INCOMPLETE BY DESIGN AND DESIGNING FOR INCOMPLETENESS[*]

Raghu Garud
Pennsylvania State University

Sanjay Jain
University of Wisconsin at Madison

Philipp Tuertscher
University of St. Gallen

February 22, 2007

[*] Draft of paper written for consideration in the special issue on *Organization studies as a science of design* being edited by Richard Boland, Marianne Jelinek and Georges Romme. Some of the ideas in this paper are based on prior discussions with Arun Kumaraswamy. We thank him. We also thank the three reviewers of this paper and the editors of this special issue for their valuable feedback.

INCOMPLETE BY DESIGN AND DESIGNING FOR INCOMPLETENESS

Abstract

The traditional scientific approach to design extols the virtues of completeness. However, in environments characterized by continual change, there are challenges in adopting such an approach. We examine Linux and Wikipedia as two exemplary cases to explore the nature of design in such a protean world. Our observations highlight a pragmatic approach to design in which incompleteness is harnessed in a generative manner. This suggests a change in the meaning of the word "design" itself – from one that separates the process of design from its outcome, to one that considers design as both the medium and outcome of action.

INCOMPLETE BY DESIGN AND DESIGNING FOR INCOMPLETENESS

Historically, much of the discourse on design has extolled the virtues of completeness. Completeness allows for the pre-specification of a problem, the identification of pre-existing alternatives and the choice of the most optimal solution. Such a scientific approach to design pervades much of management thinking, education and research (Romme 2003: 24).[1]

For instance, this approach is evident in the design of traditional organizations at the turn of the 19th century. Organizations enhanced the efficiency of their operations by systematically applying principles of scientific management to discover "the one best way" to organize (Kanigel 1997). Interchangeable parts, division of labor, routinization – each of these were features of an organizational design capable of mass producing "any color car as long as it was black" (Ford and Crowther 1922: 72).

For such an approach to work, however, there needs to be a clear and stable boundary between the entity being designed and the context for which it is being designed. Such a boundary makes it possible to fix the purpose of a design based on a stable set of user preferences and performance expectations. Clear boundaries, stable preferences and fixed goals – these form the cornerstones of the scientific approach to design as articulated by Simon (1996).

But how does such an approach to design hold up in environments characterized by continual change? What if there are multiple designers, each with their own representation of the problem? What if users of a design are also its designers? To further complicate matters, what if the process of discovering new and potentially better states only takes place through a process of participation, and the unfolding of the process itself changes the problem?

[1] In this paper, we make a distinction between a "scientific approach" to design that applies analytic thinking to address clearly defined problems to discovery an optimal solution (following the natural sciences as a role model) and a "pragmatic approach" that applies synthetic thinking to address ill-structured problems. Others have used the terms "science" vs. "design" (Romme 2003) and "decision science" vs. "design science" (Boland and Collopy 2004) to make such a distinction.

This is the new frontier in which we find ourselves. There is no clear separation between the inside and the outside, text and context. Rather, there is only an evolving and emerging network of associations (Barry and Rerup 2006: 267). Problems are ill defined, preferences are fluid and solutions emerge in action. In such situations, an emphasis on completeness is likely to result in the creation of designs that foreclose future options.

It is useful to consider the dual meaning of the word “design” within this context. As a verb, "to design" refers to the process of developing a plan for a product, structure or component. As a noun, "a design" is used to connote the outcome of the process.[2] In traditional settings, these two meanings of design have been separated from one another. One would engage in a process of design (the verb) so as to emerge with a design (the noun) for a specific context. In contemporary settings, however, designs are more appropriately viewed as being simultaneously noun and verb, with every outcome marking the beginning of a new process.[3] Put differently, designs are like dynamic jigsaw puzzles in which multiple actors assemble pieces within templates that change as a result of the actors’ engagement.

It is this proposition that we develop in the paper. We suggest that, rather than a scientific approach that tends to separate the two meanings of design, we must embrace a pragmatic approach to design that simultaneously embraces both process and outcome. Given this dual connotation, designs, by definition, have to deal with incompleteness. However, rather than pose a threat, incompleteness acts as a trigger for action. Even as actors try and complete what has been left incomplete, they generate new problems as well as new possibilities that continually

[2] Dewey’s (1934) approach to pragmatism informed our understanding of design as noun and verb, an understanding reinforced by the entry on design in Wikipedia (http://en.wikipedia.org/wiki/Design)

[3] This is clearly derived from a structurational perspective (Giddens 1979) where structure is both medium and outcome of action.

drive the design. In this way, incompleteness is both a cause and consequence of the dynamics of organizing in contemporary environments.

We begin by providing a brief overview of the scientific approach to design, and then highlight the challenges that one confronts in applying this within contemporary environments characterized by continual change. To empirically locate our observations, we examine in detail two exemplary designs that appear to be always in-the-making – the Linux operating system and the Wikipedia online encyclopedia. Our observations highlight a pragmatic approach to design. Rather than one group designing for another's consumption, we find that designs emerge through situated use as actors co-theorize across multiple settings, and, in the process, create new options. These dynamics produce self-perpetuating processes that further drive continual change.

TO DESIGN OR NOT TO DESIGN? THAT IS THE QUESTION

In his book, "The Sciences of the Artificial" (1996), Simon suggested that the design of artificial systems – meaning man-made as opposed to natural – is contingent upon the goals of the designer and the purposes for which the system is designed. A key initial task for designers involves the specification of system boundaries. As Simon pointed out, "An artifact can be thought of as a meeting point – an 'interface' in today's terms – between an 'inner' environment, the substance and organization of the artifact itself, and an 'outer' environment, the surroundings in which it operates. If the inner environment is appropriate to the outer environment, or vice versa, the artifact will serve its intended purposes" (Simon 1996: 6).[4]

[4] Specifying the boundaries is by no means a trivial task. Alexander's insightful analysis (1964) of the redesign of a tea kettle illustrates this point. At first blush, such a redesign seems simple given that the kettle is a clearly defined object and the boundaries between the kettle and its environment are obvious. However, Alexander goes on to demonstrate that by changing the nature of the problem to be addressed – for instance, by asking if it is the method of heating kettles rather than the kettle itself that needs redesigning – the solution obtained can change drastically. In reframing the question, the kettle becomes part of the outer environment and the stove becomes the inner environment.

Once an interface has been specified and the problem has been defined in terms of its context, form and goals, it is possible to proceed using the organizing principles to be found in the natural world. For instance, Simon (1962) offered the principle of ‘near decomposability’ as a concept possessing clear evolutionary advantages for both natural and artificial systems (see also Langlois 2002). Decomposability refers to the partitioning of a system in such a way that the interactions of elements within a subassembly are greater than the interactions between them. Such decomposition reduces the complexities confronted by boundedly rational human beings in their efforts to design artifacts.

To illustrate this point, Simon offered a parable of two watchmakers, Tempus and Hora. Tempus organized his work in a manner that if he had "one (watch) partly assembled and had to put it down – to answer the phone, say – it immediately fell to pieces and had to be reassembled from the elements." Consequently, every time Tempus was interrupted and forced to set aside his work, the entire unfinished assembly fell to pieces. In contrast, Hora first built stable subassemblies that he then put together in a hierarchic fashion into a larger stable assembly. Thus, when Hora was interrupted, only the last unfinished subassembly fell apart, preserving most of his earlier work.

