

# UC San Diego

## UC San Diego Electronic Theses and Dissertations

### Title

Interactive Time

### Permalink

<https://escholarship.org/uc/item/8kg6q723>

### Author

Moran, Chuktropolis Welling

### Publication Date

2013

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA, SAN DIEGO

Interactive Time

A thesis submitted in partial satisfaction of the  
requirement for the degree of Doctor of Philosophy

in

Communication

by

Chuktropolis Welling Moran

Committee in charge:

Professor Brian Goldfarb, Chair  
Professor Jordan Crandall  
Professor Nitin Govil  
Professor Chandra Mukerji  
Professor Stefan Tanaka

2013



The Dissertation of Chuktropolis Welling Moran is approved and it is acceptable in quality and form for publication on microfilm and electronically:

---

---

---

---

---

Chair

University of California, San Diego

2013

## TABLE OF CONTENTS

Signature Page .....	iii
Table of Contents .....	iv
List of Images .....	vi
Acknowledgements .....	vii
Vita .....	viii
Abstract of the Dissertation .....	ix
I. Introduction .....	1
Software.....	2
Time.....	6
Interactive Time.....	8
Overview .....	10
Note on Methods .....	13
II. Time Theory .....	21
Temporality Functions.....	24
Time as Social Practice .....	36
Conclusion: What Times Are .....	47
III. Times Relating.....	54
How Times Relate .....	56
Time History.....	69
Times Continue to Change .....	87

	Conclusion: Dominant Times and Undo .....	91
IV.	The Historical Undo Command.....	101
	Computing into the 1960s .....	109
	Brown’s Hypertext Project .....	119
	Inventing Undo Again .....	139
	Undoing in Commercial Software.....	146
	Conclusion: What Undo Does .....	171
V.	Architecture of Time .....	189
	The Fast Cycles .....	197
	Memory .....	209
	Network Connection.....	222
	Conclusion: How Conventions Have Joined.....	232
VI.	Conclusion: Interactive Time .....	243

## LIST OF FIGURES

Figure 1.1 The Times They Are a-Changin’ .....	5
Figure 3.1 Christian Siriano .....	59
Figure 3.2 Time at one Mars site, dual faced Earth watch .....	65
Figure 3.3 The Idle Apprentice .....	72
Figure 3.4 The difference between local times was too small .....	79
Figure 4.1 The computer as brain.....	114
Figure 4.2 IBM 2250 monitor .....	121
Figure 4.3 Using FRESS on a terminal .....	129
Figure 4.4 Wang Word Processor’s keyboard.....	135
Figure 4.5 Microsoft Word advertisement in Byte magazine .....	150
Figure 4.6 The consumer market was large.....	157
Figure 4.7 The menu names the action that will be reversed .....	163
Figure 5.1 The fetch cycle .....	200
Figure 5.2 A piece of flash memory .....	211
Figure 5.3 How would you get data off these old disks? .....	215

## ACKNOWLEDGEMENTS

Communication at UC San Diego, thank you for letting weird things grow.

Chapters 2 and 3, in part, are reprints of material as it appears in “Time as a Social Practice,” *Time and Society*. Forthcoming. The dissertation author was the primary investigator and author of this paper.



## VITA

### **EDUCATION**

Ph.D. in Communication, 2013  
University of California, San Diego.

B.A. in Gender Studies with Honor, 2005.  
University of Chicago.

### **SELECT PUBLICATIONS**

*Superactually: Micro-Essays on Post-Ironic Life*. Zero Books. March 2013.

“Playing with Undo: Besides the Lines of Game Time.” *Fibreculture* 16. July 2010.

“Time as a Social Practice.” *Time & Society*. Forthcoming 2013.

“Interactive Time and “Real Time” in Software and Society” *Spectator* 30(1).  
Spring 2011.

“The Generalization of Configurable Being: From RPGs to Facebook.” In  
*Dungeons, Dragons, and Digital Denizens: the Digital Role-playing Game*, ed.  
Gerald Voorhees, Josh Call, and Katie Whitlock, 343-462. New York: Continuum.  
2012

## ABSTRACT OF THE DISSERTATION

Interactive Time

by

Chuktropolis Welling Moran

Doctor of Philosophy in Communication

University of California, San Diego, 2013

Professor Brian Goldfarb, Chair

Computers have enabled new ways to act out time. Unlike times that make authoritative measures or keep everyone synchronized to the same pace, some software lets a person interact with time. The undo command is a good example of this. You can go backward to go forward. That empowers computer users. Interactive time is now a normal part of what it means to use a computer to do something. This is significant because people use computers to do many things.

## **I. Introduction**

I already knew, from a lifetime of using computers, that the time described by physicists, philosophers, and theologians did not describe the time I knew on computers. In different programs, action can start or stop in an instant, slow or reverse, be stored and reloaded, or left on pause indefinitely. In software, these are not impossibilities or even special cases of time. These are regular parts of using the machine.

I knew this, intuitively. But, like many, I lacked the words to articulate it, to respond to it, to discuss it with others, or to build from it. How can time be one thing in one program but be so different in another, and different again outside the computer? Why do these times exist, how do they work, and what are their effects? These became my questions.

In many cases, new computer-enabled times have had clear impacts. Digital video recorders have changed television by fragmenting the audience, concentrating viewer attention on favorite shows, and letting people skip the ads. For music, the trade of individual files has encouraged listening on shuffle, tuning out the world whenever in public, impulse buys of memorable songs, and an enthusiasm for tracks that sound good when they come up unexpectedly. Autosave, file repositories, and backup systems mitigate the impact of crashes and can keep older versions of documents readily accessible. There are many other cases, such as online calendars, multitasking in the operating system, loops and stutters in video editing, or the strange flow of time in the hit indie game *Braid*. Each is different and people tend to speak of

each one separately. But the trend should be clear enough: software gives us new ways to do time.

### Software

To think of other times is not new. Divine times, slowed time, time that does not pass, time that can be revisited, dreamed or rewritten, Tristram Shandy's episodic existence, Gulliver's travels to backwards lands, Rip van Winkle's slumber through historic change, the prophetic power of a Nostradamus, or Christian millenarianism each represent different variations on time.

In the last hundred years, some fantasies about time have become regular parts of popular culture, forming into a rough set of conventions for imagining time's strangeness. Since Mark Twain's 1889 *A Connecticut Yankee in King Arthur's Court* and H.G. Wells's 1895 *The Time Machine*, time travel has become one of the most popular forms of imagining strange cases of time. These fantasies make variations on a timeline, allowing one to travel across it, connect it back to itself, reverse it, slow it down, or speed it up. Related ideas, inspired by the same conventions but less strict about the timeline, can be found in characters who slow or pause time (as in *Heroes* and *The Matrix*) or who wander about between times (as in *The Time Traveler's Wife* and *Millennium Actress*). These fictions have given us a range of scenarios concerning what time is and how it might work differently. These scenarios guide the imagination.

Strange times in software are different because software codes fantasies into social practices that are widely used. In programming interfaces, in designing systems,

and in writing code that will run programs, authors are creating hypothetical ways of living and doing things for a future world. Like science fiction, programs are exercises in cognitive estrangement and speculations about how people might deal with unknown tools and situations in the adaptation to changing conditions of the future. Authors of software write out scenarios for interaction that can, for an actively engaged user, become material practices used on a regular basis. Undoing an action only reverses the flow of events at the level of representation, for example, but it is exactly at this register where work takes place. Jean Baudrillard's claim that simulations without originals now precede what they represent (Baudrillard 1981) can be restated in positive terms: simulations actively produce conditions of the real that allude to familiar terms even as they differ from them. Artifice does not subtract from reality, but adds its products to a world that includes computer files and hanging files, undoable time and clock time, desk chairs, hard disks, and human memories.

Those making software have been tremendously creative and prolific, writing for a large, committed, and growing user base. Without those billions of us who use computers, times invented by software might amount to mere speculations. "Software constructs sensoriums" (Fuller 2003, 19). In the hours spent at the computer, these artificial sensoriums become ours. Staring at the screen, we see whatever authors of software have dreamed up for us to experience. Clicking through links on a website, zooming into a photo, or editing a macro, we accept and work with meanings, affects, and perceptions structured by software. The demand for innovation, lucrative economics of software production, and promise of new technology to enhance the

capacity of the individual have drawn fantasies of time out from the imagination and put them into practice.

The new fantasies of time that users live out through software have created new conventions by which we imagine other times. The timeline is not gone, but it no longer represents constant flow. The clock remains, but what it measures has become of lesser importance. On the computer, time may start and stop at the command of the user. Synchronization depends on sequences of transmitted data more than on finely calibrated clockworks, designed to run at the same speed forever. Tiny lengths of time that fit within a second have become very important, filled with milliseconds and cycles, bundled and swapped too quickly for the user to notice.

Increasingly, social life runs on software. Laptops, tablets, smartphones, mp3 players, GPS devices, and digital cameras are only part of the story. Banking, business communications, talking to friends, making video, watching video, submitting applications, choosing restaurants, ordering food, playing games, meeting friends, clipping coupons, planning travel, and sharing photos are only some of the social behaviors that computers have absorbed partially or completely.<sup>1</sup> The spaces in which social life occurs involve software systems at many levels, from objects and personal devices to surveillance and computer-controlled infrastructure (Kitchin and Dodge 2011; Manovich 2008).

---

<sup>1</sup> *The Matrix* (Wachowski and Wachowski 1999), though usually debated as presenting a philosophical proposal, can be remembered as simply a *metaphor* for the simultaneous imprisonment and empowerment of being one small body in a computerized world.

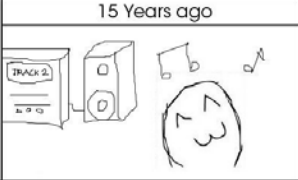
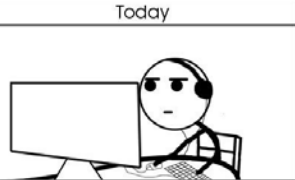

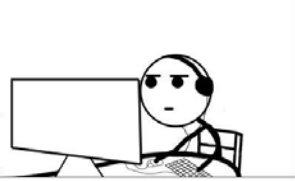

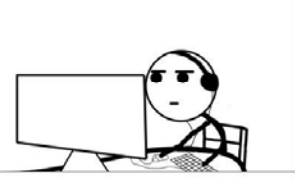

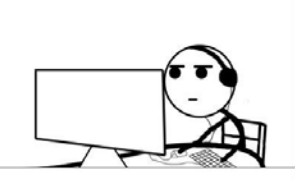

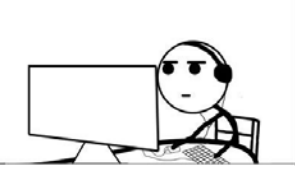
	15 Years ago	Today
Listening to music		
Watching a movie		
Contacting people		
Reading the news		
Making Music		

Image 1.1 “The Times They Are A-Changin’” by Reddit user Gordondel

Software organizes the agency of computers, determining how they will make decisions. This is the central tenet of the developing field of software studies:

[E]verything is governed, enframed and molded by software-mediated processes, while the systems/people creating and overseeing such processes have little ability or power to subject them to doubt, debate, analysis, reinterpretation or control by the public, philosophy, democratic institutions or humanist system of coordinates – whatever each of these may still be able to offer” (Goriunova 2011).

Software is changing how we live, but these changes are not subject to reasoned consideration, whatever reason's value may be. Software studies wonders what these changes are.

The academic exploration of the software mediated environment has only just begun. By looking at how software has been produced, how it has been used, how it can be modified, and how these seemingly technical topics relate to larger social forces and situated practices, software studies can learn new ways to think about computing and its influence (Fuller 2003; Fuller 2008; Manovich 2008). Software can exert an influence behind the scenes or in plain view, by mediating human action or by automating work.

My project looks at times common to software, with a special emphasis on those that give the user control over time. I use the term *interactive time* to refer to practices of time that allow the user some local modulation of the function of temporality, such as that provided by the undo command's instant reversal of action. In the course of this dissertation, I undertake careful consideration of the term "time," provide a case study of the undo command, and describe some infrastructural conventions of time in software. The goal is to understand time in software and to build from that understanding.

## **Time**

This study falls into a long tradition of looking at time in media. Although it is a familiar lament that little research has been done on the topic of time and society,



this is in fact the name of a journal that is more than twenty years old. Different projects concerning time are often about quite different aspects of time (e.g. punctuality or future-orientation) and therefore address themselves more enthusiastically to other fields (e.g. anthropology of colonialism or sociology of class), rather than to comrades studying time (Nowotny 1992).

Transdisciplinary work on time and society suggests that time is a social construction enacted by situated actors, only some of which are human (Nowotny 1994; Glennie and Thrift 2009; Adam 1990). The basic idea is that times are canals through which actions tend to flow. Time is a useful category by which we can know and understand the operations of the material world (Bluedorn 2002). To some extent, this can be seen by measuring durations and studying schedules, to see how events play out at different hours of the day or days of the week. From this point of view, computer use claims hours, accelerates processes, or makes schedules flexible.

My work builds on another line of thought, which emphasizes the constitution of different kinds of times by people, places, habits, practices, technologies, and ideas. Treating measurement as only one use for time, and a historically recent and culturally specific one at that, I focus on the great number of discrepant and concurrent experiences of time in everyday life that arise out of local interaction. Times are performed socially. From this point of view, we who use computers also use time. Just as people measure duration by a stopwatch, plan events by a calendar, or anticipate opportunities by a clock, we use computers to do things with time.

Existing studies on time and computers have come together around the thesis that instantaneity facilitates acceleration. Robert Hassan argues that information and communication technologies are displacing the importance of clock time by offering instantaneous and simultaneous communication that also allow for a greater degree of asynchrony. Processes that would have been conducted according to the clock, post, telephone, or face to face encounter now take place online or at a distance at potentially any time, day or night. This has tended to make things faster. Society operates by a simultaneity that arises from instantaneous connections. The fact that finished products can be exchanged instantly also means that you can work all night to get something done or do your banking online on any day of the week, making local pockets of asynchronous activity (Hassan 2003; Hassan 2007).

But if the simultaneous and instantaneous define drama on the network, what is happening in the asynchronous moments before, after, and alongside connection? When I use a computer, or even a mobile device designed for networked communication, I often feel isolated, autonomous, and quite disconnected from a simultaneous present shared with the network society. The computer's ability to act instantly provides a relaxed time where I can make things go or stop, run in parallel, delay, pause, wait, or restore. In the gaps of the network, what time is this?

### **Interactive Times**

Living in a highly engineered environment where software mediates actions of all kinds, we have become accustomed to luxurious forms of time where objects are constantly available and well-behaved, where conditions last longer than just a

moment, and where our actions will not ruin the whole setup. These are not the conditions of the kitchen, motorway, or courtroom. They are quite rarefied and difficult to produce.

Interactive times are practices of time where functions of temporality dynamically respond to user input. The user pushes a button and the past becomes the present, an opportunity opens, two objects synchronize, events delay, or a rhythm changes. Some interactive times make time appear quite unfamiliar, allowing the user to browse through time (Mancini, Dix, and Levialdi 1996), edit a series of possible events before or after they have happened (Deutsch and Lampson 1967, 799), or rewind certain elements of a system at different rates or as a side effect of other actions (Blow 2008).

Interactive times provide a limited amount of control: a user pushes a button and thereby changes the operation of a time. The user is not free to redefine the terms of such times, but is able to do more than simply enforce what another has written. In video games and digital literature, programs are often only interactive enough to give the user an ability to control their own view of the whole; all paths have already been charted, user choices simply determine which fraction of the whole to display (Manovich 2001, 70–75; Aarseth 1999). The user has more control than that. On the other hand, the user is not the architect of time. It would be nice to recover an old version of a file that was never saved, to fast forward through the FBI warning on a DVD, or see every change ever made to a particular paragraph. But these require pre-

existing software systems and cannot be performed by the user on the spot simply because she is using a computer.

Interactive times trust the user to control certain conjoined aspects of temporality, disbursing the responsibility of time to each user. The usual function of this technology is to empower the user, but the consequences exceed the individual. Personal computing is not just personal, though this marketing illusion has been critical to its development. Those using interactive times produce different kinds of products, influenced by the conditions of their production. Using an interactive time changes how one goes about his or her business, and this shift has been strategic for those selling interactive times and for those buying them. Times can also reach beyond the dyadic relation of machine and operator, subjecting others to a time that is controlled by another user, who may well be a stranger or corporation actively trying to exploit you.

But, what is a user? By user I mean whoever happens to be at the controls of the machine at the moment. The user is a position relative to the computer (Galloway 2006, 127). System inputs come from what will be called the user, even if they are in fact generated by other pieces of software, by a cat walking on the keyboard, a company, a worker, a child, a series of different people, or a representative of a group. For this reason, Apple developed a tool for user testing called the monkey, a program that would simulate a user flailing around and making unexpected decisions of all kinds (Tesler and Espinosa 1997, 50). The user is part of the program (Woolgar 1991). In fact, specific, embedded actors take on the role of user, and each will make

different uses of interactive time because of it. Certain people have been encouraged to use more than others, and historical consideration can show who these users were imagined to be and, to some extent, who they actually were. But interactive times, like computers, remain open to unknown users; the position can be occupied and changed.

### **Overview**

To understand interactive times requires understanding time. This is the work of the second chapter. It begins by reviewing the status of time in social theory, restating the critique of common sense notions of time, and ultimately giving a positive account of what times are. The goal of this chapter is, first, to provide theoretical vocabulary on which to build and, second, to create some coherent integration of the wide ranging scholarly work that already exists on the topic of time. I do this through close readings of important writing on time by Barbara Adam and Emile Durkheim. I started research knowing what time on computers is *not*; here I argue that a rich vocabulary of time related words clarifies what temporality is and what time does.

In chapter three, I move from examining a single time to the relationships between multiple times. Exploring the great variety of ways times can interact, I distill two key modalities of relation: translation and influence. Translation occurs at the level of representation, though it must always be enacted materially. Influence, a relationship that includes the physical effects of temporality, is much broader. A brutal schedule that leaves one missing sleep or a relaxing time that helps one finish writing a memoir are both cases where one time influences another. Historically, the relations

amount to an unstable ecosystem in which a number of mutually reinforcing times have become dominant. We now see the second, hour, day, and year as a natural fit, based simply on the rotation of the Earth and present in our world as a dimension perpendicular to space. The reality is a palimpsest, with newer ideas written over half erased older ones in a series of minor improvements that took place over the last few centuries. The popular thesis that times have basically accelerated appears, in this context, as a recapitulation of the dominant perspective of certain time practices measuring other times and treating them according to their measure.

With these two chapters, it becomes clear both what a time is and how times take on relationships to one another. My thesis highlights new practices, brings attention to old practices that never fit into the dominant alliance, and provides a different perspective on all times, familiar, old, or new. In one sense, interactive times join an underground of alternative time practices. To some extent, these infra-practices, occurring below the radar, have begun to undermine the old alliance by becoming common ways that time is practiced. At another level, the balance rests in how we imagine time, and how we can build arguments about it. It is at this level that this project aims to intervene.

Chapter four takes a specific example of interactive time and gives it a robust historical treatment. The undo command, now a commonplace in software of all kinds, established an interactive time where what the user did could be immediately undone. The feature arose in a number of projects, gradually moved into a key laboratory, and then featured in some influential systems at the moment when personal computing

rocketed up from an obscure hobby to an expected future. This case study shows the role of the software market and industry, academic researchers, the millions who would have to learn to use computers, central institutions such as Microsoft, and the programmers who turned fantasies of time into real conditions of work and play. Such an approach connects software's operation to existing historical forces.

Chapter five considers interactive times as amalgams built from a wider body of conventions. These conventions have many names, such as error-checking codes, runtime libraries, and hard disk drives. A great many of them are also conventions of time. These conventions can be understood on their own, though their role in time always depends on the way they are appropriated into actual practices. This chapter explores three bodies of convention, those of the processor, memory, and network connections. Interactive times depend on processors to establish interactive situations and recursive operations. They also depend on memory to access other moments. Network connections take interactive time outside of its asynchronous cocoon to a situation where the preconditions of interactivity are more precarious, but still available in great abundance.

With an exploration of conventions, I signal the wide range of computer times: we do not yet know what a time is capable of. Though these conventions make some things easier and harder, their potential reaches ahead of us as well as behind. What new times will we live by in an age of shrinking devices and ubiquitous connectivity? Many conventions have already produced new kinds of time that have important implications already being discussed in terms of robotics, privacy, intellectual

property, connection, latency, and attention. The conventions that joined together to make interactive times have also made many others.

### **Note on Methods**

I have offered some introduction of what is to be discussed and why. Time practices, software, interactivity, and social forces are prominent themes. But who is actually involved in these practices, where are they located, and how did I go about studying them?

First, who are we talking about? Computer users? There is no way to know exactly who has access to interactive times, who could use them but is ignoring them, and who is using them all day long. For some technologies, such as digital video recording and Microsoft Office, a substantial percentage of an entire country may be involved.<sup>2</sup> Companies often do not release numbers to answer these questions, and even if they did we would still not be sure how many pirated versions or copycat products exist worldwide. For other technologies, such as online calendars and video editing, the install base is much smaller and we are really talking about specific kinds of people, such as white collar workers or video editors.

Who is using the software and what they are doing with it will vary with each technology. Some general answers are that office workers use computers more than construction workers, most home users do very regular things, mostly with Microsoft products, and most organizations of any size use computers extensively. These

---

<sup>2</sup> Areas with a personal computers for every two people include North America, Brazil, Australia, East Asia, and Western Europe.



distributions of use, though vague, are important to remember. For the most part, the new operations of time I'm describing have more relevance to the elite, to office work, to art, science, and entertainment than to other sites. Though computer companies have long claimed that their products can be useful to anyone, this clearly makes more sense for college graduates and managers than for hermits and mendicant spiritualists.

Second, where are practices of interactive time taking place? Primarily, between the user and the machine. Software exists in a "space that is constantly in-between, a mass-produced series of instructions that lie in the interstices of everyday life, pocket dictators that are constantly expressing themselves" (Thrift and French 2002, 311). Once, computers took up very large spaces, which were dedicated exclusively to them and their operation. As they became smaller, they followed a familiar pattern, carving out many small crevices from ordinary life, and gradually taking these spaces as territories of their own. This general pattern of machine space can be found in the historical erosion of urban spaces as places for people into places for cars (Horvath 1974). Now these spaces are all around us, on buses and airplanes, cafes and dormitories. The reach of a time may exceed the machine space in which it arises, exerting influence at other sites and in less direct ways. As with the question of who, it is hard to say exactly where, but the usual sites of computer use correlate highly with software saturated societies and less strongly with other places.

Third, how have I culled empirical material and made it function as evidence? The first two chapters are primarily theoretical, working from examples and using

close readings and synthetic approaches to build an argument from secondary literature, personal experience, and the studies of others.

The fourth chapter, a case study on the undo command, uses several varieties of historical evidence. I visited archives at the Computer History Museum, Brown University, and the Charles Babbage Institute at the University of Minnesota. There I found evidence on the projects on which I focus, on parallel projects, on various perspectives on computing during the period, and on projects that might have included an undo command but did not. I interviewed key figures from this history, in person, by Skype, and by email. Old copies of *Time Magazine*, *Byte Magazine*, and *InfoWorld* were especially helpful in understanding public perceptions of computing, the marketplace, design norms, and user practices.

The fifth chapter, which addresses conventions of time in computers more generally, required studying media as they exist today. This research was in a way analogous to my work on undo, except the emphasis was technical instead of historical, which put me in league with a huge number of others who wanted to learn how computers work. For this, I read technical papers and presentations, consulted textbooks, talked to experts, and learned from forums like anyone working a tech job would do on a regular basis.

My goal is to build a positive, synthetic work with clear claims that can direct action and change our impression of the world around us. Critical reasoning thus takes a back seat in the text, though it plays a formative role at certain key junctures (such as developing my theory of time and comparing designers' stated goals for an undo

command). This text aims to hone perceptions (particularly of temporality) and to expand readers' ability to imagine connections between what they know and what they can perceive. The value of this work depends entirely on the power of its argument and pleasure of reading, rather than on facts about its author, and this is a reason to keep it simple, clear, organized, and, when possible, interesting.

One of the key techniques used throughout is to multiply concepts and then summarize them in a stultifying abstraction. Often, I use limited lists to suggest some members of a class without offering a complete catalog or a set of structural terms by which the total space of the category could be known in advance. Such lists are partial and not the final word. They are products of a theoretical exploration that learns nuance from phenomena, decides between interpretations, and produces stylized readings that guide further analysis. The categorical abstractions provide useful reference points that help in the navigation of empirical complexity, without claiming to produce representations interchangeable with the material they summarize. Such an axiomatic approach runs the risk of a contradiction between potential members and the overall category. I hope that this is a productive jeopardy, allowing us to grasp ideas without squeezing the life out of them by treating them as already captured and fully known.

This technique reflects the quiet influence of Deleuze at work throughout. In the text, I explain and defend concepts without appealing to authority or relying on the rest of Deleuze's expansive oeuvre. The multiplication of concepts mismatched with summary abstractions restates the relation of the whole and the parts (Deleuze and

Guattari 1983, 42–50). I make particular use of Deleuze’s ideas on the relation to empirical material as a creative source of concepts (Deleuze 1994, xx; Rose 1999, 12–13), the distinction between the virtual and actual (Massumi 1992, 34–35; Deleuze 1988, 42–43; Terranova 2001, 107–109), the power of mutual presupposition (Massumi 1992, 13), his landmark analysis of Bergson (Deleuze 1988), discussion of inventions’ relation to social diagrams (Deleuze 1999, 34), and vision of societies of control (Deleuze 1992). My focus on embodied enactment of time departs from Deleuze’s own Bergsonian concept of time, and the style of interpretive social science strays from Deleuze in key ways. Still, we are in agreement that philosophy is nothing without the empirical.

### References

- Aarseth, Espen. 1999. “Aporia and Epiphany in Doom and The Speaking Clock: The Temporality of Ergodic Art.” In *Cyberspace Textuality: Computer Technology and Literary Theory*, edited by Marie-Laure Ryan. Bloomington: Indiana University Press.
- Adam, Barbara. 1990. *Time and Social Theory*. Philadelphia: Temple University Press.
- Baudrillard, Jean. 1981. *Simulacra and Simulation*. Ann Arbor: University of Michigan Press.
- Blow, Jonathan. 2008. *Braid*. Xbox Live Arcade. Number None Inc.
- Bluedorn, Allen. 2002. *The Human Organization of Time: Temporal Realities and Experience*. Stanford, CA: Stanford Business Books.
- Deleuze, Gilles. 1988. *Bergsonism*. New York: Zone Books.
- . 1992. “Postscript on the Societies of Control.” *October* 59: 3–7. doi:10.2307/778828.
- . 1994. *Difference and Repetition*. New York: Columbia University Press.

- . 1999. *Foucault*. New York: Continuum.
- Deleuze, Gilles, and Felix Guattari. 1983. *Anti-Oedipus: Capitalism and Schizophrenia*. Translated by Robert Hurley, Mark Seem, and Helen R. Lane. Minneapolis: University of Minnesota Press.
- Deutsch, L. Peter, and Butler W. Lampson. 1967. “An Online Editor.” *Communications of the ACM* 10 (December 1): 793–799. doi:10.1145/363848.363863.
- Fuller, Matthew. 2003. *Behind the Blip: Essays on the Culture of Software*. Brooklyn NY USA: Autonomedia.
- . 2008. *Software Studies: A Lexicon*. Cambridge Mass.: The MIT Press.
- Galloway, Alexander R. 2006. *Gaming: Essays On Algorithmic Culture*. 1st ed. Minneapolis: Minnesota.
- Glennie, Paul, and Nigel Thrift. 2009. *Shaping the Day: A History of Timekeeping in England and Wales 1300-1800*. Oxford: Oxford University Press.
- Goriunova, Olga. 2011. “Empty Internet.” *Computational Culture* (November). <http://computationalculture.net/review/empty-internet>.
- Hassan, Robert. 2003. *The Chronoscopic Society: Globalization, Time, and Knowledge in the Network Economy*. New York: P. Lang.
- . 2007. *24/7: Time and Temporality in the Network Society*. Stanford, CA: Stanford Business Books.
- Horvath, Ronald J. 1974. “Machine Space.” *Geographical Review* 64 (2) (April 1): 167–188. doi:10.2307/213809.
- Kitchin, Rob, and Martin Dodge. 2011. *Code/space: Software and Everyday Life*. Cambridge, Mass.: MIT Press.
- Mancini, Roberta, Alan Dix, and Stefano Levialdi. 1996. “Reflections on Undo”. Rapport Technique RR9611. University of Huddersfield. <http://www.comp.lancs.ac.uk/~dixa/papers/undo-techrep-96/tech9611.pdf>.
- Manovich, Lev. 2001. *The Language of New Media*. 1st MIT Press Pbk. Ed. The MIT Press.
- . 2008. *Software Takes Command*. <http://lab.softwarestudies.com/2008/11/softbook.html>.

- Massumi, Brian. 1992. *A User's Guide to Capitalism and Schizophrenia: Deviations from Deleuze and Guattari*. A Swerve ed. Cambridge Mass.: MIT Press.
- Nowotny, Helga. 1992. "Time and Social Theory: Towards a Social Theory of Time." *Time Society* 1 (3) (September 1): 421–454.  
doi:10.1177/0961463X92001003006.
- . 1994. *Time: The Modern and Postmodern Experience*. Cambridge UK: Polity Press.
- Rose, Nikolas. 1999. *Powers of Freedom: Reframing Political Thought*. Cambridge University Press.
- Terranova, Tiziana. 2001. "Demonstrating the Global." In *Virtual Globalization: Virtual Spaces Tourist Spaces*, edited by David Holmes, 95–113. London: Routledge.
- Tesler, Larry, and Chris Espinosa. 1997. "Origins of the Apple Human Interface". Lecture October 28, Computer History Museum, Mountain View, CA.  
[http://www.computerhistory.org/events/lectures/appleint\\_10281997/appleint\\_xscript.shtml](http://www.computerhistory.org/events/lectures/appleint_10281997/appleint_xscript.shtml).
- Thrift, Nigel, and Shaun French. 2002. "The Automatic Production of Space." *Transactions of the Institute of British Geographers* 27 (3) (September 1): 309–335. doi:10.1111/1475-5661.00057.
- Wachowski, Andy, and Lana Wachowski. 1999. *The Matrix*. Action, Adventure, Sci-Fi. Warner Bros.
- Woolgar, Steve. 1991. "Configuring the User: The Case of Usability Trials." In *A Sociology of Monsters: Essays on Power, Technology, and Domination*, edited by John Law. Sociological Review Monograph, 38. London; New York: Routledge.

## II. Time Theory

A clock advances regularly by drawing on forces that are themselves regular. It organizes the inevitable arrival of the future into a sequence. But it does not invent inevitability, futurity, or sequence. The clock's time mediates these operations of temporality, but does not wholly encompass them. With clocks, the challenges of scheduling events, anticipating potentials, and synchronizing rhythms become easier, or at least more calculable. Yet the clock cannot do this alone. It must be built, maintained, referenced, and trusted. Through a whole constellation of actions, human and non-human, some of the various ways temporality functions can be put to use. Despite this complex choreography of contingency, we have come to accept a simpler definition of time: what the clock measures.

This chapter argues that a *time* is a social practice that translates *temporality* into meaningful codes and organizes temporality's material influence. There are many expressions of temporality and there are many times that strengthen, harness, and direct them. This model moves beyond criticizing a traditional view of time as linear, singular, constant, and universal. That work has already been done. The goal now is to build positive accounts of many times in such a way that they can be understood together. This involves explaining terms, identifying levels of organization, and responding to possible counter arguments.

Time is worth thinking about because it matters in many specific cases. However, what it does in each situation varies tremendously. This variety has stunted

growth around the topic: while many people are writing about time, there seems to be no way to fit together the different things they are saying. What goes on in intergenerational memories (Giesen 2004) seems to have no relationship to the use of historical time in search engines (Hellsten, Leydesdorff, and Wouters 2006). In the sociological literature on time, Werner Bergmann notes that “one finds complaints everywhere about the neglect and marginality of the time problem in sociology” (Bergmann 1992, 82). Yet he reviews work on time-reckoning, historical consciousness, utopianism, time-tables, various cases of waiting, careers, developmentalism, and time in organizations.

Such diverse research on time should produce a network effect; each additional study should enrich scholarship on time as a whole. At present, however, studies are difficult to compare and so compared rarely. There are now a few major theses that could unite studies of time. These claims are: that time is accelerating, that a new kind of globally networked time is ascendant, and that there are many different times that are densely interwoven. Of these, the first simplifies the diversity of times into a single dynamic, the second describes only some practices of time, and the third makes all studies parallel but isolated. If the parallels between studies could be related in a systematic way, the third thesis might turn time into a productive term in the more general explanation of social change with which social science is tasked.

Barbara Adam’s *Timewatch* gathered and joined together crucial insights from lived knowledge of time into a general theoretical model. My work extends this project. In *Timewatch*, Adam turns away from existing work in the social sciences that



approaches time from one particular angle (such as future orientation, use of waiting, or systems of time measurement). For Adam, times have many aspects, coexist, and act in many, complex ways. For *Timewatch*, she interviewed ordinary people about time, hoping to discover not just the personal significance of the term for those interviewed, but the actual function of time in the lives of all kinds of people—something they should understand best themselves. This project revealed a huge variety of aspects of time: time when, time as a medium of exchange, time periods, time in language, time frames, and the growth and death implicit in process. Such aspects of time, she argued, were mutually implicated, inseparable from one another, and deeply imbricated with the dominant time of the clock (Adam 1995).

Adam's project points the way to a diversity of time but does not provide theoretical language by which to recognize and relate together different aspects of time. In *Timewatch*, Adam argues repeatedly against categorization, abstraction, sharp distinctions, or isolating any aspect of time. We experience times in combination, as real to varying degrees in different contexts, and in multiple registers that do not exclude one another. Abstraction, she argued, would falsify that experience. Adam is not anti-theory, but this project prioritized everyday experience in order to open space for an account of times as multiple and mutually implicated. Here, I extend her analysis by offering terms in which different times can be apprehended in their difference, so that we can better understand and discuss their combination and mixture in reality as it is lived. Though everyday experience positions us between many times, we relate to these times in different ways and understand each in its peculiarities. That

we think of a timecard in one way and our mortality in another does not deny connection between the two or the fact that they both operate on the same people; it is by knowing each that we make sense of the experience of multiple times. *Timewatch* refined intuitions of time into a rich vocabulary. Here, I attempt to turn that vocabulary into a system for thinking about times as complicated, varied, and interrelated social practices.

Every different *time* organizes a set of relationships of *temporality*. Imagine for a moment that temporality is essentially a matter of past, present, and future. Then what a time does is organize relations between these: on the calendar, the present is one box, the past all the ones before, and the future is all the boxes that lie ahead. However, there is more to *temporality* than simply the categories of past, present, and future. Synchronization, memory, repetition, delay, and opportunity, for example, cannot be reduced to the modes of past, present, or future, yet are critical aspects of what calendars actually do. Thus, a time orders relations between not just three modes, but a larger number of ways temporality functions.

### **Temporality Functions**

The idea that time has a social basis is often traced back to some cursory comments made by Emile Durkheim in *The Elementary Forms of Religious Life*, published in 1912. Durkheim begins his argument about time with a highly abridged thought experiment: try to imagine time without any of the usual measures such as hours, days, or years (Durkheim 1995, 9–10). The ideal reader will take a moment to imagine the past or future without reference to known units of time, falter, struggle

some more, and discover that time is only conceivable by telling between one moment from another, be it by days or some other marker. One way to differentiate moments is by our changing mood or mental state. Durkheim explains that this is sufficient for thinking of our own past, but is not enough as a definition of time because time is an “abstract and impersonal framework that encompasses not only our individual existence but that of humanity” (10). Time is a “continuous canvas” where all events can be located in relation to reference points that are generated in social life (11). In this argument, the word “time” refers to ways that different moments are represented socially. The calendar, he concludes, “expresses the rhythm of collective activity while ensuring its regularity” (10). It reflects and modifies cycles of social life in a relation that you might call feedback. The use of a calendar system changes how events are located, remembered, planned for, or enacted.

What does this mean for other technologies and other aspects of time? In the two pages that Durkheim dedicates to time, we do not learn the role of stopwatches, harvest rituals, or age set systems. If calendars express and ensure the rhythm of collective activity, can we say that forecasts and prophecies express and ensure the prospective thought and planning of collective activity?

Durkheim asks readers to acknowledge the role of social conventions in their thinking about time. This is a key theme for which that book is remembered: human thought is socially produced and not simply given by the nature of the mind. This claim responds to the Kantian tradition, and Durkheim’s comments on time can easily be read as responses to the philosophical writings of Henri Bergson in the previous

decades or the development of scientific psychology a decade before that. Against all of these attempts to pin down the mental experience of time, Durkheim argues that the categories of understanding are taken from social life.

By understanding the claims of his interlocutors, it becomes easier to understand the limits of Durkheim's project. For him, there is fundamentally only one time, though different groups organize it and respond to it a bit differently (Miller 2000). The laws of human thought in Kant, the basic metaphysical reality of duration for Bergson, and the measurable nature of human psychology all agree that time is ultimately of one kind. Durkheim honors this tradition by claiming that time's singularity is enforced socially. If a society did not respect such a foundational part of human thought, then "[a]ll consensus among minds, and thus all common life, would become impossible" (Durkheim 1995, 16).

Though helpful in establishing a clear case for the role of social life in something that can easily seem essentially noumenal, Durkheim's concept could use some flexibility and extensibility. The calendar, though existing in great variety, does not offer a very deep insight into the range of ways those living in a society practice time. Durkheim suggests that calendars are useful for locating the events of any member of a society. This is somewhat hyperbolic; for many events, we do not know exactly when the thing did (or will) occur. When did you first meet your best friend? When will you next go to the hardware store? At the same time, this use of the calendar does not capture well the uses to which people put clocks, personal planners,

alarms, or time-stamped computer files. There is more to time than Durkheim's continuous canvas and rhythms of collective activity.

Why do humans produce clocks or schedule meetings? There is not one answer to this question, and that fact is critical. Against those narratives of domination that see the clock as a tyrannical force in recent human history (Hammer 2011; Frank 2011; L. Mumford 1964), Paul Glennie and Nigel Thrift argue that the clock's power "comes from the fact that it can be so many things to so many people. Its power comes from the *difference* implicit in the multiple possibilities it generates" (Glennie and Thrift 2009, 97). One can use the clock to pace a process, to plan an event, to remember hours past, to trigger an alarm, to synchronize movements, to measure duration, to delay an action, or to pass the time. With the clock, we can make use of many aspects of temporality. However, there are aspects of temporality for which the clock is no use at all; the inevitable, the ultimate, the ancient, the backwards, and the hip are aspects of temporality on which clocks have no purchase.

Before anything as organized as a time develops, there are primordial influences I will refer to as ways *temporality* functions. We notice temporality because it does things and is the means by which things happen. It can act, mediate, or appear. Synchronization makes two processes begin at once that may meet later; pace mediates between forces of acceleration and deceleration; an event appears and ends. Temporality can work in these ways, or not. Temporality works by a number of distinctive functions, such as inevitability, anticipation, events, opportunity, speed, ephemerality, frequency, timed access, scheduling, and change.

These many functions cannot be reduced to a single term, despite many efforts to do just this. Master terms proposed to encompass all the heterogeneous functions of temporality include: paced repetition (Brann 1999), absolute flux (Newton 1962), ordered sequence (Gell 1992), or duration itself (Bergson 1896). Each unification attempt subsumes very different aspects of temporality to its own universal logic. Therefore, each reduction works as an exclusion: if temporal *passage* is the only real form of temporality, then eternity, event, and causation are simply redefined as non-temporal. The most extreme form of this procedure occurs in those accounts that deny time exists at all (presentism), thereby asking that we speak of the regular movement of pendulums, memory of the past, and hours shops remain open without recourse to time or temporal vocabulary (for example, Radovan 2011). However, it is exactly *because* phenomena such as rhythm, pace, change, memory, and opportunity exist that time is worth talking about in the first place.

Does giving up on time as a single essence mean violating Occam's razor? The logic behind what is called Occam's razor comes down to an aesthetic goal of elegance through simplicity and a working preference for simpler theories with greater explanatory power. Imagining more factors, such as luminiferous ether or angels, increases how much has to be assumed and can only be justified by providing a proportionate increase in how much the theory explains. In the case of time, the traditional universalizing theories can each explain some very useful things, but only at the cost of negating each other by insisting that time has a single, fixed essence. The real problem with Occam's razor as a defense of any theory of time is that the

fundamental concepts are all extremely murky and difficult to define in other words. Absolute flux, for example, even if renamed whimsically as “the swirling play of accidents” (Harman 2009, 217) or “an irreducible stream of unrepeatable events” (Glennie and Thrift 2009, 66) is still a very difficult to define qualitative happening. So the choice is not really between theories with a clear definition of time but limited explanatory power versus a theory based on the weirdness of temporality with greater reach. Really, all theories of time depend on an agreement with the audience that there is some kind of reality to time and we should discuss it, even though few definitions are better than poetic.

It is by rejecting the search for a stable foundation (a single definitional reference for what time is) that we can see the variety of times present in embodied activity and wonder what creative spark nominates them for such consideration. What makes a thing feel like it might have something to do with time? The many functions of temporality do not emerge from a single underlying nature of time, but from the variety of occasions which we cannot help but sense as temporal.

Temporality *does* things. Though it is not wholly material, temporality has an influence in the world. More precisely, it has many influences on the world. Temporality happens in every moment and we are quick to notice it because we are *susceptible to it*. The sun rises, an eye blinks, an event is scheduled, a room reminds one of the past, events happen in sequence, a library cuts back its hours. To experience the influence of temporality is not uniquely human; wood ages and cracks, plants flower and die, sedimentary layers accumulate and harden. Temporality is “out there”

in the world precisely because it is “in here” for so many objects that are themselves out there in the world.<sup>1</sup> Things feel temporality and we detect temporality’s influence, through these reactions.

To recognize the pervasive role of temporality requires an ability to sense temporality and a *willingness* to regard phenomena in temporal terms. Boredom is psychological insofar as it is a response to disappointed vigilance, but it is also temporal insofar as it indexes a specific way temporality shapes human experience (Brodsky 1995, 109; Heidegger 1949, 364). Representing other peoples as backward invokes ethnocentrism in politically charged rhetoric; yet temporality operates alongside ethnicity, race, rhetoric, and politics in backwardsness and cannot be reduced to them (Fabian 1983; Chakrabarty 1997; Butler 2008).

One step in cultivating a sensitivity to temporality is to notice its role *in writing*. As the reader becomes aware of the frequent, and inevitable, use of temporal vocabulary in this text, it should become more apparent the urgency of recognizing how functions of temporality permeate the domain of language and thought, and implicitly what they describe. The previous sentence invokes temporality with the words “becomes,” “frequent,” “inevitable,” and “urgency.” We are all already invoking temporal functions nearly every time we communicate in language.

---

<sup>1</sup> Steven Shaviro presents Alfred North Whitehead’s metaphysics in readable terms: “every event is the prehension of other events,” where prehension means “the act by which one actual occasion takes up and respond to another” (Shaviro 2009, 29,28). An object touches others by being apprehended by them; the object is experienced in a limited way and this experience is the processual occurrence of reality. Or, as Karen Barad puts it, using Niels Bohr instead of Whitehead, “phenomena are the ontological inseparability/entanglement of intra-acting “agencies”” (Barad 2007, 139).



The many ways temporality functions can be better understood with the help of theory. Using words, thoughts, and ideas, theory can stultify the influence and occurrence of temporality into patterns, types, and categories. This helps us recognize and discuss it. This stultification does not stop time or represent temporality as acting in singular and controlled ways, but it does provide a technique by which we can discern, describe, navigate, and respond to temporality as it functions. To notice temporality with theory is to have (and develop) a special sensitivity to temporality. This allows us to abstract from scenarios (the complexity of temporal experience) into functions.<sup>2</sup> The theorization of temporality processes the occurrence of temporality into an artifice that can translate temporal influence into thought and argument.

Here I will lay out four of the many temporal functions to give a sense of this technique. Each can be defined by its specific style, which is not quite reducible to combinations of other functions.

