Un nouveau moteur pour 1024

Nicolas Taffin[1], Hervé Le Crosnier[1]

À partir de ce numéro – que vous avez entre les mains – le bulletin de la SIF, 1024, sera conçu et réalisé par C&F éditions. Cet article relate la refonte par cet éditeur de la chaîne éditoriale de 1024, depuis la réception des articles fournis par leurs auteurs jusqu'à la production d'un document prêt à être imprimé. L'approche utilisée est innovante et ouverte, basée sur les technologies du Web (HTML2Print), et son intégration dans la vie du comité éditorial de 1024 est faite sans solution de continuité afin de faciliter le travail fourni par l'équipe de bénévoles.

C&F éditions est une maison d'édition indépendante qui existe depuis 21 ans et publie principalement dans les champs de la culture numérique, de l'éducation et des communs de la connaissance. Depuis le début, nous nous y posons la question des outils de production du livre, et plus généralement des divers types de document. Curieux par nature, nous avons dès le début créé un service de R&D pour expérimenter diverses technologies, notamment celles liées aux outils du Web (HTML, CSS, Javascript, XML… de préférence avec des outils logiciels libres). Depuis 2018, nous pratiquons la mise en page de livres complets en HTML structuré et CSS[2]. Les progrès que nous faisons (pas à pas) nous ont permis de nous émanciper des logiciels propriétaires qui ont un quasi monopole dans l'industrie éditoriale.

1. C&F éditions.
2. Lire le chapitre « L'édition libre de C&F » qu'Antoine Fauchié consacre à C&F éditions dans sa thèse, *Fabriquer des éditions, éditer des fabriques. Reconfiguration des processus techniques éditoriaux et nouveaux modèles épistémologiques*, 2024, université de Montréal. https://these.quaternum.net/chapitre-05/#53-l%C3%A9dition-libre-de-cf.

DOI:10.48556/SIF.1024.24.59

Un désir de rencontre

En 2022, la rencontre avec la SIF s'est faite autour de l'idée d'un partenariat destiné à offrir aux lecteurs de *1024* des livres comme *Les entretiens de binaire* ou *Qui a hacké Garoutzia ?*, dont les auteurs sont également membres fondateurs de la SIF (ces ouvrages étant d'ailleurs réalisés en HTML2Print). Et quand l'équipe de *1024* autour de Denis Pallez nous a demandé conseil sur une réorganisation du flux de production de la revue, nous étions motivés. Cela permettait en outre de concrétiser le désir d'un membre de notre équipe, Yann Trividic, artiste, informaticien et lecteur assidu du bulletin, qui était à ce moment là en alternance chez nous pour son master d'édition. Restait à se mettre au travail.

Le problème, comme pour beaucoup de revues, est de s'affranchir de la grande hétérogénéité de formats (techniques et rédactionnels) des contributions. Cela demande beaucoup de temps d'harmoniser, d'amender, de préparer techniquement et de mettre en forme tous les textes et figures, et laisse moins de temps au travail des idées, la revue paraissant semestriellement. L'ouverture, la possibilité d'évoluer, figuraient en bonne place parmi les prémices de mise en place de ce nouveau flux éditorial, ainsi que la volonté d'intégrer plusieurs formats d'entrée.

Une architecture élaborée en discutant

Nicolas Taffin venait alors de terminer avec Julien Taquet et Agathe Baëz la chaîne éditoriale permettant les éditions en *open access* multiformat de catalogues scientifiques du musée du Louvre. Le premier ouvrage consacré au peintre Van Dyck était ainsi tout juste disponible dans plusieurs éditions réalisées à partir de la même source HTML (imprimé, site Web, pdf en ligne et epub[3]). Les besoins et le format de ce dernier projet étaient certes différents de la confection d'une revue (avec un sommaire et un rubriquage vivant au fil des contributions, les allers-retours entre éditeurs et auteurs, son fameux bouclage et ses contraintes, comme le DOI — identifiant d'objet numérique[4]). Mais des jalons étaient posés, et il y eut ainsi moins d'appréhension à se lancer.

Nicolas proposa un scénario reposant sur une architecture technique adaptée aux habitudes du bulletin, allant de la conversion de fichiers Word et Latex à la publication Web et print en passant par la sélection et les corrections. Il opta pour un format lisible et convertible, facilitant ces aspects (Asciidoc). Et Yann se lança courageusement dans les premières expérimentations : trouver les briques logicielles pour compléter les savoir-faire déjà maîtrisés, et tester différentes possibilités de mise en pages adaptées à une revue (choix des caractères, lisibilité, identité).

3. https://livres.louvre.fr.
4. https://www.doi.org.

