Augmenting Literacy: The Role of Expertise in Digital Writing

Derek Van Ittersum

This essay presents a model of reflective use of writing technologies, one that provides a means of more fully exploiting the possibilities of these tools for transforming writing activity. Derived from the work of computer designer Douglas Engelbart, the "bootstrapping" model of reflective use extends current arguments in the field regarding the role of technical expertise in writing. Individually, and as a discipline, we can use Engelbart's example as inspiration for assembling unique systems of tools that extend our sense of what is possible in terms of producing and distributing texts.

I'm currently writing this paragraph in the word processing program Pages, part of the Apple iWork '09 software suite. Also open are DevonThink, a database program holding all of my reading notes from the past several years, and Tinderbox, a difficult-to-encapsulate program that I have used here for brainstorming (creating visual maps of notes and links) and outlining. Many of the relevant notes for this article were copied from DevonThink into Tinderbox and moved about in various maps until the maps cohered into finished-for-now outlines. While I reviewed the reading notes in DevonThink, the program used its fuzzy searching algorithms to suggest notes that it believed were related. As I write in Pages on one monitor, a second monitor displays several Tinderbox windows (with the outline and a map) and the DevonThink window is hidden behind these.

While at one level the writing process described above is ordinary enough (notetaking, synthesizing notes, outlining, drafting), the specific practices and programs are likely unfamiliar to many readers. This is not to say, however, that they are highly specialized or require programmer-level computer expertise. Many writers likely have customized digital workflows in a similar fashion, but they would seem equally obscure because writers rarely share their practical knowledge in any formal or systematic fashion. While computer classes or workshops could be one area where computer writing expertise would be shared, they are almost always dedicated to learning the features of a particular program rather than detailing how users might best exploit an array of software for a wide-ranging activity like writing.

Furthermore, descriptions of the instrumental sides of digital literacy remain few and far between in rhetoric and composition scholarship.[1] By relegating development of technical expertise with the tools of contemporary composition to hallway conversations and accidental revelations, rhetoric and composition scholarship reinforces popular beliefs about the ease and intuitiveness of these tools. These beliefs generally lead to increased frustra-

Composition Studies 39.2 (2011): 61–77

tion for users when tools designed to be intuitive turn out to be anything but easily learned (see, for example, Dilger). As Barbara Mirel has shown, when computer software attempts to be intuitive, what results are simple tools inadequate for complex activities. When ease of use is privileged over other qualities, such as usefulness, it becomes difficult to judge between a uselessly difficult program and one whose difficulty mirrors, and engages with, the complexity of the task. In this way, valuable programs that may require training are dismissed. Even worse, training becomes denigrated as an indication of an inferior user who finds difficulty with "easy" programs.

Since the particulars of software use are not frequently discussed in the field, some may assume that writing software is inconsequential for writing or is straightforward to use, thereby requiring no explicit training or development of skills. Those that come across challenging programs or rewarding digital writing practices are left to wonder whether they alone feel challenged and rewarded or if they are alone in using such software. Many writing teachers and scholars, though, look to computers not for a challenge to existing practices, but, as Stuart Selber has argued, to "simply transfer wholesale to the screen their existing assumptions, goals, and practices" (23).

The history of computing shows that many people, not just writing teachers and scholars, have adopted a similar approach to computers. Many early computer designers focused on the ways computers would automate everyday tasks instead of how these tools might alter such work. For example, J.C.R. Licklider, an early proponent of interactive computers, wrote in 1960 of his attempt to "keep track of what one moderately technical person [i.e. Licklider himself] actually did during the hours he regarded as devoted to work" and found that "[his] 'thinking' time was devoted mainly to activities that were clerical or mechanical" and thus determined that "the operations that fill most of the time allegedly devoted to technical thinking are operations that can be performed more effectively by machines than by men" (133-4). Computers, under this formulation, should do the drudge work of running calculations and plotting graphs, thereby freeing up the human mind to work creatively. This same attitude can be seen behind the contemporary preoccupation with "ease of use" and so-called intuitive interfaces. Developing expertise with a computer program or learning new working practices required by software often seems like tedious or tangential work, and thus flies in the face of a perspective that sees computers as solely tasked with eliminating our drudge work.

Some computer designers, though, have attempted to promote perspectives that counter the dominant focus on eliminating drudge work. One such designer is Douglas Engelbart, known popularly as the inventor of the mouse. Engelbart's research into computing, along with the systems his lab developed in the late 1960s and early 1970s, set the stage for personal computing at a time when computers were mainly used for mathematics. Engelbart argued for the creation of systems that would enhance, not eliminate, everyday work. Making similar observations as Licklider, Engelbart and his

colleagues came to different conclusions. They note that "specialized work . . . while vital to [knowledge workers'] effectivness, probably occupied a small fraction of [their] time and effort" and that the "bulk of [their] time . . . would probably be occupied by more general knowledge work" (Engelbart, Watson, and Norton 10) such as composing written documents or telephone communication. Rather than proposing that general knowledge work could or should be automated by computers, Engelbart proposed computers as a means to "augment" that work through the development of new "information-handling . . . techniques, procedures, and systems" ("Conceptual" 2). Most important for my argument here is Engelbart's focus not simply on designing new computer systems, but also new working techniques. He understood that new tools were not enough—people needed to develop the instrumental expertise necessary to make the most of new technologies and means of knowledge work.

