

Permanently Beta: Responsive Organization in the Internet Era

Gina Neff and David Stark

Center on Organizational Innovation
Columbia University
420 W. 118th Street
New York, NY 10027
ginasue@panix.com
dcs36@columbia.edu

Forthcoming in *The Internet and American Life*, Philip E.N. Howard and Steve Jones, editors,
Thousand Oaks, CA: SAGE, 2003.

Permanently Beta: Responsive Organization in the Internet Era

Gina Neff and David Stark
Columbia University

Abstract

How has the Internet influenced economic organization? Many approach this question strictly economically by examining the productivity gains from particular technological advances or the roles that dot-coms and other Internet-based organizations play with the economy. We approach this question differently—we move away from a macro-social analysis to consider the co-evolution of new technologies and organizational forms. In other words, how has the *process* of technological change in the Internet era influenced the way we organize economic activities? In this chapter we discuss how information technologies foster the emergent design and user-driven design of websites and other online media, as well as products and organizations off-line. We also consider how to mitigate the social costs of these changes.

Permanently Beta: Responsive Organization in the Internet Era

Gina Neff and David Stark
Columbia University

Introduction

Many economic and organizational processes have been, as Manuel Castells phrased it, *informationalized* since the diffusion of Internet technologies.¹ The Internet, however, could not by itself create a *new* economy. Information in this era may indeed “yearn to be free,” but it has yet to figure out how to break the boundaries of existing economic forms, for the economy is still organized around information having a price. During the Internet stock market frenzy, the conflict between openness in information and profitable business models symbolized the differences between these technological and economic values. Technological advances may have promoted information’s openness, but social and economic organization has yet to catch up to these advances. Interactivity freed information or made it more freely available in realms driven by a public sense of ownership, such as e-government initiatives, non-profit and social movement organizations, and community-developed projects like Linux. Openness, it seems, is not yet a market value.

The coding of software and information technologies does, however, have the potential to re-write social codes. Software codes incorporate this drive for openness into the technological products that we use. Code has, as Nigel Thrift wrote, “a passion for inscription” since codes are rules that “operate at a distance, so that too often, the code seems to have little to do with the situation to which it is applied.”² The political and economic implications of software and internet codes extend beyond the impact of particular applications to patterns of behavior and social routines encoded into digital form. While skeptics about the role of the Internet in society may point only to failed dot-coms or productivity promises of business applications, we need to consider those ways in which the values of information technologies have become encoded into the routines of the market and into organizational forms. The Internet has influenced the way work is organized, not necessarily because dot-coms transformed the marketplace, but through the structural influence of encoding these routines into software practice, influencing not only what activities happen online but also the ways in which things are done offline. In this respect, organizational forms and technology are influencing one another in unexpected ways, co-evolving towards more open—and possibly more democratic—structures.

If architecture is, as has been said, politics in stone, then information architecture is politics in code. Code has a politics, which if understood, point to the possibilities for positively influencing social structures through data structures.³ At the same time computer protocols are, as Alex Galloway writes, “how control exists after decentralization.”⁴ The new economy

¹ Castells, 2000.

² Thrift, 2001.

³ See Sach for more on a theory of democracy in code. For more on the politics of databases in particular see *Sorting Things Out*.

⁴ Galloway’s dissertation is one of the first we know of that uses computer protocols as its literary texts.

is dead, but left standing are the tensions it wrought between the market values of technological innovation and the values of openness and community enabled by the Internet.

Programmed products, including software and Internet sites, are never stable products. The software development process leads to a continual cycle of revision and testing. Software is “patchy”: Versions change, systems evolve, applications die.⁵ 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, due to the pervasiveness of information technologies, throughout the economy as a whole. Testing new versions, be they software or organizational structures, becomes a never-ending process. One of the outcomes of such structural influences of software and Internet design is an organizational state of flux that we call Permanently Beta. Permanently beta is a state of responsiveness in organizational form and process that mirrors innovation in products and services. The *process* of continual technological change necessitates a responsiveness to change through openness in organizational form, adaptability by employees, and, in the most positive form of permanently beta, broad participation in design. Instability and testing—of products, users, organizations and employees—are not without social costs.

