# The Making-of

*This publication was produced with a set of digital tools that are rarely used outside the world of scientific publishing: TEX, LATEX and ConTEXt. As early as the summer of 2008, when most contributions and translations to* Tracks in electronic fields *were reaching their final stage, we started discussing at OSP*[1] *how we could design and produce a book in a way that responded to the theme of the festival itself. OSP is a design collective working with Free Software, and our relation to the software we design with, is particular on purpose. At the core of our design practice is the ongoing investigation of the intimate connection between form, content and technology. What follows, is a report of an experiment that stretched out over a little more than a year.*

For the production of previous books, OSP used Scribus, an Open Source Desktop Publishing tool which resembles its proprietary variants PageMaker, InDesign or QuarkXpress. In this type of software, each single page is virtually present as a ‘canvas’ that has the same proportions as a physical page and each of these ‘pages’ can be individually altered through adding or manipulating the virtual objects on it. Templates or ‘master pages’ allow the automatic placement of repeated elements such as page numbers and text blocks, but like in a paper-based design workflow, each single page can be treated as an autonomous unit that can be moved, duplicated and when necessary removed. Scribus would have certainly been fit for this job, though the rapidly developing project is currently in a stage that the production of books with more than 40 pages can become tedious. Users are advised to split up such documents into multiple sections which means that in able to keep continuity between pages, design decisions are best made beforehand. As a result, the design workflow is rendered less flexible than you would expect from state-of-the-art

[1] Open Source Publishing http://ospublish.constantvzw.org

creative software. In previous projects, Scribus' rigid workflow challenged us to relocate our creative energy to another territory: that of computation. We experimented with its powerful Python scripting API to create 500 unique books. In another project, we transformed a text block over a sequence of pages with the help of a fairy-tale script. But for *Tracks in electronic fields* we dreamed of something else.

Pierre Huyghebaert takes on the responsibility for the design of the book. He had been using various generations of lay-out software since the early 90's, and gathered an extensive body of knowledge about their potential and limitations. More than once he brought up the desire to try out a legendary typesetting system called TEX[1] a sublime typographic engine that allegedly implemented the work of grandmaster Jan Tshichold[2] with mathematical precision.

TEX is a computer language designed by Donald Knuth in the 1970's, specifically for typesetting mathematical and other scientific material. Powerful algorithms automatize widow and orphan control and can handle intelligent image placement. It is renowned for being extremely stable, for running on many different kinds of computers and for being virtually bug free. In the academic tradition of free knowledge exchange, Knuth decided to make TEX available 'for no monetary fee' and modifications of or experimentations with the source code are encouraged. In typical self referential style, the near perfection of its software design is expressed in a version number which is converging to $\pi$[3].

For OSP, TEX represents the potential of doing design differently. Through shifting our software habits, we try to change our way of working too. But Scribus, like the kinds of proprietary softwares it is modeled on, has a 'productionalist' view of design built into it[4], which

2 In *Die neue Typographie* (1928), Jan Tschichold formulated the classic canon of modernist bookdesign.

3 The value of $\Pi$ (3.141592653589793...) is the ratio of any circle's circumference to its diameter and it's decimal representation never repeats. The current version number of TEX is 3.141592

4 "A DTP program is the equivalent of a final assembly in an industrial process" Christoph Schäfer, Gregory Pittman et al. *The Official Scribus Manual.*Fles Books, 2009

is undeniably seeping through in the way we use it. An exotic Free Software tool like TEX, rooted firmly in an academic context rather than in commercial design, might help us to re-imagine the familiar skill of putting type on a page. By making this kind of 'domain shift'[5] we hope to discover another experience of making, and find a more constructive relation between software, content and form. So when Pierre suggests that this V/J10 publication is possibly the right occasion to try, we respond with enthusiasm.

