OII OIIOI
Center on Organizational Innovation

Permanently Beta:
Responsive Organization in the Internet Era
September 2002

Gina Neff
Columbia University
ginasue@panix.com

and

David Stark
Columbia University and
Santa Fe Institute
dcs36@columbia.edu

Gina Neff is a PhD candidate in the Department of Sociology at Columbia University where she is a Research Associate in the Center on Organizational Innovation. David Stark is Arnold A. Saltzman Professor of Sociology and International Affairs at Columbia University and an External Faculty member of the Santa Fe Institute. He is currently a Visiting Scholar at the Russell Sage Foundation in New York City.

Abstract

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 offline. A cycle of testing, feedback, and innovation facilitates ongoing negotiations around making products and around organizing that production. We call the organizational state of flux that emerges from these negotiations Permanently Beta. Beta testing, open source software, and interactive communities manifest aspects of permanently beta organization. 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.

Introduction

How 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 focuses on influence of the *process* of technological change in the Internet era on the organization of economic life. In addition to technological innovations, emerging organizational innovations need to be considered in any examination of the social effects of the Internet. While skeptics of the Internet's influence may point to failed dot-coms or lackluster data on 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, perhaps not through direct influence upon the marketplace, but through encoding these new routines and new ways of doing things into daily practice.[1]

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 in this era may, as some have argued, '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 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. Interactivity freed information somewhat in those realms driven by a sense of public or collective ownership, such as in e-government initiatives, non-profit and social movement organizations, and community-developed projects such as Linux. Openness, it seems, is not yet a market value.

However, organizational forms and technology are influencing one another in unexpected ways, co-evolving toward more open structures that may, as will be argued below, 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 applications 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. The push towards more open structures is no

exception. At the same time computer protocols, as Alex Galloway writes, are 'how control exists after decentralization,'[2] and are deeply influenced by the values of the designers, organizations, and societies that create those protocols. The new economy is dead, but left standing are these tensions between the market values of technological innovation and the community values of openness enabled by the Internet.

One of the values of the contemporary economy embedded into the Internet and computer technologies is what Lev Manovich has characterized a s 'variability.' Unlike mass-produced objects, programmed products are never stable, can have a potentially infinite number of versions, and can be almost immediately customized for a particular user (2001: 36). Software is 'patchy': Versions change, systems evolve, applications die.[3] 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 because of the values of a post-industrial economy that favors just-in-time and customized 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. When the challenges of responsiveness are too great for institutional routinization, organizations' bureaucratic structures are destabilized. Heterarchies—flatter organizational structures with distributed accountability, decentralized decision making, and multiple, often competing, goals—emerge (Stark, 1999). What this chapter argues is 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 of organizations. The influence of design—whether the design of products, technology or services—and organizational form upon 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 between 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 upon participants' level of organization. This is true for 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 Gideon Kunda and John Van Mannen argue, force adaptation by those who work in permanently beta organizations (1999). Continual reconfiguration in heterarchial 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 Gernot Grabher found in his study of British advertising agencies (2002). 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 and McGrath, 2001).

But this permanently beta ethic of continual change can be thought of, borrowing from Max Weber, as a key influence upon the spirit of the information economy. Consider the following '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.[4]

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 of change and adaptability to note. First, this manifesto assumes ongoing instability both in Internet design and in the larger environment around that design. Second, this manifesto challenges a traditional sense of design that assumes products are final and are neither responsive to their users nor adaptable in their 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 that the rise of Protestantism was 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.[5]

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 alpha test.' It is 'supposed to be very close to the final product, but, in practice, it is . . . 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.[6] 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.
- 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.[7]

Beta testing is not without risks: CambridgeSoft says it 'pretty much guarantee[s] 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 becoming involved in
'your!' software development, as well as the possibility of free or cheaper software, for many
outweigh these risks. More than two million people volunteered to be one of the 20,000 beta
testers for the new version of Napster.[8] 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 it was still quite 'buggy' and that little compatible software was available.
Beta users saw the long-awaited new operating system six months before its first commercial
release, and Steve Jobs, Apple's CEO, 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.'[9] 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. Esther Dyson explains 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.[10] (Dyson, 1997: 5)

