UBIQUITOUS TYPOGRAPHY

MARCUS LEIS ALLION

Kingston University
United Kingdom

m.leis.allion@kingston.ac.uk

Keywords: Typography, Graphic Design, Code, Programming

This paper considers the impact that software is having upon typographic principles and practice, and how the conventions of typographic communication are being transformed. Inspired by the possibilities of programming many typographers have begun to work more closely with code to generate designs, facilitate production, or organise layouts. However, the very functionality of code often serves to veil its aesthetic attributes.

xCoAx 2014

2014.xCoAx.ORG PORTO, PORTUGAL COMPUTATION, COMUNICATION, AESTHETICS AND X

An integrating philosophy… does not let itself be seduced by the attraction of the 'great problems', but instead initially finds its themes in the trivial, in everyday life, in the so-called unimportant, in those things that otherwise are not worth speaking about, in petty details. Whoever wants to can, in such a perspective, already recognise the kynical impulse for which the 'low-brow themes' are not too low.
Peter Sloterdijk

1. INTRODUCTION

Gerrit Noordzij, Professor of typeface design at the Royal Academy of Fine Arts in The Hague, Netherlands, defines typography as 'writing with prefabricated letters' (Biľak 2007). This statement neatly encapsulates the attributes that make typography distinct from other visual language systems – such as handwriting, calligraphy or graffiti. The letterforms of non-typographic writing emerge from a process, and as a result makes every letter unique. Typography systematises that process by creating a reserve of glyphs that can then be called upon when required. Each glyph therefore needs to be designed before one can arrange the typography.[1] Such details would likely be of interest to a only a few, until we note how widespread the use of typographic interfaces has become. For the linear flow common to many writing tools has been replaced with the discrete keys of keyboards. Typography transforms the continuum of inscription into an ambidextrous composition of parts.

It is strange then that such a widespread and widely used system escapes the gaze of critics outside of the discipline.[2] To some extent this is understandable. The works that typography supports – and books in particular – are designed to be read. Readable typography *functions* well when its visual qualities recede, enabling one to concentrate on the text. In such a context, poorly designed typography is understood to that which calls attention to itself and so interrupts our attention.[3] Even modernist typographers have prioritised types functionality, arguing that '*how it looks*' should be subordinate to '*what it must do*' (Warde 1930). As such book typography (that which is designed for extended reading) and information design (that which is designed for navigation) tend to define how all typography is understood and categorised. As

1 The term typeface refers to the design while each particular material-technical manifestation is called a font. One could think of the distinction between typeface and font as the difference between a song and an MP3 file, respectively.

2 Marshall McCluhan's *Gutenberg Galaxy* and Régis Debray's *Socialism: A Life-Cycle* are notable exceptions.

3 What Martin Heidegger refers to as *un-readiness-to-hand*.

such, readability and legibility largely define how a typeface and its setting are assessed.

However, while such ideals are not without merit, they fail to consider a number of important factors, such as: the reader and/or writer; the content; the materials, tools, techniques and knowledges available; as well as the historical, social, and economic context in which it will be read. Many of these factors are absent in typographic evaluations. For example, the terms readability and legibility highlight a concern for a (single) reader, but within typography there is no equivalent measure for the writer. Historically, typography was the point in a system of publication that transformed the written or typed manuscript into a printed page. As such, the writer rarely worked with typography and so has elicited little attention from type designers. The rise of programming and read/write interfaces has transformed that mode of production considerably.

Similarly, the reader is assumed to be of a particular kind, to have certain capacities and abilities. For in order to function typography has to work with certain constraints (material, ergonomic, economic etc.). In that sense typography is a compromised system that works for many but not for all. As such typography participates in defining and constructing outsiders. For example, the manner in which type is commonly set will be a challenge for those with impaired vision and so may need to wear glasses. The point being that, any disadvantage is located with the individuals body and not with the manner in which the communication is made available. Indeed, one could suggest that once users are able to change communicative settings to fit their needs and preferences certain social disadvantages are nullified. Space does not permit me to extend this argument further, but hopefully one can see that typefaces and typographic settings are both informed by and have an impact upon the social sphere.

This paper, then, foregrounds the impact that computational technology has had on how typographic texts are received, understood, and communicated. From the fixed-width typefaces favoured by programmers, to the default settings of applications, all suggest that a functionalist aesthetic is becoming increasingly prevalent.

One clear example of such a trend is the electronic displays commonly found at train stations and terminals.

While the typefaces used on these systems vary according to the manufacturer, the differences between them are often minimal and often the result of technological specificities. Typically they are constructed from a modular design with little variation in character shape or letter spacing, as shown in figure 1.1.