Une chaîne de collaboration

Examinons maintenant plus en détail les principes qui nous ont guidés pour cette réalisation, puisque la SIF montre la voie à de nouveaux usages et usagers pour desserrer plus encore l'étau des logiciels propriétaires sur les productions graphiques.

Séparer le contenu de la mise en forme

La logique de fond de notre travail consiste à séparer la structure et le contenu du document de ses différentes incarnations, qui ne sont que différentes mises en forme d'une même source. Pourquoi ? D'abord parce que cela permet de travailler mieux, de se partager le travail, et aussi parce que cela rend une publication plus intelligente (la sémantique pilote explicitement la mise en forme), pérenne (formats ouverts) et interopérable (compatible avec de nombreux outils). Mais un bénéfice important de cette approche est aussi de pouvoir imaginer différents supports pour une même publication, et de les mettre en œuvre relativement simplement, ces différentes formes étant toutes des représentations d'un même document source. On appelle cela les publications multiformat ou à source unique *single source of truth*. Ce modèle renforce l'idée d'utiliser HTML / CSS, dont la séparation est au fondement des concepts, des normes et des outils. Dans cas du *single source* en HTML et CSS, cela signifie en gros que l'on utilise différentes feuilles de style pour un même document. « En gros », car en réalité c'est un peu plus complexe que cela[5].

Des traditions de composition

Changer ses habitudes est toujours un peu compliqué, et en premier lieu, pour nous-mêmes. La PAO (publication assistée par ordinateur) repose sur deux branches dominantes :

– Adobe InDesign, qui s'inscrit dans la tradition du WYSIWYG[6] popularisée par PageMaker et QuarkXPress. Un logiciel propriétaire complet et précis, mais au format fermé et au modèle «*cloud*», qui contraint au renouvellement régulier de l'abonnement sans lequel on ne peut plus accéder à ses fichiers. Les variantes WYSIWYG libres, comme Scribus, fonctionnent mais ne s'émancipent pas des limites inhérentes à ce modèle ;
– Latex qui excelle dans la composition, est libre, avec un modèle WYSIWYM[7] assez différent mais dont l'usage commun est l'emploi de classes ou modèles

5. Nicolas Taffin, « les illusions de la source unique », in *Typothérapie*, C&F éditions, 2023. En ligne : `https://polylogue.org/les-illusions-de-la-source-unique/`.
6. *What You See Is What You Get*, ou tel écran, tel écrit (une fois n'est pas coutume, la traduction en français plus compacte).
7. *What You See Is What You Mean*, ce que vous voyez est ce que vous voulez dire. `https://fr.wikipedia.org/wiki/What_you_see_is_what_you_mean`.

préétablis ou l'ajout d'une quantité de greffons qui rendent complexe l'interopérabilité et l'exploitation du code source. En outre, Latex, par sa nature même, incite à incorporer des instructions de mise en forme au contenu lui-même (au lieu de les séparer). Malgré sa précision microtypographique, Latex n'a pas été adopté par la communauté créative du design graphique.

Depuis les débuts de la composition numérique, d'autres normes ont été développées pour encoder les textes qui devaient devenir des livres. La majeure partie des langages utilisés (SGML, XML…) sont des langages à balises. Cela permet d'une part de traiter des quantités considérables de signes, de maîtriser directement le code, à la différence des traitements de texte, et de jouer pleinement de l'emboîtage des blocs pour les déposer sur la surface de rendu graphique (écran, imprimante…). Ce faisant, il s'agit de systèmes WYSIWYM contrairement aux traitements de texte et logiciels de mise en page qui ont gagné la faveur des auteurs ou des graphistes.

Avec une exception : le Web. Les personnes qui pratiquent le Web voient en lui une autre voie : séparation du contenu et de la mise en forme, universalité du format, puissance logicielle, précision du stylage, interactivité et une assez bonne lisibilité du code (qui peut être améliorée avec des variantes légères du balisage). Tout est presque là pour une PAO du futur. *Presque* : nous travaillons dans cette direction en tout cas. La route est longue, des imperfections existent, bien sûr, mais les progrès sont assez rapides et on peut se persuader que c'est une voie qui vaut la peine de s'y investir… ce que nous avons fait.

HTML

Faire le choix d'HTML ou de Latex, revient donc à utiliser une logique de formateur de texte : un code accessible avec un éditeur de texte qu'il faut interpréter pour obtenir un rendu visuel. HTML et les technologies associées (DOM — Document Object Model —, feuilles de style CSS, Javascript) sont à la fois simples (dans le nombre d'éléments à mémoriser et dans les logiques d'organisation du texte) et largement ouvertes à des programmes complexes de manipulation du document. Les navigateurs Web, s'appuyant sur un mécanisme de normalisation permanent et actif sous l'égide du W3C, sont devenus des outils rapides, puissants, adaptés à tout type de présentation sur écran. Ils peuvent donc être utilisés comme moteurs de la mise en page utilisant HTML.