### **Duration**

In Henri Bergson's notion, duration is the ongoing contraction of the past by which the present becomes the future. In his earlier work, objects give rise to images that are perceived with the aid of memory; duration is how perception is registered by the subject (Bergson 1896/1950, 170–177). Later, Bergson clarifies that duration is a qualitative multiplicity, so moments can be told apart but cannot be separated or

---

<sup>2</sup> Abstraction is not strictly a human faculty. The artifice of theory that stultifies temporality into analytic terms is not reducible to the human, as it depends on writing, education, and practices of rigor that happen through larger assemblages of which humans are only parts. This handling of the world exceeds the human.

named exactly (Bergson 1910/2002, 75–87). Though we can differentiate moments from each other, their effect on the mind is cumulative (100-106). Duration cannot be measured by the clock, because the clock works by quantitative multiplicity which only registers states and never transformation itself. Duration has both a material substrate and a basis in perception with memory. Passage, occurrence, and novelty are closely related to duration. Although Bergson's later work establishes duration as an objectively real site for the occurrence of all events (and not just a property of human mental experience), the truth of duration is best accessed by intuition, by "carefully examining our consciousness" (105). Expressing duration in language quickly solidifies sensations into names that mediate our experience of actuality (130-137).

### **Contemporary**

Although we often make statements about the contemporary world, most of them are quite exclusive. If the contemporary city is secular and lacks public space, if contemporary filmmaking is going digital, or contemporary science is moving at a breakneck speed, what about those many cases where these generalizations do not apply? Johannes Fabian presented this notion as a critique of ethnography: by interpreting those studied as timeless and external to the contemporary world of which the researcher is part, ethnographers treat them as if they were locked in backwards times. Fabian proposed, as an alternative, that ethnography should build from the experienced co-presence of fieldwork to give informants full credit as members of the contemporary world (Fabian 1983). Dipesh Chakrabarty argues that the contemporary is necessarily exclusive, and this is why it is useful. Legalizing homosexual acts means

joining the contemporary world in a way that imprisoning those accused of homosexuality does not. Even if we imagine that there are many versions of the contemporary, each going in their own direction (and few think of the word in this way), one can no longer claim that one activity or trend represents the latest or most progressive position over another (Chakrabarty 1997, 50). By connecting some things as present, the contemporary makes other things absent. The contemporary depends on temporality's ability to synchronize, make present, make out-of-date, and make some things more dynamic than others.

### **Access**

Though the word access may describe other things, access or availability is an important way that temporality functions. A shop may be open 24/7, bankers hours, seasonally, for a few years but not anymore, or by chance. Access, as a temporal phenomenon, is a property of *states* of time, usually of the clock, and therefore not reducible to the idea of duration. For the coordination of access, the future and past can be organized into schedules, such as calendars or eras. Access is a specific possibility of encounter contextualized by a schedule, though not ultimately dependent on one. In some cases, access is a kind of clairvoyance of the situation and state of other things. We can call ahead to get a shop's hours, or get a feeling that something will be accessible at this moment that is not at others. Access is a precondition to possible interaction; it is an enabling condition of potential. Access is a means by which schedules become meaningful, synchronization possible, and the contemporary useful. It is an aspect of temporality we use regularly to locate possibilities in

conjunction with time practices that order occurrence into seasons, months, or happenings.

### **Inevitability**

Inevitability is a specter of a future that cannot be denied. This function of temporality is altogether different from access, the contemporary, and duration. Access sets potentials of interaction, the contemporary joins them together, and duration extends the movement of becoming. The inevitable operates in the present as an assurance about the future. What is inevitable is something that *will come to pass*. No matter what. It has the certainty of the past but will only occur in the future. Because it will happen, it may be regarded as a fact of the present. However, just because we know it will happen doesn't mean we know *how* it will happen. Inevitability is not just a figure of speech. We rely on it in very concrete ways, as when we assume we will return home eventually or die someday, even though how this will happen is unclear. Sometimes, as with anything else, the term can be misapplied: capitalism is not inevitable. But the death of the sun, rise and fall of the tide, and success of a new pop star do function as inevitabilities. Though these can all be described without using temporal vocabulary, the implication would remain that their future can already be known in the present, and therefore acts in advance of itself.

These are four examples. More could be given. It is hard to think of them in isolation because they do not act in isolation. As Adam correctly points out, we live with the complexity of everyday times. We cannot use abstractions to deny that

complexity. However, we can focus on one specific aspect at a time, making other parts blurry for a moment (Adam 1995, 159). By focusing on a specific aspect of temporality, we can better understand how it functions and how it may relate to other aspects of temporality in a given situation.

I have argued that temporality acts in the world, creating changes and setting circumstances in which events must unfold. But doesn't this put temporality in the role of cause? Some might worry that the argument is circular. First, temporality precedes cause because it is its enabling condition: event and sequence are functions of temporality that organize causality, in the usual account of causation. Temporality is a precondition to causation, but does not usurp the power of other causal forces. Second, there are *direct* causes as well as *contributing* causes. If someone moves to a city because it provides a more contemporary feel, or because she remembers the city as a nice place to live, the contemporary and remembrance are contributing causes to the move. Third, the usual model of causality is particularly weak in its representation of complex temporal functions. The usual model produces powers of prediction and control precisely by *denying the role of* any form of temporality other than events acting on each other in sequence. This image of cause discourages us from noticing the influence of other aspects of temporality such as anticipation, memory, and backwardsness. If one *anticipates* that searching the web will provide answer a question quickly, this attracts the person to the action of searching. The easy search *drew* the person in to use it. But, phrased in the strict causal model, a *psychological experience*, not a distinct modality of temporality, caused real action. In such a model,

one is compelled to use a progressive grammatical structure where what is earlier always results in what is later, even when we know very well that what happened later acted in advance of its realization.

Luckily, we do not need to use such a stifling model to retain a powerful concept of causality. Causality may be understood as powers acting simultaneously on one another with varying success (S. Mumford and Anjum 2011). There are more ways to act than to trigger or mechanically produce a following event because there are more scales of action than the dyad of stimulus and response and because there are other ways temporality functions than the sequence relating cause and effect.

Temporality does many things, yet time is orderly. Temporality functions in many ways, is experienced by situated actors, and can be abstracted into theoretical terms. Temporality does not just mean an inevitable and constant flow of the future into the past. It is less a river than “an uncertainty open to mobile and surging forces coming from all directions: an ocean in a storm” (Moran 2010). Yet that which we call “a time” is orderly, with set patterns for how temporality occurs. Tuesday always follows Monday. Though there remain rogue functions of temporality, most fit together in a way we regard as seamless. This is the work of social practices of time, and of their naturalization.

### **Time as Social Practice**

Time is an enacted, material, social practice that organizes the functions of temporality. Though social theory of time has long recognized the artificiality of

times, the status of these constructs remains unclear. What is clock time? How does it relate to memory, the time of a religious calendar, or the temporality we experience? Here, I propose that anything that may be called a *time* is also a social practice.

By social, I do not mean something reducible to human will or something made up of the totality of connections between objects of any kind whatever. Social connections occur between actors whether they are human or not. They are a matter of associations that take very different forms, rather than expressions of one thing that is fundamentally social (Latour 2002; Harman 2009). This does not mean that relations, whether named networks or social constructs, are of a singular type that all join together in something so uniform it can always be described as a relation (Bogost 2009; Moran 2013, 53). “Social” construction happens by many modalities, each better understood by discourses and sensitivities appropriate to it. A social practice can happen entirely within a machine; if it did, a technical approach would be sufficient. Usually it depends on, anticipates, or otherwise encounters things beyond the machine, which are better described in terms of culture, practices, economics, time, or biology. Humans deserve special moral consideration, but they are hardly the only actors responsible for social processes. A gate can enact a time as well as a guard.

A time is a bundling of temporal functions that rationalizes them and it is a practice that coordinates them. I take this idea from John Koller’s analysis of Buddhist views of time. Koller defines time as “a theoretical term to cover the temporal terms and relations as these are understood conceptually” (1974, 204). *Time represents temporal terms as connected*. It is not the source of that which we recognize as

temporal, but the sense made of it. It is in practices that we experience time, and thus it is practices that make times (Shove, Pantzar, and Watson 2012, 129). A time gives temporal terms specific connections, rather than simply fusing them all together or mixing them up with each other. What does this look like?

In E.P. Thompson's 1967 article on time in factory work, he argued that a new time practice came to dominate the many forms of time that had flourished in Europe previously. This new time depended on clocks with hands, gates that locked out workers who came late in the morning, and a manufacturing process requiring synchronized labor. It organized a number of functions of temporality in specific ways: it formed activity into measured durations, purged of idleness, that had beginnings and endings positioned within the hours of the day (Thompson 1967). This practice, which is also a time, did not rationalize all functions of temporality; nostalgia, inevitability, and rituals that had made work irregular before were not part of its order. Workers could no longer extend their weekend to honor the fictitious Saint Monday. But the factory's time did take a number of specific ways temporality can act in the world (duration, idleness, event, schedule, and passage) and put them together so they did something useful.

If a time arranges temporality at a conceptual level, how does it actually go into force? For Thompson's time practice, the answer is clear: "by the division of labour; the supervision of labour; fines; bells and clocks; money incentives; preachings and schooling; the suppression of fairs and sports" (Thompson 1967, 90). Thompson drew on studies of the first generation of factory workers that showed



owners tended to use the stick often and carrot almost never (McKendrick 1961; Pollard 1963). The time of the factory, then, seems entirely dependent on *force* to move from an idea in the factory owner's mind to an actual practice workers would more or less obey.

Discipline, however, represents only one means by which times are enacted. Not all times are policies enforced by violence, although those which produce a feeling of domination do invite analysis. Accounts narrowly grounded "in either technology or in social disciplining produce arguments in which any originary role for other spheres, from consumption to everyday practices, is by definition impossible" (Glennie and Thrift 2005, 193). This has made scholarship acutely sensitive to the modern time formation whose expression is primarily in discipline. But these are not the only places where temporality matters, or even where it matters most. Research on historical time (Koselleck 1985), time in religion (Eliade 1959; Rayment-Pickard 2004; Hubert 1999), time in video games (Atkins 2007; Nitsche 2007), and time for wood boat enthusiasts (Jalas 2006) all evidence the significance of time beside disciplinary relations of domination. Times do not need henchmen to go into effect; they just need to be performed.

Thompson provides an example of time enforced as a policy. Yet social practices of time can take many other forms. Involuntary memories "break into the present in sudden, unexpected ways" but "do not last long" (Hoy 2009, 192). Trauma structures repetition in experience by having first been forgotten and then experienced as both forgotten and inescapable (Caruth 1996, 17). Cottage time (the slow and calm

situation of time spent in a summer cottage) happens without strict schedules, synchronization, or regularity. But, to come into existence this time requires one provide specially for it, preparing for a vacation and going to a cottage (Eriksen 2001, 157–158).

Practices of time in software affect their users, how work is done, and what is produced by providing a working environment. In practice that are interactive, functions of temporality relate in a flexible way: users can initiate reversal, create stillness, configure flow, or define events. The time goes into effect by empowering users, not by techniques of discipline or an arms race of acceleration. They open and close documents at any time, view according to terms available to them, switch between several simultaneous activities, and retrieve or consult past states. The user can make mistakes and correct them and this confidence lets her take more chances and move more quickly. This is common practice on the computer. Documents remain open to future changes, even if sent off or transferred to hard copy, and a project may be borrowed from or repurposed at a later date. That which has been produced by software bears the mark of its embodied time practices. Any product can be expected to have been edited repeatedly. In many cases, the editor and end user can manipulate the pace of time in viewing these products, rewinding songs or playing video at slow motion. Because software-based products remain open to corrections and alterations, it becomes tempting to put things online that might have once been printed on paper (think of the phone book) or done in person (online dating). These practices involve access, pastness, reversibility, memory, repetition, pace, and very long windows of

opportunity. Each one links together temporal functions in its own way, producing one of the many practices of times by which many work and play today.

In some of these times, processes can be reversed and actions undone. From a certain point of view, this will sound quite impossible. But what does it mean to turn back time, once flow, pastness, and futurity appear as subtending forces dealt with in different ways by different times? “The structure of history, the uninterrupted forward movement of clocks, the procession of days, seasons, and years, and simple common sense tell us that time is irreversible and moves forward at a steady rate” (Kern 1983, 29). Yet, how much of our own time is spent fighting to restore order, to reverse the sorry events of our existence, and to undo our own mistakes? It is easy to see a movie played backwards as action running in reverse. It certainly seems to be a way that temporality functions.

In what sense can actions be undone? Wood burnt to ashes, for example, cannot be turned back into wood. But many mistakes can be corrected and many things that change one way eventually change back. Although, we may imagine, the universe in its terrible size and infinite detail has an endless sensitivity to every action, most situations are simpler. Most sensation is ignored and most differences do not make a difference.

If I were to turn on a light, then several conditions would change: the light would be on, I would have exerted a bit of energy, the bulb would warm, a second would have passed during which other things happened, and I might leave a finger print on the switch. If I then flicked the switch again, turning off the light, how would

conditions change? I would burn more energy, moving my finger again, and take a another second to complete the act. The finger print would still be there and the light would be a tad warmer for a moment. The light, however, would be off just as before. To a detective, the fingerprint and, if they were quick to arrive at the scene, temperature of the bulb would show what had happened and the evidence would show that an action taken cannot be retracted. (Although, even then, it would be hard to know how many times the light had turned on and off.) To someone watching the room from a window, the light would have switched on then off again, but the energy used, fingerprint, and heat of the light would not be part of their experience of the events. To someone who entered the room after the light was switched twice, the light would simply be off and there would be nothing more to it. For myself, I would have turned on the light and then undone this action by turning the light off again.

Only some processes can be reversed and only in some respects. Fire, for example, is, for most intents and purposes, impossible to reverse. Turning on a light, on the other hand, is easy to reverse. This reversal does not necessarily reverse each exact movement composing the initial action, but it does the trick. We can often fix mistakes, reverse policies, and put gas back in the car after taking it for a spin.

What is happening here is that *the way an event is apprehended by the world around it* can be altered, even though the original event cannot.<sup>3</sup> Irreversible processes

---

<sup>3</sup> If the relevant prehensions of occurrence can be identified and altered, the present form of a past event can be changed, erased, or reversed. In the world outside of computers, however, witnesses very often get away or are not properly recognized, making it hard to turn back time.

cannot be reversed, but, in some cases, their effects can be. The light can be turned off. The fact that the switch has moved cannot be undone. But this fact may be of weak constitution; if known by only one person it will likely soon be forgotten.

In computer systems, the user experiences events in an extremely limited way, having no insight into the tiny and precise electric and mechanical action that makes the computer act the way it does. If the outcome of a command is that a video plays backwards, a word written now leaves the page, or an older version of a file replaces a newer one, then, for the user (and for the relevant parts of the programs being used), actions have been undone and a process can be reversed. The user sees the event happen and unhappen; the practice of time does not undo her experience. In the finished product, however, that which was done and then undone disappears completely. The power to undo actions and reverse the flow of events is an essential part of the times computers offer us today.

Times can be enacted in a variety of ways. Though Koller's definition of times as representations of temporal terms gestures toward the very wide range of ways particular times can coordinate aspects of temporality, his argument comes out of a Buddhist tradition wherein theoretical terms (such as time) *must* be regarded as mental constructs and therefore unreal. This precept gives him license to play a bit more freely with the interlinking concepts making up time, but his conclusion renounces what is most useful in his argument: times are made up of many temporal terms. His argument, therefore, does not provide an explanation of the social organization of temporality in material practices.

Thomas Luckmann instructively connects mental constructs of time with their social reality using the categories of the virtual and actual. On the one hand, times are mental and immaterial. They exist virtually as a social stock of knowledge, ready to be drawn on in ways that exceed their material presence. On the other hand, they subsist in transmissions or materializations, such as instruments or writing (Luckmann 1991:159). They are actualized in two characteristic ways. First, they are performed, as in instruction or mimesis. Workers hurry off to work or writers propound the goodness of time thrift. Second, times manifest themselves in artifacts, such as notations and instruments for measuring time.<sup>4</sup> The clock, bells, closed gate, and posted schedule are examples from Thompson applicable here. Both these kinds of actualization, however, depend on the field of potentials established by time knowledge. These practices depend upon, situate, and interpret representation. Thus the actual and virtual are codependent; someone must know how to read a clock for the clock to actualize a practice of time and, similarly, to tell someone they are late requires a clock showing that they in fact are.<sup>5</sup> Times operate through meaningful codes, such as hours, events, or flows. But these codes must be interpreted and put into use. Because of the interrelation of the field of potentials surrounding a practice of

---

<sup>4</sup> I have given discourse a very minor role in this argument because discourse about time is vague, has difficulty locating itself, and hastily translates other aspects of temporality into its own terms. The Foucaultian concept here points only toward the vagaries of how time is discussed, not towards the plurality of temporal functions and social practices of time by which time acts in the world.

<sup>5</sup> Gilles Deleuze differentiates between the actual and the virtual. The virtual is a qualitative potential (what can be) immanent to the actual. The actual is a concrescence of the virtual into something particular, embodied, material, and existing (Massumi 1992, 34-35; Deleuze 1988, 42-43; Terranova 2001, 107-109).

time and the actualized bodies that animate it, social practices of time are not strictly human. Times require knowledge, activity, and technology.

The human experience of time is only one element in a practice of time. Ongoing work by Paul Glennie and Nigel Thrift identify the clock as a component of distributed cognition in urban areas. “Clock time comprises a number of *concepts*, *devices*, and *practices*” that work together in historically and geographically variable ways, for various communities of practice (Glennie and Thrift 2009, 9). Time practices depend on the imperfect cooperation of these elements, but result together in the scheduling of leisure, measuring of movements in a task, and recording of history.

Despite limits on the subject’s experience of time explained by phenomenology (Husserl 1964), technologies extend and modify thought, experience, and the performance of human social life. Although humans cannot experience time as a series of discrete states (due to protension and retention), they can go back through backed up copies of their files on a computer *as if* time were a series of discrete states. In fact, navigating through a series of states is both the exact target of the phenomenological critique (Hoy 2009, 68) and the exact design suggested in at least one computer science textbook (Gamma 1995, 62–63). Computer technology, in this case, augments the human capacity to experience and act in time. What is possible for time supported by one technology will be different from what is possible in other times.

These times act together, jostling, ignoring, or overpowering one another. The mixture of times that characterizes experience is a product of many practices of time

that can be distinguished, and which we do in fact distinguish in ordinary life.

Catching a plane, checking a watch, and timing a joke all involve different kinds of time that can happen together. Often, the joke is not substantially affected by being on an airplane, and the time told by the watch, though it is has a different significance on the airplane than it did in the airport, is not hard to distinguish from the timing of the joke.

Social practices are not always neat, but they can be separated out. They are worth studying because they are not just individual habits or unexpected events; they are concretely observable patterns of behavior that are visible in their enactment, that often constitute social relations, and that become sites of contestation (Swidler 2001). Practices are repeated (Shove, Pantzar, and Watson 2012). This fact makes practices matter, because what has repeated many times before is likely to repeat again in the future.<sup>6</sup> The downside to the conceptual tidiness of the concept of practice is that communities of practice are not well defined (Glennie and Thrift 2005, 165) while the concepts of practices are defined too well; we can understand practices with a diagrammatic clarity that actual events lack. No day is average, but we still speak of an average day. Still, *what does happen does tend toward* certain repetitions that can be characterized, and in many cases, those characterizations are worth making.

### **Conclusion: What Times Are**

---

<sup>6</sup> Practices depend on and are defined by one particular function of temporality: repetition. All talk of practices describes only that which repeats, failing to prepare us to understand that which occurs singularly or by (another function, such as) hope, opportunity, or what is ancient.



We notice time most when it is strange. We become confused by time out of joint and wonder: what is time? Rather than demand a single explanation that gives a single master term for all of temporality, we should trust our instinct that time is confusing and not reducible to a timeline or experience of passage. Temporality does not work in one way; it functions in many ways. We do not practice time only to register temporal passage but for many reasons. The ways that temporality functions get caught up in social practices in large part because we want them to. Temporality happens in the world, so we involve ourselves with it. By growing living systems of time, we make more times and the times begin to animate activity. Some meet each other and social relations form. Others miss the opportunity and there is no encounter.

Different times do not do the same job. Only if we presume a single function that all times are doing (such as measuring passage or ordering sequences) is it necessary to see how all cultures do something equivalent to the modern reification of time. Times, however, organize different aspects of temporality in their own ways for their own purposes.

Interesting things are happening in time. Social practices, particularly those with new technological components, are organizing temporality in exciting ways. The older associations of time with pace, irreversibility, and mapping the moments in which events occur are no longer providing new answers. It is not that these times have died out, only that other times have made exceptions to them. Pace, once a crucial use of time symbolized by the clock's tick, has given way to variability. Almost no feature of contemporary digital technology guides us with pace. Everything

happens when the user requests it or when the computer finishes processing it. With few exceptions, we have become dramatically freer to answer a message after a day or after an hour, to take stretching breaks on our own schedule, and ignore a task while it is executed. The passing of time is becoming less important than the accrual of possible events or of sequences that can be activated at any moment. Indeed, these practices challenge the very meaning of temporal terms such as memory, event, and even the contemporary. They also exist at the edge of what most people would comfortably refer to as a time.

If there is no minimum requirement for what may be considered a time, except that it is a practice organizing temporality, are all social practices times? This suggestion contradicts everyday experience. Few would refer to the time of paying bills, though we would consider many aspects of farming as times. Historically, a small number of time practices reified a notion of time that now gives us a limited and misleading referent when trying to see time at work around us. We expect times to be those things that do what the clock does. The more a practice is like a system of gears, the more easily we accept it as a time. Ironically, this means those things most easily called times contribute the least to our knowledge of what times can do. In this sense, paying bills simply does not sound like time, though it does have its own logic of rhythm, procrastination, suddenness, opportunity, and delay. (For many, though, payday is a very real time indeed.)

A historical answer can be supplemented with a theoretical one. A social practice *referred to* by other practices for its organization of temporality *serves as a*

time. The calendar is a time because it is such a standard referent. We use farming as a model for time with harvest festivals, our associations for the seasons, and the switch to daylight savings time. Paying bills, on the other hand, is simply not referred to as a kind of time and does not widely influence how other practices organize temporality.

Today, a large number of social practices doing interesting things with temporality have become important to social life, and so we each have the hunch there are many times in effect. We are not just living in the time of the clock and the calendar; we live in the time of farming, post-Fordist labor practices, comedic timing, involuntary memories, trauma, relaxation, transportation schedules, slow historical changes, interactive computer-mediated time, and in the still-changing times of care.

The opportunity before us is to recognize the nearly invisible temporal operations caught up in events, to figure out what they are doing, and to relate this to transformations of time, large and small. Which times have accelerated? Which conjoin into a reinforcing set that complement the others and share their strength? Which do not? What has driven other times to emerge and what projects do they support? How do these various times relate to each other, and what can we expect of them? These questions are the topics of the next chapter.

Chapter 2 and 3, in part, are reprints of material as it appears in “Time as a Social Practice,” *Time and Society*. Forthcoming. The dissertation author was the primary investigator and author of this paper.

## References

- Adam, Barbara. 1995. *Timewatch: The Social Analysis of Time*. Cambridge, Mass: Polity Press.
- Atkins, Barry. 2007. "Time in Prince of Persia: The Sands of Time." In *Videogame, Player, Text*, edited by Barry Atkins and Tanya Krzywinska, 237–253. Manchester: Manchester University Press.
- Barad, Karen. 2007. *Meeting the Universe Halfway: Quantum Physics and the Entanglement of Matter and Meaning*. Durham: Duke University Press.
- Bergmann, Werner. 1992. "The Problem of Time in Sociology." *Time & Society* 1 (1) (January 1): 81–134. doi:10.1177/0961463X92001001007.
- Bergson, Henri. 1896. *Matter and Memory*. Translated by Nancy Margaret Paul and William Scott Palmer. London: Allen & Unwin.
- . 1910. *Time and Free Will: An Essay on the Immediate Data of Consciousness*. London: Routledge.
- Bogost, Ian. 2009. "Videogames Are a Mess." In Uxbridge, UK. [http://www.bogost.com/writing/videogames\\_are\\_a\\_mess.shtml](http://www.bogost.com/writing/videogames_are_a_mess.shtml).
- Brann, Eva. 1999. *What, Then, Is Time?* Lanham MD: Rowman & Littlefield.
- Brodsky, Joseph. 1995. *On Grief and Reason: Essays*. New York: Farrar Straus and Giroux.
- Butler, Judith. 2008. "Sexual Politics, Torture, and Secular Time." *The British Journal of Sociology* 59 (1): 1–23.
- Caruth, Cathy. 1996. *Unclaimed Experience: Trauma, Narrative and History*. The Johns Hopkins University Press.
- Chakrabarty, Dipesh. 1997. "The Time of History and the Times of Gods." In *The Politics of Culture in the Shadow of Capital*, edited by Lisa Lowe, 35–60. Durham NC: Duke University Press.
- Durkheim, Émile. 1995. *The Elementary Forms of Religious Life*. New York: Free Press.
- Eliade, Mircea. 1959. *Cosmos and History: The Myth of the Eternal Return*. New York: Harper.
- Eriksen, Thomas. 2001. *Tyranny of the Moment: Fast and Slow Time in the Information Age*. London, UK: Pluto Press.

- Fabian, Johannes. 1983. *Time and the Other: How Anthropology Makes Its Object*. New York: Columbia University Press.
- Frank, Adam. 2011. "Beyond The Punch-Clock Life: The Tyranny Of Modern Time II." *NPR*. October 14.  
<http://www.npr.org/blogs/13.7/2011/09/27/140818962/beyond-the-punch-clock-life-the-tyranny-of-modern-time-ii>.
- Gamma, Erich. 1995. *Design Patterns: Elements of Reusable Object-Oriented Software*. Reading, Mass: Addison-Wesley.
- Gell, Alfred. 1992. *The Anthropology of Time: Cultural Constructions of Temporal Maps and Images*. Oxford: Berg.
- Giesen, Bernhard. 2004. "Noncontemporaneity, Asynchronicity and Divided Memories." *Time Society* 13 (1) (March 1): 27–40.  
 doi:10.1177/0961463X04040741.
- Glennie, Paul, and Nigel Thrift. 2005. "Revolutions in the Times: Clocks and the Temporal Structures of Everyday Life." In *Geography and Revolution*, edited by David Livingstone, 160–198. Chicago: University of Chicago Press.
- . 2009. *Shaping the Day: A History of Timekeeping in England and Wales 1300-1800*. Oxford: Oxford University Press.
- Hammer, Espen. 2011. *Philosophy and Temporality from Kant to Critical Theory*. Cambridge; New York: Cambridge University Press.
- Harman, Graham. 2009. *Prince of Networks: Bruno Latour and Metaphysics*. Prahan Vic.: Re.press.
- Heidegger, Martin. 1949. *Existence and Being*. Translated by Werner Brock. London: Vision Pr.
- Hellsten, Iina, Loet Leydesdorff, and Paul Wouters. 2006. "Multiple Presents: How Search Engines Rewrite the Past." *New Media & Society* 8 (6) (December 1): 901 –924. doi:10.1177/1461444806069648.
- Hoy, David. 2009. *The Time of Our Lives: A Critical History of Temporality*. Cambridge MA: MIT Press.
- Hubert, Henri. 1999. *Essay on Time: A Brief Study of the Representation of Time in Religion and Magic*. Oxford: Durkheim Press.
- Husserl, Edmund. 1964. *The Phenomenology of Internal Time-Consciousness*. Bloomington: Indiana University Press.

- Jalas, Mikko. 2006. "Making Time: The Art of Loving Wooden Boats." *Time Society* 15 (2-3) (September 1): 343–363. doi:10.1177/0961463X06066945.
- Kern, Stephen. 1983. *The Culture of Time and Space 1880-1918*. Cambridge Mass.: Harvard University Press.
- Koller, John M. 1974. "On Buddhist Views of Devouring Time." *Philosophy East and West* 24 (2) (April): 201–208.
- Koselleck, Reinhart. 1985. *Futures Past: On the Semantics of Historical Time*. Cambridge Mass.: MIT Press.
- Latour, Bruno. 2002. "Gabriel Tarde and the End of the Social." In *The Social in Question: New Bearings in History and the Social Sciences*, edited by Patrick Joyce, 117–132. London: Routledge.
- McKendrick, Neil. 1961. "Josiah Wedgwood and Factory Discipline." *The Historical Journal* 4 (1) (January 1): 30–55.
- Miller, William Watts. 2000. "Durkheimian Time." *Time & Society* 9 (1) (March 1): 5–20. doi:10.1177/0961463X00009001001.
- Moran, Chuk. 2010. "Playing with Game Time: Auto-Saves and Undoing Despite the 'Magic Circle'." *Fibreculture* (16) (July).  
<http://sixteen.fibreculturejournal.org/playing-with-game-time-auto-saves-and-undoing-despite-the-magic-circle/>.
- . 2013. *Superactually: Micro-essays on Post-ironic Life*. Ropley: Zero.
- Mumford, Lewis. 1964. "Authoritarian and Democratic Technics." *Technology and Culture* 5 (1) (January 1): 1–8. doi:10.2307/3101118.
- Mumford, Stephen, and Rani Lill Anjum. 2011. *Getting Causes from Powers*. Oxford; New York: Oxford University Press.
- Newton, Isaac. 1962. *Sir Isaac Newton's Mathematical Principles of Natural Philosophy, and His System of the World*. Berkeley: University of California Press.
- Nitsche, Michael. 2007. "Mapping Time in Video Games." In *Situated Play: Proceedings of the 2007 Digital Games Research Association Conference*, 145–151. Tokyo: The University of Tokyo.  
[http://www.digra.org/dl/display\\_html?chid=http://www.digra.org/dl/db/07313.10131.pdf](http://www.digra.org/dl/display_html?chid=http://www.digra.org/dl/db/07313.10131.pdf).

- Pollard, Sidney. 1963. "Factory Discipline in the Industrial Revolution." *The Economic History Review* 16 (2). New Series (January 1): 254–271. doi:10.2307/2598639.
- Radovan, Mario. 2011. "Time Is an Abstract Entity." *Time & Society* 20 (3) (November 1): 304–324. doi:10.1177/0961463X10371882.
- Rayment-Pickard, Hugh. 2004. *The Myths of Time: From Saint Augustine to American Beauty*. London: Darton Longman & Todd.
- Shaviro, Steven. 2009. *Without Criteria: Kant, Whitehead, Deleuze, and Aesthetics*. Cambridge Mass.: MIT Press.
- Shove, Elizabeth, Mika Pantzar, and Matt Watson. 2012. *The Dynamics of Social Practice: Everyday Life and How It Changes*. Thousand Oaks, CA: SAGE.
- Swidler, Ann. 2001. "What Anchors Cultural Practices?" In *The Practice Turn in Contemporary Theory*, edited by Theodore Schatzki, 83–102. New York: Routledge.
- Thompson, E. P. 1967. "Time, Work-Discipline, and Industrial Capitalism." *Past & Present* 38 (1) (December 1): 56–97. doi:10.1093/past/38.1.56.

### III. Times Relating

Social practices of time grow around one another densely. Some are parasites, surviving off the strength of others. Some spring up wherever there is space, only to wither quickly in the shade of another. Some times provide for others, yield a useful substrate, or mend what is broken. Some keep to themselves and survive a little while. A very few change the ecosystem where they live and keep themselves large and strong. There are many social practices of time. How do they relate to each other? How have existing practices of time actually related to each other historically?

There is a commonly repeated answer to these questions: times have accelerated. Though this seems an apt summary for economic changes and some cultural ones, it is less appropriate for other times. The speed of shipping, traveling, sending a report, or getting a new career and leaving it have accelerated. But gardening, social justice, the migration of birds, or the time spent working on a craft have not. The time-space compression thesis, laid out most clearly by David Harvey, summarizes changes in time practices by translating them into *measures of time* and then recognizing how they have been bent to serve this measure. Because traveling by air, for example, has been interpreted, through measurement, as a number of hours spent doing a generic activity called traveling, we can now say that traveling has accelerated even if, at 30,000 feet, one experiences less speed than on a locomotive. Although the acceleration thesis seems to be able to describe practices of any kind, I will argue that it is a product of the ways that certain practices of time have become dominant and have come to regularly represent other times. To understand how times



have related to each other historically requires paying attention to how times are enacted in the context of each other.

Times relate in specific ways, as a rule, but two general types of relations are worth special attention: translation and material influence. Translation refers to the way one time represents, or fails to represent, another time. The times involved in scheduling a flight, getting to the airport, waiting for the plane, waiting on the plane, and arriving can all be translated into a practice of time wherein the only relevant number is the duration of the flight. Translation is a common relation because, as established in the previous chapter, every time is a social practice giving meaningful codes by which to understand actual events; one time translates another by interpreting events in its own terms, ignoring the other codes of interpretation, such as the traveler's experience or structured practice of those flying the airplane. A second primary type of influence is material influence. This is a broad category reflecting the embodied nature of times as practices. One time can entrain the rhythm of another, deny its resources, act upon the actors of the other time, encourage the growth of one time, or provide wider usage for another time. These two types connect to many variants and ultimately to an unknowable plurality of potential relations.

In this ecosystem of times, a few have become widespread, have found commonality among themselves, and have entered an alliance to create a reification of time. This reification is now our common sense. In some cases, these practices have been self-serving; in others they have been in the right place at the right time. In each case, it is because practices have been useful that institutions, practices, and groups

have given them force. The history of time is not a simple thing because times are not practiced all at once, neither by all people equally nor always in the same way; they intermingle and are used in various ways, observed with partial regularity, and developed by interests whose power may come or go. At one time, colonial interest in standard times at their various colonial ports succeeded in realigning time; at another point, scientific work dealing with milliseconds organized and implemented a change. Despite the differences, several features built up and found common ground. The solar day became the clock's hours which became connected to a system of time zones, new practices for maintaining clock accuracy, and eventually these practices of time took as their basis the second and its multiples rather than the day and its fractions. Today, we accept without hesitation that time is that dimension of the universe measured by clocks. This is quite a bit of metaphysical pomp surrounding the relatively simple mechanics of a clock.

This chapter explores the relations between times in order to further the overall research question about interactive times. If some times are interactive, why did people start living with these strange things at all? How can they survive in a world that is, at least apparently, thoroughly dominated by times that are not interactive? What is the current state of the time ecosystem in which interactive times appear? To approach these questions requires a clear sense of how times relate and how they have related.

### **How Times Relate**

The basic syntagm is familiar: we used to get together to watch *Project Runway* in 2008 on TiVo. The memory, the historical period, the TV season, the day of the week, the time of the workday, the scheduled time of the show, the wait while everyone showed up, the hour we spent together, the segmentation of content and advertisements, the sequence of the episodes, the timeline within the show, the editing of an episode, the stylistic references of the designers, the harried pace of the design process. These are the paradigmatic times involved. But what are the paradigmatic *relations*?

Times always relate to each other in ways specific to a situation, but we can still name some general types of relationships that tend to form. In *Timewatch*, Barbara Adam describes the relations between times with the catchall term *mutual implication*. The phrase suggests play and mutual presupposition; complex relations cannot be frozen into a single image (Adam 1995, 159). Through the book, the phrase provides several specific meanings. In this section, I draw out types of relations from these varied uses of the term. This section will begin with times irrelevant to one another, explore how times translate each other, and then explore material influence and mutual reinforcement.

Mutual implication, as a term, ambiguously suggests that all times somehow relate to each other in the end. However, Adam is quick to point out that gaps are more common than connections, and that non-interactions can be more important than connections.

Most of these times are implicit, taken for granted, and seldom brought into relation with each other: the times of consciousness, memory and anticipation are rarely discussed with reference to situations dominated by schedules and deadlines. The times expressed through everyday language tend to remain isolated from the various parameters and boundaries through which we live *in* time. (Adam 1995, 12)

Though times may all influence one another at some point, disjuncture is more common than connection. Watching TV with friends, the time we agree to meet is unaffected by the timing of stoplights between our homes, cannot change or be changed by the retro style of a dress in the show, and has no bearing on the relentless intensity of the advertisements we TiVo through. Disconnection can be active or passive; one time may willfully ignore another (we set a meeting time without consulting one person's schedule), or two times may happen to have no relation (the retro dress and the stoplights).



Figure 3.1 Christian Siriano creates a winning outfit inspired by Don Andrés de Andrade y la Cal in episode 11 of Project Runway Season 4, originally aired February 13 2008 (Murillo 1665).

Most times co-exist in a degree of passive non-interaction. Because the times do not envelop and overrun each other, opportunities remain for times to grow, flourish, and find a niche. Using the examples of the dissemination of the mechanical clock and of railway time, Jon May and Nigel Thrift suggest that there have been “various (and uneven) networks of time stretching in different and divergent directions across an uneven social field” (May and Thrift 2001, 5). A practice of time will interact with times within its reach, changing local patterns of life and giving new terms to thoughts of the past and future. But, outside its reach, it may very well not. Towns near railroad stations feel the importance of the times used by railroads, but those further away easily find other points of reference. For friends huddled around

the TiVo, the time of the show is controllable. We can rewind to watch a segment again, pause on an image of text and read all the words, fast forward, start and stop when we please, or take a break and come back in a bit. Within this situation, the interactive time of the TiVo can influence or overpower some actors and some times, but has no power over others (such as the season or time of day). The TiVo, like many computer based times, radiates a small space wherein its time has power. Geographic distance is one expression of non-interaction, but times may also ignore each other up close and personal. The times of history, season, and waiting may operate in the same space and on the same actors, without having much relation to one another. But non-interaction between times isn't always a result of simply being of different kinds or in different places; ignorance can be educated.

Active non-interaction is a politically significant failure of communication. A town that refuses to accept the railroad's time, a factory that does not recognize Monday as a drinking holiday, a government that gives no parental leave. Feminist studies of time emphasize that the time spent caring for other people has been continuously ignored by employers and policymakers. Dominant practices of time are numb not only to hours spent (mostly by women), but to the *kind of time* care represents. Staying up with a sick child is not exactly measurable as hours on the clock; living with someone older who needs special attention will condition one's working hours but does not simply consume them (Bryson 2007). Sub-practices of one time can be apprehended and reinterpreted in the meaningful codes of another.

Active non-interaction is really a failure of translation. One time does not register another properly. Sometimes this is done by choice: an employer would rather not give more sick leave. Very often, though, one time actually cannot represent aspects of the other time in its own idiom. If the system is to count up the number of hours spent doing something, how many hours can one claim for the effort of caring for a sick child? Times cover some temporal terms and catch them up in a practice; other temporal terms, and other parts of other practices, may not fit at all. In care, an inconsistent pace of events interrupts other activities; the schedule recognizes only beginnings and endings connected by measured durations. Adam describes the mutual implication of the time of care and the economic measurement of time as not just intermeshed with the commodity, but “evaluated through the mediating filter of that economic time” (Adam 1995, 99). That which has value in the arena of exchange becomes visible, that which cannot be traded cannot be recognized. The coded representations by which one time works subsume another time.

Bliss Lim further develops the idea of translating time through Henri Bergson. Though clocks are, from a Bergsonian perspective, fundamentally translating duration, they cannot grasp duration as a qualitative multiplicity. However, they still systematically represent it. This is a regularized mistranslation. In the same way, Lim argues, the dominant time practices of modernity represent the times of other cultures, denying a culture’s own ways of understanding time. Ghosts no longer stand for ancestors visiting the present; they become delusions and myths demonstrating a culture’s rich *storytelling tradition*. Despite its attempt to translate a culture’s own

time into the secular schema of modernity, modern time cannot account for specters *in terms of time* and so relegates them to the status of superstition and folklore.<sup>1</sup> Here, one time sorts another into an intelligible portion (a model of the original) and a cultural supplement. For Eurocentric developmentalist narratives, the time of other cultures appears as a muddled and discrepant version of their own time – of what they would no doubt say time really is (Lim 2009). From the point of view of those undertaking the project, the translation effort has been successful: modern time has converted lunar and solar calendars into its terms, assigned hours to the day, reconstructed histories on its timeline, and translated local rhythms into periods of days or years. Though much is lost in this rough translation, this representation of other times has become dominant in many parts of the world (Postill 2002).

With translation comes the possibility for the universalization of a time. If a time can account for most everything belonging to other times, what do other times have to offer? Here is a common view among students learning about suspiciously complicated theories of time: time is a dimension, and that is good enough. The dimensional account can provide a translation of any other time, so nothing more need be said. Ergo, historical time is a matter of how academics write; time systems of different cultures are just approximations of the Western system; philosophical ideas about time describe psychology and not reality. Like singular accounts of time's

---

<sup>1</sup> Stefan Tanaka describes such a displacement in Meiji Japan, where important parts of a traditional time, such as shell mounds from giants or catfish that make earthquakes, were rediscovered as *folklore* to be scientifically catalogued (stories about gigantism) while the physical phenomena in question (shell mounds and earthquakes) were reclassified into other domains – namely, geology (Tanaka 2004).



essential nature, this enthusiasm for a single time establishes a universalizing pretension. It does this, however, at the expense of many aspects of other times that are simply redefined outside the terms of time, as with the displacement of ghosts into folklore.

This universalization, haughty as it might be, has actually been very rewarding for scientific concepts of time and practices of time central to modern bureaucracy because it allows them to expand their empires. Most science uses a roughly Newtonian model of time, making work in different experiments—even in different fields—more easily combinable; this allows a larger empire to develop more quickly. The arrow of time in science, for instance, can offer

compelling explanations of a vast panoply of phenomena, from the emergence of life to the appearance of a leopard's coat. If we dismiss the arrow of time as an illusion, we must forfeit all the insights we have gained. This would surely be an enormous sacrifice - and all we would gain are the absurdities of a world-view in which bowls of soup could heat up of their own accord, and snooker balls mysteriously pop out of their pockets” (Coveney 1991, 260).

It is as if other accounts of time were to be feared.

When people regularly use one time to translate other times, times can take on the relationship of *referent*. A referent time serves as an ideal toward which other times should be directed, or at least with which other times become easier to think about. The clock is our most general referent for times. It's difficult to even recognize something as a time except to the degree it resembles the clock. Adam discusses times as models in another passage on mutual implication. In the relation of the “archetypal and endogenous temporality of the birthing process and the rational clock time of

obstetrics,” Adam argues that these times do not just oscillate, but “interpenetrate and mutually inform each other’s meanings” (Adam 1995, 49–50). The obstetric schedule has a basis in the experience of birthing and can vary during a birth to better model it. However, a reciprocal modeling takes place too often, where the actual birth may be condensed to fit a given obstetric schedule.

In the relation of a model, one time is understood, designed, or practiced in the image of another, or with significant borrowing of elements. The model often changes the very form of another time, not just translating content into it, but actually changing it to better accommodate that content. Segments of advertisements during *Project Runway* that are modeled on viewer’s attention will coax and shape that attention; the time in the evening when the show runs models the work schedule of an intended viewership. Because many practices of time model themselves on others, modeling can be iterative. For example, when NASA scientists worked on the Mars rover, the hours of their work had to switch to Mars time, a variation of the usual solar day of Earth. This time depends on the clock (special watches that run a bit slower than usual), but *models* its structure on the day, which itself *refers* to the rising and setting of the sun (Mirmalek 2009). By reference to already accepted times, many practices organizing temporality become legible *as times*.



Figure 3.2 From left to right: time at one Mars site, dual faced Earth watch, fatigue measure, and solar time for a second site on Mars. (Mimalek, 2008)

The common sense account of time might imagine times modeled on other times as derived from increasingly faithful originals. We traditionally see practices of time as imitations of an underlying transcendental essential reality. We often imagine that the clock is measuring the constant flux by which all clock-like mechanisms can engage in movement. The clock seems an index of a more fundamental dimension of the universe that is time's true nature. But regular motion is just another universal account, preferring mysterious simplicity to tangible complexity. By instead seeing practices as related to one another by reference, we focus attention on how time is enacted and lived, rather than an ideal form to which some of these practices allude. Idealizations and referents are tactics within the many practices of time and their shifting relations. Most times are not conceived in this way; the time practices of passengers in an airport or comedians improvising on stage are rarely seen as manifestations of a deeper underlying essence of time. Lacking profundity in their references, then, such times are often seen as derivative, when they should instead be

recognized as material enactments that, unlike others, simply do not attract an interest from those speculating on metaphysics.

As can be seen with reference, universalization, translation, and numbness, the relations between times are not all conceptual. In each of the above cases of translation, someone must make a translation and someone must act on it. Times are not just concepts covering temporal terms; they are living practices shaping and connecting temporal influences. NASA scientists trying to keep on Mars time lost sleep, obstetricians rush birth, colonial powers govern societies with very different calendars, and work schedules wreak havoc on the time of care. It matters if one time translates another properly because the mistranslation will have an impact. If we schedule a weekly meetup but ignore one person's schedule, such that she is late every week, our meeting will de facto begin late every week on account of the one time's insensitivity to another. If we schedule an hour to watch the show but skip the commercials, we only take 42 minutes. The now familiar compression of one hour shows to the 42 minutes of programming is a clear product of interactive time.

Translation matters materially and material influence is itself a critical register at which times (themselves partly material) relate. In most of the passages where Adam discusses mutual implication, this is what she means. Times affect other times. Times are active simultaneously (15), the addition of new times changes those that already exist (28), birthing time and obstetric schedules intermesh (50), and again commodified time and the time of care "affect each others quality and meaning" (Adam 1995, 99). Mutual implication also means mutual change.

