Writing with Complex Type

Jason E. Lewis
Obx Labs / Concordia University
1455 de Maisonneuve Blvd. W.
Montréal QC H3J 2P7
+1 514 848.2424
jason.lewis@concordia.ca

Bruno Nadeau
Obx Labs / Concordia University
1455 de Maisonneuve Blvd. W.
Montréal QC H3J 2P7
+1 514 848.2424
bruno@obxlabs.net

ABSTRACT
Writing practices that integrate dynamic and interactive strategies into the making and reading of digital texts are proliferating as more of our reading experiences are mediated through the screen. In this paper we argue that rarely do current approaches to creating digital texts operate at the basic textual level, that of the letterform itself. We argue further that such neglect is partially the result of the fact that current font technology is based on print paradigms that make it difficult to work programmatically at the level of individual letters. We then discuss work that has been made with software produced in our lab that suggests the creative possibilities in being able to easily specify behaviours at such a level. We conclude by proposing that writers, typographers and programmers start thinking beyond Postscript-like formats such as OpenType or TrueType to collaboratively develop a new ComplexType format (or formats) that is designed for the twenty-first century as opposed to a simulation of the fifteenth.

Categories and Subject Descriptors
I.3.6 [**Computer Graphics**]: Methodology and Techniques – *Standards.* I.7.m: [**Document and Text Processing**]: Miscellaneous. D.3.0 [**Programming Languages**]: General – *Standards.*

General Terms
Algorithms, Design, Experimentation, Standardization, Languages.

Keywords
typography, electronic literature, computer graphics, textuality, digital type, ComplexType, font format.

1. INTRODUCTION
In this paper, we propose the development of ComplexType, a new font format designed and programmed specifically for digital environments. The necessity for this proposal is grounded in a review of recent history of experimentation in type format. We anticipate that implementing ComplexType will open up new avenues of experimentation for the writer of digital texts.

As people spend more time reading from screens, authors are spending more effort experimenting with methods for writing in the digital environment. Electronic literature, new media poetry and writing for programmable media are three of the many terms currently in use to describe such texts [31]. The ongoing proliferation of labels used to describe acts of creative writing that are in some way essentially digital points to the newness of the field and uncertainty about the relationship of digital texts to their print predecessors.

What all such efforts share is an interest in understanding how the qualities unique to the digital environment can be discovered, developed and employed to create engaging, perhaps even innovative, literature. Thus we have seen writing that, among other strategies, uses hyperlinks to create non-linear reading experiences [18], that combines author-composed texts with texts generated by algorithm [34], that integrates various other media (image, video, audio) to present a multimedia text [1], that incorporates textual movement and transformation in response to the reader's actions [5], and that explores the third dimension to create architectural reading spaces [35].

What very few of these explorations do, however, is experiment with the letterform itself. We argue that this is at least partially a consequence of the material available with which digital writers can work. More precisely, all of us writing in digital media are working within a paradigm for representing letters that was developed for print. The mismatch between tools and currently available techniques makes it difficult to use letterforms in ways that, ultimately, are not that far removed from their print use. A promising way too overcome this difficulty and more fully exploit the rich opportunities that working within a digital environment can offer to authors is to reconceptualize text rendering as a computational process that assumes the screen as its primary display.

2. MOVING BEYOND PRINT
2.1 Print Bias
The current font formats used by most personal computing systems are TrueType, OpenType and Apple Advanced Typography (AAT), all three of which were developed in the nineties from the PostScript template developed in the eighties. PostScript is a powerful technology [40], and particularly in its OpenType permutation capable of a very limited amount of real-time programmatic behaviours. Unfortunately these formats were developed within a print paradigm, which prioritized features based on their usefulness in transferring type from the screen to paper with high fidelity.

Hinting and kerning are examples of how these formats continue to bias towards print surfaces, and it is worth a brief foray into typographic minutiae to illustrate that bias. Both processes were developed to address the gap between what an algorithm can understand about what is aesthetically pleasing to a reader and

what humans will accept [38]. Hinting provides the rendering engine with the information to fine-tune the final output to obtain high-quality text independent of the resolution of the display. It assumes the flatness of paper, and applies to perfection the rules governing a certain view of typography, which Hermann Zapf saw, in 1960, as "two-dimensional architecture" [6].

Kerning, which also assumes a flat two-dimensional space, refers to what, to non-typographers, might seem to be a minor issue: proper optical letterspacing. To typographers, creating a font in which all viable letter pairs appear well-spaced is centrally important to whether a text set in the font will not have any unsightly gaps or jambs and, as a whole, will be pleasing to the eye (see Figure 1). Presently—almost fifty years after the development of digital fonts—an extraordinary amount of a type designer's time is spent performing this task manually. Modern formats implement various strategies to assist with this process, and all of these strategies are clearly engineered with a typesetting mentality. To kern a TrueType font, for instance, a table appropriately named 'kern' stores glyph pairs and a kerning value, a positive or negative value indicating the number of units that the second glyph of the pair should be moved by when the pair is typed contiguously. The font creation software employs an algorithm to provide a first-order pass at specifying these tables, and then the type designer will address problem combinations manually.

