

# 11

---

## *Permanently Beta*

Responsive Organization in the internet Era

***Gina Neff***

*Columbia University*

***David Stark***

*Columbia University and Santa Fe Institute*

**H**ow has the internet influenced economic organization? Rather than approach this question by examining the productivity gains produced by internet technologies or the roles that dot-coms play in the economy, this chapter examines how the *process* of technological change influences the organization of economic life. The social effects of the internet include emerging organizational innovations in addition to technological innovations. Although skeptics may point to failed dot-coms or the internet's broken promises of productivity gains, the values embedded in widely used information technologies have become encoded into the routines of the market and into organizational forms. Thus, the internet has already transformed economic life not just through direct influence on the marketplace but also through encoding these new routines and new ways of doing things into daily practice.

---

AUTHORS' NOTE: The authors thank the volume editors for their insightful comments as well as Pablo Boczkowski, Danyel Fisher, Gernot Grabher, Rajiv Shah, and the participants in the National Science Foundation's internet Scholars Program at the University of Maryland for comments on an earlier draft. This chapter is adapted from a longer article that is available online ([www.coi.columbia.edu/workingpapers.html](http://www.coi.columbia.edu/workingpapers.html)).

Many economic and organizational processes have been *informationalized* since the diffusion of internet technologies (Castells, 2000), but informationalization alone could not create a *new* economy. Information during this era may indeed “yearn to be free,” but it has yet to figure out how to break the boundaries of existing economic forms considering that the economy is still organized around information having a price. During the internet stock market frenzy, the conflict between information’s openness and profitable business models symbolized the differences between these technological and economic values. Technological advances and the values of the people involved in creating them may have promoted information’s openness, but social and economic organization has yet to catch up. In those realms driven by a sense of public or collective ownership, such as in e-government initiatives, nonprofit and social movement organizations, and community-developed projects (e.g., Linux), interactivity has meant somewhat freer access to information. But openness in general, it seems, is not yet a market value.

However, organizational forms and technology are influencing one another in unexpected ways, coevolving toward more open structures in both that may present the opportunity for broader participation in the design of both products and organizations. The coding of software and information technologies has the potential to extend beyond the economic implications for particular products to the rewriting of social codes and patterns of behavior, influencing not only what activities happen online but also the ways in which things are done offline.<sup>1</sup> The push toward more open structures is no exception, even as tensions remain between openness and control, between market and community, and between conflicting values. Computer protocols, as Galloway (2001) wrote, are “how control exists after decentralization”<sup>2</sup> and are deeply influenced by the values of the designers, organizations, and societies that create those protocols. The heyday of the new economy is over, but left remaining are the tensions between the market values of technological innovation and the community values of openness enabled by the internet.

One of these values of the contemporary economy embedded into the internet and computer technologies is what Manovich (2001) characterized as “variability.” Unlike mass-produced objects, programmed products are never stable, can have a potentially infinite number of versions, and can be customized for a particular user almost immediately (p. 36). Software is “patchy”; versions change, systems evolve, and applications die.<sup>3</sup> This cycle shapes the organizational and economic forms around it, so that adaptability, flexibility, and responsiveness become the norms within the technology industry and, increasingly, throughout the economy as a whole. Adaptation to variability emerges partly due to the

pervasiveness of information technologies but also due to the values of a postindustrial economy that favors just-in-time and customizable production over mass production. The internet or any technology, of course, does not exist outside of society; the variability of the internet both embodies and magnifies the ongoing effects of the shift away from industrial production.

Variability produces organizational flux. Organizations' bureaucratic structures are destabilized because the challenges of responsiveness are too great for institutional routinization. "Heterarchies"—flatter organizational structures with distributed accountability, decentralized decision making, and multiple (often competing) evaluative principles—emerge (Stark, 1999). This chapter argues that the internet, in addition to magnifying the variability of production, enables a closer connection between products' design and their use—a connection that, in turn, transforms organizational processes. Continual change necessitates responsiveness in the design of both products and organizations. The influence of design—whether the design of products, technology, or services—and organizational form on each other emerges partly due to the process of continual technological change in which the cycle of testing, feedback, and innovation facilitates ongoing negotiations around what is made and how to organize making it. We call the organizational state of flux that emerges from this negotiation "permanently beta." *Permanently beta is a fluid organizational form resulting from the process of negotiation among users, employees, and organizations over the design of goods and services.* The instability associated with being permanently beta is not without social costs, but it may present opportunities for organizing broader participation in the design of products and organizations.

