# OnePub : une plate-forme collaborative pour repenser l'édition multi-formats

CAMILLE GOBERT, Université Paris-Saclay, CNRS, Inria, LISN, France
MICHEL BEAUDOUIN-LAFON, Université Paris-Saclay, CNRS, Inria, LISN, France

Mots Clés et Phrases Supplémentaires: chaîne éditoriale, collaboration, *local-first*, source unique, sorties multiples

## 1 Motivations

Le processus éditorial moderne est souvent perçu comme une suite de tâches bien ordonnées : un auteur rédige un manuscrit, un éditeur le commente et le valide, un graphiste le met en page et un imprimeur produit le livre papier. En réalité, la collaboration entre ces différents acteurs inclut de nombreux aller-retours supplémentaires : entre l'auteur et l'éditeur (pour itérer sur le contenu du livre), entre l'éditeur et le graphiste (pour itérer sur la mise en page du livre), ou encore entre les graphiste et l'imprimeur (pour adapter le PDF aux contraintes de l'imprimeur). Outre le coût des échanges en eux-même, qui sont souvent réalisés par email, le fait que chaque acteur utilise ses propres logiciels, tels que Microsoft Word et Adobe InDesign, qui utilisent chacun leur propre format de données, rend nécessaire de coûteuses conversions entre formats plus ou moins manuelles à chaque itération. Ce second problème devient d'autant plus visible dans le cas d'une publication *multi-formats*, c'est-à-dire lorsqu'un même ouvrage n'est pas seulement publié au format papier mais également dans un ou plusieurs formats numériques tels qu'une publication web (HTML et CSS) ou pour liseuse électronique (ePub). Produire de tels formats requiert généralement d'utiliser des chaînes éditoriales entièrement différentes, dédoublant ainsi la charge de travail requise. Il s'agit pourtant d'un besoin croissant chez les éditeurs, chez qui les formats numérique représentaient 12,7% de leur chiffre d'affaire global en Europe et 9,6% en France en 2023. [1]

En parallèle de cette approche, plusieurs langages de description de documents ont été conçus afin de décrire le contenu, la structure sémantique voire le style de documents sous la forme de texte brut. Ces langages se spécialisent aussi bien pour des publications paginées, tels que LaTeX, SILE et Typst, que pour des publications non-paginées, tels que Markdown, Pollen et AsciiDoc. Plus récemment, la pratique du *web-to-print* [1] propose d'utiliser de tels langages afin de créer un document paginé destiné à l'impression (*print*) à partir d'un document non-pagigné destiné à la lecture sur écran (*web*). Celle-ci permet ainsi de produire plusieurs sorties à partir d'une source unique, en ne dupliquant que les règles CSS qui varient d'un média à un autre. Elle a par exemple été utilisée pour produire le catalogue des sculptures de la Villa Chiragan pour le musée Saint-Raymond de Toulouse, dont la version imprimée et la version web proviennent d'une unique source rédigée en Markdown. [2]

En outre, plusieurs progrès dans l'édition collaborative de documents ont également permis de construire des alternatives synchrones à l'échange de fichiers par email ou par dossier partagé et au versionnage à l'aide d'outils comme Git. La transformée d'opérations (*operational transform*, ou « OT »), permet de collaborer en temps réel à l'aide d'un serveur central qui réordonne les opérations de tous les utilisateurs. On la retrouve aujourd'hui mise en œuvre dans des éditeurs collaboratifs tels que Google Docs et Overleaf. Les types de données répliqués sans conflit (*conflict-free replicated data types*, ou « CRDTs ») permettent de se passer du serveur central : si l'état de chaque client peut temporairement diverger des autres (par exemple, en cas de coupure réseau), les algorithmes de résolution de

1. D'après les chiffres du Syndicat National de l'Édition et de la Fédération des Éditeurs Européens.
2. Cette chaîne éditoriale expérimentale, mise en place par Julie Blanc et Antoine Fauchié, est décrite en détail dans un billet de blog de Julie.