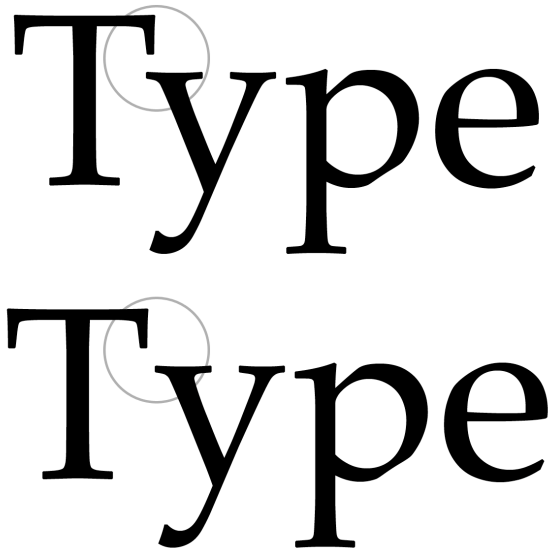

Figure 1. “Type” in Palatino Linotype (top) with kerning and (bottom) without kerning

The kerning table provides a series of flags, one of which indicates the direction, horizontal or vertical, in which rendering engines should adjust the glyph. The binary option is evidence of how the format is optimized for the linearity of typeset text, favoring the grid-like aesthetic of centuries of material printed within the movable type paradigm. AAT and OpenType extend the kerning options to provide more control over the positioning of glyphs, to support cursive (connected) fonts, and to provide precise placement of diacritical marks and of contextual combinations, but continue to presume the flat surface of typeset text. (An even closer look at the details shows how the technology was implemented with Western writing practices in mind, but we leave the politics of typographic technology for future investigations.) [14]

2.2 Wrestling with Postscript

In the late 80's some computer scientists and typographers experimented with PostScript to create dynamic fonts that exploited seldom-used properties of the language to produce differences every time a character was generated. André and Borghi’s interpretation of Knuth’s Punk font [2; 20], Scrabble [4] and Beowolf [39] are the pioneering exemplars of dynamic fonts that harness the programmatic possibilities of digital media by exploiting features of the PostScript Type 3 format (see Figure 2), a version of PostScript Type 1 that Adobe made available to other type foundries. However, the Type 3 format was generally neglected in favor of TrueType, which was considerably better at the task of high-fidelity reproduction but did not incorporate the computationally promising features of the Type 3 format.

Figure 2. Beowolf

The Punk, Scrabble and Beowolf fonts offer an early, tantalizing glimpse of what might have been the future of digital typography. Though they focused on randomness, André and Borghi [3] in particular were already envisioning a more advanced notion of dynamic type aware of and capable of altering context.

Figure 3. Letter 'W' from Move me MM

Another example that pushed standard typographic technology beyond the assumptions of its engineers was Lucas de Groot's typeface Move me MM [29 from 7] published in Fuse 11 [13]. This font exploited the characteristics of Adobe's PostScript-based Multiple Master (MM) format, which allowed typographers to design the extremes of a typeface between which the system could perform interpolations to generate an infinite number of styles. Move me MM exploited this feature, which was ordinarily used to quickly produce a wide range of weights (e.g. light, regular, bold), to create glyphs that could be animated by manipulating the sliders in MM-supporting font design software (see Figure 3). While this exploit did not allow authors to write with the animated typeface, it did demonstrate what might be possible given the right support. Unfortunately, the MM technology suffered the same fate as the Type 3 format, left behind in favor of the resolutely print-oriented TrueType.

The experiments of André , Van Blokland and Van Rossum's, and De Groot's are three of the few explorations made into using extensions of PostScript as an dynamic language for drawing letterforms. Their work hinted at an avenue of development that never fully matured.

In the late 90's Lewis began working with standard PostScript fonts as the basis for SoftTypes [25]. SoftTypes were behaviours

that controlled the dynamic and interactive visualization of PostScript-based letterforms, and could be used in combination to create arbitrarily complex composite behaviours (see Figure 4). SoftTypes were designed to be applied by an author without programming intervention, using menus in the same way that she might change fonts or apply a bold style. Several years later Lee et al. [22], proposed a kinetic typography engine that took a similar approach, establishing a primitive set of behaviours that could be combined together to form complex behaviours. Both these efforts established interesting software architectures for treating such fonts dynamically, but ultimately, even though both relied on standard Open/TrueType fonts, they proved to be dead-ends because their architectures could not easily be imported into standard word processing or visual effects software.

