Technology, Language, and Public Decisions: Finding Common Ground for Experts and Citizens

David D. Jensen
Computer Science Department
University of Massachusetts
Amherst, Massachusetts, USA
jensen@cs.umass.edu

Todd M. La Porte
School of Systems Engineering & Policy Analysis
Delft University of Technology
Delft, The Netherlands
tlaporte@tmn.com

Abstract—Many conflicts over specific technologies could be alleviated by focusing attention on the design stage and by encouraging communication between designers and users. We present two approaches to achieving these goals. The first, pattern languages*, provides a common language for designers and users. The second,* component designs*, allows technologies themselves to serve as a shared workspace for designers and users.*

I. Current Problems

Technology and controversy are common companions. Many of the major technologies deployed in the latter half of this century have also been the subject of that period's most bitter disputes. Examples include nuclear power, nuclear weapons, industrial automation, agricultural chemicals, and automotive safety. More recent debates over computing technologies and communications networks lack some elements of previous controversies (e.g., public protests), but still arouse substantial conflict among their participants.

There should be little wonder that major technologies and major conflicts are related. Proponents of technologies often believe strongly that their solutions are good ones, they feel their technologies will benefit society, and they make a substantial investment in time, capital, and human resources to bring them to fruition. Opponents often feel that deeply-held values are at stake and that the technologies in question have unacceptable costs that are not understood or accounted for.

A. Controversies

Why do technologies arouse such passions? Technologies are often novel, and thus represent an agent of change that can be both identified and opposed. The basic principles of advanced technologies are often remote enough that they invite distrust and concern. Public debates help little in this regard, because they frequently devolve into debates among experts, who often invoke the public interest, but rarely make their debates accessible to the public.

Another reason is that technologies are often introduced with little involvement from those who will be most affected by them. To oversimplify, technologies are deployed in a two-stage process. First, technologies are researched, designed, and developed. This process can involve lengthy consideration of the probabilities of technical and commercial success, but this stage rarely involves broad participation and testing by potential users. Second, the technology is implemented, often accompanied by substantial public debate. This debate may be implicit, carried out in casual conversations and reporting by the news media. It can also be explicit, in the form of legislative hearings, regulatory proceedings, or union negotiations. Clearly this is an oversimplification, but it is fair to say that user involvement in research, design, and development is almost always far less than in the subsequent activities. The vast majority of technologies are presented to users as a *fait accompli*.

When technologies are implemented, the ensuing debates are often cast as referenda: Should we use nuclear power? Should we fund a light rail system? Should we use Alar? The choices are to accept the basic form of the technology (perhaps with a few hastily applied patches) or to send the designers “back to the drawing board.”

B. Costs

Regardless of the outcome, such referenda are costly. If the technology is rejected, then designers and producers must scrap years of financial and human investment. Such experiences make it less likely that potentially beneficial technologies will be introduced in the future, particularly if they represent radical departures from past practice.

Implementing a new technology can have even broader costs. New technologies can be disruptive to stable patterns of social, organizational, and personal behavior. Even when the technologies provide eventual benefits, the switch itself often has high costs. When the rate of technical change is relatively rapid, individuals and organizations have little time to reap the benefits of a fully integrated technology before the next generation comes along. A common example is the rapid turnover in computer hardware and software in many office environments.

Social costs can also accrue to non-users. For example, automobiles and freeways bring greater mobility and speed of transportation to users, but they also create costs associated with traffic and road safety, land-use, and air pollution. The benefits accrue only to users, but the costs impact nearly everyone, regardless of the automobile ownership or use.

Once substantial commitments have been made to a technology, such costs are difficult to remedy. We may ultimately ask whether a technology was really worth it, after deployment has already taken place. Not surprisingly, when it is too late, the answer is usually "well, I guess it must have been."

II. Diagnosis

Public debates about technologies are hampered by the differing perspectives and languages of the participants. Scientists, engineers, and technology managers often speak of constraints, specifications, and design tradeoffs. These experts feel confident in their ability to shape a technology, but strangely unable to convey their understanding of a technology to potential users. Potential users and other affected parties, on the other hand, often speak about their way of life and their values, how certain technologies "feel" wrong to them, and how technical specifications are inadequate to express or evaluate those feelings. These people feel powerless to influence the way technologies affect them.