If, to borrow from Max Weber, we think about the influences of the spirit of the new economy, then it could well be a permanently beta ethic of continual change—dedication to which shapes products, organizations and the people who make and use them. Consider the following from a “Manifesto” written by a Web design company in 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.⁶

Besides the fact that a company (especially one that has produced major websites for blue-chip clients such as Motorola and Sony) would even issue a *manifesto*, there are three aspects here of change and adaptability that are well worth noting. First, this manifesto assumes a “constant flux” both in Internet design products and in the world and environments those products are in. Products that matter, “truly integrated” products, must necessarily change along with their environments. Second, products should also adapt to “people’s different needs and contexts.” The traditional sense of design that this manifesto rails against believes in a final product, one which is designed by designers and is neither responsive to its users nor adaptable in its use. Finally, this particular Internet company wants to “practice” what it

⁵ 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.

⁶ <http://www.plumbdesign.com/manifesto>. The emphasis is ours.

preaches by encouraging the openness of design forms “internally.” Just as Weber understood the rise of Protestantism linked to the rise of capitalism, here we see an almost religious belief in constant change occurring alongside a digital reformation of capitalism.⁷

Is permanently beta as liberating and open as the manifesto above leads us to believe, or does are users and employees being duped into being human guinea pigs for the organizations that produce innovative products? Are those who sign up by the thousands to test new software doing “free” work for technology companies or do they play a critical role in designing new applications? Under a similar principle are the participants in drug experiments receiving life-saving, cutting-edge therapies or are they being exploited by pharmaceutical companies who profit from their participation in the design of new drug treatments? Under a condition of continual testing, there may be more opportunities for organizational openness and more participation in design, but, as we will see below, these positive outcomes depend on the participants’ level of organization in order to counter the instability’s social costs.

Beta Testing

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. According to Techweb’s TechEncyclopedia, a beta version is “a pre-shipping release of hardware or software that has gone through alpha test. A beta version of software 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 in the first place under real conditions. Given the complexity and ambiguous standards in the PC industry, it is impossible to duplicate the myriad of configurations that exist in the real world.”⁸ Beta testing is “real world” testing that could never be fully emulated inside software “laboratories.” On their website, CambridgeSoft, a company that makes software for life sciences research, elaborates on this “real world” aspect of the beta phase of software design by defining it as exposing a new product “to a large number of real people, real hardware, and real usage.” According to CambridgeSoft, the benefits to beta testers include:

- Getting a look at the new features before anyone else.
- The pleasure of finding unsuspected bugs.
- Making our (i.e., your!) software better as a result of detecting those bugs.
- Possibly, affecting our future direction of development through your suggestions.⁹

Beta testing is not without risks: CambridgeSoft writes on their website that they “pretty much guarantee that you will receive buggy software,” which “may crash your computer (or worse).” Being one of the first to have a new application, the “pleasure” of debugging, and

⁷ 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 Boltanski and Thevenot, *Le Nouvel Esprit du Capitalisme*.

⁸ <http://www.techweb.com/encyclopedia/>

⁹ <http://www.cambridgesoft.com/about/betatesting.cfm>

becoming involved in “your!” software development outweigh the risks for many people. More than two million people volunteered to be one of the 20,000 beta testers for the new version of Napster.¹⁰ Although Apple’s “public beta” release of OS X, their first completely new operating system since 1984, cost \$29.95, thousands downloaded it despite reports that the operating system was still quite “buggy” and little software was available for it. Beta users got to see the long-awaited operating system six months before its first commercial release, and they were thanked in advance by Apple’s CEO, Steve Jobs, who said at its unveiling, “We’re excited to have our users test drive this public beta version and provide us with their valuable feedback.”¹¹ Devoted Apple fans and the press provided invaluable buzz about the new system as they were beta testing. Beta may still have bugs, but beta testers are first on their blocks to have (and brag about having) the latest technology and can be involved in the final stages of the product design.