Notamment, l'interface DOM permet à des scripts de modifier le rendu du code HTML directement au sein du navigateur sans avoir à modifier le document source en HTML.

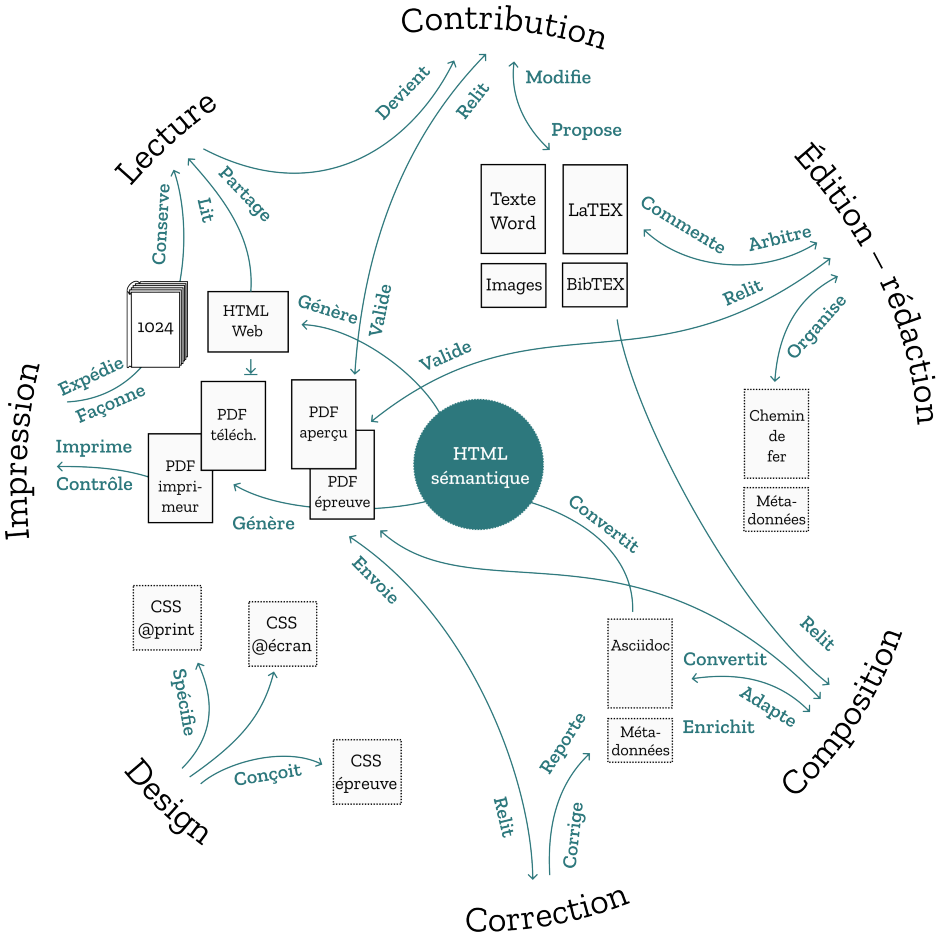

Fig. 1. Prendre en compte les échanges qui font la vie d'une publication. Ce graphique représente les rôles ou des casquettes, une même personne peut endosser plusieurs d'entre eux, mais dans l'idéal chacun apporte ses compétences.

Asciidoc, langage pivot pour rédiger le code HTML

HTML a des avantages et des inconvénients. Depuis sa création par Tim Berners-Lee pour échanger des documents, il est très répandu, très documenté, très outillé. Il a cependant trois défauts gênants pour l'édition : il est limité, il n'est pas fait pour l'imprimé, et il devient rapidement difficile à lire et écrire.

