèles textuels prévus à cet effet, contrairement aux protocoles actuellement répandus comme l'est Git pour le versionnement, alors adapté à des lignes de code informatique et non à des textes savants.

Au-delà des exemples de développements possibles, ces perspectives de recherche montrent que les modélisations textuelles et les théories qu'elles incarnent peuvent encore être affinées et enrichies, ouvrant ainsi la voie à de nouvelles formes d'écriture et de production du sens.

Finalement, l'écriture scientifique gagnerait à se doter de modèles textuels spécifiquement adaptés à ses contenus, plutôt que de s'appuyer sur des modèles principalement conçus pour le rendu graphique (affichage). Les bénéfices d'une telle approche sont nombreux : un modèle textuel adapté permettrait d'affiner la structuration et le sens des documents scientifiques. Par exemple, des éléments textuels clés comme la problématique, la méthodologie ou l'hypothèse pourraient être explicitement identifiés et mis en valeur, facilitant ainsi leur recoupement avec des éléments similaires dans d'autres publications.

Un deuxième exemple de l'emploi d'un modèle textuel précis serait de repenser le mode de citation des références bibliographiques, qui repose actuellement sur le numéro de page. Ce système montre ses limites dans le contexte des publications numériques, où la pagination peut être variable ou inexistante, entraînant des incohérences. Un modèle adapté aux éléments textuels que l'on a cité offrirait

ainsi une base plus robuste et cohérente pour les références et citations dans les textes scientifiques.

Ces exemples montrent qu'il ne suffit pas d'enrichir les contenus ni de simplement mieux les indexer pour augmenter leur interopérabilité, comme le proposent des formats tels que XML ou les technologies du web sémantique.

L'enjeu d'une modélisation plus avancée consiste à poser les fondements d'une véritable théorie du texte scientifique, capable de représenter avec précision la diversité des éléments textuels présents dans les publications scientifiques. Une telle modélisation permettrait à nos environnements d'écriture d'interpréter ces structures de manière beaucoup plus fine – une capacité encore absente aujourd'hui, mais dont la perspective ouvre des développements stimulants.

Bibliographie

Angles morts du numérique ubiquitaire : Glossaire critique et amoureux (2023), sous la dir. d'Y. Citton, M. Lechner et A. Masure (Dijon : les presses du réel, 16 jan.), 400 p.

Anonymous (2017), « Complex Singularity versus Openness », Interoperable Europe Portal (15 oct.), https://interoperable-europe.ec.europa.eu/collection/open-source-observatory-osor/document/complex-singularity-versus-openness, visité le 30 jan. 2025.

Bachimont, B. (2000), « Intelligence Artificielle et Écriture Dynamique, de La Raison Graphique à La Raison Computationnelle », in P. Fabbri et J. Petitot, *Au Nom Du Sens. Autour de l'oeuvre d'Umberto Eco* (Colloque de Cerisy ; Grasset), 290-319.

Bachimont, B. (2007*a*), « Indexation et archivage de contenus multimédias », *Documents numériques Gestion de contenu* (10 nov.). doi : 10.51257/a-v1-h7500.

Bachimont, B. (2007*b*), *Ingénierie des connaissances et des contenus : le numérique entre ontologies et documents* (Paris : Hermès science, 26 oct.), 278 p.

Bachimont, B. (2007*c*), « Nouvelles tendances applicatives : de l'indexation à l'éditorialisation », in P. Gros, *L'indexation multimédia : description et recherche automatiques* (Paris, France : Hermes science publications).

Bachimont, B. (2010*a*), « Le numérique comme support de la connaissance : entre matérialisation et interprétation », in (PUR et INRP, 30 juin), 75.

Bachimont, B. (2010*b*), *Le Sens de La Technique : Le Numérique et Le Calcul*, sous la dir. de F. Sebbah (A Présent ; Encres Marines / Les Belles Lettres), 191 p.

Bachimont, B. (2017), « L'archive et la massification des données : une nouvelle raison numérique », *La Gazette des archives*, 245/1 : 27-43. doi : 10.3406/gazar.2017.5510.

Bachimont, B. (2022*a*), « Donner du sens aux données : les ruses du numérique : Les disciplines du document face à la métis du calcul », *Interfaces numériques*, 11/2 (2 26 juill.). doi : 10.25965/interfaces-numeriques.4838.

Bachimont, B. (2022*b*), « La complexité herméneutique à l'épreuve du calcul », *Interfaces numériques*, 10/3 (3 15 fév.). doi : 10.25965/interfaces-numeriques.4686.

Bachimont, B. (2022*c*), « Théorie, Modèle, calcul et variabilité du réel », *Intelligibilité du numérique*, 3.

Ballmer, S. (2003), « Steve Ballmer Speech Transcript - Microsoft Office System Launch - Stories », Microsoft (21 oct.), https://news.microsoft.com/2003/10/21/steve-ballmer-speech-transcript-microsoft-office-system-launch/, visité le 30 jan. 2025.

Barad, K. (2007), *Meeting the Universe Halfway : Quantum Physics and the Entanglement of Matter and Meaning* (Second Printing edition, Durham : Duke University Press Books, 1er juill.), 544 p.

Barad, K. (2023), *Frankenstein, la grenouille et l'électron : Les sciences et la performativité queer de la nature*, trad. par L. Balice et C. Degoutin (Le Pré Saint Gervais : Asinamali, 22 sept.), 304 p.

Bardiot, C., et al. (2023), « Éditorial. Faire revue », *Humanités numériques*, 8 (déc.). doi : 10.4000/revuehn.3886.

Bergin, T. J. (2006*a*), « The Origins of Word Processing Software for Personal Computers : 1976-1985 », *IEEE Annals of the History of Computing*, 28/4 (oct.) : 32-47. doi : 10.1109/MAHC.2006.76.

Bergin, T. J. (2006*b*), « The Proliferation and Consolidation of Word Processing Software : 1985-1995 », *IEEE Annals of the History of Computing*, 28/4 (oct.) : 48-63. doi : 10.1109/MAHC.2006.77.

Berners-Lee, T. (1999), *Weaving the Web : The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor* (New York : HarperOne), 240 p.

Blanc, J., et Haute, L. (2018), « Technologies de l'édition numérique », *Sciences du Design*, 8/2 : 11-7. doi : 10.3917/sdd.008.0011.

Blanc, M., et al. (2005), *Rapport final de l'action spécifique « Document et Organisation »* (RTP 33, mars), 134.

Bolter, J. D. (1989), « Beyond Word Processing : The Computer as a New Writing Space », *Language & Communication*, Special Issue Transformations of the Word, 9/2 (1er jan.) : 129-42. doi : 10.1016/0271-5309(89)90014-1.

Bolter, J. D., et Grusin, R. (1998), *Remediation : Understanding New Media* (Cambridge, MA, USA : MIT Press, 15 oct.), 312 p.

Bonhomme, S. (1998), « Transformation de documents structurés, une combinaison des approches explicite et automatique », thèse de doct. (Université Joseph-Fourier - Grenoble I, 21 déc.).

Bord, B. (2025), « Éloge de La Moulinette », Je Hais Le Printemps (23 fév.), https://jehaisleprintemps.net/blog/fr/2025/02/23/eloge-de-la-moulinette/, visité le 23 mai 2025.

Borges, J. L. (1994), *Histoire de l'infamie, Histoire de l'éternité* (Paris : 10/18).

Bouchardon, S. (2014), « L'écriture Numérique : Objet de Recherche et d'enseignement », *Les Cahiers de la SFSIC* (juin).

Bouchardon, S., et Cailleau, I. (2018), « Milieu numérique et "lettrés" du numérique », *Le français aujourd'hui*, 200/1 (29 mars) : 117-26. doi : 10.3917/lfa.200.0117.

Brauer, B., et al. (2005), *Open Document Format for Office Applications (OpenDocument)*, version 1 (1er mai) (approved).

Bray, T., Paoli, J. et Sperberg-McQueen, C. M. (1997), *Extensible Markup Language (XML)*, 7 août.

Briet, S. (1951), *Qu'est-Ce Que La Documentation ?* (Editions Documentaires Industrielles et Techniques, Paris).

Broudoux, É. (2022), *Éditorialisation et autorité : Dispositifs info-communicationnels numériques* (Louvain-la-Neuve : De Boeck Supérieur), 224 p.

Broudoux, É., Bootz, P., et al. (2005), « Auctorialité : Production, Réception et Publication de Documents Numériques » (1er oct.).

Broudoux, É., Chartron, G. et Chaudiron, S. (2013), « L'architecture de l'information : quelle réalité conceptuelle ? », *Études de communication. langages, information, médiations*, 41 (41 1er déc.) : 13-30. doi : 10.4000/edc.5379.

Buckland, M. K. (1997), « What Is a "Document" ? », *Journal of the Association for Information Science and Technology*, 48/9 (1er sept.) : 804-9. doi : 10.1002/(sici)1097-4571(199709)48:9<804::aid-asi5>3.0.co;2-v.

Burdick, A., et al. (2012), *Digital_Humanities* (Cambridge, MA, USA : MIT Press, 16 nov.), 152 p.

Campbell-Kelly, M. (2001), « Not Only Microsoft : The Maturing of the Personal Computer Software Industry, 1982–1995 », *Business History Review*, 75/1 (avr.) : 103-45. doi : 10.2307/3116558.

Candel, É., et al. (2019), « E : ÉCRITURE / ÉCRITURE / ÉCRITURE / ÉCRITURE (très ordinaire) / ÉDITEUR (edere, editor...) / ÉDITEUR (revue d') / ÉDITEURS (passage en revue) / EMPATTEMENT / ENGAGEMENT / ENTRETIEN / ÉPREUVES / ÉVALUATION / ÉVALUATION », *Communication & langages*, 200/2 (11 sept.) : 103-38. doi : 10.3917/comla1.200.0103.

Chaudiron, S. (2007), « Technologies linguistiques et modes de représentation de l'information textuelle », *Documentaliste-Sciences de l'Information*, 44/1 : 30-9. doi : 10.3917/docsi.441.0030.

Chikofsky, E. J., et Cross, J. H. (1990), « Reverse Engineering and Design Recovery : A Taxonomy », *IEEE Software*, 7/1 (jan.) : 13-7. doi : 10.1109/52.43044.

Christin, A.-M. (1999), « Les Origines de l'écriture : Image, Signe, Trace », *Le débat*, 4 : 28-36.

Christin, A.-M. (2009), *L'Image écrite ou La déraison graphique* (Enlarged édition, Paris : Flammarion, 16 sept.), 464 p.

Chun, W. H. K. (2023), *Codes, races, climat, habitudes : Implications sociales de la numérisation* (Dijon : les presses du réel, 26 mai), 192 p.