According to Simon, the differential cost of incompleteness that the watchmakers confront can be explained by the interplay between their short and long-term memories. When individuals address complex problems, transactions are carried out in their short-term memory. Given the limits to short term memory, any interruption to a task can exact a toll. This is because any intermediate outcome that might have been accomplished before the interruption is lost. While it is possible to store intermediary outcomes in long-term memory, this too exacts a toll as the transfer between long and short term memory often requires considerable effort.

However, the toll posed by interruptions can be reduced if individuals have “templates” of intermediate results stored in their long-term memory. With these templates, the effort required to transfer intermediate results between the long and short-term memory is much lower than if they were not used. It is precisely such stable, pre-specified templates that Hora employs. Specifically, he assembles the myriad parts into sub-assemblies that fit within an overall hierarchy to make up the watch. He does not need to know the workings of all the watch parts simultaneously because he has decomposed his overall task. Rather, he only needs to know what is required to complete the specific sub-assembly that he is working on. Thus, when Hora is interrupted, only the last incomplete sub-assembly falls apart, preserving most of his earlier work and thereby reducing the cost of incompleteness that he incurs.

Scientific design in practice

The concept of decomposability that emerges from this parable has generated a lot of attention from scholars who study modularity (Baldwin and Clark 2000). Specifically, complex systems can be decomposed in a way similar to what Simon suggested – in other words, into "modules" (Langlois and Robertson 1992). Each module only interacts with another through standardized interfaces. As a result, each module becomes a “black-box” (Rosenberg 1982), possessing the detail required for its functioning, but hiding such detail from other interdependent modules (Parnas 1972).

An implicit assumption in this literature is that the overall system architecture needs to be completely specified *a priori* (Baldwin and Clark 2000). In terms of Simon’s parable, this means that both Tempus and Hora work with designs that have clearly defined boundaries and pre-set user preferences (accurate time keeping, for example). In such cases, the key design decisions revolve around issues such as detailing the elements that comprise the architecture, establishing

stable interface specifications to ensure smooth functioning between modules and "black-boxing" the modules to mask their complexity.

These decisions are integral to a design approach that values completeness. The form and function of a system must be clearly specified. Only with such a complete representation is it possible to identify clear and stable boundaries between self-contained components that mask much of the complexity. These facets are evident in much of the engineering literature with its focus on detailed definition of requirements (Petroski 1996).

In organizational studies, such a scientific approach to design is most evident in the work of Fredrick Taylor and his colleagues (Kanigel 1997). Once the purpose of a corporation was fixed – for example, to produce widgets with certain predetermined features in as efficient a manner as possible – this approach could be used to identify the "one best way" to organize. The design process was driven by the need to completely understand and optimize all the cause and effect relationships that could influence the outcome of an organization's activities.[5] Theorizing was done by specific individuals such as industrial engineers but not by those engaged in ongoing operations whose job it was to "do and not to think". Decomposability was manifest in division of labor and the presence of a hierarchy, as well as the use of interchangeable parts that were tied together through stable interface specifications within an overall organizational architecture. Costs associated with incompleteness that arose from interruptions were to be minimized at all costs by keeping the assembly line running even if there were defects in the system (Kanigel 1997). The result was the design of the quintessential lean and efficient manufacturing process that could mass-produce goods for pre-set user preferences.

[5] See Boland & Collopy (2004) for the use of linear programming methods in inventory control, which is based on the maximization of a clearly specified objective function given a set of pre-specified constraints.

Pragmatic approach to design

A scientific approach to design – one that requires complete representation of the problem and identifies the optimal solution – is based on the assumption that the environment is stable. For decades, this assumption held. Contexts within which such designs were deployed changed infrequently, if at all. Consequently, boundaries and preferences could be specified and stabilized, and a design with enduring qualities established, to be tweaked as changes in the environment took place.

However, such an approach is likely to run into problems in environments characterized by continual change (Barry and Rerup 2006). In such contexts, system boundaries are often unclear and user preferences are both heterogeneous and evolving. As a result, the goals and purpose of the design are likely to remain a continually moving target (Rindova and Kotha 2001).

The advent of new information technologies has made such fluidity possible. Different material artifacts and social groups can now be easily associated in a dynamic network. Rendering the functionality of any product or service through software enables real time changes to a design. Indeed, these technologies make it possible for customers to explore their preferences in use and for different social groups to engage with each other in an emergent fashion.

Simon appreciated the significance of such dynamic situations. For instance, in a section titled “Designing without final goals”, Simon explored a paradoxical but pragmatic view of design – to motivate activity that in turn generates new goals. Offering the example of the rebuilding of Pittsburgh, where new goals emerged after initial goals had been accomplished, Simon (1996: 163) concluded:

> "Making complex designs that are implemented over a long period of time and continually modified in the course of implementation has much in common with painting in oil. In oil painting, every new spot of pigment laid on the canvas creates some kind of pattern that provides a continuing source of new ideas to the painter. The painting process is a process of cyclical interaction between the painter and canvas in which current goals lead to the new application of paint, while the gradually changing pattern suggests new goals."

In these observations we see how a design approach need not be a static representation of a problem, but can involve a "theory-design-fly-test and start-all-over-again" methodology (Boland et al. 2006). Despite this acknowledgement, there has been relatively little work that explores the nature and implications of such a design approach. Indeed, as Boland (2004: 109) states, "Much of Simon's concern centers on the local and immediate experience of an individual who faces an environment that is essentially unchangeable. It is a given to which the managers must mold the organization."

The challenges that arise in applying a scientific approach to design in dynamic environments become all the more apparent when we consider the nature of change that is upon us. If we accept Woodward's (1965) powerful insight that technologies of work shape the way in which we work with technologies, new information technologies not only link islands of unconnected activities into an action net (Czarniawska 2004), but, in enabling such connections, change the very meaning of the term "design" to connote continual evolution via interaction. Jelinek (2004) fully understood the implications of this change when she stated:

> "A genuine revolution of possibility flowed from the move between prior flat file systems into relational databases and, in computer systems, the hyperlinked nodes of the internet. Similarly, the virtual organization – often pro tem, frequently voluntary, and broadly distributed – is the iconic organization of our times. But how does one design it? Or should one perhaps instead invite it to emerge? Is the issues deliberate design, or is it design of a process of interactions that allows a collaborative process to emerge? … How should we theorize about such organizations? … Should we embrace the ephemeral organization as a new norm?"

An image of an organization that is not a series of nested black boxes (March and Simon 1958) operating in an immutable environment but, rather, a hyper text organization that continues to emerge is a radical shift indeed. The design problem, then, is not one of developing a static interface (an edge in network terms) that connects the inside and the outside; rather, it involves the creation of multiple edges between many nodes in a dynamic network. In such an action net, each node can potentially act as a boundary object, "remaining between different realms, belonging to all of them simultaneously, and seen from different points of view" (Czarniawska 2004: 104). Given this, "When such an organization does emerge, it may be both transient and protean" (Jelinek 2004: 115).

These observations further highlight the difficulties associated with designing for completeness in a world that is continually changing. But, what does it mean to design for incompleteness? Is this an oxymoron? We think not. There are now a number of new organizational forms that suggest that incompleteness, rather than pose a threat, can instead be a virtue. We examine two such cases in this paper – the Linux operating system and the Wikipedia online encyclopedia. These cases provide us with an appreciation of the generative nature of incompleteness. They amplify what Weick (2004: 43) astutely pointed out, "life persists when designs are underspecified, left incomplete, and retain tension."

