Academic writing and publishing beyond documents

Cerstin Mahlow
School of Applied Linguistics, Zurich University of Applied Sciences
Winterthur, Switzerland
cerstin.mahlow@zhaw.ch

Michael Piotrowski
Faculty of Arts, Department of Language and Information Sciences, University of Lausanne
Lausanne, Switzerland
michael.piotrowski@unil.ch

ABSTRACT

Research on writing tools stopped in the late 1980s when Microsoft Word had achieved monopoly status. However, the development of the Web and the advent of mobile devices are increasingly rendering static print-like documents obsolete. In this vision paper we reflect on the impact of this development on scholarly writing and publishing. Academic publications increasingly include dynamic elements, e.g., code, data plots, and other visualizations, which clearly requires other tools for document production than traditional word processors. When the printed page no longer is the desired final product, content and form can be addressed explicitly and separately, thus emphasizing the structure of texts rather than the structure of documents. The resulting challenges have not yet been fully addressed by document engineering.

CCS CONCEPTS

• **Applied computing** → **Markup languages**; *Format and notation*; *Document scripting languages.*

KEYWORDS

WYSIWYG, document structure, scholarly publishing, interactive editing

ACM Reference Format:
Cerstin Mahlow and Michael Piotrowski. 2022. Academic writing and publishing beyond documents. In *DocEng '22: Proceedings of the 22nd ACM Symposium on Document Engineering.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3558100.3563840

1 INTRODUCTION

By the early 1990s, Microsoft Word had become the de facto standard for word processing software: Word was bundled with many PCs, so users could just turn on their new PC and start writing. This ubiquity had various effects: writers became accustomed to the appearance, features, and affordances of Word, resulting in a halt of serious development of writing technology as there was no actual demand. Even more importantly, Word's file format became the standard format for interchange between co-authors, for submissions, and for further processing in publishing houses. The printed page produced with Word was considered the canoncial final document, i.e., the "what you get" of WYSIWYG ("what you see is what you get").

However, by the late 1990s, the printed page as the end product of WYSIWYG authoring tools was confronted first with the development of the Web and then, a decade later, with the advent of mobile devices. Both developments enabled—and required—new types of documents and thus new tools, as well as new practices and processes for producing those documents. These general technological changes, which prompted the development of writing tools that no longer primarily target print, coincided with academic trends, such as open science and an increasing emphasis on reproducibility and data sharing. Academic publishing is very conservative and change happens only slowly, but we now begin to see effects even there.

In this vision paper we will first look at the production of documents from a writer's perspective. We focus on scholarly documents to show how the intended use and the publishing and distribution formats influence the tools and procedures involved in document creation. We argue that scholars are increasingly demanding tools that can be tailored to their needs, which also fosters a movement towards new tools and practices.

2 WRITING TECHNOLOGY FROM A WRITER'S PERSPECTIVE

2.1 Historical development

For a long time, the final product of professional and academic writing was a physical object: printed pages. The term "word processing" was first used in the 1960s to refer to complete solutions consisting of hardware and software, when typewriters were combined with existing technologies for electronic storage of texts [for the history of word processing see, e.g., ref. 9]. Reflecting the traditional division of labor, writing and typesetting were also on computers initially two distinct stages, with typesetting programs such as troff [13] producing the formatted document from files containing a mix of text and commands. The advantage is that the document structure is explicit and that writers can, in principle, focus on writing and revising content. However, the look of the final document could only be seen once it was typeset and printed.

In the mid-1970s, programs such as Electric Pencil or EasyWriter were developed for the then new home and personal computers, but were later displaced by products like WordStar and then WordPerfect, which were in turn displaced by Microsoft Word—mainly due to bad decisions by the management of the respective companies and the market power of Microsoft, not due to features or writers' preferences. By the early 1990s, Microsoft had effectively established a monopoly with Word. [2, 3].