In the sociological literature on time, there are many examples of powerful interactions between rhythms. In one case, the schedule of the local trolley influenced the shift schedule of workers at a prison (Bluedorn 2002, 148). In another case, government changes to the week in revolutionary France (the switch to a 10 day week) and Soviet Russia (the dissolution of the weekend) resulted in a split between the obedient cities and traditional countryside. This undermined the rule of law and rural workers simply took holidays from both the new calendar and the old one (Zerubavel 1985, 10, 42–43). Lefebvre's incomplete project of rhythmanalysis would have put this interaction at the center of all social processes: the world as rhythms of social life in complex interactions (Lefebvre 2007). But these visions are perhaps a bit monocausal and mechanistic; they stem from a worry that time must unify society or that all social life must have behind it a single common aspect of time.

Influence can come in many forms beside translation and rhythm. The Taylorist administration of time that arose in the workplace has moved to home life, where it now drives employees to spend more time at their post-Taylorist jobs where they can relax for a moment (Hochschild 1997). One practice drives its practitioners to another practice. Contemporary practices of listening to music have encouraged shifts in compositional strategy, because audiences enter and exit the piece at the push of a button (Kramer 1988, 45). Audiences become used to one kind of time and this makes them ready for other practices they would not have enjoyed as much before. The ability to quickly retry a segment of a video game encourages short-term planning and easier gameplay at the expense of virtuoso play and memorization (Moran 2010).

Changes in time practices change the nature of computer game play. Many have argued that the relaxed and reversible time afforded by word processors “tends to undermine the investment in the original act of writing” (Simpson 1995, 66) or encourage a “conversational style of emails, tolerant of typos, betraying little anxiety over literary *amour proper*” (Tomlinson 2007, 110, emphasis in original).

Though it only tells part of the story, material influence is an important aspect of how times relate. For friends watching *Project Runway*, the finale may bring us together more punctually than other episodes, the TV season and harried process of design influence the way the show is edited, and the historical period in which the show comes out is not irrelevant to the designers’ stylistic references. By understanding times as practices, it becomes clearer how easy it is to affect them, and, conversely, how influential each practice is itself. Often, these material relations work in conjunction with other relations to accomplish a larger effect. The mixture of times, in their mutual implication, does not make all times equal, at a political level. Some are doing much better than others.

Times reinforce, complement, contradict, or replace one another historically. The ways that times relate can be usefully understood in generic terms, although their operation will always vary in specific cases. Many times have no direct relationship with each other, and this makes it worthwhile to see connections as they do exist and where they do develop. Times translate each other, with incomplete success, and often in regularized ways. In translation, they can actively fail to translate one another, claim a universal status for their own representation of temporality, or serve as a

model for other times. Times influence each other materially, blocking each other, privileging some times over others, accelerating other times, or interacting in some other way. Times have related to each other in various ways for a long time. These plays of power and accident involve many kinds of relations and happen between several times at once. Adam refers to a reinforcing set of times that have together reified time as an external object (Adam 2004, 138). To understand this accomplishment requires some idea of the growth and change of major practices of time up to the present moment.

### **Time History**

It is not possible to tell the history of all kinds of time. Conversely, it is all too easy to tell a very partial story of the early life of certain technologies and ideas recognizable now as fundamental to time. Such an account goes like this: Socratic and enlightenment philosophy of time paired neatly with the development of clocks to produce scientific models of time (of Newton and then Einstein) which have their own advanced correlate in a very precise system of internationally synchronized timekeeping (Mainzer 2002).

Such a history presumes the end result as the form to which all intermediaries tend. It ignores the other forms of time that grew up and faltered, that were profoundly modified, or were excommunicated from the set of practices now composing what is colloquially referred to as time. It correlates a small set of practices, usually restricted to the elite, with a wide range of ideas, summarized best in a few texts, and ignores the geographic and political situations enlisting times into larger practices. It is not a

useless history; it is actually quite useful for those times that are dominant today, but a broader perspective shows adoption and transformation of times in relation to their uses and mutual implication.

There have been many practices of time, but a dominant, reinforcing set has emerged in the last few hundred years that makes other kinds of time relatively unthinkable. The set has established itself through a series of modifications—which happened for various reasons—that have made the translation of seconds to minutes to hours to days to weeks to months to years natural. The whole system seems now to be determined by the motion of the Earth, a fundamental dimension of the universe, and a completely normal way to pace and schedule labor, leisure, and so much more of life. This is a historical occurrence built up primarily in Europe and then North America, and later adopted or applied elsewhere, that might have happened otherwise or not at all (Lim 2009, 84–86). By understanding how changes *did* enshrine dominant times, it becomes clearer how contemporary changes may undo the reinforcing set.

Clocks were not always important. Religious calendars, sand hourglasses used by blacksmiths, the position of the sun in the sky, church bells, and time cues from everyday life allowed common people in Europe before 1450 to know the general time of day (Thrift 1981, 57–59). The idea of a day divided into hours was not unfamiliar, but a machine used to register this would have been. Clocks were rare, hard to make, and not particularly useful. At best, they supplemented existing time practices. In Europe at this time, most social organization centered around the completion of tasks rather than the number of hours spent engaged in the task. Workers worked on a



specific job, not for a specific number of hours; leisure was similarly arranged around activities that found their ends for reasons other than the hour (Glennie and Thrift 2005, 170–171).

However, time found a new importance with changes in the work ethic (following the Reformation) and growing desires for what industriousness could make. The protestant work ethic's attack on idleness and its careful attention to the proper use of time (Weber 1905/2005, 104) is well known, though perhaps overstated. The general idea is important: there became theological reasons for time thrift, turning any short stretch of time into a medium for productive activity. Though records are imperfect, watchmakers in France and Germany were disproportionately Protestant from 1500-1700 (Landes 1983, 92). In the same period, a revolution in consumer practices began to fuel the Industrial Revolution. Changing tastes preferred market goods to homemade ones; tea (which could not be made locally) became more exciting than beer (often brewed at home). It therefore became worthwhile for people to work (or, more exactly, work to produce money) when they otherwise would not. The hours of night and the less busy seasons of agriculture, once bastions of more relaxed time practices, became colonized with work intended to produce money with which to acquire market goods. These crafts done at home required different paces of work, requiring new strategies for domestic production (de Vries 1993). A principled opposition to idleness and a new motivation for industriousness pressured existing practices of times, encouraging the growth of more precise and productive means of scheduling and coordination.



Figure 3.3 From William Hogarth's Industry and Idleness series, this engraving illustrates the dangers of idleness. Here, the idle apprentice is gambling when he should be in Church.

Clocks became common in urban spaces through the seventeenth century.

Settlements grew and, along with them, the number of clocks. Urbanization in England doubled from the tenth to fourteenth century, then leveled off until the sixteenth century when it began rising again. The trend was similar across Europe, with the plague hitting denser settlements harder, and urban growth taking off in the sixteenth century (Rigby 2010). With urbanization, an increasing number of clocks in the immediate environment gradually made it easier to know the clock's time (which was still not understood as identical with other kinds of time). Until the eighteenth

century, it was a rare clock that had a dial; most worked by sound rather than sight (Glennie and Thrift 2009, 41). A cacophony of bells rang with different sounds produced by various forms of striking and different clock bells. These filled the day in many cities, indicating events of various kinds, such as the time for fish merchants to come to market, certain ships to come to port, meetings to commence, and soldiers to come or go (Dohrn-van Rossum 1996, 206–209).

Because clocks kept invariant hours, they transformed the meaning of the hour. Traditionally, the period of daylight was divided into twelve parts, each of which was an hour. The same was done for the night. Today, winter days give as little as nine hours of daylight. In the older system, every day had twelve hours of sun by definition; a shorter day just happened to have shorter hours (thus, nine hours of daylight would yield forty-five minute hours). This sounds strange to us now because we assume an hour *is* sixty minutes, but the minute and second were almost exclusively theoretical concepts until the late sixteenth century (Dohrn-van Rossum 1996, 282–283). Thus the length of hours varied seasonally and was different for hours of night and day. Clocks disrupted this system by keeping the same hours day and night, thereby forcing a redefinition of hours (Sherover 2003, 14).<sup>2</sup>

Clocks were primarily public. Most clocks were in churches, but town halls often had them as well (Tittler, 1991: 131-139). Schools drilled clock time into the young (to prepare them for work) and factories into the slightly older, with strict and

---

<sup>2</sup> Clocks can be built so that every day has twelve hours of daylight, but this is harder and rarely done.

standard penalties for tardiness (Thrift, 1981: 61-65). The workhouse also deployed clock time assiduously for the supposed benefit of the poor, especially children (Driver, 1993). Monumental public clocks in European cities enhanced their prestige, gave the city an identity, and signified good governance (Dohrn-van Rossum 1996, 146–156). Only in the late seventeenth century did domestic clocks become common (Glennie and Thrift 2005, 24–25).

Yet, even with a proliferation of clocks, their meaning and use remained far from clear. With clocks common in many houses and many wearing wristwatches by the seventeenth century, accuracy and punctuality were still considered social niceties rather than obligations (Landes 1983, 128). Into the nineteenth century, farmers consulted almanacs where “astrology and astronomy, theology and husbandry, medicine and science combined to schedule work” (O’Malley 1996, 16). What the clock measured was only one small consideration when thinking of time. Further, those using the clock did not all agree on what days it divided into hours; although the Gregorian calendar has become dominant today, many other calendars remained in use even into the twentieth century.

Clock time did not take over; existing practices found clocks useful. The idea of usefulness provides a middle path between visions of technologies as overpowering or as overpowered. On the one hand, many would readily claim that changes in time alone forced a new pace of life on peasants, for example, capturing them in a “chronological net” (Harvey 1989, 228). On the other hand, the image of different

groups using technologies for their own varied purposes can imply that a technology does nearly anything that is asked of it. A technology is supple, but not infinitely so.

Usefulness is the ability of a technology to be applied in a specific role in a general scenario. These scenarios have a correlate in actuality, but tend to be paradigmatic and most stable in the imagination. If a new technology fits into existing needs, practices, and budgets, it may be more useful and become regularly used (Constant 1978; Landes 1983). Different groups found the clock's time useful relative to different scenarios. Early factories found times based around the clock useful as part of a disciplinary production system in a capitalist economy (Thompson 1967; Pollard 1963; McKendrick 1961). Foucault goes further with this theme, showing how several deployments of time (most using the clock) were useful for various aspects of disciplinary institutions (Foucault 1977, 141–162). Public maintenance of time had special meaning for federal and regional government in the United States in the late nineteenth century, who used it to establish their authority and associate their power with order in the world (McCrossen 2007). More could be said about the special cases of churches, military timetables, organization of leisure events, school bells and deadlines, production schedules, and the maze of rates, periods, and delays handled in accounting.

Among these various ways that clock times could be useful, two commonalities deserve special attention. Clock time was useful as a form of authority and a means of measurement. The clock displaced and extended many forms of authority, thereby increasing its own. Cities and governments using clocks to represent

order thereby invested the clock with this association; factories teaching respect for the owner's rules by means of the clock transferred respect to the clock; events that might have been organized by the authority of the sun's setting were increasingly linked to an hour indicated by the clock. Almanacs and religious calendars presented time as an expression of the authority of nature and the divine, but clocks expressed the authority of the institutions and interests that governed daily life (O'Malley 1996, 98). Slavoj Žižek defines authority as a powerless call that we feel obliged to follow without regard for the reasoning behind it (Žižek 2001, 94). We do not follow authority because we understand it but because of its position *as* authority. The clock has received this authoritative quality from centuries of use by authorities of all kinds, and today we respect what the clock says because it is the clock, not because we understand the machine, its time, or the uses by which it has come to prominence. One acts because of what the clock reads, almost never questioning this relation of obedience.

Second, clock time has generally been used to direct the productive activities of people and machines by making possible comparisons in terms of productivity per unit time. The clock can measure activities of almost any kind by creating very reductive translations; these translations lend themselves to quantitative manipulation such as miles per hour, words per minute, or man-hours of work. Charles Lorenzo Simpson has argued against a technological mode, now common, in which we tend to reduce activities to their consequences, then design and use technologies to create these outcomes more efficiently. Instead of washing dishes by hand, together with

others with whom we have shared a meal, we use a dishwasher. The dishwasher produces optimal outputs, relative to certain criteria (Simpson 1995). But efficiency depends for its meaning on parameters to be maximized and minimized. Clock time is almost always one of these parameters, giving a reductive representation of activities so they can be seen as improvable and improved. For example, consider the fact that before the assembly line, it took twelve and a half hours to build a car at Ford's factory. After the line was introduced, it took only an hour and a half to build one (Nielsen 2006, 6). This is not just a fact of interest to antiquarians and Ford fanciers; it is the exact justification for the assembly line and the key term by which Fordism proceeded to reorganize labor more broadly. The five dollar day, the pink slip, and thug control of the labor force were other advancements made to keep operations statistically impressive, especially in regard to the clock's measure (Braverman 1975, 150; Moffit 2002, 297; Norwood 2002, 176).<sup>3</sup> The ability of one time to translate another is a key usefulness that has been found for the clock and industrial efficiency engineering is one example of it. As the clock has become a means of the powerful, more within their sphere of influence has been seen and treated as it appears through the term of the clock, often in terms of efficiency measured and improved.

There are commonalities as well as differences in how time has been made useful. But there are also contradictions within common approaches. Postal systems, railroads, and telegraph networks found that clocks could be kept synchronized in

---

<sup>3</sup> Robert MacNamara's early successes treating military operations as mathematical problems probably owe a considerable debt to the callousness and belligerence of Colonel Curtis LeMay.

different cities, but the indicated hour varied by longitude; there was a tension between synchronization and astronomical time, making scheduling and measuring more difficult (O'Malley 1996, 60–61). The clock had always referred to the sun: 12 o'clock was set to match solar noon, the moment in the day when the sun was at its highest in the sky. But the sun does not rise everywhere on earth at the same time; the Earth's rotation gradually exposes its surface to the sun. Every degree of longitude (usually representing a distance of fifty miles) translates to an average difference of four minutes in solar time. (This ratio grows further from the Equator.) When travel was slow and irregular, a few minutes difference hardly mattered. If traveling fifty miles took five or six hours, who would notice four minutes? With faster travel following a more precise schedule, the difference becomes obvious. This situation became a problem for railroads, which had to make rather complicated schedules accounting for a different local time at nearly every stop. More complex schedules increased the chances of errors, and errors could result in very costly collisions. British railroad companies standardized a national time in 1847 (Thrift 1981), and US companies finally settled on a standard in 1883 (Stephens 2002, 111–115). Although these changes made sense to network owners and operators, common people did not travel far or often and found the standard time zones off-putting and irrelevant (O'Malley 1996, 119). For them, clocks properly referred to the sun, not a network of railroads connecting other clocks.



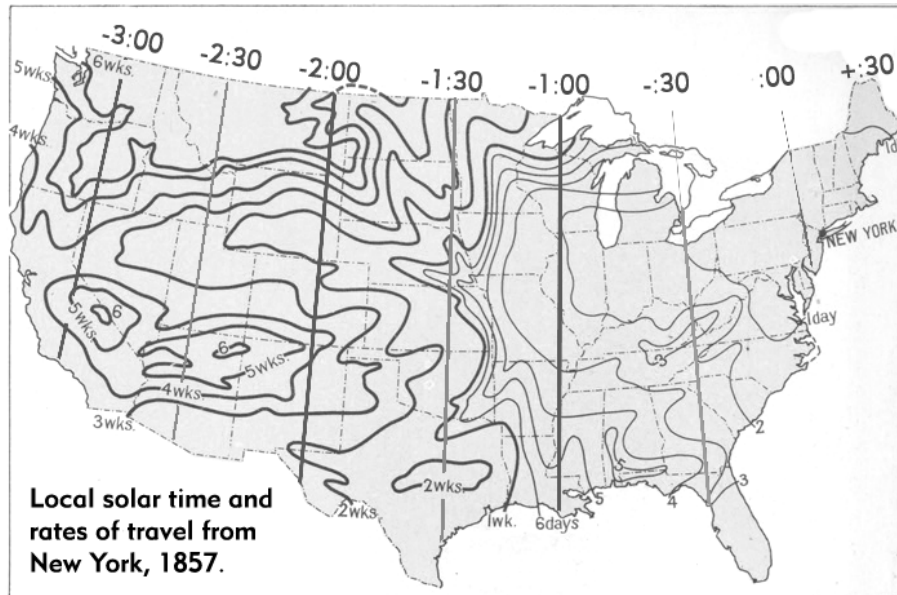


Figure 3.4 In 1857, it would take more than a full day's travel from New York to arrive in a place where local time was off by a full hour. For most people, the difference between local times was too small to matter. (Base map is from Paullin and Wright 1932, 138c)

Further standards developed to resolve contradictions in a global system of clock time. In 1884, the International Meridian Conference met in Washington D.C. with representatives from 25 nations, mostly from the Americas and Western Europe (but also including Meiji Japan and the Kingdom of Hawai'i). Unlike the railroad's concerns with different cities off by a few minutes or hours, this conference aimed to solve the problem of the date line and local times of colonial possessions. Ships sailing around the Earth sometimes found their schedules off by a day when they arrived in port. Colonial regulations also lacked standards: the local times in Portuguese Macau and the nearby Spanish Philippines differed by several hours. (Today they are in the same time zone.) Astronomy and astrology also shared an interest in a standard that

could simplify international coordination.<sup>4</sup> Advocates at the conference argued a unified system would yield benefits in efficiency like those of the metric system. The conference ultimately accepted the GMT (Greenwich Mean Time) set of 24 time zones and placed the global standard time at Greenwich and the date line in the Pacific Ocean on the opposite side of the globe (Bartky 2007).

Adoption of this standard by governments took more time, with Japan implementing the proposal in 1886 (96), the German Empire in 1893, and many other European countries soon after. In the German Empire, well-respected Field Marshall Helmuth von Moltke the Elder, regarded as one of the greatest strategists of the nineteenth century, made the case that unification of the time should follow from unification of the country and that, as a practical problem, regional variation required carefully writing and copying timetables for each region, which involved a high chance of errors that could prove dangerous, particularly in time of war. Ultimately, his death six weeks after speaking out on the topic made implementation of uniform time a way to pay respects to his legacy (122-127). In 1910, the Eiffel Tower and an antenna in Norddeich, Germany began transmitting standard time on high power radio signals (139).

In 1967, another international agreement on time set the length of a second by vibrations of Caesium, an obscure metal. The second would be equal to 9,192,631,770 transitions between two hyperfine levels of the ground state of the Caesium 133 atom, replacing astronomical time, including the GMT (Jones, 2000). Astronomical time had

---

<sup>4</sup> Astrology was, by most measures, more important than astronomy at this time.

long defined the second as a fraction of (a minute and of an hour and therefore of) a day. In that system, the basic referent is a day, and the second can be calculated from this. But, again, this common approach included small contradictions. In this case, the trouble was milliseconds. The earth's rotation is not entirely stable: the planet wobbles slightly on its axis and is always decelerating slightly (Smith 1982). This makes the astronomical second unreliable at very small scales because of wobble and, in the very large scale, due to deceleration. The difference between a second defined as a fraction of a day and a second defined by the Caesium atom is small but significant for technical work that depends on fractions of a second or precise counts of millions of seconds elapsed between two events. To hold together various uses to which a set of times were put, the whole system's referent switched from the middle term of days to what had previously been its smallest imagined term of seconds.

Similar clarifications of time for technical standards have happened outside of government channels. A persistent weakness of clock time is that most clocks are inaccurate. Watches fall behind or drift ahead, clocks driven by a pendulum, a spring, or powered by batteries will all shift by at least a few seconds over enough days. Building a perfect mechanism is extremely difficult and even a decent one is hard enough. For most purposes, though, we are as insensitive to this inaccuracy as those traveling by horse were to local times offset by a few minutes. But, for specialized users, the difference of a few milliseconds can be quite significant and drifting off by a single second can create serious complications. Computer networks now use the Network Time Protocol to synchronize clocks on the Internet with great precision.

This is a voluntarily adopted Internet standard with special usefulness for airline reservations, auction websites, and any system where the clock is not permitted to fall behind by more than a minute each month (Mills 2006). However, it is also a standard used by most personal computers, where it provides a very precise update to an imperfect internal clock. This time protocol has supplemented time's actual basis, for computers of all kinds, it is no longer just precise mechanisms running in place; a careful synchronization made every few days corrects the inevitable errancy of the counting mechanism. Here, synchronization reinforces the clock's invariant interval, rather than replacing it.

At a conceptual level, those practices of time that make a linear measurement of time found a metaphysical correlate in the idea of time as a dimension of space. Though the division of space into three dimensions dates back to the Greeks, the suggestion that time might constitute a fourth one arises only in the late eighteenth century with the work of Joseph Lagrange. Lagrange argued that, within the mathematics of mechanics (the set of physical laws describing the motion of bodies), time was a fourth dimension perpendicular to the three of space. In a way, this association can seem quite obvious to the student of mechanics: in mechanics, both time and space are represented as real numbers and can be interchanged quite smoothly. Lagrange formalized what was probably a common intuition, but did not claim that this described time outside of the mathematics of classical mechanics. In mathematics, a number of explorations were made of higher dimensional and non-Euclidean spaces but these concerned objects with volumes or surfaces in various

numbers of dimensions and did not address time at all. It was in the physics of the very late nineteenth century, particularly the work of Henri Poincare and Hermann Minkowski, that time came to be treated formally as a fourth dimension (Phillips n.d.). These scientists were, of course, only expanding on mechanics, but the idea became widespread and people began to eagerly describe time as a fourth dimension. Science fiction writers seized on the idea of time as a perpendicular dimension, eager to explore the narrative possibilities afforded by characters using science to circumvent normal rules of storytelling . H.G. Wells' *The Time Machine*, published in 1895, made its claim explicit that duration is the fourth dimension. Since then, this assertion has been repeated endlessly in science fiction. Today we imagine that what the clock measures is our movement through a dimension called time. The movement of the sun, Earth, and gears of a clock are all expressions of this uniform movement along an invisible dimension.

In these histories of time's formation, we find that modern time is not one particular thing, did not follow from the interests of any one group, and was never put to use equally or universally. At all points, other kinds of time operated. None of these large-scale changes have fundamentally altered religious lunar calendars or the usefulness of an hourglass. What has happened is that a few kinds of time have been specially modified to reinforce each other. Many practices found a usefulness for the clock, created new practices of time dependent on the clock and nested within a time given by the calendar, gave the clock authority, and ran into some problems that were fixed with standards.

It's hard to know why, really and exactly, each of these changes happened. The recorded arguments for standardization focus on tiny gains in efficiency. In many cases, these don't seem very convincing. At their strongest, arguments for efficiency symbolized modernization, a rejection of the conservative, an appeal for simplicity, and the importance of unification. The case of Prussia is most clear. A respected Field Marshall endorsed the efficiency of the new time standard, making specific reference to its importance in mobilization for war and claiming that unification of time should follow from national unification. Ultimately it was not even his argument that was persuasive; it was his death. By adopting GMT, politicians paid their respects. Symbolic reasons carried the day.

In some ways, the standardization of times resembles the adoption of the metric system. The metric system took a century to find acceptance, starting with the French Revolution and becoming standard in Europe and Latin America by the late nineteenth century (Hallock and Wade 1906, 105–108). The reason for adopting a uniform system of weights and measures was clear enough: without one, cheating and mismeasurement in commercial transactions would remain common, generally at the expense of the more trusting party (81). The metric system, more than any other proposed standard, had the weight of years of scientific use behind it. A series of international conventions endorsed it as the best standard system, particularly for its decimal division (41-79). Metric made for simpler judging at international exhibitions of science and technology whereas local measurement systems made it hard to

compare different accomplishments. An advocacy organization eventually formed that agitated for policy changes endorsing the metric system (Cox 1958).

Metric had advocacy, scientific legitimacy, and a compelling commercial justification. But it also had clear disadvantages. Important barriers to adoption included the instability of regimes in European government in the nineteenth century (Hallock and Wade 1906, 81), lack of follow-through by governments in the form of secondary standards and offices maintaining standards (64), resistance from uneducated populations and their governments (94), and the substantial cost of retooling instruments and measuring devices for the new units of measurement (96-97).

Compared to metric, changing the time was relatively easy. Adopting the metric system involved real obstacles, yet these were overcome for the sake of the system's substantial benefits. Metrification had less precedent and happened in an international situation with considerably less stable governments than that of time standards. Governments and their people may have been more receptive to standard time after the successful adoption of metric and the means by which changes could be advocated and implemented were better developed as well. Changing the standard time of clocks seems to have been an altogether simpler affair. This possibility suggests that the reinforcing set of modern time now dominant is not held together by such powerful forces after all.

In the proliferation of public clocks, institutional use of time, standardization of Meridian time, tweaking of the length of a second, and addition of synchronization

to control accuracy, the alliance of several times (and the ease with which they can be mutually translated) has grown strong. Contradictions and inconsistencies have been lessened, as much as possible, by advocates, government policy, and changes of practices on the ground. The length of an hour, authority of the clock, the time of noon, time of noon in other cities, the date across the world, length of a second, accuracy of clocks to the second, and definition of what time measures have all changed to establish a harmonious set of practices that we now would simply call time. Institutionalized practices of time have found a rather stable alliance and produced practices with a great deal of inertia. In the United States, the nine to five workday is nearly unshakable, though exceptions have become common. Through scientific work, the currency of the clock's measurement has established it as a fundamental dimension of the universe. For projects working at the quick and precise scale of milliseconds or smaller, the alliance of practices allows work that it is hard to imagine doing another way.

Yet, by extending a form of time through alliance, contributing forces may also strain it and make it vulnerable. Any contradiction between allies threaten the reification. If the needs of capitalism no longer depend on blocks of time that measure regularized activity, a schedule of hours loses economic importance. If physics rejects the dimensional model of time, something quantum theory presents as a serious possibility, clock time becomes a useful convention lacking metaphysical significance. If a growing part of the population ceases to accept the clock as efficient or authoritative, the set may lose its anchoring reification that time is what the clock



measures. These weaknesses have already begun to call for contemporary investigations of changing time practices. It is not just that many practices have been changing, this has happened throughout history. The contemporary situation is that time has accumulated into a fairly consistent *thing*, which was open to critique in the middle of the twentieth century and is now beginning to appear as many things nestled closely together.

### **Time Continues to Change**

The shifting institutional agendas, political climate, economic arrangements, and local conditions that have given rise to the reinforcing set of dominant time practices have not settled. Although it is easy to speak of time-space compression, timeless time, and acceleration, no one narrative can fully account for all changing practices of time. In some ways, practices of time known simply as time have come unbundled, while in other ways they have become more powerful and efficient than ever. New technologies have their own kinds of time and the clock must fit into rather different practices. These technologies have afforded us different measures and means of living in time (Hassan 2007a). Mobile phones, fax machines, and email all provide different systems of access, presentness, sequence, event, scheduling, memory, and action that fracture the singular meaning of the clock (Hassan 2007b). Changes in the social context within which time is put to use have created openings for new practices of time to flourish. Flexible labor, outsourcing, time-shifting, regular long-distance communication, and automatic cooperation over great distances demonstrate the malleability of practices of time and call for changes in time. The most common

summary of the nature of these changes comes under the title of time-space compression. But is this summary apt?

Originally developed by David Harvey, the time-space compression thesis claims that social life has accelerated for economic reasons. This idea combines two tendencies. The first pattern is the in-built tendency of capitalism to increase profit by accelerating the turnover of capital (delay between investment and return) and decreasing input time (Castree 2009). This has happened in many ways, from labor to distribution to fashion. Second, the experienced reality is that faster means of transportation decrease the number of hours separating two places while increasing the amount of places that can be reached in a certain number of hours (Schivelbusch 1986, 35). Transportation and communication are the main examples of this change: the far away is close at hand, and this has increased the number of people, places, and things to which one has access. Things are being done faster than before and more is in reach than before. These changes are not just economic, for Harvey, because they drive social life as well (Harvey 1989, 203–204, 228, 254, 344). They express themselves in many specific situations: organizational changes following Fordism, the spinoff of corporate divisions, just-in-time delivery, improved communication technology and information flow, rationalization of distribution, electronic banking, credit cards, fashion in mass markets, the change in emphasis from products to services, disposability, learning to live with volatility, and the short average tenure of a CEO all contribute to the effect (284-287).

We are all enthusiastic to be validated in our feeling that life is too harried and stressful and this predisposition should give us pause. The history of slowness into the contemporary period remains unwritten. Urry's concept of glacial time is one small attempt to turn attention to slower time (Urry 2000, 157–160). Histories of technology tend to follow the brilliance of invention and newness, ignoring the dull persistence of technologies and practices that have not changed (Edgerton 2006). It is of course less interesting to write the history of things that have not changed, but the bias this produces is that our world seems to be changing when what changes are only some things in it. Acceleration should not be regarded as a general condition, but as a phenomenon observable in some cases and not others.

So long as we ask of time only whether it is getting faster or slower, we force discussion into the reductive terms of one time and ignore the opportunity for detailed analysis of other relations between times. The foundational evidence of experienced acceleration comes from accounts of the elite riding the European rail system, as in *The Railway Journey* (Schivelbusch 1986). Though the transformations they noticed were real, elites were more sensitive than most of the population to these changes. Their interpretation framed the issue in terms of progress: things are faster than they were and will become faster yet (Stein 2001, 119). Narratives of progress, self-congratulatory for the elite, omit the persistence of other practices and anticipate a structure of directed change, sometimes with a heavy hand.

Different practices of time proceed in various ways. The vision of accelerated blocks of activity is how the dominant set of times sees the world and how it often

treats it. Measurement is a key symbolic intermediary in dominant time practices, and, in practice, tends to be used as a parameter to be optimized. John Tomlinson observes that the world has opted for speed over slowness despite cultural critique because, in each case, speed seems more rational and practical (Tomlinson 2007, 39). This follows from the general usefulness of the clock as a means to define and increase efficiency. But this is still only a *perspective* and not a finished project of domination. Acceleration is not the *zeitgeist* or the essential story of the modern world.

Responses to acceleration (and exceptions from it) are as much a part of the world as acceleration itself. Watching TiVo is quite relaxed compared to the usually strict playback cycle; chat and text messages are slower than face to face or telephone conversations; the production of information technology happens in companies that emphasize temporal autonomy and non-quantifiable time (O'Carroll 2008), we do not sleep less now than in the eighteenth century (Ekirch 2005); a growing number now travel by bicycle and enjoy foods that are not prepared quickly; new movies extend old stories one episode at a time; high rates of literacy have encouraged a radical growth in the number taking the time to become authors (Pelli and Bigelow 2009). The point is not to show that the world is slowing, rather than speeding up; the point is that it is more interesting to discuss the details than to live in fear of a very broad generalization. If speed is inevitable and stressful, if it sacrifices reflection and creativity, we are left mournful and completely unprepared for the varied forms that time takes today.

But speed can have different meanings and different effects across practices of time. Word processors let one write faster because everything can always be changed later. This encourages experimentation, self awareness, and play. The perspective of scheduled blocks of activities does show the changes that it has been used to make. Other perspectives might summarize our situation differently. Has it been useful, for every time in every relation of usefulness, to increase speed as measured by the clock and calendar? Is there even just one way to make this measurement?

It is a totalizing claim that the world has gotten faster or time is busier now than before. Many things remain slow, despite the powerful translations by which some practices have reduced others to measures of speed. Still, these translations do not encompass or dominate the world, are not made uniformly, and may encourage transformations of very different kinds. Time are social practices, real both in their basis and consequences, that take many varied forms between different cities and farms, arise from machines or nothing but habit, change gradually or quite quickly, and may at some point fade from view.

### **Conclusion: Dominant Times and Undo**

Times have come and gone. When a technology fits into a practice of time and this practice fits into a usefulness, a time can become very powerful. When such a time relates to other times constructively, they can be mutually supportive, often at the expense of others. The legacy of such power remains in the authority with which the clock provides an idiom that interprets all other times. Those practices of city governance, market organization, domestic industriousness, workplace discipline,

railroad scheduling, colonial shipping, scientific and computer work at the millisecond level, and network communication that put time to use in powerful ways made the times they supported powerful. They did this by starting with a familiar model, such as the solar day, putting it to use in a new way, and eventually getting the new practice accepted as a standard time. It is a consequence of these practices, and the commonality between them, that, when we think of the clock, we take for granted that it is set correctly, geared reliably, displayed publicly, referred to regularly, and used to measure, schedule, and plan human activities.

This dominant set of practices of time is not entirely coherent, but its inefficiencies have not yet brought it to a stop. There are many persistent incongruities between times: the allocation of paid time off, alignment of working hours with our sleep rhythms, measurement by advertisers of how many people saw a TV ad that they may have skipped over, and sequencing of courses for students are highly imperfect. But these differences do not splinter society or ruin the Durkheimian collective rhythm. The ecosystem of times does not work because it is strictly organized; it lives on by being loose, allowing movement, and recovering from failure. It “operates by delays and omissions, fuzzy relationships and vague orientations, accelerations and flexibilities and it can avoid breakdowns only because it is not tightly but loosely coupled” (Giesen 2004). This looseness provides exceptions, wiggle room, and opportunities.

In the jumble of times, many specific situations can be understood in terms of a more basic relationship of translation or influence. Times organize temporality hand

over hand, often acting through the same objects or making different uses of the same temporal realities. Watching Project Runway, many times happen at once. In translation, times represent functions of temporality that are also already part of another time. Translation entails many more specific kinds of relation: most times happen not to translate one another at all (towns far from the railroad ignored railroad time), many actively fail to understand another (the factory disregards Saint Monday), some claim a universal significance and routinely ignore others (the dimensional model of time in mechanics has done quite well), and most refer to some other time as a model. Those times that are referred to most often are not more fundamental than other times, but they do set the standard for what counts as a time. They carry greater legitimacy, are more familiar, and encourage us to use a temporal vocabulary when describing them. Ironically, their value as a reference has generally come at the cost of their power as a practice. The solar day has become less important than the clock's representation of the solar day. The calendar has similarly gained power over the seasons and astronomical year. Practices, more than their referents, organize temporal influence. Times do not only interact by representing temporality, they also touch each other through the medium of their enactment: all times act on the same world. One time can bring another into its rhythm (the train and the shift schedule), regenerate the powers of those actors involved in another time (post-Taylorist workplaces for workers), or make the requirements of another time inaccessible (Saint Monday was forgotten).

Thus far, my entire argument has amounted to a social theory of time. It describes how the functions of temporality become caught up in a regular organization that is embodied in a practice. These practices form in contact with other times and relate to them in specific ways that are sometimes well summarized as translation and influence. Some of these times have formed a dominant set at the expense of others. We can hardly imagine, let alone accept, the time of care, the time of birthing and the time systems of other cultures as times. The authoritative translation overpowers these other times, to the point that we hesitate to recognize these others as times at all. And yet, despite the force of these practices, other times do not die off. In many cases, different times, though they have been poorly understood, have maintained themselves and found a usefulness within other practices of time where they are flourishing.

In a world rapidly shifting its diverse social infrastructures into relatively uniform computerized mechanisms, new practices are organizing temporality in powerful ways. We who live with these times know that something is different, though enthusiasm and fear have drowned out critical reflection. Being sensitive to the operations of temporality, it is clear that something is amiss when forward and backward are equal options, when a process can start or stop at any moment, or when the pace of events is not beholden to ritual but to the individual's own present. However, these practices do not resemble the clock, the calendar, or the second. They are not useful for measuring and increasing efficiency. Their widespread use has not required a redefinition of any aspect of the dominant set of times. Though they refer to others times, such as the clock, they do not represent those times naturalistically; these



times are artificial variations on older times and enact this unapologetically. These times emerge from the technical, itself a social product, and join historical forces to create enacted practices.

Though theory gives us instincts with which to understand actual cases, it is empirical matter that puts it in conversation with reality, and this evidence is always incomplete, misleading, and, in some ways, an exception rather than a representative. To pursue further the question of interactive times requires extending the theoretical into the realm of the empirical. How have new times formed, become common, related to other times, and had an impact?

Thus we move from general ideas of time and society to the lowly and oft-ignored undo command. The command is a conventional feature of software, and not a time just by itself. However, the technology joined a practice of time that found many uses and so became practiced widely. Its model is not the clock but the sequence of actions that a person using a computer has just inputted. In reference to this model time, undoing allows temporality to reverse the flow of local events, going backward to go forward and moving slowly to move quickly. In a world where the clock's authoritative translation of other practices into terms of efficiency dominated, undoing saved time for some by making a gentler time for many. It did this by establishing a new practice of time whose locus of control was not in the sun, the international standards defining when now is, or even in the clockwork mechanisms driving a practice; the time of undoing depends on local agents each making their own decisions about when to go forward and when to go back, how to use this power of reversal, and

how to go about the whole thing while also living within a number of other more powerful times.

Chapter 2 and 3, in part, are reprints of material as it appears in “Time as a Social Practice,” *Time and Society*. Forthcoming. The dissertation author was the primary investigator and author of this paper.

### References

- Adam, Barbara. 1995. *Timewatch: The Social Analysis of Time*. Cambridge, Mass: Polity Press.
- . 2004. *Time*. Malden MA: Polity.
- Bartky, Ian. 2007. *One Time Fits All: The Campaigns for Global Uniformity*. Stanford, CA: Stanford University Press.
- Bluedorn, Allen. 2002. *The Human Organization of Time: Temporal Realities and Experience*. Stanford, CA: Stanford Business Books.
- Braverman, Harry. 1975. *Labor and Monopoly Capital: The Degradation of Work in the Twentieth Century*. New York: Monthly Review Press.
- Bryson, Valerie. 2007. *Gender and the Politics of Time: Feminist Theory and Contemporary Debates*. Bristol: Polity.
- Castree, Noel. 2009. “The Spatio-temporality of Capitalism.” *Time & Society* 18 (1) (March 1): 26 –61. doi:10.1177/0961463X08099942.
- Constant, Edward W. 1978. “On the Diversity and Co-Evolution of Technological Multiples.” *Social Studies of Science* 8 (2) (May 1): 183 –210. doi:10.1177/030631277800800202.
- Coveney, Peter. 1991. *The Arrow of Time: A Voyage Through Science to Solve Time’s Greatest Mystery*. 1st American ed. New York: Fawcett Columbine.
- Cox, Edward Franklin. 1958. “The Metric System: A Quarter-Century of Acceptance (1851-1876).” *Osiris* 13 (January 1): 358–379.

- De Vries, Jan. 1993. "Between Consumption and the World of Goods." In *Consumption and the World of Goods*, edited by John Brewer, 85–132. New York: Routledge.
- Dohrn-van Rossum, Gerhard. 1996. *History of the Hour: Clocks and Modern Temporal Orders*. Chicago: University of Chicago Press.
- Ekirch, A. Roger. 2005. *At Day's Close: Night in Times Past*. New York: Norton.
- Foucault, Michel. 1977. *Discipline and Punish: The Birth of the Prison*. 1st American ed. New York: Pantheon Books.
- Giesen, Bernhard. 2004. "Noncontemporaneity, Asynchronicity and Divided Memories." *Time Society* 13 (1) (March 1): 27–40. doi:10.1177/0961463X04040741.
- Glennie, Paul, and Nigel Thrift. 2005. "Revolutions in the Times: Clocks and the Temporal Structures of Everyday Life." In *Geography and Revolution*, edited by David Livingstone, 160–198. Chicago: University of Chicago Press.
- . 2009. *Shaping the Day: A History of Timekeeping in England and Wales 1300-1800*. Oxford: Oxford University Press.
- Hallock, William, and Herbert Wade. 1906. *Outlines of Evolution of Weights and Measures and the Metric System*. New York: Macmillan.
- Harvey, David. 1989. *The Condition of Postmodernity: An Enquiry into the Origins of Cultural Change*. Cambridge Mass.: Blackwell.
- Hochschild, Arlie. 1997. *The Time Bind: When Work Becomes Home and Home Becomes Work*. 1st ed. New York: Metropolitan Books.
- Kramer, Jonathan. 1988. *The Time of Music: New Meanings, New Temporalities, New Listening Strategies*. New York, NY: Schirmer Books.
- Landes, David S. 1983. *Revolution in Time: Clocks and the Making of the Modern World*. Cambridge, Mass.: Belknap Press of Harvard University Press.
- Lefebvre, Henri. 2007. *Rhythmanalysis: Space, Time and Everyday Life*. London: Continuum.
- Lim, Bliss Cua. 2009. *Translating Time: Cinema, the Fantastic, and Temporal Critique*. Durham: Duke University Press.
- Mainzer, Klaus. 2002. *The Little Book of Time*. New York: Copernicus.

- May, Jon, and Nigel Thrift. 2001. *TimeSpace: Geographies of Temporality*. Critical Geographies 13. New York: Routledge.
- McCrossen, Alexis. 2007. "Conventions of Simultaneity." *Journal of Urban History* 33 (2) (January 1): 217–253. doi:10.1177/0096144206294738.
- McKendrick, Neil. 1961. "Josiah Wedgwood and Factory Discipline." *The Historical Journal* 4 (1) (January 1): 30–55.
- Mills, David. 2006. *Computer Network Time Synchronization: The Network Time Protocol*. Boca Raton FL: CRC/Taylor & Francis.
- Mirmalek, Zara. 2009. "Working Time on Mars." *KronoScope* 8 (2) (June): 159–178. doi:10.1163/156771508X444602.
- Moffit, Tom. 2002. "Origin of the Pink Slip." In *Henry Ford: Critical Evaluations in Business and Management*, edited by John Wood, 1:297. New York: Routledge.
- Moran, Chuk. 2010. "Playing with Game Time: Auto-Saves and Undoing Despite the 'Magic Circle'." *Fibreculture* (16) (July).  
<http://sixteen.fibreculturejournal.org/playing-with-game-time-auto-saves-and-undoing-despite-the-magic-circle/>.
- Murillo, Bartolomé Estebán. 1665. *Don Andrés de Andrade y La Cal*. Oil on canvas. The Metropolitan Museum of Art.
- Nielsen, L. 2006. *Vintage Cars*. New York: Crabtree Pub.
- Norwood, Stephen. 2002. *Strikebreaking and Intimidation Mercenaries and Masculinity in Twentieth-century America*. Chapel Hill: University of North Carolina Press.
- O'Carroll, Aileen. 2008. "Fuzzy Holes and Intangible Time." *Time & Society* 17 (2-3) (September 1): 179–193. doi:10.1177/0961463X08093421.
- O'Malley, Michael. 1996. *Keeping Watch: A History of American Time*. Washington: Smithsonian Institution Press.
- Paullin, Charles Oscar, and John Kirtland Wright. 1932. *Atlas of the Historical Geography of the United States*. Washington D.C.: Carnegie Institute of Washington and the American Geographical Society of New York.
- Pelli, Denis G., and Charles Bigelow. 2009. "A Writing Revolution." *Seed Magazine*, October 20. [http://seedmagazine.com/content/article/a\\_writing\\_revolution/](http://seedmagazine.com/content/article/a_writing_revolution/).

- Phillips, Stephen M. n.d. "A Short History of the Fourth Dimension." <http://www.smphillips.8m.com/a-short-history-of-the-fourth-dimension.html>.
- Pollard, Sidney. 1963. "Factory Discipline in the Industrial Revolution." *The Economic History Review* 16 (2). New Series (January 1): 254–271. doi:10.2307/2598639.
- Postill, John. 2002. "Clock and Calendar Time." *Time & Society* 11 (2-3): 251–270. doi:10.1177/0961463X02011002005.
- Rigby, Stephen. 2010. "Urban Population in Late Medieval England: The Evidence of the Lay Subsidies." *The Economic History Review* 63 (2) (May 1): 393–417. doi:10.1111/j.1468-0289.2009.00489.x.
- Schivelbusch, Wolfgang. 1986. *The Railway Journey: The Industrialization of Time and Space in the 19th Century*. Berkeley Calif.: University of California Press.
- Sherover, Charles. 2003. *Are We in Time?: And Other Essays on Time and Temporality*. Evanston Ill.: Northwestern University Press.
- Simpson, Lorenzo. 1995. *Technology, Time, and the Conversations of Modernity*. New York: Routledge.
- Smith, Humphry. 1982. "The Steady March of Atomic Time." *New Scientist*, February.
- Stein, Jeremy. 2001. "Reflections on Time, Time-Space Compression and Technology in the Nineteenth Century." In *TimeSpace: Geographies of Temporality*, edited by Jon May and Nigel Thrift, 106–119. Critical Geographies 13. New York: Routledge.
- Stephens, Carlene. 2002. *On Time: How America Has Learned to Live by the Clock*. Boston: Bulfinch Press.
- Tanaka, Stefan. 2004. *New Times in Modern Japan*. Princeton, N.J.: Princeton University Press.
- Thompson, E. P. 1967. "Time, Work-Discipline, and Industrial Capitalism." *Past & Present* 38 (1) (December 1): 56–97. doi:10.1093/past/38.1.56.
- Thrift, Nigel. 1981. "Owners' Time and Own Time: The Making of a Capitalist Time Consciousness, 1300-1880." In *Space and Time in Geography: Essays Dedicated to Torsten Hägerstrand*, edited by Torsten Hägerstrand and Allan Pred. Lund: CWK Gleerup.