A. Design Flexibility

The problem of different or even divergent views of technologies can be summed up in the notion that different values, economic, technical, social and even political, inform the design and construction of technologies. There usually are many ways to design, build, and implement a given artifact, system or process. The notion that there is "one best way" now seems quaintly Taylorist, overly rigid and deterministic. "There's more than one way to skin a cat!" is more likely our usual experience with technologies we use or build.

Yet many technology designers or developers seem to limit their view of useful alternatives to a narrow range drawn from the technical community or milieu from which they themselves arise. The degree of variation in computer graphical user interfaces seems to be shrinking, not growing, over time. VCRs seem *all* to have the same degree of user-unfriendliness, despite the fact that there are dozens of different models.

B. Examples: Communication Networks

To illustrate, consider how two different groups, traditional telecommunications engineers and computer network designers, approach the problem of providing highly reliable person-to-person communications. Telephone engineers developed regional switching centers with large computer switches and an organization dedicated to nearly error-free operations with massive peak-load capacity and network control software to manage signaling and routing.

Computer network designers, on the other hand, saw such highly centralized switching systems as a liability for truly reliable communications, so they developed packet-switching, which permits data to flow asynchronously from one modest computer node to another, with virtually no centralized organization to manage day-to-day operations. This system has some delay in communications between parties, making it more amenable to text, rather than voice, communications; the two systems are not perfectly comparable. But with the advent of such software as Internet Phone, upgraded routers, and increased capacity, even packet-switched networks may be able to handle large amounts of real-time voice communications.

Another set of examples is drawn from current debates about broadband networks, where disputes are about what goals the technologies should serve, often as a function of what the technologies can provide. The very definition of "broadband" may change as a function of whose technology solution is being discussed (See Box I).

It is no secret to technology developers in communications industries that their systems embody different values. This perspective is helpful in demonstrating to the *rest of us* what second-order economic, social, political and cultural values are associated with these technologies, and that they are not neutral in their social effects: design matters to technologies' use and effect, and to their insertion into society.

Furthermore, it is helpful in showing that conflict is an inherent part of the activity, and should be taken into account at the outset. This being so, it is important that a wide range of perspectives be attended to in the design and development process.

B. Impediments to Better Public Debates About Technologies

Public debates about technologies are hampered by three main problems. First, users often lack experience on which to base preferences. People have relatively little experience with different designs or configurations of technical artifacts or systems, yet experience is the primary mode for understanding preferences. Preferences only become apparent after a technology is deployed and used, and, perhaps, after an alternative path is experienced. Individuals familiar with standard American coal, gas or oil-burning central-generation electric power systems have a hard time understanding how a solar power infrastructure might be usable or even preferable to their current situations. Absent demand for alternatives, it is difficult for them to be deployed anywhere.

Technology demonstrations are a key way to overcome this problem, but for many technologies, large-scale deployment is essential to understand the full implications of the alternative, and also prohibitively expensive. Electric vehicles have

Box I: Differing Designs of Broadband Networks

The design of broadband networks provides a good example of the implications of different design choices.

Should broadband be essentially text-oriented data computer communication, such as e-mail? This could be provided with relatively inexpensive computers and modems on just about any telephone line, no matter how old, remote, or noisy.

Should broadband be more video-based, oriented primarily to videoconferencing and telephony? This could be provided by telephone companies through existing copper lines and increasingly available ISDN services.

Should broadband be video entertainment oriented, with ancillary voice and computer communications? This could be provided by telephone companies through existing local copper lines and fiber to neighborhood central offices, using ADSL technologies, essentially very fast modems (about 1.5 Mbits/second, the effective minimum for what is known as "VCR-quality" video). Each of these three conceptions would permit the telephone companies to maintain their centrality to electronic communications regardless of which scenario for a widespread broadband network use is ultimately adopted.