Countering the social costs of instability depends on the level of organization of participants—users and employees alike. Within permanently beta organizations, as products and the structures around them are being tested, individual users and employees bear a greater responsibility for adaptation to change. The "changing scripts" of workplaces, as Kunda and Van Mannen (1999) argued, force adaptation by those who work in permanently beta organizations. Continual reconfiguration in heterarchical organizations can be exhausting, especially in work environments organized around projects that exert extraordinary time pressures and demand a fast pace of action, as Grabher (2002) found in his study of British advertising agencies. The responsiveness required for the flexibility and adaptability of work puts new pressures on employees, thrusting a "large number [of people] into a condition of permanent survival-oriented tension" in "unfettered" organizations within an information-intensive economy (Child & McGrath, 2001).

But this permanently beta ethic of continual change can be thought of, borrowing from Max Weber, as a key influence on the spirit of the information economy. Consider the following “manifesto” written by a Web design company during the spring of 1997:

For better or worse, we have decided to enter into an industry that does not make things that enjoy a spatial or temporal existence. . . . *Since the world is in constant flux, any work that is truly integrated into its environment can never be viewed as a finished entity*, but rather a point in an ongoing dialectical process. . . . Our approach is not about design in any traditional sense. Instead, we make work that may be adapted to people’s differing needs and contexts. *Internally, we must practice what we preach*. . . . Through the open design of both our physical and electronic environments, we will foster the exchange of information between our employees and allow them to share ideas and engage in critical dialogue.<sup>4</sup> (emphasis added)

Besides the fact that a company—especially one that has produced major Web sites for blue-chip clients such as Motorola and Sony—would even issue a manifesto, there are three aspects of change and adaptability that are well worth noting. First, this manifesto assumes ongoing instability both in internet design and in the larger environment around that design. Second, the manifesto challenges a traditional sense of design that believes in a final product that is neither responsive to its users nor adaptable in its use. Finally, this particular dot-com wants to “practice” internally what it preaches by encouraging the openness of organizational forms within the company. Just as Weber understood the rise of Protestantism linked inexorably to the rise of capitalism, this manifesto proclaims an almost religious belief in “constant flux” that is occurring alongside a digital reformation of capitalism.<sup>5</sup>

Three aspects of applications design—beta testing, encoded responsiveness, and community development—illustrate how permanently beta organizational structures emerge alongside older organizational forms.

### **Never Leaving the Beta Phase**

Permanently beta encompasses the social implications of continual technological change and is, of course, a play on the term for software testing. Software and

internet sites that are being tested are called *beta versions*, and strictly speaking, permanently beta would be a product that never leaves the test phase. The internet makes it possible to distribute products that are continually updateable and almost infinitely customizable—products that, in effect, never leave a type of beta phase.

According to Techweb's *TechEncyclopedia*, a beta version is "a pre-shipping release of hardware or software that has gone through [an] alpha test." It is "supposed to be very close to the final product, but in practice, it is more a way of getting users to test the software . . . under real conditions." Rather than duplicate "the myriad of configurations that exist in the real world," beta tests expose software to those conditions and users.<sup>6</sup> According to CambridgeSoft, a company that makes software for life sciences research, the benefits to beta testers include getting a look at the new features before anyone else does, the "pleasure" of finding unsuspected bugs, making the software better as a result of detecting those bugs, and perhaps affecting the company's future direction of development through beta testers' suggestions.<sup>7</sup>

Beta testing is not without risks: CambridgeSoft states that it "pretty much guarantee[s] that you will receive buggy software" that "may crash your computer (or worse)." Being one of the first to have a new application, experiencing the "pleasure" of debugging, and becoming involved in "your!" software development, as well as the possibility of free or cheaper software, outweigh these risks for many. More than 2 million people volunteered to be one of the 20,000 beta testers for the new version of Napster (Ross, 2002). Although Apple's "public beta" release of OS X, its first completely new operating system since 1984, cost \$29.95, thousands downloaded it despite reports that it was still quite "buggy" and the fact that little compatible software was available. Beta users saw the long-awaited new operating system 6 months before its first commercial release, and Steve Jobs, Apple's chief executive officer, thanked them in advance by saying, "We're excited to have our users test drive this public beta version and provide us with their valuable feedback."<sup>8</sup> Apple fans and the press provided invaluable buzz about OS X as they tested it. Beta may still have bugs, but beta testers are among the first outside of a company to have the new product.