DocEng '22, September 20–23, 2022, Virtual Event, CA, USA

ACM ISBN 978-1-4503-9544-1/22/09
https://doi.org/10.1145/3558100.3563840

Bitmap displays with a relatively high resolution made on-screen previews possible to avoid time-consuming and expensive printing of drafts. With increasing computing power it became possible to write and edit documents directly in a form close to the final product. Bravo, developed in 1973 at Xerox PARC for the Alto workstation, is considered the first WYSIWYG word processor [7, pp. 447–448, 17, pp. 370–373]. Word is ultimately derived from Bravo. WYSIWYG word processors integrated editor and formatter, and writers assumed also the roles of typesetters and layouters. Content and form became intertwined.

The habituation of writers to Word and its perception as the standard word processor resulted in a general assumption that every new writing and editing facility must resemble the look and feel of Word and include its main features to offer a familiar user experience. This is also visible in the design and functionality of Google Docs and LibreOffice. Both also have the printed page as default final product and as default view, although Google Docs recently started to offer a "pageless" format.[1]

2.2 Research on writing tools

In the early 1980s, Rosson [21] and Whiteside et al. [28] had explored how people "really use text editors" to draw conclusions for future developments of editors. The need for such research was confirmed by Holt and Williams [10], who stated after the failure of a large project on writing support that "in order to produce computer based tools to support writers and the writing process we must increase our knowledge of how writers conduct their craft" [10, p. X]. They stressed that developing the next generation of writing tools would require a better understanding of writer's requirements and the tasks involved in writing.

However, research on writing tools and their use more or less stopped when institutions and businesses had settled on "standard" software and Word had achieved monopoly status. Word processing was considered a "solved problem."

Experimental systems developed until the late 1980s at universities had little impact as they required minicomputers or workstations, while commercial products ran on inexpensive PCs [17, 27]. Research on advanced writing support and new writing technologies in this period—e.g., RUSKIN [29], Writer's Assistant [24], Intelligent Workstation [11], and Editor's Assistant [5]—either did not take into account actual user needs (RUSKIN), or the required resources for natural language processing (NLP) were not yet mature enough to be used in real-world applications. The computing power of PCs in the early 1990s was insufficient for real-time analysis and generation which led to applications that were too limited for practical use (Intelligent Workstation and Editor's Assistant). Integration of NLP technology into word processors became a niche research topic [e.g., 4, 11, 12, 16] and did not result in commercial products.

Today's computing power and NLP resources make new ventures in this direction more promising. Since 2013, iA Writer as a commercial product features NLP-based information functions [20], in particular highlighting of different parts of speech. The developers stress the analogy to syntax highlighting in code editors, pointing out that highlighting parts of speech in a text editor fulfills a similar function, helping writers "spotting stylistic errors such as bloat through adjectives and adverbs, illogical conjunctions, weak verbs and repetitive nouns."[2]

Writing research studies writing in practical settings and works on writing pedagogy, but shows little interest in improving writing *tools* or in the development of new ones. Only occasionally the influence of the writing tool and medium are acknowledged [e.g., 15, 22]. We believe, however, that the ubiquity and variety of electronic devices, which also turns writing into an ubiquituous practice, warrants closer study.

3 THE DISSOLUTION OF THE FINAL FORM

The "paperless office" was already announced in the 1970s, but for a long time, documents continued to be distributed and consumed on paper or as print-oriented PDF files, simulacra of paper documents. Tools and computer applications were developed around this central idea, thus mimicking traditional practices and physical tools referring to their original features and affordances.

The development of the Web in the 1990s enabled and demanded *dynamic* documents with respect to both form and content. The understanding of "text" changed at the turn of the century to include "interactive, hypertextual documents—many of which reside on the Internet—[which] use color, sound, images, video, words, and icons to express their messages" [8, p. 282]. This clearly required tools to allow writers to create and edit such documents; hypertext linking represented a new challenge for authors and called for specific support.