Figure 4. "Pull" deformed with SoftTypes in It's Alive

Before moving past PostScript it is worthwhile to briefly discuss the Neville Brody-directed Fuse CD-ROM journal. Though not focused on the computational infrastructure behind Postscript technology, Fuse encouraged those interested in type design to push, hard, on the limits of the still relatively-new Type 1 format using standard font design tools. Every issue of the journal featured four fonts, each one custom designed for Fuse by a leading typographer or designer. Over eighteen issues the project accumulated a diverse array of experiments that envisioned a much greater range of font use than that which finally settled into the status quo. Tobias Frere-Jones' Reactor [12] in which each successive character slowly fuzzes up earlier characters with visual noise as you continue typing, is a prototypical example of how the Fuse work anticipated some of our concerns with the letterform exceeding its standard printerly constraints by altering its context (see Figure 5).

Figure 5. Reactor

2.3 Post Postscript

At the same time Lewis was developing SoftTypes, Cho was taking a different approach to addressing the issue of typographic interactivity and dynamics [9]. His work followed in the footsteps of researchers at the MIT Media Lab—such as Cooper [10], Maeda et al [28] and Soo [37]—who explored kinetic typography starting in the mid 80's. Cho put aside PostScript and its descendents, abandoning the whole concept of a universal system for digital type. He offered instead multiple custom computational models open for adaptation in static, dynamic and interactive media (see Figure 6).

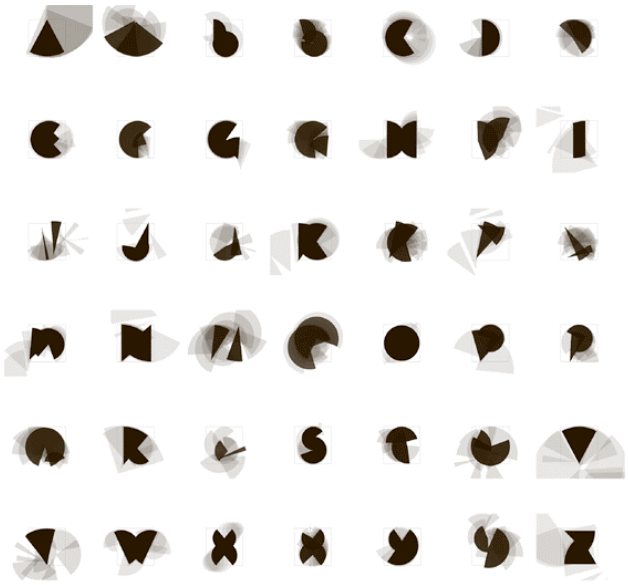

Figure 6. Type me, Type me not

Ten years later, Hillner continued this 'clean room' approach with his experiments in 'virtual typography' [15]. Their low-level, build-it-from-scratch approach towards letterforms added greatly to our understanding of how diverse the notion of a 'font' could be in the digital context while severely highlighting the limitations of standard PostScript-like formats in supporting such diversity.

2.4 Writing Space

Cooper and her team at the Media Lab were also interested in the spatiality of text. Their attempts to extend digital type into the third dimension shed further light on the limitations of standard print-centric typographic technology. Early on, Small et al. [36] pointed out several design challenges in implementing their typographic spaces caused by breaking away from the flat surface, including the difficulty of integrating dynamic type and movable viewpoints into their documents as well as more mundane issues such as text presented in 3D becoming visually distorted. Their conclusions made it clear that the rules of readability developed for flat static typography needed serious revision for dynamic typographic spaces.

Such issues with type in the third dimension faced other researchers and artists, working on different platforms, but always confronted with the difficulties of using digital type in environments that are designed to support specific types of graphical objects (such as textures on flat surfaces). Matthew Kirschenbaum's Lucid Mapping [19] project is one example of text written in a three-dimensional environment. In contrast to Cooper's pioneering work, Kirschenbaum used more readily accessible software and hardware, but he was faced with similar issues when integrating text into the 3D VRML typographic space. In the process of integration, Kirschenbaum had to first type the text in a graphic design software (Adobe Photoshop 4.0), export the composition to an image file and construct the typographic space in VRML by applying those typographic composition as texture on flat-polygons. This fragmented act of

writing, which involves writing text first, redefining it into graphical elements compatible with the writing surface, and integrating them into a typographic space can produce a captivating visual experience and an interesting read, but it shows how a standard typographic technology that evolved to ease the typesetting of text can lead to problems for writing practices that embrace the graphical and dynamic properties of type.