HTML est limité, il a relativement peu de structures. La définition de type de document HTML comprend des structures de base comme les titres, paragraphes, citations, quelques types de listes, du code, et des éléments interactifs (liens, formulaires). En édition, on peut avoir besoin de plus d'éléments, comme des exemples, des encadrés de différent types, des paragraphes spéciaux (chapeau, surtitre ou rubrique), des notes de bas de page, etc. Heureusement, cette limitation peut être outrepassée avec l'utilisation de classes, mais en la matière, il n'y a pas de standard, on passe donc en mode « personnalisé », chaque site Web développant ses propres classes (et les feuilles de style associées). Une fois les éléments fondamentaux définis, on prend en charge leur apparence visuelle à l'aide des feuilles de style CSS ou en manipulant le DOM. Le langage XML a essayé de dépasser cette limite, faisant en sorte que chaque fichier XML suive un modèle prédéfini, enregistré dans un schéma ou une DTD et accessible en ligne. Toutefois, les besoins sont souvent très différents et ce modèle devient rapidement assez lourd pour des usages ponctuels ou spécifiques, notamment des sites Web interactifs. Le vocabulaire de la TEI (*text encoding initiative*[8]) a été développé comme un outil généraliste pour décrire toutes les variantes de texte, avec beaucoup de précision. Il s'accompagne cependant d'une certaine complexité de mise en œuvre, ce qui le réserve encore aux éditions scientifiques en sciences humaines (l'excellente chaîne de production Metopes, utilisée par des presses universitaires s'appuie sur ce modèle[9]).

HTML n'est pas fait pour l'imprimé. Le Web déroule des documents comme des *rotulus*,[10] dans des formats d'écran très variés, dépendant des ressources de ses « clients », ce qui ne cadre pas tellement avec le degré de contrôle que demande une véritable édition imprimée sur une double page dans un format bien défini. Alors que le Web (et sa déclinaison dans le format ePub) adapte la mise en page à l'outil de lecture, l'histoire de la typographie imprimée cherche au contraire à ce que le lecteur puisse voir exactement ce que l'éditeur à organisé sur la page. C'est cette distinction qui rend difficile l'adoption d'une norme pour les médias paginés, malgré le travail du W3C en ce sens.[11] Pour adapter précisément le résultat à la fixité du média paginé, il faut donc en complément des styles définis par CSS,

8. https://tei-c.org/.

9. https://mrsh.unicaen.fr/pluridisciplinaire/poles-pluridisciplinaires/pole-document-numerique/metopes/ et https://www.metopes.fr/metopes-methodes.html.

10. Rouleau de parchemin qui se lit dans le sens de l'enroulement à la différence du *volumen* qui se lit perpendiculairement à son enroulement.

11. https://www.w3.org/TR/CSS2/page.html.

modifier *a posteriori* le DOM produit pour répondre aux besoins de l'imprimé. C'est avec l'aide de Javascript que le média paginé pourra être préparé et mis en forme, principalement avec le script libre et ouvert PagedJS, et plusieurs modules qui sont ajoutés dans la chaîne de traitement.

HTML devient vite un peu compliqué à lire et écrire. Un document HTML est composé de texte et de balises symétriques (ouvrantes et fermantes), ainsi que de différentes entités. Le besoin de découper précisément le texte au moyen de ces balises éloigne le code source HTML du texte tel qu'il sera rendu visuellement. Cela peut finir par être gênant, quand on écrit ou quand on relit son texte. Peu de gens *écrivent* directement en HTML. Des initiatives d'allègement du balisage existent depuis longtemps, comme Markdown, créé par John Gruber, avec l'aide d'Aaron Swartz en 2004. Markdown permet d'écrire du code HTML qui ressemble beaucoup à du texte simple, et sera pourtant interprété et transformé en bonne et due forme, lors d'une conversion en HTML. Mais Markdown reste sommaire et ne couvre qu'une partie de ce que l'on peut faire en HTML. C'est normal et c'est même sa qualité : il se concentre sur l'essentiel nécessaire à un blogueur. Un autre format plus expressif est Asciidoc, qui est inspiré de Docbook (un vocabulaire XML) dont il reprend l'essentiel des caractéristiques. Asciidoc permet d'écrire de la documentation technique et incorpore beaucoup plus d'éléments que Markdown, comme des exemples, des tableaux, des notes, une bibliographie, etc. Il est en outre en train de devenir une norme.