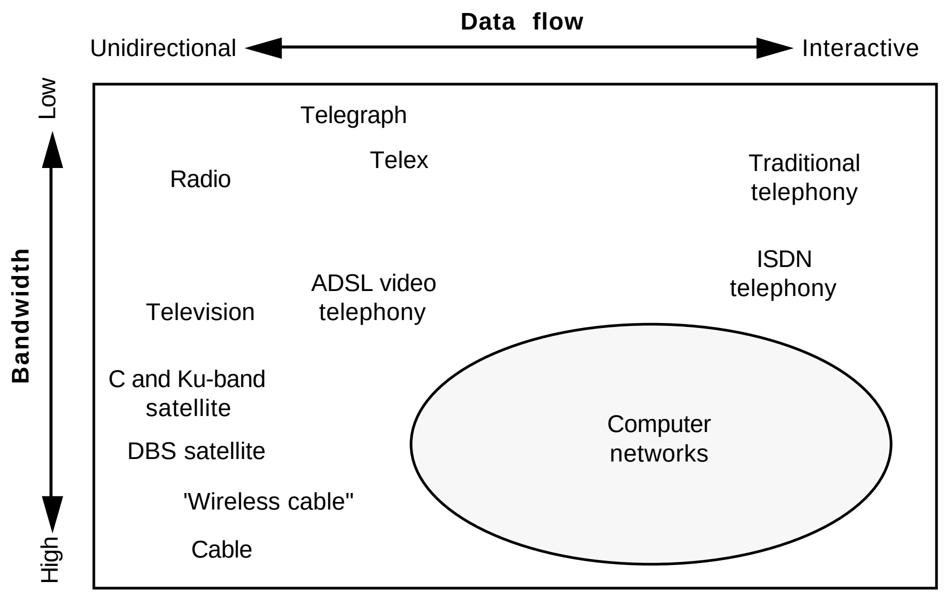
A different model comes from the cable television industry, which sees video entertainment as the main service objective. Cable television already delivers massive bandwidth, though generally not switchable or individually addressable.

Several other broadband technologies are also available, including traditional television. Television is a prodigious provider of bandwidth (6 Mhz per analog channel, which could sustain a digital data rate of about 9 Mbits per second), but is not interactive. Other terrestrial broadcast systems are coming along, such as the so-called "wireless cable" LMDS and MMDS.

And then there are a number of satellite services, each with different bandwidth offerings and degrees of interactivity. Satellite broadcasts in the C and Ku bands can be picked up by large dish antennas, often seen in rural areas where cable is unavailable. Direct broadcast satellite (DBS) communications services, such as Primestar or DirecTV, now provide high bandwidth entertainment services to users equipped with small dish antennas and converter boxes. And lower bandwidth services for global mobile telephony are coming online.

Fundamental differences in advanced communications networks are reflected in both the technologies and the uses to which they are put. These technological alternatives are backed by powerful industrial interests, which use both political and economic means to advance their preferred systems. The outcome that each industry would prefer is usually a combination of cost and efficiency considerations, *and* value orientations or concerns, including organizational power or control.

Figure I: Characteristics of Broadband and Other Communication Technologies



had this problem for years, even though they are an interesting alternative to internal combustion engines: the lack of a widespread refueling system makes potential users skeptical about the real utility of electric cars.

Second, technology promoters and potential users often speak different languages. Technology promoters often have a clear concept of "their" system, and they seek public decisions consistent with that vision, even to the point of adopting a language that inherently supports it. These experts feel confident that their technology is an appropriate solution to a problem they see. For example, nuclear power generation is an elegant solution to the problem of efficient electric power provision with low or no atmospheric emissions.

However, technology proponents often seem unable to convey their understanding of a technology to potential users, and find themselves frustrated that users don't see the technology in the "right" way. In the case of nuclear power, many users want highly reliable and safe operations without long-term radioactive waste storage problems even more than technical efficiency. But because they don't have a very well developed conceptual vocabulary for talking about the implications of a technology, they are unable to articulate uneasiness about it's design or operational requirements. Thus, opposition to particular design elements of nuclear power becomes opposition to nuclear power as a whole.

It is important to note that stakeholders aren't just dummies, unschooled and unschoolable in technological design, but instead have unarticulated conceptual expertise in their own right. What is missing is a vocabulary which allows stakeholders to participate in the dialogue that accompanies many technological developments.