Engelbart's focus on writing as the first activity to be transformed through computing gives his work continued relevance for the field of composition. By focusing on Engelbart's research and technologies, I will show how his efforts to develop writing technologies to transform everyday knowledge work provides a model of computing that has been submerged in rhetoric and composition, and in popular representations of computing more broadly. By countering approaches that seek to make computing skills invisible or inconsequential, Engelbart's work shows the benefits of developing expertise with tools that transform writing activity. My aim in drawing on this history of computing is, ultimately, to find ways to more fully balance our approach to computer writing software and its use and instruction by positioning computer expertise as an integral component of contemporary literate activity.

Technical Expertise

Several scholars within rhetoric and composition have begun the work of reframing expertise and computer literacy to better fit rhetorical practice. One of the central contributions of Selber's *Multiliteracies for a Digital Age* is its reappraisal of functional literacy. Primarily through an in-depth analysis of the "tool metaphor" dominant in conceptions of computer use, Selber argues that the field needs to break out of a tendency to see computers solely as neutral or overdetermining. "As a human extension," he writes,

> the computer is not self-determining in design or operation. The computer, as a tool, depends upon a user, who if skilled enough can use and manipulate its (non-neutral) affordances to help reshape the world in potentially positive ways. (40)

Functional literacy, then, is not an insignificant skill to be ignored in writing classrooms, nor a dangerous orientation blinding writers to the political consequences of computer use. It is an inseparable component of ethical and effective communication.

Selber's argument resonates strongly with others concerned with the disappearance of "know-how" in the realm of tool use. Robert R. Johnson, for instance, seeks to re-emphasize the knowledge users have in an effort to counter current tendencies by designers to view computer users "as idiots who must have the technologies 'dumbed down' to their level, a level that has no knowledge of its own" (13). Bradley Dilger traces the cultural assumptions that tie together ease and technology, noting the dangers for writers in the long-standing belief that technology should be transparent and effortless (118). Jenny Edbauer Rice has recently identified composition's resistance toward instruction in functional literacy as a conflation between teaching rote grammar exercises (mechanics) and rote technical skills (367). In upending this negative view of mechanics, Edbauer Rice privileges rhetorical production, which requires a wide variety of technical skills. All of these scholars work to show that expertise with digital composing tools is an integral rather than separate component of writing.

Researchers in technical communication have already begun to detail the kinds of productive expertise required for complex writing tasks. Clay Spinuzzi has traced the productive work of computer users who customize their workspaces with texts such as sticky notes (1). Through analysis of videos of technical writers at work, Shaun Slattery has outlined the components of "textual coordination," a set of practices that function

> as an attempt to control the environmental conditions in which [writers] do knowledge work. Those writers who are best able to control environmental conditions and maintain operational fluency are best able to stay focused on achieving higher-order goals (357).

In other words, skillful manipulation of software and print documents by these technical writers enabled successful rhetorical performances. As the work of these scholars has shown, technical skill in using writing technologies is inextricable from the forms of rhetorical production our field most values. Yet writers outside these technical fields are faced with "dumbed-down" software that positions them as idiots without technical skill or are discouraged from wasting time learning the "mechanics" of more complex software that is seen as controlling and autonomous.

Some in rhetoric and composition have sought to avoid the challenges posed by the instrumental demands of new software and technologies by returning to print-based techniques. Peter Kratzke, for example, has recently argued that the technique of recopying text between drafts promotes revision "in the truest sense of the word—to re-see expressions and ideas" (9). Because word processing encourages editing existing text on the screen, "the physical concept of a draft has radically changed" (10). Resisting this change by printing out drafts, Kratzke argues, makes "recopying a six-line run-on sentence . . . all but impossible. . . A nonsensical paragraph dissolves before our fingers hit the keys" (17). While I might quibble with the certainty of these claims (recopying is no more of a magic bullet for improving student

writing than any other computer or print-based technique), I certainly can appreciate the value of such a practice and the goal of improving students' ability to revise. What troubles me is the overarching argument that such print-based practices are all that is needed to help students "learn to use the technology better" (9), Kratzke's purported goal. Recopying may teach students to revise better, but I can't see what it teaches them about the use of computers for writing beyond avoiding and distrusting the many computer tools available that may help with revision. Scholars like Kratzke, and writers in the popular media such as Nicholas Carr, promote human empowerment through limiting or even eliminating technology use. It is hard to argue that turning off the automatic email alerts and eliminating one's blog feeds, as Carr advocates, will have some benefits for concentration and mental health (200). However, these strategies of avoiding technology will not teach writers to use advanced text manipulation software, or how to create and mine databases for information and profitable connections, or how to use these techniques to increase their rhetorical effectiveness.

