Typeface as Programme

The Nature of Type Design in the Digital Age

Written by Jürg Lehni for the book Typeface as Program, published in 2009 by ECAL and JRP.

Like many disciplines dependent on technology for execution or production, type design has undergone a series of fundamental revolutions and transitions in the past century. Driven by technological advance, this process has completely changed the way people work with type, to the point where someone employed in the field had to adapt to a significantly changing situation multiple times throughout a career. The change went from 19th century hot metal typesetting with its very complex and expensive mechanized equipment invented by Monotype and Linotype, through a period of opto-mechanical photocomposition systems, in which printing with cast letter-forms was replaced with exposure of optical outlines on spinning disks of glass onto light-sensitive paper, to the digital simulation of similar processes, formulated in computer programs and executed first by huge room-filling installations and later by affordable home computers.

The advent of computer technology and the digital revolution has had similar impacts on many other creative fields, such as graphic design, photography, film editing, or audio recording, with changes often similar in nature. Highly expensive equipment was made redundant by computer technology running software that simulates the same processes. The software and the user interfaces often uses metaphors from within the field known from the time before the revolution, and the role of the computer is that of a machine simulating other machines or processes as a sort of a meta-tool. Even today, software is largely defined as that, and therefore computers function mostly as replacements for previously existing processes, the type-writer and postal service being two of the most common examples.

Democratization is another important part of these developments. The sudden general availability of processes through computerization has continued to increase the number of people who have access to and start engaging in them. In the

creative sector, this also led to a change in the nature of the work being done, often to the disapproval of the previous specialists in the field. While type design in the 19th century was a craft accessible to very few selected typographers who together with punch cutters worked on designs for one of the companies producing typesetting equipment, it is now a discipline that anyone who has access to a computer and a license for a type design software can engage in.

These are generally known aspects of this revolution that have been looked at closely many times before. But the role of software is rarely analyzed beyond this point. It appears that the general function of the computer is still accepted simply as a simulation machine, and the question what software could or should provide in any given field is rarely raised. Instead, the status quo is often accepted as a given, a language we use in our daily work and that we have stopped questioning, since it is so ubiquitous that is it almost invisible.

Furthermore, in the historic discourse of digital typefaces, questions regarding the definition and nature of digital typefaces are hardly risen and the status quo is rarely questioned beyond the boundaries of the industrial standards.

Fonts, Tools and Software

Traditionally, a font was a complete set of metal characters of a particular typeface in a given size and style. Etymologically, the word goes back to the French word fonte and the verb fondre, meaning to melt or to cast, referencing the way fonts were produced by type foundries. Fonts were one of the ingredients needed in the printing process in order to be able to print text, and they were bought in full sets from the foundries. A set included more copies of some letters than others, depending on the statistical occurrence of each letter in any given language. The structure of the letter cases that hold the letters represented this distribution.

A font was not a full, independent tool in itself, but rather a part of a tool based process, which, without it could not take place. Given its physical nature at that time, it is imaginable that fonts were perceived as tools in themselves. At the same

time they could also be seen as an artwork designed by a typographer and executed by a punch cutter.

Today, digital fonts are legally defined as software, once again as the digital counterpart of a tool. This has broad consequences for the way fonts are distributed and sold, and the way type designers are earning their money, since licensing schemes similar to the ones found in software applications are in place: The End User License Agreements (EULA) entitle the end users of fonts to install them on a defined number of computers within the same household or office. The degree of usage of the font in this case has no impact on the price. As soon as the user has bought the license, he owns the right of usage within the defined boundaries and therefore can use the font as a tool as much as he likes, as long as he does not infringe the rules of the agreement. This leads for example to the absurd situation where a big TV company in certain circumstances may pay the same amount of money for a font that is aired daily for years on millions of TV screens as a small graphic design office that uses the font once for a client's job. Both buy the basic right to use the font as a tool for whatever they need it for, and the creative work in the typeface is unaccounted for.

While there are foundries that have created complicated agreements for such special cases, the basic problem of unequal usage remains and is criticised by many type designers: the fact that the creative work is not taken into account in the definition as a tool, ignoring the fact that a typeface is also an artistic work by a creative individual.

An alternative way of defining typefaces is as library or a family of graphical shapes along with rules that describe how to assign these to letters, and how to adjust the space between them. If this definition was used legally, another system would suggest itself: one based on royalties, as in the music industry or applied photography, both fields where an artwork or a composition is licensed for specific media based distribution. The licensing costs then mostly depend on the duration of the segment, the size of the image, visibility, distribution, etc. Specific associations claim these royalties and distribute them among their members, enforcing copyright law and ensuring rights of authorship for the protected works.