RESEARCH DESIGN

Our objective is to offer a set of observations that form the basis for an ongoing conversation among those interested in understanding the nature of design in continually changing environments. By no means do we claim to offer a full fledged theory – to do so would defeat the very premise of our argument. Here, we subscribe to the notions proposed by Boland & Collopy (2004), Romme (2003) and others who suggest that the value of theorizing lies in the

options that are generated rather than the uncertainties that are resolved. Along these lines, our intent is to sensitize readers to a pragmatic approach to design that harnesses the generative forces of incompleteness.

The research approach that we adopt in this paper involves a detailed exploration of two exemplary cases. Research on analogical thinking (Loewenstein et al. 1999) suggests that individuals who are presented with multiple cases exhibit greater ease in their abilities to identify underlying patterns. Based on this finding, we decided to provide not one but two in-depth cases. We deliberately chose to examine the Linux operating system and the Wikipedia online encyclopedia as each represents a design that is continually evolving. We tracked available information from a wide variety of online sources as a means of collecting raw data for our case narratives. The multiple data sources helped us "triangulate" (Jick 1979) in that there were very few disagreements among the data sources on the factual details involved. Two of the authors separately developed the case studies. The individual inferences drawn from each of the cases were discussed and verified with the other authors. Our aim is not to reach a state of theoretical saturation (Eisenhardt 1989; Glaser and Strauss 1967). Rather, much like the phenomena that we are studying, our aim is to offer a set of observations that will hopefully generate “theoretical tension” and form the basis for ongoing debate and understanding of this phenomenon.

LINUX: PERPETUALLY IN THE MAKING

Linux originated as a hobby project of Linus Torvalds, a computer science student at the University of Helsinki. Torvalds was interested in working on Unix, but the university had a limited number of machines running on this operating system for student use. Rather than wait to access a terminal, Torvalds decided to tinker on Minix, a Unix-like operating system created for personal computers.

Initially, he wrote programs to understand how the operating system interacted with the hardware and how he could use the features of the then-new 386 processor. Soon, he had written task-switching programs, a disk driver and a small file system. At this stage, Torvalds realized that he was actually working on a new operating system, and, as a result Linux 0.01 was born. Shortly thereafter, he posted the following message on comp.os.minix, an internet user group (Torvalds 1991b):

> "Hello everybody out there using minix – I'm doing a (free) operating system (just a hobby) for 386(486) AT clones….I'd like any feedback on things people like/dislike in minix, as my OS resembles it somewhat….I'd like to know what features most people would want. Any suggestions are welcome, but I won't promise I'll implement them :)"

At this stage, Linux lacked many of the functionalities that users had come to expect from an operating system. Its applications domain was non-existent. Indeed, even the numbering of the version – 0.01 – signified its unfinished status (Diedrich 2001).

Torvalds initial development efforts can be described as 'discovering design goals in action'. His announcement to the newsgroup suggests that he was unsure about what others might want in the emergent system. Although such lack of closure could be a problem from a traditional design perspective, it is interesting to observe how Torvalds turned this into a virtue. Upon releasing Linux version 0.02, he described the initiative as follows (Torvalds 1991a):

> "This is a program for hackers by a hacker. I've enjoyed doing it, and somebody might enjoy looking at it and even modifying it for their own needs. It is still small enough to understand, use and modify, and I'm looking forward to any comments you might have…If your efforts are freely distributable (under copyright or even public domain), I'd like to hear from you, so I can add them to the system….Drop me a line if you are willing to let me use your code."

This message illustrates how Torvalds, by ceding some control over his design, was able to harness the energies of the larger programming community. Key here was his decision to allow others to adapt the emergent system to their own use. Within two months of his

announcement, about thirty people had contributed close to 200 reports of errors and problems that they had confronted using Linux. In addition, these individuals developed new features for the nascent operating system (Moon and Sproull 2000). These extensions, in turn, became the basis for future directions along which the system evolved.

Traditionally, there has been a clear distinction between the designer and the user. However, Torvalds' actions brought about a blurring of boundaries between these two actors – the user could also be the designer (cf. von Hippel and von Krogh 2003). By catalyzing such a co-creation process, Torvalds ensured that the purpose and functionalities of Linux would now emerge from the distributed efforts of multiple contributors. Participation by these actors helped address the deficiencies that arose (such as missing features, bugs and the like) as the system emerged. Providing users with an opportunity to inscribe their local contexts into the design enabled the development and linkage of components in diverse and sometimes non-obvious ways. As these inputs were incorporated, the very purpose and functionality of the platform itself changed.

Technology is society made durable, suggested Latour (1991), and in the case of Linux we can recount many instances of how technology made engagement among contributors possible. Clearly, access to technological tools such as the Internet and related mailing lists and news groups made it possible for actors to engage with the emergent platform and with one another. More specifically, Linux benefited from the availability of a set of tools from the GNU project, a Unix-like platform that was easily portable to Linux (Kerner 2006)[6]. In particular the GNU C compiler was critical for building executable files from Linux source code. Similarly,

[6] While the GNU collaboration had developed a valuable set of tools for Unix-like operating systems, they were still having problems with their own kernel at the time Torvalds started his project. The reuse of the GNU tool set offered Linux a lot of functionality whereas the new operating system filled the gap of the missing kernel in the GNU project. The incompleteness of both systems allowed them to be merged them into a flourishing platform: the Linux kernel and the GNU tool set.

"installers" that facilitated reuse and distributed modifications contributed to the growing functionality of the system. For example, the Linux community experienced a large growth in 1992 when the Yggdrasil distribution made it possible for users to install Linux from a CD-Rom (Diedrich 2001). Finally, Torvalds redesigned the Linux kernel to have one common code base that could simultaneously support a separate specific tree for different hardware architectures, thereby greatly improving its portability (Torvalds 1999). Taken together, these tools made it possible for contributors to continuously extend Linux in a decentralized manner.

Moreover, social rules built into the technology (Lessig 1999) further fostered generative engagement by actors. A key development on this front was Torvalds decision to release the kernel under the General Public License (GPL), which mandated that any user modifying or adding to the source code would in turn have to make their own contributions available to everyone else (Stallman 1999). This decision signaled to the community that ownership and control of Linux was not Torvalds alone. Rather, it facilitated the establishment of a stable "generalized exchange system" (Kollock 1999) in which people both contributed to and benefited from the assistance of others. Establishing mechanisms that recognized the contributions of individuals further reinforced the feeling of common ownership. A "credits" file made available with every released version of Linux listed various contributors and their roles in developing and maintaining the kernel.

The availability of the source code played a critical part in the actor's generative engagement with the platform. Through documentation in repositories such as the CVS (Concurrent Version System), the source code served as a design trace that provided a memory of how the platform had evolved until a specific point in time. While creating options for the future, contributors could go back to the past to find out how solutions to related problems had

been developed. Moreover, the design trace provided the interpretive flexibility (Pinch and Bijker 1984) required to recontextualize the existing platform and make it work in new application areas without losing coherence with the original platform. Finally, enabling developers to leave their footprint in the source code ensured that their identities became intertwined with the platform and served as a strong attractor for them to contribute.

Overall, these social and technical mechanisms made it easier for contributors to tinker with the system and incorporate their own notions of how it should be further developed. In doing so, they extended Linux in ways that collectively covered a far greater domain than any single individual could have imagined. This manifested itself in the development of a variety of utilities and device drivers that supported specific hardware and peripherals (Thiel 1991). In an interview, Torvalds had this to say about the community's engagement with the platform:

> "After I had published Linux on the web, other users requested features I had never thought of and more and more new ideas emerged. Instead of a Unix for my own desktop, Linux should now become the best operating system ever. The requests by other people and their patches and help later on made the whole thing more interesting." (Diedrich 2000)

Our observations from Linux, then, suggest that incompleteness is generative in two different ways. At one level, incompleteness serves as a trigger for the creation of many diverse ideas on how a design can be extended and further developed. At another level, engagement with such a system both transforms the design as well as creates new avenues for ongoing engagement which, in turn, attracts a new set of contributors who bring into the fold their own contextualized needs, purposes and goals.