A typical nomenclature denotes beta versions. Dyson (1997) explained the numbering of beta versions and continually updated versions in her book, *Release 2.0*:

*The very title of this book embodies the concept of flexibility and learning from errors:* In the software business, “Release 1.0” is the first commercial version of a new product, following test versions called 0.5, 0.8, 0.9, 0.91, 0.92. It’s fresh and new, the realization of the hopes and dreams of its developers. It embodies new ideas, and it is supposed to be perfect. Usually the vender comes out with Release 1.1 a few months later, fixing unexpected bugs and tidying up loose ends. . . . Release 2.0 is a total rewrite, hammered out by older, wiser programmers with feedback from thousands of tough-minded, skeptical users. Release 2.0 is supposed to be perfect, but usually Release 2.1 comes out a few months later. (p. 5, emphasis added)<sup>9</sup>

Commercial products do eventually leave the beta stage in “shrink-wrap” or as purchasable products for consumers. The quote from Dyson, however, points to software development’s never-ending cycles of innovation, real-world testing, feedback, and revision, a cycle that is much shorter than that in manufacturing design but longer than that in Web site development. The internet is compressing this cycle to the extent that, as suggested by Garud, Jain, and Phelps (n.d.), “it is difficult to distinguish between one product generation and the next” (p. 3). They examined the releases of Netscape, which had 39 beta versions between the beta stage of Navigator 1.0 and the release of Communicator 4.0. In the words of Marc Andreessen, the founder of Netscape, the philosophy behind so many beta releases was to “kick it out the door. It may not even work reliably. . . . [but] go out and get feedback. . . . [Customers] will tell you, often in no uncertain terms, what’s wrong with it and what needs to be improved” (quoted in Garud et al., n.d., p. 14). Andreessen’s attitude toward building software that could rapidly integrate user response into the design—not simply the number of beta versions—made the early days of Netscape permanently beta.

Netscape risked the reliability of its product for increased responsiveness to its users. However, beta testing does not always incorporate users into the design process or treat users as a community with a stake in the outcome of the product. In fact, several large software makers are known for their hierarchical organization and rigid divisions between product design and product testing. Although users’ participation in design and redesign of products may be limited in practice, beta testing as a process provides a model for how to involve users more fully in products’ design cycles. *Permanently beta* as a phrase highlights this aspect of product variability during the internet era and the impact of that variability on the line between users and producers.

## Encoded Responsiveness

Permanently beta forms also emerge where responsiveness is designed into products and organizations. Continual testing is one way in which to incorporate user response, but responsiveness can be encoded into products and organizations in other ways.

Open source projects are one such example where the line between users and producers of a product is blurred. Open source projects are a hodgepodge of formal and informal organization; for-profit and volunteer organizational goals, processes, and values; and hierarchical and heterarchical modes of organizing. In a now classic essay on open source, *The Cathedral and the Bazaar*, Raymond (1999) described Linus Torvald's management of the development of the operating system Linux. Raymond pointed to two key aspects of open source philosophy that challenge traditional models of software engineering: "release early and often" (p. 38) and "all bugs are shallow with enough eyes" (p. 41). At one point in the evolution of Linux, a new kernel (the most fundamental part of an operating system) was released *daily*. So long as the volunteers examining the code could see that their input mattered, they would keep testing it, pushing the project to its limits and ensuring that the code was sound. Bugs, the inevitable glitches in any programming project, could best be eliminated not through the attempts of a few people to release flawless software but rather through the efforts of many to examine software in use, identify problems, propose fixes, and watch carefully to ensure that good ideas are implemented. Treat the users of a program as codevelopers, Raymond argued, and they will act like codevelopers. Raymond's cathedrals and bazaars describe more than an approach to debugging software. Responsiveness to users in the design of products has the power to influence organizational form.