Contribution numérique : cultures et savoirs (2024), sous la dir. de M. Severo et R. Delannay (Les ateliers de sens public, 15 juin).

Coombs, J. H., Renear, A. H. et DeRose, S. (1987), « Markup Systems and the Future of Scholarly Text Processing », *Communications of the ACM* 30/11 (1er nov.) : 933-47. doi : 10.1145/32206.32209.

Cotte, D. (2004), « Le concept de “document numérique” », *Communication & Langages*, 140/1 : 31-41. doi : 10.3406/colan.2004.3265.

Couture, S. (2012), « L'écriture collective du code source informatique », *Revue d'anthropologie des connaissances*, 6/1 (1 1er mars). doi : 10.3917/rac.015.0061.

Crozat, S. (2012), « Chaînes Éditoriales et Rééditorialisation de Contenus Numériques », in L. Calderan et al. (éd.), *Le Document Numérique à l'heure Du Web* (Sciences et Techniques de l'information ; ADBS), 179-220.

Crozat, S. (2016), « Écrire avec une machine à calculer, écrire pour une machine à calculer », *I2D - Information, donnees documents*, 53/2 (5 juill.) : 62-4. doi : 10.3917/i2d.162.0062.

Crozat, S., et al. (2011), « Éléments pour une théorie opérationnelle de l'écriture numérique », *Document numérique*, 14/3 : 9-33.

Debouy, E. (2022), « Stylo, un éditeur pour les sciences humaines et sociales », *RIDE – a review journal for digital editions and resources*, 15. doi : 10.18716/RIDE.A.15.3.

Dehut, J. (2023), « Rhétorique et Informatique : à La Source Des Humanités Numériques », thèse de doct. (Normandie, 11 déc.).

Delannay, R., Fauchié, A. et Vitali-Rosati, M. (2023), « Stylo en 2023 : de nouvelles fonctionnalités pour l'écriture et l'édition scientifique », in *Humanistica 2023*, Association francophone des humanités numériques (Poster ; Genève, Switzerland, juin).

Delannay, R., Severo, M. et Gabrysiak, L. (2024), « Mémoires du Covid-19 et archives du Web : pour une analyse quantitative du dépôt légal de la BNF », *Humanités numériques*, 9 (1er juin). doi : 10.4000/11wmx.

Denis, J., et Pontille, D. (2012), « Travailleurs de l'écrit, matières de l'information », *Revue d'anthropologie des connaissances*, 6/1 (1 1er mars). doi : 10.3917/rac.015.0001.

DeRose, S. (1999), « XML and the TEI », *Computers and the Humanities*, 33/1 (1er avr.) : 11-30. doi : 10.1023/A:1001771114509.

DeRose, S., et al. (1990), « What Is Text, Really ? », *Journal of Computing in Higher Education*, 1 (déc.) : 3-26. doi : 10.1007/BF02941632.

Despoix, P. (2012), « Traverser dix ans d'Intermédialités », *Intermédialités : histoire et théorie des arts, des lettres et des techniques / Intermediality : History and Theory of the Arts, Literature and Technologies*, 20 : 7-10. doi : 10.7202/1023521ar.

Doueihi, M. (2011), *La Grande Conversion numérique : suivi de Rêveries d'un promeneur numérique* (Paris : Points, 15 sept.), 352 p.

Dürkop, A. (2020), « Independent Publishing Based on the Values of Open Science », Generation Research, https://genr.eu/wp/independent-publishing-based-on-the-values-of-open-science/, visité le 4 juin 2025.

ECMA-376 (2006), *Office Open XML File Formats : Part 3*, version 1 (déc.) (Published).

ECMA-376 (2006), *Office Open XML File Formats : Part 4*, version 1 (déc.) (Published).

ECMA-376 (2012), *Office Open XML File Formats : Part 1*, version 4 (déc.) (Published).

ECMA-376 (2012), *Office Open XML File Formats : Part 2*, version 4 (déc.) (Published).

ECMA-376 (2012), *Office Open XML File Formats : Part 3*, version 4 (déc.) (Published).

ECMA-376 (2012), *Office Open XML File Formats : Part 4*, version 4 (déc.) (Published).

ECMA-376 : (2006), *Office Open XML File Formats : Part 1*, version 1 (déc.) (Published).

ECMA-376 : (2006), *Office Open XML File Formats : Part 2*, version 1 (déc.) (Published).

ECMA-404 : (2017), *The JSON Data Interchange Syntax*, version 2 (déc.) (Published).

Engelbart, D. (1970), *X-Y Position Indicator for a Display System*, brevet 3541541.

Epron, B., et Vitali-Rosati, M. (2018), *L'édition à l'ère numérique* (Repères ; Paris : La Découverte).

Fauchié, A. (2024), « Fabriquer des éditions, éditer des fabriques : reconfiguration des processus techniques éditoriaux et nouveaux modèles épistémologiques », thèse de doct. (Université de Montréal, 19 mars).

Fauchié, A., et Audin, Y. (2023), « The Importance of Single Source Publishing in Scientific Publishing », *Digital Studies / Le champ numérique*, Special DSCN Collection #9 (Special DSCN Collection #9 17 oct.). doi : 10.16995/dscn.9655.

Fauchié, A., Delannay, R., et al. (2023), « Exploring New (Digital) Publishing Practices with *Le Pressoir* », *Pop ! Public. Open. Participatory*, 5 (sept.).

Fayet-Scribe, S. (2000), *Histoire de la documentation en France : Culture, science et technologie de l'information, 1895-1937* (CNRS Éditions), 328 p.

Ford, B. (2004), « Parsing Expression Grammars : A Recognition-Based Syntactic Foundation », in *Proceedings of the 31st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL04 : The 31st Annual ACM Symposium on Principles of Programming Languages (Venice Italy : ACM, jan.), 111-22. doi : 10.1145/964001.964011.

Friesen, N. (2018), *Media Transatlantic : Developments in Media and Communication Studies between North American and German-speaking Europe* (Softcover reprint of the original 1st ed. 2016 édition, Cham : Springer), 224 p.

Galloway, A. R. (2004), *Protocol. How Control Exists after Decentralization* (Leonardo ; Cambridge : MIT Press), 260 p.

Gardey, D. (2001), *La dactylographe et l'expéditionnaire : histoire des employés de bureau, 1890-1930* (Paris, France : Belin), 335 p.

Genette, G. (1979), *Introduction à l'architexte* (Paris : Seuil), 102 p.

Gessler, N. (1998), « Skeuomorphs and Cultural Algorithms », in *Evolutionary Programming VII*, sous la dir. de V. W. Porto et al. (Berlin, Heidelberg : Springer), 229-38. doi : 10.1007/BFb0040776.

Ghitalla, F. (2000), « L'espace du document numérique », *Communication & Langages*, 126/1 : 74-84. doi : 10.3406/colan.2000.3044.

Godfrey, M., et Hendry, D. (1993), « The Computer as von Neumann Planned It », *IEEE Annals of the History of Computing*, 15/1 : 11-21. doi : 10.1109/85.194088.

Goldfarb, C. (1996), « The Roots of SGML – A Personal Recollection », in *Proceedings of the SGML '96 Conference. Celebrating a Decade of SGML*, Graphic Communications Association (GCA), 19-24.

Goldfarb, C. (1997), « SGML : The Reason Why and the First Published Hint », *Journal of the American Society for Information Science*, 48/7 : 656-61. doi : 10.1002/(SICI)1097-4571(199707)48:7<656::AID-ASI13>3.0.CO;2-T.

Goldsmith, K. (2018), *L'écriture sans écriture : du langage à l'âge numérique*, trad. par F. Bon (Paris : Jean Boîte Editions), 192 p.

Goody, J. (1979), *La Raison graphique : la domestication de la pensée sauvage*, trad. par J. Bazin et A. Bensa (Sens commun, Paris : Minuit), 272 p.

Goossens, M. (1995), « Introduction Pratique à SGML », *Cahiers GUTenberg*, 19 : 27-58.

Guyot, B. (2004), « Sciences de l'information et Activité Professionnelle », *Hermès, La Revue - Cognition, communication, politique*, 38 (6 mai) : 38-45. doi : 10.4267/2042/9422.

Habib, A. (2021), « ... et des poussières. Autour de Zen for Film (1962-1964...) de Nam June Paik », *Communication & langages*, 208–209/2 (4 oct.) : 95-113. doi : 10.3917/comla1.208.0095.

Hadot, P. (1992), *La Citadelle Intérieure. Introduction aux Pensées de Marc Aurèle* (Paris : Fayard, 4 nov.), 386 p.

Hadot, P. (1995), *Qu'est-ce que la philosophie antique ?* (Paris : Folio, 14 nov.), 455 p.

Hadot, P. (2002), *Exercices spirituels et philosophie antique* (Nouv éd rev. et augm édition, Paris : Albin Michel, 4 nov.), 416 p.

Hanson, M. A., et al. (2024), « The strain on scientific publishing », *Quantitative Science Studies*, 5/4 : 823-43. doi : 10.1162/qss_a_00327.

Havelock, E. A. (1967), *Preface to Plato* (New York : Grosset & Dunlap), 350 p.

Havelock, E. A. (1989), « Orality and Literacy, an Overview », *Language & Communication*, Special Issue Transformations of the Word, 9/2 (1er jan.) : 87-98. doi : 10.1016/0271-5309(89)90011-6.

Hayles, N. K. (2005), *My Mother Was A Computer : Digital Subjects And Literary Texts* (2nd ed. édition, Chicago : University of Chicago Press, 15 oct.), 288 p.

Hayles, N. K. (2015), *Parole, écriture, code* (Dijon : les presses du réel, 3 juin), 88 p.

Heer, J., et al. (2023), « Living Papers : A Language Toolkit for Augmented Scholarly Communication », in *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology* (UIST '23 ; New York, NY, USA : Association for Computing Machinery, 29 oct.), 1-13. doi : 10.1145/3586183.3606791.

Herrenschmidt, C. (2023), *Les trois écritures : Langue, nombre, code* (Paris : FOLIO), 624 p.

Hesse, C. (1990), « Enlightenment Epistemology and the Laws of Authorship in Revolutionary France, 1777-1793 », *Representations*, 30 : 109-37. doi : 10.2307/2928448, JSTOR : 2928448.

Higgins, D. (1966), « Statement on Intermedia », https://www.artpool.hu/Fluxus/Higgins/intermedia2.html, visité le 21 nov. 2024.

Hudon, M., et Hadi, W. M. E. (2012), « Introduction », *Études de communication. langages, information, médiations*, 39 (39 1er déc.) : 9-14. doi : 10.4000/edc.3893.

Hutchins, E. (1995), *Cognition in the Wild* (Cambridge, Mass. : MIT Press), 402 p.