A typical nomenclature denotes beta versions. Beta releases are numbered below 1.0, the standard number for the first commercial release of a software application. Esther Dyson explains the connection between beta versions and commercial releases 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.¹²

Commercial products do eventually leave the beta stage in shrink-wrapped packages for consumers. The quote from Dyson, however, points to software development’s never-ending cycles of innovation, real world testing, feedback, and revision. The Internet has compressed this cycle to the extent, as suggested by Raghu Garud, Sanjay Jain, and Corey Phelps, “that it is difficult to distinguish between one product generation and the next.”¹³ They examined the releases of Netscape, which had 39 beta versions in the period 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. . . . 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.”¹⁴ Netscape risked reliability for responsiveness. Andreessen’s attitude toward building software that could rapidly integrate user response into the design—not

¹⁰ Rachel Ross. “Born-again Napster takes baby steps,” *Toronto Star*, January 11, 2002, E04.

¹¹ A press release is available at <http://www.apple.com/pr/library/2000/sep/13macosx.html>

¹² Dyson 1997, page 5, emphasis added.

¹³ “Unpacking Internet Time Innovation,” Raghu Garud, Sanjay Jain, and Corey Phelps. Unpublished manuscript.

¹⁴ Quoted in Garud et al, page 14.

simply the number of their beta versions—made the early days of Netscape permanently beta.

Encoding Responsiveness

Open source is another example of a permanently beta organizational form. In his classic essay on open source projects “The Cathedral and the Bazaar,” Eric Raymond describes the lessons he learned from the way Linus Torvald managed Linux, the operating system built by a disperse collection of volunteer programmers. Raymond points to two aspects of the philosophy of open source that challenges traditional “cathedral” models of organizing work in software engineering—“Release early and often,” and “all bugs are shallow with enough eyes.”¹⁵ At one point in the evolution of Linux, a new kernel (the most fundamental part of an operating system) was released *daily*. The reason for this was simple: as 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. This type of transparency led Raymond to question the standard philosophy of software design that strives to make products that seem flawless to the user, rather than rapidly and openly incorporating users’ concerns and “fixes” to the bugs they catch in use. Instead of a model of development based upon a single group of master builders, as cathedrals were constructed, Raymond saw the model of open source projects like a bazaar, where having an array of many different ideas allows for the quick exchange of solutions. Rather than a few building something for everyone to use, users could become actively involved in supporting a product in an ongoing evolution. 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 the efforts of many to examine software in use, identify problems, propose fixes, and watch carefully to ensure good ideas are implemented. Treat the users of a program as co-developers, Raymond argues, and they’ll act like co-developers.

Open source projects are beginning to be studied for what they may teach us about organizational forms.¹⁶ Open source projects display a responsiveness that has allowed users of products to be more directly involved in their design and in the *design-in-use* of these products. Having users emerge as the ultimate designers of products and services begins to open up the distinctions between finished products and beta products; between the designers and the users of products. The same technological advances of the Internet that have lowered distribution costs foster this sort of continual updating, encoding responsiveness into software products. 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 change organizational form.

When a user downloads a version of Mozilla, an open source and community-developed¹⁷ Internet browser, she is greeted with the following friendly message:

¹⁵ Raymond 2001. Also available online at <http://www.tuxedo.org/~esr/writings/cathedral-bazaar/>

¹⁶ See Kogut 2001 and O’Mahony, 20XX.

¹⁷ O’Mahony points out the difference between source code being open and community development of software. Several but not all open source projects are community-developed; Mozilla and Linux are two examples of community-developed open source projects.

