Chapter 1

Revealing Errors

Benjamin Mako Hill
Massachusetts Institute of Technology

Introduction

In "The World is Not a Desktop," Marc Weisner, the principal scientist and manager of the computer science laboratory at Xerox PARC, states that, "a good tool is an invisible tool."[1] Weisner cites eyeglasses as an ideal technology because with spectacles, he argued, "you look at the world, not the eyeglasses." Although Weisner's work at PARC played an important role in the creation of the field of "ubiquitous computing," his ideal is widespread in many areas of technology design. Through repetition, and by design, technologies blend into our lives. While technologies, and communications technologies in particular, have a powerful mediating impact, many of the most pervasive effects are taken for granted by most users. When technology works smoothly, its nature and effects are invisible. But technologies do not always work smoothly. A tiny fracture or a smudge on a lens renders glasses *quite* visible to the wearer.

Anyone who has seen a famous "Blue Screen of Death"—the iconic signal of a Microsoft Windows crash—on a public screen or terminal knows how errors can thrust the technical details of previously invisible systems into view. Nobody knows that their ATM runs Windows until the system crashes. Of course, the operating system chosen for a sign or bank machine has important implications for its users. Windows, or an alternative operating system, creates affordances and imposes limitations. Faced with a crashed ATM, a consumer might ask herself if, with its history of rampant viruses and security holes, she should *really* trust an ATM running Windows.

Technology is powerful for a number of reasons. First, technologies make previously impossible actions possible and many actions easier. In the process, they frame and constrain possible actions. For example,

communication technologies allow users to communicate in new ways but constrain communication in the process; in a very fundamental way, these technologies define what their users can say, to whom they say it, and how they can say it—and what, to whom, and how they cannot. Second, in a related sense, technology mediates. Acting as an intermediary, technology stands between any given user and an other, the users' goals, or information. As a function of its operation, technology transforms information as it passes it on. In the process, technology gives its operators and providers the immense power associated with the ability to monitor or change users' messages—often without users' knowledge. Third and finally, in a very broad sense, technology hides and abstracts by presenting users with a series of "black boxes." By intentionally controlling what users can see and understand, technology designers' and providers' implicit power becomes more fully entrenched and firmly enforced. In all three cases, the values of technology designers and providers play an important role in shaping users' experience.

Humanities scholars understand the power, importance, and limitations of technology and technological mediation. Weisner hypothesized that, "to understand invisibility the humanities and social sciences are especially valuable, because they specialize in exposing the otherwise invisible."[2] Technology activists, like those at the Free Software Foundation (FSF) and the Electronic Frontier Foundation (EFF), understand this power of technology as well. Largely constituted by technical members, both organizations, like humanists studying technology, have struggled to communicate their messages to a less technologically savvy public. Before one can argue for the importance of individual control over who owns technology, as both FSF and EFF do, an audience must first appreciate the power and effect that their technology and its designers have. To understand the power that technology has on its users, users must first *see* the technology in question. Most users do not. Both the EFF and the FSF have struggled in their appeals to technology users who are not also technologists and developers—the communities both organizations are explicitly dedicated to serve.

Errors are underappreciated and underutilized in their ability to reveal technology around us. By painting a picture of how certain technologies facilitate certain mistakes, one can better show how technology mediates. By revealing errors, scholars and activists can reveal previously invisible technologies and their effects more generally. Errors can reveal technologies and their power and can do so in ways that users of technologies confront daily and understand intimately.

Affordances

Errors can reveal distinct and overlapping aspects of the technologies that mediate our lives and the designers of those technologies. First, and perhaps most fundamentally, errors can reveal the affordances and constraints of technology that are often invisible to users. Through these affordances and constraints, technologies make it easier to do some things, rather than others, and either easier or more difficult to communicate certain messages. Errors can help reveal these hidden constraints and the power that technology imposes.