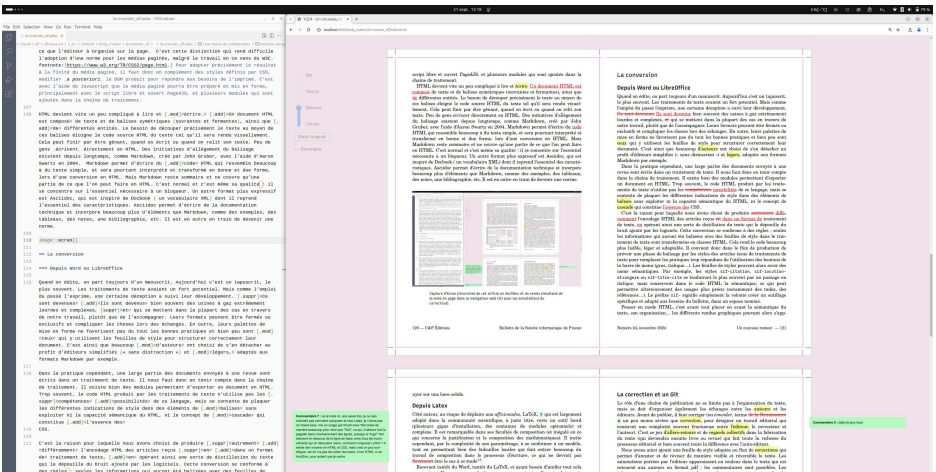

Fig. 2. Capture d'écran (récursive) de cet article en Asciidoc et du rendu simultané de la mise en page dans le navigateur web (ici avec les annotations du correcteur).

Le traitement des contributions

Depuis Word ou LibreOffice

Quand on édite, on part toujours d'un manuscrit. Aujourd'hui c'est un tapuscrit, le plus souvent. Les traitements de texte avaient un fort potentiel. Mais comme l'emploi du passé l'exprime, une certaine déception a suivi leur développement. Ils sont devenus bien souvent des usines à gaz extrêmement lourdes et complexes, qui se mettent dans la plupart des cas en travers de notre travail, plutôt que de l'accompagner. Leurs formats peuvent être fermés ou exclusifs et compliquer les choses lors des échanges. En outre, leurs palettes de mise en forme ne favorisent pas du tout les bonnes pratiques et bien peu sont ceux qui y utilisent les feuilles de style pour structurer correctement leur document. C'est ainsi que beaucoup d'auteurs ont choisi de s'en détacher au profit d'éditeurs simplifiés (« sans distraction ») et légers, adaptés aux formats Markdown par exemple.

Dans la pratique cependant, une large partie des documents envoyés à une revue sont écrits dans un traitement de texte. Il nous faut donc en tenir compte dans la chaîne de traitement. Il existe bien des modules permettant d'exporter un document en HTML. Trop souvent, le code HTML produit par les traitements de texte n'utilise pas les possibilités de ce langage, mais se contente de plaquer les différentes indications de style dans des éléments de balises sans exploiter ni la capacité sémantique du HTML, ni le concept de cascade qui constitue l'essence des CSS.

C'est la raison pour laquelle nous avons choisi de produire différemment l'encodage HTML des articles reçus dans un format de traitement de texte, en opérant ainsi une sorte de distillation du texte qui le dépouille du bruit ajouté par les logiciels. Cette conversion se conforme à des règles : seules les informations qui auront été balisées avec des feuilles de style dans le traitement de texte sont transformées en classes HTML. Cela rend le code beaucoup plus lisible, léger et adaptable. Il convient donc dans le flux de production de prévoir une phase de balisage par les styles des articles issus de traitements de texte pour remplacer les pratiques trop répandues de l'utilisation des boutons de la barre de menu (gras, italique…). Les feuilles de styles peuvent alors avoir des noms sémantiques. Par exemple, les styles `sif-citation`, `sif-locution-etrangere` ou `sif-titre-cite` se traduiront le plus souvent par un passage en *italique*, mais conservent dans le code HTML la sémantique, ce qui peut permettre ultérieurement des usages plus précis (notamment des index, des références…). Le préfixe `sif-` signifie simplement la volonté créer un outillage spécifique et adapté aux besoins du bulletin, dans un espace nommé.

Penser en mode HTML, c'est avant tout placer en avant la sémantique du texte, son organisation… les différents rendus graphiques pouvant alors s'appuyer sur une base solide.

Depuis Latex

Côté auteur, au risque de déplaire aux *afficionados*, Latex, qui est largement adopté dans la communauté scientifique, à juste titre, reste un outil lourd (plusieurs gigas d'installation, des centaines de modules optionnels) et complexe. Il est remarquable dans ses facultés de composition (et inégalé en ce qui concerne la justification des paragraphes et la composition des mathématiques). Il incite cependant, par la complexité de son paramétrage, à se conformer à un modèle, tout en permettant bien des bidouilles locales qui font entrer beaucoup du travail de composition dans le processus d'écriture, ce qui ne devrait pas être le cas à ce stade[12].

Recevant tantôt du Word, tantôt du Latex, et ayant besoin d'unifier tout cela pour y travailler, nous ne souhaitions pas nous contenter d'un simple export. Nous avons choisi de faire d'abord converger le contenu de tous les tapuscrits en Asciidoc pour pouvoir continuer à les travailler, les mettre en forme puis échanger des épreuves avec les auteurs, autour d'un format unifié de travail et de conservation avant de le publier sous la forme HTML avec quelques transformations quand c'est nécessaire.