Hyysalo, S., Pollock, N. et Williams, R. A. (2019), « Method Matters in the Social Study of Technology : Investigating the Biographies of Artifacts and Practices », *Science & Technology Studies*, 32/3 (3 14 sept.) : 2-25. doi : 10.23987/sts.65532.

Ingold, T. (2013), *Une Brève Histoire Des Lignes* (Zones sensibles), 256 p.

Innis, H. A. (1964), *The Bias of Communication* (Toronto : University of Toronto Press).

ISO 8879:1986 (1986), *Information processing — Text and office systems — Standard Generalized Markup Language (SGML)* (oct.) (Published).

ISO/IEC 26300:2006 (2006), *Open Document Format for Office Applications*, version 1 (1er déc.) (Published).

ISO/IEC 29500-1:2008 (2008), *Document Description and Processing Languages — Office Open XML File Formats* (nov.) (Withdraw).

Jeanneret, Y. (2011), *Y a t-il (vraiment) des technologies de l'information ?* (Savoirs Mieux ; Villeneuve d'Ascq : Presses universitaires du Septentrion), 200 p.

Jeanneret, Y., et Souchier, E. (2005), « L'énonciation éditoriale dans les écrits d'écran », *Communication & Langages*, 145/1 : 3-15. doi : 10.3406/colan.2005.3351.

Jones, J. (2003), « Abstract Syntax Tree Implementation Idioms », in *Proceedings of the 10th Conference on Pattern Languages of Programs (PLoP2003)*.

Kay, A., et Goldberg, A. (1977), « Personal Dynamic Media », *Computer*, 10/3 (1er mars) : 31-41. doi : 10.1109/C-M.1977.217672.

Kembellec, G. (2019), « Semantic publishing, la sémantique dans la sémiotique des codes sources d'écrits d'écran scientifiques », *Les Enjeux de l'information et de la communication*, 202/2 : 55-72. doi : 10.3917/enic.027.0055.

Kembellec, G. (2020), « L'érudition numérique palimpseste », *Hermes, La Revue*, n° 87/2 : 145-58.

Kembellec, G., et Scopsi, C. (2012), « Étude comparée de quatre logiciels de gestion de références bibliographiques libres ou gratuits », *Documentation et bibliothèques*, 58/4 : 187-97. doi : 10.7202/1028836ar.

Kirschenbaum, M. G. (2016), *Track Changes : A Literary History of Word Processing* (Illustrated edition, Cambridge, Massachusetts : Belknap Press : An Imprint of Harvard University Press, 2 mai), 368 p.

Kittler, F. (2015), *Mode protégé* (Dijon : les presses du réel), 87 p.

Kittler, F. (2018), *Gramophone, Film, Typewriter* (Dijon : les presses du réel, 1er jan.).

Knuth, D. (1984*a*), « Literate Programming », *The Computer Journal*, 27/2 (1er jan.) : 97-111. doi : 10.1093/comjnl/27.2.97.

Knuth, D. (1984*b*), *The TeXbook* (SPI edition, Reading, Mass. : Addison-Wesley Professional, 1er jan.), 496 p.

Knuth, D. (1997), *Art of Computer Programming, The : Volume 1 : Fundamental Algorithms, 3rd Edition* (3rd, Addison-Wesley Professional., 7 juill.), 672 p.

Kovacs, S. (2023), « Analyses dispositives en Sciences de l'information et de la communication : le cas de la culture informationnelle », *Approches Théoriques en Information-Communication (ATIC)*, 7/2 : 67-80. doi : 10.3917/atic.007.0067.

Krämer, S. (2016), « The Messenger as a Model in Media Theory. Reflections on the Philosophical Dimensions of Theorizing Media », in N. Friesen (éd.), *Media Transatlantic : Developments in Media and Communication Studies between North American and German-speaking Europe* (Cham : Springer International Publishing), 197-213. doi : 10.1007/978-3-319-28489-7_11.

Krewinkel, A., et Winkler, R. (2017), « Formatting Open Science : Agilely Creating Multiple Document Formats for Academic Manuscripts with Pandoc Scholar », *PeerJ Computer Science*, 3 (8 mai) : e112. doi : 10.7717/peerj-cs.112.

Kruse, O., et Rapp, C. (2023), « Word Processing Software : The Rise of MS Word », in O. Kruse et al. (éd.), *Digital Writing Technologies in Higher Education : Theory, Research, and Practice* (Cham : Springer International Publishing), 15-32. doi : 10.1007/978-3-031-36033-6_2.

Lampson, B., et al. (1979), *Bravo Manual*.

Larroche, V. (2018), *Le dispositif : Un concept pour les sciences de l'information et de la communication* (ISTE Group, 1er sept.), 213 p., Google Books : noxxDwAAQBAJ.

Le Crosnier, H. (2006), « Architecture Informatique et Économie Du Document Numérique : Deux Questions Étroitement Liées » (20 sept.).

Le Crosnier, H. (2007), « Réseau, bibliothèques et documents numériques : architecture informatique et construction sociale », thesis (Université de Caen, 7 déc.).

Le Crosnier, H. (2010), « La Gouvernance Du Document : Référence Au Cadre Des Biens Communs de La Connaissance », in *Enjeux Politiques Du Document Numérique : Actes de La Troisième Conférence Document Numérique et Société* (France), 139-53.

Le Deuff, O. (2012), « Littératies informationnelles, médiatiques et numériques : de la concurrence à la convergence ? », *Études de communication. langages, information, médiations*, 38 (38 30 juin) : 131-47. doi : 10.4000/edc.3411.

Le Crosnier, H., et al. (2011), « Vers Les " Communs de La Connaissance " », *I2D – Information, données & documents*, 48/3 : 48-59. doi : 10.3917/docsi.483.0048.

Leroi-Gourhan, A. (1964), *Le Geste et la Parole*, i, 2 t. (Illustrated édition, Paris : Albin Michel, 30 mars 2022), 400 p.

Leroi-Gourhan, A. (1965), *Le Geste et la Parole*, ii, 2 t. (Paris : Albin Michel, 30 mars 2022), 368 p.

Levy, D. M. (1994), « Fixed or Fluid ? », *Proceedings of the 1994 ACM European conference on Hypermedia technology - ECHT '94*.

Lévy, P. (2023), « Calculer la sémantique avec le langage IEML », *Humanités numériques*. doi : https://doi.org/10.4000/revuehn.3836.

Licklider, J. C. R. (1960), « Man-Computer Symbiosis », *IRE Transactions on Human Factors in Electronics*, HFE-1/1 (mars) : 4-11. doi : 10.1109/THFE2.1960.4503259.

Lipsyc, C., et Ihadjadene, M. (2013), « Architecture de l'information et éditorialisation », *Études de communication. langages, information, médiations*, 41 (41 1er déc.) : 103-18. doi : 10.4000/edc.5406.

Lund, N. W., et Buckland, M. K. (2008), « Document, Documentation, and the Document Academy : Introduction », *Archival Science*, 8/3 (1er sept.) : 161-4. doi : 10.1007/s10502-009-9076-3.

Mailloux, P. (1980), « La définition de l'édition et les AACR 2 », *Documentation et bibliothèques*, 26/1 : 29-34. doi : 10.7202/1054264ar.

Marc Aurèle (2015), *Pensées pour moi-même*, trad. par A.-I. Trannoy (Paris : Les Belles Lettres, 20 nov.), 224 p.

Maxwell, J. W. (2022), « The Care-ful Reviewer : Peer Review as If People Mattered », *Pop ! Public. Open. Participatory*, 4 (9 nov.).

McLuhan, M. (1967), *Galaxie Gutenberg*, trad. par J. Pare (Paris : CNRS, 2017).

McLuhan, M. (1977), *Pour comprendre les médias* (Paris : Points, 1er mars).

McLuhan, M., et Nevitt, B. (1973), « The Argument : Causality in the Electric World », *Technology and Culture*, 14/1 (jan.) : 1-18. doi : 10.2307/3102728.

McWilliams, P. (1983), *The Word Processing Book : A Short Course in Computer Literacy* (Los Angeles, Calif. : Prelude Press ; New York, N.Y. : Distributed by Ballantine Books), 332 p.

Mersch, D. (2018), *Théorie des médias - Une introduction* (Dijon : les presses du réel, 1er jan.).

Merzeau, L. (2013*a*), « Éditorialisation Collaborative d'un Événement », *Communication et organisation*, 43 : 105-22. doi : 10.4000/communicationorganisation.4158.

Merzeau, L. (2013*b*), « L'intelligence des traces », *Intellectica. Revue de l'Association pour la Recherche Cognitive*, 59/1 : 115-35. doi : 10.3406/intel.2013.1088.

Merzeau, L. (2016), « Le profil : une rhétorique dispositive », *Itinéraires. Littérature, textes, cultures*, 2015–3 (2015–3 21 juin). doi : 10.4000/itineraires.3056.

Merzeau, L. (2017), « Mémoire partagée », in M. Cornu, F. Orsi et J. Rochfeld, *Dictionnaire des biens communs* (Paris : Presses universitaires de France, 20 août).

Mille, A. (2014), « D'Internet au web », in M. E. Sinatra et M. Vitali-Rosati (éd.), *Pratiques de l'édition numérique* (Parcours numériques ; Montréal : Presses de l'Université de Montréal, 1er mars), 31-48.

Mollier, J.-Y. (2015), *Une autre histoire de l'édition française* (Paris : La fabrique éditions), 429 p.

Monjour, S., et Sauret, N. (2021), « Pour une gittérature », *XXI/XX Reconnaissances littéraires*, 2 : pp. 237-252. doi : 10.48611/isbn.978-2-406-12363-7.p.0237.

Mounier, P. (2010), « Manifeste des Digital Humanities », *Journal des anthropologues. Association française des anthropologues*, 122–123 (122–123 1er déc.) : 447-52. doi : 10.4000/jda.3652.

Mounier, P. (2020), « Les plateformes de diffusion numérique, un enjeu politique », in É. Anheim et L. Foraison, *L'édition en sciences humaines et sociales : Enjeux et défis* (Cas de figure ; Paris : Éditions de l'École des hautes études en sciences sociales), 81-8. doi : 10.4000/books.editionsehess.28843.

Mourat, R. de (2020), « Le vacillement des formats : matérialité, écriture et enquête : le design des publications en Sciences Humaines et Sociales », thèse de doct. (Université Rennes 2, 18 sept.).

Müller, J. (2000), « L'intermédialité, une nouvelle approche interdisciplinaire : perspectives théoriques et pratiques à l'exemple de la vision de la télévision », *Cinémas : revue d'études cinématographiques / Cinémas : Journal of Film Studies*, 10/2–3 : 105-34. doi : 10.7202/024818ar.