Fig. 1.1 A typical dot matrix design

Such an approach often produces strange glyphs. In figure 1.1 for example the descender of the lowercase ‘g’ has been pushed up to sit on the baseline. The difference between the uppercase A and R is minimal which may lead to confusion. The figures too could be confused with uppercase designs as they share the same width. Most problematic is the spacing which is tight and erratic. For example the lack of internal space with the letters ‘ili’ means they visually bunch together.

Fig. 1.2 A typographically informed dot matrix design

I have produced an alternative design, (fig 1.2) one that works with the same conditions but is typographically informed. The resultant forms are not only more distinctive but enable the spacing to be more consistent.

In figure 1.3 I degraded these two designs through blurring. While the common design (top) begins to congeal into one mass the typographically informed design remains distinct enough to be legible.

Fig. 1.3 The typographically informed design (bottom) remains legible

Moreover, as the fonts are digital there is the possibility for introducing variability into the system. That is, the typography could adapt to various contexts, conditions, or user requirements.

2. COMPUTATIONAL MATERIAL

Over the past thirty years there has been an increasing development in the field of typography due to the widespread adoption of digital technologies. Such technologies have enabled typographers to question some of the material and temporal conditions that have defined types production, function and appearance. For example, if a different size of a particular typeface was needed it required producing a completely new font. Digital type is not beholden to that same materiality. A digital typeface can easily be scaled to the required size.[4]

The new tools have transformed how type can be produced and so has facilitated the production of new glyph shapes, type styles, formats, as well as production processes. With these new tools designers have begun to explore the possibilities of typography at the macro level of layout as well as the micro level of type design. Indeed, since the introduction of easy to use software the number of typefaces available today has exploded exponentially. The majority of these designs follow the structuring that moveable type required despite the new contexts in which they operate and the affordances such environs enable.

However, replicating historical models presented challenges that required new software to be developed, a deeper understanding of the hardware, as well as a transformation of the models of production and distribution. For example, Beowolf was a groundbreaking typeface created in 1989 by Just van Rossum and Erik van Blokland of LettError. The design used a program that randomly transformed the position of numerous vector points that made up the outline of each glyph. The program ran each time a character was keyed, adjusting the corresponding characters outline. This not only softened the edges of the typeface but also made each glyph unique.

In hot-metal typography it is the font that is fixed while the process of printing produces minute but discernible variability. Digital type appears to shift this relationship, as a computational font is a series of co-ordinates and conditions that define the types appearance in relation to the place it will be seen. That is, printed type is effected by the process of visualisation (pressure, ink and paper etc.) But like platonic forms, digital typography has seemingly escaped the challenges of materiality. In part this is the result of how computing is understood as an

Fig. 2 FF BeoSans & Beowolf designed by Erik van Blokland and Just van Rossum.

4 Scaling type, however, necessitates a reconsideration of the spatial settings and, when available, selecting an optically appropriate size of font.

abstract process. The difference between the same file on a different system or the same file on different screens remains difficult to recognise. We have been told that the digital file can be copied exactly and so fail to discern the impact from a variety of factors, including: the operating system, applications, interfaces, screens, printers, etc. But computational media and the various configurations of software they run are unique material constructs. As such the differences between devices or even a single device remain largely unnoticed.

The typeface Beowolf takes its reference from the irregular patterns of the print process but makes the algorithm that defines the font the site where difference occurs. The font is now the variable while the process and substrate are assumed to be fixed. As such Just van Rossum and Erik van Blokland produced a radical new way of approaching type design. Inspired by such work and the possibilities of code in general many typographers have begun to work more closely with programming languages themselves.

Indeed, typefaces, such as Underware's Liza Pro (2009), have been developed that explore type beyond the level of letter and introduce design at the level of word, employing algorithms to select glyphs according to a carefully worked out character sequence. Such an approach means the design can replicate some of the conditional variants of hand-lettering, something of a challenge for many script typefaces.

Fig. 3 Liza Pro designed by Bas Jacobs, Akiem Helmling and Sami Kortemäki

Paraphrasing the artist Eric Gill,[5] type designer Erik van Blokland has neatly captured this shift in mediums: 'Letters are programs, not results of programs' (Fraterdeus 1997, 130). Indeed, beyond the specialism of type design

5 'Letters are things, not pictures of things' Eric Gill, Autobiography

this is indicative of a wider trend, in which the *sofwarisation* of typography is transforming our visual language more generally.