- Tomlinson, John. 2007. *The Culture of Speed: The Coming of Immediacy*. Los Angeles: SAGE.
- Urry, John. 2000. *Sociology Beyond Societies: Mobilities for the Twenty-First Century*. London: Routledge.
- Weber, Max. 1905. *The Protestant Ethic and the Spirit of Capitalism*. Reprint. London: Routledge.
- Zerubavel, Eviatar. 1985. *The Seven Day Circle: The History and Meaning of the Week*. New York: Free Press.
- Žižek, Slavoj. 2001. *Enjoy Your Symptom!: Jacques Lacan in Hollywood and Out*. New York: Routledge.

#### IV. The Historical Undo Command

An undo command is a procedure in a piece of software that restores the state of a document to what it was just before the user's last action. Delete a sentence, then undo the deletion and the sentence is back. While the command has had many forms, today it negates almost any action whose effects *can* be undone. If you send an email or print a document, you can't undo the action because commands have already been passed on to other systems that cannot undo their actions: the printer puts ink on paper and the email server communicates with another server. The printer cannot remove the ink and return the page to its original tray and the email server cannot make other servers return or delete a message.<sup>1</sup> There are many ways to design the command at a technical level, but they all produce the same function: the user can take one tiny step back.

Sometimes it feels as if, by undoing, one is going back in time. It seems that, with an undo command at the ready, one is living in a time where reversal is always possible. This little time machine is my insurance policy, my protector, my super-power. Ctrl-Z.

This chapter shows that the historical emergence of an undo command was a response to how people used computers, what they could do with them, and what others wanted them to do with them. It established a general practice of time that was

---

<sup>1</sup> Can email be unsent? Yes, if all sender and receiver accounts are on the same system. When this is available, the feature is rarely used (Cabitza and Loregian 2008). Some email providers delete undelivered emails if they are known to be spam, but deleting delivered messages of any kind usually scares users.

repeated in the millions of cases where a person typed and clicked buttons, while looking at a screen, in order to get something done on a computer. When the user used software, an undo command made temporality function in an uncommon way: actions could be undone. This function was part of a new problematic wherein user input produced *actions* that changed the *state* of a frequently updated *document*. An undo command would not have made sense for a 1950s mainframe, would not have been possible for another technology (such as an engine or phonograph), and did not come into existence in payroll or wind tunnel simulations. Undo empowered *knowledge workers* at *workstations* to relax, experiment, explore, change their plans, and improve their products. We live in the aftermath of these transformations.

Understanding the historical context of the emergence of the undo command is important because inventions express and change the situations that create them. It is exciting to imagine that an invention comes to life, like Frankenstein's monster, and runs off changing the world forever. However, technology is slower and designers more conventional than that. As Deleuze puts it, technology is "social before it is technical;" "there is a human technology which exists before a material technology" (Deleuze 1999, 34). One does not design user-friendly interfaces without their context and likely use in mind. Innovators work with existing conventions to produce new technologies, which, when successful, will become conventions in turn. Buzz about the future of technology tends to guide invention, and most scientists expect that if they do not finish an invention quickly enough, someone will beat them to it (Merton 1961). The myth of inevitability is strong even though often incorrect (Moran 2013,



27–28, 214–215) Those making new technologies chase images of the future that they regard as inevitable and, in the process, make things that express their situation.

If the right people find a way to make a technology useful, it can enter into regular use and, if things work out, a technology may become a common convention, in turn expressing and changing the situation where it arose. Gunpowder and guns were well understood technologies in many places for almost a thousand years before an army gave every soldier a gun; the distance from marginal existence to widespread standardization can be very large.<sup>2</sup> The prison has existed in many societies, but took on greater significance and utility in disciplinary ones, becoming, more recently, a response to the problems of suburban anxiety and reelection.

When a technology does become a standard, as undo did, people wonder who invented it first. This question brings various tinkerers' projects to light and retrospectively groups them together as variations on the same idea. Now that undoing exists, we see creativity's redundancy. We are noticing a pattern. Historical perspective here does not just reveal the pastness of the past or tell the history of the present; it shows the old functions that sustain the inertia of the present and the designs that structure present potentials for change.<sup>3</sup> The situation that made undo

---

<sup>2</sup> What is the distance between gunpowder and regiments of rifleman? To equip an army with guns requires, at minimum, a constant supply of metal, highly developed and reasonably common skill at smithing, rifling to reduce friendly fire, training for all soldiers and skill on their part, a supply line for ammunition, ways to prevent theft of the weapons, a large scale production operation, and a General willing to experiment.

<sup>3</sup> A history of the present identifies the historically specific problematic of a practice and draws connections between current and past forms of that practice in order to extract from the present an alternative that allows a departure from that present (Roth

popular is not so different from our own and that is why we relate unrelated toy projects together as the history of the command.

While undo developed in specific conditions, it is not reducible to the social forces animating it. Both social and technological actors had some determining force. The undo command came to life, but from its first implementation, was a material actor. It acted with, and against, those who made it, and their best laid plans. In some cases, a sophisticated undo command failed. In others, it was of special use to expert users or was an incentive to consumers on the fence about which of two products was worth their money. To say that it was material is to say its reality exceeded all representations that could be made of it. It did things that could not be foreseen and were not commanded by its own powers.

To understand the command as a material actor suggests some of its unpredictable development and application. But the command also had a *systematic effect* that can be understood best in the idiom of time. The command formed a social practice of time by linking together a number of temporal terms into a practice that, while initially unique, eventually became repeated across almost all software being made. It linked together the inevitability of user error and the stillness of the computer's idle cycles into a sequence of actions stored in temporary memory accessible and ready to be undone. In an editing program, the future of a document exists in the potentials established by the set of commands that the system recognizes;

---

1981, 43). In this vein, what matters is that undoing made time a generic approach to error correction, when others might have prevailed.

undoing was that command which would advance the state by taking it back a step. This changed the meaning of that future of possible commands, making them all interesting experiments, rather than lasting commitments. Although different undo commands work in different ways, they all reverse actions by reversing their effects. No undo can change the fact that, at some point, a certain command *was entered*. What they can do is change the state of a document and reverse one flow of actions within a very particular context.

This time occurs in relation to others. Undoing happens on the clock or after hours, on one day of the week or another, and in conversation with those other times of anxiety, belonging, craft, and so on. Historically, it was in relation to these other times that the command became a practice that we might today recognize as a time. We refer to this interactive time implicitly when thinking of how things go when one works on a computer and how temporality is organized in those spaces of word processors and video editors, where the rules of other times are in abeyance. The *interactive computer*, running real time programs by idling away extra cycles and only making updates upon instruction from the user, took on (in historically specific ways) an *interactive time*. Undo was only one technology providing a time subject to the manipulation of a proximate human actor known as the user.

The case of undo is an important form of interactive time for a few reasons. First, the undo function amounts to a radical denial of common sense ideas about time. In folk wisdom, we can slow down ageing and decay or record ephemera in fixed forms, but we cannot take back death, the expenditure of energy, or a nasty remark

(Adam 1995, 18–19). With undoing, this rule does not apply. The user *can* take back a poor turn of phrase, a premature ending, or an unlikely experiment. One might expect this to cause problems. The weight of tradition and dominance of the reinforcing set of time practices has a clear exception. Yet it does not make so much as a ripple, so long as undo is not considered as a time but as only a computer command. Second, undo is one of the earliest forms of interactive time. Before undo, there had been interactive computing, but very few systems offered users a variety of commands that would instantly alter documents in potentially harmful ways. Implemented first in word processing and programming interfaces, undo preceded media playback and other work and interactive consumer applications by a few years. By leading the way, undo gave people a sense of what time on computers could be like, and how new programs ought to work. It worked as a prototype for other times, for their architects and their users. Third, the history of the undo command touches on most of the central moments in the rise of interactive time practices generally, such as real time computing, knowledge workers on the computer, the graphical user interface, the study of human-computer interaction, and the development of user-friendly software. Despite the breakneck pace of change in information technology, many practices around computers can still be linked quite directly to these same moments.

This chapter explores the historical roots of undoing as a command on software systems developed first in 1969. The command emerged on real time interactive computers as a way to empower knowledge workers by automation. Although computing had before not been very interactive, many institutions found

ways to put computers to work as part of the daily operations of their workers. In these interactive systems, the operator's actions were usually defined quite specifically, and therefore so was error correction. A program for inputting bank transactions provided operators with a way to void a transaction; a spreadsheet allowed cells and formulas to be modified more than once. In a few academic projects, the role of the user of an interactive system became broad enough that many kinds of mistakes could be made, and they were not all as easy to correct as they were to make. Undo was invented at least three separate times within a two year period before it caught on with anyone. It caught on at the right place at the right time (Xerox PARC in the 1970s), and came to Apple's operating system and to Microsoft Word. The command slowly replaced a variety of other related commands that counteracted the effects of specific actions.

The gendered nature of these transformations is worth noting. Undo was one example of office automation displacing female labor and extending the power of male workers. In many cases, office automation proletarianized clerical work by lowering working conditions, reducing the role of employees' judgment, and closing chances at upward mobility (Glenn and Feldberg 1977). At the same time, economic growth created skilled jobs that were often occupied by women, though ultimately at lower pay and in worse positions than men for the same types of work (P. Kraft 1987; P. N. Edwards 1990). Into the 1980s, those using interactive computers in creative ways would mostly be workers in science and engineering. This group was then—and still is now—composed primarily of men (Mather 2007). Software that would augment the powers of the individual was therefore designed by and for men and

automated away much work previously done by women for pay. This does not represent a willful attack on women so much as a series of *enthusiastic intensifications of the powers of smart men*, first by the hiring of women as office workers to augment the powers of male thinkers and then, later, by replacing these women with computers. By the late 1960s, the power of men to change the world through innovation (rather than virtuous action or physical strength) was well established (Oldenziel 1999). In undo's genesis and rise to banality, this enthusiasm was always palpable. By the end of its expansion, however, the undo command became part of the work done in many occupations, including feminized jobs such as that of secretary and teacher. While women workers were initially a more or less deliberate target for those streamlining and intensifying knowledge work, the long term consequences for women are less clear. Like undo itself, software became a generic approach to working with ideas.

Undo became the standard way to negate the effects of an action and restore a previous state. It problematized the time of user action in terms of series and reversal. This new problematic gained traction during the 1980s thanks to prevailing dynamics in the commercial software market that selected for it. It was a nifty new feature, a way to make programs easier to use, allowed designers to make more complex programs, and became an expectation in just a few years. Due to its inclusion in Apple's user interface guidelines and the majority of word processors (and other applications), IBM and Microsoft added the command to their interface standards specifications. Dominant throughout software for personal computers, the function has been surprisingly uncommon on web and mobile platforms. Yet its legacy remains in

the expectation that anything you do on the computer, you should be able to undo. We intuit now, whether we would say it or not, that the time that passes in the computer's operations should be, to the extent possible, subject to the control of the user. How did interactive time become so normal?

### **Computing into the 1960s**

The scene where undo first appeared represented a radical (though not wholly original) departure from what computing had meant before. Starting in the 1940s, computers were high tech gizmos that could take on a surprising range of jobs, usually replacing human labor. They coordinated large amounts of data: to give a computer a new job required changing only what the data would represent, and programming a new set of operations to be performed on it. This approach could work whether the data was profiles of colleges for prospective students, noises that engines made when in need of repair, names and demographic details for mailing lists, or the motion of stars and planets. Decades later it would become hard to imagine anything could not be rendered as data. However, the extraction from human labor of something that would now be called electronic data processing still depended on human labor; computers displaced the job of thousands (mostly women) to a few engineers (mostly men) clustered around mainframes running batch operations, usually on punch cards and magnetic tape (Haigh 2001).

Computing machines distinguished themselves from the many kinds of high technology equipment of the mid-twentieth century by the variety of contributions they could make to different institutions that could afford them. The 1940s and 1950s

used computers for basic research in science, weapons simulations, bomber navigation and targeting, weather forecasting, wind tunnel experiments, control of traffic lights and oil refineries, and payroll.

In most cases, computers explicitly replaced human labor. In a sense, this was all there was to replace. A good deal of this labor was female: women pushing numbers through formulas at desks, women programming mainframes (Light 1999), women doing clerical work (Morgall and Vedel 1985), and women doing bookkeeping at banks (Fisher and McKenney 1993). The 1957 film *The Desk Set* dramatized these fears: an aloof systems man designs a computer to work the reference library for a large company; the librarians (all women) fear they will lose their jobs, but in the end the machine complements their work and everyone is happy. Although a dominant image into the 1960s was of the computer as a brain (“Science: The Thinking Machine” 1950; Friedman 2005, 23,66), the computer had no mind. Its faculties really amounted only to brute force and simple tricks. With these powers, a computer (often with a feminine name) could do in a few days what a human with a calculator could do in a few years. It was the cadre of engineers surrounding the computer that directed this “intelligence” and made it useful. Together, engineers and computers could take on an increasing number of jobs.

Although much has been made of the military’s role in developing computers, major funders encouraged computers’ use for large-scale data-intensive applications of many kinds. Nuclear weapons testing, targeting, flight controls, rocket science, and early-warning air defense were important early applications. However, most



computers, even when developed under government contracts, made little direct contribution to violence or its intensification (Ceruzzi 2003, 7–8). UNIVAC, one of the most important early computers, was first designed for the 1950 U.S. Census; when it was completed too late to be used there, the Air Force used it for logistics (e.g. the distribution of goods and scheduling of transportation) and the US Atomic Energy Commission ran simulations of the effects of nuclear explosions, rather than detonating real devices and creating fallout.<sup>4</sup> General Electric's 1954 UNIVAC purchase (one of the first computers purchased by a corporation) had the computer working on payroll, scheduling, inventory control, billing, and accounting; the computer soon took on other roles in market forecasts and production process design (Ceruzzi 2003, 32–33)

In the public imagination, computers were brains, and the risk was that they would turn evil, as did the NOVAC (a takeoff on UNIVAC) in the 1954 science fiction film *Gog*. The reality was much less exciting. Computers were not crazed dictators; they were finicky, data-obsessed whiz kids. They were controlling an increased range of processes (door locks, radio communications, thermostats, and automatic controls of all kinds). But these functions did not amount to extensions of a single intelligent master. The real danger of computers was a far cry from the simplistic personification of computers as omnipotent masterminds: computers have prioritized data collection for processing and encouraged management practices that

---

<sup>4</sup> Radioactive fallout was just being discovered at this time. Rendering explosions as data produced less pollution than real tests, which is quite a different benefit than automation as a savings on labor costs.

regard data representations as reality. They have had a substantial but difficult-to-recognize influence on every operation they became a part of, and they have become part of almost everything. What they have *not* done is take over the world with robots. However, in 1954 it was far from clear what dangers would be real and which would play out only on the silver screen. Certainly computers were strange and, even as they began to influence everyday life, they remained hidden in basements and backrooms, out of site.

Throughout the 1960s, businesses and governments of all kinds put computers to work in pivotal roles (Ceruzzi 2003, 110). DMVs, police departments, the Social Security Administration, US Post Office, and IRS all switched to computer systems to hold, receive, and check records as well as to direct their current activity. The police could radio in and run a license plate number, the Post Office could sort mail with typewritten addresses automatically, and the IRS adopted a system that would help process the records of the many new tax payers produced by tax reforms of World War II (Ceruzzi 2003, 119)—partly to intimidate people into honest reporting (“The U.S. Taxpayer: Due, Blue, and 97% Pure” 1962). Government economists and policy analysts, churches, rail yards, realtors, courthouses, libraries, horse breeders, and makers of prepared foods found uses for computers by the end of the decade.

Many of these systems responded to dire circumstances. In banking, the amount of data to process was exploding. From 1943 to 1952, the number of checks written each year doubled from four billion to eight billion; analysts predicted that by 1955, each year would see one billion more checks written than the previous year. The

stress of handling this data already forced banks to close early and hire more personnel (mostly young women) to a job with a 100% attrition rate. The banks needed help. Bank of America, a large bank expanding apace with California's growing population, contracted with Stanford Research Institute and General Electric to devise a system to automate check processing (Fisher and McKenney 1993). In office work, consultants warned of a similar problem. US businesses were adding 4000 pages of documents per employee each year, on average. A 1967 IBM promotional video directed by Jim Henson called this the "paperwork explosion" (Henson 1967). Heightened aspirations of often college-educated workers caused a major problem with attrition, and therefore training. At the same time, the cost of electronics and communications lines had been falling consistently. Automating office work, especially supervisory and management work, would also improve employee morale and result in benefits passed down to customers (Fronk 1980). Computers were called on to allow existing businesses to grow their operations to a scale that they were not otherwise prepared to handle.

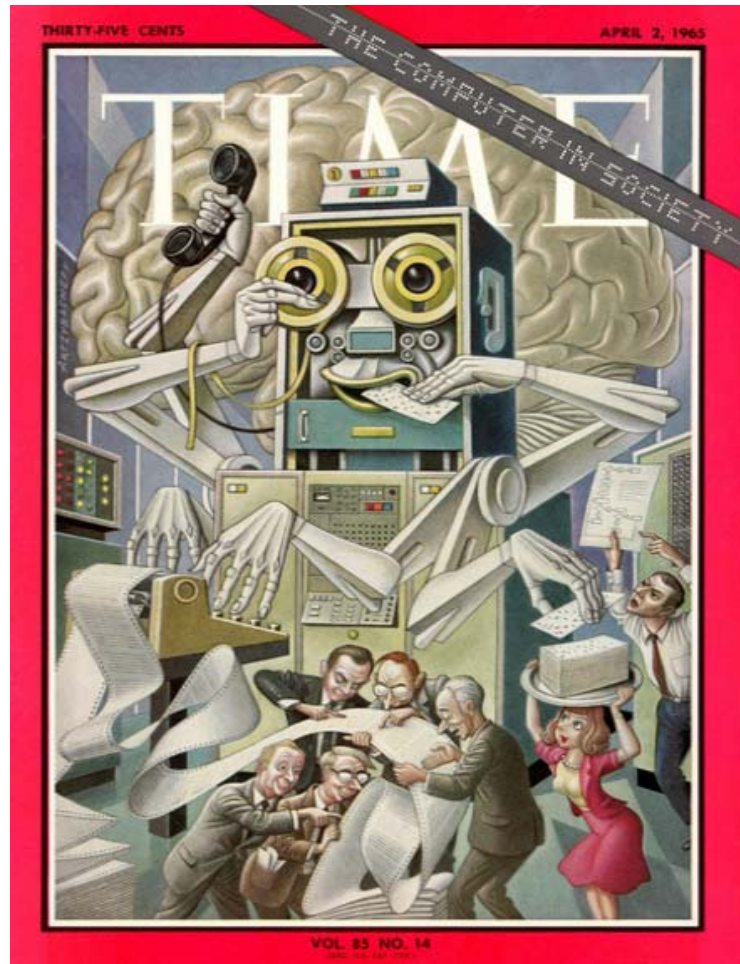


Figure 4.1 A 1965 cover from Time Magazine shows the computer as brain, keeping the madness of the office under control for men in suits.

Computers, during this period, were replacements for human power, rather than extensions of it. Any task had to be fed into a computer and then executed, with an output only received in the case of an error or a final product. This mode of use, called batch processing, centered around expert users; those who had been downsized would never have come into contact with the computer that now did their jobs. It was not just a gripe of the unemployed that automation took jobs, it was the point of automation. A cover story in Time magazine on “the Cybernated Generation”

consoled readers that every major technological advance had caused temporary unemployment, but in the long run, change was always for the better. In theory, new jobs would be created in other sectors, often providing services for a population whose basic needs were met by automated production and distribution (Bolz 1969). Yet, it seemed clear that those who had planned to make a living at a job that was now done by a computer were simply out of luck (“Technology: The Cybernated Generation” 1965). What computers would do was create an automated environment, for the benefit of consumers and the corporations that thrived on them. Humans would be displaced, rather than altered.

There were a few cases, however, where batch processing was clearly insufficient. In these places real time computing allowed computers into time-sensitive roles, generally replacing operators. The first such case was Semi-Automatic Ground Environment (SAGE), the military’s early warning system. The idea was simple: Soviet bombers could reach American targets very quickly and only a machine-coordinated response could detect them and deploy fighters fast enough. SAGE ran on the largest physical computer ever built and was obsolete by the time it was finished in 1959. However, the project educated hundreds who took their experiences with SAGE out to the computer industry (Campbell-Kelly 2003, 36–41). A second case of real time computing was airline reservations. Not only must each ticket reserve a seat on the flight, but agents across the country needed up-to-date information on prices and availability. Sabre, for American Airlines, was the first announced project, but by 1960 four others were completed for smaller airlines (“A Survey of Airline

Reservation Systems” 1962). A third case of real time computing was to track stock prices. A number of systems, starting in 1960, used computers to improve on ticker tape, giving instant access to opening, high, low, and current prices (Phister 1989).

In the transition from batch processing to real time computing, computers switched from a role in one time practice to another. In batch processing, computers saved on billable man-hours. Running the computer was expensive, but the hours it ran were worth the money if they replaced hours that many humans would have worked instead. The computer was invisible in the world of day to day operations and its principal function was data processing. It was important to those few engineers clustered around it and to those above who saw its contribution to total performance over measured units of duration. With real time computing, however, the computer became part of the active operations of employees. This embedded it in practices of time more familiar to employees. Reliability, performance, and human interfaces became important because traders needed updates on the double, travel agents needed tickets, and military radar stations needed to know what other stations were reporting. With real time, what the computer did in terms of man hours depended on what it could do at the fingertips of actual employees. Real time interactive computing had to be fast enough for humans and efficient enough to cover its own hourly rate. It would soon be discovered that it would be able to accomplish much more if, rather than going fast or sparing little time, it could match the varying pace of its human user, whether operator or customer.

Real time computing had the potential to displace human labor further, but it also opened the door to a very new perspective on computers. Real time computing turned archives (that were analyzed retrospectively) into available resources for day-to-day operations. Information could be combined dynamically, passed along to agents in other cities, and modified immediately. Memory became available instantly, combinable, and usable for planning future operations and executing corrections. What this meant for organizations that might adopt real time systems was the potential to replace human operators with more efficient, errorless, strictly regulated, and easily monitored systems (Head 1963).

There was, however, still a human laborer necessary for even a real time system that did away with operators as best it could. The elimination of those jobs, and of some responsibilities for others, required the addition of an “army of systems analysts, programmers, and software specialists” (Ceruzzi 2003, 9). This new laborer (usually male) worked very near a computer that responded in real time, received input from a lightpen or keyboard, and could be used for work outside of its job description. Among those “sitting at a computer four or five hours a day” was J.C.R. Licklider, the psychologist who became director of the major government funding source for computer research (Licklider 1988, 30). Licklider had found, studying his own habits, that 85% of the time he would have said he spent thinking was instead spent getting himself in the position to do the actual thinking – plotting graphs, hunting down information, interchanging data formats, and teaching others to do these same things (Licklider 1960). The computer, he reasoned, could someday boost

productivity of those in his position by doing all the busy work and letting its user focus on contemplating deeply. Licklider found intellectual camaraderie in Cambridge, at Harvard and then MIT, developing an enthusiasm for interactive computing and memory sharing (Licklider 1988, 17,25). He became convinced that artificial intelligence systems would someday cut out human users altogether, automating operators and perhaps even taking over maintenance and programming (30), but until then, some body would remain hunched over the keyboard. Immediate research should endeavor to augment the computer's human symbiote.

In 1962, the Advanced Research Project Agency (ARPA) hired Licklider to run the Information Processing Techniques Office (IPTO). President Eisenhower established ARPA in February of 1958, six months after the Soviet Union launched Sputnik. By 1962, the agency's primary focus was on ballistic missile defense and nuclear test detection, though it sustained many other projects at this time, including fuel cell research (Ruina 1989). Although Licklider was hired for work on behavioral sciences, he ended up making more lasting contributions with a budget of about 12 million dollars as director of IPTO.<sup>5</sup> Because IPTO was a tiny part of ARPA and word around the Pentagon was that Licklider would talk your ear off, IPTO received little

---

<sup>5</sup> Why did Licklider succeed in making computer science important and not behavior science? Work in behavioral sciences was more difficult politically because projects were easier to ridicule; if a congressperson could make a bit of research sound silly, it would make trouble for ARPA (Ruina 1989, 2–3). Because computing research was harder to understand and make jokes about, it was a safer investment. Further, money invested in computing produced a network effect, with resources and conventions shared quickly and widely. Behavioral science projects did not see an advantage in sharing space or resources, and thus they failed to build a community in the way those working on computers did (Licklider 1988, 26–27).



oversight—a case of benign neglect (Licklider 1988, 27, 42). Licklider directed for only two years, but during this time he set the agenda for the organization and encouraged (i.e. funded) work on interactive, real time computing, putting special emphasis on time-sharing as a means to that end. This had a lasting influence on computing history. His choice of Ivan Sutherland as a successor kept the vision of interactive computing alive through the 1960s, even as the actual deployment of computers in government and industry was still trying to replace humans with computers rather than join them (Kita 2003).

### **Brown's Hypertext Project**

If the 1960s had been the decade when computers were normalized into institutions as replacements for human labor, Brown University was no exception. Publications through the Association for Computing Machinery (the ACM is the world's largest learned society for computing) from Brown up until 1969 addressed algorithms for division, numerical integration, precision calculations, error bounds for linear equations, and compact data structures for line drawings. In sociology, James Sakoda was exploring statistical applications of computers in sociology, such as family building and population dynamics (Potter and Sakoda 1966). In linguistics, a database of real life uses of the English language offered new computational approaches to the study of language, as it was spoken, rather than as it is structured grammatically (Kučera and Francis 1967). The campus mainframe primarily served scientists, mathematicians, and administrators who ran analyses of complex data sets,

or tested the limits of what was possible with a computer. Interactive computing was not on the agenda.

In 1965, Brown's Department of Applied Mathematics hired Andy van Dam to work on computer graphics. Having just completed his PhD building graphics on top of an associative memory system (a simulation of neural connections), van Dam worked on the enormous, and very expensive, IBM 2250 monitor. The usable display was twelve inches on a side and showed vector graphics.<sup>6</sup> Van Dam got tenure in just three years. In 1967, he met up with an old friend from college and agreed to set up a side project at Brown making an editor for non-linear, associative texts. Using the 2250 monitor, and working on Brown's IBM System/360, the two began work on the Hypertext Editing System (HES). That friend was Ted Nelson (Andy van Dam and Simpson 2011).

---

<sup>6</sup> Contemporary computers use a raster display made up of lots of tiny pixels, out of which lines and volumes can be approximated. A vector display instead makes images by drawing lines, which is one reason for the association of sans serif fonts and simple line drawings with computers.

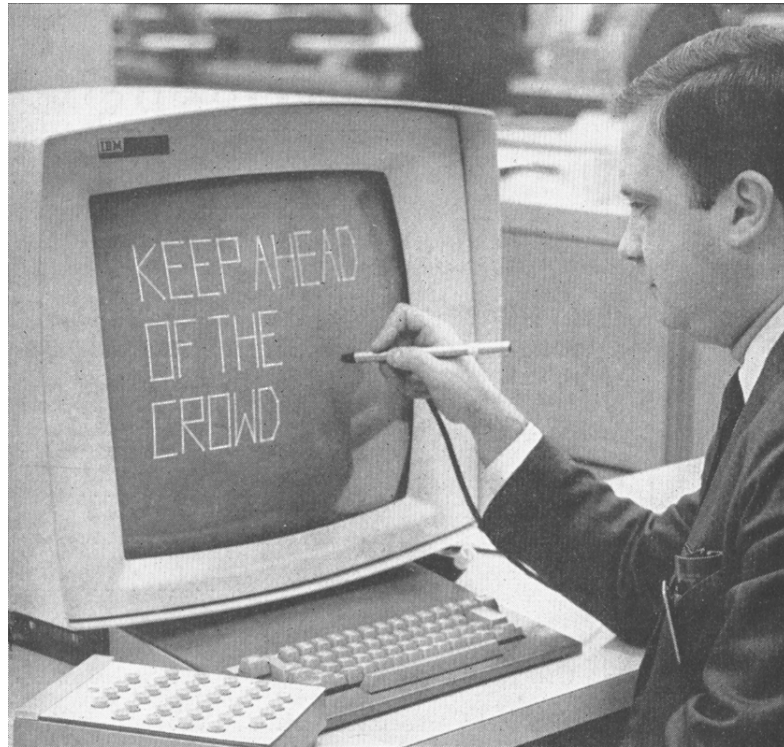


Figure 4.2 IBM 2250 monitor, from a Sikorsky Aircraft advertisement in *Communications of the ACM*, Vol.11 No.11, November 1968.

Ted Nelson coined the phrase “hypertext,” first publishing it two years before joining forces with Andy (T. H. Nelson 1965). The concept has since become associated with a range of meanings, from html to CD encyclopedias, but generally describes a system of interconnected texts and media that allow markup and navigation in a relatively free way. The concept has often been traced back to Vannevar Bush’s Memex. The Memex is a hypothetical system for building traces of connection between microfiche records as a way to automate the mental work of a researcher following leads for a project (Bush 1945). Bush was a senior government bureaucrat for the sciences, who may have been less an inspiration than a reference point helping Nelson explain his own ideas (T. Nelson 2010). Nelson describes his

vision of hypertext as a way to reproduce Russian author Leo Tolstoy's editing system on a computer. Tolstoy, Nelson states, used to dictate a draft of his work to two daughters. He would save one version and "cut up the other copy and rearrange it, adding material and deleting, and eventually pasting the pieces and new notes into their new sequence" (34). Hypertext would hold all the bits of material together, allowing paths of connection be authored freely. Though hypertext can be seen as an idea passed along through an intellectual history, the links of influence are weaker than they appear.<sup>7</sup> Hypertext might instead be understood as a name applied to a variety of systems that link together bits of text of media in theoretically unlimited configurations.

At Brown, Nelson's creative efforts to imagine hypertext served a different function. Andy van Dam, always preferring to do two things at once, was happy to experiment with the possibilities of literature, but he also needed a better tool for writing. His whole group was "constantly writing proposals and course materials and papers," with nothing better than a card editor or line editor. Such text editors centered on blocks or lines of text, rather than sentences, pages, or paragraphs, and were hard to use to write the kinds of documents Andy wanted. The Hypertext Editing System, designed by Nelson, van Dam, Brown freshman Steven Carmody, and others was both an experiment in hypertext and a practical tool for word processing (Andy van Dam

---

<sup>7</sup> Tim Berners-Lee, who designed the HTML (HyperText Markup Language) and HTTP (HyperText Transfer Protocol) specifications in the early 1990s for the web, was not directly aware of Ted Nelson's work at the time even though Nelson coined the term hypertext (Berners-Lee 1999:5; personal communication, November 15, 2011).

and Simpson 2011).<sup>8</sup> A working version of HES was finished in 1968, running on the /360 and displayed on the IBM 2250. The system aimed to

provide the user with unrestricted "spatial" options, and not to bother him with arbitrary concerns that have no meaning in terms of the work being performed. The entities he would be concerned with would correspond to the content of conventional writing: words, sentences, paragraphs, sections, and also non-structured, arbitrary fragments of text to be rearranged and spliced into appropriate combinations. He would not encounter line numbers, page numbers or footnote numbers, all of which are extraneous artifacts of conventional writing "hardware", that is, paper. His activities, too, would correspond to the operations ordinarily performed upon text by writers and editors. He would be able to perform manipulations directly upon pieces of text: correcting, moving, linking and copying, etc. Such actions would correspond directly to the "scissors-and-paste" operations of rearranging manuscripts. In addition, the writer would be able to do various other things which are usually very costly in time and/or money, or downright impossible: file previous drafts, spin off alternative versions for separate tinkering, and communicate between separate versions, lifting or replacing sections as desired. (Carmody et al. 1969, 299–300).

The conflicting priorities of paper-writing and creating a full hypertext system (with transclusion, two-way links, and transcopyright) became an interpersonal conflict as

---

<sup>8</sup> Andy van Dam explained to me, "I built Brown Computer Science on the notion that even freshman, after they've had one rigorous course under their belt can make serious contributions and so you'll find all the early papers the dominant contributors are undergraduates. FRESS was largely designed by undergraduates, Bob Wallace was an undergraduate - on drugs! [...] so the way I work with the kids I never did any of my own implementation because I'm not as fast as they are and I don't have the time, the luxury to do what they do, which is to sit down with their coke and their pizza and go for 8 hours or 10 hours or 14 hours of design and implementation and debugging marathons" (Andy van Dam and Simpson 2011).

well. Van Dam, busy on other projects, had little time or interest in pursuing Nelson's truly radical design program.<sup>9</sup>

In HES, real time interactive computing, made possible by late-night mainframe use and a display unit almost no one could afford (including most other researchers working on graphics), empowered knowledge workers by letting them do alone what before many did together. The pattern was not new: a group of programmers using a mainframe, working with text for programming and other kinds of writing, has almost always designed a word processing application to meet their needs (Haigh 2006, 13–15). HES allowed hypertext facilities which may have been useful, but it also allowed something more basic that was not to be popularized for another decade: full-screen video word-processing. In a historical moment where computers were tools used by large organizations to crunch through analyses of large data sets, HES let the uninitiated do things they had never done with text before. Van Dam recalls, “once you spent fifteen minutes with HES, you were hooked! Totally addictive, because of the rapidity with which you could author and edit, and then push a button and it will come off the line printer, I mean that was friggin magic.” At the time, most professionals producing documents (van Dam calls them knowledge workers) used a different intermediary: women. “You gave your work to your girl (always a girl) as in “my girl will call your girl to set up the luncheon date” – that was

---

<sup>9</sup> Nelson's main idea is that the computer is a media machine that can do anything we want it to do. It should be a system linking together everyone's notes, videos, artwork, books, etc. into constellations that are themselves interconnected. Nelson has elaborated on this vision, and many other related systems, in a number of his publications, including the flip side of *Computer Lib*, a book called *Dream Machines*.

absolutely the mindset.” With HES, the radical new possibility was to cut out excess women from a knowledge workers work, “you don't want to give it to some third party that doesn't know what you're thinking, and you don't want to give them the finished work product in handwriting; you want to conceptualize, ideate at the computer and use it as your brainstorming tool” (Andy van Dam and Simpson 2011).

Thomas Edison claimed that genius is one percent inspiration and ninety-nine percent perspiration, but a more common attitude (common to van Dam, Nelson, Licklider, and Henson’s “Paperwork Explosion” ad for IBM) was that the perspiration should be minimized, to make more room for inspiration. Henson’s video repeated the slogan like a mantra: “machines should work; people should think.” Peter Drucker, credited with the idea of a knowledge worker, makes this perspective clear in his definition of knowledge not as “facts (whatever this slippery metaphysical term might mean)” but “a new vision, a new pattern, a new attitude” (Drucker 1959, 26). The knowledge worker uses theories and concepts to process information and is improved by being changed by this information (Kidd 1994). The thinking was that, if computers could help downsize women from the office, they could also restrict their apparently unhelpful role in the production of serious thought (Losh 2009, 316–321).

Having seen HES, and copied features freely<sup>10</sup> from Doug Engelbart’s NLS (a related system whose 1967 demo sent shockwaves through computing), Andy van Dam began work on a follow-up system called File Retrieval and Editing SyStem

---

<sup>10</sup> Until the mid-1980s, copyright was *not* the norm, most software was regarded as free, and copying features was simply the fastest way to make a better design (Campbell-Kelly 2003, 107).

(FRESS). All through its development, HES had been a side-project and a benign abuse of IBM's display screen. Van Dam had focused his efforts on graphics research and gotten tenure for it. But with HES fully operational, van Dam could bring the side project to light. He showed his sponsor from IBM, who was impressed, and the group got meetings at Time-Life and the New York Times. Nelson demonstrated the system and reactions were very positive, though the idea of people using computers to write and layout the news sounded a bit like science fiction to the journalists. The system seemed like a part of the far off future that had arrived too early. Ultimately both companies began using computer systems for these purposes within the next decade. Those in van Dam's group discussed starting a business around HES. Ultimately, negotiations broke down due to a lack of trust in their primary investor and disagreements about how the work of each member would be compensated. Nelson left Brown,<sup>11</sup> and van Dam developed FRESS slowly with his undergraduates. The new system would allow collaborative work, have a more robust set of commands, and not depend on the IBM display.

With less hurry to get the system finished, more confidence in the project, a larger user-base and more time during which features could slowly be added, FRESS, HES's successor, was the first editor to include an undo command. The year was 1969. The feature was one of many novelties; some new features were overkill and

---

<sup>11</sup> Nelson's experience at Brown remains unclear, is recalled differently by van Dam and himself, and is the subject of a chapter that he ultimately edited out of his recent autobiography.



almost never used, such as locking sections of a document, while others had always been planned, but had simply not made it into HES, such as bidirectional linking.

An ability to undo was built into the system from the beginning. Van Dam explains the command as a logical extension of the backups and checkpoints common to computer culture. Since the 1950s, it had been common practice to make backups of projects and entire mainframe memories, because glitches and mistakes are inevitable. In developing a project, whenever things were going to be changed significantly, a backup would be made (called a checkpoint), to which the team could retreat if new developments made a mess of things. If this concept already existed at a larger scale, van Dam was driving for “the microscopic version of that, where, for the “Oh Shit!” moments, particularly when you delete something, you can get it back (and I have oh shit moments all the time)” (Andy van Dam and Simpson 2011). Regret, then, offered one justification for an undo function. Nelson’s original paper on hypertext mentioned an undo command that “would permit the user to retrace chronologically everything he does on the system” for safety’s sake (T. H. Nelson 1965, 94–95). Thinking of FRESS in terms of hypertext, the undo command mimics the back button (now well known on web browsers) that Nelson had successfully introduced in HES (as Return). Hypertext set out a whole world of pathways, leading from a present view to any number of proximate and distant connections; “hypertext positions one in a continuing present in which something is about to happen” (Landow 1990, 51). Proceeding along these links would eventually take the user on a tangent from which he or she would

want to return. What the back button could do for navigation, undoing could do for authoring and editing.

Each of these four explanations came to the same conclusion. First, if files could be backed up and restored every week, day or hour, they could be stored every second. If the scope of this backup could be an entire drive, a set of folders, or a single file, it could also be just a part of a file. Undo could recover the smallest possible difference made in a document, which was a single user edit. Van Dam's second, related, concept of a mechanism to take care of intense regret would require a feature that could do more than edit the document. It would have to function as a step back in time, a way to retreat from *anything* whatsoever that had happened. Third, Nelson's very early concept of an infinite undo that never disappears produced a sense of safety. Fourth, from the hypertext perspective, an undo command would not just help the viewer/editor stay in the right *here* spatially, but in the right *now* temporally. Undo would be a way temporality functioned, allowing the user to step back. The step would not be of a minute hand or a space on a calendar, but of the smallest meaningful granule of change on the interactive system: a command.

The undo command in FRESS was surprisingly close to the version we know today. A surviving manual from FRESS's last years explains the function as the user would experience it. The command, called Revert, undid the last editing operation. Revert reset the editor's view to the point in the file at which the now reverted edit was initiated. It only undid the most recent editing operation. The user could navigate around the document without losing the ability to Revert the previous editing action. A

related command, Accept, could finalize an edit so it could not be undone (Prussky 1979, 201,114).

This undo command was unique because it could be called at any time, not just at the moment an edit was made. This difference is like playing a game of chess where you cannot change your move once your hand is off the piece versus being able to take back your turn at any point up until you've made your next move. The undo in FRESS made the state of a document prior to the last user action permanently available; you could always go back to just a moment earlier, accepting the definition of moment in terms of commands in a program rather than seconds on a clock.

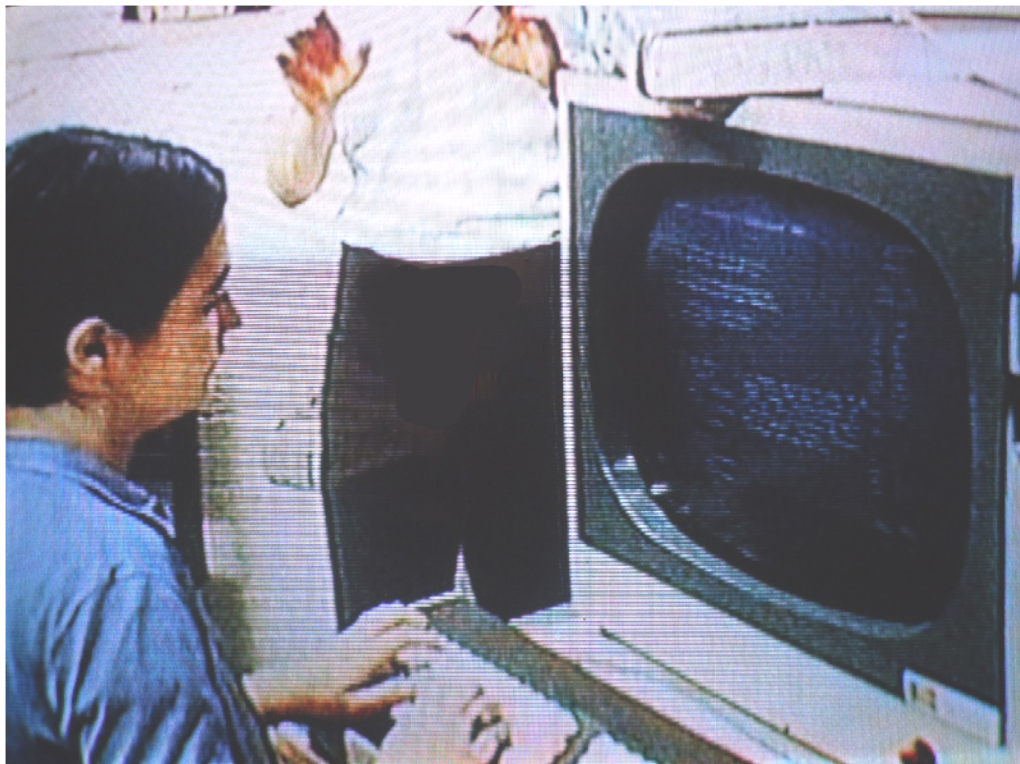


Figure 4.3 Using FRESS on a terminal (Brown University Department of Computer Science 1976)

For FRESS, van Dam explains, “you could undo most things and you could undo it after travel, which is very important to me because when the “Oh, Shit” moment occurs, it typically is somewhere else already” (Andy van Dam and Simpson 2011). HES was the first pancake, partly a proof of concept and partly a very quick way to get a better word processor. HES had a Cancel command which would interrupt a command before it was completed or undo a command immediately after it was completed. This left the user a very short future line of development before they lost the ability to go back. Instead, HES, like many other editors, encouraged users to undo the effects of commands by using other commands: scroll forward is undone by scroll backward, navigating forward could be undone by going back (Return), formatting commands (which were actually non-printing text inserted between the regular text) could be deleted (Carmody et al. 1969, 314).

Revert functioned on the individual command, because this was the smallest unit of action recognized by the system. In batch processing, operators fed an entire program into the computer, which then ran through the whole thing and printed results. No cycles were wasted; the computer ran continuously at a constant speed always doing work. Real time computing made the computer responsive to a user. This required the computer to do something extravagant and inadvisable: it required the computer to wait. The computer’s architecture requires it not to wait; it runs through processor cycles at a set rate per-second, and must do something in each cycle. The computer was too expensive to leave sitting around, yet was capable of exciting things if it would only give a user time to interact with it in an ongoing

dialogue. Time-sharing was an early solution to this dilemma, putting the computer to work on one job while it waited for someone else to respond. Today, computers run an idle process, twiddling their thumbs, to make up for the difference between their maximum possible performance and what they're usually being asked to do. This process must be simple, short, and ideally decreases power consumption. The computer does as little as possible. It is still. A command cuts into this idleness. Users initiate commands mechanically, as if pushing buttons or flipping switches. Initiating a command, the computer runs a procedure of arbitrary complexity whose function the user may never understand (Pold 2008).

In a system where the computer is still and the user only touches it by discrete actions that initiate complex procedures, the relevant flow of events to be reversed is that of the action. Undo allows a reversed course through a series of events that are discrete procedures. If the software's state were continuously changing (if the computer did not wait idly), or if the user could give continuous input (rather than choosing either to initiate a command or to do nothing), undoing would look more like rewinding. Some video games have included such rewind features, emboldening the player by allowing things to be attempted more than once (Moran 2010).