Consider, for example, Steven Johnson's depiction of his writing process in a 2005 essay in the *New York Times*. With the DevonThink database software mentioned above,[2] Johnson used fuzzy searching (e.g., when Johnson employs the word "sewage" as search criteria, he gets results that include similar terms, like "waste"), to draw connections between notes he wrote and stored in the database that he had not previously seen as connected. By trying to reconcile why notes about the biology of calcium waste products were returned in a search for "sewage," Johnson thought of an analogy between how cities use waste productively and how vertebrate animals evolved when cells turned calcium waste into bones. In reflecting on this thought, Johnson suggests that "I'm not at all confident I would have made the initial connection without the help of the software. The idea was a true collaboration, two very different kinds of intelligence playing off each other, one carbon-based, the other silicon" ("Tool for Thought" 27). As Johnson details in a blog post follow-up to his *Times* essay ("Tool for thought"), several particular practices contributed to the successful collaboration between the machine and his thoughts, including storing only interesting bits of information and cutting each bit into 50-500 word chunks (as opposed to importing whole articles and books and storing them as single files). What is valuable about Johnson's description is not the focus on the unique intelligence of Steven Johnson or the particular features of the database software, but instead the affordances of the system that is constituted by the skillful and reflective coordination of these elements. In other words, Johnson clearly represents an empowered user who uses technology to transform his writing.

Johnson's orientation toward software resonates strongly with Douglas Engelbart's project to augment human intelligence with computer technology. As his research articles and hardware designs illustrate, Engelbart saw the value of computers as more than offering faster ways of accomplishing rote tasks (such as calculations). Instead, he argued that computer tools

are useful because they allow people to think in new ways. Furthermore, Engelbart's colleagues put their ideas to the test by designing and then using tools and techniques that would transform their everyday working practices, and it is this process that can provide a model for writing scholars as we seek to integrate and exploit computer technology more fully in our research and teaching.

Bootstrapping Augmentation

The term "augmentation" may conjure up images of human-computer cyborgs with electronic eyes and wires connected to their brains. In fact, Engelbart meant it as a more general term, referring to almost any aspect of cognition. In a book chapter published in 1963, Engelbart explains his view of human activity:

> Individuals who operate effectively in our culture have already been considerably "augmented." Basic human capabilities for sensing stimuli, performing numerous mental operations, and communicating with the outside world are put to work in our society within a system—an H-LAM/T system—the individual [human] augmented by the language, artifacts, and methodology in which he is trained. ("Conceptual" 8)

Engelbart had concluded that our mental and physical tools were no longer adequate to address the problems of the day, and his research program sought ways to enhance people and their tools collectively. Thus, rather than focus on either Language, Artifacts, or Methodology (or an improvement in Training in these areas) alone, Engelbart's augmentation approach posited that improving these elements together as a system better reflects the ways they always affect each other in practice. "The important thing to appreciate here," Engelbart argues, "is that a direct new innovation in one particular capability can have far-reaching effects throughout the rest of your capability hierarchy" ("Conceptual" 7). Focusing solely on an innovation in one area, such as a new writing software application, may prevent one from seeing how use of the software leads to language changes or new composing processes.

Augmentation, as Engelbart defined it, was enacted within his research lab, the Augmentation Research Center (ARC), through the strategy of "bootstrapping." Originally referring to the way a computer "pulls itself up by its bootstraps" when relatively simple software routines programmed into the circuit boards initiate the more complicated ones that run the computer (now simply called "booting"), Engelbart and his ARC colleagues employed this term to describe the process of using the tools they made to enable the creation of even more useful tools.

> Within ARC we do as much work as possible using the range of online capabilities offered. We serve not only as researchers, but also as the subjects for the analysis and evaluation of the augmentation system that we have

> been developing. Consequently, an important aspect of the augmentation work done within ARC is that the techniques being explored are implemented, studied, and evaluated with the advantage of intensive everyday usage. We call this research and development strategy "bootstrapping."
>
> In our experience, complex man-machine systems can evolve only in a pragmatic mode, within real-work environments where there is an appropriate commitment to conscious, controlled, exploratory evolution. (Engelbart, Watson, and Norton 12)

Within ARC, bootstrapping through "conscious, controlled, exploratory evolution" began with the creation of writing tools. Functioning hardware and software components became more efficient tools for writing software and reports. In exploring the new practices afforded by these tools, the ARC researchers discovered new possibilities for activities that were not imaginable before their computer was built. As Engelbart described it during a retrospective interview with Adams and Lowood in 1987:

> You build something that you can use. You start using it and then you improve that and build a different one. That was the projection. I would start with text-processing by the computer that could help me do the text work. It composes reports and memos along the way, and software. (Interview)

It is somewhat surprising that Engelbart saw writing (or at least "text-processing") as a foundational activity that would support further co-evolution between humans and computers. This view, although resonant with that of many writing scholars, put Engelbart and his research partners within ARC at odds with others within the computer research community in the late 1960s. For instance, in 1961 a program was written for the PDP-1 computer at MIT that functioned as a crude text editor, allowing a user to type into the computer (that had no display) and print using a Flexowriter. Its name was "Expensive Typewriter," and was a kind of joke (Waldrop 188).

