After Word: l'avenir du traitement de texte

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 20 janvier 2001. Dernière mise à jour le 8 juin 2007

https://www.bortzmeyer.org/afterword.html

Aujourd'hui, si vous écrivez un document, c'est probablement avec Microsoft Word. Bien que sa part de marché des traitements de texte WYSIWYG soit variable selon les pays (particulièrement forte en France, elle est plus faible aux USA ou en Allemagne) et qu'elle soit perçue comme plus importante qu'elle ne l'est réellement car de nombreux systèmes de production de documents professionnels sont moins visibles (moins « médiatiques »), mais très utilisés, MS-Word domine les esprits et une bonne part du marché.

Cela soulève plusieurs critiques : elles vont de l'agacement que l'on ressent lorsqu'un correspondant inconnu vous envoie d'autorité un document MS-Word, sans même avoir pris la peine de se demander si vous avez ce logiciel, à la prise de conscience des dangers d'une telle dépendance vis-à-vis du format non documenté d'un logiciel particulier d'une entreprise privée. Si vous écrivez vos documents avec MS-Word, Microsoft est le vrai propriétaire de vos données : elles ne pourront être lues que si cette société le veut bien. Ainsi, les documents réalisés avec les premières versions de MS-Word ne sont plus lisibles par un Office d'aujourd'hui.

La domination de MS-Word sur les esprits est telle que, si un logiciel tiers a du mal à ouvrir un document MS-Word (format non documenté, rappelons-le), c'est ce logiciel qui est accusé de ne pas être « assez compatible »

Il existe de nombreuses alternatives souvent citées à MS-Word. Les partisans du logiciel libre et plus précisément les défenseurs des systèmes Linux en connaissent par cœur plusieurs, pour pouvoir les asséner aux conservateurs inquiets d'abandonner Microsoft.

En se limitant à celles qui tournent (plus ou moins bien) sur un PC avec Linux, il y avait WordPerfect, de Corel, Applixware, de VistaSource, StarOffice, de Sun. Tous sont des logiciels commerciaux (StarOffice a une version libre désormais, voir plus loin). Tous ont un format d'enregistrement des données privé (comme MS-Word) et tous ressemblent à MS-Word : même inflation des fonctions et des ressources consommées, mêmes concepts sous-jacents, interface souvent proche. C'est même un argument commercial : « nous sommes comme MS-Word ». Les seuls points sur lesquels se distinguer sont donc le fait

de tourner sur Linux, le prix (StarOffice est même gratuit) et l'anti-Microsoftisme primaire (≪ Bill Gates, grand méchant ≫, etc).

À noter que, la demande d'un logiciel compatible MS-Word étant très forte, les promoteurs des logiciels libres n'hésitent pas à se faire délégués commerciaux pour des logiciels très fermés, du moment qu'ils tournent sur Linux.

Par contre, toujours dans la même catégorie, il existe des traitements de texte libres. Le serveur Freshmeat en recense beaucoup <http://freshmeat.net/appindex/X11/Office%20Applications.html>, souvent utilisant un format de sauvegarde documenté. C'est le cas de Ted (qui utilise RTF), AbiWord ou OpenOffice, originellement la version libre de StarOffice.

Aucun de ces logiciels n'a été un de ces grands succès qui enchantent les promoteurs du logiciel libre et qui permet à beaucoup de projets d'attirer des compétences techniques et des zélateurs enthousiastes. La plupart sont restés à moitié finis et plantent plus souvent qu'à leur tour. Plusieurs, qui semblaient très prometteurs, ont déjà sombré. Seul OpenOffice a fini par décoller et à rassembler une communauté significative.

Les logiciels de cette section sont donc très différents, par leur part de marché, par leur stabilité, par leur statut (libre ou commercial). Mais ils ont tous quelque chose en commun : ils reposent sur un paradigme hérité du papier, ils sont WYSIWYG ("*What You See Is What You Get*"). Ce terme avait été inventé au début des écrans graphiques pour désigner les logiciels qui présentaient sur l'écran, avant l'impression, une image plus ou moins exacte de ce que le document allait être sur le papier. À l'époque, c'était un progrès : on n'avait plus besoin d'attendre l'impression pour découvrir qu'un paragraphe était entièrement en italique alors qu'on n'avait voulu faire apparaître ainsi qu'un seul mot.