We illustrate these dynamics by recounting recent efforts to establish Linux as a desktop operating system. While non-existent a decade ago, this initiative made the graphical user interface a central component of current Linux versions (Diedrich 2000). At the same time, this

new context (“Linux on the desktop”) attracted the interest of a broader set of developers and opened up the system to new applications such as word processing (as exemplified by Sun’s Star Office). This observation highlights a key benefit of designing for incompleteness – generative engagement with a platform is self-reinforcing in nature, with the boundaries of both its technical and social architecture co-evolving with one another (see also Neff and Stark 2003). While generative engagement enables developers to cope with incompleteness in the present, it also is the source of incompleteness in the future.

Such generative engagement by a multitude of users, however, needs to be channeled to prevent fragmentation. Since its inception, critics of Linux have maintained that the laissez-faire approach to its development (reinforced by the inclusive yet voluntary nature of participation) has left the system vulnerable at various levels (Singer 2005). For instance, it is possible for individuals to introduce security holes or malicious code intended to cause damage. Even contributors with the best of intentions can sometimes inadvertently bring about damage to the system. More fundamentally, the lack of control can lead to fragmentation of the platform as factions pursue different avenues of development. For example, the many different distributions of Linux (e.g. RedHat Linux, SUSE, or Debian) all come with their own patches when updates to the system are released. In some cases, there have been so many changes made to the original package that the distributions have become incompatible (Wolf 2001). Given these challenges, what governance mechanisms can be put in place to enable incompleteness to be harnessed beneficially?

Linux’s governance approach is best described as “centrally facilitated yet organizationally distributed” (Garud and Kumaraswamy 2005). A combination of technological and social rules have worked in concert to keep the platform from falling apart. Ongoing

documentation of the development process, inscribed into the source code and CVS, served as a form of cohesive tissue. Communication tools such as chat rooms and bulletin boards acted as threads that ordered interactions among contributors across space and time (Lanzara and Morner 2005). These mechanisms were buttressed by an overarching meritocracy within the community in which developers focus on providing new code and modules that added features and functionality to the platform while others, based on their reputation and prior contributions, assumed the task of maintenance which involved evaluating code, modules and patches submitted by developers for inclusion into new releases. Sitting atop this meritocracy was Torvalds who centrally facilitated decisions on strategic issues to enhance and further develop the Linux kernel. Decisions to deliberately change the platform were entrusted to a much smaller group of individuals when compared to the total number of contributors. And finally, the provisions of the GPL provided the legal and cultural basis of interaction among community members.

The governance mechanisms underpinning Linux, then, have facilitated access to knowledge, encouraged constant tinkering and curbed private appropriation and free riding. Together, these mechanisms have operated with an almost invisible touch vis-à-vis ongoing developmental activity on the platform. This has enabled extensive experimentation to take place on the system even as it maintains a coherent core. By contrast, a tightly controlled design would bear the risk of falling apart given the difficulties of accommodating the contradictory requirements of a heterogeneous community.

A number of Linux's governance mechanisms have themselves emerged over time in a relatively unstructured fashion. One such convention focused on maintaining the integrity of each successive Linux release even as distributed development progressed unhindered. An even

numbered release (for instance, version 2.4) denoted that the release is stable and ready for use, whereas an odd numbered release (for instance, version 2.5) denoted that the release is still being built and under evaluation. The announcement of an odd numbered release initiative served as the call for new code and modules that would drive the current stable release forward. After a period of development, testing, discussion and reviews, no new additions were allowed so that the entire operating system could be tested for performance and reliability. Once testing had ended and modules of code that compromised reliable performance were removed, the operating system was ready to be released as a stable even-numbered version for widespread use. Such a convention served to establish regularity in the process of distributed development.

Overall, our case description of Linux highlights what it means to be incomplete by design. To the extent that the incipient system has "latent evolvability" – i.e., the capacity to incorporate functionality relatively easily – it is possible to harness its generative properties. Blurring the distinction between users and producers, assuming that preferences are heterogeneous and evolving and maintaining abstractness in goal definition often facilitates this generative process. These facets lie in contrast to scientific approaches to design that favor clear boundary definitions, fixed user preferences, design closure and platform stability. While systems that are designed for incompleteness may be ugly and messy by conventional design evaluation metrics, they can often outperform traditional designs by being extremely adaptable to continually changing contexts. Adopting such a design approach, then, involves appreciating design as the interplay between outcome and process, with one influencing the other on an ongoing basis. Designs are animate, living entities that are in an ongoing state of change.

WIKIPEDIA: A LUMPY WORK IN PROGRESS

Initiated in 2001 by Jimmy Wales and Larry Sanger, Wikipedia's ambitious mission has been to "create and distribute a free encyclopedia of the highest possible quality to every single person on the planet in their own language" (Wales 2005). As of 2006, it had become one of the most visited websites in the world, with 5 billion page views monthly. Its English-language version now has over 1 million articles and by this measure is almost 12 times larger than the print version of the Encyclopedia Britannica. Currently, it is growing at around 5-6,000 definitions and updates a day. What is truly remarkable is that Wikipedia has run as a non-profit organization since 2003 and has five employees in addition to Wales. It meets most of its budget through donations, the bulk of these being contributions of $20 or less (Schiff 2006).

The typical encyclopedia, as represented by the Britannica, is the epitome of a product that has been designed to be complete. Borrowing from the principles of scientific management, its production process involves assembling a large group of experts, working under the direction of a manager, each performing a task on a detailed work chart to produce a work of enormous breadth. This product is then packaged and bound in a set of volumes as an authoritative and accurate source for knowledge. Updates to this tome are made available on a periodic basis (typically annually). On this front, the encyclopedia closely subscribes to a scientific approach to design, with a clearly defined purpose, a production process that is neatly modularized, starkly delineated boundaries between producers and users, and an overarching emphasis on constructing a product that is stable and reliable.

Contrast this with the inner workings of Wikipedia. While the objectives of this initiative are ostensibly the same as that of the Britannica, any registered user can write an article that others subsequently modify and refine. This implies that participation in its development is

deliberately inclusive, blurring the boundaries between user and producer. This conceptual shift – along with other related mechanisms that enable generative engagement – has contributed to the construction of a system that has now begun to fundamentally question the ontological basis of a traditional encyclopedia.

Technically, entries are made using a wiki, a tool first developed by Ward Cunningham in 1995 that allows multiple users to create, edit and hyperlink pages. The word "wiki" comes from the Hawaiian word for "quick", but also stands for "what I know is...", and these definitions, taken together, provide an essence of the design approach underlying Wikipedia. Anyone can initiate a new article by making a basic entry - a “stub” in Wikipedia parlance. Content must be both verifiable and previously published. The website has an online style manual that provides guidelines for article-writing and layout that are remarkable in their simplicity: a clear introduction, short paragraphs, appropriate images, interesting quotations on the topic and cross-links to related subjects within the site and beyond. While articles in their early stages may feature little of this, they are slowly "wikified" towards the preferred format (Eisenberg 2005). More significantly, if you search the site for a topic not already covered, you are encouraged to write it yourself. Knowledge, from this perspective, is assumed to reside in a distributed fashion within the entire online human network.