Ong, W. J. (2014), *Oralité et écriture : La technologie de la parole* (1ère édition, Paris : Les Belles Lettres, 20 mars), 240 p.

Otlet, P. (1934), *Le livre sur le livre : Traité de documentation* (Bruxelles : Les Impressions nouvelles, 2015).

Paveau, M.-A. (2012), « Activités langagières et technologie discursive. L'exemple de Twitter », La pensée du discours (27 fév.), https://penseedudiscours.hypotheses.org/8338, visité le 18 sept. 2024.

Paveau, M.-A. (2013), « Technodiscursivités Natives Sur Twitter. Une Écologie Du Discours Numérique », *Epistémè*, 9 (juill.) : 139-76.

Paveau, M.-A. (2015), « Ce qui s'écrit dans les univers numériques », *Itinéraires. Littérature, textes, cultures*, 2014–1 (2014–1 1er fév.). doi : 10.4000/itineraires.2313.

Payeur, C., et Zacklad, M. (2007), « Dispositifs d'articulation entre espaces physique et virtuel pour accéder à l'offre de presse », *Études de communication. langages, information, médiations*, 30 (30 1er oct.) : 39-53. doi : 10.4000/edc.446.

Pédauque, R. T. (2003), « Document : Forme, Signe et Médium, Les Re-Formulations Du Numérique » (8 juill.).

Pédauque, R. T. (2006), *Le document à la lumière du numérique* (Caen, France : C&F éditions).

Pédauque, R. T. (2007), *La redocumentarisation du monde* (Toulouse : Editions Cépaduès, 11 jan.), 213 p.

Perret, A. (2022), « De l'héritage épistémologique de Paul Otlet à une théorie relationnelle de l'organisation des connaissances », thèse de doct. (Université Michel de Montaigne - Bordeaux III, 2 déc.).

Pierazzo, E. (2016), *Digital Scholarly Editing : Theories, Models and Methods* (London : Routledge, 16 mars), 252 p.

Platon (2007), *Phèdre* (Paris : Le Livre de Poche), 320 p.

Poirier, P., et Genêt, P. (2014), « La fonction éditoriale et ses défis », in M. E. Sinatra et M. Vitali-Rosati (éd.), *Pratiques de l'édition numérique* (Parcours numériques ; Montréal : Presses de l'Université de Montréal), 15-29. doi : 10.4000/books.pum.312.

Pratiques de l'édition numérique (2014), sous la dir. de M. E. Sinatra et M. Vitali-Rosati (Parcours numériques ; Montréal : Presses de l'Université de Montréal), 224 p.

Quintas, P., et Demaid, A. (2000), « Reverse Knowledge Engineering : A Methodology for Knowledge Management », in *Proceedings of the 2000 IEEE International Conference on Management of Innovation and Technology. ICMIT 2000. 'Management in the 21st Century' (Cat. No.00EX457)*, ii (Singapore : IEEE), 702-7. doi : 10.1109/ICMIT.2000.916785.

Ramsay, S., et Rockwell, G. (2012), « Developing Things : Notes toward an Epistemology of Building in the Digital Humanities », in M. K. Gold (éd.), *Debates in the Digital Humanities* (NED - New edition, University of Minnesota Press), 75-84, JSTOR : 10.5749/j.ctttv8hq.8.

Ranganathan, S. R. (1963), *Documentation and its facets* (London : Asia Publishing House).

Référentiel général d'interopérabilité (RGI) (2009), 12 mai.

Rice, S. (1978), *Book Design : Text Format Models* (New York : R.R. Bowker Co.), 244 p.

Rice, S. (1980), « On Publishing », *ACM SIGOA Newsletter*, 1/3 (1er juill.) : 5-7. doi : 10.1145/1017923.1017925.

Roig, J., et Ribera, M. (2021), « Implementation of the OOXML Standard since Its Approval until Today », in *Proceedings of the 9th International Conference on Software Development and Technologies for Enhancing Accessibility and Fighting Info-exclusion* (DSAI '20 ; New York, NY, USA : Association for Computing Machinery), 129-34. doi : 10.1145/3439231.3440607.

Ronconi, F. (2022), « Écrire par schédê, schédia, schédiasmata et schédaria », *Genesis. Manuscrits – Recherche – Invention*, 55 (55 1er déc.) : 45-56. doi : 10.4000/genesis.7287.

Rose, M. (1994), *Authors and Owners : The Invention of Copyright* (Cambridge, Mass. : Harvard University Press), 176 p.

Rubinstein, S. (2006), « Recollections : The Rise and Fall of WordStar », *IEEE Annals of the History of Computing*, 28/4 (oct.) : 64-72. doi : 10.1109/MAHC.2006.69.

Salaün, J.-M. (2004), « Chronique inachevée d'une réflexion collective sur le document », *Communication et langages*, 140/1 : 9-17. doi : 10.3406/colan.2004.3263.

Sapiro, G. (2014), « Droit et histoire de la littérature : la construction de la notion d'auteur », *Revue d'histoire du XIXe siècle. Société d'histoire de la révolution de 1848 et des révolutions du XIXe siècle*, 48 (1er juin) : 107-22. doi : 10.4000/rh19.4660.

Sauret, N. (2020), « De La Revue Au Collectif : La Conversation Comme Dispositif d'éditorialisation Des Communautés Savantes En Lettres et Sciences Humaines », thèse de doct. (Paris 10, 20 nov.).

Schrader, A. (2010), « Responding to Pfiesteria Piscicida (the Fish Killer) : Phantomatic Ontologies, Indeterminacy, and Responsibility in Toxic Microbiology », *Social Studies of Science*, 40/2 (1er avr.) : 275-306. doi : 10.1177/0306312709344902.

Schrayer, M. (1977), *The Electric Pencil Word Processor Operator's Manual.*

Sénèque (s. d.), « Lettres à Lucilius », http://agoraclass.fltr.ucl.ac.be/concordances/sen_luciliusX/lecture/3.htm, visité le 4 sept. 2023.

Severo, M. (2013), « La Liste du patrimoine mondial : une approche par le système d'information », *Culture & Musées*, 22 : 47-70. doi : 10.3406/pumus.2013.1751.

Shannon, C. (1948), « A Mathematical Theory of Communication », *The Bell System Technical Journal*, 27/3 (juill.) : 379-423. doi : 10.1002/j.1538-7305.1948.tb01338.x.

Silvast, A., et Virtanen, M. J. (2023), « On Theory–Methods Packages in Science and Technology Studies », *Science, Technology, & Human Values*, 48/1 (1er jan.) : 167-89. doi : 10.1177/01622439211040241.

Simonyi, C. (2008), *Oral History of Charles Simonyi*, Interview transcript, Computer History Museum, Reference number : X4428.2008, Interviewed by Grady Booch, February 6, 2008, Mountain View, California. Transcript available at Computer History Museum (Ref : X4428.2008) and online : https://archive.computerhistory.org/resources/access/text/2015/06/102702232-05-01-acc.pdf.

Sinatra, M. E., et Vitali-Rosati, M. (2014), « Introduction », in idem (éd.), *Pratiques de l'édition numérique* (Parcours numériques ; Montréal : Presses de l'Université de Montréal), 7-11. doi : 10.4000/books.pum.308.

Souchier, E. (1998), « L'image du texte pour une théorie de l'énonciation éditoriale », *Les cahiers de médiologie*, 6/2 : 137-45. doi : 10.3917/cdm.006.0137.

Souchier, E. (2012), « La "lettrure" à l'écran. Lire & écrire au regard des médias informatisés », *Communication & langages*, 174/4 : 85-108. doi : 10.4074/S0336150012014068.

Souchier, E., Candel, É. et Gomez-Mejia, G. (2019), *Le numérique comme écriture. Théories et méthodes d'analyse* (Codex ; Paris : Armand Colin), 358 p.

Souchier, E., et Jeanneret, Y. (1999), « Pour une poétique de l'écrit d'écran », 6 : 97.

Swaine, M., et Freiberger, P. (2000), *Fire in the Valley : The Making of The Personal Computer* (2nd edition, New York : McGraw-Hill, 29 nov.), 463 p.

Tadier, E. (2021), « "L'intermédialité montréalaise, ou l'art de penser la relation" », *Communication langages*, 208209/2 (4 oct.) : 9-25. doi : 10.3917/comla1.208.0009.

Tadier, E., et Méchoulan, É. (2021), « Tentative d'épuisement de l'intermédialité : entretien avec Éric Méchoulan », *Communication langages*, 208209/2 (4 oct.) : 27-49.

Team, O. (2024), « Lodel 2.0 est déployé sur OpenEdition Books », L'Édition électronique ouverte (30 avr.), https://leo.hypotheses.org/21740, visité le 30 mai 2025.

The Unicode Standard : Core Specification (2024), version 16 (10 sept.) (Published).

Tufte, E. (2003), *The Cognitive Style of PowerPoint* (Cheshire : Graphics Press), 32 p.

Turing, A. (1936), « On Computable Numbers, with an Application to the Entscheidungsproblem », *Proceedings of the London Mathematical Society*, s2-42/1 : 230-65. doi : 10.1112/plms/s2-42.1.230.

Vallée, J.-F. (2011), « L'image globale : la pensée électrique de Marshall McLuhan », in O. Asselin, S. Mariniello et A. Oberhuber (éd.), *L'ère électrique. The Electric Age* (Transferts culturels | Cultural Transfers ; Ottawa : Les Presses de l'Université d'Ottawa | University of Ottawa Press), 85-112.

Vanier, P. (1995), « Manifestes médiologiques », *Études de communication. langages, information, médiations*, 16 (16) : 195-9. doi : 10.4000/edc.2507.

Vitali-Rosati, M. (2014), « Les revues littéraires en ligne : entre éditorialisation et réseaux d'intelligences », *Études françaises*, 50/3 : 83-104. doi : 10.7202/1027191ar.

Vitali-Rosati, M. (2015*a*), « An Editor for Academic Papers (Xml, Html, Md, TeX, Pdf and If You Really Need It Rtf) », Culture numérique. Pour une philosophie du numérique (16 mars), http://blog.sens-public.org/marcellovitalirosati/an-editor-for-academic-papers-xml-html-md-tex-pdf-and-if-you-really-need-it-rtf/, visité le 17 juin 2025.

Vitali-Rosati, M. (2015*b*), « Éditorialisation : état de la recherche », Culture numérique. Pour une philosophie du numérique (21 fév., 3:24 PM), http://blog.sens-public.org/marcellovitalirosati/editorialisation-etat-de-la-recherche/, visité le 21 mai 2025.

Vitali-Rosati, M. (2016), « Qu'est-ce que l'éditorialisation ? », *Sens public* (mars), Publisher : Université de Montréal, Département de littératures de langue française.