Nous avons exploité également la capacité d'Asciidoc à conserver telles quelles les commandes Latex pour avoir un meilleur rendu des équations (mode STEM), ainsi que Bibtex pour les bibliographies fournies dans ce format. Nous avons donc développé deux convertisseurs dédiés qui convergent depuis les deux formats sources et préparent un dossier de travail extrayant au besoin les images incluses dans le document Word, en ajoutant un fichier de métadonnées pour les légendes, crédits, licences et alternatives accessibles à ces images.

On ne l'imagine pas toujours, mais c'est souvent là que le travail commence vraiment (la fameuse *émendation* des éditeurs depuis la Renaissance) : préparation, relecture, discussion, complétion, harmonisation, report des corrections d'auteur, etc. Le tout autour d'épreuves composées et unifiées. Pour cela on passe le relais à HTML et CSS au moyen du générateur de site statique Eleventy (ou 11ty)[13]. Nous avons également profité de l'occasion de cette refonte pour proposer quelques recommandations aux auteurs de *1024*, qui, espérons-le, en feront bon usage. Certains essaieront peut-être même d'envoyer leur contribution directement en format texte, Markdown ou Asciidoc !

La correction et un Git

Le rôle d'une chaîne de publication ne se limite pas à l'organisation du texte, mais se doit d'organiser également les échanges entre les auteurs et les éditeurs. Avant de publier, il faut corriger (ou *émender*, terme un peu moins

12. Daniel Allington, *The Latex fetish (Or: Don't write in Latex! It's just for typesetting)*, 2016. `http://www.danielallington.net/2016/09/the-latex-fetish/`.
13. `https://www.11ty.dev/`.

sévère que *correction*, pour désigner un travail éditorial qui construit une complicité souvent fructueuse entre l'éditeur, le correcteur et l'auteur). C'est ce jeu d'allers-retours et de regards collectifs dans la fabrication du texte (qui deviendra ensuite livre ou revue) qui fait toute la richesse du processus éditorial et bien souvent toute la différence avec l'auto-édition.

Nous avons ainsi ajouté une feuille de style adaptée au flux de corrections qui permet d'annoter et de réviser de manière visible et réversible le texte. Les annotations portées par l'éditeur apparaissent en couleur dans le texte qui est retourné aux auteurs en format pdf ; les commentaires sont possibles. Les échanges peuvent s'organiser autour du texte, de manière intégrée, jusqu'à validation de la version définitive. Nous espérons que cela enrichira les échanges entre la rédaction et la communauté des auteurs.

L'avantage de travailler avec des formats ouverts et textuels, est qu'on peut disposer des dépôts de code et de leurs fonctionnalités pour partager, suivre, collaborer et conserver tous les états et toutes les versions du travail en cours. Nous nous appuyons sur Git et Gitlab pour cela. Toute l'équipe peut être synchrone, un éditeur en ligne associé à Gitlab permet même de participer au projet sans logiciel dédié (à l'exception d'un navigateur Web, bien entendu).

L'intégration avec Eleventy/11ty

C'est à l'occasion de la conception des publications du Louvre que nous avons pratiqué 11ty, un générateur de site statique écrit en Javascript, qui permet d'effectuer les conversions et transformations du format source, à travers des templates, vers des différents dossiers de publication (en ligne ou pour l'impression). Les pages Web destinées à l'impression sont prévisualisées dans le navigateur Web, grâce à PagedJS[14], *polyfill* (ou autre logiciel supplétif), qui permet de supporter la recommandation CSS pour les médias paginés du W3C, une recommandation qui n'est pas (encore) implémentée en standard dans les navigateurs. Julien Taquet, qui a longtemps travaillé sur le développement de pagedJS, ainsi que sur 11ty, nous a rejoint pour résoudre quelques casse-tête.

Grâce à cette structure, l'image de la double page imprimée est présentée dans le navigateur entièrement en HTML, sans avoir à installer d'autre logiciel, et permet d'utiliser les outils d'inspection pour évaluer les problèmes qui peuvent être rencontrés. La composition des équations Latex dans le navigateur est prise en charge par MathJax. Le fichier PDF pour l'imprimeur est construit au moyen de la commande d'impression du navigateur appliquée au document en cours.

Chaque article se voit attribuer des métadonnées et le DOI qui permet de déclarer son emplacement. On peut ensuite déposer le dossier Web du numéro complet obtenu via 11ty sur le serveur Web de la revue, simplement par FTP. Et voilà !

14. https://pagedjs.org.