The misprinted word

Catalyzed by Elizabeth Eisenstein, the last 35 years of print history scholarship provides both a richly described example of technological change and an analysis of its effects.[3] Unemphasized in discussions of the revolutionary social, economic, and political impact of printing technologies is the fact that, especially in the early days of a major technological change, the artifacts of print are often quite similar to those produced by a new printing technology's predecessors. From a reader's purely material perspective, books are books; the press that created the book is invisible or irrelevant. Yet, while the specifics of print technologies are often hidden, the affordances of any particular print technology has important effects on what is printed and how, effects that are often exposed by errors.

While the shift from scribal to print culture revolutionized culture, politics, and economics in early modern Europe, it was near-invisible to early readers. Early printed books were the same books printed in the same way; the early press was conceived as a "mechanical scriptorium."[4] Gutenberg's black-letter Gothic typeface closely reproduced a scribal hand. Of course, handwriting and type were easily distinguishable; errors and irregularities were inherent in relatively unsteady human hands.

Printing, of course, introduced its own errors. As pages were produced *en masse* from a single block of type, so were mistakes. While scribes would reread and correct errors as they transcribed a second copy, no printing press could. More revealingly, print opened the door to whole new categories of errors. For example, printers setting type might confuse an inverted *n* with a *u*—and many did. Of course, no scribe made this mistake. An inverted *u* is only confused with an *n* due to the technological possibility of letter flipping in movable type. As print

moved from Monotype and Linotype machines, to computerized typesetting, and eventually to desktop publishing, an accidentally flipped *u* retreated back into the realm of impossibility.[5]

Most readers do not know how their books are printed. The output of letter presses, Monotypes, and laser printers are carefully designed to produce near-uniform output. To the degree that they succeed, the technologies themselves, and the specific nature of the mediation, becomes invisible to readers. But each technology is revealed in errors like the upside-down *u*, the output of a mispoured slug of Monotype, or streaks of toner from a laser printer.

Changes in printing technologies after the press have also had profound effects. The creation of hot-metal Monotype and Linotype, for example, affected decisions to print and reprint and changed how and when it is done. New mass printing technologies allowed for the printing of works that, for economic reasons, would not have been published before. While personal computers, desktop publishing software, and laser printers make publishing accessible in new ways, it also places real limits on what can be printed. Print runs of a single copy—unheard of before the invention of the type-writer—are commonplace. But computers, like Linotypes, introduce their own affordances and constraints and render certain formatting and presentation difficult and impossible.

Errors provide a space where the particulars of printing make technologies visible in their products. An inverted *u* exposes a human typesetter, a letterpress, and a hasty error in judgment. Encoding errors and botched smart quotation marks—a *?* in place of a ″—are only possible with a computer. Streaks of toner are only produced by malfunctioning laser printers. Dust can reveal the photocopied provenance of a document.

Few readers reflect on the power or importance of the particulars of the technologies that produced their books. In part, this is because the technologies are so hidden behind their products. Through errors, these technologies and the power they have on the "what" and "how" of printing are exposed. For scholars and activists attempting to expose exactly this, errors are an underexploited opportunity.

Typing mistyping

While errors have a profound effect on media consumption, their effect is perhaps more strongly felt during media creation. Like all mediating technologies, input technologies make it easier or more difficult to

create certain messages. It is, for example, much easier to write a letter with a keyboard than it is to type a picture. It is much more difficult to write in languages with frequent use of accents on an English language keyboard than it is on a European keyboard. But while input systems like keyboards have a powerful effect on the nature of the messages they produce, they are invisible to recipients of messages. Except when the messages contain errors. Typists are much more likely to confuse letters in close proximity on a keyboard than people writing by hand or setting type. As keyboard layouts switch between countries and languages, new errors appear. The following is from a personal email:

> *hez*, if there's not a subversion server *handz*, can i at least have the root password for one of our machines? I read through the instructions for setting one up and i think i could do it. [emphasis added]

The email was quickly typed and, in two places, confuses the characters *y* with *z*. Separated by five characters on QWERTY keyboards, these two letters are not easily mistaken or mistyped. However, their positions are swapped on German and English keyboards. In fact, the author was an American typing in a Viennese Internet cafe. The source of his repeated error was his false expectations—his familiarity with one keyboard layout in the context of another. The error revealed the context, both keyboard layouts, and his dependence on a particular keyboard. With the error, the keyboard, previously invisible, was exposed as an inter-mediator with its own particularities and effects.