Authors' Contact Information: Camille Gobert, gobert@lisn.fr, Université Paris-Saclay, CNRS, Inria, LISN , 91400 Orsay, France; Michel Beaudouin-Lafon, mbl@lisn.fr, Université Paris-Saclay, CNRS, Inria, LISN , 91400 Orsay, France.

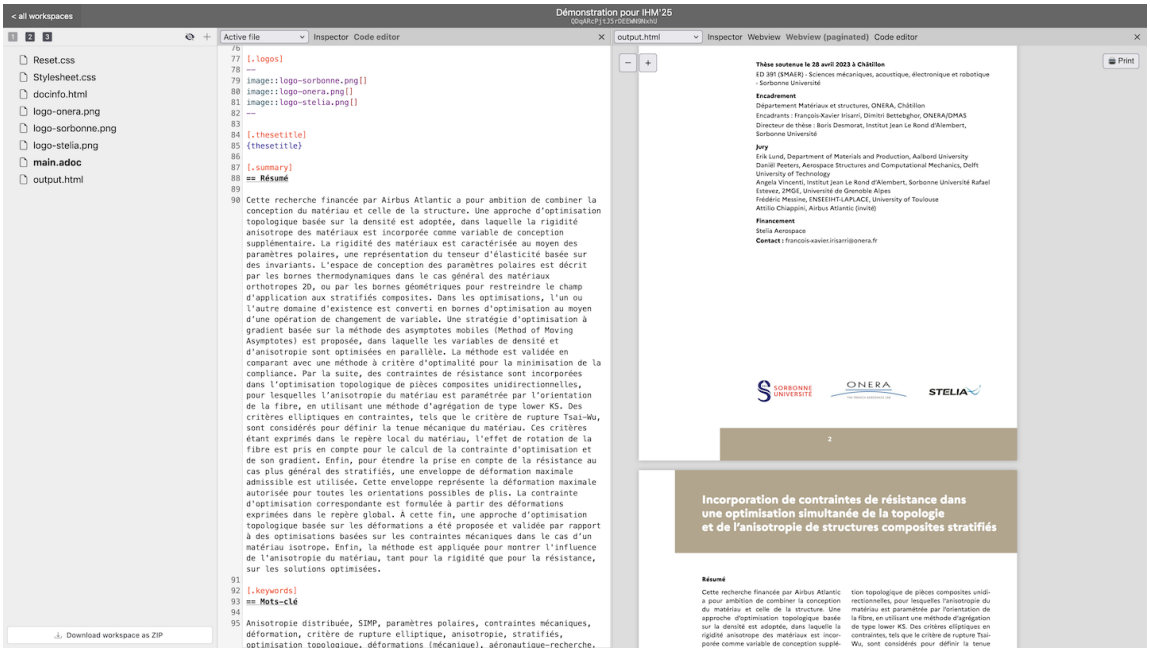


Fig. 1 – Interface de OnePub montrant un résumé de thèse adapté depuis une version PDF (extraite de la liste des thèses soutenues à l'ONERA en 2023, pp. 50–51). Sont illustrés, de gauche à droite : (1) la liste des fichiers faisant partie des sources partagées de l'ouvrage ; (2) le contenu du document au format AsciiDoc dans un éditeur de code ; et (3) le rendu paginé de la sortie HTML.

conflits propres à chaque type de donnée garantissent que deux utilisateurs ayant échangé leurs changements mutuels convergent vers le même état [7]. Les CRDTs sont un élément central de la notion de *local-first software* [4], qui soutient que chacun devrait pouvoir disposer d'une copie locale de ses données sans que cela n'entrave pour autant la possibilité de collaborer de manière synchrone avec d'autres personnes.

À notre connaissance, il n'existe aucun environnement d'édition multi-format qui soit à la fois *web-to-print* et *local-first*. Le système que nous développons constitue une preuve de concept fonctionnelle qui combine ces deux concepts.