3. FROM COMPLEX SURFACES TO COMPLEX TYPE

3.1 Start with the Screen

In his paper "Writing on Complex Surfaces", John Cayley [8] asks how we could develop an approach to text-based work that is "…faithful to graphics, typography, visuality and textuality all at once..." A central concern of his paper is to grasp the creative and phenomenological consequences of texts that are constructed to have an active, real-time interplay between text, visual aesthetics and structure. He focuses his answer on the increasing richness of reading surfaces and the opportunities their spatial and computational complexity afford for creative exploitation. We are interested in answering the same question by focusing instead on the letterform itself, in particular by moving away from a three-decade old type technology designed for print and towards a born-digital format. We call such a format ComplexType.

ComplexType will allow a common means of employing letterforms that can exploit several unique characteristics of digital media. Examples include spatial letterforms that function as virtual objects with volume and/or which can move in three-dimensional space such as those proposed by Miller [30]; interactive letterforms that can respond to the actions of the reader, other glyphs and other media within the same display environment, such as with Lewis and Weyers' ActiveText [25]; variable letterforms that change, evolve and mutate [9]; and letterforms designed to take advantage of network connectivity like Twin's use of meteorological data to specify appearance [27].

3.2 Lessons from Mr. Softie

The ongoing development of our Mr. Softie application has served as a test-bed for thinking through some of the issues that would be involved in creating ComplexType fonts [24]. Mr. Softie, working with vector representations of characters, provides users with functionality that gives them access to different features of the letterform, allowing them to work at the level of the entire glyph or at the level of constituent contours and vertices (see Figure 7).

Our use of Mr. Softie has shown that making multiple aspects of the letterform easily accessible for transformation can produce innovative results, but it also sheds light on the limitations of typefaces stored as outlines optimized for static rendering such as with standard commercial font formats.

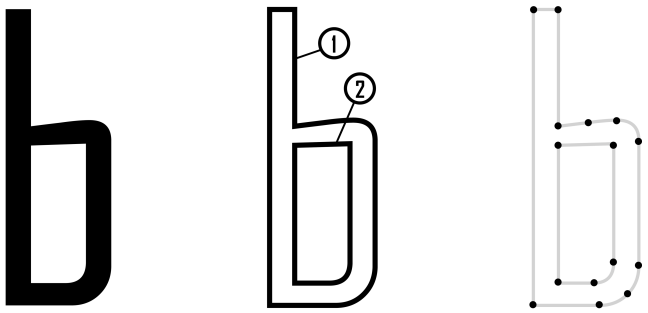

Figure 7. Levels of outline elements - (left) glyph, (center) contours, (right) vertices

During the development of Mr. Softie, it became clear that the optimized glyph definition of PostScript font technology, represented by a series of connected line and curve segments, was a significantly limiting factor when attempting to use standard fonts dynamically and interactively. Although it is possible to convert the outlines into more flexible representations which approximate the desired result, it quickly becomes computationally demanding when interacting with more than several glyphs at once. Furthermore, the algorithms used for transforming the outlines tend to generate results that are not as finely rendered as is expected by people who pay close attention to letterform (see Figure 8). It is possible to algorithmically manage problematic outlines, but it comes at a cost in decreased ease-of-use and performance; utilizing ComplexType fonts designed with dynamic and interactive environments in mind would facilitate the composition process.

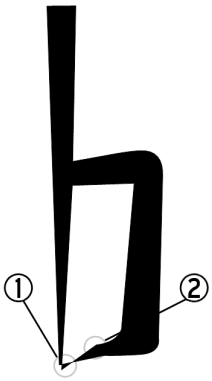

Figure 8. A glyph drawn by Mr. Softie showing problematic outline (1) overlap and (2) discontinuity

3.3 Possible Frameworks

The complexity and diversity of typographic spaces offered by computational media make it unlikely that we can devise a universal font format that would allow writing on any complex surfaces with the same facility and quality. Standard typographic technology like Open/TrueType are universal in that they facilitate the distribution and exact reproduction of type on any devices for which the typographic space is limited to a simulation of print. ComplexType would use a modular approach, taking inspiration from more flexible formats like Unified Font Object (UFO) [23] and Scalable Vector Graphics (SVG) fonts [11], where basic modules necessary for all fonts would be combined with modules that target writing in specific environments.

Typically, a ComplexType font would include a general font information module and a character map, two modules required by all fonts. If the typeface was designed for a 3D environment, it would include a module of glyph definitions represented as 3D geometry (e.g. list of triangles). The font could also include a module indicating the dynamic properties affecting the glyphs or their constituents. Futhermore, a separate interactive module could be included to indicate actions triggered by different input mechanisms. An application that supports ComplexType fonts would check which modules were present and make them available to the user. It is clear that a certain level of standardization is required if ComplexType fonts are to be supported in a range of applications that advance beyond the experimental stage, but this process should take place through collaboration with authors and designers to address the various typographic spaces of interest to these groups.