By the end of 2008, Pierre starts carving out a path in the dense forest of manuals, advice, tips-and-tricks with the help of Ivan Monroy Lopez. Ivan is trained as mathematician and more or less familiar with the exotic culture of TEX. They decide to use the popular macro-package LATEX[6] to interface with TEX and find out about the tong-in-cheek concept of 'badness' (depending on the tension put on hyphenated paragraphs, compiling a .tex document produces 'badness' for each block on a scale from 0 to 10.000), and encounter a long history of wonderful but often incoherent layers of development that envelope the mysterious lasagna beauty of TEX's typographic algorithms.

Laying-out a publication in LATEX is an entirely different experience than working with a canvas-based software. First of all, design decisions are executed through the application of markup which vaguely reminds of working with CSS or HTML. The actual design is only complete after 'compiling' the document, and this is where TEX magic happens. The software passes several times over a marked up .tex file, incrementally deciding where to hyphenate a word, place a paragraph or image. In principle, the concept of a page only applies after compilation is complete. Design work therefore radically shifts from the act of absolute placement to co-managing a flow. All elements remain relatively placed until the last *tour* has passed, and while error messages, warnings and hyphenation decisions scroll by on the command line, the sensation of elasticity is almost tangible. And

[5] See: Richard Sennett. *The Craftsman.* Allen Lane (Penguin Press), 2008

[6] LATEX is a high-level markup language that was first developed by Leslie Lamport in 1985. Lamport is a computer scientist also known for his work on distributed systems and multi-treading algorithms.

indeed, when within the acceptable 'stretch' of the program placement of a paragraph is exceeded, words literally break out of the grid (see page 34 example).

When I join Pierre to continue the work in January 2009, the book is still far from finished. By now, we can produce those typical academic-style documents with ease, but we still have not managed to use our own fonts[7]. Flipping back and forth in the many manuals and handbooks that exist, we enjoy discovering a new culture. Though we occasionally cringe at the paternalist humour that seems to have infected every corner of the TEX community and which is clearly inspired by witticisms of the founding father, Donald Knuth himself, we experience how the lightweight, flexible document structure of TEX allows for a less hierarchical and non-linear workflow, making it easier to collaborate on a project. It is an exhilarating experience to produce a lay-out in dialogue with a tool and the design process takes on an almost rhythmical quality, iterative and incremental. It also starts to dawn on us, that *souplesse* comes with a price.

"Users only need to learn a few easy-to-understand commands that specify the logical structure of a document" promises *The Not So Short Introduction to LATEX*. "They almost never need to tinker with the actual layout of the document". It explains why using LATEX stops being easy-to-understand once you attempt to expand its strict model of 'book', 'article' or 'thesis': the 'users' that LATEX addresses are not designers and editors like us. At this point, we doubt whether to give up or push through, and decide to set ourselves a limit of a week in which we should be able to to tick off a minimal amount of items from a list of essential design elements. Custom page size and headers, working with URL's... they each require a separate 'package' that may or may not be compatible with another one. At the end of the week, just when we start to regain confidence in the usability of LATEX for our purpose, our document breaks beyond repair when we try to use custom paper size with custom headers at the same time.

[7] "Installing fonts in LATEX has the name of being a very hard task to accomplish. But it is nothing more than following instructions. However, the problem is that, first, the proper instructions have to be found and, second, the instructions then have to be read and understood". http://www.ntg.nl/maps/29/13.pdf

In February, more than 6 months into the process, we briefly consider switching to OpenOffice instead (which we had never tried for such a large publication) or go back to Scribus (which means for Pierre, learning a new tool). Then we remember ConTeXt, a relatively young 'macro package' that uses the TeX engine as well. "While LaTeX insulates the writer from typographical details, ConTeXt takes a complementary approach by providing structured interfaces for handling typography, including extensive support for colors, backgrounds, hyperlinks, presentations, figure-text integration, and conditional compilation"[8]. This is what we have been looking for.