In 1969, Cancel and Revert were not the best system van Dam could muster. They represented a simplified version of yet another side project. In the paper published on HES, the authors claimed they would create a facility for "retaining a complete, or a user-specified, chronological trail of editorial changes, and reconstituting any previous state of the textual contents from them" (Carmody et al.

1969, 328). This probably refers to simultaneous work being done by van Dam's students Warren Elliott and David Potas on a system "preserving the finely delineated details of a text's evolution without requiring retention of specific drafts" (Elliott, van Dam, and Potas 1971, 534). That project would have stored every modification to a document, as it was made, turning all actions into modified versions of other versions. Although some of us hate how paper can easily be lost or notes can be hard to read, this group saw the *mass* of edits made on a page to be the main problem with hard copy editing. This could be fixed by treating all actions taken on a document as *parts of* the document, to be navigated through spatially. Ultimately, the project required an extremely complex data structure, with a record of the difference between two document states referring to yet more records of differences, "putting deltas on top of deltas on top of deltas" (Andy van Dam and Simpson 2011). The project was never completed.

Revert in FRESS undid the most recent editing action taken by a user. It was simpler but it did the job. It was more specific than restoring a previous version of a file, more general than a command that could only restore deleted text, less powerful than more sophisticated undo systems, and quicker to work with than scissors and tape.

A number of other solutions emerged in other systems to the problem of machine glitches, unintentional actions, and users changing their minds. Bank of America's automated check system, Electronic Recording Machine: Accounting (ERMA), read account and routing numbers automatically from checks but depended

on operators to type in the actual dollar amounts. Here they could err. To correct these mistakes, the item could be reversed, which marked it with a special symbol that rendered it void, and a new entry was made using the same account and routing information (“ERMA: Electronic Recording Machine, Accounting” 1955). In batch processing, an error on the part of the user (the programmer) resulted in the computer printing out the entire contents of its memory in a dump that was intimidating and often difficult to interpret; the only way to undo a mistake was to modify the input and run it through again from the top (Ceruzzi 2003, 99–100). Similarly, in Programmed Logic for Automated Teaching Operations (PLATO was the first large-scale system for education, operational from the 1960s) mistakes could be fixed by backspacing, resetting an entry, or navigating backward through a program (Avner and Tenczar 1969, 11–14). Another comparable yet different system is Ivan Sutherland’s *Sketchpad*; here, users make graphics not by coloring pixels but by modifying attributes of vertices, lines, curves, and shapes. All these attributes remain open to modification, making it relatively easy, in most cases, to undo actions (Sutherland 1963).<sup>12</sup> The same alternative to undo can be found in other early systems, such as *Visicalc*. In this, the first spreadsheet and most successful early application for personal computers, any action the user took resulted in the modification of cells on a table, and any of these cells could be manually edited again at any time. The content of the table could be modified in other ways, and there was no obvious reason why

---

<sup>12</sup> *Sketchpad* was Sutherland’s dissertation project years before he succeeded Licklider as director of IPTO at ARPA. According to its longest running director, “the best [ARPA] program managers have always been freewheeling zealots in pursuit of their goals” (Tether 2008).

one should have to “restore” a previous state when they could rather easily recreate it (“Visicalc: A Visible Calculator For the Apple II: Reference Card” 1979).

What these other approaches have in common is that they leave the present open to change. The user is not forced to concede the events of the past as a lasting influence in the present, because something can be done to restore what has been lost. But, in most cases, the past can only be recreated by additional actions, not restored from memory. Phrased in this way, these various operations are not as simple or generic as undo. The obvious way to return to the previous state is to undo, but this is not their purpose. To counteract the effects of a specific action had many meanings; in one case it may be to backspace over a mistyped letter and in another it would be to clear a misread entry. The problem that undo solves had yet to become understood as a generic problematic. It was not yet obvious that a system needed to provide users with a way to return to the recently forgotten past. This phrasing of the problem (to which undo was the solution) *followed* the distribution of undo. For the most part, it was not easy to lose the past because the actions available on computers were of such a limited and discrete nature. If you entered an incorrect answer in PLATO, there was not much to be undone but to answer the question again. A worker entering in amounts for



checks had only to reverse the entry, type it properly, and move on.

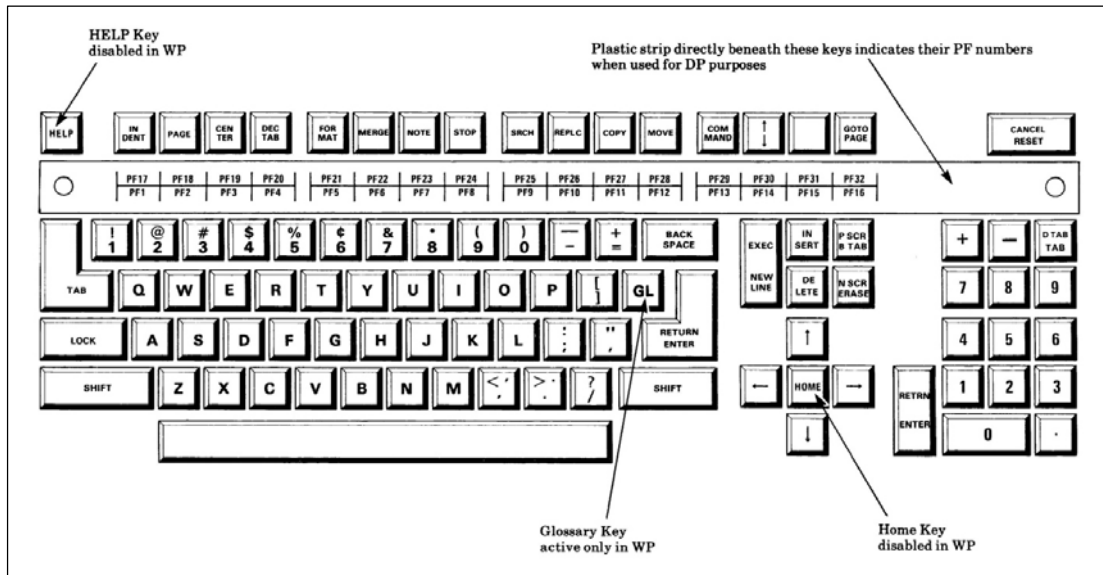


Figure 4.4 Though it had many special control keys, including a cancel/reset key, in 1983 the Wang Word Processor's keyboard did not have an undo command ("Word Processing Reference Manual" 1983, 1–24).

Other systems used very different means to ensure that mistakes would not be permanent. IBM's offering to office automation was the extremely popular Selectric family of typewriters. The machine used correction tape, and later tape that removed ink by lifting it off the typed page. In 1964, IBM announced the Magnetic Tape Selectric Typewriter. On this system, letters are stored on magnetic tape before being transferred to a piece of paper. This lets the typist backspace and retype without using correction tape of any kind (Pieslak 1974, 15, 109). Wang Laboratories dominated word processing into the 1980s with standalone systems (that were ousted by networked personal computers by the end of the decade). Across versions, Wang systems actually improved very little over the IBM Selectric in the area of correcting operator error. The user could backspace over characters, delete any selection of text,

and cancel operations (such as find and replace) before they had been executed (“Word Processing Reference Manual” 1983). Microsoft’s first entry into productivity software, Multiplan (a spreadsheet application), allowed commands to be canceled after they had been specified but before they had been implemented and allowed cells to be locked to foreclose the possibility of any accidental modification (“Microsoft Multiplan: Electronic Worksheet for CP/M-80” 1982, 20,145). *Red Pencil*, released in 1986, facilitated markup of electronic documents to correct mistakes just as one might with a real red pencil. Writers could make mistakes in the writing process, and this program would allow editors (including the original writer) to fix them, or make other modifications, during a later review stage. Surprisingly, the product was not always better than an actual red pencil (“Writing and Editing Tools” 1986, 24). None of these systems included an undo and thus none suggested that the way to correct an error was to return to a moment before it had been made.

In all these examples, one starts to see why undo did not appear in other systems and how widespread its neglect was. FRESS can be seen as the first working prototype of word processing (as we now understand it) that ran on commercial hardware that was also a hypertext system accessible to beginners (Barnet 2010). Despite its importance to the historian, the system is not just forgotten now, it was quite obscure at the time. “No one, to a first approximation, knew what HES was,” although one group at another university ran it on color terminals and expanded on it, “there might have been another half dozen people in the world outside Brown who

knew about HES. We were not influential and FRESS, unfortunately, was not influential” (Andy van Dam and Simpson 2011).

The image of undo implemented in FRESS created a solution to a new problematic: the time of successive user actions. The stack of past actions that might be undone became the meaningful code by which a system could keep time reversible and by which the user could make use of the command. Designed for a system that allowed more commands, including actions that were not easy to reverse (such as find and replace), undo approached complexity with a simplified abstraction. Some kinds of error correction, certainly, have still not been reduced to undo: switching between editing modes, changing the view of a document, or conducting file operations from a program (such as saving or opening) are not undoable actions. But for most things the user did, the program could keep actions as a stack of inputted commands to be undone.

FRESS did not lead to the adoption of undo, but became a teaching tool, used at Brown throughout the 1970s. Finding funding for educational applications from Exxon and the National Endowment for the Humanities, Brown professors taught classes with material and assignments on FRESS. These research projects concluded that computers were a better supplement to teaching than replacement for it, that students using FRESS wrote much more text than they did otherwise, and that students could use the network environment to get into a nasty flame war. As opposed to the traditionally respectful classroom environment or messages written out and then delivered later, with FRESS, “you just flap any goddamn thought that comes into your

mind” (Andy van Dam and Simpson 2011). Ultimately, the research program, like many others, made some findings about the possibilities of educational computing while it simultaneously gave students valuable computer experience. At this time, trends in automation threatened those without computer skills and rewarded those familiar with a machine that was still quickly gaining power and importance.

Hypertext or not, poking in commands and reading error messages was valuable classroom experience.

The invention of the undo command at Brown University resulted from a confluence of forces responding to very different concerns. While computers stormed into governments and corporations during the 1960s to replace human labor of many kinds, Brown’s hypertext project sought to empower those with access to a computer with a new, more efficient, more freely structured means of composing texts. The system would intensify creativity and innovation, facilitate expression, and let genius focus on thought and automate the busy work. This effort had little influence and did not satisfy Ted Nelson’s original goals. Andy van Dam’s plan to kill two birds with one stone did get one bird, as the system produced a viable form of word processing of a kind that would not be available against until the 1980s. The university was rarely supportive of the project, and an administrator almost shut it down calling it an unnecessary substitute for typewriters—despite the fact that Brown luminary Rod Chisolm found his productivity had doubled while using the system (Andy van Dam and Simpson 2011).

The undo function had to be invented again in a few other projects before it began to be recognized, let alone standardized.

### **Inventing Undo Again**

Because an undo function offered practical advantages in specific situations, projects of various kinds came up with their own versions of undoing. In many cases, there seem to be no connections between these inventions except that engineers found themselves dealing with the same situations and came up with similar ideas for responding to them. However, when one particular undo command (that of BBN LISP) became sufficiently well known by the right group of people (researchers at PARC), undo blossomed. This success was not enough on its own. The evolution of splendor in the commercial software market of the 1980s and 1990s sealed the fate of specific features such as undo, and the command quickly became expected. Though the feature was standard on Apple systems, it was the success of Windows that catapulted Microsoft Word to its position as the word processor of choice for almost all personal computer users and exposed everyone who used it to undoing. After these events, formal standards for the IBM PC and then Microsoft Windows locked in undo as a key part of user interface guidelines and, therefore, as a part of most commercial software. This process started slowly, in the early 1970s, accelerated in the mid 1980s, and then became very still for the 1990s and 2000s, only to gradually dissolve since the late 2000s.

At Stanford Research Institute (SRI), Pentti Kanerva rewrote SRI's popular old text editor TVEDIT for a newer computer, and wanted to make sure that, in his

system, no one would have to do anything twice. The program, called TVEDIT, was first written in 1965 (for the PDP-1 by Brian Tolliver), then rewritten for a graphical minicomputer (the Imlac). Kanerva had come to Stanford in 1967 and used previous versions of TVEDIT extensively. This had been his first experience with a full screen text editor, and he was excited about it. In 1971, SRI acquired a newer computer, and Kanerva had the opportunity to rewrite TVEDIT for the PDP-10, a machine that would put fewer constraints on memory and therefore allow more features. In his version of TVEDIT, the user would be less likely to lose work. Text could be changed between upper and lower case, the user could duplicate a block of text, repeat the last command, and use a special command called Oops. Oops could undo a delete command (restore the lost text at the current position), restore the line currently being edited to what it had been before the user started editing it,<sup>13</sup> or restore the entire file to the version loaded at the beginning of the session (personal communication, October 7, 2011). These commands, individually, were not original. What Oops did was consolidate these functions as part of a single command whose operative logic was expressed elegantly by its four letter name: Oops.

For Kanerva, the goal was to minimize manual redoing. What he had that previous designers did not was the increased memory capacity of the PDP-10. The outcome of his situation was a command named for the same feeling that would drive one to undo; van Dam's "oh, shit" became Kanerva's "oops."

---

<sup>13</sup> This editor was line-oriented, meaning text had to belong to a line of up to 128 characters (Andries Van Dam and Rice 1971, 102). The limits of such editors were a major motivation for HES.

Despite Revert at Brown and Oops at Stanford, undo was invented afresh one more time before it caught on. Bolt, Beranek & Newman (BBN) was a high technology company in Cambridge Massachusetts that had transitioned from acoustics work to innovation in computer software. In the late 1960s, when BBN was developing the beginnings of ARPANET, Warren Teitelman (a recently graduated PhD from nearby MIT whose adviser had been Marvin Minsky) designed a programming environment for LISP (a programming language closely connected to artificial intelligence). Teitelman was concerned that the job of programming could now be done at a real-time interactive computer, but the computer was still doing very little to help the process. The programmer has to “write some code, run the program, make some changes, write some more code, run the program again, etc.” (Teitelman 1966, 1). It would be better if software could raise the level of interaction between programmer and computer (2). The computer should not just wait for the human to complete its work; the machine should actively participate in production. If the machine could make it “easier to solve problems” this would make it “possible to solve harder problems” (18).

Teitelman first implemented undo in his Programmer’s Assistant for BBN-LISP in 1970. As he recalls, it was after accidentally deleting a chunk of data that he realized the computer ought to be able to save enough information to keep track of his changes and revert them. At first the feature could only reverse the more frequently used commands. But this made undo unreliable, his colleagues complained; if the user could undo all commands, then the command would offer something really comforting

and exciting. By 1971, users could undo the last few commands (any commands) in arbitrary order (undoing, for example, the third action back, rather than the most recent one). Teitelman added the feature to the version of LISP in use at BBN and, as he recalls, the command became part of the workplace vernacular. In casual conversation, workers would express frustration and regret as a desire to undo real life events, “I wish I could UNDO THRU TUESDAY” (personal communication, October 24, 2011).

In 1972, Teitelman moved to Xerox’s Palo Alto Research Center, bringing undo with him. Xerox PARC is now famous for pioneering laser printers, bitmap graphics, graphical user interfaces operated by mouse, Ethernet, object-oriented programming, and contemporary WYSIWYG word processing (What You See Is What You Get, meaning that what you see when editing a document is roughly what it will look like when printed). Xerox was a very large company rich off xerography patents and photocopiers that used them, but it had taken an interest in the office of the future. Its research center in Palo Alto enjoyed considerable freedom to develop new technologies because it hired the best and brightest (often away from other projects), was far from corporate headquarters, was cheap enough (in Xerox’s overall budget) to keep around, and was uninteresting enough to influential higher-ups that no one really told it what to do, because they wanted nothing from it (Hiltzik 1999). At PARC, Teitelman continued work on LISP and a LISP programming environment (including Do What I Mean, an early foray into autocorrect). In 1974, PARC produced *Bravo*, a word processing application with many innovative features, including a single-step



undo command (Gregory Kusnick, personal communication, October 3, 2011).

*Bravo*'s general undo command would eventually go back several steps, reversing the effect of any command (Lampson 1979, 36, 52).

The technology transfer of undo occurred first by Teitelman's move to PARC and second by people seeing and using the feature in his version of LISP. Personal contact, the human presence of one who understands how something works and actually goes about using it regularly, is a crucial mechanism of technology transfer in many contexts. At PARC, every new employee gave a talk about their work and then was assaulted by a notoriously harsh question period in which well-respected (and sometimes arrogant) engineers tested the newcomer for weakness (Hiltzik 1999, 146). Larry Tesler, who rewrote *Bravo* into a user-friendly modeless editor called *Gypsy* in 1974, remembers undo as something people at PARC attributed to Teitelman (personal communication, October 8, 2011). Although intellectual history readily notes the chains of similarity between a number of technologies and their inventors, engineers are often more impressed by a working prototype than a set of old ideas. To see a machine in use is not the same as hearing about it. To have it right there, one can imagine its future, connect it to their own work, and imaginatively link it to other devices with which it might one day interact (Barnet 2010).

From PARC, undo crept into work at Microsoft and Apple. Xerox's PARC facility was extremely productive. Its chief output was high-potential innovations that were very relevant to current and anticipated changes across the industry. What it lacked was a parent corporation prepared to turn talented engineers' heartfelt

experiments into products for the market. Steve Jobs and Bill Gates both recruited talent away from PARC without too much difficulty, especially as the spirit of the organization wilted with the end of the 1970s. Those who left PARC brought with them a familiarity with PARC's office of the future. Though it may be an exaggeration to say Microsoft or Apple *copied* PARC inventions (Horn 1996), technologies created by these employees bear a stronger resemblance to PARC technology than those technologies created by anyone else

Microsoft recruited employees from PARC and soon developed Microsoft Word. In 1981, Microsoft hired Charles Simonyi as director of application development. Simonyi planned a consistent graphical user interface for a suite of productivity applications based on ideas at PARC (Allan 2001, 12/22). Simonyi, who had been a lead on *Bravo*, hired others who had worked on the project, and its successor, *Gypsy*. One of these, Richard Brodie, scrapped code passed down from Microsoft's existing *Multi-Plan*, and, from 1982-1983, created a simple word processor with a number of features that the competition would not have for years to come (Tsang 2000, 57). Microsoft *Word 1.0* was a WYSIWYG, full-screen text editor with a mouse driven graphical user interface that supported proportional fonts, style sheets, multiple windows, footnotes, and an undo/redo feature that built on *Bravo's* piece table architecture.<sup>14</sup>

---

<sup>14</sup> Piece table architecture appends edited bits of text to an established document file. This took less memory, allowed one to edit a large document without having to load it all into memory, and meant one was never altering the underlying file, but just a data structure that showed which pieces of the file were intact and which had been edited.

There are some common features in the several known projects that invented an undo function from 1969 to 1971. Certainly the availability of computer hardware was a prerequisite to an undo command; each project needed an interactive real-time computer interface, an editing program (in each case, for text), and a bit of excess memory.<sup>15</sup> Additionally, in all three cases, designers needed some leeway to pursue their own ideas, without having to make too compelling a case for the idea's viability or necessity. Each undo command presupposed a system with a great number of commands in which a user could build complex documents the creation of which would entail many mistakes and wrong turns. Finally, the designers each had a personal motivation that made them want to be able to undo actions at the terminal.

What should we make of the multiple, nearly simultaneous, inventions of what is almost the same command? One response is to infer from these inventions that conditions were right for several inventors to create almost identical inventions simultaneously (Scharf 2009). The situation expressed itself in the technologies it attracted from designers, and, though these technologies had very little impact at first, they eventually did change the scene from which they were born. But the historical reality is still one of details. Undoing did not simply arise out of a cultural longing for stability or an early 1970s desire for the comfort of eternal return. That a few tinkerers tried similar things can also be explained as a product of random chance (Simonton

---

This made it easy for the program to keep a record of edits so that they could be reversed by undo (Richard Brodie, personal communication, September 14, 2011).

<sup>15</sup> Undo probably developed on text editors because there were lots of them, they did not always tax computer memory to the limit (as image editors did, for example), and because programmers needed text editors to do their own work.

1986). Undo was not the only or most original concept in these projects, there were many other original projects at the time, and pioneering work was happening during this period in many other fields.

Undoing became a possible cultural expression only *after* it became common enough that other people might have heard of it (as happened early on at BBN). *In hindsight*, Van Dam, Kanerva, and Teitelman produced different commands that all resemble the contemporary concept of undo. But would they have considered their commands similar at the time? It was through the standardization of these techniques at PARC, Apple, Microsoft, and then in the commercial software market of the 1980s and 1990s that particular features from a few old programs begin to appear as multiple, simultaneous inventions. The question is not, “Why did several people invent the same technology?” (the technology was not the same). The question is “How and why did these different technologies come to appear as one?”

To follow the development of undo from a few experimental systems used by small groups into a standard feature on many applications used by millions, we must turn to commercial software.

### **Undoing in Commercial Software**

Up to the late 1960s, software was not a commodity. A program that worked on one computer would not work on another. At this time, software and customer service went hand in hand. If IBM sold General Electric a computer, they would also send people along to install it, get the programs running on it, train staff at GE to use

it, and answer questions as they came up. GE estimated this process took an average of one man-year (“IGE Presentation” 1962, 5). The programming of a computer was always specific to the computer and there was no expectation that a program written for one machine would work on another. (Software companies today may use this older model prior to productization; if the product won’t simply install and run smoothly on everyone’s computer and for everyone’s uses, a company can sell it as a service provided on contract.) The paradigm we know today, where software comes overpackaged in colorful boxes, downloaded directly, or burned to optical media emerged in the 1970s.

The software industry separated gradually from hardware for a few reasons. First, in the summer of 1969, IBM announced that it would unbundle its software services from its hardware services. This decision responded to a number of concerns for IBM. First, IBM was at the beginning of anti-trust inquiries that pointed out its very real dominance of the computer industry. Second, bundling software with hardware left the company vulnerable to competitors who would design compatible hardware and then run IBM software without paying for it (Humphrey 2002). IBM’s own 1964 decision to develop a standard product line of compatible computers (the System/360) promised to save corporations money because software (usually custom for each company) could be used on different computers *within* the IBM line (Ceruzzi 2003, 144–145). What worked on one /360 would work on all the others. Or at least that was the idea. Additionally, floppy disks first came out in the early 1970s and allowed users to copy files between computers relatively easily. Finally, with the

nascent popularity of personal computers in the early 1980s, there was a pressing need for quality software to help hardware manufacturers sell computers (Meserve 1983, pcw-3). By the 1980s, this need could no longer be met by individuals and small teams, because the programs that would sell were complex, versatile, reliable, and interoperable with existing systems.

Although the beauty of the computer has always been its range of applications, one type of program was more important in software than all others throughout the history of the industry: word processing. When we think of word processing now, we think of a person sitting in front of a large, color display typing on a keyboard, editing as they go, formatting the document, and printing or sending along the file to others. This has not always been the case. The term “word processing” came from IBM, where it was adopted because it sounded equally important as “data processing.” Data processing made and sold mainframes, for which IBM was well known. Those working in word processing made and sold typewriters. Word processing became a key part of the very popular concept of office automation. It was particularly attractive for corporations because it symbolized futuristic automation, only required a minimal departure from existing practices, and could easily be forced onto relatively powerless workers (secretaries, typists, and other “girls” in no position to resist the changes). Directly replacing typewriters, word processing usually described all-in-one machines with a dedicated function, such as those made by Wang Laboratories. These glorified typewriters actually had less capacity to manipulate text, in many ways, than FRESS or TVEDIT. But they were affordable and helped corporations handle the paperwork

explosion. As personal computers became cheaper and more powerful, hobbyists, schools, employees, and eventually large organizations began using them for whatever they could—and word processing was one thing these minicomputers could do (Haigh 2006). Early systems reproduced standalone word processing systems, but as the IBM PC and Apple Macintosh emerged as stable hardware platforms, many word processing programs began to compete, only consolidating in the mid-1990s (Bergin 2006).

Microsoft released its highly advanced word processor (that included undo) for DOS in 1983 to weak sales. The program debuted in a period where a new trend in software was for *idea* processing, rather than just data or word processing (Shapiro 1984, 81). Brodie's *Microsoft Word* was not just another typewriter; it consolidated a number of related functions, such as formatting and footnoting, into the job of one capable person sitting at a rather powerful personal computer. Functions that had once been done by different techniques at different stages by different people were all within the purview of the new program and its user.

The program sold poorly. One reviewer suggested that *Word*, while great for formatting documents and rich with new features, “may have gotten too far in front of today's hardware” (Markoff 1983). The more specific reasons for *Word*'s weak sales are instructive. Performance was sluggish on most machines. *Word* used a mouse at a time when most users had never touched one. The copy-protection scheme required the user to keep the application disk in the drive while working (if your document was on a disk, you'd have to switch between that disk and the program's own disk,

disrupting your work), which was a deal-breaker for some companies (Calta 1984). *Word* was one more entrant in the already packed IBM PC word processing market (on Apple computers, where the mouse was standard, it was virtually alone). Most users had already cut their teeth on *WordStar* and saw little reason to learn Microsoft's new system. Still, the program did better with later versions, and became a top competitor by the mid-1980s (Petrosky 1985).

**Undo. Windows. Mouse. Finally.**

New Microsoft® Word. It makes your IBM Personal Computer think it's better than a \$10,000 word processor.

With Microsoft Word, what you see on the screen is what you get on the paper. So it's easy to spot mistakes. **Boldface**, underline, and *italics* look like this, not this: ^Bboldface^B, ^Sunderline^S, ^Iitalics^I.

And, when you make changes, paragraphs are automatically reformatted. Flush right, flush left, centered or justified. It even gives you several columns on a page, like a newspaper.

**Word forgives and doesn't forget.**

There's an "uh-oh" command called Undo. Make a mistake? Or just want to experiment? Hit Undo.

**Word does windows.** Up to eight, to be exact. So you can transfer or edit between eight different documents. Or between eight different pieces of the same document.

**Word travels fast.**

Word has a Mouse, a handy little critter that lets you move copy, select commands and edit faster than you can say "cheese."

Word also lets you create your own style sheets, so you can standardize your documents, memos, files and letters.

It's not surprising that Microsoft has a way with Word. We designed the MS-DOS operating system that tells the IBM® PC how to think. And we pioneered the first microcomputer BASIC, the language spoken by nine out of ten micros worldwide.

For a few final words, call 1-800-426-9400 (in Washington State call 206-828-8088) for a free Word brochure and the name **MICROSOFT** of your nearest Microsoft dealer. **MICROSOFT** The High Performance Software

Figure 4.5 *Microsoft Word* advertisement in *Byte* magazine, July 1984.

Before *Word*, undo in word processing usually meant only undelete, but by 1986 it became grounds for complaint if a program didn't have a full undo. *VEDIT*, a word processor that advertised aggressively in computer magazines, touted among its



features ““Undo” key to restore line” starting in 1981.<sup>16</sup> But Word and Apple had set a new standard for undo. Projects with full undo features, such as FRESS and BBN-LISP, continued to pop up outside the mainstream of commercial software; Convergent Technologies released an operating system in 1982 with an undo command, and Apricot Computers released a personal computer that could undo in 1983. WordStar’s late adoption of an undo function (offering only unerase) contributed to the program’s declining reputation (Miller 1986), and those reading a review of Wang’s word processor application for the IBM PC were reminded that most full-featured word processors at the time had an undo command of some kind (Freeze 1986). Programs without an undo function were subject to the criticism that they had poor error-handling and were not quite up to date (Lombardi 1986).

Macintosh, Apple’s 1984 runaway success of a personal computer, standardized undo, allowing the user to undo just about any action taken on the machine (Neudecker 1984, 90). This was great for users, but required extra work from developers. To make a program for the Macintosh required conforming to the user interface established by *Macwrite* and *Macpaint*. This meant not only including an undo command, but also working with a bitmapped display, a mouse, and a clipboard that would hold information in a format compatible with other Macintosh programs (Watt and McGeever 1985). This challenge decreased competition on the Macintosh as a platform but it did not deter all developers. Eager to work in the Macintosh environment because it was similar to the Alto, Xerox PARC’s personal computer,

---

<sup>16</sup> This wording is from the 1983 ad, but VEDIT ads spotlighted an undo command of some kind from 1981 through 1994. (Green 2011)

and because it represented a possible future for computing, Microsoft released *Word* for the Macintosh in 1985. The program soon became the standard word processor on Macintosh, and gave Microsoft the opportunity to perfect their application for a windowed graphical user interface.

Apple and Microsoft Word exposed huge numbers of people to undo.

Although Word was successful on the early IBM PC, and dominant on Apple, with the rise of Microsoft Windows, it became just about the only word processor anyone used. Word and Apple's graphical operating systems had both been based on ideas from PARC. But Apple had never been top dog in personal computing; their focus had been on users concerned with graphics and usability (Eran 2006). Thus, Apple's standardization of the undo command would never reach the average office worker, or any specialized markets Apple did not target (such as computers for programmers, data-entry, or the sciences). Microsoft's software work for Apple operating systems had trained the company for the windowed graphical user interface that was seen as the way of the future. *Microsoft Windows 1.0* came out in early 1985, a year when a number of similar systems debuted. All of these systems did poorly, leaving customers waiting for a decent windowed, graphical operating system. IBM collaborated with Microsoft to develop *OS/2*, a project that was to have been the definitive solution to the question of windows. However, collaboration faltered and Microsoft returned to development of Windows, leaving *OS/2* to IBM. Most companies with popular word processors continued developing for *OS/2*, but Microsoft focused its applications on its own operating system. When *OS/2* came out in 1988, it was expensive and

underwhelming. Two years later, *Windows 3.0* came out with a large-scale marketing campaign and was received enthusiastically (and was bundled with many new PCs). This was the environment that, for seven years, Word had been developed for, first within DOS and then on the Macintosh. The odds were stacked in its favor (Campbell-Kelly 2003, 246–252).

Ultimately, it was the fact that Word ran on the new and ubiquitous operating system called Windows that brought it the masses. Secure on both major operating systems for personal computers, business practices surrounding the product changed dramatically. Word was already king of the hill: it had to maintain that position and expand the size of the hill. This made giveaways, with charges for updates, and free licenses for apologetic pirates reasonable tactics. Word also had to prove itself useful for more varied kinds of use. During this period, as huge numbers of people became familiar with Microsoft Word, undo, because it was included in the program, became something that anyone who had ever worked seriously on a computer would have used. As they used undo, they became familiar with it, and expected it.

But why did undo proliferate across software so quickly from 1983 to 1986?

From the 1960s, the most common perspective on changes in computing was a narrative of progress. New products were more or less advanced, computers represented new or old technology, and the future had to be anticipated. Alan Kay, at Xerox PARC, voiced the rare opposite opinion, that “the best way to predict the future is to invent it.” PARC was well funded and talent rich. For everyone else, the future was something to chase after. If a new device was less efficient than another, if a

program came out for a system that was losing users, if the niche or market for which one was designing was about to change, or if you were building for time-shared mainframes when the next thing was personal computers, the accomplishments of an individual, a project, or an entire company could vanish like a pipe dream.

However, a consideration of market dynamics shows that the future was not inevitable. The future that actually occurred was a result of a complex playing field and the particular moves that were made on it. The case of undo demonstrates some of the forces guiding the evolution of splendor in commercial software during this period.<sup>17</sup> New features were cool, these cool features became expected very soon after they were introduced, consumers demanded compatibility, user-friendly software sold well in a quickly expanding market, and companies had to intelligently target those making purchasing decisions. Each dynamics is worth considering.

New features helped products sell because they were promising. It was during the 1980s that a huge number of features we now take for granted in word processing first appeared: hyphenation and justification for word wrapped text, multiple windows, movable columns in a table, printer drivers, tools for writing formulas and equations, designs for many sizes of paper, subscript and superscript, decent-looking icons, fonts, kerning, and file management. Versatility of a program meant that a product could be used in new ways, and some programs made a point to allow features such as music composition or scholarly references that would cater to a specialized market. By

---

<sup>17</sup> Evolutionary theory tends to emphasize reproduction of the unit of selection, such as the individual or colony, but alongside this evolution of individuals, aesthetic practices mutate and proliferate as well (Lingis 2005, 20–54)

writing a program more cleverly, or with greater emphasis on a feature, a new product could claim to multiply old features, such as by allowing one to work with documents *three* times longer than before or having *twice* as many fonts as a competitor. New capacities on a program also made the program a substitute for more than one previously existing program: word processors that included spell-check replaced not only previous word processors, but also previous spell check programs. Changes in an interface that responded to consumer complaints were exciting because they expressed the ambitions of many who had used older programs. Such enthusiasm had a part in the replacement of large menu bars with pop-up menus, for example.

New programs copied features from other programs because they knew this would make them look better in side-by-side comparisons. Many reviewing periodicals, such as *InfoWorld* and *The Seybold Report on Desktop Publishing*, lined up products next to each other to compare features. Any program without an undo feature, for example, would appear lacking in this kind of categorical analysis. Reviews expressed the disappointment experienced users and industry observers would feel with new products. New programs should come with “the niceties we would expect,” and “all of the features one would expect” (“Desktop Publishing” 1986, 23,24).

In the burgeoning computer market, compatibility between devices was a necessity for software. A program must be able to run on the operating system of a potential customer for it to even be considered an option. But this is not enough. The program should also work with other display, input, and printing devices that the

potential customer either already owns or may have plans to acquire. Consolidations of the market into dominant platforms, operating systems, and data interface formats made it easier to produce software that more people might buy and made it easier for those who already owned equipment to select the software they really wanted, rather than the software they had to buy because it was all their hardware supported.

Software (and hardware) that was easier to use for those unfamiliar with a computer had several advantages. First organizations could spend less money on training. Prior to IBM's unbundling, software had been fused with the service department. Training was included in the price of software. Unbundling meant that those buying a program had to internalize the costs of training people to use the program. This was a general problem for all software suites, and meant that something easier to pick up and to master was cheaper than something that required training to use. Although this concern seems abstract, business and professional users were the most important source of revenue for personal computing into the 1980s ("Research Memorandum: User Spending Forecast" 1981, 6). New software could create new inabilities for established higher-ups in a company, as some took to it more easily than others. Easy to use software could potentially bypass this political quagmire. Outside of organizations, the same savings-on-training took other forms. For software producers, an easier to use program required less technical support by phone. For users, a more approachable program was better because they wouldn't have to read the manual or sit on hold calling a support number. Second, easier to use programs could be introduced more smoothly in the large education and consumer markets. Apple, for

example, established itself in education and reaped the reward when students hoped to use the same computer at home that they had learned to use at school (57).

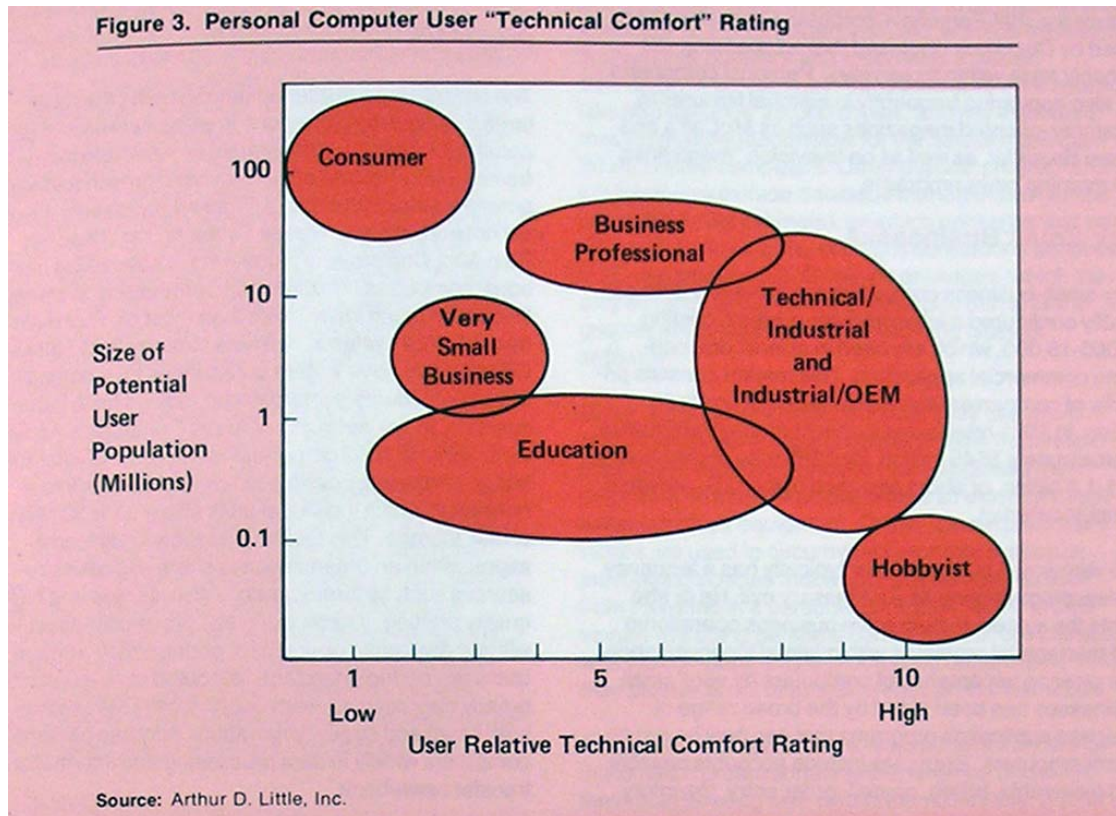


Figure 4.6 This chart from a consultant's report in 1981 showed that though the consumer market was large, it had very little technical comfort and would require very approachable software (Meserve and Wright 1981, 15).

Third, previous decades had cemented a fear of computers so that customers had to be coaxed into using them. Those films where the computer goes out of control and starts killing humans were but one part in a broader prejudice against computers. Because of this, having useful programs on quality machines was not enough. The thing had to look friendly, whatever that could possibly mean (Meserve and Wright 1981, 14–16). Fourth, the user base of computer software expanded by drawing in

many people who had not before used computers. The old ways of using a computer were of interest to a declining minority; new people saw all systems as new and wanted one they could understand easily. These people would be less impressed with products for working the way old products did and more excited about products that they could make sense of quickly and actually get something done with.

For software to sell, someone had to buy it. An interesting problem with software (a relatively unknown and difficult to classify good at the time) was that it was not clear who should be the one to buy it. Was this the responsibility of technical support people, office managers, secretaries, or someone further up the chain of command? The difficulty of this problem can be seen most clearly in the story of *PageMaker*. Paul Brainerd had worked at newspapers laying out pages with a variety of technologies and began his own company to produce a piece of software to do layout for newspapers. His original plans to market the product to daily newspapers fizzled as he slowly discovered that no one he met with was in a position to purchase such a product. At many offices, he found that such a decision would have to go through a corporate chain of command, with approvals and oversight that would delay the decision by at least a year. At that speed, his small start-up would be bankrupt by the time it made its first sale. Instead, he discovered a relatively unrecognized market of small-time publishers: churches, businesses, clubs, and schools who had sufficient funding and circulation to buy a product that could streamline their production process. *PageMaker* came out in 1985 for the Macintosh, with some help from Apple, and championed the new idea of *desktop* publishing (thereby helping Apple sell laser



printers) (Brainerd 2005). This story demonstrates the interstitial locus of decision making to which commercial software had to respond. The market for personal computing was, in this way, easier to deal with than corporate customers, even though the big companies were still spending the most money on software. With personal computing and smaller organizations, at least it was clear who was buying. This problem did not have a single resolution, but it did mean software had to be designed for those making purchasing decisions, not just for the ideal end user.

The undo command thrived in these conditions. The feature was new and interesting, but even better, made it safe for producers to add even more features to their product. Programmers often come up with interesting ideas, but if all of them go in the final package, a program becomes bloated and users will make mistakes they don't even understand (Heckel 1982, 26). An undo feature makes an overloaded program more comfortable. A mistake made with *any* command (even one you don't understand) can be cleared up with one quick undo. The feature became expected in other programs very quickly, and the fact that it is easy to refer to ("undo") probably helped it spread. However, the catchy name also made it less clear what really counted as an undo feature or what, exactly, an undo should do. Though it did nothing to make a program compatible with other hardware or software, the command benefitted tremendously from the standardization produced by operating system compatibility. To minimize training, undo is quintessentially user-friendly (and was developed for exactly this reason at Brown, Stanford, and BBN.) It helped users recover from errors and gave them a "tremendous security blanket" that let them experiment more and

worry less (Williams 1983, 39). Undo did not help software producers target decision-makers, but helped give a product a gentler feel to those who used it and those who advocated on behalf of a purchasing decision.

Outside of word processing, it was harder to implement an undo command, yet it was still the norm. Undoing is easier for text than images or video. The amount of data actually altered by word processor commands is small compared to other kinds of editing. If someone deletes a whole page, or replaces all instances of a word, an undo feature will require that all the missing text be retained along with the location of the changes. Image, sound, and video editing programs require more. To delete a segment of a sound file isn't much worse, but to undo an effect that alters the entire clip (such as an echo) requires either an algorithm that reverses the effect (often impossible) or a buffer that stores an entire copy of the file prior to the change. With images, some programs came up with new approaches to make undoing easier. *Live Picture*, for example, stored changes to an image as layers on top of the original image. Adding a blur effect to the image only really added a layer of blur over the old image, producing a temporary copy of the image for the user to view while preserving the original file (Borzo 1993).<sup>18</sup> Something similar happened again with Smart Filters and Smart Objects, introduced later in Photoshop CS 3 (in 2008): elements of an image could be modified without damaging the original source. This is also the concept behind non-destructive video editing: the editor modifies a set of references pointing to original

---

<sup>18</sup> *Live Picture* was exceptional and ahead of its time (Neel 2011). *The Language of New Media* used it as an example of how computer graphics can get beyond the limits of pixels (Manovich 2001, 53).

footage, rendering small clips as needed. No amount of editing changes the original footage.

Because many programs included an undo command, it was standardized into user interface guidelines set by Apple, IBM, and eventually Microsoft. Early on in my research I assumed that these guidelines had standardized undo themselves. However, the effect of these documents has been closer to the function of ordinary laws; they conserve practices that have already become the norm by specifying them formally and enforcing them with varying amounts of success. The interplay of magazine reviews, word of mouth, impressions of consumer demand, and other factors played out according to certain dynamics and resulted in a de facto standardization of undo that prompted those producing operating systems to formalize the standard.

Apple began setting standards for its user interface in 1978 with the arrival of Bruce Tognazzini. Tognazzini had owned a computer store selling the Apple II in San Francisco. There, he wrote an application with his partner that would introduce new users to the system's interface. To figure out what worked, he ran experiments with users to see what they could and couldn't do on the system and how easily. In this process, he began to settle on firm opinions about how interfaces ought to be designed. Employees at Apple found standards attractive for a few reasons. First, standards seemed to decrease the amount of time spent on user training (and its analogues such as phone support). Second, there was a delay of several weeks between when a manual was finished and when it was available on paper. During this delay, work on the system continued, probably at an accelerated pace, but the manual was already

finalized. With standards, technical writers could confidently state what a program would do before it was actually done. Third, Tognazzini based interface guidelines on the publications style guide he had already written. The publications style guide gave consistent explanations for all aspects of Apple systems, including interface, and this made design more consistent as well.

Apple's standards were not popular with developers but ultimately derived their value from the evidence of user testing. Developers could design for the IBM PC, Apple, or another system. Why should they choose Apple and why, if they did, should they accept Apple's approach to interface design, when there was a whole unexplored world out there of user interface design? Developers wanted to see what they could accomplish, to distinguish their products from each other, and to spend as little time as possible jumping through hoops for Apple. What Apple had, however, was a quasi-scientific data source. In addition to tests run in Tognazzini's computer store, Apple ran experiments continuously. After orientation, new employees at Apple had the option to be a user test subject for a few hours before starting work. New employees counted as naïve users because, while Apple would have preferred to hire candidates with computer experience, most new recruits had little to none.<sup>19</sup> Subjects sat for an hour or two with a computer, observed by a designer, and recorded for later scrutiny (recordings were only audio at first but included video in later years). Apple conducted a few sessions of testing that were more rigorous and scientific, with

---

<sup>19</sup> Brown undergraduates who had used FRESS would have had an advantage getting a job from Apple, quite aside from any educational benefits of hypertext. Computer experience, even a decade after FRESS, was valuable and rare.

double-blind setups and subjects who weren't employees. Through these many tests, a number of concepts taken from PARC turned out to need significant reworking. For example, before "OK" and "Cancel" became default alternatives, Apple had a prompt reading "Do it" or "Cancel." In testing, they discovered that some users misread the first option as "Dolt" and consistently chose to cancel rather than accept the insult (Atkinson and Hertzfeld 2004, 20–21).

For Apple, the tests resolved contentious debates between employees in various departments about look and feel. For developers hesitant to do things any way but their own, user tests made a compelling case that standards would add value to a product rather than frustrate the design process (Tesler and Espinosa 1997).