3.4 Potential ComplexType Fonts

Looking at the history of recent experimentation in this area suggests, even at this early stage, several possible ComplexType font designs. A notable genre of typeface, which was a significant part of the early experiments conducted by Maeda and later Cho, is type with letters constituted of multiple graphical elements or particles that arrange together to form the glyphs of a typeface. The flexibility inherent in the fragmentation of the glyph provides interesting options to an author. Cho's Type me, Type me not (see Figure 6) exploits the adaptability of glyphs made of similar circle sections to create type that smoothly transforms from one letter to another. In the same way that kerning is an activity that targets the flat surface of typeset text, in a ComplexType font it would be possible to define the 'spanning', or the transitioning from one glyph state to another.

Figure 9. Hydra

In addition, typefaces made of particles offer interesting possibilities for interactivity. Nadeau's Hydra typeface [32] combines a common outline of lines and curves with a large number of small dynamic circular particles to form each letter. The particles, which can appear anywhere in the typographic space, are attracted by certain letters and move continuously once trapped in an outline. The particles are always subject to dislodging forces applied by interacting readers who have a certain control over the formation and breaking apart of the text (see Figure 9).

Figure 10. Origin

ComplexType fonts would also include features that provide a fine degree of control over the rendering process. Nadeau's Origin typeface [33] combines several particles that appear differently over the formation of each glyph. Starting as animated spermatozoa, the particles reach the glyph to which they belong and become specks leaving subtle traces around the letter, slowly revealing it in the negative space (see Figure 10). Type design necessitates close attention to the balance between positive and negative space, and standard typographic technology provides type designers with precise tools to typeset text, but experimenting with this combination balanced in a dynamic and interactive environment shows promising results.

Figure 11. Polymorphous

Another approach towards type design, which was central to the experiments of Miller [30] and also present in Cho's work, is type built for 3D virtual environments (see Figure 11). Although it is possible to integrate standard outline fonts into the third dimension, the expensive process of converting the glyphs of these fonts to a format supported by 3D virtual environments adds

considerable friction to writing practices that do not assume the immutability of the letterform. In addition, a glyph definition that specifically targets these environments would lead to new typeface design directions responding to the attributes of 3D space.

Stroke-based fonts are another approach, one that defines the glyphs of a typeface as a series of strokes, simulating a pen or brush applying ink on paper. This technique dates back to Donald Knuth's pioneering mathematical typography research of the mid-70's, which led to the implementation of the METAFONT system [21]. In addition to METAFONT, stroke-based fonts are commonly used to minimize the number of vertices required to represent the complex ideograms of East Asian languages. A stroke-based ComplexType font would extend Knuth's method in order to incorporate properties specific to dynamic and interactive environments. This, in turn, would facilitate the manipulation of the letterform over time and by the reader.

A ComplexType specification would involve a core font specification that would allow the use of ComplexType fonts in various types of applications. In the same way that you now specify a typeface (Helvetica, Times, etc.), a style (bold, italic, etc.) and a point size, a ComplexType font would allow you to specify how its glyphs behaved (movement, lifespan, interaction, etc.) The individual glyphs in such a font would integrate capabilities for recognizing and using temporal change, handling interaction with the user as well as other letterforms and media elements, processing external data sources and communicating across the network. All these characteristics can be used to determine, in real time, how glyphs appear and evolve over time. The integration of dynamic and interactive behaviours into the ComplexType format would provide writers (and typographers and designers) with a high degree of control over reasonably complex behaviours.

4. FROM COMPLEX TYPE TO COMPLEX WRITING

In order for ComplexType to be useful to writers, writers need to be involved in the process of developing the specification. Previous experiments with active glyphs rarely engaged writers centrally, more often being the products of investigations into design and typographic form (Maeda, Cho, LettError) or the computational properties of PostScript (Andre et al). They have not been investigations into creative language use.

In the NextText project we proposed a tight integration of writing, designing and programming in order to more deeply explore the production of innovative digital texts [24]. Among the conclusions we reached three years into the project was that making work which provides a reading experience as rich as its dynamic and interactive experience (and vice-versa) requires that the author be able to directly engage interactivity, etc., as part of a conscious strategy of meaning making *within* the writing process (as opposed to the post-writing process.) Such *complex writing* will benefit enormously if every aspect of the technology through which it is realized has been rethought in terms of the digital.

The Mr. Softie application mentioned above has allowed us to experiment with rudimentary forms of complex writing. In the SoftSketches series [26] we wrote poems specifically with Mr. Softie in mind, and then used the application to manipulate the appearance and further edit the texts. The resulting works take part in the concrete poetry tradition of intensive focus on the presentation of the text. The text of Dependency, which can be summed up in the line "an / electric thread ties and bonds the / agglomerated mind", evolved into a mass of letterforms intertwining in two knotted masses grasping at each other across the canvas (see Figure 12).