Commercial products do eventually leave the beta stage in 'shrink-wrap' or as
purchasable product 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 which is much shorter than that in manufacturing design, but longer than that of
Internet site development. The Internet is compressing this cycle to the extent, as suggested
by Raghu Garud and his co-authors, 'that it is difficult to distinguish between one product
generation and the next' (Garud, Jain, and Phelps, n.d.: 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 Andreesen, 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' (Garud, Jain, and Phelps, n.d.:

14). Andreesen'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 nor 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 testing. While 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 in 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 to incorporate user response, but responsiveness can be encoded into products and organizations in other ways.

Open source projects are one example in which 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 heterarchial modes of organizing. In a now classic essay on open source, 'The Cathedral and the Bazaar,' Eric Raymond describes Linus Torvald's management of the development of the Linux operating system. Raymond points to two key aspects of open source philosophy that challenge traditional models of software engineering—'Release early and often,' and 'all bugs are shallow with enough eyes' (Raymond, 1999). At one point in the evolution of Linux, a new kernel (the most fundamental part of an operating system) was released *daily.* 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. 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 that good ideas are implemented. Treat the users of a program as co-developers, Raymond argues, and they'll act like co-developers. 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[11] Internet browser, she 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.[12]

Mozilla was the original code name for the product that became known as Netscape Navigator.[13] 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 source code that its products were based on. Mozilla has now grown into its own open source program, quasi-autonomously of Netscape, and at the time of writing is releasing beta versions of an Internet browser.[14] 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. Yet Mozilla states, '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.'[15] 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 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.'[16] 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 useable web browser.

Open source projects display a responsiveness that has allowed users of products to be more directly involved in the design and the *design-in-use* of these products. Open source organizational forms challenge the traditional notion of boundaries between for-profit and non-profit organizations, as they often have more in common with social movements than traditional formal organizations. Siobhan O'Mahony argues 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 (2002: viii). Open source projects combine aspects of community and commerce into a new hybrid of organizational form.

Open source projects are not necessarily permanently beta, but they often provide a good example of what Eric von Hippel calls 'user innovation communities' (2001: 82). These communities are not just in software and hardware production. Von Hippel describes the user innovations in high performance windsurfing. Windsurfers testing out innovations on the open beach shared them with each other and ultimately with equipment manufacturers (2001: 85). The Internet can be an open beach of sorts, with other users able to 'see' one other and user innovations online. The emergence of 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 in which there is 'product development without manufacturers' (von Hippel, 2001: 82), but it also can occurs in settings where user feedback is more fully integrated into the design process. By shortening the cycle of feedback and reengineering, by creating reputational benefits for users who innovate, and by fostering communities around innovation, the Internet can strengthen these feedback loops and reinforce the process of continual updating.[17] 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 other settings. Users 'consume, modify, domesticate, design, reconfigure and resist technologies' and through this process shape and are shaped by those technologies (Oudshoorn and Pinch, forthcoming). Most often, users and technology interact in private settings, unlike the interaction in the more open, visible, and public interactions with technology that occur online (Boczkowski 2001). Permanently beta organizations emerge when the institutional barriers to user involvement in the design process are overcome. Open source communities are one such example, 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 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.[18] Teensite.com (a pseudonym) began as a content-driven online magazine trying to become 'America's online high school newspaper.'[19] In its original business model, Teensite created a youth community for commercial access: as young people came onto the site for the news and entertainment that editors and writers thought relevant, the teens would be a targeted demographic group for marketing and focus research.

The teen users staged a minor revolt, getting involved in the design process in several ways. First, by examining the use patterns of the teens online, 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. Teensite'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 they want to be in the spotlight,' and Teensite tries to give them a sense of both. Referring to the teens, the executive vice president said, '*They* own Teensite. We just put up the framework.' Moreover, the teens themselves began making demands of the editors to be allowed greater interactivity on the site, more control over the commercial access to their information, and the ability to refuse to be 'marketing guinea pigs' for the company. Originally management thought, according to Teensite's executive vice president, 'We know best. We create the stuff, you use it.' But interactivity demanded a responsiveness on the part of Teensite 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.