This effect does not change in mobile devices where new input methods have introduced powerful new ways of communicating. SMS messages on mobile phones are constrained in length to 160 characters. The result has been new styles of communication using SMS that some have gone so far as to call a new language or dialect called TXTSPK.[6] Yet while they are obvious to social scientists, the profound effects of text message technologies on communication is unfelt by most users who simply see the messages themselves. More visible is the fact that input from a phone keypad has opened the door to errors which reveal input technology and its effects.

In the standard method of SMS input, users press or hold buttons to cycle through the letters associated with numbers on a numeric keyboard (e.g., *2* represents *A*, *B*, and *C*; to produce a single *C*, a user presses *2* three times). This system makes it easy to confuse characters based on a shared association with a single number. Tegic's popular T9 software

allows users to type in words by pressing the number associated with each letter of each word in quick succession. T9 uses a database to pick the most likely word that maps to that sequence of numbers. While the system allows for quick input of words and phrases on a phone keypad, it also allows for the creation of new types of errors. A user trying to type *me* might accidentally write *of* because both words are mapped to the combination of 6 and 3 and because *of* is a more common word in English. T9 might confuse *snow* and *pony* while no human, and no other input method, would.

Users composing SMSes are constrained by the affordances of SMS technology. The fact that text messages must be short and the difficult nature of phone-based input methods has led to unique and highly constrained forms of communication like TXTSPK.[7] Yet, while the influence of these technological affordances is profound, users are rarely aware of them. Errors can expose the particularities of a technology and, in doing so, provide an opportunity for users to connect with scholars speaking to the power of technology and to activists arguing for increased user control.

Of course, affordances and constraints are far from arbitrary. In the case of SMS length restrictions, there are historical technical reasons for the limitation. In the case of T9, constraints imposed by small handhelds with a small number of buttons informed that technology's design. But affordances and constraints are ultimately designed and implemented by human designers who have enormous power to impose their own values in their design. This power, and the important design decisions through which it is enacted, is also revealed through errors.

Any T9 user attempting to input profanity onto their phone will run into such an affordance. To avoid the impropriety of suggesting a four-letter word as a possible completion, Tegic removes all profanity from its T9 wordlist. The result, lampooned at one point by the British comedy duo Armstrong and Miller on the BBC, is improbable suggestions like "shiv" and "ducking" (both first suggestions for their respective key combinations) followed by decreasingly plausible options.

As ugly and offensive as they may be to some, "shit" and "fucking" are English words and the inability to type them on a T9 system is an error from the perspective of any user unable to complete a profane message using T9's software. Built from word frequency counts of SMS messages and existing databases, T9's inability to compose profanity is an intentional decision on the part its designers. Not only does the error reflect Tegic's corporate values, it codifies them and forces them onto its users.

The result is that it is easier to write a non-profane message using T9 than it is to write a profane one. It is easier to write "darn" than it is to write "damn." In a small but important way, T9 forces the values of its designers on its users every time to every user who tries to swear using their technology. But in that the result is, from the perspective of the user, in error, it also makes those values visible. In the process, it reveals one of the affordances of a normally invisible technology, points toward a plausible description of "why," and reveals the power of designers, in at least one small way, to determine what users can and cannot say.

Hidden Intermediaries

While affordances constrain what users *can* do or say, the role of technology as an intermediary gives technology designers the ability to change what users say after the effect. In that it transmits, repeats, and copies users messages, communication technologies in particular intermediate as a fundamental function. Especially in online environments, any message is invisibly passed through a barrage of technological intermediaries that can monitor, censor, and even alter its content. To the degree that these technologies are invisible, this intermediation is invisible as well. Of course, errors can reveal the presence of this hidden mediation.

Clbuttic

Tegic's desire to avoid profanity, as described in the previous section, is hardly unique. Their method however, of making swearing possible, but more difficult, than not swearing, is less heavy handed than some other approaches. Another large class of errors results from intermediaries that monitor, censor, and even change user input to avoid profanity. A now famous mistake, repeated several times, has become known as the "clbuttic" mistake.