While initially popular, WYSIWYG HTML editors (e.g., Microsoft FrontPage) modeled on Word have proven to be a dead end: unlike a printed page, a Web page is a dynamic multi-media assemblage of diverse elements, configurable to preferences of readers—and often also collecting their data. Modern Web sites are assembled on the fly by content management systems from reusable content elements; the Web page a visitor may see at any given time is nowhere stored in that form. This requires a strict separation of content and form and some form of abstract, template-based layout, which makes WYSIWYG pointless: "what you get" depends on the individual, the device, its software and configuration, etc., as well as on server-side factors.

Text structure is thus limited to a relatively small set of generic elements such as headers, paragraphs, emphasis, and links. Lightweight, wiki-style markup is therefore convenient and sufficient; what we clearly see here is the return of the explicit separation of content and form: what writers see during writing and editing is *not* what the rendered document will look like—there even is no one single target document any longer. If desired, a continously updated preview is possible, but it only shows a *potential* rendering: what the reader will see depends on the individual circumstances.

The style sheets and templates of WYSIWYG word processors are an attempt to separate form and content to some extent: they help to format documents consistently according to specific rules, but they are essentially static, as they are designed for printed output. They are also inherently alien to the WYSIWYG paradigm,

[1] https://support.google.com/docs/thread/150905607/

[2] https://ia.net/writer/support/writing-tips/parts-of-speech

which privileges direct manipulation. There is no visible difference between formatting from a style sheet and ad hoc applied formatting, and the user is free to mix them up at will, leading to errors and inconsistencies when the style sheet is changed.

The explicit separation of content and form in newer writing applications, often based on lightweight markup, such as iAWriter, Bear, or Zettlr, is more in line with the idea of "single-source publishing," which allows the author to write text once and use and reuse it for a wide variety of output formats and distribution channels; rendering is left to a later stage.

4 WRITING TOOLS: BACK TO THE FUTURE

In some respects, we now see a development back to ideas and applications of the late 1960s, before projects like NLS were "pushed aside in favor of computer systems more oriented toward print practices" [26]. While print is certainly not dead, for many, if not most, purposes it has been replaced by dynamic text on screens of varying sizes. To a large extent, the idea of the final printed document has thus become obsolete, and with it many related concepts such as pages and static page layout.

When the page-oriented document structure disappears, what remains? The structure of the *text*, i.e., the minimal structure necessary to dynamically display the text on a wide variety of displays and to interact with it [14].

For a long time, it was believed that SGML and later XML would finally be used directly by end users when WYSIWYG editors would eventually become available [19]. But it turns out that the obsolescence of the printed document as the end goal of document preparation has rendered both WYSIWYG and traditional typesetting systems, such as LaTeX, obsolete. Even though LaTeX makes it easier for authors to explicitly encode the structure of a text, it is fundamentally designed for traditional paged media.

New tools explicitly separate writing and rendering, and the rendering is delegated to the machine: new writing and editing applications are advertised as being non-WYSIWYG. Instead, there is a very strong emphasis on helping writers to "focus on the text," to offer "a unique writing experience that lets you concentrate and clarify your message," with an interface that "is crafted to cut out noise" (iAWriter). Everything else that seemed so important only 15 years earlier is left to subsequent, mostly automatic stages in the production process.

We also see experimental applications using recent technological possibilities to finally approach writing in a way earlier experiments tried and proposed but did not succeed: One such example is Tilio, which tried to implement ideas proposed by Sharples [23]. The project was cut short by the COVID-19 situation in 2020, but the technical feasibility has been shown in an alpha version, so we might see another attempt later.