ConTeXt was developed in the 1990's by a Dutch company specialised in 'Advanced Document Engineering'. They needed to produce complex educational materials and workplace manuals and came up with their own interface to TeX. "The development was purely driven by demand and configurability, and this meant that we could optimize most workflows that involved text editing"[9].

However frustrating it is to re-learn yet another type of markup (even if both are based on the same TeX language, most of the LaTeX commands do not work in ConTeXt and vice versa), many of the things that we could only achieve by means of 'hack' in LaTeX, are built in and readily available in ConTeXt. With the help of the very active ConTeXt mailinglist we find a way to finally use our own fonts and while plenty of questions, bugs and dark areas remain, it feels we are close to producing the kind of multilingual, multi-format, multi-layered publication we imagine *Tracks in Electr(on)ic Fields* to be.

However, Pierre and I are working on different versions of Ubuntu, respectively on a Mac and on a PC and we soon discover that our installations of ConTeXt produce different results. We can't find a solution in the nerve-wrackingly incomplete, fragmented though extensive documentation of ConTeXt and by June 2009, we still have not managed to print the book. As time passes, we find it increasingly

[8] Interview with Hans Hagen http://www.tug.org/interviews/interview-files/hans-hagen.html

[9] Interview with Hans Hagen http://www.tug.org/interviews/interview-files/hans-hagen.html

difficult to allocate concentrated time for learning and it is a humbling experience that acquiring some sort of fluency seems to pull us in all directions. The stretched out nature of the process also feeds our insecurity: Maybe we should have tried this package also? Have we read that manual correctly? Have we read the right manual? Did we understand those instructions really? If we were computer scientists ourselves, would we know what to do? Paradoxically, the more we invest into this process, mentally and physically, the harder it is to let go. Are we refusing to see the limits of this tool, or even scarier, our own limitations? Can we accept that the experience we'd hoped for, is a lot more banal than the sublime results we secretly expected? A fellow Constant member suggests in desperation: "You can't just make a book, can you?"

In July, Pierre decides to pay for a consult with the developers of ConTEXt themselves, and once and for all solve some of the issues we continue to struggle with. We drive up expectantly to the headquarters of Pragma in Hasselt (NL) and discuss our problems, seated in the recently redecorated rooms of a former bank building. Hans Hagen himself reinstalls markIV (the latest in ConTEXt) on the machine of Pierre, while his colleague Ton Otten tours me through samples of the colorful publications produced by Pragma. In the afternoon, Hans gathers up some code examples that could help us place thumbnail images and before we know it we are on our way South again. Our visit confirms the impression we had from the awkwardly written manuals and peculiar syntax, that ConTEXt is in essence a one man mission. It is hard to imagine that a tool written to solve particular problems of a certain document engineer, will ever grow into the kind of tool that we desire too as well.

In August, as I type up this report, the book is more or less ready to go to print. Although it looks 'handsome' according to some, due to unexpected bugs and time restraints, we have had to let go of some of the features we hoped to implement. Looking at it now, just before going to print, it has certainly not turned out to be the kind of eye-opening typographic experience we dreamt of and sadly, we will never know whether that is due to our own limited understanding of TEX, LATEX and ConTEXt, to the inherent limits of those tools themselves, or to the crude decision to finally force through a lay-out in two weeks. Probably a mix of all of the above, it is first of all a relief that the publication finally exists. Looking back at the process, I am reminded of the wise words of Joseph Weizenbaum, who observed that “Only rarely, if indeed ever, are a tool and an altogether original job it is to do, invented together”[10].

While this book nearly crumbled under the weight of the projections it had to carry, I often thought that outside academic publishing, the power of TEX is much like a Fata Morgana. Mesmerizing and always out of reach, TEX continues to represent a promise of an alternative technological landscape that keeps our dream of changing software habits alive.

Femke Snelting (OSP), August 2009

[10] Joseph Weizenbaum. *Computer power and human reason: from judgment to calculation.* MIT, 1976