Why might someone contribute to Wikipedia? To understand this, it is important to realize that the project functions within “open source” principles (Raymond 1999) in that its contents are under the GNU Free Documentation License. This allows contributors to work for the benefit of a worldwide audience without fear that their efforts might be hijacked. For many, the goals of Wikipedia – a free (as in freedom as well as money) encyclopedia that is a fun experience to create – are an important draw. This sometimes transforms itself into a strong level

commitment for a sub-set of these individuals, who end up spending a considerable number of their waking hours tending to various facets of the project. As Oliver Brown, a high school teacher evocatively describes his attachment to the website "You can create life in there. If you don't know about something, you can start an article, and other people can come and feed it, nurture it." (Pink 2005)

Given the minimal restrictions on membership, the project has turned fundamental assumptions related to design of a knowledge system on their head. The system is truly democratic in that it does not favor certified experts over the well-read amateur. As Wales puts it, “To me, the key is getting it right. I don’t care if they’re a high-school kid or a Harvard professor” (Schiff 2006). Over 100,000 individuals have made contributions to the website since its inception, with a 24-year-old University of Toronto graduate being the site’s premier contributor, having written or edited more than 72,000 articles since 2001.

How can a system that is based on the participation of literally any individual become a useful knowledge source? On Wikipedia, every article has an open invitation to edit it (using the wiki), with the operational motto being "Be bold". The expectation is that most edits will be improvements, and they are. As more people visit the website, the hope is that they will fill out missing pieces and make entries better. Wikipedia, then, appears to operate from the presumption that any individual’s knowledge is by definition incomplete, and that multiple ongoing revisions enabled by mass collaboration tools and involving a large, inclusive group of “eyeballs” will produce a reliable yet continually evolving knowledge repository. More fundamentally, it reflects an appreciation of the inherently accretive nature of knowledge, one in which the content of any article provides the generative basis for the next set of changes. As Earl (2005) commented, “The philosophy of Wikipedia is that an article gains validity and maintains

currency by continuous widespread updating. Thus, hitherto it has not declared any item finished".

Such updating is enabled by the access to users of a log of all the prior activities that have transpired relating to a specific entry. In other words, the wiki keeps track of all the changes made by users and allows them to compare multiple versions of an article even as they make edits to it[7]. This information serves as a design trace which allows actors to understand how and why an article has emerged over time. Possessing such an overarching temporal perspective is critical to repairing damaged elements in the entry as well as reverting to an older version if the current one is inaccurate. Equally significantly, this trace can form the basis for generating new directions in which the article can be developed. The design trace, then, both chronicles and initiates generative engagement with an article. In doing so, it serves as a locus of coordination as well as a point of departure, allowing an article to remain in a state of perpetual change.

In addition, social rules embedded in the technology comprising Wikipedia's website further enable generative engagement. For example, tags such as "The neutrality of this article is disputed" or "This article or section is in need of attention from an expert on the subject" represent calls for action to improve an article. Moreover, disambiguation notices placed at the bottom of certain articles help readers find other articles with similar names. This both enables users to accurately locate the information they are looking for as well as helps them figure out which article they may want to contribute to. Finally, the creation of an ontological scheme via categories aids users and contributors in organizing the information available on the site. More significantly, categorization of entries itself facilitates a process of adaptive structuration (DeSanctis and Poole 1994) in which the contribution of articles with different content change

[7] We invite the reader to experiment with the design trace for the Wikipedia entry on "Design". On the website http://en.wikipedia.org/w/index.php?title=Design&action=history, the evolution of roughly 500 revisions of this article can be traced.

the meaning of a category that subsequently gets relabeled. This dynamic highlights how the community structures Wikipedia in use.

An interesting by-product of such an evolving knowledge project is its ability to instantaneously respond to current events. When the Indian Ocean tsunami erupted in late 2004, contributors produced several entries on the topic within hours. By contrast, the World Book, who's CD-ROM allows owners to download regular updates, had not updated its tsunami or Indian Ocean entries a full month after the devastation occurred (Pink 2005). This ability to provide instant information is a metric on which a knowledge repository organized to change perpetually outperforms a traditional encyclopedia.

The flip side of this design approach, however, is that it often produces entries that are amateurish at best. While the facts may be sturdy, clarity and concision is often lacking. The initial contributor to an article can set its tone and is not necessarily highly knowledgeable in the area (Wattenberg and Viegas 2006). Disagreements on an article can lead to repeated back-and-forth editing, with the user who spends the most time on the site – or who yells the loudest – prevailing in the end. Given the obsession of many users to rack up edits, simple fixes often take priority over more complex edits. Moreover, vulnerabilities are inherent in such a system precisely because it allows open access. There have been many incidences of vandalism on the site that involve individuals inserting obscenities and absurdities within entries. Given these shortcomings, what governance mechanisms can be put in place to ensure that an article does not devolve into an unproductive morass?

Here again, a mix of technical and social elements working together serve to maintain the integrity of the website. Five webbots continually troll the site, searching for obscenities and evidence of mass deletions and reverting text as they go. Any editing on an article is

automatically logged on a "Recent Changes" page that participants can monitor. If someone sees something inane, false, biased, or otherwise defective, he or she can quickly and easily change the text. On this front, many regular users set up watch lists for entries they care about, so that they are notified immediately of new edits. Moreover, every article also carries with it a separate discussion section on which debates about what to include on the page are encouraged. Besides this, users employ IRC (Internet Relay Chat) to discuss ongoing issues, from article details to general policy. The integrity of information on the website, then, is maintained by a core group of committed volunteers.

In addition, these distributed contributions are held together by an underlying community structure. This includes anonymous contributors, people who make a few edits. Next, there are registered users who make edits using their byline. Administrators are individuals who can delete articles, protect pages, and block IP addresses. Another group includes bureaucrats and stewards, who appoint administrators. Finally, there are the super elites who can make direct changes to the Wikipedia software and database. There is also a mediation committee and an arbitration committee that rule on disputes.

The reputation of the individuals involved in such governance is important as it serves as a key proxy for the validity of the knowledge and intent of the contributors. In this specific case, the inner circle – referred to as Wikipedians – know each other and value their reputations, which are themselves built from the bottom-up through their past activity on the site. The consequence of such reputation building has been the creation of a meritocracy with individuals occupying more central positions because of their ongoing valued contributions when compared to others. As Jimmy Wales described how governance works on Wikipedia:

> "When people first encounter Wikipedia and see that anyone can come in and edit pages, they imagine that a million different people each added a sentence, and it

> somehow turned into this cohesive work. But the project isn't really like that. Really there are 200-300 people that I would consider the core of the community, who are organizing, reviewing, supervising, and checking things. It ends up being those people who are writing the bulk of Wikipedia." (Eisenberg 2005)

In terms of policies, the founders of Wikipedia instituted a NPOV (Neutral point of view) rule early on, according to which contributors are encouraged to present conventionally acknowledged "facts" in an unbiased way as well as accord space to both sides when arguments occur (Poe 2006). Over time, Wikipedia has instituted more rules, reflecting the complexities involved in managing a growing decentralized community. In 2004, the 3R rule was formalized that blocked users, who reverted the same text more than three times in a 24 hour period, from editing for a day. More recently, a policy that prevented certain contentious subject matters from being openly edited by users was ratified (FinancialWire 2006).