Finally, better public debate about technologies is hampered because debate about design, if it happens at all, occurs *after* the technology is fixed. Designers and promoters are committed to their vision, and may have invested substantial resources to make it come about. Users coming late into the game generally lack the power to force reconsideration of options, or can do so only at great cost.

III. Focusing on Design

Is there a way to connect designers and users in a more constructive relationship? We believe that such a relationship is possible and that a focus on design is a key concept. These beliefs are central to the emerging field of *user-centered* design. Many theoretical discussions and case studies have been produced about the benefits of involving users in the design process, and we are fundamentally in agreement with much of this work.

Our work differs from much of the user-centered design literature in two important respects. First, it explores *conceptual mechanisms* through which the ideal of user involvement can be achieved. Calling on designers and users to interact more fully will do little good unless the conceptual frameworks available to them facilitate this interaction. Second, our work explores mechanisms that will function on a broad, *public* level. Many studies of user-centered design quite sensibly focus on designing technologies within organizations and workgroups. Our focus is on broader venues where technologies have created public controversies.

A focus on design is a way of avoiding unproductive "yes/no" referenda and unproductive shouting matches. By encouraging designs that respond to social, environmental, or other concerns, the benefits of technologies that promoters desire may be reconciled with the legitimate concerns of those on the receiving end of technology deployments. Sensitive design can minimize the social and economic costs that often lie at the heart of technological conflicts.

Why focus on design? The design stage is where basic changes in a technology can be made most easily and inexpensively. Few major resources have been committed for manufacturing and deployment, and a wider variety of development paths are open. It is also the stage where the vital creativity of designers and engineers is most easily enlisted.

However, how can we focus on design when designers and users speak such different languages? How can users participate when it is so difficult to foresee how technologies will shape the lives of users? How can we increase the flexibility of design and facilitate the involvement of users earlier in the process?

We propose two general approaches that address these questions. First, we argue that conceptual vocabularies of design, what have been called *pattern languages*, can greatly enhance communication between designers and users and help clarify design tradeoffs. Second, we argue that technologies with a specific character, what we call *component design*, offer designers and users a shared workspace that can provide important insights.

IV. Pattern Languages

A common language for both technology developers and technology users would allow both to discuss concepts that are relevant to each. One approach is to construct a "pattern language," a set of specific design concepts about a particular technology.

An elegant and nearly comprehensive example of such a language is provided by urban planner and architect Christopher Alexander and his colleagues at the University of California at Berkeley. Their books, *The Timeless Way of Building* and *A Pattern Language,*[1] lay out a coherent approach to the design and building of dwellings, workplaces, towns and communities that emphasizes the human and social use of the built environment.

1 Christopher Alexander et al., *The Timeless Way of Building* and *A Pattern Language,* (New York: Oxford University Press, 1977).

The central idea of a pattern language is a set of shared concepts that inform design, just as there is a set of shared concepts that inform linguistic communication. For the Berkeley team, the language consists of core, invariant concepts about the relationship of buildings and built structures to people, and how they live and work. Patterns, which are elements of a language, are descriptions of frequently encountered design problems and their solutions which result in good, comfortable, or effective environments. These patterns may be used in many situations, without giving the same result: they are like words of a language, with which a nearly infinite number of things can be said.

Alexander and associates distilled towns, buildings and construction techniques into 253 such patterns, drawing from vernacular architecture and existing structures which have found great success, either as conscious design contributions, or more often, as survivors of a long-term selection process. The concepts they identify are likely to be the residual, and fundamental, core elements of design for people, since the time span over which many of their examples are drawn is so long. In the endless process of building and rebuilding , those design principles which support human needs and preferences have been reinforced, while others disappeared.

Alexander's group proposes that these patterns compose an integrated whole, but that there may be many different pattern languages, of which theirs is only one. A shared pattern language is essential for societies to collectively negotiate the construction and use of physical living spaces.

We believe that the idea of pattern languages also can be applied to other technologies. Each technical domain may have one or more. In fact, nascent within each technical community is usually some notion of the relationship between design and use. If these notions were made more explicit and accessible, as regional planning, architecture and construction are within *A Pattern Language*, then better dialogue between technology developers and the public might ensue. We note that this type of descriptive and analytic approach has been used by technology assessment scholars and practitioners, but has general not been accompanied by the development of such conceptual languages as is proposed here.