Similarly, the integration of various features and services in support of *writing* rather than formatting (e.g., outlining, reference management, note-taking, data plotting, and synchronous and asynchronous collaboration and messaging) into a single application (such as Scrivener, Author, or Zettlr) can be seen as a comeback of some of the ideas of Engelbart [6]. Liberated from the requirements imposed by the printed document as final form, in which the microstructure of the text disappears behind the macrostructure of the document and its organization into pages, it now takes center stage and demonstrates the power of the computer as a writing tool that goes beyond mimicking its historical predecessors.

5 PRODUCING SCHOLARLY DOCUMENTS BEYOND THE CONCEPT OF "PAGE"

With the focus on online-only scientific publications (e.g., journals and blogs), the limits of the printed physical page and the processes involved in printing start to disappear. Tables and graphics are not restricted to particular sizes as readers are able to zoom in on such elements. More and more we see dynamically produced graphics: the research data is plotted directly into the publication. Even a step further go Jupyter Notebooks; they are living scientific publications and hold all elements essential to a particular research project: drafts on experiments, experimental setups and collected data, scripts for evaluating those data, plotting configurations to display the results graphically, and actual notes written as documentation. All this forms the final publication, all this is needed to produce a scientific document. It might feel strange for writers familiar with and accustomed to traditional practices, but all this today actually constitutes academic writing. It also realizes the goals of open science by making all data available and reproducible together with the natural language publication.

Thus a tool for academic writing in data-rich and experimental research should include support for these aspects of documentation. Via interdisciplinary and transdisciplinary projects also researchers from (so far) more traditional publishing cultures might get used to these approaches and tools.[3] Jupyter Notebooks, text editors, GitHub and GitHub-style editors, WordPress installations might have been intended for specific user groups, but the integration with other functionalities (particularly with respect to layout and styles for referencing literature) also make them useful for other user groups and purposes. Their affordances are clearly oriented towards specific text *genres* but not towards specific *content*, and besides that, they can be customized easily—which makes them more attractive than traditional word processors like Word. Users can easily add information on their writing by using metadata and established visualization techniques and create personalized writing support tailored to their needs in a very specific project as Ollivier [18] shows.

Even academic publishing by well-established publishing houses in traditional journals in non-technical disciplines moves away from physical pages: the printed journal is no longer the default distributional channel, but also its digital simulacrum—the PDF mirroring the traditional printed form—is increasingly questioned. Thus, it is probably high time to seriously considering to abandon the concept of "pages" in general. However, even in 2021, "most of today's digital document formats are 'simulating paper' based on the WYSIWYG […] principle and are therefore not fully embracing the new opportunities offered by digital media" [1], as Beat Signer (the developer of the RSL hypermedia metamodel [25] and the interactive paper platform iPaper) remarked. This also means that the authoring tools have not fundamentally changed either.

[3] A good example is the Journal of Digital History.

6 CONCLUSION

If we accept that there is no single final form of a document, we could stop interchanging rendered documents and instead distribute the content with some basic markup, possibly with simple rendering instructions. This view would also change our *conceptual understanding of documents*—and of their creation.

The print-oriented view makes the final product the focus of writing: writers aim—consciously or not—to produce a document that meets formatting requirements. In contrast, we propose to see documents solely as instantiations of content produced by writers. Instantiations of texts as documents are needed for distribution and display, but many different instantiations are possible, and writers may not even be aware of all possibilities during writing. We doubt that editing in (or close to) a distribution form actually supports the writing process. It would be preferable to focus on text structure—and publishing venues should define submission guidelines in terms of *text structure*, not in terms of typesetting or layout.

While many tools are already available today, they are often repurposed rather than explicitly designed for scholarly writing. We believe that this represents an important area of research for document engineering to answer questions such as: How to best produce these content objects? What tools are needed for supporting writing, revising, and editing content? How to distribute them, and how can we ensure that a dynamic rendering is faithful and does not suppress or alter content? And finally: What is a document?

ACKNOWLEDGEMENTS

The work of C. Mahlow has been supported by ZHAW DIZH Fellowship Call 2019.