The term "cbluttic" is the result of a computer program that monitored user input to an online discussion forum and swapped out instances of profanity with less offensive synonyms. For example, "ass" might become "butt," "shit" might become "poop" or "feces," etc. To work correctly, the script must look for instances of profanity between word boundaries—that is, profanity surrounded on both sides by spaces or punctuation.

On a number of occasions, programs did not. As a result, not only was "ass" changed to "butt," but any word that contained the letters "ass" were

transformed as well. The word "classic" was mangled and left as "clbuttic." Today, web searches for "clbuttic" turn up thousands of hits on dozens of independent web sites. In the same vein, one can find references to a "mbuttive music quiz" a "mbuttive multiplayer online game," references to the average consumer is a "pbutterby," a "transit pbuttenger executed by Singapore," "Fermin Toro Jimenez (Ambbuttador of Venezuela)," "the correct way to deal with an buttailant armed with a banana," and a reference to how "Hinckley tried to buttbuttinate Ronald Reagan."[8]

For any reader, each error reveals the presence of an intermediary in the form of an anti-profanity script; obviously, no human would accidentally misspell or mistake the words in question in any other situation.

What is perhaps more impressive than this error is the fact that most programmers do not make this mistake when implementing similar systems. On thousands of web sites, posts and messages and interactions are "cleaned-up" and edited without authors' consent or knowledge. As a matter of routine, their words are silently and invisibly changed. Few authors, and even fewer of their readers, ever know the difference. Each error is a stark reminder of the power that technology gives the designers of technical systems to force their own values on users and to frame—and perhaps to substantively change—the messages that their technologies communicate.

Tyson Homosexual

In the lead up to the 2008 Olympic games in Beijing, One News Now, a news web site run by the conservative Christian American Family Association published an Associated Press article with the headline, "Homosexual eases into the 100 final at the Olympic trials." The AP published no article with such a headline. The first paragraph revealed the source of the error and a hidden intermediary running on the ONN web site in its explanation that, "Tyson Homosexual easily won his semifinal for the 100 meters at the U.S. Olympic track and field trials and seemed to save something for the final later Sunday." Of course, there is no U.S. sprinter named Tyson Homosexual; but there is one named Tyson Gay.

ONN provides Christian conservative news and commentary. One of the things they do is offer a version of the industry standard Associated Press news feed. Rather than just republishing it verbatim, however, AFA runs software to modify the feed's language so it more accurately reflects their organization's values and choice of terminology. They do so with a hidden intermediary in the form of a computer program.

Indeed, the error is similar to the "clbuttic effect" described above—an errant text filter attempting to "clean up" text by replacing offensive terms with theoretically more appropriate ones. Among other substitutions, AFA replaced the term "gay" with "homosexual"—many conservative Christian groups prefer the term homosexual which they argue sounds more clinical and pathological than "gay." In this case, their software changed the name of champion sprinter and U.S. Olympic hopeful Tyson Gay to "Tyson Homosexual."

AFA never advertised the fact that it changed words in its AP feed, and it did so silently. One must assume that most of ONN's readers never realized that the messages they received and the terminology used was being intentionally manipulated by AFA. AFA's substitution, and the error it created, revealed the presence of a hidden script downloading news articles and processing them before they were published. And yet, ONN had been publishing its edited feed for years before anyone realized the presence of the intermediary. The reason, of course, is that unlike the clbuttic example, most of the time, AFA's intermediary worked correctly. It was not until the system produced an error was AFA's invisible mediator thrust into view.

Although few edit the content of the article quite like ONN, every news feed uses technology to parse and edit articles before publishing to some degree. Errors can reveal the presence of these hidden intermediaries. In doing so, these errors can highlight the power that technological designers and service providers have over our communication. Nearly every message we send or receive is as vulnerable to the type of manipulation that the AP's article is.

However, the Tyson Gay error also reveals a set of values that AFA and ONN have about the terminology around homosexuality and the way that these values invisibly frame users' experience of reading an AP article through their system. It is possible that many of ONN's readers know and support AFA's values and their choice of terminology changes. What they did not know is the ways in which the media they're consuming are being automatically and silently manipulated and mediated. AFA's error thrust hidden technology into view, giving a clear image of the power that designers and service providers have over their users.