A. Examples of Patterns

Understanding of pattern languages may be easier with examples. One important pattern is *South Facing Outdoors:* "People use open space if it is sunny, and do not use it if it isn't, in all but desert climates." Therefore, according to this pattern, one should "always place buildings to the north of the outdoor spaces that go with them, and keep the outdoor spaces to the south." Extensive observation of people's gathering habits in various situations demonstrates the truth of this pattern; by enunciating it, and putting it into context with other patterns, the authors provide both planners and users a concept which can guide design decisions. If an architect proposed a building with no southern exposure, a client or user could point to this pattern as support for an alternative design.

Another pattern, *Light on Two Sides of Every Room*, advises that "[w]hen they have a choice, people will always gravitate to those rooms which have light on two sides, and leave the rooms which are lit only from one side unused and empty.... Locate each room so that it has outdoor space outside it on at least two sides, and then place windows in these outdoor walls so that natural light falls into every room from more than one direction." This pattern accounts for why many offices feel so cramped or stultifying, because they rarely have light on two sides, and why corner or penthouse offices are so desirable.

Patterns languages have emerged in other technology domains as well. For example, computer interface designs pioneered by Xerox, Apple, and others make rich use of graphical metaphors such as windows, desktops, folders, files, buttons, and dialog boxes, as well as pointing devices such as a mouse.

This technology is based on a pattern language that differs dramatically from the one which characterized its predecessor, the command line interface. Principal patterns of this new interface include *visible options* (e.g., menu items and buttons), *direct manipulation* (clicking and dragging), *confirmation feedback* (menu blinking to confirm selection, button highlighting to indicate clicks), *visible modes* (use of graphical cursors to indicate long-term modes), and *facilitate exploration* (cancel buttons and an undo command).

The pattern language for the graphical user interface came about through detailed studies of how adults and young children organize and process information, and how they actually work. Xerox and Apple designers experimented with users to find out how they felt about the systems they were designing, and the design itself has gone through many versions.

Yet another area where implicit pattern languages have emerged is in the design of automotive transportation systems. There is a broad consensus on the patterns of "good" auto design: *speed*, *fuel economy*, *safe interiors* (padded dashboards, recessed switches, and safety glass), *compensation steering* (the tendency of a car to straighten itself out after being turned),[2] and *maximum visibility* (the increasing amount of glass and removal of visual obstacles in modern autos).

B. Properties of Pattern Languages

With this admittedly short discussion of pattern languages, we can point to several of their main properties. They are:

- **Dialectical:** They develop iteratively and reciprocally

2 Not all designs have this characteristic. Rear-wheel steering can provide high maneuverability, but also can be unstable at high speeds.

between technology developers and other stakeholders.[3]

- **User-centered**: Users are considered primary elements of the design from the outset, placing the language halfway between designers and users, rather than solely in the private domain of the designer. New information about user preferences and needs is not disregarded as irrelevant or wrong. The resulting technologies are shared designs, the result of many users and producers working together, over time, facing differing constraints, possessing different values.[4]
- **Comprehensible**: They articulate users' experiences within a language of design.
- **Explicit**: They are clearly articulated, and can be used by a variety of people, not just technology developers.
- **Consistent**: They express a consistent vision of a design. Individual patterns can, in some cases, conflict. But most should not. Fundamentally different visions should result in separate pattern languages.
- **Composable**: They are capable of complex and varied meaning construction. They do not simply state obvious design principles, but permit extended description and analysis of connections among and between different aspects of technical and organizational systems.

C. How Pattern Languages Can Be Used

Systematic analytic categories describing the social relationships around technologies, such as pattern languages, can be used to improve the technology design process. They can make technology producers more sensitive to users and society, improving their ability to create products that people want, and reducing the likelihood that the technology development will encounter social or political resistance. Similarly, pattern languages can improve the economic performance of technology-deploying firms, since the technology is more likely to match user preferences, ensuring better acceptance.

Pattern languages can also reduce negative technical surprises leading to social strain. With greater user and citizen involvement in design and operations, monitoring of a technology's social or environmental effects is likely to occur making it more likely that problems will be corrected.

