

Making Books

Henning Hraban Ramm

This is a response to Keith’s article in this journal, as well as a summary of my talks at ConT_EXt meetings about publishing books.

My background

My first ConT_EXt project in 1999 was a reader for the German Unitarian Church¹. I was used to QuarkXPress, but I wanted to try T_EX, because it could create an index; in QuarkXPress I would had to create it manually. I knew about T_EX because when I was about 16 an older friend started studying physics. A while later, he proudly showed me what he could do on the shell of a UNIX (probably MINIX). I was not impressed, because I was used to CP/M and DOS and didn’t understand the superiority of the UNIX toolset. Then, he and a math student started submitting articles for our Unitarian youth magazine, typeset with T_EX. I got used to the look of Computer Modern without ever attending a university.

I still did most of my work with QuarkXPress (up to 4.x) and InDesign (since 2.0), but ConT_EXt became more and more important.

After another book² for the Unitarian publishing house (my father), I typeset three books³ for a friend’s shamanism-focused publishing house⁴. Two of these needed intricate image placement – I learned to use postponing this way.

After I met my present business partner, Benedikt, in Bishkek, I typeset my first architectural guide⁵ of the Bonn series (actually an outlier about a church in Wittlich), written by Benedikt’s father in 2015. Our architectural guide for Bishkek⁶ was made in InDesign. It would have made no sense to try with T_EX. But I used OpenStreetMap maps for the first time, with the programmatic workflow outlined in my talk and article “Architectural Guides for Bonn”⁷. While I worked at the book, I made many contributions to OpenStreetMap.

Benedikt was involved with the magazine for German literary studies, “Kritische Ausgabe” (KA), that started as a student project at Bonn university. The architec-



Fig. 1. My first architectural guide

¹ Was glauben sie eigentlich? Die Deutschen Unitarier, eine freie Religionsgemeinschaft. Verlag Deutsche Unitarier 2000, Ravensburg.

² Kofi Annan: UNvollendeter Weg. Verlag Deutsche Unitarier 2003, Ravensburg.

³ Nebenwelt – Alte und neue Feengeschichten. Maple Tree 2012, Konstanz. / Adrian Oswald: Die Weisse Eule – ein schamanisches Ritual aus dem Hier und Jetzt. Maple Tree 2014, Konstanz. / Annette Schwarz von Specht: Mip und die Baumleute. Maple Tree 2015, Konstanz.

⁴ Can you still call it a ‘house’, if it’s not even a room?

⁵ Wolfram Viertelhaus: St. Paul. Werkstatt Baukultur 2015, Bonn.

⁶ H. Ramm & B. Viertelhaus (ed.): Architekturführer Bishkek. DOM publishers 2016, Berlin.

⁷ TUGboat #44, CG Journal 2023

tural guides were a side project of it. Another project was “Edition KA”, a series of monographs in the same field as the magazine. When I typeset the last volume⁸ that was published by a different publisher, Weidle, I used InDesign, because the book had a lot of photos and the chapter layout varied.

These InDesign projects are still relevant, since when we founded our publishing house, we took over Edition KA, the architectural guides, as well as the design and distribution of the magazine (its editorial staff was still based at Bonn University, but fell asleep during the pandemic).

Our publishing house

Benedikt Viertelhaus, Marcel Diel, and I founded Dreiviertelhaus⁹ in 2017. The other two studied German literature in Bonn and are working as booksellers (Benedikt has a shop in Berlin, Marcel works for the online department of a chain), while I studied typesetter and printing engineer¹⁰ and work as a technical editor for a machine factory. We mostly moonlight and live in different places across Germany.

Fig. 2. Benedikt, Hraban, and Marcel in their tiny booth at Leipzig book fair

We started with several small architectural guides and three volumes in the Edition KA series, all typeset with ConT_EXt. I love scientific typesetting. The liberal-arts-oriented books weren't planned with extensive indexes, but I created these because I could. In one book, the author even learned something, because I found the same person mentioned in different places, where he didn't see a connection before.