REFERENCES

[1] Claus Atzenbeck. 2021. Interview with Beat Signer. *SIGWEB Newsletter*, Winter, Article 2, (Feb. 2021), 5 pages. DOI: 10.1145/3447879.3447881.

[2] Thomas J. Bergin. 2006. The Origins of Word Processing Software for Personal Computers: 1976–1985. *IEEE Annals of the History of Computing*, 28, 4, 32–47. DOI: 10.1109/mahc.2006.76.

[3] Thomas J. Bergin. 2006. The Proliferation and Consolidation of Word Processing Software: 1985–1995. *IEEE Annals of the History of Computing*, 28, 4, 48–63. DOI: 10.1109/mahc.2006.77.

[4] Robert Dale. 1997. Computer assistance in text creation and editing. In *Survey of the State of the Art in Human Language Technology*. Studies in Natural Language Processing. Giovanni B. Varile, Antonio Zamponelli, Ronald Cole, Joseph Mariani, Hans Uszkoreit, Annie Zaenen, and Victor Zue, (Eds.) Cambridge University Press, Cambridge, New York, Melbourne, 235–237. DOI: 10.5555/278696.278806.

[5] Robert Dale and Shona Douglas. 1996. Two Investigations into Intelligent Text Processing. In *The New Writing Environment: Writers at Work in a World of Technology*. Mike Sharples and Thea van der Geest, (Eds.) Springer, Berlin, Heidelberg, New York. Chap. 8, 123–145.

[6] Douglas C. Engelbart. 1962. Augmenting Human Intellect: A Conceptual Framework. Tech. rep. Stanford Research Institute, (Oct. 1962). http://sloan.stanford.edu/mousesite/EngelbartPapers/B5_F18_ConceptFrameworkInd.html.

[7] Richard Furuta, Jeffrey Scofield, and Alan Shaw. 1982. Document formatting systems: survey, concepts, and issues. *ACM Computing Surveys*, 14, 417–472. DOI: 10.1145/356887.356891.

[8] Cheryl Geisler et al. 2001. IText: Future Directions for Research on the Relationship between Information Technology and Writing. *Journal of Business and Technical Communication*, 15, 3, (July 2001), 269–308. http://jbt.sagepub.com/cgi/content/abstract/15/3/269.

[9] Thomas Haigh. 2006. Remembering the Office of the Future: The Origins of Word Processing and Office Automation. *IEEE Annals of the History of Computing*, 28, 4, (Oct. 2006), 6–31. DOI: 10.1109/mahc.2006.70.

[10] Patrik O'Brian Holt and Noel Williams, (Eds.) 1992. *Computers and Writing: State of the Art*. Springer Netherlands, Dordrecht.

[11] Gerard Kempen, Gert Anbeek, Peter Desain, Leo Konst, and Koenraad De Smedt. 1986. Author environments: Fifth generation text processors. In *ESPRIT'86 Results and Achievements*. The Commission of the European Communities: Directorate General XIII: Telecommunications, Information, Industries & Innovation, (Ed.) North-Holland, Amsterdam, 365–372.

[12] Gerard Kempen and Theo Vosse. 1992. A language-sensitive text editor for Dutch. In *Computers and Writing: State of the Art*. Patrik O'Brian Holt and Noel Williams, (Eds.) Kluwer, Boston, Dordrecht, London, 68–77.

[13] Brian W. Kernighan and Michael E. Lesk. 1982. UNIX document preparation. In *Document Preparation Systems*. Jürg Nievergelt, Giovanni Coray, Jean-Daniel Nicoud, and Alan C. Shaw, (Eds.) North-Holland, Amsterdam, 1–20.

[14] Cerstin Mahlow. 2022. Structure! You get more than you see. In *XML Prague 2022 – Conference Proceedings*. XML Prague 2022 (Prague, June 9–11, 2022). Jirka Kosek, (Ed.), 125–140. https://archive.xmlprague.cz/2022/files/xmlprague-2022-proceedings.pdf#page=135.