When a user downloads a version of Mozilla, an open source and community-developed<sup>10</sup> internet browser, the user is greeted with the following friendly message:

Congratulations! You've downloaded a Mozilla build. This means that you've volunteered to become part of the Mozilla testing community. Great! Welcome aboard. Helping out won't take much of your time, doesn't require special skills, and will help improve Mozilla.<sup>11</sup>

Mozilla was the original code name for the product that became known as Netscape Navigator.<sup>12</sup> Mozilla.org, the main coordinating group of Mozilla source

code, is a nonprofit organization that began under the aegis of Netscape to develop the source code on which its products were based. Mozilla has now grown into its own open source program, quasi-autonomously of Netscape, and at the time of this writing it was releasing beta versions of an internet browser.<sup>13</sup> According to its mission statement, the Mozilla organization provides the technical and architectural direction for the Mozilla project, synchronization of the releases of the browser and code, coordination of the discussion forums, and “roadmaps” to help organize projects based on the code. However, “We are *not* the primary coders. Most of the code that goes into the distribution will be written elsewhere, both within the Netscape Client Engineering group, and increasingly, *out there* on the net, at other companies and other development organizations” (emphasis added).<sup>14</sup> Just as those testing Mozilla are part of a community, so too are those who actually do the work of developing the code for a new browser, and the organization exists primarily to coordinate these independent programmers. To that end, Mozilla.org promises that it will, above all, “be flexible and responsive. We realize that if we are not perceived as providing a useful service, we will become irrelevant, and someone else will take our place.”<sup>15</sup> Without responsiveness, the organization would become irrelevant to the community of volunteers organized around developing the code, but without some form of coordination, the project would not have developed into a fully usable Web browser.

Thus, a key aspect of permanently beta is the practice of *design-in-use*, in which products are formed by their use, not simply in their design. Open source projects, for example, display a responsiveness that has allowed users of products to be more directly involved in the design-in-use of these products. Open source models also challenge the traditional notion of boundaries between organizational forms in that they often have more in common with social movements than with traditional formal organizations. O’Mahony (2002) argued that the processes of conflict and compromise in open source projects explain the synthesis of old and new institutional elements into emergent new organizational forms (p. viii). Open source projects combine these elements of organization from community and commerce into a new hybrid organizational form.

Open source projects are not necessarily permanently beta, but they often provide a good example of what von Hippel (2001) called “user innovation communities” (p. 82). These communities are not just in software and hardware production. Von Hippel described the user innovations in high-performance windsurfing, where windsurfers testing out innovations on the open beach share them with each other and ultimately with equipment manufacturers (p. 85). The

internet can be an open beach of sorts, with users able to “see” one another online and share innovations. Permanently beta organizational forms arise when user innovation communities overcome the “stickiness” of innovation transfer back into the manufacturing or production process. This occurs in user-produced projects where there is “product development without manufacturers” (p. 82), but it also can occur in settings where user feedback is more fully integrated into the design process. By shortening the cycle of feedback and reengineering, creating reputational benefits for users who innovate, and fostering communities around innovation, the internet strengthens these feedback loops and reinforces the process of continual updating.<sup>16</sup> Responsiveness becomes encoded into software products.

Just as the internet lowers the costs of cooperation in user innovation communities, it can also help to overcome the difficulties of transferring those innovations back to the manufacturer in industries other than software. Users “consume, modify, domesticate, design, reconfigure, and resist technologies,” and through this process they shape and are shaped by those technologies (Oudshoorn & Pinch, in press). The “working relations of production,” as Suchman (2002) termed it, is an “increasingly dense and differentiated layering of people, activities, and things, each operating within a limited sphere of knowing and acting that includes variously crude or sophisticated conceptualizations of the others.” However, the interactions between users and technology occur most often in private settings where users are separated from the producers, unlike the more open, visible, and public interactions with technology that can occur online (Boczkowski, 2001). Permanently beta organizations emerge when the institutional barriers to user involvement in the design process are overcome, and the internet and other interactive tools can facilitate this involvement. Open source communities provide such example of users becoming producers, but other kinds of community building online can bring users into the production process.

## **Community Development**

Increasingly, commercial internet sites interested in creating “community” are learning that design-in-use instead of top-down design is the best way in which to recruit and maintain users. For a project on work in Manhattan’s “Silicon Alley,” one of us (David Stark) conducted field research at a company that

developed an internet community and e-commerce site.<sup>17</sup> BetaTeen.com (a pseudonym) began as a content-driven online magazine trying to become “America’s online high school newspaper.”<sup>18</sup> In its original business model, BetaTeen created a youth community for commercial access. As young people came onto the site for the news and entertainment that editors and writers considered relevant, the teens would provide a targeted demographic group for marketing and focus research.