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

connect them. Users of almost any product can find each other to share ideas, to talk about product use, or organize to make demands of the manufacturer, demands that are generally 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, although as Steve Epstein (1995, 1997) has written about the AIDS community and its clashes with pharmaceutical companies, these differences in values can sometimes produce negotiations toward a mutual goal. AIDS activists wanted wider access to healthcare, including experimental new drug treatments; companies wanted to design and market new for-profit drug treatments. While 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, 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). We call these '*practicing communities.*' Practicing communities link organized users with professional expertise in order 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 use this power to influence design.[20] 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' which allows for the organization of rivalrous logics, values, and organizational principles (Girard and Stark, 2002: 3). Charles Sabel and Michael Dorf (1998) have 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 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. Each of these examples points to different sets of values. In each, those who use the software, read the content, or test for bugs gained some 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 must be negotiated in a similar manner.

Through these testing forms, through experimentation, and through negotiation, values are incorporated in the design of the products themselves. Permanently beta forms produce products that are in themselves negotiations, like the multiple versions of software with its subsequent multiple betas, release versions, and patches. The design process is considered ongoing rather than having a final endpoint, as each of those releases offers an

opportunity to go back and incorporate options or fixes previously left out. Bringing users into the design and testing in permanently beta settings involves a genuine interaction with the user, not an attitude of 'We know best,' to recall Teensite's vice president. Figure 1 summarizes the differences between permanently beta and traditional approaches to design. Permanently beta forms necessarily leave things out to be completed by the user, 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, buyers to manufacturers.

Figure 1: Permanently Beta Design versus Traditional Design

	Permanently Beta Design Process	**Traditional Design Process**
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, at least at this juncture in its history, is quite an unstable place: Websites die, disappear, and are modified in a flash. More importantly, the Internet facilitates users being a part of that continual updating. Websites 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 by catching the 'bugs' in published media reports, drawing connections between stories, and appending their own commentary and analysis to a circulating 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.

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 (2000: 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 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"' (2002: 26). While 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 he was creating 'the freest medium around.'[21] 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 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, some people had to adapt alone within an economy where the rules were changing.[22]

If, as it 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 the future of code. Thus 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 like codes have values, and as a society we can either actively engage in the negotiations around these values or chose imprudently to ignore them.

There are no guarantees 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 co-existing values (Boczkowski, 2001; Galison, 1997; Stark, 1999). Nor are innovation and creativity in 'practice' necessarily integrated into the organizational process (Brown and Duguid, 2001: 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, negotiated, and may prove ultimately to be more inclusive. While the lack of settlement on a single path in permanently beta settings can be confusing, frustrating or worse, this structure affords an opening to influence encoded values—to incorporate our values—into organizations and technology.

References

Boczkowski, Pablo. (2001) *Affording flexibility: Transforming information practices in online newspapers.* Unpublished doctoral dissertation. Cornell University, Ithaca, NY.

Boltanski, Luc and Eve Chiapello. (1999) *Le Nouvel Esprit du Capitalisme.* Paris: Éditions Gallimard.

Brown, John Seely and Paul Duguid. (2001) 'Creativity versus structure: A useful tension,' *MIT Sloan Management Review,* 42(4): 93-95.

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.

Galison, Peter. (1997) *Image and Logic: A Material Culture of Microphysics.* Chicago: University of Chicago Press.

Galloway, Alex. n.d. 'Protocol: Or, How Control Exists After Decentralization,' unpublished manuscript, Duke University.

Garud, Raghu ,Sanjay Jain, and Corey Phelps. n.d. 'Unpacking Internet Time Innovation,' Unpublished manuscript.