As conceived by the researchers at ARC, bootstrapping was a lengthy process, but one that could provide tangible results along the way if approached reflectively and strategically. Beginning the bootstrapping process with writing technology was a primary strategy, as Engelbart recounts:

> Just learning how to harness, to integrate all this technology into our way of thinking and working, would take a long, long time. But some paths that you would follow would get to the benefits and effectiveness much sooner than others. The strategy was important to me. These are things that made so much difference. I'd say, "I'm starting to work with documents, with the support of expositional communication and thinking and developing an argument that can be communicated, and with integrating lots of other people's thoughts and considerations." That would be right at the heart of how you learn from experience and integrate it. (Interview)

"Support of expositional communication" took several forms in the computer built by Engelbart and his colleagues around 1967. Because it was a

networked, time-sharing computer system, their system enabled real-time and asynchronous collaboration on text documents (both "expositional communication" and programming code). Several users could be logged into any one computer at a time, and users from different computers could communicate through the networking protocols. Users could also leave messages for each other, either in a direct form like email, or as a note appended to a specific section of a document or piece of code. In these ways, their system supported collaboration on both the programming work necessary to develop more computer software, and also the written work required to receive government funding and share the results of their augmentation research in publications and at conferences, as well as amongst themselves.

Bootstrapping Writing

To examine the strategy of bootstrapping as a means of augmenting writing practices, I now turn to a brief history of Engelbart's development of writing technologies that illustrates how experience and expertise with an array of writing technologies led to valuable new writing practices. The first major innovation in writing technology discussed by Engelbart is a hypothetical writing machine that would have consisted of a special electric typewriter and a stylus. Together, these devices offered what we now call "cut and paste" functionality, allowing writers to type text normally with the typewriter, but also select bits of typed text with the stylus that would subsequently be printed by the typewriter.[3] In his description of the hypothetical machine, he argues that it would facilitate "a new process of composing text." As he explains,

> trial drafts can rapidly be composed from rearranged excerpts of old drafts, together with new words or passages which you insert by hand typing. Your first draft may represent a free outpouring of thoughts in any order, with the inspection of foregoing thoughts continuously stimulating new considerations and ideas to be entered. If the tangle of thoughts represented by the draft becomes too complex, you can compile a reordered draft quickly. It would be practical for you to accommodate more complexity in the trails of thought you might build in search of the path that suits your needs. ("Conceptual" 7)

Engelbart imagined that such a machine would facilitate revision through rapid rearrangement, thereby giving writers more freedom to compose on the page rather than in their heads. He argues, prior to the creation of any kind of word processor, that the ability to move text around easily would make writers more likely to follow complex trails of thought because a messy draft with tangential ideas could be compiled into a "reordered draft." While the above description may suggest that Engelbart's vision of computer tools is too deterministic, later accounts of his work with prototype technologies reveal he is well aware that writers must expend serious

effort—this hypothetical device is not creating more complex drafts on its own. Later in that same report, Engelbart suggests that it is writers who will find themselves wanting to "incorporate more complex procedures in [their] way of doing things" ("Conceptual" 7), as opposed to arguing that the device will somehow create better drafts through its functions.

By 1965, Engelbart had designed and begun to write with a computer. Although it differed in design from the hypothetical writing machine described above by forgoing the stylus, it did not differ much in practice. The Z-code system, as it was called, was an off-line computer program—a designation that, prior to the Internet, referred to the interactivity of a computer. Off-line computing meant typing something onto paper-tape, punch cards, or some other storage system that would later be fed into the computer for processing. Working on-line meant that the computer responded immediately to user input. Engelbart described his process of writing with an off-line system in terms of efficiency—since he didn't have access to a computer for more than four hours a day during his early research, efficiency during on-line time was paramount. With the computer helping to rearrange his drafts according typed commands, his own writing style became more efficient:

> When I work at a typewriter, the ideas develop as the words flow, and when the thinking is hard, the sentences get butchered, with much insertion and deletion (penciled arrows and interlineation, etc.), until they say what my current thought requires. After a page or two of this, the real ideas begin to emerge and the whole thing needs reorganizing. . . . It occurred to me that if I used a private "tapewriter" (our term for a paper-tape-punching typewriter) in my office for such thought development I could then feed the paper tape into the computer at the beginning of my on-line working session, and thus take direct advantage of off-line thinking time. (*Augmenting* 24)

Writers today may have a hard time seeing such an off-line system as efficient. Common keyboard commands in contemporary software are typically simple and take effect instantly. Off-line commands, on the other hand, were typed into the text itself and had no effect until the document was reprinted by the computer. Such a system, one can imagine, would easily break a writer's concentration by requiring constant shifting between typing complicated codes and composing.[4]