Such languages can also be used to improve the functioning of democracy. If technologies can be considered constitutions, in that they authoritatively establish basic social arrangements,[5] then there must be some means to amend the constitutive technology. Dialogue about the design of important technologies among various stakeholders is in essence a process of deliberation. If the legitimacy of democratic institutions depends on the government's ability to protect citizens from harm and improve their welfare, then such processes will strengthen those democratic institutions.

[3] Single designers may also develop valid and useful pattern languages, but if they do so they have probably been attentive user needs and preferences.

[4] This is more difficult for technology developers than it sounds. For many, users are ignorant and difficult: to "idiot-proof" something is to make it suitable for the general public. At the other extreme, some technology designs force users to change their behavior as a condition of use, rather than adapting to it.

V. Component Designs

Pattern languages are one approach to bridging the gaps between designers and users. Another is to allow technologies themselves to become a means of discourse between these two groups. This is possible when technologies are designed as *components* — modules that can be recombined in ways that match the needs of particular users, and that are potentially unforeseen by the original designer.

Perhaps the most obvious examples of component designs involve consumer electronics, including technologies for video (televisions, video cassette recorders, video cameras, and remote controls), audio (amplifiers, tuners, speakers, and CD players), and computing (system unit, disk drives, monitors, keyboards, modems, and CD players). These technologies are often designed, produced, and used as components. The most common component designs are so natural and ingrained in our culture that they are nearly invisible; clothing, furniture, and replaceable parts in mechanical devices are all examples of component designs in common use.

Software is another excellent example of component design. Software has almost always used some form of component design and the trend is continuing. Standard programming constructs such as looping and iteration, as well as various data structures, are components provided to ease the task of programming. Individual software applications such as word processors and spreadsheets are components that allow users to customize a computer to their needs. More recently, object-oriented programming and component software technologies have recast the level at which software components might exist.

Component software technologies provide for much greater sharing of function than most of today's software. For example, OpenDoc, a standard supported by IBM, Apple, and many other vendors, allows users to share one component's function (e.g., spell-checking) across all other applications (word processing, email, graphics, etc.). Indeed, OpenDoc changes the very definition of an application, because many components can be used together to create a single seamless document. Other approaches to component software (e.g., Microsoft's OLE) offer greater integration of large applications. Still others (e.g., Sun's Java technology) offer a way to seamlessly

[5] Langdon Winner, *The Whale and the Reactor: A Search for Limits in an Age of High Technology*, Chicago: University of Chicago Press, 1986.

deliver software functions in large networked environments.

A. Components and Communication

Component designs arise for many reasons. In some cases, the needs of individual users are diverse enough that no unified design is feasible. In others, the components were originally conceived as separate units, and only combined later. Finally, component designs allow modules to be separated for easy repair and less expensive replacement.

However, component designs have another important purpose. They provide a powerful means of communication between designers and users. Although this effect is often unintended, components allow users, designers, and producers to participate in an implicit design collaboration. Users can add and remove individual components to meet their needs, and designers and producers can respond by producing components that meet the needs of particular market niches.

Component designs also allow participation by a broader array of designers and producers. Individual components can be designed and produced without the need to produce the entire technology. Thus, small firms and even individuals can produce specialized components that extend or enhance existing components. Examples include companies that produce enhanced remote controls rather than entire video systems, that specialize in speakers rather than entire audio systems, and that create operating system extensions rather than whole computer operating systems. This effect of encouraging greater participation is an explicit reason that some producers facilitate add-on components.

Component designs and pattern languages are similar, but complementary. Pattern languages provide a way of discussing design concepts. Components can provide a (limited) vocabulary for expressing designs themselves. Component designs occupy a middle ground on the design spectrum. They pose far fewer constraints than unified technologies, but they provide a much narrow range of choices than those possessed by the original designers. They provide a highly-constrained design environment, but one that still allows users to supplement and explore.