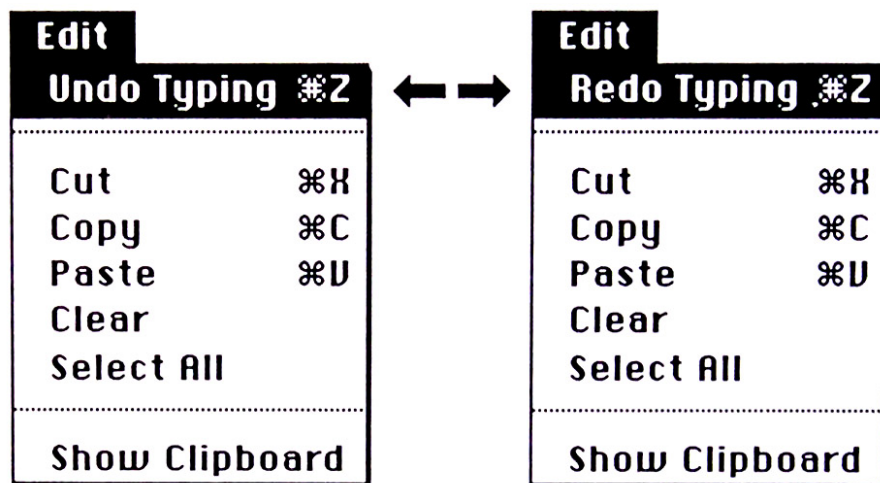


Figure 4.7 The menu names the action that will be reversed, or redone (Apple Computer 1987)

The undo command actually fared poorly in early user tests yet was included in the interface guidelines from at least 1980. Apple Writer II shipped with only undelete mechanisms ("Apple Writer II User's Manual" 1981, 30–31), but the command

appeared in Lisa User Interface Standards by 1980. Most users who had used a computer had never seen an undo command at this point in time and didn't understand what it meant. This may be a reason the Apple User Interface Guidelines of 1985 specified that "the actual wording of the Undo command as it appears in the Edit menu is ""Undo xxx", where xxx is the name of the last operation" (Tognazzini 1985, 99). "Undo typing" is a bit easier to understand than just undo alone. However, the command's unpopularity suggests that what failed in user testing was not always abandoned, but was treated with more care. The Lisa specification calls for a single-layer undo command that applied only to edit commands, adding that, "in later releases it will apply to all" commands (Atkinson 1980, 16). By the 1987 guidelines, the command was generic and unremarkable (Apple Computer 1987, 82).

IBM set its Common User Access guidelines in its Systems Application Architecture document in 1987. This set of standards aimed to encourage interoperable software for different platforms and operating systems on IBM PCs. The Common User Access rules had several justifications. They would lead to better user interfaces by solving common problems and formalizing best practices, increase consistency between products, streamline the development process by providing ready-made answers for many design questions, and ensure that what a user learned with one IBM program would probably work in another (Berry 1988, 283, 289). The standard was designed to be able to evolve and respond to changes. In 1987, it encouraged developers to provide undo capability, but did not specify its functionality (Berry 1988, 295). The 1989 edition included an undo command of unspecified depth

that should appear under the edit menu with a dynamic wording, like Apple's, that specified what type of action will be undone (IBM Corp. 1989, 3.7.6.15).

Unlike Apple's reliance on user testing to support its standards, IBM's guidelines simply had the right supporters. Because they came out of a time of collaboration between Microsoft and IBM on *OS/2*, both companies endorsed the guidelines and used them in their respective operating systems (*OS/2* and Windows for DOS). Other key backers included Lotus, Aldus, WordPerfect (these three produced some of the most popular PC software of the era), and the Open Software Foundation. IBM encouraged the CUA as an open standard and released *Easel* (a development environment for those building for *OS/2*), which made the CUA easier to follow (Berry and Reeves 1992, 417).

Since that time, Microsoft developed its own user experience and inductive interface guidelines, Apple has updated its guidelines many times, and IBM has stopped producing operating systems. The contemporary documents describe undo in almost unchanged language, but bring the command up in many more contexts to emphasize the more general principle, common to both Microsoft and Apple, of *forgiveness* (Taylor 1990, 126; Apple 2009, 8). The formalization of undo through interface guidelines has standardized its function and made that function an implicit part of wizards, confirmations, destructive changes, and actions that commit the user to a particular future. Forgiveness is one of the fundamental design principles for both sets of rules, and these documents relate its meaning by frequent references to undo commands. For example, a program should warn users before they do anything that

can't be undone and should use an after the fact undo command instead of a popup confirmation window whenever possible. It has become a rule that the user can reverse the flow of actions and need not be swept away by it.

Still, the role of voluntarily adopted standards is easy to overstate. Usability studies by Jakob Nielsen and collaborators have shown that developers often fail to comply with standards. They violate guidelines when designing, fail to notice violations in other programs, and end up producing products that break up to half the rules in a set. Two identified reasons are that developers rely more on the examples rather than the rules in a standards document and, second, that they use experience from other user interfaces to make decisions, rather than just sticking with the rules (Thovtrup and Nielsen 1991). This suggests that the weight of tradition is a more important force in the proliferation of a feature than formal standardization.

Yet, by standardization and tradition, the substantial variation between undo functions disappeared and the command took on a much more singular identity. Teitelman's Undo allowed any of the last several commands to be undone. Kanerva's Oops could restore deleted text by undoing a delete, restoring a line, or restoring the whole file. Van Dam's Revert would restore, from memory, the part of a document that had been modified by the last command entered. In 1984, Jeffrey Vitter, also of Brown University, proposed US&R, an undo system that turned all commands into units that could be recombined rather freely by choosing which command one wanted to have undone or redone (Vitter 1984). After standardization, things got simpler. "Undo reverses the effects of the previous operation" (Apple Computer 1987, 82), or,



to say the same thing in a different dialect of technical writing, undo “reverses the most recently executed user action” (IBM Corp. 1989, 2.5.4.1.2). The historical course by which a technology, considered useful, joined practices established a common practice of time whose precursors now appear as imperfect attempts at the same thing.

The same cannot be said for the conceptual scaffolding that created each undo command. Standardization did not simplify the ideas. Each creator gave a different justification for the command, but, over time, we are no closer to consensus. Instead, we use undo as a matter of course. Though IBM’s CUA and Apple’s HIG subordinated undo to the principle of forgiveness, few read these documents. Standardization has helped undo flourish as a material actor, used by individuals at the computer to practice new kinds of time.

Each invention of undo discussed here had its own explanations for the affective role of undo. In the original HES paper, a future undo facility was desirable “both for reference to previous drafts, and for return to earlier document states deemed to have been preferable to some present condition” (Carmody et al. 1969, 328). Van Dam now remembers the command’s invention as a way to instantly ameliorate an “Oh Shit moment” (Andy van Dam and Simpson 2011). Kanerva was less concerned with regret and more with frustration, trying to recreate the TVEDIT system so users would have to repeat work as rarely as possible (personal communication, October 7, 2011). Teitelman stated clearly his aim of computer-assisted programming in his dissertation (Teitelman 1966). Rather than frustration or regret, he emphasized that, as a practical matter, undoing was more efficient for memory and computing time than

keeping around backup copies and restoring from them when something went wrong. As an aid to the computer user, the existence of an undo command gives the person at the controls confidence, lets the user relax, cuts out distractions from having to restore lost work, and allows the person to experiment without thinking through all the implications of a series of actions in advance (Teitelman 1966, 918–919). Again, in Vitter’s recreation later of undo, the justification is different: “the ability to recover from unforeseen errors gives users more freedom to experiment with the system’s advanced commands” (Vitter 1984, 39).

The intention to improve operations by influencing the user, rather than the computer, worked quite well. The nascent field of human-computer interaction argued that usability was at least as important as functionality, that it increased adoption, and increased efficiency for expert users (Goodwin 1987). One study found that typographers used undo commands to correct mistakes and experiment with page layout (Bødker 1989, 184). Computer programs are tools that make people powerful; an undo command effectively reduced the risks of powerful accidents (Dix, Mancini, and Levialdi 1996). In the early days of interactive computing, those at a terminal had a rather good idea how the computer worked and what was going on at any given moment; since that time, the norm had shifted to the point where most users were uncertain what was happening or how they should go about using the program. In this atmosphere of heightened ambiguity, undo helped out (Howard et al. 1997). For those producing recommendations for future design based on observed use, undo became understood as a user intention, not just a system function (Abowd and Dix 1992).

A number of key figures in usability threw their weight behind the command. The first book on human-computer interaction included undo as a key principle, arguing that expert users then spent 30% of their time recovering from errors (Card, Moran, and Newell 1983, 423). Authoritative figures in the new field including Bruce Tognazzini, Jef Raskin, Ben Shneiderman, Donald Norman, and Jakob Nielsen all supported universal undoing as a fundamental principle for design, some even going so far as to recommend a universal undo/redo key on all keyboards (Tognazzini 2003; Raskin 2000; Shneiderman 1986; Norman 2002, 131; Nielsen 1993). Because it was found useful in real life cases of computer use, the command was canonized in human-computer interaction and preached to professors' students, consultants' clients, and any who would listen. Various theorists would claim it as essential to letting the user suspend disbelief (Laurel 1991, 114–115), achieving a flow state (Kay 2004), or whatever other theory they used to understand people using computers.

Further research revealed further details about the command's actual use. Users, it turned out, generally undo by clicking an icon or finding it in the menu, rather than using the slightly quicker keyboard shortcut, unless they work with computers for a very large number of hours and work near others who use shortcuts rather than the mouse (Lane et al. 2005; Tak, Westendorp, and Rooij 2013). Older (often higher-ranking) individuals fear the embarrassment of making a mistake on the computer and, insofar as undo provides safety in general, it may help mitigate fear in this influential class (Riggs 2004, 15–16). The command also streamlines operations for expert users by letting them skip confirmation pop-ups (Taylor 1990, 127). For

users of all kinds, the affective contribution of undoing is to make people more bold and creative.

These are exactly the kinds of changes of working conditions that make knowledge workers more productive, a field of inquiry also known as ergonomics or usability. Knowledge workers need autonomy, will have to learn new things and master new systems, benefit from boldness that leads to innovation, and are more valuable for producing quality than quantity (Drucker 1999). Computer systems that could save workers from repeating anything, require minimal memory, decrease frustration, and encourage habit patterns were considered good for productivity, in a rather broad sense (Fried 1982)

Today, the standards that formalized the undo command as a requisite feature for all software have lost power. Development for Microsoft Windows and Apple OS X remains subject to the old guidelines, and there undo remains the norm. But the more exciting projects these days are for the web and for mobile devices. Undo options are common on Facebook, but not on Google Maps, Tumblr, Yelp, Soundcloud, or Netflix. These interfaces assume the user can figure out how to reverse the effects of an action on her own. Similarly, for mobile platforms, undo is not a standard on as many apps as it could be. In its iOS Human Interface Guidelines, covering the iPhone and iPad, the emphasis is on a simple and clean interface, at the expense of features or productivity; forgiveness is no longer a design principle (Apple 2011). For the web, there are no authoritative interaction or interface guidelines. The innovations in interface design that Apple once blocked with usability studies are now

routine. Designers are less impressed by one set of quasi-scientific research, less concerned with naïve users, and more excited about distinguishing their product and producing the solution they think will work best. The dictates of fashion and user testing keep major websites similar, but a huge amount of variation exists in the margins of the Internet, most of it without undo.

### **Conclusion: What Undo Does**

The undo command was reinvented, developed, and carried on between programs because the *feature* was good, not because the idea was good. Van Dam's concept of micro-backups, or Kanerva's of never doing the same job over again, did not come along. The feature's history shows that it was adopted for different reasons at different stages and by different institutions. 1950s and 1960s visions of an automated environment had no need for an undo command because computers replace operators. As small research projects brought Licklider's alternative vision of a human-computer symbiote to life, various forms of error correction prospered. Undo's problematization of the time of user action eventually replaced other techniques because of its success in the software industry at a time when personal computing overtook automation. Because Apple and Microsoft, who got undo from PARC, were able to set the standard, both informally through dynamics of the software market and, later, formally through interface guidelines, we now see undo as a standard feature with historical origins in a few squirrely research projects.

The undo command materialized a previously obscure function of temporality by applying reversal to actions which were rendered as objects, material only in the

symbolic register of a software system. The undo architecture stored and represented whatever the user did on the computer as a series of discrete commands with specific parameters. The system enforced this representation of time by giving the operator a command that would navigate backward through this stack. Undoing provided users control over a way temporality functions that was quite powerful within the editing environment. The inevitability of mistakes, series of actions constituting the recent past, future of all available commands, emotional state of the user, and calm present of wasted cycles connected together in this practice of time. The computer and user enacted this together; the time had great power within the editing environment, but seemed to stop at the edge of the screen.

The practice fit rather comfortably into the times around it. In the authoritative terms of management's time practices, undoing accelerated training and perhaps increased efficiency. For the time of one attempting to use a computer to get something done, undoing made some parts of the recent past accessible and had an affective value that improved the operator's side of the work, rather than the machine's. The practice of real time computing became more approachable with the addition of undoing, allowing it to be adopted more places more quickly. For the computer's own practices of time, undoing was another bit of software code to be executed as easily as any other. Undoing is uncanny, relative to common sense notions of time (Cudmore 2004; Browning 2008). But the fact that the user could, within the confines of a program, reverse the flow of time did not amount to a real problem as long as it worked well with other times. The time practice enabled by undo

strengthened some related times (a form of influence). But, much more often, it was ignored by other times—the working calendar, for example, fails to translate undo—and this benign negligence allowed it to spread.

Thus, other practices of time were not the primary motors of undo's spread. Undoing appeared as a minor feature common to a number of particular programs and systems. As a detail in a larger picture, the fate of the command rested in many specific systems and programs, which were themselves subject to forces that have in some ways changed little.

The many historical trajectories nudging undo to standardization can be restated as a few ways that the command, in a larger package, was found useful. These uses remain relevant today. First, having an undo feature made computers easier to use. Although the command was not itself intuitive, it helped make software more user-friendly and made variations or innovations in software more tolerable to users. The command had become standard by the mid-1980s, which was long after corporate and government backbones of institutional life had adopted computers, but just before personal computing proliferated to homes, schools and workplaces in the cultural industries. Today, apps and sites are simpler and have fewer commands. Quite often, undo is not one of them.

Second, undoing made it easier to brainstorm, create, and edit at the same time. This combination of tasks into a single location made the computer appear more powerful, because it could host the work of a number of other processes. Along with copy, paste, save, load, new window, fullscreen, find, exit and other standard

commands, undo defined an archetypal experience of computer use that could be grasped and applied in seemingly limitless ways. Once you have mastered the system and gained some computer literacy, new projects become viable. In many cases, workers took on a wider range of jobs than ever before and were expected to respond to work on shorter, more ad hoc schedules (Riggs 2004, 71). This valorization of wearing many hats has intensified. The recent recession has driven employers to value a candidate's versatility to help expand into new types of business and hire one person to do what would have previously been more than one job (Testa 2010; Casserley 2012).

Third, because the computer user could more easily undo mistakes and more easily test out ideas, the undo function helped make computers seem superior to the analog technologies they replaced. We take for granted now that any document can be changed around quite easily, but this was much less true for those editing film with scissors, photographs in darkrooms, or documents on typewriters. Undoing, and other affordances that made time into another interactive element, provide an added value for those making purchasing decisions. Undo and interactive time have definite curb appeal. Digitally Video Recorders are a clear example of this: digitally recording live television for later playback on a small computer yields several advantages that are easy to advertise, whether they are actually desirable or not. Hype has been an important force for much of the history of computers, though today the simple undo command has lost its luster.



Fourth, the command helped reduce reliance on feminized, low-pay, low-status jobs that seemed, to those above, to be better done away with (Garson 1988).

Ultimately, it remains unclear whether such office automation helped or hurt women in the workforce because women soon found information technology jobs and would exit the often-crappy job of typist (J. F. Kraft and Siegenthaler 1989). The issue exceeds the scope of this project. Systems explicitly replaced women, but may have implicitly displaced other groups, such as the elderly or working class. Many women were displaced early on, but those who learned typing skills as training for a job as a typist and secretary found that they could find work on the computer keyboard instead of the typewriter (Riggs 2004, 16).

Finally, the undo command helped empower knowledge workers. Male genius, augmented by appropriate computer technology, would be able to do more, if it could undo a little as well. Even if undoing saved the user no time at all, it feels nicer to be assisted by the computer than to have to repeat work or do boring fixes oneself. These were exactly the sorts of sentiments that motivated van Dam, Kanerva, and Teitelman. Undo empowered users by making errors less problematic, increasing one's ability to concentrate and experiment, and by giving even lowly interns the sense they could control and reverse time. Popular culture continues to depict computer technology as a source of great power. Aside from works explicitly portraying computers as the ultimate power in the universe, such as *The Matrix* and *Tron: Legacy*, many other movies, such as *Jurassic Park*, *The Dark Knight* and *Skyfall*, take it as a given that the

power of computers is hard to match. To a limited extent, software can enact these fantasies.

Undoing made computers easier to use, more versatile, seemingly more powerful, better able to replace other jobs, and more valuable as tools for knowledge workers. Those using a computer could do more than before, though usually for the benefit of others. This empowerment accumulated huge gains for those whose power was being shared. IBM, Apple, and Microsoft on the one hand, and all those businesses and organizations benefitting from computer use—such as schools, fan clubs, pornographers, police agencies, and financial traders—on the other. Also, empowerment secured users complicity in those large-scale plans to bring computers to new markets. By making computers nicer to use, undo fed the growth of organizations using computers and increased the spread of computerization. It thus also furthered the ability of data representations to be prioritized over that which they represent.

The forces that standardized undo chose a simple command over more powerful ones. Radical systems can allow speculative, branching scenarios, gradual rollback undos, or local (but not system-wide) undos. In some specialized editors still used today, for example, the user can undo previous commands in any order. The *Timewarp* system tracks document changes on a branching timeline, between different users, and allows lines of development to split or recombine (K. Edwards and Mynatt 1997). Why have systems such as *Timewarp* or Vitter's *US&R* not become common? As human-computer interaction discovered, undoing has been useful to users as a

simple, general condition of work, rather than another system to be figured out. Most people using software are inexpert and not using the program in quite the way its designers intended. This is the condition of contemporary computer use, for work or for play. Undoing allows this situation to continue.

Undoing is the tip of the iceberg. It represents one specific combination of conventions and it is itself just one more partial object of software, enabling just one kind of time. This time is interactive, but not all times have been designed to give temporal control to the user, and not all times of our computer saturated environment were designed as times.

Software is built up of partial objects (Manovich 2008, 83): an undo command, a paste command, an edit menu, a window, procedures for recognizing the double clicking of a mouse, keyboard shortcut keys, and resource allocation techniques for graphics. Each of these partial objects travels between programs with popularity, prototypes, programmers, and designers. Each building block has its own history wherein it was produced in many different implementations. Each faces its own challenges in the evolution of splendor. At certain technical levels, these pieces function similarly: they depend on electric power, they are stored in binary code, they are written with computationalist logic. But each piece does something different from the other parts and with them.

These partial objects relate to each other concretely. This is how undo was able to find a niche in so many programs. But it is not the only partial object of software that has found a regular usefulness because of the time it enables. A quick processor, a

large and reliable memory, and a nearly instant connection to the Internet all host families of time conventions that put temporality to work in embodied practices. Compared to this teeming mass of ways time works in computers, interactive time can actually seem quite small. It is only one kind of designed time and only some times were designed intentionally. We have come to live in many other times that also have a basis in the computer. The sad fact that personal information does not go away, the happy fact that weather and news are available at any time of the day, and the tedious fact that computers work in cycles and loops are also evidence from ordinary experience of these other times. By turning to this broad horizon of subtending and parallel conventions of time, interactive time can be understood relative to the computer conventions that enable it, and that do quite a bit else.

### References

- “A Survey of Airline Reservation Systems.” 1962. *Datamation*, June.
- Abowd, Gregory D., and Alan J. Dix. 1992. “Giving Undo Attention.” *Interact. Comput.* 4 (3) (December): 317–342. doi:10.1016/0953-5438(92)90021-7.
- Adam, Barbara. 1995. *Timewatch: The Social Analysis of Time*. Cambridge, Mass: Polity Press.
- Allan, Roy A. 2001. *A History of the Personal Computer*. Allan Publishing.  
[http://www.archive.org/details/A\\_History\\_of\\_the\\_Personal\\_Computer](http://www.archive.org/details/A_History_of_the_Personal_Computer).
- Apple. 2009. *Apple Human Interface Guidelines: Human Interface Design*.  
<http://developer.apple.com/mac/library/documentation/UserExperience/Conceptual/AppleHIGuidelines/OSXHIGuidelines.pdf>.
- . 2011. “iOS Human Interface Guidelines: Introduction”. Apple.  
[http://developer.apple.com/library/ios/#documentation/userexperience/conceptual/mobilehig/Introduction/Introduction.html#//apple\\_ref/doc/uid/TP40006556-CH1-SW1](http://developer.apple.com/library/ios/#documentation/userexperience/conceptual/mobilehig/Introduction/Introduction.html#//apple_ref/doc/uid/TP40006556-CH1-SW1).

- Apple Computer. 1987. *Apple Human Interface Guidelines: The Apple Desktop Interface*. Reading Mass.: Addison-Wesley.
- “Apple Writer II User’s Manual.” 1981. Apple.
- Atkinson, Bill. 1980. “Lisa User Interface Standards Document”. Apple Lisa Computer Info 420. Apple.
- Atkinson, Bill, and Andy Hertzfeld. 2004. “Oral History of Andy Hertzfeld and Bill Atkinson” Interview by Grady Booch. Computer History Museum.  
[http://archive.computerhistory.org/resources/access/text/Oral\\_History/102658007.05.01.acc.pdf](http://archive.computerhistory.org/resources/access/text/Oral_History/102658007.05.01.acc.pdf).
- Avner, RA, and Paul Tenczar. 1969. “The TUTOR Manual”. CERL Report X-4. UIUC: Urbana, IL.  
[http://eric.ed.gov/ERICWebPortal/Home.portal?\\_nfpb=true&ERICExtSearch\\_SearchValue\\_0=avner+Tenczar&ERICExtSearch\\_SearchType\\_0=au&\\_pageLabel=ERICSearchResult&newSearch=true&rnd=1212929173503&searchtype=basic](http://eric.ed.gov/ERICWebPortal/Home.portal?_nfpb=true&ERICExtSearch_SearchValue_0=avner+Tenczar&ERICExtSearch_SearchType_0=au&_pageLabel=ERICSearchResult&newSearch=true&rnd=1212929173503&searchtype=basic).
- Barnet, Belinda. 2010. “Crafting the User-Centered Document Interface: The Hypertext Editing System (HES) and the File Retrieval and Editing System (FRESS)” 4 (1).  
<http://www.digitalhumanities.org/dhq/vol/4/1/000081/000081.html>.
- Bergin, Thomas J. 2006. “The Proliferation and Consolidation of Word Processing Software: 1985-1995.” *IEEE Annals of the History of Computing* 28 (4) (October 1): 48–63.
- Berners-Lee, Tim. 1999. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web By Its Inventor*. 1st ed. San Francisco: HarperSanFrancisco.
- Berry, R. E. 1988. “Common User Access—A Consistent and Usable Human-computer Interface for the SAA Environments.” *IBM Systems Journal* 27 (3): 281–300. doi:10.1147/sj.273.0281.
- Berry, R. E, and C. J Reeves. 1992. “The Evolution of the Common User Access Workplace Model.” *IBM Systems Journal* 31 (3): 414–428. doi:10.1147/sj.313.0414.
- Bødker, Susanne. 1989. “A Human Activity Approach to User Interfaces.” *Hum.-Comput. Interact.* 4 (3) (September): 171–195. doi:10.1207/s15327051hci0403\_1.

- Bolz, Roger William. 1969. "Re-education for an Age of Automation." In *Automation and Society*, edited by Ellis L Scott and Roger William Bolz, 193–203. Center for the Study of Automation and Society.
- Borzo, Jeanette. 1993. "Mac Image Editing Takes a Quantum Leap; Live Picture Heralds Speed Boost." *Info World*, August 9.
- Brainerd, Paul. 2005. "Brainerd (Paul) Oral History" Interview by Suzanne Crocker. Computer History Museum.  
[http://www.computerhistory.org/collections/accession/102657986\\_1](http://www.computerhistory.org/collections/accession/102657986_1).
- Brown University Department of Computer Science. 1976. *Hypertext: An Educational Experiment in English and Computer Science at Brown University*. Brown University.
- Browning, Guy. 2008. "Weekend: Starters: HOW TO... UNDO THINGS." *The Guardian (London) - Final Edition*, August 9, sec. GUARDIAN WEEKEND PAGES; Pg. 10.
- Bush, Vannevar. 1945. "As We May Think." *The Atlantic*, July.  
<http://www.theatlantic.com/magazine/archive/1945/07/as-we-may-think/3881/>.
- Cabitza, Federico, and Marco Loregian. 2008. "Much Undo About Nothing?: Investigating Why Email Retraction Is Less Popular Than Apologizing." In *Proceedings of the 5th Nordic Conference on Human-computer Interaction: Building Bridges*, 431–434. NordiCHI '08. New York, NY, USA: ACM.  
doi:10.1145/1463160.1463212. <http://doi.acm.org/10.1145/1463160.1463212>.
- Calta, P. Adrian Z. 1984. "Review Responses." *Info World*, February 6.
- Campbell-Kelly, Martin. 2003. *From Airline Reservations to Sonic the Hedgehog: A History of the Software Industry*. Cambridge Mass.: MIT Press.
- Card, Stuart K, Thomas P Moran, and Allen Newell. 1983. *The Psychology of Human-Computer Interaction*. Hillsdale, N.J.: L. Erlbaum Associates.
- Carmody, Steven, Walter Gross, Theodor H. Nelson, David Rice, and Andries van Dam. 1969. "A Hypertext Editing System for the /360." In *Pertinent Concepts in Computer Graphics Proceedings.*, edited by Michael Faiman and Jurg Nievergelt, 291–330. Urbana: University of Illinois Press.
- Casserley, Meghan. 2012. "The 10 Skills That Will Get You Hired In 2013 - Forbes." *Forbes*, December 10.  
<http://www.forbes.com/sites/meghancasserly/2012/12/10/the-10-skills-that-will-get-you-a-job-in-2013/>.

- Ceruzzi, Paul. 2003. *A History of Modern Computing*. 2nd ed. Cambridge Mass.: MIT Press.
- Cudmore, Doug. 2004. "Ctrl+Z Defies Reality." *The Toronto Star*, July 31, sec. LIFE; Pg. L01.
- Deleuze, Gilles. 1999. *Foucault*. New York: Continuum.
- "Desktop Publishing." 1986. *The Seybold Report on Desktop Publishing*, December 8.
- Dix, Alan J., Roberta Mancini, and Stefano Levialdi. 1996. "Alas I Am Undone - Reducing the Risk of Interaction?" In Imperial College London.
- Drucker, Peter. 1959. *The Landmarks of Tomorrow*. New York: Harper and brothers.
- . 1999. "Knowledge-Worker Productivity: The Biggest Challenge." *California Management Review* 41 (2): 79–94.
- Edwards, Keith, and Elizabeth Mynatt. 1997. "Timewarp: Techniques for Autonomous Collaboration." In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, edited by Steven Pemberton, 218–225. Atlanta, Georgia.  
<http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.15.3409>.
- Edwards, Paul N. 1990. "The Army and the Microworld: Computers and the Politics of Gender Identity." *Signs* 16 (1) (October 1): 102–127. doi:10.2307/3174609.
- Elliott, W. D., A. van Dam, and W. A. Potas. 1971. "Computer Assisted Tracing of Text Evolution." In , 0:533. Los Alamitos, CA, USA: IEEE Computer Society. doi:<http://doi.ieeecomputersociety.org/10.1109/AFIPS.1971.84>.
- Eran, Daniel. 2006. "The Apple Market Share Myth." *Roughly Drafted*. July 21.  
<http://www.roughlydrafted.com/RD/Home/D579148C-8563-4FFB-8E97-C2613215F98E.html>.
- "ERMA: Electronic Recording Machine, Accounting." 1955. Menlo Park, California: SRI.
- Fisher, A. W, and J. L McKenney. 1993. "The Development of the ERMA Banking System: Lessons from History." *IEEE Annals of the History of Computing* 15 (1): 44–57. doi:10.1109/85.194091.
- Freeze, Ken. 1986. "Wang Enters PC Market with Word Processor." *Info World*, June 23.

- Fried, Louis. 1982. "Nine Principles for Eergonomic Software." *Datamation*, November.
- Friedman, Ted. 2005. *Electric Dreams : Computers in American Culture*. New York: New York University Press.
- Fronk, Robert L. 1980. "Driving Forces for Office Automation: The Effects of Social, Economic, Business, and Technical Forces". Automated Office Overview. CPS-IDC.
- Garson, Barbara. 1988. *The Electronic Sweatshop: How Computers Are Transforming the Office of the Future into the Factory of the Past*. New York: Simon & Schuster.
- Glenn, Evelyn Nakano, and Roslyn L. Feldberg. 1977. "Degraded and Deskilled: The Proletarianization of Clerical Work." *Social Problems* 25 (1) (October 1): 52–64. doi:10.2307/800467.
- Goodwin, Nancy C. 1987. "Functionality and Usability." *Commun. ACM* 30 (3) (March): 229–233. doi:10.1145/214748.214758.
- Green, Ted. 2011. "20 Years of VEDIT." *VEDIT: The Universal File Editor*. Accessed November 30. <http://www.vedit.com/20Years.htm>.
- Haigh, Thomas. 2001. "Inventing Information Systems: The Systems Men and the Computer, 1950-1968." *The Business History Review* 75 (1) (April 1): 15–61. doi:10.2307/3116556.
- . 2006. "Remembering the Office of the Future: The Origins of Word Processing and Office Automation." *IEEE Annals of the History of Computing* 28 (4) (December): 6–31. doi:10.1109/MAHC.2006.70.
- Head, Robert V. 1963. "Impact of Real-Time Systems Upon Banking". New York City: IBM Systems Research Institute.
- Heckel, Paul. 1982. "Communication Is the Key." *Info World*, August 9.
- Henson, Jim. 1967. *Paperwork Explosion*. [http://youtu.be/\\_IZw2CoYztk](http://youtu.be/_IZw2CoYztk).
- Hiltzik, Michael. 1999. *Dealers of Lightning: Xerox PARC and the Dawn of the Computer Age*. 1st ed. New York: HarperBusiness.
- Horn, Bruce. 1996. "On Xerox, Apple and Progress". Foklore.org. [http://www.foklore.org/StoryView.py?project=Macintosh&story=On\\_Xerox,\\_Apple\\_and\\_Progress.txt](http://www.foklore.org/StoryView.py?project=Macintosh&story=On_Xerox,_Apple_and_Progress.txt).



- Howard, Steve, Judy Hammond, Gitte Lindgaard, Roberta Mancini, Alan J. Dix, and Stefano Levialdi, ed. 1997. "Dealing with Undo." In *INTERACT '97: Proceedings of the IFIP TC13 Interantional Conference on Human-Computer Interaction*. London, UK, UK: Chapman & Hall, Ltd.
- Humphrey, W.S. 2002. "Software Unbundling: a Personal Perspective." *Annals of the History of Computing*, *IEEE* 24 (1): 59–63. doi:10.1109/85.988582.
- IBM Corp. 1989. "Common User Access: Basic Interface Design Guide". SC26-4583-00. Systems Application Architecture. IBM.
- "IGE Presentation." 1962. BPO 62-079. GE Computer Department. Charles Babbage Institute.
- Kay, Alan. 2004. "First Courses in Computing Should Be Child's Play" presented at the OOPSLA 04, October 26, Vancouver, Canada.  
<http://www.cs.uni.edu/~wallingf/miscellaneous/alan-kay/turing-transcript.html>.
- Kidd, Alison. 1994. "The Marks Are on the Knowledge Worker." In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 186–191. CHI '94. New York, NY, USA: ACM. doi:10.1145/191666.191740.  
<http://doi.acm.org/10.1145/191666.191740>.
- Kita, C. I. 2003. "J.C.R. Licklider's Vision for the IPTO." *IEEE Annals of the History of Computing* 25 (3) (September): 62– 77. doi:10.1109/MAHC.2003.1226656.
- Kraft, Joan F., and Jurg K. Siegenthaler. 1989. "Office Automation, Gender, and Change: An Analysis of the Management Literature." *Science, Technology & Human Values* 14 (2) (April 1): 195 –212. doi:10.1177/016224398901400204.
- Kraft, Philip. 1987. "Computers and the Automation of Work." In *Technology and the Transformation of White Collar Work*, edited by Robert E Kraut, 99–111. Psychology Press.
- Kučera, Henry, and W Nelson Francis. 1967. *Computational Analysis of Present-day American English*,. Providence: Brown University Press.
- Lampson, Butler. 1979. "Bravo Manual." In *Alto User's Handbook*, 31–62. Xerox PARC. <http://www.bitsavers.org/pdf/xerox/alto/AltoUsersHandbook.pdf>.
- Landow, George P. 1990. "Popular Fallacies About Hypertext." In *Designing Hypermedia for Learning*, edited by David H Jonassen and Heinz Mandi, 39–60. London, UK: Springer-Verlag.  
<http://dl.acm.org/citation.cfm?id=115547.115543>.

- Lane, David M., H. Albert Napier, S. Camille Peres, and Aniko Sandor. 2005. "Hidden Costs of Graphical User Interfaces: Failure to Make the Transition from Menus and Icon Toolbars to Keyboard Shortcuts." *International Journal of Human-Computer Interaction* 18 (2): 133–144.  
doi:10.1207/s15327590ijhc1802\_1.
- Laurel, Brenda. 1991. *Computers as Theatre*. Reading Mass.: Addison-Wesley Pub.
- Licklider, J. C. R. 1960. "Man-Computer Symbiosis." *IRE Transactions on Human Factors in Electronics* HFE-1 (March): 4–11.
- Licklider, J.C.R. 1988. "Oral History Interview with J. C. R. Licklider" Interview by Jack Aspray and Arthur Norberg. University of Minnesota.  
<http://purl.umn.edu/107436>.
- Light, Jennifer S. 1999. "When Computers Were Women." *Technology and Culture* 40 (3): 455–483.
- Lingis, Alphonso. 2005. *Body Transformations: Evolutions and Atavisms in Culture*. Routledge.
- Lombardi, John. 1986. "Word Processing and the Office: Selecting the Right Package." *Info World*, February 24.
- Losh, Elizabeth. 2009. *Virtualpolitik: An Electronic History of Government Media-Making in a Time of War, Scandal, Disaster, Miscommunication, and Mistakes*. Cambridge Mass.: MIT Press.
- Manovich, Lev. 2001. *The Language of New Media*. 1st MIT Press Pbk. Ed. The MIT Press.
- . 2008. *Software Takes Command*.  
<http://lab.softwarestudies.com/2008/11/softbook.html>.
- Markoff, John. 1983. "Microsoft Word." *Info World*, December 12.
- Mather, Mark. 2007. "Closing the Male-Female Labor Force Gap - Population Reference Bureau". Population Reference Bureau.  
<http://www.prb.org/Articles/2007/ClosingtheMaleFemaleLaborForceGap.aspx>.
- Merton, Robert K. 1961. "Singletons and Multiples in Scientific Discovery: A Chapter in the Sociology of Science." *Proceedings of the American Philosophical Society* 105 (5) (October 13): 470–486.
- Meserve, Bill. 1983. "The PC War of '84". International InfoTran Profile. Arthur D Little Decision Resources. Charles Babbage Institute.

- Meserve, Bill, and Elizabeth S. Wright. 1981. "The Outlook for the U.S. Personal Computer Industry". Arthur D. Little Inc.
- "Microsoft Multiplan: Electronic Worksheet for CP/M-80." 1982. 8901-100-01b. Bellevue, WA: Microsoft Corporation.
- Miller, Michael J. 1986. "Micropro's Easy Hasn't Caught Up To Worstar." *Info World*, June 23.
- Moran, Chuk. 2010. "Playing with Game Time: Auto-Saves and Undoing Despite the 'Magic Circle'." *Fibreculture* (16) (July).  
<http://sixteen.fibreculturejournal.org/playing-with-game-time-auto-saves-and-undoing-despite-the-magic-circle/>.
- . 2013. *Superactually: Micro-essays on Post-ironic Life*. Ropley: Zero.
- Morgall, Janine, and Gitte Vedel. 1985. "Office Automation: The Case of Gender and Power." *Economic and Industrial Democracy* 6 (1) (February 1): 93–112.  
 doi:10.1177/0143831X8561005.
- Neel, John. 2011. "Live Picture: Software That Was Way Ahead of Its Time." *Pixiq*.  
<http://www.pixiq.com/article/live-picture>.
- Nelson, Theodor. 2010. *POSSIPLEX: Movies, Intellect, Creative Control, My Computer Life and the Fight for Civilization: AN AUTOBIOGRAPHY OF Ted Nelson*. First edition. Hackettstown New Jersey: Mindful Press.
- Nelson, Theodore Holm. 1965. "Complex Information Processing: A File Structure for the Complex, the Changing, and the Indeterminate." In , 84–100. ACM Press.  
 doi:10.1145/800197.806036. <http://dl.acm.org/citation.cfm?id=806036>.
- Neudecker, Thomas. 1984. "Review: Apple's Macintosh." *Info World*, March 26.
- Nielsen, Jakob. 1993. *Usability Engineering*. Boston: Academic Press.
- Norman, Donald A. 2002. *The Design of Everyday Things*. New York: Basic Books.
- Oldenziel, Ruth. 1999. *Making Technology Masculine: Men, Women and Modern Machines in America, 1870-1945*. Amsterdam: Amsterdam University Press.
- Petrosky, Mark. 1985. "Word Processor Users Get More for Less." *InfoWorld*, April 29.
- Phister, Montgomery. 1989. "Quotron II: An Early Multiprogrammed Multiprocessor for the Communication of Stock Market Data." *IEEE Annals of the History of Computing* 11 (2): 109–126.

- Pieslak, Raymond F. 1974. "Magnetic Tape Selectric Typewriter." Rutgers State University, Curriculum Lab.  
<http://www.eric.ed.gov/ERICWebPortal/contentdelivery/servlet/ERICServlet?accno=ED112083>.
- Pold, Søren. 2008. "Button." In *Software Studies: A Lexicon*, edited by Matthew Fuller. The MIT Press.
- Potter, Robert G., and James M. Sakoda. 1966. "A Computer Model of Family Building Based on Expected Values." *Demography* 3: 450.  
 doi:10.2307/2060170.
- Prusky, Jonathan. 1979. "FRESS Resource Manual Release 9.1". Brown University. Computer History Museum.
- Raskin, Jef. 2000. *The Humane Interface: New Directions for Designing Interactive Systems*. Reading Mass.: Addison-Wesley.
- "Research Memorandum: User Spending Forecast." 1981. IDC 2202. Framingham, Massachusetts: International Data Corporation - Corporate Planning Service.
- Riggs, Karen. 2004. *Granny @ Work: Aging and New Technology on the Job in America*. New York: Routledge.
- Roth, Michael S. 1981. "Foucault's 'History of the Present'." *History and Theory* 20 (1) (February 1): 32–46. doi:10.2307/2504643.
- Ruina, Jack. 1989. "Oral History Interview with Jack P. Ruina" Interview by Jack Aspray. University of Minnesota. <http://purl.umn.edu/107614>.
- Scharf, Sara. 2009. "Multiple Independent Inventions of a Non-Functional Technology: Combinatorial Descriptive Names in Botany, 1640-1830." *Spontaneous Generations: A Journal for the History and Philosophy of Science* 2 (1) (January 26): 145.
- "Science: The Thinking Machine." 1950. *Time*, January 23.  
<http://www.time.com/time/magazine/article/0,9171,858601,00.html>.
- Shapiro, Ezra. 1984. "A First Look at Dayflo." *Byte*, March.
- Shneiderman, Ben. 1986. *Designing the User Interface: Strategies for Effective Human-computer Interaction*. Reading, MA: Addison-Wesley.
- Simonton, D. K. 1986. "Stochastic Models of Multiple Discovery." *Czechoslovak Journal of Physics* 36 (January): 138–141. doi:10.1007/BF01599747.

- Sutherland, Ivan. 1963. "Sketchpad: A Man-Machine Graphical Communication System". MIT.
- Tak, Susanne, Piet Westendorp, and Iris van Rooij. 2013. "Satisficing and the Use of Keyboard Shortcuts: Being Good Enough Is Enough?" *Interacting with Computers* (February 18). doi:10.1093/iwc/iwt016.  
<http://iwc.oxfordjournals.org/content/early/2013/02/17/iwc.iwt016>.
- Taylor, K. 1990. "IBM Systems Application Architecture: Common User Access from First Principles." *Computing & Control Engineering Journal* 1 (3) (May): 123–127.
- "Technology: The Cybernated Generation." 1965. *Time*, April 2.  
<http://www.time.com/time/magazine/article/0,9171,941042,00.html>.
- Teitelman, Warren. 1966. "PILOT: A Step Toward Man-Computer Symbiosis". MIT.  
<http://handle.dtic.mil/100.2/AD638446>.
- Tesler, Larry, and Chris Espinosa. 1997. "Origins of the Apple Human Interface". Lecture October 28, Computer History Museum, Mountain View, CA.  
[http://www.computerhistory.org/events/lectures/appleint\\_10281997/appleint\\_xscript.shtml](http://www.computerhistory.org/events/lectures/appleint_10281997/appleint_xscript.shtml).
- Testa, Bridget. 2010. "Multiskilled Employees Sought as Versatility Becomes a Workplace Virtue." *Workforce Management*, September 20.  
<http://www.workforce.com/article/20100920/NEWS02/309209998/multiskilled-employees-sought-as-versatility-becomes-a-workplace-virtue>.
- Tether, Tony. 2008. *Statement by Dr. Tony Tether Director Defense Advanced Research Projects Agency*. Washington D.C.
- "The U.S. Taxpayer: Due, Blue, and 97% Pure." 1962. *Time*, April 13.  
<http://www.time.com/time/magazine/article/0,9171,827250,00.html>.
- Thovtrup, Henrik, and Jakob Nielsen. 1991. "User Interface Standards: How to Evaluate." *Useit.com*. <http://www.useit.com/papers/standards.html>.
- Tognazzini, Bruce. 1985. "The Apple II Human Interface Guidelines". Apple Inc.
- . 2003. "First Principles of Interaction Design". asktog.com.  
<http://www.asktog.com/basics/firstPrinciples.html>.
- Tsang, Cheryl. 2000. *Microsoft First Generation: The Success Secrets of the Visionaries Who Launched a Technology Empire*. New York: Wiley.

- Van Dam, Andries, and David E. Rice. 1971. "On-line Text Editing: A Survey." *ACM Comput. Surv.* 3 (3): 93–114. doi:10.1145/356589.356591.
- Van Dam, Andy, and Rosemary Simpson. 2011 Interview by Chuk Moran.
- "Visicalc: A Visible Calculator For the Apple II: Reference Card." 1979. Sunnyvale, CA: Software Arts, Inc.
- Vitter, Jeffrey Scott. 1984. "US&R: A New Framework for Redoing." *IEEE Software* 1 (4) (October): 39–52. doi:10.1109/MS.1984.229460.
- Watt, Peggy, and Christine McGeever. 1985. "Macintosh Vs IBM PC At One Year; Compared to IBM PC's First Year, Mac Owners Are Starving for Software." *Info World*, January 7.
- Williams, Gregg. 1983. "The Lisa Computer System." *Byte*, February.
- "Word Processing Reference Manual." 1983. 700-7611B. Lowell, MA: Wang Laboratories.
- "Writing and Editing Tools." 1986. *The Sybold Report on Desktop Publishing*, November 3.

## V. Architecture of Time

Undo makes one time. By studying it, the relaxed and reversible time of personal computing becomes easier to understand. Focusing on military funding, competition in the software industry, prerogatives of small academic labs, human interface guidelines, and other molar factors, we can see the contingency of its development and the way it expressed the social field in which it developed. But computers have created many other times too. Twitch gaming, robots with lightning fast reactions, one's own faulty digital archive, a world of consumer services available 24/7 online, an acceleration of social processes that follows computerization, and personal information that cannot be deleted are also products of new times supported by software.

Undo is one offshoot in a thick branch of conventions. The convention of undoing grew out of conventions of software, firmware, and hardware and it supported new ones in interface design and usability. Conventions are patterns, usually approximate, but sometimes very strict. Though some are well described in technical terms, they are not only technical. They have other aspects: social, cultural, literary, political, artistic, aesthetic, historical, and temporal.

Many basic parts of regular computer activity function as conventions of time. While two different ways to organize menus or render text fields might not make much difference to the user's experience of time, some things do. Though they may be called hardware specifications, manufacturing standards, services, or network protocols, they join functions of temporality together into a repeated behavior, make

certain functions available and others less, and give specific expressions to functions of temporality that existed before computers, often augmenting and redefining them rather completely. They have changed what we think of as the contemporary and given a new air of virtuality to the term access.