Engelbart, however, found the system useful enough in his everyday work to continue developing it. He describes using the Z-code system to write his second report on the augmentation project:

> The [writing] techniques were not smooth: the system would occasionally backfire from unwitting violation of a subtle point in convention; many extra hours went into composition of some parts because the system was not compatible with the thought processes of the author; and considerable constriction was experienced. However, the feeling of being tied into a system made a surprising difference in the work itself, revealing new kinds of inadequacies, and new kinds of possibilities. (*Augmenting* 3)

Here Engelbart takes on a detached tone, writing about himself in third person and with passive voice ("constriction was experienced"). As test subject for his experiments, but also the designer of them, and thus heavily invested in the outcome yielding results favorable for his program in augmentation, Engelbart occupied several potentially conflicting positions at once, both in real life and rhetorically in the report. He did not want to suggest that his military contract funding was being misspent on technologies that did not work or were too much trouble. Therefore, any difficulty with the writing system became, in his report, a sign of the potential for augmented activity once the human and machine had co-evolved a bit more. Although his attention to the "new kinds of possibilities" the Z-code system afforded may have been prompted by his need to show progress, it also illustrates the value of keeping the usefulness of writing technologies in mind, even when the usability is lacking.

In a passage reminiscent of his 1963 projections regarding the use of a hypothetical writing machine, Engelbart observes that with the Z-code system

> I found that for stretches where the words flowed relatively easily, the system was quite helpful, and with later recycling capability [the ability to store text on magnetic tape for use in other documents/files] I think it will prove to be very helpful over a wide range of writing and rewriting applications. For tougher stretches of composition, the system got in the way. For some of these sections, I went back to more primitive means, a ballpoint pen, with lots of scratch-outs, arrows, and marginal scribbling and a skilled typist to translate to clean text. This was much preferable—and I admit this with no sense of failure regarding the utility of the eventual improved Z-code System. (*Augmenting* 46)

In this passage, Engelbart begins to sketch out the ways such computer-aided writing could become a useful activity different from typewriting. When he struggled to find the right words "the system got in the way" and print tools were more useful, which Engelbart found in keeping with his theories on co-evolution. The computer introduced changes with its storage capability, though, affording writing or re-writing through pasting together existing text.

By 1968, Engelbart could write online with a new computer (the On-line System, or NLS) and leave the Z-code system (and its limitations) behind.[5] He described the new affordances of the NLS in a report for NASA, then a major funder of ARC's research. The report states that in order to reduce the stress introduced by learning to write with the NLS, it should be used for long periods of time, six hours or more, to accustom the writer to its features. The following quoted section pulled from the report was written by Engelbart "at the end of an eight-hour working session" (*Human* 48). To begin, Engelbart offers a more detailed description of the composition-by-rearrangement process he wrote about in 1963 and 1965:

> To accommodate and preserve a thought or piece of information that isn't related to the work of the moment, one can very quickly and easily insert a note within the structure of a file at such a place that it will neither get in the way nor get lost.
>
> Later, working in another part of the file, he can almost instantly (e.g., within two seconds) return to the place where he temporarily is storing such notes, to modify or add to any of them.
>
> As any such miscellaneous thought develops, it is easy (and delightful) to reshape the structure and content of its discussion material.
>
> It is also easy and delightful to see a number of initially disconnected notes mature to the point where they are ripe to be integrated under one 'topical' heading. (*Human* 48-9)

The searching and rearranging practices described by Engelbart were afforded by the hierarchically-structured writing environment of the NLS. Thus, during a writing session with the NLS, users could move disconnected statements to a special branch in the hierarchy of the document, and access this branch at various times to pull statements from it into the main sections of the document.

Bootstrapping Composition

In the previous section I have emphasized Engelbart's reflective practices with regards to the co-evolution of his writing tools and practice as he engaged in the bootstrapping method. While the bootstrapping method of augmenting intelligence was enacted in ARC by using experimental technologies to design more experimental technologies, I believe the general approach has value for scholars not involved in technology creation. At its core, the bootstrapping method was oriented toward continually exploring the possibilities available from the intersection between situated tasks and available technologies. Too often, however, technologies are adopted based on their ability to seamlessly match with existing ways of working—thereby forgoing the possibility of reimagining these practices based on the affordances of different technologies.

Engelbart suggested that people's rejection of innovative technologies stemmed from their inability to fully imagine what it would be like to work with them. In the 1965 Z-code report, he attempts to counter anticipated criticisms of his goal to write "live" on a computer with a screen:

> Considering this Service facility solely from the point of view of being able to compose and modify displayed information—to erase that word, move this sentence up and have the rest of the text rearrange itself accordingly, begin a new paragraph there, insert the following word, change that symbol to logical AND—makes it extremely hard to place a value upon that facility until it has been integrated into a coordinated and practiced way of working. A man could predict that he would have no use for such a facility, but until he appreciated from experience, what it was like to have the added capability, his evaluation would be premature. (*Augmenting* 19-20)

Engelbart's sense that new tools bring with them unimagined possibilities is echoed by Bruno Latour in a discussion of the affordances of tools. "With [a hammer] in hand," Latour argues,

> the possibilities are endless, providing whoever holds it with schemes of action that do not precede the moment it is grasped. . . . Those who believe that tools are simple utensils have never held a hammer in their hand, have never allowed themselves to recognize the flux of possibilities that they are suddenly able to envisage. (250)

The bootstrapping method continually leads users toward the recognition of, as Latour says, the "flux of possibilities they are suddenly able to envisage" with their new tools.