Such authorship based systems are not necessarily a viable way for typefaces, as they have their own share of problems in the digital age, namely software piracy. Digital Rights Management (DRM) as a possible solutions proposed by big corporations is in the process of failing and is mostly being abandoned at the moment of writing, since the consumers are not willing to follow to the rules they force upon them. Nevertheless it remains curious that this legal definition as software has been chosen, especially since there is little evidence that digital typefaces actually really work as software.

While it is true that the technologies used today for the digital definition of typefaces such as PostScript hold qualities of software and programming languages, there is little evidence that the file formats that describe typefaces such as Type 1 and OpenType actually take advantage of this flexibility. PostScript is a page description language developed by Adobe Systems Inc. In order to offer the greatest flexibility and future scalability, it was designed as a full programming language. Type 1 are the type-specific aspects of this language, and just like the rest of PostScript, typefaces in PostScript are formulated as sequences of program code. But since these codes describe graphical shapes and lists of sizes and spacing between characters, there is little reason that it really should be considered software. The recent introduction of a new open font format named Unified Font Object (UFO) that is entirely based on XML descriptions proves that all this information can be stored without being written as software, since XML is a descriptive markup language like HTML, not a programming language. TrueType is omitted in this comparison as basically the same applies to that format, and both Type 1 and TrueType formats are now merged in the more recent OpenType standard.

Another line of reasoning is that if typefaces were full software, they would not have to rely on a computer operating system (OS) and its underlying typesetting mechanisms. Just like the metal fonts that were an ingredient for a typesetting-machine, the digital fonts are data for a host software that knows how to read it and lay it out.

So if typefaces are legally defined as software, but are not currently behaving like software, this raises questions: Does the current definition of digital typefaces hold

unused potential? Could or should digital type design incorporate the possibilities of software more?

Approaches to Typefaces as Software

The process of digitalization and computerization of type-oriented technology is probably a never ending one since new innovative approaches are continuously being found for how to draw and produce type designs, but the most fundamental changes and revolutions in the field have happened, and the process of software standardization is largely completed. At the beginning of this process, there was the question of how typesetting is best represented in software and executed or output by printing devices. With the introduction of pixel based display technology such as CRT monitors, there was also the problem of how to represent glyph outlines appropriately on such low resolution devices and not loose the font's main characteristics. There were many different proposals, and through a slow process of selection and advancement, some of them were abandoned while others merged and became standards.

This exciting time of technical innovation has lead to many different efforts and resulting systems, but now at the end of this process of standardization, there is primarily one system the whole industry is focused on: the previously mentioned OpenType, a standard coined by Microsoft together with Adobe Systems as a result of the "Type War" between Apple's TrueType standard and Adobe System's PostScript. Microsoft, who previously licensed the TrueType technology from Apple, decided to move ahead and create their own standard based on TrueType in the early 1990s, after negotiations with Apple to license their advanced typography technology called "GX Typography" failed. Adobe Systems joined in 1996 and added support for the glyph outline descriptions based on its PostScript's Type 1 fonts. In 2005, OpenType started migrating to an open standard under the International Organization for Standardization (ISO) and the process was completed in 2007 when it was accepted as a free, publicly available standard.

This system has become the standard for type on most of today's modern operating systems such as Mac OS X, Windows and Linux, and most typesetting applications

support its extended features. But there is a rather large niche in which one of the other proposals from the period of early digital type technology has survived until today: The typesetting system TeX with its spin-off project LaTeX, a collection of macros to simplify TeX, and its font system Metafont, used mostly in academia, especially in the mathematics, computer science, and physics communities. Both TeX and Metafont were conceived and designed by highly acclaimed computer scientist Donald E. Knuth as a solution to the problems of typesetting complex mathematical formulas and more generally scientific publications. TeX has been noted as one of the most sophisticated digital typographical systems in the world. TeX (and therefore LaTeX) have adapted to the same wider spread font standards mentioned above. Nevertheless Metafont is still relevant, as it is largely unknown in the domain of type design and has a history that is still of interest for more recent experiments in programmatic type design based on the principles of parametric variations.

TeX and Metafont as an Example of Early Parametrised Digital Typesetting

As the author of the highly acclaimed monograph The Art of Computer Programming, listed by the American Scientist as one of the 12 best physical-science monographs of the 20th century, Donald E. Knuth was always concerned with the printed appearance of his works and fascinated by the technical facilities and the skills of their operators. The quality of the first three published volumes of his monograph, all typeset in Monotype Modern 8A on mechanical hot type machines from the same company provided great satisfaction.