This chapter explores some of the subtending *temporal conventions* of computing and their formative role in the genesis of well known computer-mediated practices of time—particularly interactive ones. Where my study of undo began with a feature, this section begins with the underlying engineering that can create time practices such as undo. The work of infrastructure though crucial in many domains, tends to be invisible (Star 1999). The infrastructure of computing includes the platform: “whatever the programmer takes for granted when developing, and whatever, from another side, the user is required to have working in order to use particular software” (Montfort and Bogost 2009, 2). The platform enables higher levels of construction. These dependencies are not just historical: at run time, undo is part of a complex, mostly hierarchical, structure of conventions.

The practices of time that can come from these layers do not always show the marks of their dependencies. Though others have treated this effacement of low level functionality as a case of ideological mystification (Chun 2006, 20–21), each particular time makes careful use of some conventions in order to accommodate different forms of activity for the user. This is more akin to deliberate duplicity in Goffman than political hegemony in Gramsci. Designers intentionally show a front to the user, who usually ignores politely what is visible of the backstage. We have



learned not to speak of dropdown menus or Internet load times for the same reason designers have produced appealing abstractions such as buttons to replace more cumbersome realities such as defining variables. Users and designers conspire to simplify mental models of computer use. We want to buy tickets or set an alarm without having to think about the computer mediation these acts involve.

Computer technology is a growing accumulation of conventions, usually explicitly in conversation with only the most proximate work from which it builds. Conventions accumulate because they afford better opportunities for creation. The use of integrated circuits, of digital code for memory, of graphical display, and of processors executing instructions establish some of the terms by which what is built with them will operate. Some describe this in terms of a lock-in that occurs when new designs rely on older conventions that should probably be changed. Data on a hard drive, for example, does not need to be represented as files in folders, but working *without* this convention has become very difficult (Lanier 2010, 7). However, this process of accretion is also how software stands on the shoulders of giants and the actual historical route by which computers have developed from large calculators into tiny multi-function smart phones. Working with analog computers, custom audio formats, rare ports and jacks, paper memory, forgotten run-time libraries, proprietary compilers, and completely original graphics techniques is harder than using existing conventions. It requires work to debug, maintain, and make compatible with other systems. When an organization gives specifications for a convention and promulgates this convention in specific ways, as Apple, IBM, and Microsoft did for undo, it

becomes a standard; this allows widespread interoperability and creates a more hospitable environment for improvements to be made atop the old (Robinson and Cargill 1996).

As a whole, conventions are a pile of rubble, with some connected to others, many potentially relatable, and others quite disconnected. Computer scientists play in this sandbox, dedicating their time to making conventions that are new by modifying, imitating, or combining old ones. Engineers and programmers raid the junkyard to build structures out of conventions, and eventually figure out and rationalize the structure of the dependencies between the conventions they have used. Companies sell the structures made by engineers and programmers as platforms on which other things can be done. Users experience these platforms as tools (or toys) that interact with other partial objects only through conventionalized protocol. For example an image editor can manipulate images if they are of a readable format. Historians can trace connections between conventions, showing how they have grown over the years, but most links have been lost and regrown many times over. Undo is one convention among many variations, an object ready to be used, a part of many structures, a facet of another tool, and a branch in an old growth.

Because this chapter reviews fundamentals to computer hardware and software that are matters of common practice, rather than claims attributable to any one source, it contains relatively few citations. My explanations draw on two textbooks (Hennessy and Patterson 1998; Stallings 2000), many Wikipedia articles, countless explanations offered in different online forums, the stark prose of computer science course

websites, articles and blogs reviewing fundamentals in order to explain more complex concepts, and conversation with people who understand these things better than I do, including Cynthia Taylor, Steve Chekaway, Elizabeth Petrick, Ryan Baker, Graeme Worthy, and Tom Moran. In this journey down the rabbit hole, I was reassured to discover that the endless complexity of computing is not fully understood by anyone.

Although many conventions of time in computers have an influence in our world, three groups stand out as particularly important for their role in computing and for their explanatory power: *processing, memory, and networking*. The processor is the fundamental hardware unit of a computer, doing the actual work of processing symbols that qualifies the machine *as* a computer. Memory has also been fundamental to the computer since the beginning, storing symbols to be processed, the programs that conduct processing, and the output of those processes. Networking is a more recent addition, but has quickly become the basis for the one phenomenon by which our era will be remembered best: the Internet. Together these computer parts have some role in anything that happens on the computer. Thus, the time conventions they establish have some role (usually many overlapping ones) in all that computers do.

There is a reason that the activities of computers are worth talking about and that is the explosive growth by which they have come to saturate the human world. There is a very simple explanation of this growth: Moore's law. Stated originally as an observation (not a law) about integrated circuits in a magazine article, Caltech Professor Gordon Moore wrote, "the complexity for minimum component costs has increased at a rate of roughly a factor of two per year" (Moore 1965). Moore pointed

out that miniaturization was making transistors smaller, denser, and cheaper per unit. Moore looked forward to possibly another ten years of consistent change. Since then, the rule has been broadened to claim that processor power doubles every year—a claim that, surprisingly, has held true for decades.

Miniaturization increased processor speeds for several reasons. First, signals travel at a constant speed, but the distances traveled shrink with the circuit, increasing the rate at which signals arrive at their destinations. Second, the higher density of transistors provides more memory per square inch, which gives processors more resources to work with, allowing increases in effective power. Though these two raw expressions of Moore's original thesis drove progress for decades, they have provided diminishing returns and might have faltered, in a world depending on increased computer power, had researchers not found other frontiers. A third phenomenon supporting Moore's law is that having more transistors actually allows for more sophisticated microarchitecture within the processor. This accomplishes speed gains in various ways (e.g. by executing instructions in parallel, increasing the efficiency with which instructions are executed, and establishing efficient specialties within the processor).

It seems that miniaturizing transistors may have reached a physical limit as the heat they leak becomes greater at smaller sizes (Borkar and Chien 2011). Recent progress in processor power has had less to do with processor speed than with efficient use of multiple processors running concurrently. However, several technologies, including carbon nanotubes, promise to extend miniaturization even further.

What is known as Moore's law may more accurately describe the industrial pressures on research and development of integrated circuits than the specific engineering technique that provides the (expected and required) breakthrough for its decade. Moore's original observation became a powerful myth of inevitability. If the company's lab fails to double computing power in a given year, the chief scientist may need to find another job.

Although apparently only a claim about processing, Moore's law has significant implications for memory and networking too. The pressure on researchers to continually provide increased power have worked, and the rate of growth in hard drive storage has actually outpaced that of processors (Walter 2005). Miniaturization lets memory hold more data in less space. More recently, the exact product of Moore's law—more transistors in less space—has become the basis for solid-state memory (e.g. flash drives) that use transistors as a storage medium. The increase in processing power described by Moore's law has increased the processing power of every individual computer, while increases in memory have made each able to store more information on its own. These have directly contributed to the power of networking, as stronger computers have more to offer the network and can get more out of it. Computers with fast processors and large memories let people share pictures, collaborate on documents, find out what to order at a restaurant they've never been to, and do all the other charming things advertised by spunky web companies.

In essence, the human race has recently specialized in one form of technology that is the technique by which it handles a still-increasing range of activities. In most

cases, this technology would be a very inefficient solution were we not already so far along in our use of it. To watch videos other people made, for example, could have been done more directly using the existing cable infrastructure. Keeping a todo list or set of contacts can be done quite well on paper. The miniaturization of transistors, development of graphical user interfaces, and refinement of database query algorithms are not the most direct routes to any of these goals. However, because we have the technology, usually developed for some other project, computers can be very good tools for many kinds of situations.

At another level, though, the growth of computing results not from an increase in raw power, but from the production of clever software, an increased range of situations to which computers were brought, and the addition of new peripheral devices such as speakers, microphones, color monitors, mice, and modems. Each application seemed within reach of current computer technology, though some innovation and modifications would be required. Incrementally, then, by using conventions, computers have gotten to each specific place, doing each specific thing they do today.

This chapter outlines conventions of time and applications of these conventions in the three key domains of processing, memory, and networking. For the processor, the basic form of time is a cycle. The faster the cycles, the faster the computer. This structure repeats in the operating system, where the scheduler cyclically doles out units of processor cycles, and then again in software, where loops and iteration in code allow smaller programs to do more things. For memory, the chief

issue is the interplay between the fixed form by which things can be preserved and the movement by which they can be maintained and accessed. Ironically, movement both makes memory very fragile and is the key to digital memory's fabled longevity. For networking, the sequences of data transmission pair with the connection medium to synchronize a seemingly instant worldwide present, in which the more that is connected, the more connection becomes desirable. Each is a case of designed times in which the old movers of time, such as the sun or pace of a regular pendulum or crystal vibration, has given way to modular, configurable routines enacting many times at once. Of these, some give the reins to the user, encouraging her to intelligently modulate temporality's function as she sees fit. Other times condition this power or are up to something else entirely.

### **The Fast Cycles**

Imagine that your computer is running slow. Very slow. The thing stutters playing music, letters appear in spurts following your keystrokes, websites load in steps with very long gaps between them, games are hard to play because the frame rate has dropped to only a few images per second, and your cursor turns into some symbol meant to gently convey the sad fact that you cannot click on anything until the computer works through something you know not what. At such a moment, the processor is trying to process but is stuck waiting while data passes through channels, buffers, processes, and memory. Delays, slowness, pauses that may become final stops, and moments of inaccessibility become your working conditions, if you are working, or the conditions of play, if you are playing. You have become slow.

Inefficient, your productivity might indicate, dim, your play might suggest. But, even now, when the computer is taxed to the extreme, the processor still spends much of its time idle. The processor goes through its regular cycles at its ordinary rate, shifting from a task that requires some waiting to one that does not.

A computer is a programmable device that can carry out a finite set of logical operations. These operations, in theory, could be handled in many different ways; but all contemporary computers use processors that handle operations sequentially in a pre-defined cycle. Alan Turing gave this idea, of a computer carrying out a set of operations, its first, clearest statement in 1937 when he described an automatic machine, now known as a Turing machine. The Turing machine manipulates symbols on a strip of tape according to a table of rules that can change with the machine's state. It reads a symbol and reacts according to the rule set of its present state; it can modify the symbol, change its own state, and/or move to another location on the strip. Imagine this as a menial job. For shelving books in a library, take the next book on the cart, go find the shelf where it belongs, put it in place, check the shelf's own order (fixing it involves a simple subroutine), and move on to the next book. This procedure changes when the shelver's mode switches from working, usually by the advent of a break, empty cart, or the end of a shift. A Turing machine can enact any particular set of operations on its symbols, such as coding a message or searching for all appearances of a symbol along its data strip. A *universal* Turing machine is simply a Turing machine that can simulate any other Turing machine. The logical formulation of a computer does not specify hardware; Turing imagined this work would be done by a



human, a connection that was common for those thinking about computation in the 1930s. Formal protocological specifications, that do not clarify how they should be accomplished, are still the norm in computing and this tells us something about the field. Specifications exist and must be met *by others*, be they hardware components, software programs, or just us humans. The modern computer is a universal Turing machine built mostly out of transistors. At this level of abstraction, though, computing is symbolic manipulation performed in discrete steps according to given rules. The pattern of discrete steps can be thought of as a cycle.

The cycle of a Turing machine is fundamental to the contemporary computer and the practices of times that build from it. Though processors do many kinds of work, they always use the same basic technique of a loop. The processor's process is simple: it fetches data, interprets that data as an instruction and a bit of information that may be the object of the instruction, executes the instructions using the relevant information, and repeats.

The process begins with the processor knowing the address of the next bit of data. It fetches that data. The data consists of two parts: an operation code and some information. The operation code, interpreted according to a set of rules, gives the processor an instruction. The processor has been designed to be able to satisfy these operation codes by its instruction set architecture, which is hardwired in the microprocessor architecture. The instruction it reads may require doing arithmetic, looking up data from disk, writing data back to memory, changing what address shall be fetched next, or a variety of other functions. The default for which address to visit

next is simply the next address in the memory, but this can be changed. When the next address to visit is one that has already been visited, the hardware enacts a software loop. Just before completion, the cycle checks for inputs from other parts of the system. These interrupts could be keystrokes on the keyboard, expected information from the hard disk, a timer ringing, or an error. Then the process repeats.

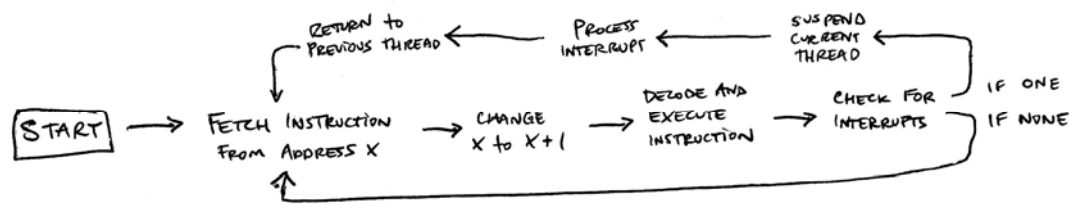


Figure 5.1 The fetch cycle.

If only given one instruction, a processor is little more than a calculator borrowed from a friend for a moment. The power of the processor comes from its reuse. By doing thousands of instructions it can do many transformations on a large data set. This was what computers did as electronic data processors from the 1940s through the 1970s. During this period, few things could be effectively treated as data (symbols manipulated by a Turing machine). Payroll, census, and wind tunnel simulations were popular uses; representing a painting, a movie, a molecular structure, the best way to drive home, a castle full of Nazis, or the sociogram of one's relationships were not. Since that time, it seems increasingly as if everything around us is data or can be represented this way (Shaviro 2003, 248–250). This is the result of increases in computing power won by progress in engineering. By executing more instructions, and by making more transformations to data, the computer can simulate

nearly any occurrence. Booking seats for airlines, editing documents, or simulating explosions all became possible because they could be modeled as data undergoing transformation.

Fast cycles are a crucial design element in the architecture of time. Cycles fit into the design of a time and enable it, in specific ways. Real time services depend on updating information so quickly humans experience the change as continuous. Each update is, in fact, a process executed on a processor that moves old data to a log, reads data from an input source, and displays the new information as the truth of the present moment (Moran 2011). The same principle also drives machine feedback. A real time data feed (that is, a very quickly updating one) can pass along information to a control unit with very fast reaction times. In contemporary robotics, this allows ultra-rapid responses that let machines find equilibrium points for balance and movement. Computer-controlled stabilization lets multicopters turn barrel rolls and maneuver through windows —moving with their center of mass offset from their geometric center (Mellinger et al. 2011; Mellinger, Michael, and Kumar 2010).<sup>1</sup> It is also the means by which the Segway drives: when you lean, it moves to catch up. Similar techniques are used throughout computing, as in digital signal processing responding to live signals, artificial intelligence reacting to changing conditions, attempts to coordinate multiple processors, and time-critical safety systems. Processor cycles

---

<sup>1</sup> In fact, multicopters have been built since 1920, but have not been workable until real time computer-controlled stabilization made them a very convenient UAV platform and RC toy (Roberts et al. 2007). Drones depend on processors.

make available the instantaneous, continuous, and simultaneous for any manipulations performed on what can be represented as data.

The processor does not cycle alone. Turing's abstraction is subject to a further level of representation and control. In modern computing, the *operating system* divides the processor's cycles into a unit of time called a *quantum*, and distributes these to the many concurrent processes running on a system. The operating system is a collection of software that makes the machine's hardware accessible as resources to programs, so they can run on the computer. One of its essential components, the *scheduler*, gives *threads* (sequences of instructions) to the processor, and switches these out. The scheduler does the very important work of ensuring the processor works while waiting for inputs rather than going idle and allowing multiple programs to run on a computer at once. Waiting would be a serious risk for the processor without a scheduler. Many kinds of instructions require information from an input/output device, such as a hard disk drive or the mouse. Relative to the millions of instructions the processor can execute every second, human inputs such as typing are very slow (120 words per minute is only ten characters per second). Waiting on each communication would slow down overall performance considerably. The scheduler also allows multiple processes to run concurrently; by switching between them frequently enough, they appear to run simultaneously. This is the magic of multi-tasking. The scheduler must allocate time between threads fairly, with differential responsiveness, and with efficiency. Efficiency, in this case, means maximum throughput, minimum response times, and the ability to accommodate multiple users.

Each quantum of processor time, given by the scheduler to each process, represents processor cycles as an interchangeable unit of work. This unit also becomes the minimum interval of activity on most computers. To update the computer clock, set in motion a scheduled event (such as an alarm), or make a response, a process needs a quantum of processing time. Generally the quantum is about 10-100 milliseconds—short enough to multitask and give good response times, but not so short that too much time is spent switching between processes.

The quantum functions as the de facto matrix of events within which computer life can take place. In this regard, it is a form of event-based time, not unlike the agricultural cycle of a society that depends on produce. After intensively studying the lunar calendar of Trobriand islanders, Malinowski discovered that he had, in asking after moons and their names, missed the real basis of time-reckoning in the society. Gardening, not astronomy or mythology, gave people a reason to act, a sense of anticipation, a guide for memory, and a system by which to organize events. The clearing of the scrub, when the vine supports are positioned, and the harvest's end were the meaningful events structuring time (Malinowski 1927). Other anthropologists studying time-reckoning have come to similar conclusions; in many societies, it is regular events, rather than abstract measures, that pace social life (Beidelman 1963; Evans-Pritchard 1939). An important insight from this ethnographic work is that social practices of time may depend on a regular kind of event to organize and pace all other events. This is the role of the quantum in the computer.

Like days on a calendar or seconds on a clock, quanta are the ether of moments into which all other events must be translated and on which they must be based. Any scheduled event, any thread of computing, any active process, and any response to user action (or other input) can only happen within a quantum and only when other quanta have finished. Most quanta will simply run a default idle thread, repeatedly executing a halt command, to save power by temporarily reducing the processor's functionality. But all quanta function to pace, separate, and organize other operations run through the processor. To run an alarm, for example, the processor does not check every cycle whether the designated time has arrived; this would be a very busy form of waiting and would waste a considerable portion of the processor's time. Instead, the scheduler provides occasional quanta for a timer-checking service that sees if the current time on the clock ought to trigger any alarms or scheduled actions. The quantum provided to the timer-checking service represents the maximum precision with which the timer can operate, so a single quantum would be the height of exactitude and also the maximum resolution of the system's clock (Etsion, Tsafir, and Feitelson 2003). Quanta allow many things to happen at once, define events, and pace processes, yet they also limit the computer's clock precision. Operating systems have been built in other ways, but the quantum is a highly standard approach to the problem of scheduling.

In addition to the basic hardware level of processor cycles and the operating system's fundamental scheduling loop, cycles have a critical role in code. The importance of loops, cycles, recursion, and iteration in software derives from the

traditional importance of efficiency in the field. At its most naked, programs once needed to be short because there was little space available to store them. The early hacker ethic promotes elegance by simplicity, in part due to this condition on writing (Coleman 2013, 93–122). Programs should therefore reuse as much of their code as possible, handling different situations as minor variations of other situations. This approach to writing, quite different from novelists or screenwriters making newness in place of cliché, has made computers able to do a huge number of things, but mostly by using menus, buttons, and forms that are remarkably similar between applications.

Cycles also make for cleaner code. Computer code had, in many cases, been a text authored and used by a single person, but as software distinguished itself from hardware and grew more complex, a program became a project created by many, used by many more, and translated, modified and evaluated by still others. Programming no longer meant simply instructing hardware how to operate, but writing a text that others would have to read. Simplifying code, adding comments, and organizing operations in a way that would be clear to human readers became a priority. As software came into its own, high-level languages, no longer machine code, enabled an approach called structured programming that prioritized organization into subroutines, block structures (a number of lines of code nested within an operation), and loops. In this climate, Edsger Dijkstra wrote his now famous “Go To Statement Considered Harmful,” stating loud and clear that loops were the more efficient and scientific technique for programming (Dijkstra 1968; Knuth 1974).

Program cycles have become a subtle, but ubiquitous part of contemporary life. What began as efficiencies in design have long since become norms of user experience and a familiar part of how we are treated by the machine. “To live today is to know how to use menus,” (A. R. Galloway 2006, 17) and, among those for whom this rings true, procedures of code and cycles of the computer are all too familiar. Slowly, these terms seem apt not only to describe programming, but everything else in our world. At another moment in history, social realists saw film as an opportunity not to record theatrical performances but to represent the truth of the world as a slow unfolding of unimpressive and quotidian events. Hollywood starlet Mary Astor countered that this could never capture the specificities of individuals lives and missed entirely that, to the person living a life, theirs is a grand story of drama, represented well by theater (Astor 1971, 93). Today, advocates of ubiquitous computing claim that “our lives are built from basic, daily operations,” (Greenfield 33), which can usually be improved by the addition of more tiny computers. Cyclic operations seem a better model for regular life than either high drama or the mundane occurrence of one thing after another. As Ian Bogost puts it,

Because computers function procedurally, they are particularly adept at representing real or imagined systems that themselves function in some particular way—that is, that operate according to a set of processes. The computer magnifies the ability to create representations of processes. (Bogost 2007, 5)

The computer slowly habituates us to experiencing activities as procedures.

David Golumbia, focusing on the vanguard of this change in academic disciplines including linguistics and philosophy, refers to this as the cultural logic of computation:



“the power and universal applicability of computation has made it look to us as if, quite literally, *everything* might be made out of computation” (Golumbia 2009, 19 emphasis in original). In the limitless horizon of this vision, cyclic process is the means by which all events occur; all the temporal variety of occurrence, change, rhythm, and pace seem built up entirely of pulsating mobs of computer cycles.<sup>2</sup>

The practical value of the cycle as a powerful, approachable, and reusable unit of work has made it popular in music. Though repetition has always been important to music, loops driven by the computer cycle have become a dominant feature in the last few decades. In the mid twentieth century, experimentalists used loops of cassette tape to explore new musical possibilities. In the 1970s, DJs invented hip hop by switching back and forth between two copies of the same record, so the break (an interlude between two sections of a song) could keep repeating for several minutes. This beatjuggling encouraged new forms of dance (breaking) and spoken word performance (MCing) during the looped break. Loops play *sound*, not just music, and this has encouraged a focus on rhythm and timbre (Stillar 2005). Today, drum machines, looper pedals, keyboards, and, now, software running on laptops have become key technologies in the production of music. In each of these devices, loops arise from a computer running through a set of instructions and looping back to the start at the end of a sequence. The computer has several advantages for looping.

Computer-based loops require less exertion to perform, are more consistent than

---

<sup>2</sup> Is everything procedural? Hopefully, the afterimage of iteration and cycles can also provoke a concomitant sense of what eludes the cyclic structure. Quite possibly, those who see continuity will produce others who see discontinuity as a reaction to their narrow-mindedness.

beatjuggling on vinyl, and are easier to modify than loops physically assembled with cassette tape. Furthermore, loops based on computer cycles allow for multiple loops to run at different speeds simultaneously (Latartara 2010). The use of loops in music is partially an aesthetic phenomenon, connecting the experimentalism of Terry Riley and Steve Reich to the hip hop DJing of Kool Herc to the futuristic techno of Derrick May and Juan Atkins (Carter 2008). But loops have also been an engineering solution arrived at by untrained musicians to the limits of memory available on early digital media, as was the case in video games (Collins 2007). Computer cycles did not invent the loop in music, but they did make looping considerably easier and thereby encouraged its use, transforming music and establishing hip hop and techno as genres.<sup>3</sup>

The computer cycle is a specific practice of time almost identical at each level. A program organizes instructions into cycles, while the instructions themselves run on a processor handling each in a cycle, subject to the control of a scheduler which puts the processor to work on one process or moves it to another. Unlike cycles of other kinds, these are not tremors that grow in intensity, plateau, and come to an end. There are no fixed matches between processor cycles and quanta or between quanta and processor cycles. There is no intermediary between cycles; the end of one cycle is always immediately the beginning of the next. The cycle of the computer is even,

---

<sup>3</sup> Looping has been important for other forms of art as well, sometimes because of computer's time conventions. Installation video art, for example, relies heavily on looped footage to occupy gallery space. Loops have also proven useful in computer animation and video, though computers are directly reproducing a known technique from cell animation (Manovich 2001, 314–322).

reliable, and calm. But it is also hyperactive, extremely fast, and optimized to meet many different demands.

The cycle, at the level of processor, operating system, and program has a clear and important role in most computer enabled practices of time. The cycle is not interactive, though it is the precondition to a responsive computer. It makes computers fast, able to process many things as data, and uses repetition for efficiency to maximize its power. It let several things happen at once, paces and spaces them as events, and makes the discrete appear continuous. These fact cycles are a key reason that with computer we rarely experience a set pace. The machine is almost always waiting for us.

Anything running on a computer must, at a technical level, use cycles, but cycles do not always have notable temporal influence. Cloud computing, ATMs, and streaming video run on programs and processors that make heavy use of cycles. Yet they react based on network traffic and enigmatic delays, develop technologically at their own rate, and are present to users in a way that has very little clear relation to the cycle.

## **Memory**

What is computer memory? Like many technical terms, the word can be misleading. Computer memory is an abstract space of addressed entries mapped onto a physical space divided into tiny columns and rows. The Turing machine's tape of symbols is one dimensional, but this abstraction can represent many dimensions. The

memory contains many small markings, which are readable as binary digits, but the computer treats these all as clusters of information. The abstract map can be impressed upon a number of specialized media. Optical media (such as CDs and DVDs) usually store information in indentations and flat spaces (pits and lands) melted by a laser into a thin layer of aluminum on a plastic (polycarbonate) disc. On a hard disk drive, bits of data exist physically as varying levels of magnetic charge that can be detected or altered by an electromagnet. In flash memory, every single one or zero represents a very small transistor which is either holding charge or not.<sup>4</sup> RAM stores bits (one or zero) in transistors, holding their charge with a constant flow of electricity; this makes the technology faster to access but means that power loss wipes it clean.

The Turing machine processes symbols. To do this it needs a supply of symbols and a way to retain and modify its own state. The computer is thus a machine for *storing* data as well as *processing* it. Processing happens by recurring cycles but storage depends on memory. Computer memory works by making abstract representations of physical media and rewriting this media to utilize, add to, and modify information processed by software. This rewriting of data makes digital information capable of accurate and durable copies, but it also incurs constant risks of corruption and loss. The question is, for whom?

---

<sup>4</sup> New technologies store multiple bits of information in place of a single bit by piling charges vertically. Multi-level flash memory, like multi-layer optical discs, are good examples of the progressive change that characterize engineering. Relative to certain criteria, more is better.

Computer memory represents highly-engineered media (such as the DVD or flashdrive) as an addressed neighborhood populated by blocks of data. Different levels of the system represent this differently, with the processor using one reference system, applications using another, and users seeing yet another. On the actual medium, the smallest unit of information is binary, though only approximately. At every higher level of abstraction, these units are treated as clusters of greater size and variability, providing the end user with the impression that computer memory is wholly immaterial (Kirschenbaum 2008, 135).

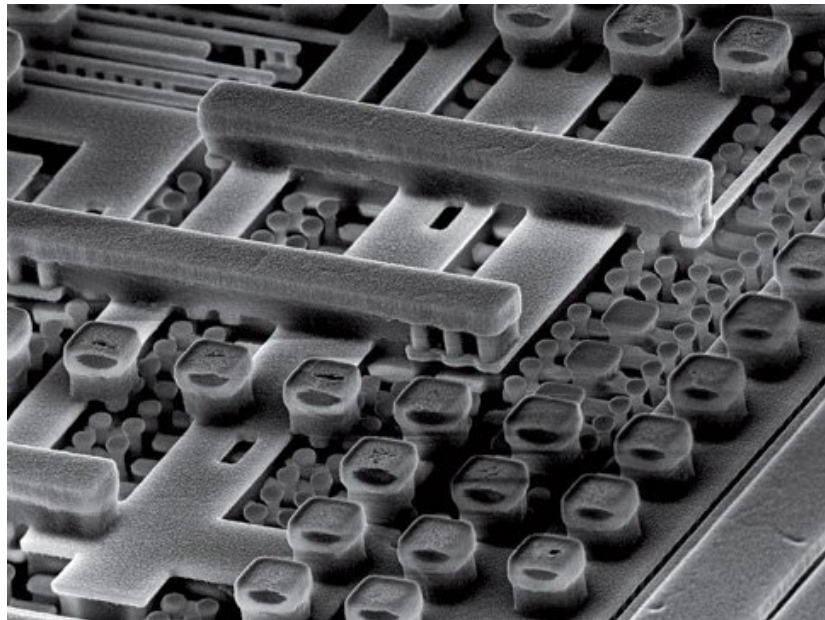


Figure 5.2 A scanning electron microscope reveals the complex structure routing signals from transistors in a piece of flash memory (Aaronson 2008).

Because computers treat memory with a variety of abstractions, software can create multiple types of records and use these in combination. Users tend to see computer memory as primarily a matter of storage space for folder and files. This accords with our usual experience of memory in other media where an object holds

records of the past in just one form, such as words in a book or grooves on a record. Computer memory, however, also involves page files, temporary files, swap space, system files generally invisible to the user, and a whole range of volatile forms of memory that clear themselves only on power off. These kinds of memory allow software to keep other kinds of information that the user usually does not know about. Generally, these operations make software more responsive; for example, when streaming video from the Internet, a computer creates a buffer of video that is ready to play in order to prevent hiccups and uneven playback speed. Most programs use many stacks to record the value of a number of variables and parameters that will shift during a session of use. Undo commands rely on a small collection of memory called the undo stack, which stores the ordered list of actions that could be undone.

Computer memory depends on constant movement between temporarily fixed locations. Just as the processor must load instructions from addresses, programs running on a computer request things from memory, make use of them, create more information to be stored in memory, and hopefully tidy up after themselves by deleting entries they have created but no longer need. To prevent the processor from having to wait, computers use what is called a memory hierarchy. The hierarchy represents a general solution to the problem that faster memory is expensive. Most computers have a large amount of memory that is slow to access, a much smaller amount of faster memory, and a very small amount of extremely fast memory. The concept of the hierarchy is to use the fastest memory for information that changes most quickly, and the slower memories for what is updated less often. At the top of

this hierarchy are registers and caches on the processor, followed by random access memory (RAM), then flash memory, and finally hard drives and other media that depend on mechanical motion. The computer must move data between these forms of memory quickly and often.

Because all memory media depend on movement and operate as if they contain binary information, software can move any information between registers and turn the present state of a process into an item stored on a disk. Memory is fungible. It can be copied between different devices. Any information that goes through the computer can be captured and stored. This is the fundamental reason a computer that can play a DVD can copy the content off the disk, and that anything that can display through a computer can be grabbed and kept. This is a serious problem with copyright enforcement in the digital age: any data that moves through a personal computer for playback can be captured and distributed. Ripping CDs, recording audio streams, downloading images from the web, saving the current state of a video game, and undoing an action in a word processor all rely on the same principle of capturing temporary memory in a more permanent format. In many realms of human experience, the present cannot be frozen and restored at a later moment in such a complete form. But, with computer memory, the living present and a stored file are not so different that the one cannot be turned back into the other. This convention of time can allow users to switch processes between active and static states. This allows piracy and yet is also the basis for undo's ability to restore, video game's ability to save, and operating system's use of hard disk space to supplement RAM.

Memory is fragile. It is tiny, complex, and constantly in motion.

Extraordinarily small and precise variations in physical material (e.g. metal or plastic) are vulnerable to tiny errors that render portions of the medium useless for storage. Keeping every bit in place becomes a surreal challenge when their physical size is less than a billionth of a square inch. At such scales, a grain of sand or a dust mite becomes a large, bulky, dangerous object (which is why hard disk drives are sealed). IBM research suggests that trace amounts of radiation from standard industrial cleaning processes and cosmic rays cascading down from the atmosphere damage one bit per 64 megabytes of RAM per day, though more at higher altitude (Ziegler et al. 1996, 7). Over the course of several years, the risks become substantial. The upgrade-happy industry keeps consumers switching to new storage media, but old hard disk drives die quickly after five years of use (Schroeder and Gibson 2007; Pinheiro, Weber, and Barroso 2007). Physical media aside, the incredible complexity of the computer, and reality that those working on one aspect of it know very little about the rest, ensures data corruption. Database mismanagement, malicious code, software bugs, weaknesses in the operating system, firmware errors in the storage device, and fluctuation of power supply can all cause data corruption (Borisov et al. 2011; Leddy 2003). Finally, the constant movement involved in using memory causes wear that eventually puts storage devices out of commission. The hard disk drive operates by spinning at high speeds (the faster the better) with its electromagnetic head nanometers from the surface, floating on a cushion of air. When the head accidentally hits the plate, due to any number of causes, it scratches it, often destroying the drive



permanently. Optical media, such as CDs and DVDs, are useful because you can pull them out, store them, perhaps mail them across the country, and put them back in another drive. Yet, this exposes them to the dangers of fingerprints, humidity, temperature, prolonged horizontal storage, and tiny pieces of grit sliding across the surface. Although we hope optical media will last for decades, many discs fail in only a few years, particularly consumer-grade writable discs made and sold at the lowest possible cost (Marken 2004; Woolf 2001). For digital memory, the physical medium is a weak point.



Figure 5.3 How would you get data off these old disks?

In addition to these problems, sustained innovations in computing technology may render specific memory formats obsolete. Many have noticed that old movies on VHS and data on floppy drives are practically inaccessible because finding a working player is difficult and transferring to a more contemporary format is even more work.

But these formats are clearly dead. What is more surprising is that early CD-ROM formats are not readable by later CD drives and that memory cards designed for digital cameras from the early 2000s (such as xD) may be nearly impossible to read. MIT Libraries host a website called “Chamber of Horrors: Obsolete and Endangered Media,” tracking the endangerment and extinction of such digital storage media. Beyond changing physical media, some old material depends on proprietary software from a folded company, lost access codes, or a programming language whose documentation and compilers have all been lost. Changes in software technology make such items extremely difficult to maintain in an accessible state over the course of decades (Smith 2005). The basic problem is that, lacking associated software, a file is nothing but a series of bits on a piece of memory; we may lose the correct means of interpreting it (Rothenberg 1995). Physical media, even if intact, may be hard to read within just a few years.

In the long term, digital memory is a problematic way for individuals to keep things from their past. Individuals are usually aware that they should make backups (and most have lost things they regret losing), but do not in fact understand how to do this, put little energy into regular backups or archive management, do not organize their old materials in a way that will make sense in the future, store media in less than secure sites (cardboard boxes in the garage), do not convert from more specific to more general formats, are less likely to have reliable storage media, and often put their trust in websites that do not value consumer data highly (Marshall, Bly, and Brun-Cottan 2006; Marshall, McCown, and Nelson 2007; Marshall 2008). Though digital

memory can potentially keep photos, diaries, art, and records of all kinds intact for years, for most people, much is lost. Memory is fragile, at least for individuals.

At the same time, however, computer memory represents one of the highest forms of precise replication ever known. Three reasons for this are that digital memories are clearly defined, that digital media are relatively stable (at least for a short while), and that the procedures for reading and writing data are executed with great care.

First, digital copies are generally higher fidelity than analog copies because the content being copied can be defined more precisely. In analog media, the thing to be copied is hard to distinguish from its containing medium. A painting is paint and canvas; a film is patterns of chemicals in celluloid. What exactly is the thing itself? How can it be preserved in its infinitely particular materiality? If you tried to copy it, what exactly is it that you would want to copy? But with digital media, the physical medium is clearly distinct from its content. Matthew Kirschenbaum refers to this double-life of objects stored in digital memory in terms of forensic materiality and formal materiality. The *forensic materiality* of an object is the electromagnetic charge on pieces of metal, the traces and marks made on surfaces, and the tiny bits of evidence that can be made to speak of the object. A piece of memory on the computer is unique and irreplaceable at the level of forensic materiality because every hard disk is different. An object's *formal materiality* depends on symbolic representations made by the computer, ultimately grounded in nothing deeper than zeros and ones. At this level, the data read as an image from one's last vacation is fundamentally a set of

symbols. Formal materiality makes digital memories discrete and well-specified (Kirschenbaum 2008, 13–15).

Second, although the long term prospects of storage media are not impressive, their ability to hold large amounts of information in a compact format that can be read and written quickly is nearly unmatched. At this moment, magnetic storage is difficult to beat for capacity, reliability, performance, and price. And, though solid state drives have become popular because they are less fragile, weigh less, use less power, and are more compact, the long term future of storage media may eventually require other materials. Research has already demonstrated the potential of DNA, quantum holography, and liquid crystals for storage (Church, Gao, and Kosuri 2012; Moon et al. 2009; Kim et al. 2009).

Third, digital memory works because of the care with which it is read and written. Machines make very good copies of well specified objects and check their work. Humans *could* do this with their own memories. One could write out a memory (ground it in a kind of formal materiality), memorize it, devise methods to check that the memory had not changed, review these often, and pass the memory on to others who would be equally careful with it. This could work to remember a phone number, keep a calendar, or record a story. In practice, humans show this care rarely and computers do it all the time. Because digital memory relies on accuracy to the bit level and tiny errors happen for a variety of reasons, devices routinely use a variety of error correction procedures to detect and fix small errors. These error correction codes confirm the content of a stretch of data, and allow electronics on a storage device to

check the accuracy of what the device reads or writes; the central processing unit of the computer repeats this check. The error correction codes used for storage (such as the Reed-Solomon algorithm) are common to many information technologies, such as cellular phone transmission.<sup>5</sup> In the case of hard disk drives, the main risks are single bit errors stored on the disk, weakly charged bits, or mistakes made in reading and transmission, therefore the procedures used for correcting errors are less intensive than in other applications and occupy a tiny percent of drive space. CD-ROMs, being more fragile, devote 10% of their capacity to error correction (Kozierok 2001). In the case of multi-level cell flash memory, a greater risk for error plus more careful error correction actually produces a net-gain in capacity and affordability (Deal 2009). In addition to error correction codes, hard disk drives reallocate data from damaged sectors and report on their performance. This attention to detail is rare in humans.

Because computer memory is actually fragile but potentially invulnerable, it favors certain practices of copying. Those who implement these practices will be able to maintain their (usually valuable) collection of virtual artifacts quite well. This is the vision of cloud computing: a company whose livelihood depends on its ability to keep data secure will do all it can. It will store information redundantly on several devices in separate physical locations, replacing and maintaining the physical media and

---

<sup>5</sup> Error correction codes interpret data as a series of symbols that can be fed into a mathematical function to produce an abstraction of minimal complexity that is consistent with, but not uniquely associated with, an original message. A very simple technique is to include a bit indicating whether a string of bits is even or odd. This would only catch a single error, and would not show where the error was. Better techniques catch more errors, can also recognize erasures, and allow some errors to be corrected. As with all engineering, the basic idea is simple, but techniques have gone through so many generations of refinement that they are now very hard to learn.

scrubbing the data for maximum preservation. Google Docs, Hotmail, Flickr, Youtube, DropBox, Amazon Cloud Drive, and iCloud offer a secure and permanent home for consumer data.

DSpace at MIT Libraries and LOCKSS (Lots of Copies Keep Stuff Safe) at Stanford use the cloud concept of virtualization by redundancy to circumvent the risk of dying media formats, encouraging the conversion of archives to digital format and the preservation of materials by making huge numbers of copies of everything. For large institutions who can afford it, best practices in data preservation promise to make digital memory a substantial advance in what can be recorded and how long it will last. This is very good news in a world where the number of books published has more than doubled every year for a decade.

For individuals, commercial services may store one's records for a while, but with some level of risk that the service will change its policies, fail or exploit private data. Social networking sites such as Facebook, people search sites such as Spokeo, and video sites such as Youtube use the same techniques as LOCKSS and DSpace, but to permanently preserve personal information that might otherwise be private. The cloud protects memory from the usual threats of error, environment, and death. But this is an interactive time whose user is a corporation, not you. Our traditional notions of reputation and identity depend on the disappearance of certain kinds of memories, but commercial websites have challenged these concepts of personhood (Greenfield 2006, 128). Viktor Mayer-Schonberger proposes an optimistic solution by changing digital memory itself: information should come with an expiration date (Mayer-

Schonberger 2009). But, as with piracy, if someone gets a copy of data just once and wants to keep it, the low-level conventions of digital memory make this possible. The familiar, soft, and placid connotations of clouds obscure both the irregularities of service and the often unnerving business practices driving companies to provide cloud storage in the first place (Coley and Lockwood 2012). There are great rewards to preserving certain data properly.

Memory provides connected temporal functions. It allows easy conversion between the frozen, still, and fixed world of files and the living, hyper-active domain of information being accessed. It can enhance performance speed by decreasing interruptions to processor cycles. Though it stores perfect copies, its physical basis makes data fragile and ephemeral. On the other hand, digital memory can also make something that might have been temporary become quite permanent. This is creating dramatic changes in the sense of unity that a self must have between past, present and future. But memory can do more.

Drawing on memory, software can create interactive times. These times are interactive in that they can be modulated by local controllers and are not dependent on central authorities, such as those keeping the most accurate time of the clock. The lack of central authority implies a decentralization of power, but not an end to power. Each time bends to the will of its individual master, and the illusion that such interactivity is to the benefit of the individual flatters the egocentric consumer. If everything is interactive, the chirpy individualist imagines, then it will all be subject to my personal control! That everyone can scan their photos and keep them digitally in a data center

where they will never get wet or thrown away during spring cleaning does not mean they are under one's personal control; interactivity empowers any controller, whether human, corporate, or computer. Automated labor works just as well for any master.

### **Network Connection**

No laptop could index the entire web and search it. When a laptop (a client) connects to (a server at) Google and searches for a query, Google's computers retrieve a few lists of top matches for each search term and combine them. These lists organize and cull the findings of crawlers that explore the web constantly, checking for changes, copying content, and studying links. The laptop itself does not search the web; the client prompts a server to search its database for the results of years of actual exploration. The total amount of information exchanged in this operation is very small, and the computation involved in a single search request is also. But the search taps into much larger operations, from which it gleans only the most summary information. This is the basic reality of network connections.

A universal Turing machine *can* operate as if it were any particular Turing machine. The computer in an mp3 player could, in theory, simulate supercomputers that play chess, forecast the weather, or map the web. However, the small computer is not going to do a good job of it. You would get tomorrow's weather forecast in a few years, or use a simplified model and get very inaccurate results today. Simpler than having one machine simulate another machine is to have the two communicate. For two Turing machines, conversation consists of executing instructions on symbols to produce symbols in an outgoing channel, and then reading symbols from an incoming



channel. It is as if each Turing machine had several strips of symbols, and some of these strips stretched from one to another. Network connection, data exchange between two or more computers, has become a huge phenomenon because it allows one computer to express the symbolic operations of another. The explosive growth of the web represents the empowerment consumers experience from having their one (usually small) computer do a huge range of things it could not, practically speaking, do on its own.

Several unique conventions of time arise from the basic facts of network connection and have important material consequences. Packet switching—the way communication passes through the network—establishes a fragile sequence of operations that paces Internet usage and encourages 24/7 access to services of many kinds. The line through which this information passes creates experiences of fast and slow connections and of accelerated communication. Through network effect, the Internet has grown in importance at an astounding rate, and produced a new cutting edge of the present moment that has divided our attention between the many things we can find online. Those in the loop make the present; everyone else is living in the past.

The basis of efficient, multi-party network connectivity today is packet switching. This mode of communication is full of sequences. Before transmitting a message, each computer must fragment it into small units that each have a short header, including information such as where they come from and where they are going to. This turns the wildly varied forms of data that are being sent over the network into a single type of packet, easily readable by any computer. When a packet, marked with

a destination outside the network, arrives at a router (a specialized computer) it sends the packet along to an ISP (Internet Service Provider), which forwards the packet toward its destination. This forwarding process is complicated technically and economically, since there are many algorithms to find paths, many kinds of hardware filtering and redirecting messages, and, on top of this, those who control routing hardware may prefer that packets take one route to another. Once a packet arrives, the destination sends an acknowledgment and, usually, a response. Some packets don't make it, because their data becomes corrupted, they become mixed in with another packet (collision), they take a path that is too long and reach their hop limit,<sup>6</sup> they are rejected by a proxy or firewall, or something else happens to them along the way. The reading, organizing, and directing of packets all work by protocol formalized by standards organizations (such as the IETF, ICANN, IANA), and then adopted voluntarily—or partially, or not at all—by users of all kinds (A. Galloway 2004). These protocols specify much more than the basic mechanism of packet switching. For example, before a computer can navigate to a url (such as Wikipedia.org), it must find a server that knows the IP address of the url (such as 204.74.112.1), then it must make a TCP handshake (a series of three messages confirming an intention to communicate) with the server at that IP address. All activity on the Internet depends on these conventional sequences and some bear the imprint.