Vitali-Rosati, M. (2018*a*), *On Editorialization : Structuring Space and Authority in the Digital Age* (Theory on Demand, 26 ; Amsterdam : Institute of Network Cultures), 114 p.

Vitali-Rosati, M. (2018*b*), « Stylo : un éditeur de texte pour les sciences humaines et sociales », Culture numérique. Pour une philosophie du numérique (3 juin), http://blog.sens-public.org/marcellovitalirosati/stylo/, visité le 17 juin 2025.

Vitali-Rosati, M. (2020*a*), « Pour une théorie de l'éditorialisation », *Humanités numériques*, 1 (1 1er jan.). doi : 10.4000/revuehn.371.

Vitali-Rosati, M. (2020*b*), « Qu'est-ce que l'écriture numérique ? », *Corela. Cognition, représentation, langage*, HS-33 (HS-33 20 nov.). doi : 10.4000/corela.11759.

Vitali-Rosati, M. (2021), « Pour une pensée préhumaine », *Sens public*, SP1596 (SP1596 28 juill.). doi : 10.7202/1089654ar.

Vitali-Rosati, M., et Larrue, J.-M. (2019), *Media Do Not Exist* (Theory on Demand ; Amsterdam : Institute of Network Cultures), 107 p.

Vitali-Rosati, M., et Monjour, S. (2017), « Littérature et Production de l'espace à l'ère Numérique. L'éditorialisation de La Transcanadienne. Du "Spatial Turn" à Google Maps » (25 août).

Vitali-Rosati, M., Sauret, N., et al. (2020), « Écrire les SHS en environnement numérique. L'éditeur de texte Stylo », *Revue Intelligibilité du numérique* (1 | 2020 13 mars). doi : 10.34745/numerev_1697.

Von Neumann, J. (1945), *First Draft of a Report on the EDVAC* (W–670–ORD–4926, Moore School of Electrical Engineering University of Pennsylvania, 30 juin).

Voorhoeve, N. A. J. (1964), « F. Donker Duyvis and standardization », in W. E. Clason et N. I. voor Documentatie en Registratuur (éd.), *F. Donker Duyvis : his life and work* (The Hague : Netherlands Institute for Documentation et Filing).

Waldrop, M. (2001), *The Dream Machine : J. C. R. Licklider and the Revolution That Made Computing Personal* (New York : Viking Penguin).

Yami, S., Chappert, H. et Mione, A. (2015), « Séquences stratégiques relationnelles : le jeu coopetitif de Microsoft dans le processus de normalisation OOXML », *M@n@gement*, 18/5 : 330-56. doi : 10.3917/mana.185.0330.

Zacklad, M. (2004), « Processus de Documentarisation Dans Les Documents Pour l'Action (DopA) : Statut Des Annotations et Technologies de La Coopération Associées (Nouvelle Version Corrigée) » (13 oct.).

Zacklad, M. (2012), « Organisation et architecture des connaissances dans un contexte de transmédia documentaire : les enjeux de la pervasivité », *Études de communication. langages, information, médiations*, 39 (39 1er déc.) : 41-63. doi : 10.4000/edc.4017.

Zacklad, M. (2019), « Le design de l'information : textualisation, documentarisation, auctorialisation », *Communication langages*, N° 199/1 (13 mai) : 37-64. doi : 10.3917/comla1.199.0037.

Annexe 1 : Entretien avec Stéphane Pouyllau

Cette annexe fait l'objet d'un compte-rendu d'un entretien semi-directif réalisé avec Stéphane Pouyllau, responsable de l'Huma-Num Lab, le 9 janvier 2024 sur le campus Condorcet. Le sujet de l'entretien concerne l'implication de la TGIR Huma-num dans le projet Stylo.

Huma-Num existe depuis un peu plus de 10 ans (2013) et a pour vocation d'accompagner les chercheurs en SHS ainsi que de proposer une infrastructure numérique cohérente pour répondre aux différents besoins de la recherche aujourd'hui.

Selon le site web d'Huma-Num :

« **La principale mission de l'IR** est de construire, avec les communautés et à partir d'un **pilotage scientifique**, une infrastructure numérique de niveau international pour les SHS. »

Rappel : Depuis 2020, Huma-Num est partenaire du projet Stylo développé à la CRCEN et héberge l'application sur ses serveurs.

Début de l'entretien

Question : Comment une institution française et une canadienne se rencontrent et deviennent partenaires pour le développement d'un outil comme Stylo ?
Réponse : Les réflexions démarrent au congrès DH2017 à partir d'un travail entre Marcello, Michael et moi-même autour d'Érudit. À ce moment, Marcello et Nicolas mènent déjà une recherche sur l'écriture savante en contexte numérique : ils utilisent déjà plusieurs briques technologiques que l'on va retrouver plus tard dans Stylo, par exemple les formats de texte brut, sans que ces briques soient réunis dans une même interface.

Lors du congrès qui réunit les membres à l'origine du projet, les réflexions portent sur l'implémentation d'une chaine pour OpenEdition et Érudit.

Question : Comment en est-on arrivé à ce partenariat entre les deux institutions ?
Réponse : Le premier prototype fonctionnel de Stylo a été livré en 2019. Du début du projet jusqu'à cette date, Stylo était développé en interne par la CRCEN avec Servanne Monjour, Nicolas Sauret, Marcello Vitali-Rosati, Arthur Juchereau, Margot Mellet et Antoine Fauchié.

À partir du premier trimestre 2020, Huma-Num a pu devenir un partenaire officiel / institutionnel de la CRCEN et du CRIHN et a financé une partie du projet.

Avant cette date, il n'y avait pas de cadre institutionnel pour pouvoir financer un projet au Canada. Une entente entre le CRIHN sous la direction de Michael

Sinatra) et Huma-Num sous ma supervision a été montée pendant les deux années de développement du prototype (2017-2019) pour pouvoir financer Stylo.

Il faut savoir que le structure d'Huma-Num a changé durant les prémices de Stylo. Le HN Lab n'existait pas en 2017-2019 et à ce moment-là j'étais directeur technique d'Huma-Num.

Nous avons eu de la chance lors de l'établissement de l'entente car la personne qui en avait la charge à l'UdeM a travaillé au préalable au CNRS ce qui en a facilité le montage. Les papiers ont été signés début 2020 au début de la pandémie.

La signature de cette entente entre le CRIHN et Huma-Num correspond également à la fin du doctorat de Nicolas Sauret. Ainsi, lorsque j'ai monté le HN Lab, j'ai proposé d'accueillir Nicolas au sein de cette structure pour continuer à piloter une partie du projet et renforcer les liens entre les deux institutions.

Stylo est logé directement dans l'axe sur l'écriture scientifique du Lab.

Ca me semblait indispensable de mettre de l'argent dans Stylo qui fait un pont entre le monde de l'écriture solitaire et celui de l'édition.

Le deuxième axe des missions du HN Lab porte sur ce que je regroupe sous le terme d'écriture scientifique. Cet axe comporte trois parties :

— saisie du texte et du code avec Stylo et Jupyter Notebook
— stockage des données avec Nakala
— versionnement et déploiement continu (CI/CD) avec Gitlab

C'est le design du cheminement qui est intéressant, pas le choix des technologies. L'enjeu de l'axe 2 est épistémologique, pas technique.

Question : Comment a été décidée la répartition des missions autour des développements/maintenances de Stylo ?
Réponse : L'entente entre le CRIHN et Huma-Num consiste à statuer que le HN Lab peut contribuer financièrement aux développements de Stylo si quatre conditions sont respectées :

— migration de l'hébergement de Stylo (vers les serveurs d'Huma-Num)
— interconnexion à Isidore (c'est actuellement le cas avec les mot-clés contrôlés)
— authentification Human-Id
— deux réunions de feuille de route par an

Aussi le HN Lab fait le choix de ne pas se mêler des choix de partenariats ou de la communication que fait la CRCEN autour de Stylo.

Question : Quelles sont les ressources techniques mises à disposition pour Stylo ?
Réponse : Stylo fait l'objet de deux machines virtuelles (VM) sur les serveurs d'Huma-Num (basés à Lyon sur d'anciens DELL R630), une VM pour la préproduction de Stylo (`https://stylo-dev.huma-num.fr`) et une VM pour l'application en production utilisée par la communauté (`https://stylo.huma-num.fr`).

Les machines virtuelles sont dotés de 48Go d'espace disque et fonctionnent grâce au système de virtualisation KVM, performant malgré qu'il soit peu souple / ergonomique. Pour illustrer ce manque de souplesse, pour augmenter la RAM allouée à la VM, il faut la stopper puis exécuter de multiples commandes pour arriver à ce résultat avant de la relancer. Le système d'exploitation des VM est soit Debian/Ubuntu soit CentOS (sachant que ce dernier n'est plus maintenu,

Huma-Num a décidé de migrer les VM sous CentOS sous le système d'exploitation Rocket Linux)

Les racks sont gérés par l'hébergeur (CNRS qui héberge à Lyon dans un centre de calcul créé en 1986).

Question : Quels sont les moyens humains disponibles pour Stylo ?
Réponse : Avant la convention (entente) : il n'y avait pas de pôles différents (pas de HN Lab).

Depuis qu'il y a l'entente nous avons deux pôles différents :

— HN Lab pilotage et développements (Julie pour la partie *backoffice* et statistiques, et moi-même pour la direction)
— HSTC (pôle infrastructure piloté par Gérald et Thomas) (réseaux et orchestrer la VM)

Question : Quel est le coût de Stylo pour l'IR Huma-Num ?
Réponse : Il y a plus de 300 VM qui tournent chez Huma-Num. On ne peut pas établir le coût individuel de chacune des machines virtuelles.

Toutefois, Huma-Num estime que le coût consolidé, c'est-à-dire comprenant toutes les charges salariales, énergie, maintenance, etc. s'élève à environ 2500 euros par an pour chaque machine virtuelle. Étant donné que Huma-Num déploie deux machines virtuelles pour Stylo, on peut estimer le coût annuel à environ 5000 euros. En réalité il doit être légèrement inférieur puisque ce sont de petites machines virtuelles.

Ajouté à cela, Huma-Num fournit une enveloppe supplémentaire d'environ 30000 euros chaque année pour financer une partie des développements de l'application.

Question : Comment vois-tu le futur de Stylo ?
Réponse : Lors de cette première période allant de 2020 à 2024 il fallait stabiliser Stylo et y ajouter quelques fonctionnalités, ce que nous avons fait. Maintenant on entre dans une période charnière de Stylo. La phase suivante est une phase de consolidation : on professionnalise Stylo, c'est-à-dire qu'il faudrait une entité institutionnelle intégralement dédiée au projet pour pouvoir être réactif auprès de la communauté (par exemple en créant un laboratoire commun dédié à cette mission).