As a field of writing scholars and teachers, we should investigate ways we might collectively apply bootstrapping strategies in our departments, writing programs, and professional organizations. Furthermore, we should focus particularly on developing and teaching bootstrapping methods directly to writers. The bootstrapping method as practiced at ARC facilitated the production of an innovative working system consisting of new hardware, software, writing practices, and collaborative working methods. As writing researchers and scholars, we too can use this method to facilitate assemblies of similarly innovative systems. Under this more personalized conception of bootstrapping, writers maintain a critically reflective attitude toward their working goals and available tools in an effort to leverage new possibilities that may be lurking unnoticed within existing software, hardware, or situations. One particularly rich example from within the field will illustrate the benefits of this approach.

At the University of Georgia, a group of researchers have developed and implemented an electronic portfolio system, named <emma>, for the first-year writing program that provides students and teachers with the means to "tag," or label, structural and rhetorical aspects of electronic texts and then display tagged material in several different ways. For instance, students might be asked to tag their thesis statements and supporting evidence and claims in an argumentative essay. Then, the <emma> system provides three different options for displaying the text: "[1] as plain text; [2] marked so as to reveal [through colored highlighting] the relationship between thesis and support; and [3] marked to reveal only the thesis and its supports in a 'collapsed' post-facto outline" (Desmet et al. 36). Furthermore, essays tagged in this way and collected into an archive of all student work enables the collection of valuable data. As Desmet et al. describe,

> With a corpus of work marked-up by the student and the evaluator, it should be possible to search, for example, for all the excellent instances of thesis sentences in any a given assignment, for all examples of particular errors, or for all uses made of a selected quotation or reference. (27)

What makes <emma> particularly relevant to my argument here is the way its development process fits the bootstrapping model. <emma> is built with a technology similar to HTML called XML. These researchers did not begin with an idea to use XML in first year composition classes as a component of an overarching eportfolio document management system. Instead, they describe,

> With a basic understanding of the power and flexibility of XML, several of us began to experiment, in classes at the undergraduate and graduate level, with how XML could be leveraged to manipulate documents, from primary texts to student responses to text. From there, a grant was written and a graduate seminar composed to nurture such ideas. (27-8)

Attention to XML and its affordances came before any clear sense of how to apply it to particular writing situations or pedagogical goals. Like the researchers at ARC, the <emma> team at Georgia began to use the technology in their courses in an effort to see what kinds of work it could do. Throughout their account of the project their attention to the usefulness of XML and <emma> is paramount. As they remark toward the end of the article: "we need to consider ways to make <emma> substantially more than an automated highlighter" (45). Their case illustrates the crucial reflective components of bootstrapping that ask researchers to consider both the affordances of the technology and the demands of particular situations.

Additionally, their description of the <emma> project alleviates concerns that the bootstrapping method, by attending too closely to the demands of technology, deviates into technological determinism. It is clear from their description of a graduate course where participants argued over the pedagogical application of XML tagging that they were not overrun by enthusiasm with the tool and were not letting it dictate the goals or means of teaching writing. Furthermore, they discuss their efforts to customize their deployment of XML by modifying existing software projects to better align with their goals or needs (28), again showing that one particular tool or even one version of that tool is not determining their use. At the same time, though, the two narratives of particular classroom encounters with <emma> shared in the article show that teachers have made many changes to their existing ways of working in order to accommodate many of <emma>'s demands, and continue to reflect on ways they might cater more fully to these demands. In other words, bootstrapping departs somewhat from our field's depictions of the dangers of technological determinism, where the technology must always be considered last, after all pedagogical goals have been developed. In an important way, XML technology is actively shaping the kinds of goals teachers imagine are possible in a writing course or individual class session. However, clearly these researchers are not abandoning existing commitments to teaching process approaches and argumentative structures. The XML technologies and the <emma> system overall have become important actors that have an influence within

the classroom and the curriculum, but in Desmet et al.'s account <emma> is neither the most important actor nor the one with the most influence in all settings or situations.

In addition to the collaborative, institutional approach to bootstrapping represented by the <emma> project, more small-scale, even personal, approaches are also possible. A general practice for such an approach would consist of exploiting the materials at hand to engage new ways of working (either new tools or processes). One example of this practice would be making use of versioning features in writing software and other programs. With word processing software, it is simple to save multiple versions of documents as multiple files, perhaps with a naming scheme indicating draft number or date. Remembering the differences between these drafts during later review can be difficult, though, so this system can quickly lead to confusion. The "compare documents" feature, available in Microsoft Word and OpenOffice Writer, can alleviate the problem somewhat by quickly identifying any differences in the text of two particular files. While writing my dissertation, I came to rely on these features so that I could make wholesale changes to the text (moving or deleting large sections, rewording paragraphs) without having to worry about how difficult it might be to undo those changes.