Opening Black Boxes

As technologies become more complex, they often become more mysterious to their users. While not invisible, users know little about

the way that complex technologies work both because they become accustomed to them and because the technological specifics are hidden inside companies, behind web interfaces, within compiled software, and in "black boxes."[9] Errors can help reveal these technologies and expose their nature and effects. As technology becomes complex, the purpose of technology is to hide this complexity. As a result, the explicit creation of black boxes becomes an important function of technological design processes and a source of power. Once again, errors that break open these boxes can reveal hidden technology and its power.

Google News denuded

Google's News aggregates news stories and is designed to make it easy to read multiple stories on the same topic. The system works with "topic clusters" that attempt to group articles covering the same news event. The more items in a news cluster (especially from popular sources) and the closer together they appear in time, the higher confidence Google's algorithms have in the "importance" of a story and the higher the likelihood that the cluster of stories will be listed on the Google News page. While the decision to include or remove individual sources is made by humans, the act of clustering is left to Google's software.

Because computers cannot "understand" the text of the articles being aggregated, clustering happens less intelligently. We know that clustering is primarily based on comparison of shared text and keywords—especially proper nouns. This process is aided by the widespread use of wire services like the Associated Press and Reuters which provide article text used, at least in part, by large numbers of news sources. Google has been reticent to divulge the implementation details of its clustering engine but users have been able to deduce the description above, and much more, by watching how Google News works and, more importantly, how it fails.

For example, we know that Google News looks for shared text and keywords because text that deviates heavily from other articles is not "clustered" appropriately—even if it is extremely similar semantically. In this vein, blogger Philipp Lenssen gives advice to news sites who want to stand out in Google News:

> Of course, stories don't have to be exactly the same to be matched—but if they are too different, they'll also not appear in the same group.

> If you want to stand out in Google News search results, make your article be original, or else you'll be collapsed into a cluster where you may or may not appear on the first results page.[10]

While a human editor has no trouble understanding that an article using different terms (and different, but equally appropriate, proper nouns) is discussing the same issue, the software behind Google News is more fragile. As a result, Google News fails to connect linked stories that no human editor would miss.

But just as importantly, Google News can connect stories that most human editors will not. Google News's clustering in April of 2006, for example, of two stories—one by *Al Jazeera* on how "Iran offers to share nuclear technology," and one by the *Guardian* on how "Iran threatens to hide nuclear program," seem at first glance to be a mistake. Hiding and sharing are diametrically opposed and mutually exclusive.

But while it is true that most human editors would not cluster these stories, it is less clear that it is, in fact, an error. Investigation shows that the two articles are about the release of a single statement by the government of Iran on the same day. The spin is significant enough, and significantly different, that it could be argued that the aggregation of those stories was incorrect—or not.

The "error" reveals details about the way that Google News works and about its limitations. It reminds readers of Google News of the technological nature of their news's meditation and gives them a taste of the type of selection—and mis-selection—that goes on out of view. Users of Google News might be prompted to compare the system to other, more human methods. Ultimately it can remind them of the power that Google News (and humans in similar roles) has over our understanding of news and the world around us. These are all familiar arguments to social scientists of technology and echo the arguments of technology activists. By focusing on similar errors, both groups can connect to users less used to thinking in these terms.

Show Me The Code

A while ago, software engineer Mark Pilgrim wrote about being prompted with the license agreement shown in Figure 1.1.[11]

Most people have difficulty parsing the agreement because it is not the *text* of the license agreement being presented but the "marked up" XHTML code. Of course, users are only supposed to see the processed