## 2 Démonstrateur

OnePub est un environnement permettant de créer ses propres chaînes éditoriales collaboratives. Les documents qu'il permet de créer sont composés de plusieurs fichiers, tels que des fichiers de contenu, écrits dans le langage AsciiDoc ; des fichiers de style, écrits dans le langage CSS ; ou encore des fichiers de sortie, écrits dans le langage HTML et pouvant être paginés à l'aide de la bibliothèque Paged.js.[3] Chaque fichier peut être édité par une ou plusieurs personnes en parallèle en utilisant un ou plusieurs environnements d'édition : par exemple, deux collaborateurs peuvent chacun éditer le même fichier AsciiDoc, l'un sous forme de code, l'autre sous forme de texte enrichi. L'interface du système (Figure 1) est composée de plusieurs panneaux permettant d'interagir avec les fichiers d'un document à la façon d'un explorateur de fichiers et d'afficher un ou plusieurs fichiers en utilisant la vue de son choix.

Le fonctionnement de OnePub repose sur un système de fichiers réactif et distribué de notre conception, baptisé `r3` (*reactive replicated resources*), dans lequel un fichier est encodé à l'aide de CRDTs et peut être définit de deux

3. Paged.js est une bibliothèque JavaScript qui permet de paginer un document HTML dont le rendu est assuré par un navigateur web.

façons : soit par les données qu'il contient, soit par une suite d'opérations permettant de (re)calculer son contenu. Dans le second cas, chaque opération correspond à une action distincte, telle que lire le contenu d'un autre fichier ou le convertir dans un autre format, qui, à chaque exécution, renvoie la liste des fichiers dont dépend cette opération. De cette manière, `r3` maintient à jour un graphe des dépendances entre tous les fichiers qui composent un document, à la façon de FileWeaver [3]. Cela permet notamment de recalculer automatiquement le contenu d'un fichier lorsqu'une de ses dépendances est modifiée, peu importe que le changement soit local ou provienne d'un collaborateur distant : par exemple, recalculer la sortie HTML lorsque l'auteur modifie le contenu en éditant le fichier AsciiDoc dont il dépend ou lorsque le graphiste met à jour les règles de style spécifiées dans le fichier CSS déclaré comme feuille de style du fichier AsciiDoc.

Notre démonstrateur permet d'illustrer trois aspect caractéristiques de OnePub : (1) la multiplicité des environnements d'édition d'un même fichier source, (2) la multiplicité des formats de sortie générés et (3) la collaboration synchrone qui fonctionne également hors ligne. Deux personnes du public pourront utiliser différentes sortes d'éditeurs afin d'interagir avec le système en parallèle de manière à modifier le contenu et le style d'un document (vierge ou issu d'exemples que nous fournirons). Le reste du public pourra observer la mise à jour de deux formats de sortie du document (paginé pour l'impression, non-paginé pour le web) en temps réel sur deux écrans séparés.

## 3 Perspectives

Le démonstrateur que nous proposons, quoique fonctionnel, demeure limité. Répondre à ces limites requiert à la fois un travail d'ingénierie et de recherche afin d'évaluer les limites du système et les besoins de ses utilisateurs potentiels, développer de nouvelles fonctionnalités et repousser l'état de l'art des systèmes réactifs et synchrones.

Du point de vue de l'ingénierie, une première perspective consiste en l'ajout de nouveaux environnements d'édition : par exemple, une interface permettant aux éditeurs d'annoter un fichier AsciiDoc et une autre permettant à un graphiste d'interagir avec une maquette de page spécifiée en CSS sans avoir besoin de lire et écrire du code. Nous envisageons également d'exposer le graphe de dépendances aux utilisateurs, qui pourraient alors visualiser les interactions entre les fichiers d'un document et opter pour différentes stratégies de recalcul selon leurs besoins. De cette manière, un auteur qui modifie le contenu d'un livre pourrait par exemple choisir de bloquer la prise en compte des modifications d'un fichier CSS effectuées en parallèle par le graphiste afin que la mise en page du rendu du document demeure fixe le temps que l'auteur termine son chapitre.