For subsequent writing projects, however, I have leveraged my experience with close review of previous drafts to explore new programs that offer more support for these practices. I now use a "distributed version control system," a class of programs popular with computer programmers who use them to maintain histories of changes to large libraries of computer code (one popular DVCS is Git, another is Mercurial). This software saves each version of a file or group of files, along with annotations regarding the changes, in a simple interface that depicts the changes as a chronological timeline. Each time a new version of a file is saved, I am prompted to write a brief note regarding the changes I have made. These notes make it simple to identify particular sections of drafts I might want to restore. Furthermore, if files are saved in "plain text" format (the way most programmers save their code files), powerful comparison tools can be used to merge particular drafts or to pinpoint particular changes to revert while leaving others alone. These tools make it possible to create multiple "branches" of a text, where two different versions are updated independently and then merged later—allowing a writer to experiment with different ways of revising the same text or letting multiple writers work concurrently on a text and have their changes merged later (if they both make changes to the same portion of text, the program prompts the user to choose which change to keep). Although Mercurial or Git can be used successfully with only a basic understanding of these main features, these programs offer many more features, designed mainly for programmers, that writers might successfully appropriate. Additionally, gaining familiarity with these programmers' tools allows writers to transfer that experience to other programs—such as specialty text editors or search mechanisms (like grep or regular expressions). As with the

<emma> example, with this bootstrapping approach neither the tools nor the practices take precedence—they are used reciprocally to explore new ways of working and imagining what is possible.

Conclusion

The bootstrapping method I have described here supports the calls made by Selber, Cynthia Selfe, and others who ask that writers maintain reflective and critical attitudes toward their use of technology. What I believe an orientation toward bootstrapping adds to these efforts is a long-term strategy of co-evolution with our tools. I am far from advocating that more writing scholars should learn to create computer technologies. Instead, a strategy of continually exploring the affordances of different combinations of technologies and practices will either give us a way of transforming our work, or, as individuals and as a profession, a stronger voice and more specific vocabulary for our requests to designers of writing and teaching tools. Efforts to make writing tools useful for every person in every context rarely encourage all users to explore how their work may be transformed through their use of such tools. To move past these limits, the bootstrapping method, as I have proposed it, keeps a firm balance between individual strategies of reflection based on one's own use on the one hand, and group-oriented strategies on the other.

Much as we share our teaching practices through both informal and formal channels, we can achieve a balance between individual, idiosyncratic use and more collective definitions of effective use through sharing our computer expertise. Too often computer instruction remains prescriptive and solely focused on manipulating the most obvious features of one program to accomplish tasks as the software designers intended. Sharing the results of our bootstrapping efforts could supplement those sessions with materials like informal narratives posted on listservs or recounted at conferences, article-length descriptions of new practices afforded by a unique combination of tools, or blog posts with screen capture videos. These ways of sharing expertise need not function prescriptively to show people the only way to use a program, but instead offer a myriad of personal ways of working with situated tasks. Furthermore, the differences in institutional computing resources make it unlikely that any particular combination of tools and practices could be prescribed to a large swath of writers or writing programs. Instead, these descriptions of computer expertise could function similarly to the resistant, alternative model of sharing lore that Bruce Horner contrasts with North's models of lore.[6] Rather than distributing expertise throughout the field prescriptively, like a "straightforward transmission of commodified knowledge" (Horner 199), highly situated stories of assembling software and practices could motivate and inspire equally situated responses. If, as Horner argues, "theories inhere as much in such material practices [of composition] as they do in statements" (201), then increased

attention to computing practices through bootstrapping functions as relevant and crucial work for our field.

Notes

1. While discussions of the instrumental aspects of software use are rare in rhetoric and composition more broadly, such work does appear in venues dedicated to the exploration of computers and writing (e.g., *Computers and Composition* and *Kairos*) or technical communication. My point, though, is that such practical knowledge should not be pushed to the margins of the discipline, but instead be featured alongside other discussions of the many components that make up writers' processes.
2. In the interest of full disclosure, Johnson's article and subsequent blog posts encouraged me to try DevonThink and my own techniques for using it were inspired by his setup.
3. Surprisingly, although Engelbart would later invent the computer mouse to increase the ways people interacted with computers, he did not imagine using it like the stylus, as we do now, to select text on the computer screen. It was book editors and publishers who later suggested this function for the mouse (see Moggridge).
4. The Z-Code is illustrated in the lines below (the first line of each example represents what was typed, and the second, bolded line represents the computer's output):