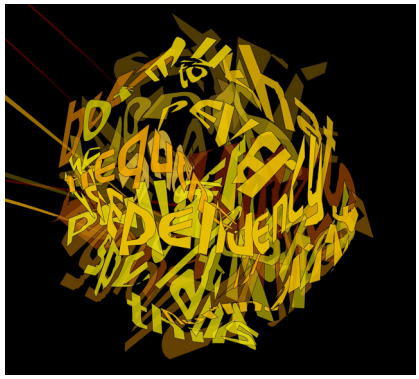

Figure 12. Dependency (detail)

In History, a poem about how the past becomes abstract and flattened the further away we get, we worked the letterforms to the point where they became a collection of abstract geometries layered one on top of another (see Figure 13). In both cases, the ease with which Mr. Softie allowed us to apply complex behaviours promoted an ability to work intensely and deeply with the visual representation to the poems, allowing us to shape them however the text demanded.

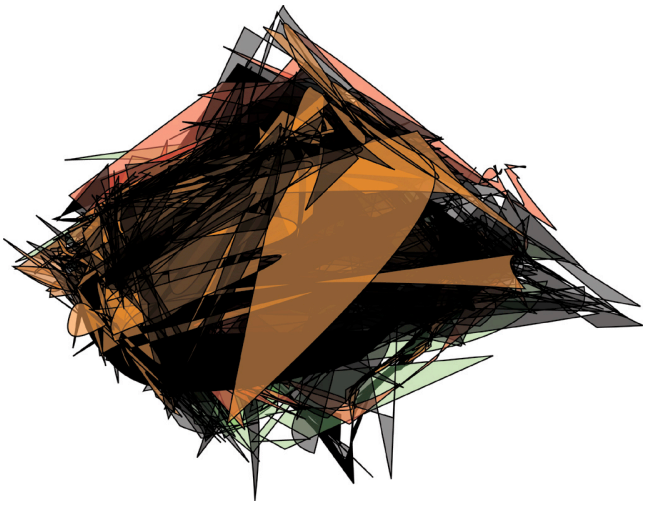

Figure 13. History

In 2008, we invited poet David Jhave Johnston to collaborate on the Mr. Softie development process by creating text works with the application and providing us with his critique of the tool. He produced Softies, a series of media and motion works for the web that employed Mr. Softie to create the texts [16]. One of the series, Stand Under, contains the lines "Develop an understanding / Stand under / Humility understands", and is represented (at first) by a strange, elongated tower of text in which parts of "understand" are crushing the rest of the poem (see Figure 14).