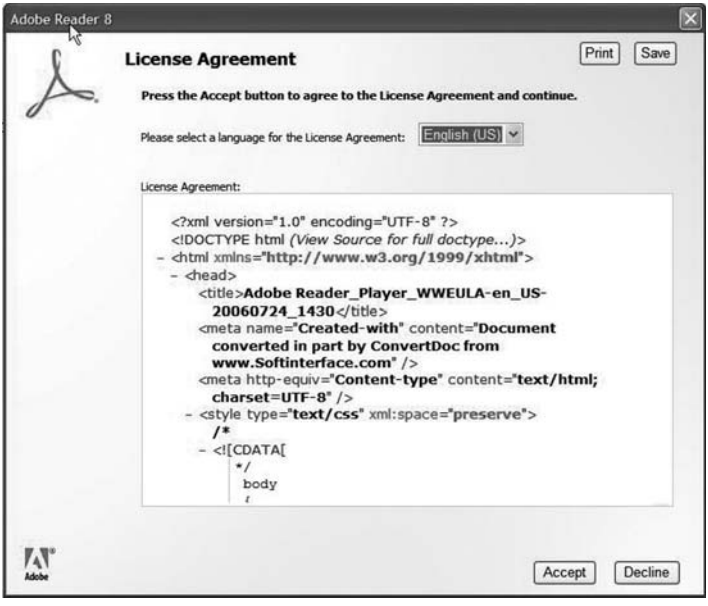

Figure 1.1 An unintended code reveal, as captured by Mark Pilgrim, http://diveintomark.org/archives/2007/11/26/wtf-adobe-reader-8

output of the code and not the code itself. Due to an error, Pilgrim was shown everything. The result is useless.

Conceptually, computer science might be described as a process of abstraction. In an introductory undergraduate computer science course, students are first taught syntax or the mechanics of writing code that computers can understand (e.g., SICP). After that, they are taught abstraction. They will continue to be taught abstraction, in one form or another, until they finish their careers. In this sense, programming is just a process of taking complex tasks and then hiding—abstracting—that complexity behind a simplified set of interfaces. Then, programmers build increasingly complex tools on top of these interfaces and the whole cycle repeats. Through this process of abstracting abstractions, programmers build up systems of incredible complexity. The work of any individual programmer is like a tiny cog in a massive, intricate machine.

The error Mark Pilgrim encountered is interesting because it shows a ruptured black box—an acute failure of abstraction. Of course, many errors tell us very little about the software we're using. With errors like Pilgrim's, however, users are quite literally presented with a view of parts of the system that a programmer was explicitly trying to hide. While the

error Pilgrim showcases is embarrassing for authors of the software that caused it, it is reasonably harmless. Users can understand how a technology works in more detail but learn little that the technology designer would be hesitant to share. In other cases, the view into a broken black box can be shocking.

On two occasions, Facebook accidentally configured their web server to publish source code to the software that runs the Facebook network service—essentially all of the code that, at the time, ran Facebook. Reports at the time show that people looking at the code found little pieces including these code snippets (comments, written by Facebook's authors, are bolded):

```
$monitor = array( '42107457' => 1, '9359890' => 1);
//*Put baddies (hotties?) in here

/* Monitoring these people's profile viewage.
Stored in central db on profile_views.
Helpful for law enforcement to monitor stalkers and stalkees. */
```
[12]

The first block describes a list of "baddies" and "hotties" represented by user ID numbers that Facebook's authors have singled out for monitoring. The second stanza is simply a comment that should be self-explanatory but that raised important concerns for privacy cautious users skeptical about their viewing habits being monitored in collaboration with law enforcement.

Facebook has since published a "source protector" plugin to their web server which will help them, and others who use it, avoid errors like these in the future. As a result, users are less likely to get further views of this type into the Facebook code. Of course, we have every reason to believe that this code, and perhaps other codes like it, still runs on the active version of Facebook. But as long as Facebook's black box works better than it has in the past, users will never know exactly what Facebook is doing with their data.

Like Facebook's authors, many technologists do not want users knowing what their technology is doing. Very often, designers use black boxes to create a better usability experience or a more manageable and modular implementation. Sometimes, like Facebook, technology is designed to do things that users are shocked and unhappy to learn about. Errors that provide clear views into black boxes provide a view into some of what we might be missing and reasons to be discomforted by the fact that many technologists go to extreme lengths to keep users in the dark.