   EXAMPLE ZC-3
   A SIMPLE INSERTION FIRST .Z1I4W(EXAMPLE)Z2I
   A simple insertion example first.
   DELETE AS AN EXAMPLE A GIVEN PRIOR WORD .Z1I7W(Z1W)Z2I
   Delete as an example a prior word.
   REPLACE ONE ITEM WITH ANOTHER .Z1I4W(Z1W WORD)Z2I
   Replace one word with another. (Engelbart, *Augmenting* 36)

   The first code in each example (Z1) alerts the computer that a command is being issued. The next part gives the position to be affected (in number one, the fourth word in the sentence). Next is either the word to be inserted, or a command to delete the word directly before the position specified. Last is the closing command, alerting the computer that the command has finished.
5. A famous videotaped demonstration of the NLS can be seen at <http://1968demo.org> or using Google Video: <http://video.google.com/videoplay?docid=-8734787622017763097>.
6. Horner contends that North's attempts to show the value of lore and Practitioner knowledge ultimately backfire because North continues to judge lore by the "usual criteria for achieving status as disciplinary knowledge" (Horner 176). Instead of positioning lore as contributing to "reified knowledge," Horner draws on depictions of the practical knowledge of lay workers (discussed by Giddens and others) to argue that the knowledge that constitutes lore "exists only in its practice rather than representing a storehouse of tools on which to draw" (177).

Works Cited

Carr, Nicholas. *The Shallows: What the Internet Is Doing to Our Brains*. New York: W. W. Norton & Company, 2010. Print.

Desmet, Christy et al. "<emma>: Re-forming Composition with XML." *Literary and Linguistic Computing* 20.Supplement 1 (2005): 25-46. Print.
Dilger, Bradley. "Ease and Electracy." *New Media/New Methods: The Academic Turn From Literacy to Electracy*. Ed. Jeff Rice and Marcel O'Gorman. West Lafayette: Parlor, 2008. 109-38. Print.
Edbauer Rice, Jenny. "Rhetoric's Mechanics: Retooling the Equipment of Writing Production." *CCC* 60.2 (2008): 366-87. Print.
Engelbart, Douglas C. "A Conceptual Framework for the Augmentation of Man's Intellect." *Vistas in Information Handling: The Augmentation of Man's Intellect by Machine*. Ed. Paul W. Howerton and David C. Weeks. Washington: Spartan, 1963. 1-30. Print.
—. *Augmenting Human Intellect: Experiments, Concepts, and Possibilities*. Menlo Park: Stanford Research Institute, 1965. Print.
—. *Human Intellect Augmentation Techniques*. Menlo Park: Stanford Research Institute, 1969. Print.
—. Interview by Judy Adams and Henry Lowood. "Englebart Oral History Series: Interview 1." *Stanford and the Silicon Valley: Oral History Interviews.* Stanford University Libraries & Academic Information Resources, 19 Dec. 1986. Web. 15 Sept. 2011.
Engelbart, Douglas C., Richard Watson, and James Norton. "The Augmented Knowledge Workshop." *Proceedings of the AFIPS Conference June 4-8, 1973*: *National Computer Conference.* New York, 1973. 9-12. Print.
Horner, Bruce. *Terms of Work for Composition: A Materialist Critique*. Albany: SUNY P, 2000. Print.
Johnson, Robert R. *User-Centered Technology: A Rhetorical Theory for Computers and Other Mundane Artifacts*. Albany: SUNY P, 1998. Print.
Johnson, Steven. "Tool for thought." N.p., 29 Jan. 2005. Web. 13 Sept. 2011.
—. "Tool for Thought." *The New York Times.* New York Times, 30 Jan. 2005. Web. 2 Sept. 2010.
Kratzke, Peter. "Recopying to Revise: Composition in an Old Key." *Composition Studies* 36.2 (2008): 9-22. Print.
Latour, Bruno. "Morality and Technology: The End of the Means." *Theory, Culture & Society* 19.5/6 (2002): 247-60. Print.
Licklider, J.C.R. "Man-Computer Symbiosis." *A History of Personal Workstations*. Ed. Adele Goldberg. New York: ACM Press, 1988. 131-40. Print.
Mirel, Barbara. *Interaction Design for Complex Problem Solving: Developing Useful and Usable Software*. San Francisco: Morgan Kaufmann, 2003. Print.
Moggridge, Bill. *Designing Interactions*. Cambridge: MIT Press, 2006. Print.
Selber, Stuart A. *Multiliteracies for a Digital Age*. Carbondale: Southern Illinois UP, 2004. Print.
Selfe, Cynthia L. *Technology and Literacy in the Twenty-First Century: The Importance of Paying Attention*. Carbondale: Southern Illinois UP, 1999. Print.
Slattery, Shaun. "Technical Writing as Textual Coordination: An Argument for the Value of Writers' Skill with Information Technology." *Technical Communication* 52.3 (2005): 353-60. Print.
Spinuzzi, Clay. *Tracing Genres Through Organizations: A Sociocultural Approach to Information Design*. Cambridge: MIT Press, 2003. Print.
Waldrop, M. Mitchell. *The Dream Machine: J.C.R. Licklider and the Revolution That Made Computing Personal*. New York: Viking, 2001. Print.