La version Web intégrale

Cette méthode apporte un bénéfice non négligeable : à partir du code HTML balisant le contenu, il est possible de développer une autre feuille de style pour rendre le fichier comme une page Web. La version Web intégrale des articles de la revue peut ainsi, modulo les ajustements nécessaires (il s'agit de deux éditions différentes d'une même source, donc deux adaptations à des supports de lecture différents) devenir lisible en ligne. Et ceci sur n'importe quel terminal, notamment des téléphones mobiles ou des tablettes, grâce à une feuille de style *responsive*. C'est un progrès pour la revue et son ouverture. Cette page permettrait également le téléchargement de formats détachés comme le PDF.

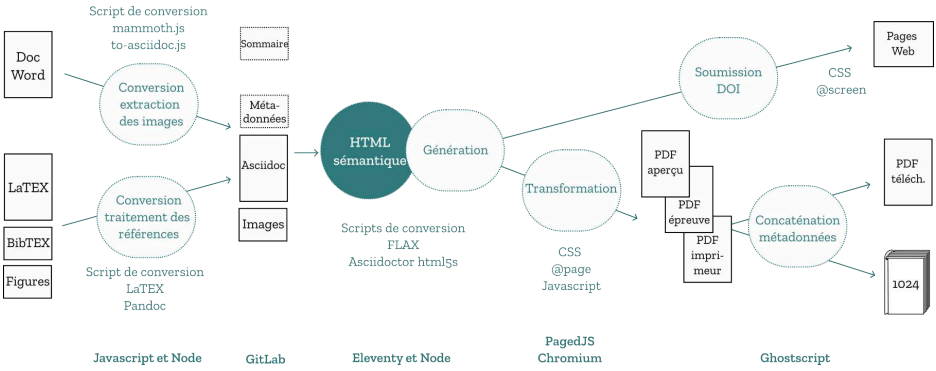

Fig. 3. Chaîne de traitement de *1024* : les moteurs.

Un modèle qui permet à chacun de se consacrer à son travail

On passe ici des étapes, mais nous avons présenté le déroulement général des opérations. Le modèle utilisé ne prétend pas être parfait, mais à chaque étape nous sommes libres d'améliorer et de modifier, et nous ne sommes nullement prisonniers d'un outil unique ou membres d'une chapelle. De plus, les outils du Web évoluant très rapidement, nous pouvons espérer profiter des progrès réalisés pour améliorer différents points de la chaîne de production proposée. Notre collaboration avec *1024* a été orientée vers l'outillage de chaque membre participant à l'élaboration de la revue. En synchronisant tout le monde autour d'une chaîne, nous espérons permettre à chacun de mieux se consacrer à son rôle : auteur, rédac'chef, correcteur, designer, compositeur, imprimeur, webmaster… et tout au bout les lecteurs, tout le monde doit y gagner. C'est ça une revue : un partage d'intelligence, pas artificielle.

Composer avec CSS

La mise en page avec HTML et CSS repose sur une recommandation du W3C pour les médias paginés qui n'a malheureusement pas été complètement implémentée par les navigateurs Web. Il existe plusieurs logiciels (propriétaires ou libres) qui comblent cette lacune. Nous avons choisi PagedJS qui agit dans un navigateur Web, est écrit en Javascript, et respecte intégralement cette recommandation et permet de s'appuyer sur la vitesse de traitement du navigateur Web pour obtenir un aperçu remis à jour en permanence. PagedJS est en outre extensible, ce qui permet de l'augmenter avec de petites améliorations personnelles que nous partageons volontiers avec la communauté des utilisateurs[15].

Les avantages et les limites de la mise en page Web

Mettre en page avec HTML et CSS présente des avantages très nets : vous l'avez compris, cela permet de séparer la structure du document de son apparence. D'un côté le document est décrit assez complètement (l'emploi de classes permettant d'étendre la sémantique comme détaillé plus haut), de l'autre CSS, est un très puissant moyen de spécifier l'aspect visuel de ces éléments, du général au particulier, de la structure de la page à la microtypographie. Le code HTML, CSS ou Javascript est lisible, et permet de travailler avec une grande variété d'outils, ce qui est important dans une chaîne ouverte ayant plusieurs intervenants. Quand on fait attention à respecter les emboîtages HTML et les règles de cascade des CSS, le code reste léger et clair, interprétable par des non-spécialistes. Enfin, ce code balisé et appuyé sur des normes solides et reconnues est suffisamment pérenne pour être conservé ou converti ultérieurement.