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.¹⁸

Mozilla was the original code name for the product that became known as Netscape Navigator.¹⁹ Mozilla.org, the main coordinating group of Mozilla source code, is a non-profit organization that began under the aegis of Netscape to develop the open source code that Netscape products were based upon. That code is open and available to any group or company that wants to develop products based upon it, but, according to Mozilla.org's mission statement, for the openly developed code "to grow and mature and continue to be useful and innovative, the various changes made by disparate developers across the web must be collated, organized, and brought together as a cohesive whole." Mozilla has now grown into its own open-source program, quasi-autonomously of Netscape, and at the time of writing is releasing a beta version of an Internet browser named Mozilla.²⁰ According to their 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. Mozilla.org specifically states they are not responsible for the coding: "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."²¹ Just as those testing Mozilla are part of a community, so 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 they 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."²² Without responsiveness, the organization would become irrelevant to the community of volunteers organized around developing Mozilla, and without some form of coordination, the project would not have developed into a fully useable web browser.²³

These innovations in organizational responsiveness are not limited to open-source or beta-tested software. Increasingly, commercial Internet sites interested in creating "community"

are learning that design-in-use of these community sites, rather than a top-down design, is the best way to recruit and maintain users, as well as incorporate their demands into the site. For a different research project on work in Manhattan's "Silicon Alley," one of us (David Stark) conducted field research at a company that was developing an Internet community

¹⁸ <http://www.mozilla.org/start/>

¹⁹ Mozilla stands for the Mosaic Killer, or the application that would replace Mosaic, the first graphical interface application for the Internet.

²⁰ Available on www.mozilla.org.

²¹ See the Mozilla.org mission statement at <http://www.mozilla.org/mission.html>

²² Ibid.

²³ Mozilla can be downloaded for use at <http://www.mozilla.org/>. At the time of writing, the first "release" of Mozilla is scheduled for June 2002.

and e-commerce site.²⁴ The site, which we'll call Teensite.com, began as a content-driven online magazine oriented towards high school students. The original business model of Teensite was to create a community of youth for commercial access: as young people came onto the site for news and entertainment the editors thought was relevant to them, they would provide a targeted-demographic group for focus testing and marketing. Editors and writers from some of the Internet's top magazines provided content with the goal of creating, in the words of Teensite's executive vice president, "America's online high school newspaper."²⁵

Practice, though is always messier than the best-laid plans. In what could be called a minor revolt of the teen users, the youth users became directly involved in the design process in several ways. First, by examining the traces of what the teens actually did online, the editors found that the teens were more likely to read essays by other kids, first those by the teen reporters working with the editors. Then, a user-as-producer model of content emerged from the use patterns on the site: as more kids participated in creating the content on the site, traffic grew. As the executive vice president said, "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 they want to be in the spotlight," and Teensite tries to give them a sense of both. Referring to the teens themselves, the executive vice president, said, "*They own Teensite. We just put up the framework.*"

Their original intent was, again in the words of the executive vice president, "We know best. We create the stuff, you use it." But interactivity demanded a responsiveness on the part of Teensite to their users' demands and as well as integration of users into the process of content production. The design of the "product"—from long essays to short, chat-driven, user-written opinions and reflects—incorporated the demands of those who used it.

With the rise of interactive technologies, user emerge as content producer, resulting in what Pablo Boczkowski calls the "distributed construction" of interactive media sites like community-oriented websites, chatrooms, and email lists.²⁶ Ending the analysis there, however, ignores a growing phenomenon of the Internet era. After all, do we consider telephone users as the producers of content for telephone companies? While a letter to the editor of a newspaper would, of course, not be considered "unpaid content," any publication that refused a forum for response would rightly outrage its readers.

Practicing communities—towards permanently beta organization.

In the examples above, the difference between the user merely providing content or services for a product and being actively engaged in design of a product depends on the level of organization of users. Organized users determine whether or not permanently beta organizational forms are beneficial to those who are a part of them. Software beta testers sign up in part to get a say in new software design. Teensite kids, understanding the

²⁴ For more on this research project area in general, see Girard and Stark.

²⁵ All quotes about Teensite are directly cited from field notes and from interviews with Teensite management.