The teen users staged a minor revolt, getting involved in the design-in-use in several ways. First, by tracking how teens used the site, the editors found that teens were more likely to read essays by other teens, and a user-as-producer model of content emerged. As more teens participated in creating the content on the site, traffic increased. BetaTeen’s executive vice president explained, “We don’t have people sitting around thinking, ‘What do teens want?’ It doesn’t work. Even if you could figure it out, it wouldn’t last. You can try to write for them, but it doesn’t work. Now, 95% of our content is written by teens themselves.” The teens “want to give their opinions” and “want to be in the spotlight,” and BetaTeen tries to give them a sense of both. Referring to the teens, the executive vice president said, “*They own BetaTeen. We just put up the framework.*” Second, the teens themselves began making demands of the editors to be allowed greater interactivity on the site, more control over the commercial use of information about themselves, and the ability to refuse to be “marketing guinea pigs” for the company. The original intent, according to BetaTeen’s executive vice president, was as follows: “We know best. We create the stuff; you use it.” But interactivity demanded a responsiveness on the part of BetaTeen to its users’ demands and the integration of users into the process of content production. The design of the “product”—from long essays to short, chat-driven, user-written opinions—incorporated the demands of those who used it.

Boczkowski (2001) identified this new form of media production as “distributed construction,” which emerges from an information architecture that inscribes users as “co-constructors” of content, a network of multidirectional flows of messages across the medium, and content production characterized by “relationships of interdependence, distributed authority, and multiple rationalities” (p. 30). The interactivity of online communication can create communities that can exist quasi-independently and transparently from manufacturer or producer control. As on BetaTeen, a company can build a framework in which users find each other but then can ultimately communicate independently of it. Once BetaTeen users see one another online, they no longer necessarily need BetaTeen

to connect them. Users of nearly any product can find each other to share ideas, talk about product use, or organize to make demands of the manufacturer (such demands are more easily communicated back to the producer through the current structure of the medium). This openness in communication design creates the possibility for permanently beta structures.

Creating community in commercial settings does not guarantee permanently beta organizations. Community values and commercial values are often at odds, but as Epstein (1995, 1997) wrote about the community of AIDS activists and their clashes with pharmaceutical companies, these differences in values can sometimes produce negotiations toward a mutual goal. AIDS activists wanted wider access to health care, including experimental new drug treatments, whereas companies wanted to design and market new for-profit drug treatments. Although two sides' negotiations did not make drug companies community oriented, changes in the approval process did incorporate many of the users' demands. In many traditional settings, users might not have the same goals as those who design the products, and beta testers might not consider themselves part of the same community as software engineers (and vice versa). We call these "practicing communities," which link organized users with professional expertise so as to inform the design process. In a practicing community, an organized group of users become acknowledged as experts in how products are used and can sometimes realize this power to influence design.<sup>19</sup> From product design to organizational design, permanently beta settings have this potential to be participatory.

## **Architecture in Code**

These permanently beta examples point to a process that might be called *collaborative engineering*. Although typically referring to the relationship among producers and not between producers and users, collaborative engineering is "a discursive pragmatics" that allows for the organization of rivalrous logics, values, and organizational principles (Girard & Stark, 2002, p. 1929). Sabel and Dorf (1998) referred to the process of *simultaneous engineering*, a concurrent design process by which separate teams develop different proposals for the final design. Permanently beta is, in part, a form of simultaneous and collaborative design and engineering that brings users into the process. Software beta testers want a first look at new versions of software, whereas software companies need their

experience to help determine, with little or no pay (and in the case of Apple, at a *cost* to the testers themselves), the quality of the software. The BetaTeen users want the spotlight on their stories, whereas the BetaTeen editors need teens to visit the site to support e-commerce and marketing functions that make money. Each of these examples points to different sets of values along a divide, and in each of the examples, those who use the software, read the content, or test for bugs gain some voice in the design process. These are not examples of competing companies vying for a contract through simultaneous engineering codes or of subcontractors working under collaborative organizational principles. The various actors in permanently beta settings, however, hold disparate values and principles that must be negotiated in a similar manner.