---

<sup>6</sup> A hop limit, previously called time to live, is a number in a packet's header that is reduced by one at every router that processes it. The number starts somewhere between one and 255; when it reaches zero, the packet is destroyed and a report sent to its source. A traceroute derives from this convention, sending a series of packets with a hop limit of zero, then one, then two and so on.

The various protocological sequences involved in packet switching all take a measurable number of milliseconds. These tiny delays add up to the familiar effect of latency. The latency on the Internet itself is rather small, near to the theoretical minimum. The machines processing packets tend to be very specialized and powerful. However, the latency introduced by one's own computer, or by the modem connecting a local network to the Internet service provider, may be substantial. The minimum time to send a message may be a tenth of a second, which sets an absolute upper limit to the response time of any activity done online. The latency introduced by these weak links cannot easily be overcome without getting a new computer or modem.

Chrome, the most popular browser as of this writing, attempts to circumvent it. By anticipating what sites a user might need next, the browser can look up the future site's IP address and make a TCP handshake (Grigorik 2012). (Other designs go a step further and begin loading pages that the user may soon visit, a practice called prefetching links.) In these schemes, the fractions of a second required to connect to a server and load a website can happen, or at least begin, before the user even starts waiting. This technique does not *reduce* latency, but does reduce the *experience* of latency.

The time of latency inherent in Internet communication is quite important in online gaming. Though a slowly loading page tests our patience, or makes a website less desirable to visit, a delay in gaming can make a decisive difference within the game. If, gaming, I enter a room and find myself in a showdown, a tenth of a second can very easily be the difference between being the first to draw and first to die.

Network latency can create a delay between user input and avatar actions or between the actions of others and their display on the user's machine. Though gamers have some understanding of lag, and often see it as an important part of online gaming, they usually are unable to do much about it (Tseng et al. 2011). Latency supplements the sporadic flow of new information (whether website or game) with a small, ineliminable delay.

One of the most frequently remarked upon transformations of contemporary practices of time is that more things are available 24/7. Not so long ago, shopping, banking, and buying tickets had to be done at specific hours of the day. Waiting until after dinner would simply not do. Now these services are available on the web and can be done at any time. This change in time results quite directly from the nature of Internet connections and from the economics of running a server. One website, in one place, serves users everywhere.<sup>7</sup> The many people who might want to use a web service may have very different schedules from each other and probably live in different time zones. Requests to a server can therefore come any time during the day. Usual business hours represent only one third of the total number of hours in the day, and their use has always depended on the synchronization of human populations in their patterns of work, sleep, and childcare—patterns more or less aligned with daylight. Although a website could decide to turn off its servers in the evening,

---

<sup>7</sup> Though not all points on the planet are connected to each other equally, there are strong economic motivations to build infrastructure responding to existing demand for connection. This usually produces a link between two points on the Internet. On the other hand, when it fails, the disconnected machine is simply not counted as on the Internet at all.

according to local time, there are economic and cultural reasons not to do this. Given the very large costs associated with creating a website, running the business, and drawing in customers, the costs of keeping a server powered on at night is usually worth the marginal cost of power consumption. Though there may be very few visitors, the machine can perform automated maintenance, will be available to crawlers, and will suffer less thermal stress if its components remain at a constant temperature (Hamilton 2008). Culturally, 24/7 accessibility has become an expectation for the web. Though some orthodox religious groups limit the functionality of their websites on holy days, and some government websites still depend on batch processing systems that restrict functionality to business hours, these are exceptions that prove the rule: the web must be accessible 24/7 (Dubner 2012).

The communication line by which packets move is itself a significant condition on time in computers. The line establishes a maximum rate of communication, which is often faster in one direction than another, and is being slowly improved by the use of fiber optic cables. What is speed on the network? The speed at which a signal moves across a copper wire is at least two thirds the speed of light (light moves at 300 million kilometers per second in a vacuum). A single impulse, then, is very fast. Why, then, do Youtube videos often load so slowly? The question of speed in an electronic connection is actually a question of how many impulses can be sent per second—frequency. The trouble with copper wire is that a higher frequency signal creates inductive reactance, which causes the signal to weaken over distance. Repeaters can receive a fading signal and repeat it with greater strength and clarity, but these cost

money. This is the basic reason for the widespread shift to fiber optic cables; fiber optics transmit higher frequency signals further. Because fiber optics don't carry current, they are also better protected from interference caused by electrical fields, such as from the electrical lines they inevitably parallel or cross. Different lines run at different speeds. In the U.S., telecommunication companies have improved the trunk lines for Internet traffic by building out from telephone networks or networks established by the federal government, thereby privatizing the Internet backbone quickly and with very little public visibility (Shah and Kesan 2007). These companies collect rent on traffic moving through their lines, and increased Internet traffic has encouraged them to lay more fiber optic cables. As with latency, the major bottleneck for connection speed is usually the infrastructure between the user's fingertips and the Internet service provider's nearest hub.

Despite latency, despite limited bandwidth, and despite bottlenecks, the Internet is faster than what it replaces. This simple concrescence of shortened response time, increased throughput of data, and quicker distribution of information has changed many previously existing social practices of time. The news cycle of nightly news on TV and morning headlines of paper has been recontextualized by Internet news, which is constantly available and constantly updating.

The pace of doing business was, like the pace of fashion or personal communication, entrained to the rhythms of older technologies. With the increasing centrality of computers linked by a network, the delays of postal communication, information processing within an office, and transfer of records between different

departments (from a warehouse to marketing, for example), has become much shorter, though it is still never quite instant. Stock trading, email, and telecommuting are well known examples of this transformation. In each case, the pace of information passed through a network connection is sufficiently faster than previous methods to change how people do things (Hassan 2007). However, most of the force behind these transformations comes from the saturation of social space with computers. Email without regular computer access would probably not replace paper mail so completely; telecommuting would be impossible without a powerful home computer.

Computers combine together many kinds of activities, allow new kinds of services, and connect with each other, thereby increasing the value of each individual computer. This is called network effect. The more things that are attached, the more powerful the network becomes; the more powerful the network, the more things will attach to it. As the early web became populated with interesting and useful sites, it became more worthwhile for others to connect. More recently, Facebook and other sites hosting user-generated content have made use of the same principle. As more people use the site, it becomes more attractive for others to join. In fifteen years, global Internet traffic grew from 1.9 petabytes per month in 1996 to 27,483 petabytes per month in 2011 (“Internet Traffic” 2012). As of 2012, there were 1.3 billion used IP addresses (Carna Botnet 2013) and more than two billion Internet users, with half of that number on Facebook (“World Internet Users Statistics Usage and World Population Stats” 2012). Billions of tangled Turing machines swap symbols, process them, and continue.

The huge number of computational services available through the Internet have made it a font of newness and established it as the cutting edge of the present. The Internet allows consumers to choose from a huge range of options 24/7, and access these options with lower overhead than the usual channels. Entertainment industries such as movies, pornography, TV, video games, and music have re-oriented their business models around the Internet. The Internet has always meant connecting one computer to another, but things have changed from the original model of occasionally updated documents accessed by curious strangers (Berners-Lee 1999). In web 2.0, a revitalization of this earlier model, special efforts were made to create user interaction with servers; the page served to a user would be personalized and invite new contributions, that would be shared with others. This has drastically increased the number of authors updating the web and this is essential to getting information relevant to the current moment. Aggregating sites prioritize newer material (presumably to satisfy users), thereby creating a browsing experience obsessed with the present moment. The Network Time Protocol that keeps computers' clocks synchronized is an afterthought to these main dynamics; the protocol estimates latency, requests several signals from a time server, and calculates an average to which it sets the clock. In comparison to the Internet, everything else is always a bit out of date. The newspapers actually at newsstands, the political opinions of someone who has not read the latest news, and the taste of those who have not had access to the web, those without smartphones, those who spend less time online, and those with no connection at all are practically members of a pre-dead past.



The bonanza of stuff to look at, accessible thanks to network effect, has taught new patterns of reading and attention. We have become fickle and, in some regards, lazy. If a story is not interesting, we move on. If a website is not immediately useful, we try hitting a link, hoping to get something better. Jakob Nielsen's articles on user behavior explain that the average visitor to a website spends less than ten seconds on the page and reads less than twenty words. Readers scan for text and disregard images unless they are big and easy to understand. Generally, the visitor is looking for something in particular, and ignores everything else. The eyes move in an F shape pattern, concentrating on the top and left side of the page, checking only the first few lines of the center. Often, users don't scroll down the page at all.<sup>8</sup> These patterns of attention set the rhythm for the time of browsing. Those making content reinforce these patterns by designing with this reading practice in mind.

Network connectivity establishes several conventions of time that have direct and indirect expressions, depending on how they are appropriated. Discussion of the Internet's delicate protocological sequences, 24/7 nature, lag or high speed, acceleration of social processes, power to generate novelty, and transformation of our patterns of attention each describe different applications of the same technologies. The Internet can create a time of terrible slowness, when your latency is high or bandwidth is insufficient, or a time characterized by intense speed, when it replaces something slower. It can be a rich library calmly pored over for hours, or a blur of superficial non sequiturs. It can be sequenced, with hundreds of tiny operations required for every

---

<sup>8</sup> Articles on these topics and more are available on Jakob Nielsen's Alterbox at <http://www.nngroup.com/articles/>.

second of operation, and it yet it can be practically instantaneous. As with processing and memory, the diversity of times that use the technology arises from the variety of ways that conventions of time are actually taken up by material practices. There are patterns to this appropriation.

### **Conclusion: How Conventions Have Joined**

Computers have enabled many new kinds of times. Some speed us up, others slow us down; some require caution and care, others demand that we step back and wait. Each project making a time forms out of a heap of conventions, ignoring some, mimicking others, and depending on a great many. Each time, then, is the product of work building on an unsteady pile of dependencies. In some cases, this feat will be remembered, repeated, and eventually reinterpreted as itself another convention. The times produced by conventions have been the topic of this chapter: the time of undo, the ether of moments defined by the scheduler, the musical loops facilitated by software, the uncertain future of our personal digital archives, the 24/7 web, and the fragmented attention of web browsers.

The conventions covered in this chapter will give a partial explanation of most times that computers can produce, but this is hardly the last word on the topic. Many other conventions of time exist that are not closely connected to the times of processing, memory, or networks. There are significant conventions of time in file formats, in graphical user interface, in email, in hardware, firmware, design, software, and user behavior. Digital video, for example, depends on hundreds of methods for compressing and playing back video, at different framerates and bitrates, with

different requirements for transmission, storage, and processor power at playback. Conventions for online calendars, built up by decades of digital schedulers systems, include automatic scheduling, public and private events, available time, recurring events and alarm systems.

I call these conventions of time because they are known formulas that put temporal influence to work. They are patterns ready to be put into practice. Some conventions set a single parameter of time's function, such as rate or schedule. Others refashion functions of temporality into new forms, creating new kinds of immediacy, memory, or delay. Most put several aspects of time into a practice that produces an abstract type of time, actualized by many different implementations. Latency, for example, will matter in very different ways for a file repository than for someone playing an online video game, though it is one convention of time arising, in both cases, from the same technical basis. Conventions are not themselves times, which are more material, or temporal functions, which are less. Cycles and loops, prominent in computing, are specific versions of repetition, with certain conditions resulting directly from the machine. Conventions are repetitions recognized; these patterns make names for themselves and find new uses.

Conventions combine to make various kinds of times. Together, they can provide an interactive time, where the user decides when things start, pause, stop, or save. To do this, the processor provides a responsive environment where almost any object can be represented as data and manipulated; memory converts freely between the living present and stored images of it, preserving the past for the lifetime of the

machine's storage medium; the network lets each client computer do things it could not do alone, operates with minimum latency to give the user a sense of instant connection, and can share or unshare the work at any time. Alternately, conventions can produce a time that excludes one or many people, leaving them not only with no ability to control the time, but even trapped forever in its past. Network connections can be closed, the position of the present moment can be contested too hotly, and users may have read-only access to the 24/7 events announced by powerful processors running software loops on some other server. Many other times are conceivable. A website could offer a moment always a step in the future, where everyone would add their immediate plans to a collaborative image of what is expected to soon be the present. A museum could open an old-fashioned website, re-enacting the web as it was in 1993 even down to the speed of the user's connection. A game developer could make a world that runs backwards as easily as forwards, with this flow applying unevenly to some objects but not others.

Time can be designed, and today's architects of time do this by appropriating conventions. Software engineers, game developers, computer scientists, electrical engineers, user interface designers, and those making and contributing to computers infrastructures may, at some point in their work, inadvertently or quite intentionally, design new times that become conditions for the lives of others. They take the fungibility of memory and create from it a series of static images of a streaming file or a temporary state. They design programs around the habits of the operating system's scheduler to produce a smooth and responsive experience for their users. They study

hard disk failure, monitor their drives, and follow best practices to keep an eternity of protected data.

In some of these cases, the user gains some mastery of time. In others, they do not. An Internet service provider that limits download speeds establishes a rate of flow for all effected users, but the people using the throttled connection are *subject* to this designed time, not in control of it. An automated stock trading system uses its heightened reactions and low-latency network connection to privilege trades made on behalf of its owner over those made by anyone else. Times designed for one class of user affect others, but, once designed, these others are not in a position to manipulate the time as it acts upon them. In most cases the key criterion for success in the design of time is user satisfaction, not the consequences for other people. If the user to be satisfied is a rich, centralized political authority, designed times can be quite unpleasant things to be around.

We are, in a way, fortunate that designed times, which could have done many other things, are often interactive. Though things could be different, in many cases the user has, within certain bounds, been given a degree of control over the functioning of time. When we make changes to a file, we benefit not only from an ability to undo but also from autosaves and an archive of previous versions. When we listen to music, we expect to be able to stop, pause, skip or skip around within that piece of music. For TV, digital video became a normal part of home life for many because it allows the person with the remote control to control time. In social networking, almost any action that can be taken can be taken back. Sound and video editing go further, turning the

occurrence of events into blocks of movement that can be shifted around and manipulated in hundreds of ways, as the editor sees fit, to boldly sculpt time in new ways. Backup systems, content versioning systems, games that allow the user to control the pace of gameplay, and any system that puts the influence of temporality in the hands of its user are interactive times produced by the artful use of the time conventions of computing. Interactive times were no mistake. Their original designers created reversible times intentionally; other adopted the convention and, in many cases, the industry created a standard out of it.

Interactive times are one case of designed times which are themselves only the intentional arrangement of underlying conventions of computing. These times are not alone in the world. They do not extinguish the beauty of a slow moment of fulfillment and hope, the slow passing of a season of fleeting sunsets, or the progress by invariant intervals of the clock.

Times exist as contested territory. Machine driven times are not the only, or even the most important times in the social world. We live in practices of time and we live through them. Whether alone on the computer or resting intimately in the branches of a tree, there is no exit from times and the social situation that they express. While the conventions of time present in computing can innovate and imagine lots of strange little times, and have established some very common ones, they are still subject to the translation and influence of other times. They are still invested in the same fields of force connecting material objects. This makes them vulnerable, threatening, or attractive to other times.

Interactive times have become widespread because they have been useful. Built of conventions that are usually less forgiving, and responding to them, interactive times, enacted by computers and their users, allow local modulation of time. The knowledge worker, a category into which a great number of people now fall, if only for parts of their day, has gained a limited power over time. This empowerment is psychological insofar as interactive time is emboldening and encourages computer use. But it is also a very real form of power with consequences visible all around us. To recognize this power clearly is a good step towards coming to live with it.

### References

- Aaronson, Lauren. 2008. "How It Works: The Sturdiest Solid-State Storage." *Popular Science*, March 13. <http://www.popsoci.com/gear-gadgets/article/2008-03/how-it-works-sturdiest-solid-state-storage>.
- Astor, Mary. 1971. *A Life on Film*. New York: Delacorte Press.
- Beidelman, T. O. 1963. "Kaguru Time Reckoning: An Aspect of the Cosmology of an East African People." *Southwestern Journal of Anthropology* 19 (1) (April 1): 9–20. doi:10.2307/3628919.
- Berners-Lee, Tim. 1999. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web By Its Inventor*. 1st ed. San Francisco: HarperSanFrancisco.
- Bogost, Ian. 2007. *Persuasive Games: The Expressive Power of Videogames*. Cambridge: MIT Press.
- Borisov, N., S. Babu, N. Mandagere, and S. Uttamchandani. 2011. "Dealing Proactively with Data Corruption: Challenges and Opportunities." In *2011 IEEE 27th International Conference on Data Engineering Workshops (ICDEW)*, 34–39. doi:10.1109/ICDEW.2011.5767656.
- Borkar, Shekhar, and Andrew A. Chien. 2011. "The Future of Microprocessors." *Communications of the ACM* 54 (5) (May 1): 67. doi:10.1145/1941487.1941507.

- Carna Botnet. 2013. "Internet Census 2012: Port Scanning /0 Using Insecure Embedded Devices." <http://internetcensus2012.bitbucket.org/paper.html>.
- Carter, David. 2008. "Clever Children: The Songs and Daughters of Experimental Music?" Dissertation, Queensland Conservatorium Griffith University. [http://www.griffith.edu.au/\\_\\_data/assets/pdf\\_file/0007/177856/Carter\\_PhD\\_eThesis.pdf](http://www.griffith.edu.au/__data/assets/pdf_file/0007/177856/Carter_PhD_eThesis.pdf).
- Chun, Wendy. 2006. *Control and Freedom: Power and Paranoia in the Age of Fiber Optics*. Cambridge Mass.: MIT Press.
- Church, George M., Yuan Gao, and Sriram Kosuri. 2012. "Next-Generation Digital Information Storage in DNA." *Science* 337 (6102) (September 28): 1628–1628. doi:10.1126/science.1226355.
- Coleman, E. Gabriella. 2013. *Coding Freedom: The Ethics and Aesthetics of Hacking*. Princeton: Princeton University Press.
- Coley, Rob, and Dean Lockwood. 2012. *Cloud Time*. Winchester: Zero Books.
- Collins, Karen. 2007. "In the Loop: Creativity and Constraint in 8-bit Video Game Audio." *Twentieth-century Music* 4 (02): 209–227. doi:10.1017/S1478572208000510.
- Deal, Eric. 2009. "Trends in NAND Flash Memory Error Correction". Cyclic Design.
- Dijkstra, Edsger W. 1968. "Go To Statement Considered Harmful." *Commun. ACM* 11 (3) (March): 147–148. doi:10.1145/362929.362947.
- Dubner, Stephen J. 2012. "This Website Only Open During Business Hours." *Freakonomics*. <http://www.freakonomics.com/2012/08/20/this-website-only-open-during-business-hours/>.
- Etsion, Yoav, Dan Tsafir, and Dror G. Feitelson. 2003. "Effects of Clock Resolution on the Scheduling of Interactive and Soft Real-time Processes." *SIGMETRICS Perform. Eval. Rev.* 31 (1) (June): 172–183. doi:10.1145/885651.781049.
- Evans-Pritchard, E. E. 1939. "Nuer Time-Reckoning." *Africa: Journal of the International African Institute* 12 (2) (April 1): 189–216. doi:10.2307/1155085.
- Galloway, Alexander. 2004. *Protocol: How Control Exists After Decentralization*. Cambridge Mass.: MIT Press.
- Galloway, Alexander R. 2006. *Gaming: Essays On Algorithmic Culture*. 1st ed. Minneapolis: Minnesota.



- Columbia, David. 2009. *The Cultural Logic of Computation*. Cambridge Mass.: Harvard University Press.
- Greenfield, Adam. 2006. *Everyware: The Dawning Age of Ubiquitous Computing*. Berkeley, CA: New Riders.
- Grigorik, Ilya. 2012. "Chrome Networking: DNS Prefetch & TCP Preconnect." *Igvita.com*. <http://www.igvita.com/2012/06/04/chrome-networking-dns-prefetch-and-tcp-preconnect/>.
- Hamilton, James. 2008. "Should We Shut Off Servers?" *Perspectives*. <http://perspectives.mvdirona.com/2008/12/01/ShouldWeShutOffServers.aspx>.
- Hassan, Robert. 2007. *24/7: Time and Temporality in the Network Society*. Stanford, CA: Stanford Business Books.
- Hennessy, John L, and David A Patterson. 1998. *Computer Organization and Design : the Hardware/software Interface*. San Francisco, Calif.: Morgan Kaufmann Publishers.
- "Internet Traffic." 2012. *Wikipedia, the Free Encyclopedia*. [http://en.wikipedia.org/w/index.php?title=Internet\\_traffic&oldid=516859227](http://en.wikipedia.org/w/index.php?title=Internet_traffic&oldid=516859227).
- Kim, Jin Ki, Fumito Araoka, Soon Moon Jeong, Surajit Dhara, Ken Ishikawa, and Hideo Takezoe. 2009. "Bistable Device Using Anchoring Transition of Nematic Liquid Crystals." *Applied Physics Letters* 95 (6) (August): 063505 – 063505–3. doi:10.1063/1.3202781.
- Kirschenbaum, Matthew. 2008. *Mechanisms: New Media and the Forensic Imagination*. Cambridge Mass.: MIT Press.
- Knuth, Donald E. 1974. "Structured Programming with Go to Statements." *ACM Comput. Surv.* 6 (4) (December): 261–301. doi:10.1145/356635.356640.
- Kozierok, Charles M. 2001. "Error Correcting Code (ECC) and Data Recovery." *The PC Guide*. April 17. <http://www.pcguides.com/ref/cd/mediaECC-c.html>.
- Lanier, Jaron. 2010. *You Are Not a Gadget: A Manifesto*. 1st ed. New York: Alfred A. Knopf.
- Latartara, John. 2010. "Laptop Composition at the Turn of the Millennium: Repetition and Noise in the Music of Oval, Merzbow, and Kid606." *Twentieth-century Music* 7 (01): 91–115. doi:10.1017/S1478572211000065.

- Leddy, Christopher. 2003. "Avoid Corruption in Nonvolatile Memory." *Embedded*. August 20. <http://www.embedded.com/design/prototyping-and-development/4006422/Avoid-corruption-in-nonvolatile-memory>.
- Malinowski, Bronislaw. 1927. "Lunar and Seasonal Calendar in the Trobriands." *The Journal of the Royal Anthropological Institute of Great Britain and Ireland* 57 (January 1): 203–215. doi:10.2307/2843682.
- Manovich, Lev. 2001. *The Language of New Media*. 1st MIT Press Pbk. Ed. The MIT Press.
- Marken, Andy. 2004. "CD and DVD Longevity: How Long Will They Last?" *Audioholics Online A/V Magazine*, August 25. <http://www.audioholics.com/education/audio-formats-technology/cd-and-dvd-longevity-how-long-will-they-last>.
- Marshall, Catherine C. 2008. "Rethinking Personal Digital Archiving, Part 1." *D-Lib Magazine*, April. <http://www.dlib.org/dlib/march08/marshall/03marshall-pt1.html>.
- Marshall, Catherine C., Sarah Bly, and Francoise Brun-Cottan. 2006. "The Long Term Fate of Our Personal Digital Belongings: Toward a Service Model for Personal Archives." In *Archiving 2006*, 25–30. Ottawa, Canada. <http://www.csdl.tamu.edu/~marshall/marshall-personal-archiving.pdf>.
- Marshall, Catherine C., Frank McCown, and Michael L. Nelson. 2007. "Evaluating Personal Archiving Strategies for Internet-based Information." In *Proceedings of Archiving 2007*, 151–156.
- Mayer-Schönberger, Viktor. 2009. *Delete: The Virtue of Forgetting in the Digital Age*. Princeton: Princeton University Press.
- Mellinger, D, Q Lindsey, M Shomin, and V Kumar. 2011. "Design, Modeling, Estimation and Control for Aerial Grasping and Manipulation." In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*.
- Mellinger, D, N Michael, and V Kumar. 2010. "Trajectory Generation and Control for Precise Aggressive Maneuvers with Quadrotors." In *Proceedings of the International Symposium on Experimental Robotics*.
- Montfort, Nick, and Ian Bogost. 2009. *Racing the Beam: The Atari Video Computer System*. Cambridge, Mass.: MIT Press.
- Moon, Christopher R., Laila S. Mattos, Brian K. Foster, Gabriel Zeltzer, and Hari C. Manoharan. 2009. "Quantum Holographic Encoding in a Two-dimensional

- Electron Gas.” *Nature Nanotechnology* 4 (3): 167–172.  
doi:10.1038/nnano.2008.415.
- Moore, GE. 1965. “Cramming More Components onto Integrated Circuits.”  
*Electronics* 38 (8) (April 19): 114–117. doi:10.1109/JPROC.1998.658762.
- Moran, Chuk. 2011. “Interactive Time and ‘Real Time’ in Software and Society.”  
*Spectator (USC)* 30 (1): 23–28.
- Pinheiro, Eduardo, Wolf-Dietrich Weber, and Luiz André Barroso. 2007. “Failure Trends in a Large Disk Drive Population.” In *5th USENIX Conference on File and Storage Technologies (FAST 2007)*, 17–29.  
[http://research.google.com/archive/disk\\_failures.pdf](http://research.google.com/archive/disk_failures.pdf).
- Roberts, James F., Timothy S. Stirling, Jean-christophe Zufferey, and Dario Floreano. 2007. “Quadrotor Using Minimal Sensing For Autonomous Indoor Flight.” In Toulouse, France.  
<http://130.203.133.150/viewdoc/summary?doi=10.1.1.170.8644>.
- Robinson, G.S., and C. Cargill. 1996. “History and Impact of Computer Standards.”  
*Computer* 29 (10) (October): 79 –85. doi:10.1109/2.539725.
- Rothenberg, Jeff. 1995. “Ensuring the Longevity of Digital Documents.” *Scientific American*, January.
- Schroeder, Bianca, and Garth A. Gibson. 2007. “Disk Failures in the Real World: What Does an MTTF of 1,000,000 Hours Mean to You?” In , 1–16.  
[http://static.usenix.org/event/fast07/tech/schroeder/schroeder\\_html/](http://static.usenix.org/event/fast07/tech/schroeder/schroeder_html/).
- Shah, Rajiv C., and Jay P. Kesan. 2007. “The Privatization of the Internet’s Backbone Network.” *Journal of Broadcasting & Electronic Media* 51 (1): 93–109.  
doi:10.1080/08838150701308077.
- Shaviro, Steven. 2003. *Connected, or What It Means to Live in the Network Society*. University of Minnesota Press.
- Smith, Mackenzie. 2005. “External Bits.” *IEEE Spectrum*, July.  
<http://spectrum.ieee.org/computing/hardware/external-bits>.
- Stallings. 2000. *Operating Systems: Internals and Design Principles*. Upper Saddle River, NJ: Prentice-Hall.
- Star, Susan Leigh. 1999. “The Ethnography of Infrastructure.” *American Behavioral Scientist* 43 (3) (November 1): 377–391. doi:10.1177/00027649921955326.

Stillar, Glenn. 2005. "Loops as Genre Resources." *Folia Linguistica* 39 (1-2) (June 23): 197–212.

Tseng, Po-Han, Nai-Ching Wang, Ruei-Min Lin, and Kuan-Ta Chen. 2011. "On the Battle Between Lag and Online Gamers." In *2011 IEEE International Workshop Technical Committee on Communications Quality and Reliability (CQR)*, 1 –6. doi:10.1109/CQR.2011.5996093.

Walter, Chris. 2005. "Kryder's Law." *Scientific American*, July 25.  
<http://www.scientificamerican.com/article.cfm?id=kryders-law>.

Woolf, Gordon. 2001. "How Long Will a CD-R Last?" *PC Update*, June.  
<http://www.melbpc.org.au/pcupdate/2106/2106article14.htm>.

"World Internet Users Statistics Usage and World Population Stats." 2012. Internet World Stats. <http://www.internetworldstats.com/stats.htm>.

Ziegler, J. F., H. W. Curtis, H. P. Muhlfeld, C. J. Montrose, B. Chin, M. Nicewicz, C. A. Russell, et al. 1996. "IBM Experiments in Soft Fails in Computer Electronics." *IBM Journal of Research and Development* 40 (1) (January): 3 – 18. doi:10.1147/rd.401.0003.

## **VI. Conclusion: Interactive Time**

This project took several years. At many points along the way, I doubted my own understanding of time and of the clock in particular. So I decided to build one. I ordered a kit for a wooden gear clock and assembled it over the course of a year. I hoped to better understand how the escapement translates gravity into invariant intervals that gears organize into seconds, minutes, and hours. What I learned was a bit different: most of what makes a clock work is casing, supports, screws, mounts, and adhesives. We think of clocks as measuring the passage of time by counting off the seconds. But, for the machine to operate regularly, a much larger number of other parts have to be working properly first. Some parts must be smooth or loose. Others provide resistance, support, or serve simply to position the other bits. Decoration is intrinsic to the dial and the space that the clock will ultimately occupy is a concern from the very beginning.

Enlisting temporal function is hard, even when there are established conventions for doing it! The challenge of enacting a time lies first in the secondary roles of support, position, resistance, and looseness. What we consider the primary elements of a clock, such as the gears, dials, escapement, weight, and pendulum, will only work when all the rest is in place.

In the same way, practices of time are made up mostly of supports. The times of factory work, undo, care, or travel by airplane depend on a huge number of things happening (Thompson 1967; Luckmann 1991). When everything goes right, or basically right, congeries enact times (Durkheim 1995, 11–12). Times bundle temporal

functions together, covering temporal terms (Koller 1974). In this way, each time is quite partial, organizing only a limited range of all the heterogeneous ways temporality actually plays out. Thus, the key to understanding times is to discern the various roles of temporality. Enactment, in my account, is more important than reference.

If time is enacted, what does this mean for metaphysics? In the contemporary scene, it has become popular to define the world as a large number of objects interacting with each other to create occurrences, including other objects. By various names, this view usefully expresses the fact that there are other actors than human things, that nature is epistemological, that unexpected connections may be important, and that social construction is productive work. Seeing time as an enacted combination of temporal terms yields three contributions to these ideas. First, time is not a neutral container in which connections occur. You can undo actions in some times, but not in others. Events occur within multiple, overlapping times. They do not just happen in the essential flux of the universe; often, they are scheduled, expected, or remembered. Second, causality is not the only possible relation of influence. Anticipation can direct action as easily as can flow. Beings attract and touch each other without regard for a timeline or progressive grammatical structure of one thing leading to another. Third, objects experiencing one another are not all that sensitive. Turning on a light can be reversed, from the point of view of those who have not entered the room. If times apply temporality to particular ranges of objects, and there is no fundamental time

applying its rules to all objects, some cases of access, inevitability, history, and presence will be missed by some beings as a simple matter of ignorance and neglect.

The many projects of social theory exploring contemporary times have tended to miss each other like ships in the night. By understanding times as practices, however, it should be possible to establish a network effect for such theory. Different studies can show how various times operate and relate, in relation to dominant times or the underground of alternative times formally excluded by the dominant set but, still, in fact, in common use.

Times exist as contested territories with overlapping borders, quite like relations of power. Times are mutually implicated in very unequal ways (Adam 1995). Chapter three looked at relationships between times paradigmatically and historically. Of the many possible relationships times have taken on, the categories of translation and influence are particularly explanatory. In translation, one time captures some of the temporal contents of another time without regard for the different practice that combined these terms in the other time (Lim 2009). Influence puts translation to work and enacts one time on the supports of another, changing it.

The clock's rise to centrality in time practices reveals the supports that positioned it in its current role. Industriousness, market affairs, and public clocks in an urban environment selected a technology useful to time practices of many kinds (Dohrn-van Rossum 1996; Glennie and Thrift 2009). The technology proved particularly useful at creating authoritative measurements that directed efforts, often at increasing efficiency (O'Malley 1996; Simpson 1995).

Older kinds of time had made their own uses of many of temporality's diverse functions, such as memory, ritual, synchronization, event, inevitability, hope, eternity, and sequence. But times using the clock, reinforcing each others' accounts as fundamentally unitary, universal, and authoritative, presented a mechanically measured time as a symbol of rationality, heralding the way of the future (Tanaka 2004; Nanni 2012; McCrossen 2007; Kern 1983; Dohrn-van Rossum 1996).

The modern formulation of time as consistent across the metaphysical, everyday, scientific, and astronomical owed its power to a series of small realignments that clarified certain aspects of time that were proving problematic to certain practices. As a result, we now think that the relationship is natural between the vibrations of the Caesium atom, length of an hour, system of time zones determining what the hour is, leap year and leap second that keep astronomical time consistent with everyday representation, synchronization of our clocks with this very specific hour, and the concept of time as a dimension perpendicular to the three of space (Jones 2000; Bartky 2007; Mills 2006; Phillips n.d.).

We have become used to the idea that someone else (or, even better, something else) is in charge of time, and that human activity occurs within this already existing parameter of the universe. But it is hard to deny the concurrency of the mutually presupposing alternative: we have made time into forms which we control.

The world is not simply accelerating around us; many times coexist. Some groups are opting for speed or are not. Certainly, some drives intrinsic to capitalism have encouraged efficiency on the basis of a practice of measuring time, increasing



turnover and delivery times, and providing a cultural sense of speed and access (Harvey 1989). But many practices are not well translated by such generalization and not so obedient to this representation of what they do. Many things are slow, have become slower, remain unchanged, or have become quick without simply being accelerated. New times produced by computers and cultivated by market imperatives do not always make life faster.

Looking at specific, enacted time practices shows the impacts of computers as they have already begun to take effect. Avoiding prophecy and guesswork, chapter four describes the rise of a particularly important kind of time yielded by computers: undo. Computers switched from batch processing and a paradigm of automation to become tools assisting people in live interactions. Nelson, van Dam, and a few undergrads at Brown followed a very big idea to empower and change knowledge work and, along the way, added to interactive computing a bit of interactive time. For various reasons, these designers, along with others in other projects, created a command that turned the series of actions entered by a user into a reversible flow of events. The time of undo would not have gone far without being caught up in larger currents. At PARC, where invention did not always follow hype, the command proliferated through Teitelman's prototype. Yet it was only with the decline of PARC that the technological infrastructure for a new kind of time moved out to the software market and became a norm.

The supports of this interactive time are quite different from the little pieces of wood holding together my clock. They are rhizomatic, of various qualities, and branch

into unexpected empirical domains. Undo relies on reviews that would point out a program without the feature, interface guidelines requiring it, and office managers afraid to be seen making an irrecoverable mistake on a computer, among others. An earlier technology of time had a very different set of supports. Colonial powers, railroad companies, national interests, and revered generals threw their weight behind standardized clock times. In contrast, the primary vehicle of interactive time is the computer and its primary supports are those many, diverse interests supporting computerization in its current form.

Contemporary computing converts real phenomena into digital representations and puts knowledge workers on the job of analyzing, translating, using, and getting things done with this data. Each individual information processor need not fully understand the system. This would require far too much training and new systems are always coming along that require new skills. More people became users, using the system with little ability and less understanding. The net effect was a vast increase in functionality for computers overall.

An increased user base augmented more computer symbiotes with a sensorium constructed by software. The limits on time perception discovered by phenomenology have been circumvented by a cyborg state of perception and imagination that runs on software. Software animates fantasies, allowing new ideas of how temporality might work to become new conditions that apply to some other activity being done on a computer. With interactive time, documents remain open to future changes, even if sent off or transferred to hard copy, and a project may be borrowed from or

repurposed at a later date. Interactive time changed the products of those who used it. Any file can be expected to have been edited repeatedly. In many cases, the editor and end user can manipulate the pace of time in viewing these products, rewinding songs or playing video in slow motion. Because data remains open to corrections and alterations, it becomes tempting to put things online that might have once been printed on paper (think of the phone book) or done in person (online dating). These practices involve access, pastness, reversibility, memory, repetition, pace, and very long windows of opportunity. Each one links together these functions in its own way, producing one of the many practices of times by which many work and play today.

We do not yet know what a time is capable of. Times enacted by processor, memory, network connections, peripheral devices, and human affiliates have already produced a new international meanwhile, a number of systems trading stocks or piloting drones, and a system of memory wherein most consumers lose old files they will miss dearly but the well funded never forget a status update.

This is not because times can be authored freely by any genius using a computer. Every piece of software must be made up from an infinite slang of pre-existing conventions. Conventions combine to create violent video games, printer drivers, proxy servers, and mp3 players. Any two programs will have many conventions in common, even when it seems that they couldn't be more opposite (Montfort and Bogost 2009). The millions who work today building new software combine existing conventions to make new programs for purposes they have accepted as their own. We do not know all that these conventions can create because their

potentials change in the context of other conventions and situations, both of which are being created freshly on a daily basis.

Chapter five approached the grand pile of computer conventions in terms of time, asking how certain existing ways of doing things with computers effectively connect functions of temporality. These conventions of time are repeated, though enacted in different ways in practice. The cycle unites processors, programs, and the operating system's scheduler to pace activity, thus enabling interactive computing and real time systems. The same conventions also make software work iteratively and constrain the precision of events to the quanta of the scheduler. Memory is vulnerable in physical form, but very powerful as an immaterial procedure. Its conventions make movie piracy inevitable and data precarious or permanent depending on backup procedures. Networking establishes fast connections with latency, a network effect that makes readers hasty, and a contemporary moment always ahead of the world it describes. Following this approach, the time practices of media playback (mp3, DVR, Netflix), calendars, traffic updates, or saving in video games could be explored to see how the situated practices by which time actually operates have changed and with what effect.

The strange times enabled by computers parallel dominant times without disrupting them. They have grown up and around the older times and, for the present moment, do not conflict. The hour, day, and year are at least as important now as they were in 1937 when computers were not yet machines. But, in practice, the way we live times has changed.

A significant trend in these changes has been the production of times that are interactive. The conventions of computing, by themselves, could yield any number of speculative times. Yet, certain types of times have received support and have become defining parts of what it means to use computers, in many different contexts. A staple of these times is a degree of interactivity by which the user locally modulates functions of temporality.

Interactive times have several key commonalities. First, they are decentralized. The authoritative measurement of other time practices depended on a central concept of time, whether the hospital's main clock or the precise position of the Earth. Interactive times bend locally, pausing a show only on one TV. Second, interactive times empower individuals, often in their capacity as knowledge workers or geniuses. You are encouraged to control the flow of events, restore the files you want, or provide access to others as you see fit. Third, the products of computers bear the mark of interactive time. The very notion of what it means to do work on the computer has come to include interactive time as a presupposition. It is common knowledge that if you do something on the computer you can try out variations, go back to old versions, and fix mistakes more easily. But there is a more subtle influence beyond this. Editing together film by hand is quite different from using a computer, where all decisions are temporary, and the product tends to have more technical complexities, layers, and strange little tricks but less reflective thought than work done with actual film (Mangolte 2002). Similar arguments have already been made by others about word processing (Simpson 1995, 66; Eriksen 2001, 53) and music (Kramer 1988, 79; Loy

2007). Fourth, the way that someone controls interactive time has effects on other people. Whether by making a final product that others will receive or by dragging others along in some process, there is nothing about these times to keep them isolated to the private experience of the empowered individual.

This does not sound like the usual story of interactivity, freedom, customization, consumer empowerment, and efficient data-driven solutions we have come to associate with digital, interactive technologies. But it is the other side of the same story. Interactivity lets people make decisions on behalf of others as well as themselves. This freedom to choose is itself a condition creating its own foibles and biases. We put music on shuffle to avoid having to constantly choose a song; our ability to freely modify a bit of edited video changes the style of its final form. Customization produces a unique individual making his or her own decisions, who also seems quite dim and boring from the point of view of a scanner that sees us only darkly. Consumer empowerment is the battleground of consent by which groups, now understood as economic rather than political actors, fight for support. Data-driven solutions may be optimal relative to certain criteria applied to certain data sets, but usually these are marketing decisions made for various reasons that refer to data that was itself built on categories and interpretation decided by others earlier on for other reasons. It does not matter if everyone wants an undo command, companies decided to include one for their own reasons and it cannot be disabled.

For centuries, a dominant set of times has made canonical translations of others' times into its own terms. Sunlight and holidays became hours of a day and days of a

year. Today, the basis of a great many activities are shifting to the computer and to its conventions of time.

This is the occasion for a reciprocal translation of dominant times by those of the computer. The calendar, the clock, the measurement of labor, the hours a shop is open for business, the frequency of the news cycle, and the synchronization of work between different sites have already begun to shift on to a new basis in software. This has not eliminated old practices. In many cases, though it has changed them. Time practices that have sustained themselves outside the dominant alliance will now have their own opportunities to adopt new practices of time as they present themselves. Many of these new practices are interactive and exert a gentle influence in contemporary culture already, pulling us toward a time that is out of joint, that starts and stops, that is very present and precise, that operates in several ways in parallel, and yet can restore the past in a split second.

Time once derived from human religious and political institutions, traditions of paced change under divine control, or clocks synchronized to a central standard. Today, computers have emerged as a new force that creates and shapes times. Our new times are assemblages of various aspects of temporality and are the outcomes of multiple, flexible conventions. We live in cross-cutting temporal orders driven by machines. Of these times, only some are intentional. The death of storage media, the minima of network latency, and the moments defined by a quantum of processing power were not invented as bold new times for the human race or even exciting products for consumers. They are side effects of other engineering projects. However,

they make conditions in which we must act. These machine times have little to unite them, though many are interactive. They are complicated, there are many of them, and their effects are very different. They do not all accelerate social life, they do not all create a global village with instantaneous communications, and they do not spell the end of time. They instead indicate the rise of new times and give us a reason to become diligent about how these times work and what they do.

### References

- Adam, Barbara. 1995. *Timewatch: The Social Analysis of Time*. Cambridge, Mass: Polity Press.
- Bartky, Ian. 2007. *One Time Fits All: The Campaigns for Global Uniformity*. Stanford, CA: Stanford University Press.
- Dohrn-van Rossum, Gerhard. 1996. *History of the Hour: Clocks and Modern Temporal Orders*. Chicago: University of Chicago Press.
- Durkheim, Émile. 1995. *The Elementary Forms of Religious Life*. New York: Free Press.
- Eriksen, Thomas. 2001. *Tyranny of the Moment: Fast and Slow Time in the Information Age*. London, UK: Pluto Press.
- Glennie, Paul, and Nigel Thrift. 2009. *Shaping the Day: A History of Timekeeping in England and Wales 1300-1800*. Oxford: Oxford University Press.
- Harvey, David. 1989. *The Condition of Postmodernity: An Enquiry into the Origins of Cultural Change*. Cambridge Mass.: Blackwell.
- Jones, Tony. 2000. *Splitting the Second: The Story of Atomic Time*. Philadelphia: Institute of Physics Publishing.
- Kern, Stephen. 1983. *The Culture of Time and Space 1880-1918*. Cambridge Mass.: Harvard University Press.
- Koller, John M. 1974. "On Buddhist Views of Devouring Time." *Philosophy East and West* 24 (2) (April): 201–208.



- Kramer, Jonathan. 1988. *The Time of Music: New Meanings, New Temporalities, New Listening Strategies*. New York, NY: Schirmer Books.
- Lim, Bliss Cua. 2009. *Translating Time: Cinema, the Fantastic, and Temporal Critique*. Durham: Duke University Press.
- Loy, David R. 2007. "CyberLack." In *24/7: Time and Temporality in the Network Society*, edited by Robert Hassan, 195–215. Stanford, CA: Stanford Business Books.
- Luckmann, Thomas. 1991. "The Constitution of Human Life in Time." In *Chronotypes: The Construction of Time*, edited by John Bender, 151–166. Stanford, CA: Stanford University Press.
- Mangolte, Babbette. 2002. "Afterward: A Matter of Time." In *Camera Obscura, Camera Lucida: Essays in Honor of Annette Michelson*, 261–274. Amsterdam: University of Amsterdam Press.
- McCrossen, Alexis. 2007. "Conventions of Simultaneity." *Journal of Urban History* 33 (2) (January 1): 217–253. doi:10.1177/0096144206294738.
- Mills, David. 2006. *Computer Network Time Synchronization: The Network Time Protocol*. Boca Raton FL: CRC/Taylor & Francis.
- Montfort, Nick, and Ian Bogost. 2009. *Racing the Beam: The Atari Video Computer System*. Cambridge, Mass.: MIT Press.
- Nanni, Giordano. 2012. *The Colonisation of Time: Ritual, Routine and Resistance in the British Empire*. New York: Palgrave Macmillan.
- O'Malley, Michael. 1996. *Keeping Watch: A History of American Time*. Washington: Smithsonian Institution Press.
- Phillips, Stephen M. n.d. "A Short History of the Fourth Dimension." <http://www.smphillips.8m.com/a-short-history-of-the-fourth-dimension.html>.
- Simpson, Lorenzo. 1995. *Technology, Time, and the Conversations of Modernity*. New York: Routledge.
- Tanaka, Stefan. 2004. *New Times in Modern Japan*. Princeton, N.J.: Princeton University Press.
- Thompson, E. P. 1967. "Time, Work-Discipline, and Industrial Capitalism." *Past & Present* 38 (1) (December 1): 56–97. doi:10.1093/past/38.1.56.