²⁶ Boczkowski, p. TK.

importance of their participation to the website, demanded a more active voice in the design of the site. As Internet companies track what their users do online, users form a virtual focus group on which links are clicked, which content is read for how long, and which other online sites are visited. Where the user has power as a designer, she has an active voice in the structure of the product. There exists, unfortunately, a wide gulf between the experience of participating in the design of something and needlessly being subjected to instability—or being used for merely being a user.

In anthropology “community of practice” describes a group of people who share common goals, understand themselves as part of a community and a “sustained pursuit of a shared enterprise.”²⁷ Communities of practice can refer to a particular industry or a group with shared knowledge working within a particular company, such as people in a firm’s sales department. In many traditional settings, as we’ve seen above, users might not have the same goals as those who design the product, and beta testers may not consider themselves part of the same community as software engineers (and vice versa!). Switching the focus away from community towards practices shows how disparate groups can be united. We call these practicing communities, which link organized users with professional expertise in order to inform the design process. As master musician and novice alike need practice, so users and so-called experts negotiate knowledge practices—Mozilla users are on equal footing with the engineers at Netscape if they fix the technological problems of bugs. In a practicing community, an organized group of users become acknowledged as experts in how products are used, realizing their power to influence design. From product design to organizational design, permanently beta settings have this potential to be democratic.

So far we’ve mentioned relatively innocuous forms of testing, but testing new products can be a literally a matter of life and death. While testing new software and providing a new design for the Internet might seem inconveniences at worse, there are those who demand to be tested upon. Such was the case for “People with AIDS” or PWAs, as they called themselves in the late-1980s. At the Fifth International Conference on AIDS in June 1989, a “ragtag group of 300 protestors” stormed the conference hall during the opening ceremony, taking over the stage with their “Silence=Death” posters and thunderous chants. Tim McCaskell, an AIDS activist, grabbed the microphone and “officially” opened the conference on behalf of people with AIDS from Canada and around the world.”²⁸ More than just a seat at the world’s most important research conference on AIDS, the protestors demanded to be actively involved in the design of drug trials.²⁹ Through these demands, people with AIDS and HIV not only obtained expanded access to experimental treatments, but forced change in the mammoth bureaucracy of the Food and Drug Administration and the way it approves drugs. Including AIDS patients in the planning and design of research,

²⁷ Wenger, *Communities of Practice*, 1999. The term was first coined by Lave and Wenger in *Situated Learning*, 1991.

²⁸ Reported by Ron Goldberg in *Poz*, July 1998. Reprinted on the ACT-UP/NY website, <http://www.actupny.org/documents/montreal.html>.

²⁹ ACT-UP/New York, one of the many groups involved in the protest, released a document stating “12 Principles for a New AIDS Drug Testing System,” in which they demanded, “People with AIDS, HIV, and their advocates must participate in designing and executing drug trials.” See a copy of the press release on the archive of AIDS Treatment News: <http://www.aids.org/immunet/atn.nsf/page/a-082-04>.

according to one AIDS researcher, was not only good for the patients, but also helped to improve the integrity of the drug trials.³⁰ Creating what was called a “parallel track” in AIDS drug experiments provided access to experimental treatments to people who were too sick or otherwise not eligible for participation in controlled experiments. Community-based trials took research out of the university laboratory and into the areas where poor and isolated individuals could be included. In effect, AIDS patients, too, refused to be human guinea pigs, demanding an active role in designing research in exchange for the use of their bodies for clinical trials.

No one politely invited AIDS activists to reform years of entrenched medical practices. Through organizing themselves as users of medical treatments, the activists challenged the stigma associated with AIDS and HIV. They also challenged the very notion of expert knowledge as they became “lay experts.”³¹ As “users,” to return to our software metaphor, they challenged the idea that the “designers,” or the medical establishment, knew best. Through organizing themselves they bridged a gap between lay and expert knowledge. By learning AIDS researchers’ language³² activists literally gained a seat at the table of research, joining professional in a practicing community. Bringing users into the design of treatments transformed organizations (the Food and Drug Administration and AIDS organizations), activists and patients, and the knowledge about the disease. Vololona Rabeharisoa and Michel Callon in a similar study of mutual learning and reflexive organization in the French Muscular Dystrophy Association write the “knowledge produced by laboratories and doctors is specific and irreplaceable, but it is nurtured and deployed by the actions of organized patients, and irrigated by the flow of knowledge and questions they formulate.”³³