Through these testing forms, experimentation, and negotiation, values are incorporated into the design of the products themselves. Permanently beta forms produce products that are negotiations in themselves, like the multiple versions of software with its subsequent multiple betas, release versions, and patches. The design process is considered as ongoing rather than as having a final end point in that each of those releases offers an opportunity to go back and incorporate options or fixes left out previously. Bringing users into the design and testing involves a genuine interaction with the users, not an attitude of “we know best,” to recall the words of BetaTeen’s executive vice president. Table 11.1 summarizes the differences between permanently beta and traditional approaches to design. Permanently beta forms necessarily leave things out to be completed by the users and are like architected designs that are left partially open to the interpretation of the engineered execution. Practicing communities are enabled in permanently beta situations to link lay knowledge to expertise, constant change to responsiveness, users to producers, and buyers to manufacturers.

The internet allows us to see permanently beta in action and to become accustomed to its rhythms. The Web, at least at this juncture in its history, is quite an unstable place; Web sites die, disappear, and are modified in a flash. More important, the internet facilitates users being a part of that continual updating. Web sites such as Plastic.com and Slashdot.com recycle “the Web in real time” (as Plastic says on its site), manifesting a permanently beta approach to news. Users of these services continually update the news, not with new reporting but rather by catching the “bugs” in published media reports, drawing connections between stories, and appending their own commentaries and analyses onto circulating stories. Digital media allow this bricolage to be formed out of the pieces found online, forming a permanently beta news that is constantly updated, analyzed, reconfigured, and tested by nonreporters.

**Table 11.1** Permanently Beta Design Versus Traditional Design

|                       | <i>Permanently Beta<br/>Design Process</i> | <i>Traditional<br/>Design Process</i>        |
|-----------------------|--|--|
| Product               | Multiple versions                          | End product                                  |
| Design process        | Design-in-process                          | Design with a user in mind                   |
| Use                   | Interactive; flexible<br>and adaptable     | User-friendly; easy<br>to use but inflexible |
| Conception of user    | User as designer                           | User as consumer                             |
| Communication of user | Consciously voiced<br>preferences          | Revealed preferences                         |
| Community metaphor    | Practicing communities                     | Professional expertise;<br>isolated users    |
| Model of use          | Participation                              | Consumption                                  |
| Computer metaphor     | Adaptability                               | Usability                                    |

Users of permanently beta products may also find the experience to be frustrating. Products are not final, clean end versions but rather are destabilized, constantly changing products. Users of computer software understand the continuous updating of applications; those who have experienced the “bugginess” of new versions understand all too well the downside of continual change. Having to reconfigure ever-changing products is part of what Terranova (2000) termed “extraction of value out of continuous, updateable work” that exploits the “free labor” of users in a digital economy (p. 48).

The “new economy,” too, has shown us how quickly economic experimentation can end. Real jobs were lost just as quickly as stock option millionaires were created. Digital landscapes are much faster than our physical ones, as Girard and Stark (2002) pointed out about sites that have closed: “An abandoned warehouse is a boarded-up blight on the landscape until it is destroyed or gentrified into luxury apartments. An abandoned Web site is a Code 404, ‘File Not Found’” (p. 1947). Although many innovative start-ups of Silicon Alley and Silicon Valley pushed organizational logics to their brink, those involved felt just that—*involved*. One veteran of Silicon Alley felt that he was creating “the freest medium around” (Neff, 2003).<sup>20</sup> For many of these start-ups, the new economy boom meant that they were involved in the design of their companies—creating new kinds of work, new ways of collaborating, and new ways of relating their jobs to their lives—as much as they were involved in creating new products. Not everyone was so lucky. Income inequality grew in the United States as the internet revolution was taking place, and during the latest economic boom more jobs were created in low-end service work—where employees often have no autonomy or voice in

how their work is organized—than in high-end knowledge work. As silicon mavericks were testing their sites, their organizations, and themselves, other people were forced to adapt to an economy where the rules were rapidly becoming less favorable to them.<sup>21</sup>

If, as has been said, architecture is politics set in stone, then information architecture is politics in code. Code is not set in stone as buildings literally are, but is it as mutable as we would like to think? Code's rigidities, including path dependencies and legacies in both technological and social systems, shape what can come next. The values embedded in code at one point become the structuring factors of future development. These values are not mutable simply because of continual technical change, nor are the values in code necessarily good or bad because they have a political valence. Technological artifacts such as codes have values, and as a society we can either actively engage in the negotiations around these values or choose imprudently to ignore them.