Despite the advantages of component designs, unified designs are sometimes quite successful. Examples include “all-in-one” computers (the original Macintosh), software (Microsoft Works and ClarisWorks), fast-food “meals” (sandwich, fries, and a soda), stereos, and even housing.[6] In some cases (e.g., software and stereos), these designs became popular only after the basic components had been tested as relatively separate units. In other cases (the original Macintosh computer and some Frank Lloyd Wright houses), the designers had a sufficiently compelling vision that users were willing to adopt it as a unit, without the flexibility provided by component design.

[6] All-in-one housing is rare, but still used. Furnished apartments are one example. Another example are some of Frank Lloyd Wright's houses — Wright designed built-in furniture (and even clothing) for the occupants.

This highlights an important advantage of component technologies. Even when designers and users eventually settle on a single, unified design, component designs allow an initial period of testing and experimentation. This period can be vital to allow users to experience both the potential and the drawbacks of the technology, and to allow them to modify both the components and their configuration. Clearly, this experimentation is constrained by the characteristics of both components and their interfaces, but substantial flexibility is often preserved.

A final advantage of component designs is the psychological effect they can have on users. Rather than leaving consumers with only a single choice — to either accept or reject the whole technology — they expose users to a broader array of choices. Once the leap to reconfiguring components has been made, it is often only a small step to ask “why can't this component itself be redesigned?” Component designs can create and reinforce a focus on design — one that empowers users and facilitates discussion between users and designers.

B. Creating Components

The advantages of component designs do not come without a cost. Unified designs offer advantages to designers, producers, and users. They can enhance reliability and lower prices, because designers and producers can be certain of the characteristics of what would otherwise be separate components. Users can benefit because unified designs eliminate potential compatibility problems and require less setup and maintenance.

Creating successful component designs imposes non-trivial demands on designers. One challenge is to define relatively standard interfaces between components. Such interfaces include physical connectors on mechanical parts and data standards and communication protocols for electronics. Without such standards, individual components are not interchangeable, and the promise of component designs is eliminated. Such interfaces can often be designed without deep knowledge of other components. The recently introduced BeBox computer has a “GeekPort” — a data interface with undetermined future uses. The designers wanted to facilitate innovation, even though they did not know what those innovations would be. Similarly, some software applications have programmer interfaces that can be designed without knowledge of what other components will be used in a complete system.

Another design challenge is to define the right level of “granularity” among the components. Considerations such as expected lifecycle, user preferences, cost, and safety must be balanced to find a “natural” level at which to divide components. Much of the sophistication of component design lies in defining both the appropriate granularity and the appropriate dividing lines between components. In some cases, "natural" dividing lines appear where components have minimum interaction and convenient size, but such occasions are not universal.

Even with these challenges, however, component designs can be an elegant and successful way to explore the interactions between particular technologies and the societies that they become embedded within. They offer choice for users, an implicit communication channel between users and designers, and flexibility during a technology's introduction — perhaps the phase that most determines its later social effects.

VI. Issues and Challenges

Taken together, the two conceptual contributions of this paper — pattern languages and component designs — can facilitate better communication between designers and users. We believe that such communication is vital to avoiding the controversies and high costs that have attended the introduction of many new technologies in the latter half of this century.

However, improving communication between these disparate groups is not a simple matter. There are both general and specific challenges to applying the approaches we outline above.

A. Breadth of Application

Pattern languages and configurable technologies are most useful for those technologies that are directly applied by users. Their utility is less obvious in cases where users play no significant role in the direct application of the technology (e.g., agricultural chemicals, medical technologies, and nuclear power). Indeed, this very characteristic is one of the most troubling aspects of these cases: users have little direct choice. Such situations can leave citizens feeling disenfranchised and powerless.

How might pattern languages and configurable design be applied to these cases? Pattern languages can be useful even if the technology is beyond the scope of a single user. For example, Alexander and associates' original works on pattern languages included many patterns devoted to urban and regional planning. Pattern languages can also be useful even if citizens have little or no direct contact with the technology in question. For example, patterns such as *short-lived waste products*, *independence from imported fuels*, *minimal land use*, and *reliability* all apply to electricity generation facilities. In addition, a variety of component design can be employed even when the technologies themselves cannot be modified by users. For example, consumers cannot directly determine how agricultural goods are produced, but in many markets consumers can still "configure" a diet of pesticide-free, hormone-free foods if they so desire. Similarly, some healthcare providers allow a wide range of treatment options to patients, allowing them limited latitude to configure healthcare technologies.