Fig. 4 Adobes Source Code Pro

```
<!DOCTYPE html>
<html id="home-layout">
  <head>
    <meta http-equiv="content-type" conte
    <title>Source Code Pro</title>
      <!-- made with <3 and AFDKO -->
    <meta name="keywords" content="sans,
      monospace, open source, coding, for
    <link rel="stylesheet" type="text/css
  </head>
  <body>
    <div id="main">
```

3. NEW CONDITIONS, NEW SYSTEMS

All these developments then are linked to the discrete processing capabilities of computational devices, which rely on logical operations, binary processing that are accessed through the symbolic representation known as source code. As philosopher of media and technology, David M. Berry writes, 'code has a dual existence, as delegated code residing in a human-readable frozen state that computer programmers refer to as 'source code' and as prescriptive code that performs operations and processes' (Berry 2008).

The unusual typographic appearance of code is necessary in order for it to function as prescriptive code. The various alphanumeric symbols are arranged according to the requirements of programming languages such as compilers, interpreters, and scripting languages.

Fig. 5 xCoAx website source code

```
▾ <head>
    <meta charset="utf-8">
    <title>xCoAx 2014 - A Conference on Computation, Communication, Aesthetics and X</title
    <!--[if lt IE 9]> <script src="http://html5shim.googlecode.com/svn/trunk/html5.js"></s
    <!--[if gte IE 7]><!-->
    <script type="text/javascript" async src="http://www.google-analytics.com/ga.js"></scr
    <script type="text/javascript" src="//use.typekit.net/jea3aqn.js"></script>
    <style type="text/css">.tk-freight-sans-pro{font-family:"freight-sans-pro",sans-serif;]
    <link rel="stylesheet" href="http://use.typekit.net/c/868f51/freight-sans-pro:i4:i6:n4:
    3bb2a6e53c9684ffdc9a9bf4195b2a62ac2abccc2253da0ff4783e8b08b75860d42a4b5ea156cecf39b318
    15efde7e82bb4a328aae98daeadfa97986a34b53448308d1370e33425b2becda2ea93ec6361406e476b6f0
    <script type="text/javascript">try{Typekit.load();}catch(e){}</script>
    <link rel="stylesheet" type="text/css" href="styles.css">
    <!--<![endif]-->
    <!--[if lt IE 9]> <link rel="stylesheet" media="all" href="ie.css"/> <![endif]-->
    <!--[if IE 6]> <link rel="stylesheet" media="all" href="ie.css"/> <![endif]-->
```

Working with ‘source code’, used to be the exclusive domain of computer scientists and programmers and was therefore rarely encountered by others outside those interests. The emergence, exploration, and almost total adoption of computational processes has meant that many more people have access to the visual language of computational media. Cultural products have also popularised certain notions of what code is.

Aspects that typographers have long considered important for making a text readable, such as: typeface; styles; glyph range; line length; tracking; kerning; case setting; hyphenation; and leading are now determined by the technical capacities of the platform as well as the discursive practices of programmers.

Such decisions effect how, and subsequently, what people can communicate. For example, an italic style of typeface can be used to add emphasis but is currently unavailable on any of the popular social media platforms. As a result users have developed new methods to convey meaning, such as using an asterisk on either side of a word or section of a sentence (see fig 6). However, while this method makes it clear that there is an emphasis, the particular inflection is hard to determine.

Fig. 6 Twitter users using asterisks and uppercase setting to emphasise text

Fig. 7.1 Typographic emphasise using italic and bold

Where the italic in Fig 7.1 suggests a reminder or request, the bold suggests an order that needs to be heeded. So while the asterisked variant makes it clear that there is an emphasis, the particularities of that inflection are hard to determine. Further complications arise with the use of asterisks due to their resemblance to speech marks, as in figure 7.2.

Do not be late
Do not be late

Fig. 7.2 Asterisks are visually similar to speech marks.

All communicative systems require certain limitations in order to work. Indeed, certain typographic systems have had to negotiate with considerations similar to the ones outline above. For example, historically Blackletter typefaces did not come with either bold or italic styles, and so emphasis was conveyed through wide letterspacing.

Do not be late
“Do not” be late

However, generally speaking, typographers will work to minimise any visual disturbances, particularly when setting a text intended for extended reading. The concern of a typographer in that instance is to convey the meaning of the text in a manner that does not call attention to itself. If the typography is poorly arranged it can call attention to itself and so disrupts the reader and their attention. In that sense it is not dissimilar to the goals of the interface developers – to make it so intuitive, that it erases itself for the user. As Bolter and Grusin write, "In this sense, a transparent interface would be one that erases itself, so that the user is no longer aware of confronting a medium, but instead stands in an immediate relationship to the contents of that medium" (Bolter and Grusin, 2000: 24). Well considered settings enable one to concentrate on what is being communicated and not on how it is being communicated. So, while an unmediated experience is an impossible goal, it guides the work of typographers.