This has prompted concern that Wikipedia is rapidly becoming a regulatory thicket complete with an elaborate hierarchy of users and policies about policies. Researchers who have studied the site have found that the talk pages and "meta pages" – those dealing with coordination and administration – have experienced the greatest growth (Wattenberg and Viegas 2006). Wales, while ambivalent about the growing number of rules and procedures, recognizes them as necessary. According to him, "Things work well when a group of people know each other, and things break down when it's a bunch of random people interacting" (Schiff 2006). And Earl (2005) captured Wikipedia's dilemma as follows:

> "This is a classic knowledge-creation conundrum. On the one hand there is real advantage in assembling collective wisdom; on the other hand there are concerns about validity and about incentives or, more particularly, disincentives for content contributors."

However, taking these steps raises an even more fundamental issue: should the Wikipedia project abandon its current design approach in order to emulate a traditional encyclopedia? For

many, the incomplete nature of an article is valuable in its own right: first, such an article is better than nothing at all; second, this article might actually trigger more experienced participants to make a contribution that they would not have done otherwise. Instituting governance mechanisms that make it appear more like an encyclopedia could potentially rob Wikipedia of its unique identity and impede generative engagement by contributors. As Carr (2005) elaborated:

> "Wikipedia is not an authoritative encyclopedia, and it should stop trying to be one. It's a free-for-all, a rumble-tumble forum where interested people can get together in never-ending, circular conversations and debates about what things mean. Maybe those discussions will resolve themselves into something like the truth. Maybe they won't. Who cares? As soon as you strip away the need to be like an encyclopedia and to be judged like an encyclopedia - as soon as you stop posing as an encyclopedia - you get your freedom back."

This observation further underscores the need to embrace value in incompleteness and establish governance mechanisms that harness its benefits. Put differently, these mechanisms must coordinate distributed activities without being too restrictive. More importantly, to the extent that the goals and metrics of more traditional approaches are adopted in evaluating designs that are inherently incomplete, there exists a real danger of losing the unique value that they provide. On this front, it becomes important to appreciate how incompleteness facilitates generative engagement that is often key to long-term viability of a design.

DISCUSSION

The Linux and Wikipedia cases affirm what Boisvert (who built upon Deweys' (1934) pragmatic approach) pointed out: "Affairs are never frozen, finished, or complete. They form a world characterized by genuine contingency and continual process. A world of affairs is a world of actualities open to a variety of possibilities" (Boisvert 1998: 24). Indeed, these cases provide a deeper appreciation of the title of the paper – incomplete by design and designing for incompleteness. In an environment that is continually changing, designs that have been

completed at a point in time are likely to become incomplete over time. On the other hand, designs that anticipate their incompleteness are likely to be more complete over time.

Design participation within the pragmatic approach

The traditional scientific approach employed principles from the natural world to design an artifact with enduring qualities that fulfilled a specific purpose in an unchanging world (Simon 1996). From this perspective, design was fixed in time and space; it was opened and modified only to accommodate exogenous environmental changes. Moreover, the locus of design – i.e., the demarcation between designer and user – was clear and unambiguous.

In contemporary environments, however, the distinction between designers and users has blurred, resulting in the formation of a community of co-designers who inscribe their own contexts into the emergent design, thereby extending it on an ongoing basis in diverse and non-obvious ways. Such generative engagement by multiple co-designers is facilitated by numerous socio-technical mechanisms. Tools such as the wiki, licenses such as the GPL, forums such as bulletin boards and the infrastructure provided by the Internet, work with one another to facilitate participation and enable distributed development. This dynamic action net (Czarniawska 2004), then, contributes to the design remaining in a fluid state.

We can further contrast design participation within the scientific and pragmatic approaches by returning to the parable of the watchmakers offered by Simon. In its original version, user engagement with the design was considered an interruption resulting in watches that remained incomplete and therefore of little value. By contrast, the Linux and Wikipedia cases demonstrate that incompleteness acts as a trigger for generative engagement by co-designers. They are the ones who complete what they perceive is incomplete. They discover the purpose of a design in use. They create avenues for future development that, in turn, attract new

groups of co-designers. From a pragmatic design approach, then, what was considered to be an interruption now becomes the basis for ongoing organizing.

Design task within the pragmatic approach

For co-designers, the design task is very different from the one faced by designers adopting a scientific approach. For the latter, optimization of an objective function given constraints (as in linear programming), represented the dominant approach to designing. By contrast, contemporary designs such as Linux and Wikipedia can be conceptualized as an interlinked set of subjective functions, where one person's subjective function serves as another person's constraints. Complicating matters, the set of interlinked subjective functions is itself underspecified as it emerges in use over time. Under these conditions, the design remains incomplete as the solution to the optimization problem corresponds to more than one point in an n-dimensional space of design parameters. It is for this reason that co-designers must learn to theorize on the fly – i.e., they become reflective practitioners (Schon 1983) who generate provisional workable solutions to their immediate problems, knowing fully well that the platform that they draw upon and the context to which they apply their solutions will inevitably change.

What is the role of modularity in such an emergent system? Task partitioning (von Hippel 1990) is still possible, but, the partitioning is such that the components are not nearly as decomposed as Simon described in his watch-making example. Rather, partitioning is partial, allowing multiple actors to engage in overlapping sets of tasks in parallel with some redundancy built in. Interdependence between partitioned tasks results in a situation where changes in one task have a cascading effect on the remaining tasks. Consequently, the system never comes to a rest. Other scholars have suggested that such incompleteness in partitioning can be a problem (Ethiraj and Levinthal 2004).

The Linux and Wikipedia cases, however, suggest that there may be benefits to be realized from partial partitioning. This is because such a "shared division of labor" (Nonaka and Takeuchi 1995) allows for redundancy of functions, i.e., certain functions can be fulfilled by more than one component. This enables the system to work even if some components are damaged or are left incomplete. More significantly, such redundancy of functions in a module can be generative in that a component could be deployed for a different purpose thereby facilitating reconfiguration of the design in response to changes in the environment (Garud and Kotha 1994). It also allows the system to more easily assimilate emergent contributions from the co-designers into the ever changing platform.

Design governance within the pragmatic approach

Given the distributed, emergent and protean nature of designs, the challenge now becomes one of establishing rules that provide some stability. While an absence of rules is likely to lead to design fragmentation, too many rules can potentially stifle the design. It is this paradox that needs to be managed to preserve the value proposition that such designs offer.

An appropriate form of governance is required to coordinate real time distributed contributions in a way that preserves the design's dynamic qualities – i.e., one which allows elements of a system to inform but not determine one another (Barry and Rerup 2006: 267). Governance mechanisms need to be underspecified (Weick et al. 1999) or semi-structured (Brown and Eisenhardt 1997); that is, they possess minimum critical specifications (Emery 1980) to keep the design in a state that is neither too fluid nor too crystallized (cf. Gehry 2004 for this distinction).

The design trace is a key element that enables such governance. By providing widespread access to knowledge on who contributed what, when and why, the trace makes it easier for actors

to understand how a design has emerged over time. Possessing such an overarching temporal perspective is often critical in mitigating design fragmentation. For instance, if a new contribution turns out to be damaged or incompatible, then, the trace makes it possible to simply use an older version. Equally important, the trace can be viewed as a boundary architecture (Star and Griesemer 1989) that co-designers draw on to develop extensions to the design. In sum, the ongoing design trace serves as a locus of coordination as well as a point of departure for such designs.