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” which allows for the organization of rivalrous logics, values and organizational principles.³⁴ Charles Sabel and Michael Dorf have referred to the process of simultaneous engineering, a concurrent design process by which separate teams develop different proposals for the final design.³⁵ Permanently Beta is, in part, a form of simultaneous and collaborative design and engineering that brings the user into the process. Software beta testers want a first look at new versions of software while 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. Teensite users want the spotlight on their stories, while the Teensite editors need teens to visit the site to support e-commerce and marketing functions that make money. People with AIDS wanted expanded access to drug trials, while drug companies needed them for testing. Each of these examples points to different sets of values along a

³⁰ See Alvin Novick, “Noncompliance in Clinical Trials: I. Subjects” in *AIDS & Public Policy Journal* 5:2, 94-96. 1990. For more on the particular case of combination therapy trials, see Steven Epstein, “Activism, Drug Regulation, and the Politics of Therapeutic Evaluation in the AIDS era,” *Social Studies of Science* 27 (1997), 691-726.

³¹ For more on lay experts, see Steve Epstein, “The Construction of Lay Expertise.”

³² Ibid.

³³ Rabeharisoa and Callon, p. TK.

³⁴ Girard and Stark.

³⁵ Sabel and Dorf, p. TK

divide, and in each those who use the software, read the content or test the drugs gained voice in the design process. These aren't examples of competing companies vying for a contract through simultaneous engineering codes or subcontractors working throughout collaborative organizational principles, but the various actors in these settings hold disparate values and principles that need to be negotiated in a similar manner.

Through these testing forms, through experimentation, these values get negotiated so that they are in part incorporated in the design process and the products themselves. Permanently beta forms produce products that are in themselves negotiations, like the multiple versions of beta software and its subsequent multiple release versions and patches. The design process is considered ongoing rather than having a final endpoint, as each of those releases offers the opportunity to go back and incorporate options previously left out. Bringing users into the design and testing involves a genuine interaction with the user, not an attitude of "We know best," to recall Teensite's vice president. Users can be involved when a design is completed in use, not in the laboratory or in the studio. Thus, permanently beta forms are like architected designs that are left partially open to the interpretation of the engineered execution. Permanently beta forms necessarily must leave things out to be completed by the user, as poetry acquires meaning through its reading, not merely at the hand of its writer. Practicing communities are enabled in permanently beta situations to link lay knowledge to expertise, constant change to responsiveness, users to producers, patients to researchers, buyers to manufacturers.

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

The Internet allows us to see Permanently Beta in action, and to become accustomed to its rhythms. The Web is quite an unstable place: Websites die, disappear and are modified in a flash. Instability online is such the norm that it probably seems silly to even point it out, but when was the last time you expected a Web page to be the same as the last time you visited? More importantly, the Internet enables users to more easily be a part of that continual updating. Websites such as Plastic.com and Slashdot.com recycle "the web in real time" (as Plastic says on their site), manifesting a permanently beta approach to news. Users of these services continually update the news, not with new reporting, but by catching the "bugs" in published media reports, drawing the connections between stories, and appending their own

commentary and analysis onto a story. 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 non-reporters.

Being permanently beta presents new tensions and as well as promise. Within this state, we and the products and structures around us are being tested, and the responsibility for adaptation rests largely upon us. The “changing scripts” of continually changing workplaces, as Kunda and Van Mannen term them, force adaptation by those who work in these environments.³⁶ Permanently beta organizations have destabilized bureaucratic forms, for the challenges of responsiveness are too great for institutional routinization to emerge. Heterarchies—flat organizations with distributed accountability, decentralized decision-making, and multiple, often competing, goals—emerge. Thus, 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, as John Child and Rita Gunther McGrath have criticized.³⁷ Continual reconfiguration in heterarchical organizations can be exhausting, especially in work environments organized around projects that exert extraordinary time pressures and a demand a fast pace of action, as Grabher found in his study of British advertising agencies.³⁸