Du point de vue de la recherche, de nombreuses questions restent en suspens concernant les meilleures manières de visualiser et d'interagir avec l'historique très fin que capturent les CRDTs.[4] Celles-ci incluent par exemple la représentation de la présence et de l'activité des autres utilisateurs sur le fichier que l'on est en train de modifier, ou encore différentes manières de visualiser les modifications effectuées au fil du temps, à la façon du travail mené sur Patchwork par Ink & Switch. Outre l'interaction avec l'historique, une autre direction de recherche concerne la bidirectionnalité des dépendances entre fichiers, qui permettrait par exemple de corriger une faute d'orthographe en éditant un fichier HTML généré et voir la modification se propager automatiquement dans le fichier AsciiDoc l'ayant généré. Bien que des exemples tels que les *source maps* et SyncTEX [5], un outil permettant d'associer une position dans le PDF généré par un compilateur LaTeX à une position dans le fichier source, illustrent la possibilité de tracer la provenance d'une information issue d'une transformation, inverser celle-ci demeure une tâche difficile dans le cas général, comme en attestent de récents travaux sur les lentilles computationnelles [6] et sur l'évolution des schémas de données [2].

4. Par exemple, dans le cas d'un CRDT de texte, la structure de donnée catalogue l'insertion, la modification et la suppression de chaque caractère.

## Remerciements

Nous remercions Wendy Mackay, Nicolas Taffin, Yann Trividic et Emma-Jade De Moor pour leurs riches contributions à la conception de OnePub. Ces travaux ont bénéficié du soutien du projet ERC N° 695464 ONE « Unified principles of interaction » et du projet ERC PoC N° 101113339 OnePub « Single-source collaborative publishing ».

## Références

[1] Julie Blanc. 2023. *Composer avec les technologies du web — Genèses instrumentales collectives pour le développement d'une communauté de pratique de designers graphiques.* Ph. D. Dissertation. Université Paris 8.

[2] Jonathan Edwards, Tomas Petricek, Tijs van der Storm, and Geoffrey Litt. 2024. Schema Evolution in Interactive Programming Systems. *The Art, Science, and Engineering of Programming* 9, 1 (2024), 2 :1–2 :33. doi:10.22152/programming-journal.org/2025/9/2

[3] Julien Gori, Han L. Han, and Michel Beaudouin-Lafon. 2020. FileWeaver : Flexible File Management with Automatic Dependency Tracking. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology (UIST '20)*. ACM, 22–34. doi:10.1145/3379337.3415830

[4] Martin Kleppmann, Adam Wiggins, Peter van Hardenberg, and Mark McGranaghan. 2019. Local-First Software : You Own Your Data, in Spite of the Cloud. In *Proceedings of the 2019 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software (Onward ! 2019)*. ACM, 154–178. doi:10.1145/3359591.3359737

[5] Jérôme Laurens. 2008. Direct and Reverse Synchronization with SyncTeX. *TUGBoat* 29, 3 (2008), 365–371.

[6] Geoffrey Litt, Peter van Hardenberg, and Orion Henry. 2021. Cambria : Schema Evolution in Distributed Systems with Edit Lenses. In *Proceedings of the 8th Workshop on Principles and Practice of Consistency for Distributed Data (PaPoC '21)*. ACM, 1–9. doi:10.1145/3447865.3457963

[7] Marc Shapiro, Nuno Preguiça, Carlos Baquero, and Marek Zawirski. 2011. Conflict-Free Replicated Data Types. In *SSS 2011 - 13th International Symposium Stabilization, Safety, and Security of Distributed Systems*, Vol. 6976. Springer, 386. doi:10.1007/978-3-642-24550-3_29