Mais le WYSIWYG est inadapté à la publication moderne : il n'a en effet de sens que si on publie sur un seul support. C'était vrai autrefois : tout finissait par se retrouver sur le papier. Mais, dans un monde de "*ebooks*", de navigateurs Web, de téléphones portables WAP et de papier (qui n'a pas disparu), cette supposition ne peut plus être vraie. Il est donc impossible de concevoir des logiciels WYSIWYG quant on publie pour une vaste gamme de périphériques d'affichage.

Une catégorie d'outils très différents existe depuis longtemps mais connait un succès à éclipses (avec un net regain d'intérêt parfois). Il s'agit des outils qui ne dépendent pas du modèle WYSIWYG mais qui reposent sur l'idée de structuration de document. Un document n'est pas vu comme une image, avec des effets de présentation (en italique, en petit, etc) mais comme une arborescence d'élements : un article est composé d'un en-tête (qui lui-même comprend un auteur et une date) et de plusieurs sections (qui chacune comporte des paragraphes).

Pour assurer le rendu de ces documents, on alloue à chaque élément des attributs de présentation. Tous les titres de section en gras ou tous les liens hypertextes en bleu, par exemple. Mais l'auteur d'un document ne pense pas à cette présentation : elle peut varier selon les supports de sortie (et même être modifiée par l'utilisateur, comme un navigateur Web permet de changer la police utilisée). Il se concentre au contraire sur le contenu.Ce modèle ne fonctionne évidemment que si le contenu est important, que cela soit un roman, un texte de loi, ou une documentation technique. Les brochures publicitaires, au contraire, ne peuvent pas être traitées de cette manière : leur contenu n'a pas d'intérêt, leur apparence est primordiale.

Travailler ainsi suppose un changement radical de conception d'un document. L'idée même d'un contenu identique présenté de manière différente sur deux supports (par exemple le papier et une page

Web) a du mal à passer. C'est l'une des raisons pour lesquelles cette approche semble toujours nouvelle, alors qu'elle est en fait plus ancienne que le WYSIWYG.

Parmi les systèmes de production de documents qui fonctionnent ainsi, il y a des systèmes qui utilisent un jeu de macros au dessus d'un langage de description de présentation (comme troff avec ses macros `-me` et `-ms` ou comme LaTeX) et il y a ceux où le langage de présentation est complètement extérieur, notamment toute la famille SGML/XML.

SGML et son populaire descendant XML sont bâtis sur une idée simple : présentation et structure sont trop importants pour être mêlés. SGML n'ayant pas eu, et de loin, le même succès marketing, je ne parlerai plus que de XML. XML voit le document comme une arborescence d'élements, toute la présentation étant confiée à des programmes extérieurs (utilisant parfois des langages spécialisés comme XSL).

Naturellement, il y a des solutions mixtes : les styles de MS-Word permettent, si on les utilise avec discipline, d'éditer un document structuré, reportant les choix de présentation aux styles. Et il existe des interfaces graphiques à des langages comme LaTeX permettant d'écrire en style WYSIWYG (tant que le document n'est destiné qu'au papier).

Faisons un détour par le passé. Il y a trente ans, l'informatique était dominée par une entreprise : IBM. Plus que les parts de marché, IBM avait des parts d'esprit. L'informatique, c'était IBM. Le modèle favori d'IBM (les gros ordinateurs), les "*mainframes*" connectés à des milliers de terminaux passifs, incapables de communiquer même parfois avec des "*mainframes*" du même constructeur était le modèle dominant. Ceux qui proposaient des ordinateurs directement dans les départements d'une entreprise (sans même parler des ordinateurs dans les bureaux des utilisateurs) étaient de doux rêveurs. Jakob Nielsen décrit ce modèle avec finesse en racontant ce que dessinent, pour représenter « un ordinateur », l'utilisateur de 1970 et celui de 2000. Le premier trace une grosse boîte avec des dérouleurs de bande magnétique apparents. Le deuxième ne montre qu'un écran, une souris et un clavier.