When in 1977 due to financial restrictions the new edition of volume 2 was to be reproduced with a new optical typesetting system rather than the already disappearing Monotype machines, he saw his world collapse. The optical typesetting systems mostly used typefaces of minor quality, copied or derived from the ones carefully designed by typographers for Monotype and other hot type foundries. For Knuth the beautifully typeset texts and mathematical equations were not simply a feature nice to have, they were part of his motivation to actually write these books. Knuth was obsessed with the beauty of his printed works. In the

introduction to Computer Modern Typefaces, the book about a family of parametrizable fonts that he developed using his Metafont system, he says he has "ink running through his veins." If his works were going to be badly type set, he decided there was no point in continuing to write them since the finished products were just too painful to look at. He realized he would have to do something about it.

Excited by the impending technological revolution in print that would bring digital printing at a high enough resolution that the pixels would not be visible to the human eye, he decided to come up with a new system that would correctly compose typography in pixels, independent from machines and their resolution. The little squares that either contain 1 or 0, to represent ink or no ink, he concluded, were part of his realm as a computer scientist, so he assumed he should be able to come up with a better solution rather quickly. Knuth reasoned that if solved properly, this work could be of use for a very long time, since this basic principle of pixels as the core component of digital printing would not change, no matter how much the technology surrounding it does.

All this happened before Adobe Systems was founded and the base for the page description language PostScript was laid out. At the beginning of this endeavour, Knuth did not even have the possibility to see the results on screen. Each time he wanted to try out a change in the software he had to make digital prints on a facility without easy or regular access.These huge devices were very expensive to run and an acquisition only made sense for large corporations with continuous use.

In 1978 Knuth was invited to speak in the prestigious Josiah Willard Gibbs Lecture, established by the American Mathematical Society (AMS) in 1923 and held annually to to increase public awareness of the aspects of mathematics and its applications (http://en.wikipedia.org/wiki/Mathematics). Knuth quite bravely decided that instead of speaking purely about mathematics or algorithms, his talk should be about this new project that at the time received all his focus, preventing him from advancing with other projects. In the lecture entitled "Mathematical Typography," Knuth presented his first analysis of the problems of recent printing technology in the domain of mathematical publications to a large group of mathematicians and scientists. Studying and comparing many different examples from the "Transactions of the American Mathematical Society," a publication that

began in 1900 and has more than 230 volumes to date, Knuth found that they were printed in at least 12 different styles. Of these the quality appeared to have generally declined to an absolute low in recent years, at which time he decided there was no point in continuing to write for these publications anymore.

Knuth then proposed solutions involving a row of computer-assisted methods of composition and layout that form the core of TeX, as well as basic principles for mathematical type design. In order to explain that territory, he first focused on past proposals of typography based on mathematical and geometric constructions, such as the mostly geometry-based works by Felice Feliciano, Luca Pacioli, Francesco Torniello and Giovani Battista Palatino in Italy, as well as Geofroy Tory and later the commission of a group of artists and typographers to create a royal alphabet for Louis XIV in France, before dwelling on his proposal for a remedy that finally included some mathematical formulas to describe the characteristics of the curves he was looking for. He ended the talk by presenting a row of tests made with a rough, early version of Metafont, all playful and experimental in nature.

It is interesting to note that the term Mathematical Typography for him really goes both ways in a symbiotic, symmetrical way: there are the new typographic tools, to be created to help mathematical formulas to be correctly and appropriately typeset, and the mathematical formulas needed to solve the typographic problems that the tools required to be designed stood in a mutual relation.

To his surprise, the somewhat unconventional lecture seemed to strike a cord with many of the attendees, and soon after he received various proposals for mathematical solutions to the outlined problems: the perfect mathematical formulation of curves that offer all the flexibility and freedom to describe and modulate the strokes of fonts in an abstract and flexible way.

In his speech he formulated the basic idea of imaginary pen strokes that follow lists of co-ordinates, all parametrised, to describe the glyphs of the typeface. He assumed that such a system, formulated as a specific programming language, would offer all the required flexibility. According to Knuth there was an interesting and very pragmatic reason that led to this pen-based approach over the outline based strategy that Adobe Systems and others have chosen later: in his early experiments, Knuth tried to scan existing mechanical font glyphs in order to

compose their outlines digitally. But since the available equipment at the time was a video camera that distorted the image and was very sensitive to light changes, the results were far from satisfactory and Knuth decided that it would make more sense to produce a system that would allow the formulation of the logic of the glyphs of a font in the way that they were initially drawn and produced by their designers, rather than simply describing their final appearance for production by a printing device. The solution therefore had to be based on calligraphic principles of pen strokes with different nibs and the possibility to extend the resulting geometry with additions, such as serifs. If a font was formulated in its own inherent logic, he concluded, it would be much easier to adapt it for printing in various sizes, changing its contrast, glyph width, x-height, etc. These requirements were directly inspired by the observation that metal letters were produced for specific sizes with different features depending on the size. By making these variations automatic the highest quality possible would be achieved at any size while at the same time respecting typographic tradition.