Girard, Monique and David Stark. (2002) 'Distributing Intelligence and Organizing Diversity in New Media Projects' *Environment and Planning A* (forthcoming.) Also available online at http://www.coi.columbia.edu/workingpapers.html.

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(2): 248-264.

Kollock, Peter. (1999) 'The Economies of Online Cooperation: Gifts and Public Goods in Cyberspace.' In M. A. Smith & P. Kollock (Eds), *Communities in Cyberspace.* London: Routledge.

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): 64-80.

Manovich, Lev. (2001) *The Language of New Media.* Cambridge, MA: MIT Press

O'Mahony, Siobhan. (2002) 'The Emergence of a New Commercial Actor: Community Managed Software Projects', Unpublished Doctoral Dissertation, Stanford University.

Oudshoorn, Nelly and Trevor Pinch. (Forthcoming) "Introduction" in Nelly Oudshoorn and Trevor Pinch (eds.), *How Users Matter: The Co-Construction of Users and Technologies* Cambridge, MA: MIT Press.

Raymond, Eric. (1999) *The Cathedral and the Bazaar: Musings on Linux and Open Source from an Accidental Revolutionary.* Sebastapol, CA: O'Reilly and Associates.

Sach, Warren. n.d. 'Some Theory for Citizen-Centered Software Design' unpublished manuscript. University of California, Berkeley.

Sabel, Charles F., and Michael C. Dorf. (1998) "A Constitution of Democratic Experimentalism" *Columbia Law Review* 98(2): 267-529.

Smith, Vicki. (2001) *Crossing the Great Divide.* Ithaca, NY: Cornell University Press.

Stark, David. (1999) 'Heterarchy: Distributing Intelligence and Organizing Diversity', in John Clippinger (ed.), *The Biology of Business: Decoding the Natural Laws of Enterprise.* San Francisco: Jossey-Bass. pp 153-179.

Terranova, Tiziana. (2000) 'Free Labor: Producing Culture for the Digital Economy,' *Social Text* 18(2): 33-58.

Thrift, Nigel. (2001a) 'Software Writing Cities,' address to the Taub Urban Research Center, New York University, February 26.

    . (2001b) ''It's the Romance Not the Finance That Makes the Business Worth Pursuing.' Disclosing a New Market Culture.' *Economy & Society* 30 (4):412-432.

von Hippel, Eric. (2001) 'Innovation by User Communities: Learning From Open-Source Software,' *Sloan Management Review* 42(4): 82–86.

---

1 See Thrift (2001a) for more on software code's passion for inscription.

2 Galloway's dissertation is one of the first we know of that uses 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 http://www.plumbdesign.com/manifesto. The emphasis is ours.

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 Manuel Castells's triology, *The Rise of the Network Society* as well as to Luc Boltanski and Eve Chiapello's book, *Le Nouvel Espirit du Capitalisme.*

6 http://www.techweb.com/encyclopedia/

7 http://www.cambridgesoft.com/about/betatesting.cfm

8 Rachel Ross. 'Born-again Napster takes baby steps,' Toronto Star, 11 January 2002, E04.

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

10 Emphasis added. Her Release 2.1, the paperback version of the book, came out in October 1998.

11 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.

12 http://www.mozilla.org/start/

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

14 Mozilla can be downloaded for use at http://www.mozilla.org/. Mozilla's first 'release' was in June 2002.

15 See the Mozilla.org mission statement at http://www.mozilla.org/mission.html

16 Ibid.

17 Peter Kollock (1999) in his now classic essay on online cooperation identifies these mechanisms as supporting exchange online.

18 For more on this research project area in general, see Girard and Stark (2002).

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

20 Changes in product design can influence organizational structure, as Epstein noted in the case of AIDS drug approval process changing the organization of the Food & Drug Administration.

21 Interview transcript for Neff's dissertation, "Risk Relations" with the editor of an online division of a major publishing house, 1997.

22 See Smith (2001) for more on worker adaptation within the new economy, and Thrift (2001b) for more on the rhetorical strategies of finance that enabled the technological boom of the late 1990's to even occur.