There are no guarantees, however, that this process of negotiation over encoded values will lead to a settlement on a single value or strategy. Several scholars have outlined how multiple rationalities of production can exist simultaneously, often without mutual comprehension between those with coexisting values (Boczkowski, 2001; Galison, 1997; Star & Griesemer, 1989; Stark, 1999). Nor are innovation and creativity in "practice" necessarily integrated into the organizational process (Brown & Duguid, 2001, p. 93). Permanently beta structures show how users and producers can actively shape the technology around them. These new economic and organizational structures can be tested and negotiated, and they may prove ultimately to be more inclusive. The lack of settlement on a single path in permanently beta settings can be confusing, frustrating, or worse. Permanently beta affords the possibility of influencing which values are encoded into organizations and technologies—and for users to incorporate *their* values into the structures around them.

## Notes

1. See Thrift (2001b) for more on software code's passion for inscription.
2. Galloway's (2001) dissertation is one of the first we know of that used computer protocols as its literary texts.
3. Indeed, the name of one program, *Apache*, is a play on the patches that programmers use to smooth out software. For more on *Apache* and other open source projects, see Kogut and Metiu (2001).
4. See [www.plumbdesign.com/manifesto](http://www.plumbdesign.com/manifesto).

5. We are not the first to borrow from Weber to make a point about a new spirit of capitalism. Our analogy owes much to the theory laid out in both Castells's (2000) trilogy, *The Rise of the Network Society*, and Boltanski Chiapello's (1999) book, *Le Nouvel Esprit du Capitalisme*.
6. See [www.techweb.com/encyclopedia](http://www.techweb.com/encyclopedia).
7. See [www.cambridgesoft.com/about/betatesting.cfm](http://www.cambridgesoft.com/about/betatesting.cfm).
8. A press release is available at [www.apple.com/pr/library/2000/sep/13macosx.html](http://www.apple.com/pr/library/2000/sep/13macosx.html).
9. Dyson's *Release 2.1*, the paperback version of the book, came out in October 1998.
10. O'Mahony (2002) pointed out the difference between source code being open and community development of software. Several, but not all, open source projects are community developed, and Mozilla and Linux are two examples of community-developed open source projects.
11. See [www.mozilla.org/start](http://www.mozilla.org/start).
12. Mozilla stands for the "Mosaic Killer" or the application that would replace Mosaic, the first graphical interface application for the internet.
13. Mozilla can be downloaded for use at [www.mozilla.org](http://www.mozilla.org). Mozilla's first "release" was in June 2002.
14. See the Mozilla.org mission statement at [www.mozilla.org/mission.html](http://www.mozilla.org/mission.html).
15. Again, see the Mozilla.org mission statement at [www.mozilla.org/mission.html](http://www.mozilla.org/mission.html).
16. Kollock (1999), in his now classic essay on online cooperation, identified these mechanisms as supporting exchange online.
17. For more on this research project area in general, see Girard and Stark (2002).
18. All quotes about BetaTeen are cited directly from field notes and from interviews with BetaTeen management.
19. Changes in product design can influence organizational structure, as Epstein (1995, 1997) noted in the case of the AIDS drug approval process changing the organization of the Food and Drug Administration.
20. This quote is taken from an interview transcript for Neff's research on uncertainty in Silicon Alley with the editor of an online division of a major publishing house in 1997.
21. See Smith (2001) for more on worker adaptation within the new economy. See Thrift (2001a) for more on the rhetorical strategies of finance that enabled the technological boom of the late 1990s to even occur.