Conclusion

Reflecting on the role of the humanities in a world of increasingly invisible technology for the blog, "Humanities, Arts, Science and Technology Advanced Collaboratory," Duke English professor Cathy Davidson writes:

> When technology is accepted, when it becomes invisible, [humanists] really need to be paying attention. This is one reason why the humanities are more important than ever. Analysis—qualitative, deep, interpretive analysis—of social relations, social conditions, in a historical and philosophical perspective is what we do so well. The more technology is part of our lives, the less we think about it, the more we need rigorous humanistic thinking that reminds us that our behaviors are not natural but social, cultural, economic, and with consequences for us all.[13]

Davidson concisely points out the strength and importance of the humanities in evaluating technology. She is correct; users of technologies do not frequently analyze the social relations, conditions, and effects of the technology they use. Activists at the EFF and FSF argue that this lack of critical perspective leads to exploitation of users.[14] But users, and the technology they use, are only susceptible to this type of analysis when they understand the applicability of these analyses to their technologies. Davidson leaves open the more fundamental question: How will humanists first reveal technology so that they can reveal its effects?

Scholars and activists must do more than contextualize and describe technology. They must first render invisible technologies visible. As the revealing nature of errors in printing systems, input systems, online forums, news feeds, and social networks show, errors represent a point where invisible technology is already visible to users. These errors can reveal several important features of technologies connected to the power that it, and its designers, have over users. In particular, these examples can speak to the power of technological affordance constraints, technologies that act as intermediaries, and the technology that uses "black boxes" in explicit attempts to hide the technology in question. In all three cases, errors can also reveal the values of the technologies' designers. As such, these errors, and countless others like them, can be treated as the tip of an iceberg. They represent an important opportunity for humanists and activists to further expose technologies and the beginning of a process that aims to reveal much more.

Notes

[1] Marc Weisner, "The World is not a Desktop," *Interactions* 1, no. 1 (January 1994): 7.

[2] Ibid., 8.

[3] Elisabeth L. Eisenstein, *The Printing Press as an Agent of Change: Communications and Cultural Transformations in Early-Modern Europe* (Cambridge: Cambridge University Press, 1979).

[4] Ibid.

[5] For further discussion, see, Ottmar Mergenthaler, *The Biography of Ottmar Mergenthaler, Inventor of the Linotype* (New Castle, DE: Oak Knoll Books, 1989); Alvin Garfield Swank and United Typothetae of America, *Linotype Mechanism* (Chicago: Deptartment of Education, United Typothetæ of America, 1926).

[6] Crispin Thurlow. "Generation Txt? The Sociolinguistics of Young People's Text-Messaging," *Discourse Analysis Online* 1, no. 1 (2003), http://extra.shu.ac.uk/daol/articles/v1/n1/a3/thurlow2002003.html.

[7] John Sutherland, "Cn u txt?" *Guardian Unlimited* November 11, 2002, http://www.guardian.co.uk/technology/2002/nov/11/mobilephones2.

[8] Links to web pages that include each of these examples are in a blog post on my *Revealing Errors* blog here: http://revealingerrors.com/clbuttic

[9] Bruno Latour, *Pandora's Hope: Essays on the Reality of Science Studies* (Cambridge, MA: Harvard University Press, 1999).

[10] Philipp Lenssen, "How Google News Indexes, "*Google Blogscoped* July 28, 2006, http://blogoscoped.com/archive/2006–07-28-n49.html.

[11] Mark Pilgrim, "WTF: Adobe Reader 8," *Dive into Mark* November 26, 2007, http://diveintomark.org/archives/2007/11/26/wtf-adobe-reader-8.

[12] Although Facebook has successfully stopped the source code from being distributed, several news articles and blog posts are still available online and on the Internet Archive. Some of these are linked from an article describing this incident on my *Revealing Errors* blog here: http://revealingerrors.com/show_me_the_code.

[13] Cathy Davidson, "When Technology Is Invisible, Humanists Better Get Busy," *HASTAC* June 7, 2007, http://www.hastac.org/node/779.

[14] Richard M. Stallman, in *Free Software, Free Society: Selected Essays of Richard M. Stallman*, ed. Joshua Gay. (Boston: Free Software Foundation, 2002).