Le cœur de Stylo est composé des trois formats standards que sont le Markdown, le YAML et le BibTeX, le reste est périssable. Ce qu'il faut garder c'est ce cœur là. Dans cette même idée on pourrait imaginer un client de Stylo qui fonctionne quelle que soit la distribution : MacOS, Linux ou Windows.

Glossaire

agnostique En informatique, le qualificatif agnostique désigne une ressource indépendante du système au sein duquel elle se trouve. Par exemple, un logiciel agnostique fonctionnerait de la même manière sous différents systèmes d'exploitation. Il en gagne ainsi la propriété d'interopérabilité. 79, 99, 145, 170, 174, 175

BaseX BaseX est un système de gestion de base de données XML basé sur XQUERY et XPATH. Il permet de manipuler et de gérer de larges collections de données formatées en XML. Cette application dispose également d'une interface graphique permettant de visualiser les interactions avec les données. Ce projet est entièrement *open source* et repose sur une licence ouverte BSD.. 219

BibTEX BibTEX est un format standard permettant de décrire des listes de références bibliographiques. Les différentes données contenues dans un fichier sont indexées sous la forme de dictionnaire (`clé="valeur"`), procurant ainsi la capacité d'adapter les listes de références selon différentes normes bibliographiques. 142, 151, 169, 172, 180, 181, 196, 201, 203, 206, 208, 215, 219

couche d'abstraction matérielle La fonction principale d'une couche d'abstraction matérielle est de permettre la manipulation du matériel tout en masquant sa complexité et son fonctionnement. Ces couches peuvent être empilées les unes sur les autres, chacune dissimulant les détails de la précédente. 7, 59, 61, 69, 79, 96, 145, 149, 236–238

documentarisation La documentarisation est un terme mobilisé par le collectif Pédauque pour désigner l'accélération de la production documentaire survenue lors des différentes révolutions industrielles. 83, 87

git Git est un protocole de versionnement de documents. Plusieurs instances de ce protocole comme GitHub, GitLab ou Gitea permettent de versionner des scripts ou fichiers textes à partir d'interfaces graphiques ou plus communément à partir de lignes de commande. Le principe de versionnage repose sur la sauvegarde de toutes les modifications apportées à ces scripts sous la forme de commit. Ainsi, n'importe quel individu ayant accès au répertoire en question peut revenir sur les versions antérieures, comparer différentes versions entre elles et sécuriser ses données de travail. Au-delà d'un principe d'organisation des documents en usage personnel, Git a été conçu pour travailler collaborativement sur ces répertoires de données. 37, 38, 176, 240

générateur de site statique Un générateur de site statique est un logiciel utilisé pour créer des sites web à partir de fichiers dits statiques, en général au format HTML. Le générateur sert à convertir et organiser des fichiers en texte brut (par exemple en Markdown, YAML et BibTEX) pour obtenir une structure en HTML, CSS et JavaScript. Cette technologie permet de créer des sites web souvent plus légers que ceux, dynamiques, nécessitant des bases de données et géré avec des systèmes de gestion de contenu (CMS). 165, 214

idempotence L'idempotence, en informatique, signifie qu'une opération a le même effet et cela quel que soit le nombre d'applications de cette opération. 147

markdown Mardown est un langage de balisage léger créé en 2004 par John Gruber. Sa syntaxe, beaucoup plus légère et moins verbeuse que le HTML, permet de structurer et de décrire sémantiquement le texte. Il a été pensé pour pouvoir être converti facilement vers d'autres formats comme HTML, LaTeX ou PDF. Markdown se distingue des autre langages de balisages légers car il est déclinable en différentes variantes (ou saveurs). Chacune d'entre elles ajoute une particularité dans la syntaxe Markdown. Parmi les plus populaires, on retrouve : CommonMark, GitHub Flavored Markdown (GFM), MultiMarkdown, Pandoc, Quarto. 5, 6, 119, 142, 147, 150–152, 156–162, 165, 167–169, 172, 175, 178–182, 196, 201, 203, 205, 206, 215, 216, 218–220, 225, 226, 239

matérialité La *matérialité* est à distinguer de la *physicalité*. Nous l'employons dans le même sens que Hayles lui donne : la physicalité renvoie aux caractéristiques d'un objet, tandis que la matérialité est une propriété émergente. 20, 40, 41, 46, 53, 61, 75, 87, 137, 152, 173, 183, 197, 238

pandoc Pandoc est un logiciel libre de conversion de documents numériques développé par John MacFarlane en 2006. Ce logiciel réalisé en Haskell permet de convertir des documents dans divers formats tels que Markdown, LaTeX, HTML, ReST, DOCX, HTML, PDF, et bien d'autres. Les conversions opérées par Pandoc reposent sur la structuration des contenus. Sur le site web de Pandoc, une liste des formats pris en charge par le logiciel est disponible directement dans l'index du site. On remarque que tous les formats convertibles sont structurés, qu'il s'agisse d'une structuration par insertion de balises (HTML, XML, Markdown, etc) ou des standards spécifiques pour les références bibliographiques (CSL JSON qui propose une structure verticale des données et un système de description des items par clef : valeur). Pandoc n'utilise pas d'interface graphique autre que celle du terminal, c'est un logiciel dont les fonctionnalités sont accessibles par lignes de commande. 142, 182, 201–203, 207, 208, 214, 215, 219–221, 223–226, 228

redocumentarisation Dans la continuité de la documentarisation, la redocumentarisation désigne le second traitement documentaire que connaissent les documents à l'ère numérique. 87, 189, 190

single source publishing Le concept de *single source publishing*, traduit par Publication à partir d'une source unique, désigne le fait de générer plusieurs

formats à partir d'une seule et même source. Un seul et unique document permet de produire des formats divers, sans avoir besoin de basculer d'une version de travail à une autre. Que ce soit un format PDF pour l'impression, un export XML pour un diffuseur numérique ou une version numérique au format HTML, l'objectif est de travailler avec une même source. Cette source n'est pas forcément un seul fichier, mais peut comprendre des fichiers textes, des médias comme des images, des fichiers son ou des vidéos, mais aussi des données structurées de différentes façons. Il s'agit d'un enjeu éditorial qui soulève des questions autant théoriques que techniques, telles que la légitimation des contenus, l'évolution des pratiques d'édition ou la création d'outils adéquats. 142, 213, 225, 233

texte brut Le texte brut est une séquence de caractères encodés d'après une norme de codage (ASCII, Unicode, etc.). 154, 155, 206, 239

texte enrichi Selon la norme Unicode, le texte brut ne désigne pas sans distinction toutes les séquences de code de caractères, car il doit être distingué des caractères qui structurent le contenu du contenu lui-même. En suivant cette distinction, le contenu est nommé texte brut, tandis que les caractères qui le structurent sont rangés sous l'appellation de texte enrichi. 156

Index des personnalités

A

Aulu-Gelle 33

B

Bachimont, Bruno . 7, 12, 16, 19, 20, 30, 36, 39, 40, 50, 60, 82, 89, 96, 100, 138, 189, 196, 232
Ballmer, Steve 116, 117
Barad, Karen 41, 49–54, 75
Bardiot, Clarisse 201
Beauchef, Hélène 11, 219
Bellini, Mario 63
Bemer, Bob 68
Bergin, Thomas 63, 101, 102, 105–107, 109, 110
Berners-Lee, Tim 38, 145, 146
Berra, Aurélien 201
Blanc, Julie 142, 213
Bohr, Niels 52
Bolter, Jay David 44, 78
Bootz, Philippe 188
Bord, Bruno 196
Borges, Jorge Luis 82
Bosak, Jon 115, 116
Bouchardon, Serge 6, 16, 60, 64, 76
Bray, Tim 116
Briet, Suzanne 41, 83, 85, 120
Broudoux, Évelyne 25, 80, 186–188, 190, 194
Buckland, Michael 83, 85, 86

C

Cailleau, Isabelle 6, 16, 60, 76
Campbell-Kelly, Martin 107
Candel, Étienne 24, 72
Cerf, Vincent 144
Chappert, Hervé ..102, 112, 113, 117, 237
Chartron, Ghislaine 80
Chaudiron, Stéphane 80, 93
Chikofsky, Elliot J. 11
Christin, Anne-Marie 19, 28
Chun, Wendy Hui Kyong 64
Citton, Yves 62
Clark, James 116
Clément, Jean 188
Conlen, Matthew 223
Coombs, James 94
Cotte, Dominique 86
Crockford, Douglas 167, 168
Cross, James H. 11
Crozat, Stéphane ..16, 60, 64, 190

D

Daniel, Florence 201
de Mourat, Robin 99
de Nysse, Grégoire 23
Debouy, Estelle 142
Dehut, Julien 138
Deleuze, Gilles 38
Dely, Carole 191
Demaid, Adrian 11
Denis, Jérôme 99
DeRose, Steven 62, 91–94, 97, 116, 121
Descartes, René 31
Despoix, Philippe 44
Devireddy, Vishal 223
Donker Duyvis, Frits 85
Doueihi, Milad 87
Durand, David 62, 91, 93
Dürkop, Axel 220

E

Engelbart, Douglas 109, 115
Epictète 34
Epron, Benoît 186

F

Fauchié, Antoine 142, 186, 212
Foucault, Michel 38

Freiberger, Paul 108
Friesen, Norm 42

G

Gabrysiak, Louis 235
Gaillard, Ludovic 16, 60
Galloway, Alexander 38, 99
Gates, Bill 110
Genette, Gérard 72
Gensburger, Sarah 235
Genêt, Pascal 186
Gessler, Nicholas 76
Ghitalla, Franck 87
Giovacchini, Julie 201
Godfrey, Michael 67
Goldfarb, Charles 91, 239
Goldsmith, Kenneth 233
Gomez-Mejia, Gustavo 72
Goody, Jack . 19, 21, 29, 39, 73, 123
Goossens, Michel 93
Grossetie, Guillaume 141
Gruber, John 157
Grusin, Richard 44, 78
Grésillaud, Sylvie 188
Guyot, Brigitte 188

H

Habib, André 44, 46, 47
Hadi, Widad Mustafa El 81
Hadot, Pierre 21, 31, 33, 36
Haraway, Donna 52
Haute, Lucile 142, 213
Havelock, Eric 29, 35
Hayles, Katherine 41, 49–51, 53, 65, 71, 75, 77
Heer, Jeffrey 223
Hendry, David 67
Herrenschmidt, Clarisse 64, 76, 78
Hesse, Carla 194
Higgins, Dick 44
Hollander, Dave 116
Homère 35
Horowitz, Joshua 223
Hudon, Michèle 81
Hutchins, Edwin . 12, 21, 26, 27, 39, 99