The introduction and widespread adoption of computational devices then has clearly had a great impact on how typography is designed and produced. Historically designers worked directly with materials (metal, wood, paper, inks etc.) to create a font. Today, the material of designers – the digital hardware – is mediated through layers of software. Typefaces are sculpted in programs, and written in scripts. As such they require a textual input to enact certain actions–whether simply naming a file or writing a complex piece of code. As such, typography has itself become a tool in the production of typography. However, the functionality of interface typography and the historical and discursive practices that inform it mean there has been little reflection into its aesthetics.

One notable exception is *Pragmata Pro* by Fabrizio Schiavi (see fig 8). Fabrizio Schiavi understands the concerns of coders but manages to bring a nuanced and detailed understanding of typography to this design. For example, the face has no interline spacing making it more compact; characters shapes are designed to be distinct, making it easy to determine 'O' from '0' and the capital 'I' the lowercase 'l' and the number '1'; and has been extensively hinted – a technique that achieves a smooth screen appearance at a variety of sizes across a range of operating systems. As such it represents a good example of how an understanding of technology, contex-

Fig. 8 Pragmata Pro by Fabrizio Schiavi

tual requirements and typographic understanding can be combined to produce a design that does not just function but does so more effectively while enhancing the experience of the user.

4. THE NEW TYPOGRAPHY

Softwarised typography is type that emerges from the discourse of programmers and the cultures that subsequently engage with it. For example, Twitters use of the @ sign to define users and the octothorp (#) to create hashtags that connect common posts can be found in political protests and fashion clothing. The aesthetic of digital communication is commonplace. It is worth noting then that typography in the computational era has become less concerned with certain passive aspects of reading and increasingly organised around certain social or active pursuits. Like the photocopier and other cheap print techniques before it, typography has been passed into the hands of the public. As such the functionalist typography has become an exciting aesthetic in its own right, employed in important and significant manner, whether that be to mark and transmit a political concern or to convey your mood to the world wide web.

Fig. 9 Functionalist typography of Twitter as cultural aesthetic. *Limmy Show* television program written and directed by Brian Limond for BBC Television, UK January 2010.

These graphematic symbols and organisational structuring associated with computational communication do not just perform certain technological functions then. Nor are they simply a visual shorthand for computation. Instead they have been co-opted and disseminated to become social and cultural signifiers in their own right.

5. CONCLUSION

Alphanumeric symbols are arranged according to the requirements of programming languages–such as compilers, interpreters, scripting languages–as well as the conventions of programmers. Thus many aspects that typographers have long considered important are being redefined by programmers and the technical specificity of hardware and software. This will become increasingly significant as more people learn to program and engage with computational media. As type designer Zuzana Ličko stated, 'we read best what we read most' (1990). Functionalist typography will continue to evolve and may become the dominant manner for engaging with text. As such computational media and the visual interfaces required to work with it introduce a whole raft of new typographic approaches and possibilities that have yet to be properly investigated or understood. This paper is a small step in that direction, one that I will be expanding upon and developing. As such, thoughts and insights into how to progress will be gratefully received.

REFERENCES

Berry, David M. *A Contribution Towards A Grammar of Code.* The Fibre Culture Journal, Issue 13. 2008.

Biľak, Peter. *What is Typography?* Available: https://www.typotheque.com/articles/what_is_typography. Accessed 4.2.2014.

Bolter, Jay David, and Richard Grusin. *Remediation: Understanding New Media.* Cambridge, MA: The MIT Press, 1999. 2002.

Debray Régis. *Socialism: A Life-Cycle.* New Left Review 46, July-August 2007. http://newleftreview.org/II/46/regis-debray-socialism-a-life-cycle Accessed 4.8.2007

Fraterdeus, Peter. "Rumours of The Death of Typography Have Been Greatly Exaggerated" in *Design Culture: An Anthology of Writing from the AIGA Journal of Graphic Design* edited by Heller, Steven, and Marie Finamore. New York: Allworth Press American Institute of Graphic Arts, 1997.

Gill, E. *Autobiography: Quod Ore Sumpsimus.* S.l: Biblo-Moser, 1969.

Heidegger, Martin. *Being and Time.* Malden, MA Oxford: Blackwell, 1978.

Ličko, Zuzana. Interview by Rudy VanderLans. Emigre nº. 15.1990.

McLuhan, Marshall. *The Gutenberg Galaxy : The Making of Typographic Man.* Toronto: University of Toronto Press, 1962.

Sloterdijk, Peter. *Critique of Cynical Reason.* Minneapolis: University of Minnesota Press, 1987.

Warde, Beatrice. *The Crystal Goblet, or Printing Should Be Invisible.* 1930. http://gmunch.home.pipeline.com/typo-L/misc/ward.htm. Accessed 4.2.2014.