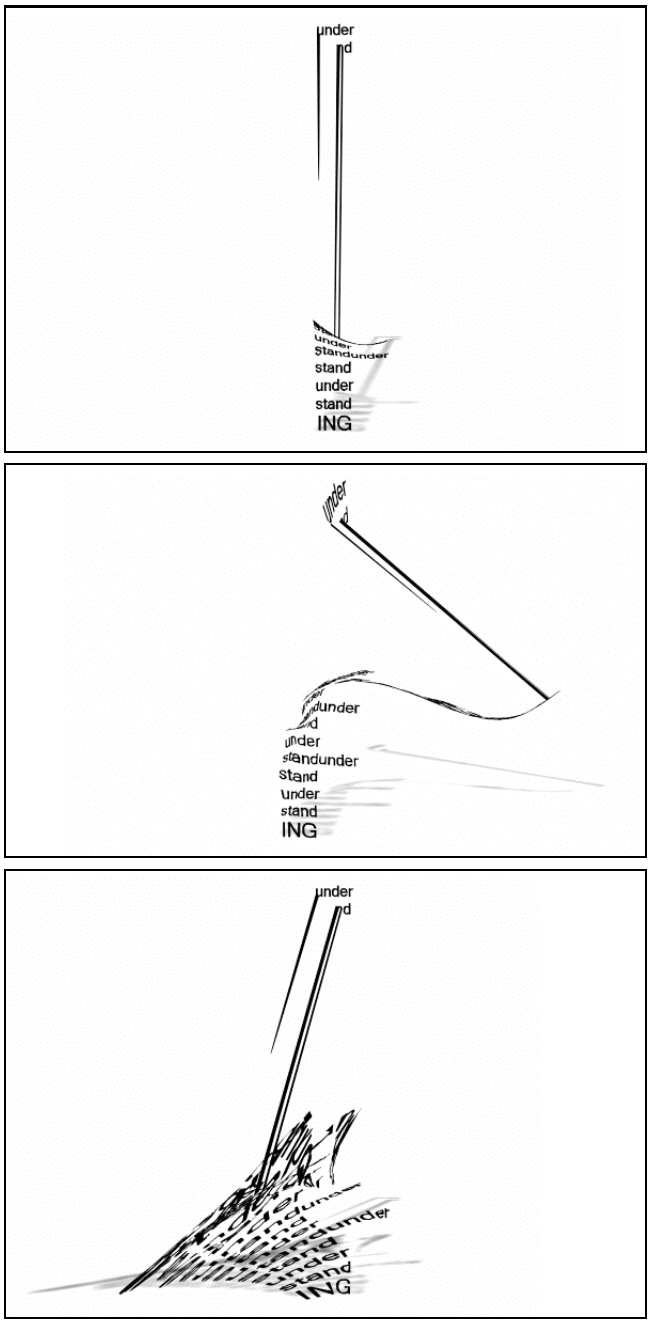

Figure 14a, b, c. Stand Under

As the piece progresses, the remaining text reasserts itself, pushing up against the "understand" and causing the whole text to wobble from side-to-side and distort drastically as it expands to take up a normal amount of space. In another piece, Poet, we see a series of images in which objects—alphabet blocks in the poet's mouth, a stuffed rabbit, scissors, glasses—are manipulated by bodiless hands. Next to the object is an evolving text that slowly forms, disintegrates and reforms (see Figure 15).

In both these works, the text, manipulated algorithmically and by hand, possesses a liveliness that would be difficult--and significantly more time-consuming--to achieve using standard tools for word processing or motion typography.

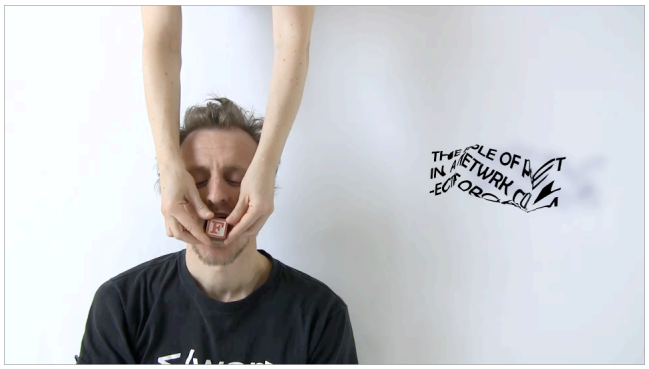

Figure 15. Poet

Johnston said of working with Mr. Softie that, "as a writer I value the fluid, intuitive, sculptural way that text can be manipulated in real-time. As a visual poet, the capacity to apply pressure and see instant folds in the surface of the letterforms represents a fusion of writing with the paradigm of 3D modeling." [17]

The same could be said of the SoftSketch works. What we find encouraging is that these series provide initial indication that supporting ComplexType-like capabilities as part of a complex writing process can produce texts that begin to tap into the full power of the computational environment.

5. CONCLUSION

In this paper we have shown how the print-bias of current technology impedes our ability to exploit the full potential of digital text. We have discussed various early efforts to get beyond the print paradigm, and have used our current research as the basis for articulating several trajectories of further exploration into ComplexType technology that should prove fruitful in producing tools and techniques useful to writers of any kind of digital texts.

We believe that such efforts will help answer some of the more significant questions involved in the evolution of writing in the digital environment: can digital technology foster new, substantive modes of writing in the same way that the printing press provided a technical foundation for the rise of the novel, the essay and the pamphlet? What techniques can be developed for allowing creators to write a text, design its appearance and program its behaviour in an integrated manner? How can these techniques be configured as tools that allow creators to easily draft, sketch and prototype without losing focus on the content of the work? And, finally, how might access to such tools expand the pool of individuals writing innovative digital texts, and consequently extend the role such texts can play in everyday life?

We are generally interested in considering how qualities unique to media presented via computing machinery can be articulated--both conceptually and technically--in support of new techniques for incorporating meaning into the presentation of texts. The more time people spend reading text on one form of screen or another, the less sense it makes to rely on a technology developed in many core aspects to address challenges related to displaying text on the printed page. Correspondingly it also makes sense for the writers of those texts to have available to them technology developed specifically for the screen in order to fully exploit the creative possibilities for displaying text on it.

6. ACKNOWLEDGMENTS

We would like to acknowledge generous support of the Social Sciences and Humanities Research Council, the Fonds de recherche sur la société and et la culture and the Hexagram Institute for Research/Creation in Media Arts and Technology's for this research.

7. REFERENCES

[1] Amerika, M. 1993. Grammatron. Web-based interactive work. www.grammatron.com. Accessed August 10, 2009.

[2] André, J. and V. Ostromoukhov. 1989. « PUNK: de Metafont à PostScript». Cahiers GUTenberg 4: 23–28.

[3] André, J. and B. Borghi. 1989. Dynamic fonts. Raster Imaging and Digital Typography: Proceedings of the International Conference, Ecole Polytechnique Fédérale Lausanne, October 1989, 198. Cambridge University Press.