We never sold many copies of our literary studies books or the magazine. Architecture finds a bigger audience. During the pandemic we started publishing fiction which sells better. We were lucky to work with some exceptional authors. Even so, we are happy if we can sell a few hundred copies. That's nothing in comparison with bigger houses, but we have no agents and can't afford a lot of advertising. Our tiny booth at Leipzig book fair eats up most of our annual profit. So we do this for fun, but we do it professionally – similar to the developers of ConT_EXt. That also means we have contracts with the three German wholesalers where most of the booksellers order. (Micro- or selfpublishers usually can only offer direct distribution.)

Authors hardly ever get rich; with a micropublisher like us, even less. Only authors that regularly do readings or other performances and have some fellowship might make a hundred Euros a year from their books. They get a bit extra from VG WORT, the German collecting society for text rights that manages the earnings from photocopy tax ('Kopierabgabe').

⁸ Bonn und seine Preußen. Danke, Berlin!? Werkstatt Baukultur/Edition KA. Weidle 2016, Bonn.

⁹ The name means “three quarters house”, since there are three of us and one of us is called Viertelhaus ('quarter house'); we just couldn't *not* use that name for an architectural publisher.

¹⁰ My title is “state-certified technician”: more than a “master” and less than an “engineer”.

Our workflow

Benedikt and Marcel take turns as editor and proofreader. They work with the author until the text is polished to everyone’s satisfaction. They use comments and change tracking in LibreOffice or MS Word. The amount of editing required depends on the authors and projects. Often, we need to completely rewrite the works of our technical authors (apparently, architects are rarely also good writers), while we try to avoid making changes to stories. Most authors are happy to work with us.

My editors are also advised to apply paragraph styles systematically.

When they are finished, they send me a DOCX file. The only comments left are for me, regarding typesetting.

When I was using InDesign to produce a book, I would copy plain text from the DOCX file into InDesign and apply the styles manually. I did the same to produce my first books using ConT_EXt, but it was a lot of tedious work. Now I have written a Python script, `docx2ctx`, to automate parts of this process.

`docx2ctx`¹¹

`docx2ctx` reads a DOCX file and produces ConT_EXt code. The book’s layout is pulled in from an environment file.

When I wrote `docx2ctx`, I first looked for a Python library to read either DOCX or ODT files. The latter format is probably better documented. I tried a few and `mammoth`¹², a DOCX to HTML converter, seemed to work best. But I found that it threw away some metadata that I wanted to use. I continued to work with DOCX files but wrote my own code.

Like several other file formats, DOCX and ODT files are actually ZIP archives containing XML files and sometimes images. My script understands and converts word markup (bold, italic), headers, lists, footnotes/endnotes, comments, tables, fonts, colours and images. If it finds items it doesn’t understand, it flags them in the ConT_EXt file; I search for these and enter the ConT_EXt code manually. My script also helps with typography; for example, it replaces double dashes with en dashes and adds a `\`, (thinspace) between the letters of an abbreviation.

The text markup in the XML files is usually a mess: language, font and colour markers appear at will, and, for example, while you would expect words marked as italics as `<i>a few words</i>`, they look mostly like `<i>a</i> <i>me</i><i>ss</i>`. Tables are particularly hard to convert to ConT_EXt, since table cells usually contain paragraphs and other kinds of markup. In the few cases where I need to convert tables, I usually write them anew. My script doesn’t try to keep list styles or any image properties – I enter these manually because I usually don’t want my book to look like the DOCX file.

¹¹ codeberg.org/fiee/context-tools/

¹² github.com/mwilliamson/python-mammoth

contextgroup > context meeting 2024

When I need additional markup, I ask the author to use a colour, because it is easy for me to convert colour marks to other ConT_EXt code. For example, if there is an index I ask them to mark items for the index with yellow. It is then easy for me to search and replace `\color[yellow]` by `\index` afterwards.