I

Ihadjadene, Madjid 190
Ingold, Tim 19
Innis, Harold 48, 50

J

Jeanneret, Yves 24, 72, 232
Jobs, Steve 110
Johnson, Lyndon Baines 62, 69
Jones, Joel 220

K

Kahn, Bob 144
Kembellec, Gérald 13, 142
Kimber, Eliot 116
King, Perry A. 63
Kirschenbaum, Matthew 63, 87, 102, 106, 111, 113
Kittler, Friedrich 16, 22, 47–49, 51, 60, 64, 75, 77, 78, 110
Knuth, Donald 63, 92, 105, 106, 115, 206
Kovacs, Susan 10, 234
Krewinkle, Albert 220
Krämer, Sybille 41, 43

L

La Fontaine, Henri 91
Lambert, Frédéric 24
Lamport, Leslie 206
Lampson, Butler 109
Lankes, David 214
Larlet, David 11, 154
Larroche, Valérie 7
Larrue, Jean-Marc .. 51, 52, 59, 99
Le Crosnier, Hervé 188
Le Deuff, Olivier 8
Lenoir, Maïtané 141
Leroi-Gourhan, André 19, 21, 23, 26, 36, 232
Levy, David 87
Licklider, Joseph 107, 108
Lipsyc, Carole 190
Lucilius le Jeune 31
Lux-Pogodalla, Véronika 188
Lévi-Strauss, Claude 26, 28

M

Macchi Cassia, Antonio 63
MacFarlane, John 151, 201, 207, 220
Magliery Tom 116
Mailloux, Pierre 185

Maler, Eve 116
Maloney, Murray 116
Marc Aurèle 34
Marc Aurèle .. 21, 23, 31, 33, 37, 55
Marcoux, Yves 207
Martin, George R. R. 106
McLuhan, Marshall .. 22, 28, 41–45, 47–49, 51, 65, 80, 99, 196
McWilliams, Peter 111
Mersch, Dieter 42, 43, 48
Merzeau, Louise 25, 191
Mille, Alain 144
Mione, Anne .. 102, 112, 113, 117, 237
Mollier, Jean-Yves 186
Monjour, Servanne 37, 177, 191, 201
Mosher, Edward 93
Mounier, Pierre 200
Moureau, Sébastien 46
Mylonas, Elli 62, 91, 93

N

Nevitt, Barrington 48
Nguyen, Tu 223

O

Ong, Walter Jackson 22, 28, 29, 31, 35, 39, 41
Otlet, Paul 83, 84, 91

P

Paik, Nam June 45
Paoli, Jean 115, 116
Parisot, Thomas 141
Patashnik, Oren 206
Paveau, Marie-Anne . 19, 21, 26, 28, 30, 50, 60, 99, 232
Payeur, Cécile 189
Perret, Arthur 155
Peyrelong, Marie-France 188
Pierazzo, Elena 82, 94
Platon 22, 33, 35, 39
Poirier, Patrick 186
Pontille, David 99
Poublanc, Sébastien 201
Pouyllau, Stéphane 13, 149
Pédauque, Roger T. . 10, 20, 83, 87, 188

Q

Quintas, Paul 11

R

Ramsay, Stephen 2
Ranganathan, Shiyali Ramamrita 86
Renear, Allen 62, 91, 93
Ribera, Mireia 13, 101, 136
Rice, Stanley 92, 239
Rockwell, Geoffrey 2
Roig, Jordi 13, 101, 136
Rojas, Estrella 188
Ronconi, Filippo 36
Rose, Mark 194
Rousseau, Jean-Jacques 31
Rubinstein, Seymour 106, 107, 109, 118, 236
Réty, Jean-Hugues 188

S

Saint Antoine de Padoue 32
Saint Athanase d'Alexandrie ... 32
Saint Augustin 32, 56
Salaün, Jean-Michel 83
Sander, Nikola 151
Sapiro, Gisèle 194
Sauret, Nicolas . 9, 37, 177, 185, 191, 195, 200, 234
Schrader, Astrid 54, 55
Schrayer, Michael 105
Scopsi, Claire 13
Seurrat, Aude 24
Severo, Marta 11, 235
Shannon, Claude 42, 99
Sharpe, Peter 116
Simonyi, Charles 107–110
Sinatra, Michael 186, 191, 213
Sottsass, Ettore 62
Soubret, Jean-Louis 24
Souchier, Emmanuël . 19, 24, 25, 72, 78, 232
Sperberg-McQueen, Michael 116
Swaine, Michael 108
Swartz, Aaron 157
Sénèque 21, 31, 32, 35

T

Tadier, Elsa 24, 45
Taylor, Bob 108
Thély, Nicolas 201
Tufte, Edward 74
Turing, Alan 61, 64, 71, 100

V

Vallée, Jean-François 48, 49
van Asseldonk, Ruud 168
Vidal, Geneviève 188
Vitali-Rosati, Marcello 25, 31, 51, 52, 59, 72, 75, 99, 149, 186, 188, 191, 195, 213
von Neumann, John 61, 67

W

Waldrop, Mitchell 108
Winkler, Robert 220
Wormser, Gérard 191

Y

Yami, Saïd ... 102, 112, 113, 117, 237

Z

Zacklad, Manuel 189

Liste des acronymes

ANSI *American National Standards Institute*. 68, 69

ANVC *Alliance for Networking Visual Culture*. 214

API *Application Programming Interface*. 142, 169–171, 174, 181

ASA *American Standards Association*. 68

ASCC *Automatic Sequence Controlled Calculator*. 104

ASCII *American Standard Code for Information Interchange*. 62, 68, 69, 100, 155, 156, 237

AST *Abstract Syntax Tree*. 202, 203, 221, 223–225, 228

BIOS *Basic Input Output System*. 66, 69, 70, 236

BNF Bibliothèque National de France. 2, 235

BOAP *Biography of Artifacts and Practices*. 11–13

CDU Classification Décimale Universelle. 91

CERN Organisation Européenne pour la Recherche Nucléaire. 145

CMS *Content Management System*. 4, 5, 14, 15, 148, 149, 197–200, 211, 213, 214, 229, 232, 235

CRCEN Chaire de Recherche du Canada sur les Écritures Numériques. 2, 7, 10, 11, 13–15, 141, 144, 149, 151, 212, 214, 215, 231

CRDT *Conflict-Free Replicated Data Type*. 143

CSL *Citation Style Language*. 207

CSS *Cascade Style Sheets*. 144, 145, 154, 163, 178, 226

CTAN *Comprehensive TEX Archive Network*. 106

DOI *Digital Object Identifier*. 211

DOM *Document Object Model*. 148, 169, 178, 179

DTD *Document Type Definition*. 92–94, 98, 151, 152

ENMI Entretiens du Nouveau Monde Industriel. 191–193

ePUB *Electronic Publication*. 17, 119, 213, 214, 226, 227

EUR ArTeC École Universitaire de Recherche ArTeC. 5

FNSO Fonds National pour la Science Ouverte. 202, 212

FTP *File Transfer Protocol*. 146

GML *Generalized Markup Language*. 91, 92, 97

HTML *HyperText Markup Language*. 17, 38, 62, 82, 91, 93, 94, 97, 98, 101, 119, 121, 122, 127, 136, 142–148, 150–152, 154, 156, 157, 159, 161–163, 165, 166, 168, 169, 178, 179, 181, 196, 205, 209, 213, 215, 218–220, 224–227, 275, 277

HTTP *HyperText Transfert Protocol*. 38, 39, 144, 146, 147, 169, 174, 175, 177, 179, 181, 275

IP *Internet Protocol*. 38, 39, 144

ISO *International Organization for Standardization*. 93, 112–115, 117, 118, 237

ISSN *International Standard Serial Number*. 211

JS JavaScript. 144, 145, 154

JSON *JavaScript Object Notation*. 153, 166, 167, 175, 176, 179, 181

LINFO *Linux Information Project*. 155

ME *Management Engine*. 70

OASIS *Organization for the Advancement of Structured Information Standards*. 112, 115

ODF *Open Document Format*. 101, 103, 112, 113, 115, 118, 120, 121, 127, 130–132, 134–137, 156, 161, 237

ODT *Open Document Text*. 8, 103, 119–122, 126–128, 130, 132, 135, 142, 150, 159, 161, 164, 168, 181, 196, 219

OG *Open Graph*. 226

OHCO *Ordered Hierarchy Content Object*. 62, 91, 94–96, 121, 151, 182, 187, 221, 223, 224, 239

OOXML *Office Open XML*. 101–103, 112, 113, 115, 117, 121, 125, 132, 135, 136, 156, 161, 237

OS *Operating System*. 70

PARC *Palo Alto Research Center*. 61, 63, 107–110, 115

PDF *Portable Document Format*. 3, 4, 6, 17, 119, 142, 150, 187, 213–215, 224, 226, 228

PRISM *Publishing Requirements for Industry Standard Metadata*. 226

PUM Presses de l'Université de Montréal. 213, 214

REST *REpresentational State Transfer*. 169

RFC *Request for comments*. 144–146, 156, 162

RGI Référentiel Général de l'Interopérabilité. 112

SDL *Schema Definition Language*. 170

SGML *Standard Generalized Markup Language*. 62, 82, 89, 91–94, 97, 101, 115, 121, 156, 239

SHS Sciences Humaines et Sociales. 7, 14, 15, 231

SMTP *Simple Mail Transfer Protocol*. 146

STS *Science and Techonology Studies*. 10

SVG *Scalable Vector Graphics*. 101

TCP *Transmission Control Protocol*. 38, 39, 144

TEI *Text Encoding Initiative*. 94, 137, 142, 151, 200, 201, 203–210, 238

TGIR Très Grande Infrastructure de Recherche. 141, 168, 201

TOML *Tom's Obvious, Minimal Language*. 167

URI *Uniform Resource Identifier*. 38, 178

URL *Uniform Resource Locator*. 76, 144, 146, 151, 163, 168, 170, 178, 188, 192, 204, 207, 208, 211, 225, 226

WYSIWYG *What You See Is What You Get*. 14, 17, 63, 87, 97, 102, 106, 109, 110, 119, 122, 130, 134, 135, 137, 138, 141, 147, 148, 154, 159, 180, 238

WYSIWYM *What You See Is What You Mean*. 14, 17, 63, 102, 105, 139, 141, 148, 154, 159, 181, 238

XHTML *Extensible HyperText Markup Language*. 201, 202, 211, 226

XML *eXtensible Markup Language*. 62, 82, 91, 93, 94, 97, 101, 103, 113, 115–121, 129, 135, 137, 142, 149–151, 156–158, 181, 196, 200, 201, 203, 204, 208, 210, 219, 237, 241