[4] André, J. 1990. The Scrabble font. The PostScript Journal 3, no. 1: 53–55.

[5] Andrews, J. 2003. Arteroids 2.5. poemsthatgo, no. 14 (Fall). www.poemsthatgo.com/gallery/fall2003/arteroids/article.htm Accessed August 14, 2009.

[6] Blackwell, L. and A. Hyland. 1992. Twentieth-century type. Laurence King.

[7] Bachfischer, G. and T. Robertson. 2005. From Movable Type to Moving Type-Evolution in technological mediated Typography. AUC Academic and Developers Conference.

[8] Cayley, J. 2005. Writing on Complex Surfaces. Dichtung-Digital.

[9] Cho, P. S. 1999. Computational Models for Expressive Dimensional Typography. Massachusetts Institute of Technology.

[10] Cooper, M. 1994. Information landscapes. MIT Technical Note.

[11] Eisenberg, J. D. 2002. SVG essentials. O'Reilly & Associates, Inc. Sebastopol, CA, USA.

[12] Fuse 7 (CD-Rom Magazine). 1993. FontShop International.

[13] Fuse 11 (CD-Rom Magazine). 1994. FontShop International.

[14] Haralambous, Y. and P. S Horne. 2007. Fonts & encodings. O'Reilly Media, Inc.

[15] Hillner, M. 2007. The poetics of transition: the ambiguous characteristics of virtual typography. Royal College of Art.

[16] Johnston, D. 2009. Glia. Web gallery of work. www.glia.ca

[17] Johnston, D. 2009. Personal communication.

[18] Joyce, M. 1997. Afternoon, a story. Eastgate Systems.

[19] Kirschenbaum, M. 1999. Lucid Mapping: Information Landscaping and Three-Dimensional Writing Spaces. Leonardo 32, no. 4: 261–268.

[20] Knuth, D. E. 1988. A punk meta-font, TUGboat, vol. 9 no. 2, August 1988, pp. 152 - 168.

[21] Knuth, D. E. 1999. Digital typography. Center for the Study of Language and Information Stanford, CA, USA.

[22] Lee, J. C., J. Forlizzi and S. E Hudson. 2002. The kinetic typography engine: an extensible system for animating expressive text. Proceedings of the 15th annual ACM symposium on User interface software and technology, 81–90.

[23] Leming, T., E. van Blokland and J. van Rossum. 2003. Unified Font Object (UFO). www.unifiedfontobject.org/

[24] Lewis, J. E. 2007. Writing-Designing-Programming: The NextText Project. Proceedings of Digital Arts and Culture '07.

[25] Lewis, J. E. and A. Weyers. 1999. ActiveText: a method for creating dynamic and interactive texts. In Proceedings of the 12th annual ACM symposium on User interface software and technology, 131-140. ACM New York, NY, USA.

[26] Lewis, J. E. and B. Nadeau. 2006. Soft Sketches. www.brunonadeau.com/softsketches/

[27] Littlejohn, D. ed. 2004. MetroLetters: A Typeface for the Twin Cities. University of Minnesota Press.

[28] Maeda, J., Worldwide Division and D. Co. 1996. Flying Letters. Digitalogue Co. Ltd.

[29] Middendorp, J. 2004. Dutch type. 010 Publishers.

[30] Miller, J. A. 1996. Dimensional typography. Princeton Architectural Press.

[31] Morris, A. 2006. New Media Poetics: As We May Think/How to Write. In New Media Poetics: Contexts, Technotexts and Theories, A. Morris and T. Swiss, eds. MIT Press.

[32] Nadeau, B. 2009. Hydra. www.brunonadeau.com/complextype/hydra/

[33] Nadeau, B. 2009. Origin. www.brunonadeau.com/complextype/origin/

[34] Seaman, B. 1996. Emergent Meaning as Examined and Explored Within a Specific Generative Virtual Environment. Dissertation, Centre for Advanced Inquiry in the Interactive Arts.

[35] Shaw, J. 1989. The Legible City. www.jeffrey-shaw.net/html_main/show_work.php3?record_id=83

[36] Small, D., S. Ishizaki and M. Cooper. 1994. Typographic space. In Conference on Human Factors in Computing Systems, 437–438.

[37] Soo, D. 1997. Implementation of Temporal Typography System. Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science.

[38] Spiekermann, E. and E. M. Ginger. 1993. Stop stealing sheep. Adobe Press.

[39] Van Blokland, E. and J. van Rossum. 1990. Random code—the Beowolf random font. The PostScript Journal 3, no. 1: 8–11.

[40] Warnock, J. and C. Geschke. 1992. PostScript Language Reference Manual. Adobe Systems Inc., Menlo Park, Calif.