IBM avait quelques concurrents officiels. Il fallait bien, pour donner l'impression de pluralisme. On les appelait le BUNCH, des initiales des cinq principaux concurrents (la plupart sont bien oubliés aujourd'hui). Aucun de ces concurrents officiels ne se démarquait d'IBM, de ses "*mainframes*" et de son absence de réseaux : tous faisaient de l'IBM, un peu moins cher qu'IBM. Aucun n'a réussi à ébranler si peu que ce soit la domination d'IBM.

Ceux qui ont secoué le cocotier, et ramené IBM à un rang plus modeste, ne faisaient pas partie du BUNCH. C'étaient des "*outsiders*" complets, comme Digital qui allait lancer la mini-informatique ou Apple, Commodore ou Atari, qui allaient promouvoir la micro-informatique. Ce qui a affaibli sérieusement IBM, ce ne sont pas les pâles copieurs du BUNCH, ce sont ceux qui changeaient le paradigme dominant.

De même, pour la bureautique (dont le traitement de textes n'est qu'une des applications), le changement, et la mise en cause du quasi-monopole de Microsoft, ne viendront pas des entreprises qui tentent de copier MS-Word en moins cher. Le changement viendra de ceux qui changent le paradigme, la façon de voir les choses.

Des logiciels de production de documents basés sur les idées de structuration exposées ci-dessus existent-ils ? Pas vraiment encore, et surtout pas en logiciels libres. Les éditeurs SGML ou XML, comme celui d'ArborText, sont tous des produits commerciaux (souvent avec des prix à faire passer Bill Gates pour un mécène). Il existe des bouts de solution d'édition structurée, des démos, des logiciels en version béta, rien encore dont on puisse honnêtement dire « Fini, MS-Word, voilà ce que j'utilise désormais ». Les pionniers sont l'éditeur libre Emacs, avec des extensions comme PSGML <`http://www.lysator.`

liu.se/projects/about_psgml.html> ou nxml-mode <http://www.thaiopensource.com/nxml-mode/> sont une préfiguration de ce que pourrait être un tel éditeur), le logiciel Conglomerate (qui ne semble plus développé, alors qu'il était loin d'avoir atteint la version 1.0), une partie du système Scenari...

Je n'ai pas personnellement les compétences pour développer un tel programme. Cela n'est pas trivial, non seulement pour des raisons techniques traditionnelles mais aussi parce qu'un tel éditeur repose sur des concepts très nouveaux et qu'il n'est pas évident de savoir comment les présenter à des utilisateurs, surtout aux utilisateurs actuels, attachés au papier (et à son corollaire, le WYSIWYG). Écrire un tel logiciel suppose une réflexion sur la meilleure façon de présenter des documents structurés à leur auteur, probablement appuyée sur de nombreux essais auprès d'utilisateurs. Il est important de se rappeler de la variété des utilisateurs de traitement de texte : du chef de service qui produit une note d'une demi-page, qui ne sera pas archivée, à l'étudiant qui rédige sa thèse, au romancier qui écrit un livre, ou à l'auteur à temps plein de documentations techniques..

Mais c'est quand même là qu'est l'avenir de la production de documents bureautiques. Il est temps de cesser d'essayer de concurrencer MS-Word avec d'autres obésiciels basés sur les mêmes principes que le logiciel dominant. Si l'histoire de l'informatique peut nous servir de guide, c'est pour nous apprendre que c'est une voie sans issue. La solution, c'est le changement de paradigme, le passage au document structuré.

Des articles intéressants sur le même thème :

— « En finir avec Word ! Pour une analyse des enjeux relatifs aux traitements de texte et à leur utilisation <https://eriac.hypotheses.org/80> » (par Julien Dehut).