B. Limited User Experience

As already mentioned, one problem with new technologies is that they are *new*. Users have no prior experience with them, and thus have little basis on which to form preferences. They gain the requisite experience only after the technology has been deployed, long after the design is determined. Given their lack of experience, how can users productively communicate with designers during the design stage?

Pattern languages offer some help. Pattern languages define user preferences at a relatively high level, providing a way for users to abstract and generalize across similar technologies. For example, if exploring pattern languages for electricity generation systems leads an individual to the pattern *independent of foreign fuel sources*, then this pattern can inform their thinking about a range of existing and novel energy sources.

Component designs also offer some help. Component designs allow users to gain experience with technologies in a highly flexible and reconfigurable way. As noted earlier, this allows experimentation and limited redesign at a very low cost.

C. Economic Forces

Another challenge is that economic forces often militate against using component designs. Unified designs make it more difficult for users to switch technologies and producers, increasing the incentive to produce a unified design. For users, the costs of switching to a new unified design generally will be higher than merely switching a single component. In addition, unified designs can provide economies of scale. A producer can be certain of getting *all* of a user's business if they sell a unified design. These arguments have been used to explain the growing size of some software packages for word processing and Internet access. The more functions that a software vendor can bundle into a single package, the higher the switching cost will be for users of that product.

Part of this motivation for unified designs probably cannot (and should not) be overcome. As already noted, unified designs can provide users with benefits. However, strong industry standards and the "design awareness" fostered by pattern languages and component designs help provide a counterbalance to the economic motivations supporting unified designs.

VII. Conclusions

Technology developers often implicitly, and less often explicitly, use theories of organized social behavior or process to inform the development of technical artifacts that they make and use that ultimately become part of the fabric of the larger society. This embodiment of values in the design of technologies has important consequences: getting the design "right" is vital to successful deployment of the technical system, and to its beneficial use in society.

However, the values embodied in technologies, and the oft-accompanying requirements that individuals and organizations conform to these built-in norms may conflict with the values by which people wish to live. This is a principle cause of discomfort and opposition to some technologies, out of which political movements may even emerge.

So, the question is, how willing are we as system developers and designers to identify the values that should inform technology development and system building? If there is willingness, is there analytic capacity to do so? Is there latitude in technology design that could provide space for competing values? Given the immense costs of developing and deploying new systems, it makes both democratic and commercial sense to think about these impacts of technology early on.

Pattern languages and configurable design concepts are two complementary approaches which may help technology developers, technology users, and other stakeholders to bridge the communication gulf, and to achieve a better understanding of each other's values and needs. Development of systematic and explicit design concepts such as pattern languages across the range of technologies and technical systems, particularly infrastructural or large-scale technologies, or technologies where standards play a major role in development or use, would lead to a significant improvement in the analytical foundation of both the practice of technology design and of the politics of technology. We believe that such an undertaking by the technology and society research community is essential to improve the foundation for meaningful dialogue among all those with an interest in the outcome of technology developments.

VIII. Acknowledgements

All registered and unregistered trademarks mentioned are the sole property of their respective companies.

David D. Jensen is Research Assistant Professor of Computer Science at the University of Massachusetts at Amherst where he is a member of the Experimental Knowledge Systems Laboratory. From 1991 to 1995, he was an analyst with the Office of Technology Assessment (OTA), an analytical support agency of the United States Congress. While at OTA, he contributed to studies on quantitative decisionmaking techniques and artificial intelligence technologies. He received his D.Sc. from Washington University in St. Louis in 1992.

Todd M. La Porte will be Associate Professor in the School of Systems Engineering, Policy Analysis and Management at the Delft University of Technology in the Netherlands, where he will be a member of the Information and Communications Technologies Program. From 1989 to 1995 he was an analyst with the Office of Technology Assessment (OTA), an analytical support agency of the United States Congress. While at OTA, he contributed to studies on international collaboration in production and trade in conventional arms, on international telecommunications trade in services, and on wireless telecommunications technologies. He received his Ph.D. in political science from Yale University in 1989.