To further elaborate, consider two different logics of engagement described by Bruner (1986). A causal logic, following the role model of the natural sciences, operates when the set of variables and relationships that define the functioning of a design are fully specified (see Romme 2003 for a more detailed explication). This logic, associated with a scientific approach to design, leads to the articulation of design rules (Baldwin and Clark 2000; Romme 2003). A narrative logic, on the other hand, operates on the basis of a narrative's internal coherence and its external coherence with the listener's existing knowledge. In providing designers with the interpretive flexibility required to generate contextualized solutions and to imagine what might be, the narrative can help coordinate distributed activities across time and space. Such an epistemology connects designs with the emergent purposes of the social groups involved.[8]

The design trace allows for these two different logics to operate simultaneously. The trace possesses a scientific logic that offers co-designers with the raison d'être of the design and the necessary details for it to be functional in real time. From this perspective, each node of a trace is but a module with tags that can be opened up and reused. At the same time, the trace also allows for a narrative logic to operate. Co-designers are motivated to participate with a design because of the flexibility that it offers, and the design in use that emerges is convincing to these

[8] We thank Georges Romme for this insight.

participants as the narratives recorded have verisimilitude (Bruner 1986). A design trace, then, possesses the equivalents of both an ostensive and a performative dimension (Feldman and Pentland 2003). By providing connections to assets across time and space, a design trace makes it possible for co-designers to engage in the present by building upon the past in order to create a new future (Ricoeur (1984).

In enabling the two logics to operate simultaneously, the trace is able to alleviate some of the problems associated with human memory and bounded rationality that Simon considered in his parable. Tempus and Hora were boundedly rational individuals, relying on their own short and long term memories. A mechanism such as the design trace could potentially have extended their memories by serving as a collective mind (Weick and Roberts 1993). As a narrative, the trace makes it easier for any individual to associate the design with templates that they have already developed. This makes it possible for interruptions to not only be welcomed, but to also serve as the basis for extension of the design.

In offering these observations we see how future options on a design are generated even as current contributions are incorporated. Moreover, recording changes in the design trace signals to co-designers that their contributions will be honored in the future. If such a trace were not to exist, part of the motivation to participate might be lost – why contribute to a commons when the contributions would not be used in the future? At the same time, if the co-designers were over-dependent on the extant trace, then the future design would largely be based on past experience and become path dependent. For a design (and its trace) to promote diachronic processes, then, expectations of the future and memories of the past should jointly inform contributions in the present.

CONCLUSION

We have explored the Linux and Wikipedia cases to sketch out the elements of a pragmatic approach to design. In continually changing environments, adopting a design approach that attempts to fix boundaries, goals and purposes is potentially counterproductive. Whereas, such an approach may produce a system that is optimal at a point in time, given continual change, the system is likely to rapidly become obsolete over time. Under these conditions, a pragmatic approach – one that views design as continually evolving and essentially incomplete -- may be more appropriate. Within such an approach, boundaries between designers and users become blurred, heterogeneous user preferences emerge in use, tasks remain partially partitioned and the goals of the design emerge through interaction. Such an approach to design acknowledges the partial nature of knowledge possessed by any one individual and focuses on the means by which distributed knowledge can be harnessed. In summary, while the scientific approach views incompleteness as a threat, a pragmatic approach harnesses its value.

Eventually, a pragmatic approach involves the fusing together of two meanings of design – that is, as both process and as outcome. Any outcome is but an intermediate step in an ongoing journey, representing both the completion of a process as well as its beginning. Whereas the scientific approach emphasizes the need to crystallize designs, the pragmatic approach highlights the value of retaining fluidity. The essence of this approach is well captured by Hedberg, et al. (1976: 43) who noted, “Designs can themselves be conceived as processes – as generators of dynamic sequences of solutions, in which attempted solutions induce new solutions and attempted designs trigger new designs.”

REFERENCES

Alexander, C. 1964. *Notes on the Synthesis of Form*. Harvard University Press, Cambridge, MA.

Baldwin, C.Y., K.B. Clark. 2000. *Design Rules - The Power of Modularity*. MIT Press, Cambridge, MA.

Barry, D., C. Rerup. 2006. Going Mobile: Aesthetic Design Considerations from Calder and the Constructivists. *Organization Science* **17**(2) 262-276.

Boisvert, R.D. 1998. *John Dewey: Rethinking Our Time*. State University of New York Press, Albany.

Boland, R.J. 2004. Design in the Punctuation of Management Action. R.J. Boland, F. Collopy, eds. *Managing as Designing*. Stanford University Press, Stanford, CA, 106-120.

Boland, R.J., F. Collopy, eds. 2004. *Managing as Designing*. Stanford University Press, Stanford, CA.

Boland, R.J., M. Jelinek, A.G.L. Romme. 2006. Call for Papers: Organization Studies as a Science of Design. *Organization Studies*.

Brown, S.L., K.M. Eisenhardt. 1997. The Art of Continuous Change: Linking Complexity Theory and Time-Paced Evolution in Relentlessly Shifting Organizations. *Administrative Science Quarterly* **42** 1-34.

Bruner, J.S. 1986. *Actual Minds, Possible Words*. Harvard University Press, Cambridge.

Carr, N. 2005. Let Wikipedia be Wikipedia, http://www.roughtype.com/archives/2005/12/let_wikipedia_b.php.

Czarniawska, B. 2004. Management as Designing for an Action Net. R.J. Boland, F. Collopy, eds. *Managing as Designing*. Stanford University Press, Stanford, CA, 102-105.

DeSanctis, G., M.S. Poole. 1994. Capturing the Complexity in Advanced Technology in Use: Adaptive Structuration Theory. *Organization Science* **5**(2) 121-147.

Dewey, J. 1934. *Art as Experience*. Minton Balch & Company, New York.

Diedrich, O. 2000. Was Anwender tun, ist niemals falsch *c't*, 90.

Diedrich, O. 2001. Happy Birthday, Tux! *c't*, 162.

Earl, M. 2005. Wikipedia's Struggle to Govern a Knowledge Democracy *The Financial Times*, London.

Eisenberg, B. 2005. Wikipedia: An "Open Source" Encyclopedia Gains Respect *Software Design*, http://www.gihyo.co.jp/magazine/SD/pacific/SD_0501.html.

Eisenhardt, K.M. 1989. Building Theories from Case Study Research. *Academy of Management Review* **14**(4) 532-550.

Emery, F. 1980. Designing Socio-Technical Systems for 'Greenfield' Sites. *Journal of Occupational Behaviour* **1** 19-27.

Ethiraj, S.K., D. Levinthal. 2004. Modularity and Innovation in Complex Systems. *Management Science* **50**(2) 159-173.

Feldman, M.S., B.T. Pentland. 2003. Reconceptualizing Organizational Routines as a Source of Flexibility and Change. *Administrative Science Quarterly* **48**(1) 94-118.

FinancialWire. 2006. Wikipedia To Change Revision Policy. Jun 19, Forest Hills, MD.

Ford, H., S. Crowther. 1922. *My Life and Work*. Doubleday, Page & Company, Garden City, NY.

Garud, R., S. Kotha. 1994. Using the Brain as a Metaphor to Model Flexible Production Systems. *Academy of Management Review* **19**(4) 671-698.

Garud, R., A. Kumaraswamy. 2005. Vicious and Virtuous Circles in the Management of Knowledge: The Case of Infosys Technologies. *MIS Quarterly* **29**(1) 9-33.

Gehry, F.O. 2004. Reflections on Designing and Architectural Practice. R.J. Boland, F. Collopy, eds. *Managing as Designing*. Stanford University Press, Stanford, CA, 19-35.

Giddens, A. 1979. *Central Problems in Social Theory: Action, Structure, and Contradiction in Social Analysis*. University of California Press, Berkeley.