## References

- Boczkowski, P. (2001). *Affording flexibility: Transforming information practices in online newspapers*. Unpublished doctoral dissertation, Cornell University.
- Boltanski, L., & Chiapello, E. (1999). *Le Nouvel Esprit du Capitalisme*. Paris: Éditions Gallimard.
- Brown, J. S., & Duguid, P. (2001). Creativity versus structure: A useful tension. *MIT Sloan Management Review*, 42(4), 93–95.
- Castells, M. (2000). *The rise of the network society* (2nd ed.). Oxford, UK: Blackwell.
- Child, J., & McGrath, R. G. (2001). Organizations unfettered: Organizational form in an information-intensive economy. *Academy of Management Journal*, 44, 1135–1148.
- Dyson, E. (1997). *Release 2.0: A design for living in the digital age*. New York: Broadway Books.
- Epstein, S. (1995). The construction of lay expertise: AIDS activism and the forging of credibility in the reform of clinical trials. *Science, Technology, & Human Values*, 20, 408–437.
- Epstein, S. (1997). Activism, drug regulation, and the politics of therapeutic evaluation in the AIDS era. *Social Studies of Science*, 27, 691–726.

- Galison, P. (1997). *Image and logic: A material culture of microphysics*. Chicago: University of Chicago Press.
- Galloway, A. (2001). *Protocol: Or, how control exists after decentralization*. Ph.D. dissertation, Duke University.
- Garud, R., Jain, S., & Phelps, C. (n.d.). *Unpacking internet time innovation*. Unpublished manuscript, New York University.
- Girard, M., & Stark, D. (2002). Distributing intelligence and organizing diversity in new media projects. *Environment and Planning A*, 34, 1927–1949.
- Grabher, G. (2002). The project ecology of advertising: Tasks, talents, and teams. *Regional Studies*, 36, 245–262.
- Kogut, B., & Metiu, A. (2001). Open source software development and distributed innovation. *Oxford Review of Economic Policy*, 17, 248–264.
- Kollock, P. (1999). The economies of online cooperation: Gifts and public goods in cyberspace. In M. A. Smith & P. Kollock (Eds.), *Communities in cyberspace* (pp. 220–242). London: Routledge.
- Kunda, G., & Van Mannen, J. (1999). Changing scripts at work: Managers and professionals. *Annals of the American Academy of Political and Social Science*, 561, 64–80.
- Manovich, L. (2001). *The language of new media*. Cambridge, MA: MIT Press.
- Neff, G. (2003). *Organizing uncertainty in Silicon Alley: Work and entrepreneurialism in New York's internet industry, 1995–2000*. Unpublished manuscript, Columbia University.
- O'Mahony, S. (2002). *The emergence of a new commercial actor: Community managed software projects*. Unpublished doctoral dissertation, Stanford University.
- Oudshoorn, N., & Pinch, T. (in press). Introduction. In N. Oudshoorn & T. Pinch (Eds.), *How users matter: The co-construction of users and technologies*. Cambridge, MA: MIT Press.
- Raymond, E. (1999). *The cathedral and the bazaar: Musings on Linux and open source from an accidental revolutionary*. Sebastopol, CA: O'Reilly & Associates.
- Ross, R. (2002, January 11). Born-again Napster takes baby steps. *Toronto Star*, p. E-4.
- Sabel, C. F., & Dorf, M. C. (1998). A constitution of democratic experimentalism. *Columbia Law Review*, 98(2), 267–529.
- Smith, V. (2001). *Crossing the great divide*. Ithaca, NY: Cornell University Press.
- Star, S. L., & Griesemer, J. R. (1989). Institutional ecology, “translations,” and boundary objects: Amateurs and professionals in Berkeley's Museum of Vertebrate Zoology, 1907–39. *Social Studies of Science*, 19, 387–420.
- Stark, D. (1999). Heterarchy: Distributing intelligence and organizing diversity. In J. Clippinger (Ed.), *The biology of business: Decoding the natural laws of enterprise* (pp. 153–179). San Francisco: Jossey-Bass.
- Suchman, L. (2002). *Located accountabilities in technology production*. Unpublished manuscript, Lancaster University. Retrieved April 27, 2003, from [www.comp.lancs.ac.uk/sociology/soc039ls.html](http://www.comp.lancs.ac.uk/sociology/soc039ls.html)
- Terranova, T. (2000). Free labor: Producing culture for the digital economy. *Social Text*, 18(2), 33–58.
- Thrift, N. (2001a). “It's the romance not the finance that makes the business worth pursuing”: Disclosing a new market culture. *Economy & Society*, 30, 412–432.
- Thrift, N. (2001b, February). *Software writing cities*. Address given at the Taub Urban Research Center, New York University.
- von Hippel, E. (2001). Innovation by user communities: Learning from open-source software. *Sloan Management Review*, 42(4), 82–86.