Internally, the script uses `ElementTrees` for metadata and links, but a SAX parser for the main document. (These are different approaches of handling XML data.) The event handler functions for the single tags directly produce ConT_EXt code, the result is a T_EX document that gets refined using regular expression replacements. That's probably not the cleanest approach, but good enough for me.

`docx2ctx` has many command line options (for example, to ignore colours and fonts) and it can use templates for components etc. Otherwise, it is not easily configurable for custom projects.

Finalising the layout

While the ConT_EXt output from `docx2ctx` is more or less complete, it needs the usual setups. I always use ConT_EXt's project structure – if a book is part of a series, this makes sense anyway, but I handle single works the same way. My environment is split up into several parts, such as common basics, layout, typography, colours and images. For the cover, I only load what is necessary. The way I create covers is covered in my article “Calculating Covers”¹³. The result is a one-page open-format PDF. If we have the inner side of the cover printed, the PDF has two pages.

I usually use ConT_EXt modes to produce PDFs for ‘print’ and for ‘ebook’. The print PDF includes high-resolution images, crop marks and bleed. The ebook PDF is also used for proof copies. It includes low-resolution images and the cover pages. I use clipping in ConT_EXt to generate these cover pages from the cover PDF.

I use extreme `hz` (horizontal character width compensation) and since 2024 have been enjoying ‘granular’ paragraph breaking and `vz` (vertical line height compensation). While I like grid setting for its traditional purity, it makes page breaking without widows and orphans very hard – and for me, avoiding single lines is more important than other typographical niceties. In technical texts, I can just change some words to make a line fit. If I can't do this, for example, in a novel, I have some dirty tricks: I might change the letter spacing (minimal, positive or negative values), or I might slightly change the text width for a page by changing the page layout. Unfortunately, that works only with manual page breaks.

Other parts of the toolset

I try to use open source software for my work. Images, usually photos, are an important part of our non-fiction books, and photo editing is the one area where I still depend on a commercial program¹⁴. I can't cope with the GIMP (well, I should try

¹³ CG Journal 2022, TUGboat #44:2, and DTK 1/2023 (in German).

¹⁴ Affinity Photo, since my old Adobe CS licenses don't work anymore.

again), and Krita is a great painting program but not suitable for prepress image manipulation.

The other ‘missing link’ in open source software is a PDF editor with prepress functions, such as crop and bleed manipulation, flattening transparencies, and PDF/X conversion. The only Linux program that comes close is Qoppa’s commercial PDF Studio Pro.

In our correction workflow, we add annotations to the typeset PDF. If I have questions or suggestions, I can add remarks to my ConT_EXt code (using `\comment` or `\PDFhighlight`), while authors and editors add their annotations to the proof copies. Foxit Reader (MacOS and Windows) is the most reliable viewer for PDFs with annotations, and it supports annotation threads. Adobe Reader crashes too often. I know of no others with good support. There’s no open source PDF viewer with usable support for annotations – Okular is the best but still cumbersome.

It would be useful to handle annotations automatically, with these three steps: extract annotations from a PDF, use SyncT_EX data to find their location in my ConT_EXt code, and insert them into my T_EX sources. The first step was recently implemented: `mtxrun --script pdf --comments filename.pdf` (or `--highlights`). The second step fails because ConT_EXt’s SyncT_EX data is unusable¹⁵. When this is fixed, the third step should be easy. I’m looking forward to even more automation in book production!

After completion of a book, I run the “print” PDF through the PDF/X checker in PDF Studio Pro. I don’t trust in it, but the only alternative would be Acrobat Pro. Hardly any of the problems that I had with printing were caused by data quality.

For example, with one book, the whole text was printed in black, including the parts that should have been grey or white. Finally the printer found the wrong setting in his digital printing press that went unnoticed since he only ever had black text or coloured pages before. For the second print run, he misadjusted the cutting machine...

Last words

Finalizing this article in June 2025, I finally fixed my ePub workflow that was broken for years. But several more volumes of architectural guides as well as belletristics are being published using the proven and only slightly updated PDF workflow.

¹⁵ As of 2024-11: The file is right, but the location is irregularly off by several lines.