XSL *eXtensible Stylesheet Language*. 8, 202, 208

YAML *YAML Ain't Markup Language*. 142, 151, 152, 156, 158–161, 165–169, 172, 179–181, 196, 201, 203, 204, 206, 210, 215, 218–220

ZAP Zone Autonome à Poétiser. 37, 38

Table des extraits de code

II.1 Exemple d'imbrication de balises OHCO 95
II.2 Exemple de mauvaise imbrication de balises OHCO 95
III.1 Espace de nom d'un fichier au format ODT 121
III.2 Espace de nom d'un fichier DOCX 121
III.3 Extrait d'un paragraphe en *WordprocessingML* 123
III.4 Structure d'un titre en ODT 126
III.5 Proposition de révision dans un document au format DOCX . . . 128
III.6 Un commentaire dans le fichier `comments.xml` de l'archive DOCX 130
III.7 Commentaire dans l'archive ODT 131
III.8 Résumé dans l'archive DOCX 133
III.9 Extrait d'un paragraphe en *WordprocessingML* 136
IV.1 Baliser une vidéo en Quarto Markdown 158
IV.2 Le paragraphe Markdown dans Stylo transformé au format HTML 162
IV.3 Extrait des métadonnées en YAML dans Stylo 165
IV.4 Requête GraphQL getMyArticle() 172
IV.5 Réponse de la requête GraphQL getMyArticle() 172
IV.6 Requête HTTP contenant la requête GraphQL de mise à jour de la version de travail 175
IV.7 Affichage du DOM pour le texte de la figure IV.8 179
V.1 Espace de nom de la TEI 203
V.2 Métadonnées en YAML dans Stylo 204
V.3 Métadonnées dans la balise xenodata de la TEI 204
V.4 Structure TEI après une première transformation par Pandoc . . 205
V.5 Référence bibliographique en BibTeX dans Stylo 206
V.6 Référence bibliographique transformée en XML TEI 206
V.7 Espace de nom de Commons publishing 208
V.8 En-tête du document TEI Commons publishing 208
V.9 Échantillon du front du document TEI Commons publishing . . . 209
V.10 Les métadonnées publiées dans l'article en ligne sur OpenEdition Journals 210
V.11 Marqueur du contenu additionnel dans le Markdown 216
V.12 Description du contenu additionnel en YAML et Markdown . . . 216
V.13 Commande Pandoc DOCX ou ODT vers Markdown 219
V.14 Échantillon de l'AST de Pandoc 221
V.15 Extrait de l'introduction d'un chapitre du livre transformé au format TeX par Pandoc 224
V.16 Extrait de l'en-tête de la version ePUB 225

Table des figures

III.1 Capture d'écran d'un titre dans un fichier DOCX de *Contribution numérique* 123
III.2 Capture d'écran d'un titre dans un fichier ODT de *Contribution numérique* 126
III.3 Capture d'écran d'un commentaire dans un fichier DOCX 128
III.4 Capture d'écran d'un commentaire dans un fichier ODT 131
III.5 Capture d'écran d'un résumé dans un fichier DOCX 133

IV.1 Schéma de l'environnement Stylo 153
IV.2 Exemple d'un titre de niveau 2 en Markdown dans Stylo 160
IV.3 Exemple d'un titre en YAML dans Stylo 160
IV.4 Un paragraphe de l'article saisi dans Stylo 162
IV.5 Une annotation dans la prévisualisation HTML de l'article . . . 163
IV.6 Résumé de l'article dans la prévisualisation HTML 166
IV.7 Échantillon de l'API GraphQL de Stylo 171
IV.8 Exemple de texte saisi en Markdown dans Stylo 179
IV.9 Schéma des étapes de matérialisation du document dans Stylo . . 180

V.1 Les transformations de Stylo vers Lodel 202
V.2 Les transformations avec le Pressoir 212
V.3 Schéma du fonctionnement de Pandoc 220
V.4 Schéma de la production du sens dans le Pressoir 227

Table des matières

Remerciements iii
Résumé v
Abstract vii
Sommaire ix
Chronologie des théories et technologies mobilisées xi
Introduction 1
Situer la recherche 1
Frictions dans l'écriture 3
Mise en place de la problématique 7
Méthodologie 10
Plan de la thèse 15
I Écriture et production du sens 19
I.1 Introduction 19
I.2 L'écriture est distribuée 22
I.3 Le support n'existe pas 40
I.4 Conclusion 56
II Écriture numérique et production du sens 59
II.1 Introduction 59
II.2 Écrire dans un environnement numérique 62
II.3 La page est un doudou 72
II.4 Document numérique et modèle textuel 81
II.5 Conclusion 96
III Écrire avec des logiciels *What You See Is What You Get* 99
III.1 Introduction 99
III.2 Histoire du traitement de texte 103
III.3 XML et le traitement de texte 112
III.4 Du texte traité au texte brut 119
III.5 Conclusion 137
IV Écrire avec Stylo 141
IV.1 Introduction 141
IV.2 Brève histoire de Stylo 144
IV.3 Modélisation des informations en texte brut 154
IV.4 La circulation des informations dans Stylo 169
IV.5 Conclusion 181

V Les transformations éditoriales avec Lodel et le Pressoir 185
V.1 Introduction 185
V.2 Écriture et éditorialisation 187
V.3 Les transformations dans *Lodel* 197
V.4 Les transformations dans le Pressoir 211
V.5 Conclusion 227
Conclusion 231
Les couches d'abstraction matérielle ne sont pas abstraites 236
Vers d'autres théories du texte 238
Bibliographie 243
Annexe 1 : Entretien avec Stéphane Pouyllau 259
Glossaire 263
Index des personnalités 267
Liste des acronymes 271
Table des extraits de code 275
Table des figures 277
Table des matières 279
Colophon 281

Colophon

Le manuscrit de cette thèse a été rédigé dans deux environnements. Le premier est un Lenovo Thinkpad T440P, un modèle de 2013, obtenu auprès de Minifree pour avoir une installation basée sur Libreboot. Je tiens à remercier Leah Rowe pour tous ses conseils et l'accompagnement à la prise en main et à la configuration de Libreboot. Libreboot est un programme *open source* qui permet d'encapsuler le BIOS propriétaire (Intel) et de passer outre ce programme pour initialiser les séquences de démarrage de l'ordinateur. Ensuite, j'ai travaillé pendant plus de trois années sous le système d'exploitation Devuan, un *fork* de Debian sans *systemd*, un *daemon* un peu trop invasif installé depuis les débuts des années 2010 sur la plupart des distributions Linux. À cela s'ajoute des essais sur une multitude d'éditeurs de texte : Neovim, Vim, Vi, Ed, Nano, Pico, VsCodium, Zettlr, Helix, Ghostwriter et, bien évidemment, le vénérable Emacs dans lequel j'ai rédigé la quasi-totalité du manuscrit.

Le deuxième environnement que j'ai paramétré pour écrire, avec l'aide de Robert Alessi, est un Lenovo Thinkpad X230 de 2012. L'architecture matérielle est sensiblement la même que celle du T440P, à ceci près que l'ordinateur est plus compact, plus léger et que la disposition du clavier n'est pas identique (Qwerty US avec une disposition ANSI pour le T440P avec 105 touches contre QWERTY UK avec une disposition ISO pour le X230 avec 104 touches). Ce dernier point aura occasionné un certain nombre de difficultés (notamment la perte de la cédille pour le caractère ç pendant plusieurs jours). Le X230 n'aura pas eu de BIOS *open source* pendant près d'une année, mais est paramétré avec le système d'exploitation OpenBSD (un autre OS Unix), orienté sur la sécurité des informations. Contrairement aux OS Linux qui sont préconfigurés, OpenBSD comporte après son installation le strict minimum pour fonctionner. Il revient à l'utilisateur de tout paramétrer pour adapter la configuration à ses propres besoins. Aussi, je n'ai pratiquement utilisé que l'éditeur de texte Emacs et LaTeX (avec la distribution TeXLive entière) pour écrire sur cet ordinateur.

Afin de sauvegarder les données et de les faire transiter facilement d'un ordinateur à un autre, j'ai largement utilisé le protocole de versionnement Git et son application Gitlab, avec l'instance mise à disposition par la TGIR Huma-Num.

L'environnement d'écriture paramétré, il nous fallait choisir des formats pour rédiger la thèse et pour faire circuler le manuscrit entre nous afin d'en discuter.

Fort de mon expérience avec l'éditeur de texte Stylo, j'ai commencé à écrire la thèse avec les formats Markdown, YAML et BibTeX en local pour produire en quelques lignes de commandes plusieurs documents de travail grâce au logiciel de conversion Pandoc. Ce fonctionnement me permettait d'adapter le document produit aux préférences de chacun en termes de relecture et annotation des contenus. Les premiers documents sont au format HTML, et déposés sur mon carnet de recherche. Les contenus peuvent ainsi être annotés avec l'application Hypothesis. Toujours avec Pandoc et des lignes de commande, j'ai aussi transformé les sources

Markdown, YAML et BibTeX en ODT, puis, si besoin, de l'ODT vers le PDF avec LibreOffice.

Ces documents ne sont que des documents de travail et ne correspondent en aucun cas au document final destiné au dépôt de la thèse.

Toutefois, cet environnement d'écriture n'était pas satisfaisant pour rédiger la thèse : le Markdown ne permettait pas d'enrichir les contenus avec des index, un glossaire, une liste des acronymes, une table des matières et de tout autre élément susceptible d'aider à la rédaction, mais aussi à la lecture. De plus, le format cible principal de ce premier environnement était le HTML pour mon carnet en ligne. Il me fallait commencer à penser une mise en page pour produire le document final (l'exercice consistant à déposer un fichier PDF à la fin).

Pour produire le document à publier, j'ai décidé d'utiliser LaTeX. Pour cela, j'ai préparé un fichier principal dans lequel se trouvent les métadonnées et les commandes de mise en page du document. Ensuite, j'ai transformé les fichiers au format Markdown en fichiers au format TeX et j'ai préservé le fichier BibTeX en l'état pour l'utiliser directement avec les paquets Biber et BibLaTeX présents dans LaTeX pour gérer la bibliographie (ce traitement est plus fin que celui proposé par Citeproc, le module de gestion des références bibliographiques de Pandoc).

La transformation de Markdown à TeX n'a pas été satisfaisante, car Markdown ne permet pas de baliser la totalité du contenu de façon à créer les commandes LaTeX dans les fichiers. Aussi, nous avons passé un certain temps à réinsérer des commandes LaTeX dans tout le manuscrit de la thèse pour obtenir le résultat actuel.

Enfin, je tiens à remercier encore une fois Robert Alessi pour la mise à disposition de la police de caractères *OldStandard* utilisée pour la mise en page de la thèse.