Users of permanently beta products may also find the experience frustrating. Products are not final, clean, end-versions but destabilized, constantly changing products. Any user of computer software understands the continuous updating of applications; those who have experienced the “bugginess” of new versions understands all too well the downsides of continual change. Having to reconfigure ever changing products is part of what Tiziana Terranova sees as “extraction of value out of continuous, updateable work” that exploits the “free labor” of users in a digital economy.³⁹

The “New Economy,” too, has shown us how quickly economic experiments 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 point 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 Website is a Code 404, ‘File Not Found.’”⁴⁰ While the innovative start-ups of Silicon Alley and Silicon Valley pushed organizational logics to their brink, many of those involved felt just that: *involved*. One dot-commer in an interview for our field research on Silicon Alley described feeling involved in creating “the freest medium around.”⁴¹ For many of them, 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

³⁶ Kunda and Van Mannen, p TK.

³⁷ Child and McGrath, p. TK.

³⁸ Grabher, 2002, p 254.

³⁹ Terranova, p. 48.

⁴⁰ Girard and Stark.

⁴¹ Interview transcript with editor of an online division of a major publishing house, 1997.

economic boom more jobs were created in low-end service work than in high-end knowledge work. As silicon mavericks were testing their sites, their organizations and themselves, some people had to adapt alone within an economy whose rules had changed.⁴²

We can't stop technological change—nor would we want to—but experimentation and testing of new technologies might point to the way that new economic and organizational forms can be tested, negotiated, and ultimately more inclusive. If the permanently beta ethic is to influence positively economic organization, then change must be accompanied by the difficult work of organizing participation in that change.

⁴² See Vicki Smith's *Crossing the Great Divide*, for more on worker adaptation within the new economy.

References

- Boczkowski, Pablo. 2001. *Affording flexibility: Transforming information practices in online newspapers*. Unpublished doctoral dissertation. Cornell University, Ithaca, NY.
- Boltanski, Luc and Laurent Thevenot. 1999. *Le Nouvel Esprit du Capitalisme*. Paris:Gallimard.
- Castells, Manuel. 2000. *The rise of the network society* Oxford: Blackwell Publishers. 2nd ed.
- Child, John and Rita Gunther McGrath. 2001. "Organizations Unfettered: Organizational form in an information-intensive economy" *Academy of Management Journal*, 44:1135–1148.
- Dyson, Esther. 1997. *Release 2.0: A Design for Living in the Digital Age*. New York: Broadway Books.
- Epstein, Steven. 1997. "Activism, Drug Regulation, and the Politics of Therapeutic Evaluation in the AIDS era," *Social Studies of Science* 27:691–726.
- _____. 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.
- Girard, Monique and David Stark. "Distributing Intelligence and Organizing Diversity in New Media Projects" *Environment and Planning A* (forthcoming.)
- Grabher, Gernot. 2002. "The Project Ecology of Advertising: Tasks, Talents and Teams." *Regional Studies* 36:245–262.
- Kogut, Bruce and Anca Metiu. 2001. "Open Source Software Development and Distributed Innovation," *Oxford Review of Economic Policy* , 17:TK–TK.
- Kunda, Gideon and John Van Mannen. 1999. "Changing Scripts at Work: Managers and Professionals," *Annals of the American Academy of Political and Social Science*, 561 (January).
- Sach, Warren. "Some Theory for Citizen-Centered Software Design" unpublished manuscript.
- Terranova, Tiziana. 2000. "Free Labor: Producing Culture for the Digital Economy," *Social Text* 18:2 (Summer), 33-58.
- Thrift, Nigel. 2001. "Software Writing Cities," address to the Taub Urban Research Center, New York University, February 26.
- _____. 2000. "Performing Cultures in the new economy. *Annals of the Association of American Geography*, 90:4, 674-92.