Glaser, B.G., A.L. Strauss. 1967. *The Discovery of Grounded Theory: The Strategies for Qualitative Research*. Aldine de Gruyter, Hawthorne, NY.

Hedberg, B.L., P.C. Nystrom, W.H. Starbuck. 1976. Camping on Seesaws: Prescriptions for a Self-designing Organization. *Administrative Science Quarterly* **21** 41-65.

Jelinek, M. 2004. Managing Design, Designing Management. R.J. Boland, F. Collopy, eds. *Managing as Designing*. Stanford University Press, Stanford, CA, 113-120.

Jick, T.D. 1979. Mixing Qualitative and Quantitative Methods: Triangulation in Action. *Administrative Science Quarterly* **24**(4) 602-611.

Kanigel, R. 1997. *The One Best Way: Frederick Winslow Taylor and the Enigma of Efficiency*. Viking, New York.

Kerner, S.M. 2006. Why is Linux Successful *Inernetnews.com*, http://www.internetnews.com/dev-news/article.php/3627061.

Kollock, P. 1999. The Economies of Online Cooperation: Gifts and Public Goods in Cyberspace. M.A. Smith, P. Kollock, eds. *Communities in Cyberspace*. Routledge, London, 220-239.

Langlois, R.N. 2002. Modularity in Technology and Organization. *Journal of Economic Behavior & Organization* **49** 19-37.

Langlois, R.N., P.L. Robertson. 1992. Networks and Innovation in a Modular System: Lessons from the Microcomputer and Stereo Component Industries. *Research Policy* **21**(4) 297-313.

Lanzara, G.F., M. Morner. 2005. Artefacts Rule! How Organizing Happens in Open Source Software Projects. B. Czarniawska, T. Hernes, eds. *Actor-Network-Theory and Organizing*. Copenhagen Business School Press, Copenhagen, 67-90.

Latour, B. 1991. Technology is Society Made Durable. J. Law, ed. *A Sociology of Monsters: Essays on Power, Technology and Domination*. Routledge, London, 103-131.

Lessig, L. 1999. *Code and Other Laws of Cyberspace*. Basic Books, New York.

Loewenstein, J., L. Thompson, D. Gentner. 1999. Analogical Encoding Facilitates Knowledge Transfer in Negotiation. *Psychonomic Bulletin and Review* **6**(4) 586-597.

March, J.G., H.A. Simon. 1958. *Organizations*. Wiley, New York.

Moon, J.Y., L. Sproull. 2000. Essence of Distributed Work: The Case of the Linux Kernel. *First Monday* **5**(11).

Neff, G., D. Stark. 2003. Permanently Beta: Responsive Organization in the Internet Era. P.E.N. Howard, S. Jones, eds. *The Internet and American Life*. Sage, Thousand Oaks, CA.

Nonaka, I., H. Takeuchi. 1995. *The Knowledge-Creating Company*. Oxford University Press, New York.

Parnas, D. 1972. On the Criteria to be Used in Decomposing Systems into Modules. *Communications of the ACM* **15**(12) 1053-1058.

Petroski, H. 1996. *Invention by Design: How Engineers Get from Thought to Thing*. Harvard University Press, Cambridge, MA.

Pinch, T.J., W.E. Bijker. 1984. The Social Construction of Facts and Artefacts: or How the Sociology of Science and the Sociology of Technology might Benefit from Each Other. *Social Studies of Science* **14**(3) 399-441.

Pink, D.H. 2005. The Book Stops Here *Wired*.

Poe, M. 2006. The Hive *The Atlantic Monthly*, 86-96.

Raymond, E.S. 1999. *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*. O'Reilly & Associates, Sebastopol, CA.

Ricoeur, P. 1984. *Time and Narrative*. University of Chicago Press, Chicago.

Rindova, V.P., S. Kotha. 2001. Continuous "Morphing": Competing through Dynamic Capabilities, Form, and Function. *Academy of Management Journal* **44**(6) 1263-1280.

Romme, A.G.L. 2003. Making a Difference: Organization as Design. *Organization Science* **14**(5) 558-573.

Rosenberg, N. 1982. *Inside the Black Box: Technology and Economics*. Cambridge University Press, Cambridge.

Schiff, S. 2006. Know It All *The New Yorker*.

Schon, D.A. 1983. *The Reflective Practitioner: How Professionals Think in Action*. Basic Books, New York.

Simon, H.A. 1962. The Architecture of Complexity. *Proceedings of the American Philosophical Society* **106**(6) 467-482.

Simon, H.A. 1996. *The Sciences of the Artificial*, third ed. MIT Press, Cambridge, MA.

Singer, M. 2005. Linux Creator: A Little Fragmentation is Good *Inernetnews.com*, http://www.internetnews.com/dev-news/article.php/3467241.

Stallman, R. 1999. The GNU Operating System and the Free Software Movement. C. DiBona, S. Ockman, M. Stone, eds. *Open Sources: Voices from the Open Source Revolution*. O'Reilly & Associates, Sebastopol, CA, 53-70.

Star, S.L., J.R. Griesemer. 1989. Institutional Ecology, `Translations' and Boundary Objects: Amateurs and Professionals in Berkeley's Museum of Vertebrate Zoology, 1907-39. *Social Studies of Science* **19**(3) 387-420.

Thiel, W. 1991. Keyboard.S with German Keyboard *Linux-Activists*, ftp://tsx-11.mit.edu/pub/linux.

Torvalds, L. 1991a. Free Minix-like Kernel Sources for 386-AT *comp.os.minix*, http://listserv.nodak.edu/cgi-bin/wa.exe?A2=ind9110a&L=minix-l&T=0&P=8303.

Torvalds, L. 1991b. What Would You Like to See Most in Minix *comp.os.minix*, http://listserv.nodak.edu/cgi-bin/wa.exe?A2=ind9108d&L=minix-l&T=0&P=4457.

Torvalds, L. 1999. The Linux Edge. C. DiBona, S. Ockman, M. Stone, eds. *Open Sources: Voices from the Open Source Revolution*. O'Reilly & Associates, Sebastopol, CA, 101-111.

von Hippel, E. 1990. Task Partitioning: An Innovation Process Variable. *Research Policy* **19** 407-418.

von Hippel, E., G. von Krogh. 2003. Open Source Software and the "Private-Collective" Innovation Model: Issues for Organization Science. *Organization Science* **14**(2) 209-223.

Wales, J. 2005. Wikipedia is an Encyclopedia, http://mail.wikipedia.org/pipermail/wikipedia-l/2005-March/038102.html.

Wattenberg, M., F.B. Viegas. 2006. The Hive Mind Ain't What It Used to Be *Edge*, http://www.edge.org/discourse/digital_maoism.html.

Weick, K.E. 2004. Rethinking Organizational Design. R.J. Boland, F. Collopy, eds. *Managing as Designing*. Stanford University Press, Stanford, CA, 36-53.

Weick, K.E., K.H. Roberts. 1993. Collective Mind in Organizations: Heedful Interrelating on Flight Decks. *Administrative Science Quarterly* **38** 357-381.

Weick, K.E., K.M. Sutcliffe, D. Obstfeld. 1999. Organizing for High Reliability: Processes of Collective Mindfulness. R.I. Sutton, B.M. Staw, eds. *Research in Organizational Behavior*. JAI Press, Stamford, CT, 81-123.

Wolf, C. 2001. The ROCK Linux Philosophy *linux devcenter.com*.

Woodward, J. 1965. *Industrial Organization: Theory and Practice*. Oxford University Press, New York.