At the beginning of this digressive side-project he thought would only take six months to complete, Knuth had little knowledge of typography and type design beyond the appreciation of his printed works and a fascination for the trade. But soon it became apparent that the project was going to be far more complex. The process of finalizing the first versions of both Metafont and TeX in the end took four years until the reprint of volume 2 could finally be produced in 1981.

After seeing the results, Knuth was highly disappointed. The volume was printed with a newer facility at a higher resolution than the one he had available for doing his tests, which brought out details he had not been able to see before. While the quality was still better than what the optical system would have produced at that time, they did not match the quality of the mechanically printed earlier volumes, and he was therefore not yet satisfied with the results. Instead of giving up, he went on to improve the situation, and the project grew to what in retrospect made him “put the rest of his life on hold” for eight to 10 years. This strongly affected the work on the missing four of the seven planned volumes of the monograph.

Immediately after the reprint of volume 2, the phase of refinement started, for which Knuth sought support from professional typographers, including Hermann Zapf, whom he first met in 1980, as well as Charles Bigelow, Matthew Carter, Kris

Holmes and Richard Southall. Based on the input of these professionals, Knuth went on redesigning and improving large parts of both Metafont and his Computer Modern typefaces. During this time, Knuth and Zapf were also commissioned by the American Mathematical Society (AMS) to work on a new typeface for mathematics, called AMS Euler. They were joined in their efforts at refinement by the PhD student John D. Hobby, who worked on improving the curve algorithms based on many observations made with the first version. In the years that followed, TeX and Metafont started to be adapted by larger groups of users mostly in the academic world. Workshops were held, and the adaption of the system beyond the scope of roman typefaces was tested, which led to further improvements. It was Knuth's aim to provide a system that could be finalised once and for all and would never have to change again once the work was finished, a system that 30 years later still would produce the same results in print, with varying quality depending on the equipment used. In 1984, Knuth finally published the revised system Metafont84 and started to be satisfied with the results. In 1987, the AMS started to encourage authors to submit their papers for the "Transactions of the American Mathematical Society" in TeX form. This was the very journal that Knuth had analyzed in his speech almost 10 years earlier, the journal for which he threatened to stop publishing articles if the quality was going to degrade beyond a point of acceptable quality.

Now, at the age of 71, Knuth is still in the process of finishing Volume 4 of The Art of Computer Programming, a work highly anticipated for decades. Volume 5 is currently planed to be released in 2015. He may never be able to finish volumes 6 and 7.

Knuth likes to think that of the 10 years spent on TeX and Metafont, 6 - 7 would be regained by being far more efficient with publishing, and maybe more importantly by again enjoying doing so. There are other reasons for the delay of his Volume 4 than simply this digression. But it is hard to completely rule it out as one of the factors for the delay of his highly respected work on computer science. It is an interesting thought that his obsession with the beauty of printed matter might play a role in the fact that he may never be able to publish the full extent of his knowledge.

As a final observation, maybe the stated technical reason of the lack of suitable equipment was partly an excuse for Knuth to indulge in this far more interesting and rewarding task than the simple description of clearly defined outlines, a project that was also closer to his heart as a passionate computer scientist: the use of computer technology and mathematics to formulate a flexible language suitable to describe letter-forms in an abstract, flexible way. It is this decision that makes Metafont such an interesting proposal that even today, 30 years later, it is as relevant as then, when it was one of the first projects for digitized type. Many of the more recent experiments with scripted typefaces, such as Calligrapher by François Rappo and myself Type Generator by Remo Caminada and Ludovic Varone or Kalliculator by Frederik Berlaen share quite a lot of its mindset and could learn a lot from Knuth's endeavors, were they more accessible to people from the field of digital typography.

While most of the questions raised about the nature of digital typefaces as software remain open, ideally they will contribute to a clearer understanding of the different possibilities and ways forward. It seems that we are just at the beginning of a general exploration into the field of self made tools to produce new results, very much along the lines of Knuth's thinking. He was willing to sacrifice a respectable amount of his scientific career to this cause. In the following interviews, such questions of licensing, authorship, design, creativity and the role of software and tools are further expanded upon.

Additional reading

Robin Kinross, Modern typography, Hyphen Press, London, 2004, p. 158-182, Chapter 13 "Modernity after modernism", sub-chapter "Letters as bits, The moment of PostScript, Font wars, Legibility wars, reform and research",

Fred Smeijers, Type Now, Hyphen Press, London, 2003. p. 60-66, Glossary

Fred Smeijers, Counterpunch, Hyphen Press, London, 1996

Robert Southall, Printer's Type in the twentieth century, British Library, London, 2005

Donald E. Knuth, Digital Typography, Center for the Study of Language and Information, Stanford, California, 1999

Donald E. Knuth, Computers & Typesetting Volume E, Computer Modern Typefaces, Addison-Wesley, Reading, Massachusetts, 1986