Bien entendu, la composition de texte avec ces outils n'atteint pas (encore) le niveau de finesse de rendu et d'intelligence de l'algorithme de Knuth et Plass[16]. Ni même du logiciel propriétaire standard de l'industrie Adobe InDesign. Mais les navigateurs — qui constituant un support de lecture très important désormais — progressent assez vite et pourraient rattraper ces autres logiciels assez rapidement. Pour aller plus loin encore dans le contrôle du mode paginé, nous avons progressé depuis 2018 et produit des extensions Javascript spécialisées. C'est un pari, avec des compromis. Il y a des limites, mais aussi des possibilités parfois plus avancées que dans les logiciels de PAO (par exemple bénéficier de la cascade et donc du contexte dans les spécifications de style, ou ne pas couper inopinément une ligne entre un article et le nom qui le suit).

CSS aide vraiment à prendre en charge différents formats, mais cela aussi avec des limites. Et on s'aperçoit à l'expérience que le multiformat implique

15. Lire à ce sujet la thèse de Julie Blanc, *Composer avec les technologies du Web, genèses instrumentales collectives pour le développement d'une communauté de pratique de designers graphiques*, 2023, université Paris 8 – ED 224 / EnsadLab / EUR ArTeC. En ligne : `http://phd.julie-blanc.fr/`.

16. Donald E. Knuth et Michael F. Plass, "Breaking paragraphs into lines", *Software-Practice and Experience*, Vol. 11, 1119-1184, 1981, `http://www.eprg.org/G53DOC/pdfs/knuth-plass-breaking.pdf`.

souvent des différences au niveau même du contenu (par exemple une vidéo ou une carte zoomable en ligne, respectivement remplacées par une image fixe et une échelle déterminée dans la version imprimée). Ce qui est plus complexe à gérer, comme nous l'avons écrit ailleurs[17]. Il faut intervenir sur la structure du document, ce qui pourrait être vu comme une rupture avec la notion de source unique. En pratique, on propose dans le même document HTML divers éléments orientés pour chaque usage. Dans la réalisation pour un usage donné, les éléments insérés pour d'autres usages sont rendus invisibles. Il s'agit là, nous en sommes bien conscients, d'un artifice qui permet de concilier la sauvegarde complète d'une source encodée, tout en s'adaptant aux divers médias de lecture.

Les défis

Notre expérience de HTML2Print a commencé il y a quelques années. Dans le premier ouvrage réalisé avec cette méthode, *Addiction sur ordonnance*, nous avons dans le colophon pointé les limites, soulevé le caractère expérimental… et invité les lecteurs à suivre la collection *Interventions* pour en évaluer les améliorations. Nos premiers ouvrages ne permettaient pas les notes de bas de page, ni la gestion des figures flottantes (le repositionnement arbitraire des figures au sein de la double page). Désormais nous avons intégré ces possibilités. Chaque réalisation apporte une meilleure maîtrise de la chaîne de production et la mise en place de subtilités qui nous aident à approcher la qualité typographique des autres outils. Dans cet ordre d'idée, réaliser une chaîne pour *1024*, comme auparavant pour le Louvre, nous incite à relever tout un lot de défis. Notamment la gestion du code et des équations, bête noire des logiciels de PAO et point fort de Tex.

On pourrait aborder d'autres aspects notables particuliers de ce chantier. Éditer une revue ou un bulletin est plus difficile qu'éditer une monographie. La diversité des textes, des types d'objet hors texte nous amène à nous adapter au fil des contributions et l'ensemble ne manque pas de surprises. Patrice Naudin, l'œil scrupuleux qui relit et amende les contributions jusqu'au «*bon à tirer*» pourrait en témoigner. Nous lui rendons ici hommage.

Faire de grands choix mais aussi les mille petits trucs et astuces qui permettent de caler une mise en page pour assurer un bon enchaînement des pages est quelque chose qui repose encore sur l'intelligence humaine. Cette humanité que ceux qui veulent tout automatiser essaieront de remplacer par une intelligence artificielle. Un jour probablement, mais pour le moment nous tablons, en ce qui nous concerne, non pas sur l'automatisation, mais sur l'outillage le plus correct possible des acteurs et de leurs compétences et le développement de modes de collaboration efficaces et agréables pour une équipe.

17. Nicolas Taffin, *op. cit.*

Un des aspects intéressants de tels développements est de pouvoir les partager, dans leur globalité (comme une chaîne) mais aussi à la découpe, un module pouvant à lui seul rendre bien des services. C'est Nous l'avons donc prévu, dès que nous aurons suffisamment « essuyé les plâtres », nous imaginons après un ou deux numéros. La documentation est faite au fil de l'eau dans cet esprit. Nous avons déjà en tête des idées d'évolutions futures et vous, lecteur, en aurez peut-être aussi d'autres à nous souffler.