[15] Cerstin Mahlow and Robert Dale. 2014. Production Media: Writing as Using Tools in Media Convergent Environments. In *Handbook of Writing and Text Production*. Handbooks of Applied Linguistics. Vol. 10. Eva-Maria Jakobs and Daniel Perrin, (Eds.) De Gruyter Mouton, Berlin, Germany, (Feb. 2014), 209–230.

[16] Cerstin Mahlow and Michael Piotrowski. 2009. LingURed: Language-Aware Editing Functions Based on NLP Resources. In *Proceedings of the International Multiconference on Computer Science and Information Technology*. Vol. 4. Polish Information Processing Society. Mragowo, Poland, (Oct. 2009), 243–250. http://www.proceedings2009.imcsit.org/pliks/101.pdf.

[17] Norman Meyrowitz and Andries van Dam. 1982. Interactive editing systems: Part I. *ACM Computing Surveys*, 14, 3, (Sept. 1982), 321–352. DOI: 10.1145/356887.356889.

[18] Tony Ollivier. 2001. "Give Me A Break"—better pacing with Scrivener. online on Medium. (Feb. 2001). https://tonyollivier.medium.com/f9acff1b22d1.

[19] Michael Piotrowski. 2019. History and the future of markup. In *Proceedings of XML Prague 2019*. Jirka Kosek, (Ed.) Prague, 323–333. http://archive.xmlprague.cz/2019/files/xmlprague-2019-proceedings.pdf#page=335.

[20] Michael Piotrowski and Cerstin Mahlow. 2009. Linguistic editing support. In *DocEng'09: Proceedings of the 2009 ACM Symposium on Document Engineering*. ACM, New York, NY, USA, (Sept. 2009), 214–217.

[21] Mary B. Rosson. 1983. Patterns of experience in text editing. In *CHI '83: Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. ACM, Boston, Massachusetts, United States, 171–175. DOI: 10.1145/800045.801604.

[22] Mike Sharples. 1996. An Account of Writing as Creative Design. In *The Science of Writing. Theories, Methods, Individual Differences, and Applications*. C. Michael Levy and Sarah Ransdell, (Eds.) Lawrence Erlbaum, Hillsdale, NJ, USA, 127–148.

[23] Mike Sharples. 1999. *How We Write: Writing As Creative Design*. Routledge, New York, NY, USA.

[24] Mike Sharples and Lyn Pemberton. 1990. Starting from the Writer: Guidelines for the Design of User-Centred Document Processors. *Computer Assisted Language Learning*, 2, 1, 37–57.

[25] Beat Signer. 2010. What is wrong with digital documents? a conceptual model for structural cross-media content composition and reuse. In *Conceptual Modeling – ER 2010*. Jeffrey Parsons, Motoshi Saeki, Peretz Shoval, Carson Woo, and Yair Wand, (Eds.) Springer Berlin Heidelberg, Berlin, Heidelberg, 391–404.

[26] Derek Van Ittersum. 2008. Computing Attachments: Engelbart's Controversial Writing Technology. *Computers and Composition*, 25, 2, 143–164. DOI: 10.1016/j.compcom.2007.12.001.

[27] Alex Vernon. 2000. Computerized grammar checkers 2000: Capabilities, limitations, and pedagogical possibilities. *Computers and Composition*, 17, 3, (Dec. 2000), 329–349. DOI: 10.1016/s8755-4615(00)00038-4.

[28] John Whiteside, Norman Archer, Dennis Wixon, and Michael Good. 1982. How do people really use text editors? *ACM SIGOA Newsletter*, 3, 1-2, 29–40. DOI: 10.1145/966873.806474.

[29] Noel Williams. 1990. Writers' Problems and Computer Solutions. *Computer Assisted Language Learning*, 2, 1, 5–25.