Management Science

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Economics of Product Development by Users: The Impact of "Sticky" Local Information

Eric von Hippel,

To cite this article:
Eric von Hippel, (1998) Economics of Product Development by Users: The Impact of "Sticky" Local Information. Management Science 44(5):629-644. https://doi.org/10.1287/mnsc.44.5.629

Economics of Product Development by Users: The Impact of "Sticky" Local Information

Eric von Hippel
Sloan School of Management, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139

Those who solve more of a given type of problem tend to get better at it—which suggests that problems of any given type should be brought to specialists for a solution. However, in this paper we argue that agency-related costs and information transfer costs ("sticky" local information) will tend drive the locus of problem-solving in the opposite direction—away from problem-solving by specialist suppliers, and towards those who directly benefit from a solution and who have difficult-to-transfer local information about a particular application being solved, such as the direct users of a product or service. We examine the actual location of design activities in two fields in which custom products are produced by "mass-customization" methods: application-specific integrated circuits (ASICs) and computer telephony integration (CTI) systems. In both, we find that users rather than suppliers are the actual designers of the application-specific portion of the product types examined. We offer anecdotal evidence that the pattern of user-based customization we have documented in these two fields is in fact quite general, and we discuss implications for research and practice.
(*User Innovation*; *Sticky Information*; *Local Information*; *Heterogeneous Markets*; *Mass Customization*; *Specialization in Problem Solving*; *Task Partitioning*)

1. Problem Statement and Overview

Providers of software often note that their products "empower users" to develop innovative solutions to their own problems. Yet, given the supposed benefits of specialization, one might legitimately ask why anyone, including users, would find this to be a good idea. Does one really want to be one's own architect or one's own doctor? In this paper, we first consider why it might indeed be attractive under some conditions to allocate the *application-specific portion* of the problem-solving work of custom product and service design to users rather than to specialist suppliers (§1). We then set the context for our empirical study of this question (§2) and explore the locus of design-related problem-solving in two industries devoted to the production of "mass customized" products and services (§§3 and 4). We find that in both, the application-specific portion of the problem-solving work of product customization is carried out by users, with the aid of standard toolkits and components provided to them by supplier firms. Finally, we discuss our findings and consider whether and when this form of partitioning of the product design process between user and supplier might be a generally attractive way to organize the innovation process (§5).

In production, an important benefit of specialization can be seen in the learning curve. Beginning with Wright (1936) a number of studies have shown that the unit cost of producing a given type of manufactured good tends to decline significantly as more are produced. It has been argued that this effect is the result of the development of increasing skill in production attained by what Arrow (1962) has termed "learning by doing." In problem-solving also, the ability to solve a given type of problem has been shown to improve with practice. Studies of problem-solving expertise show that it contains elements such as an inventory of knowledge regarding solutions that "work" for the problem type at issue, and a repertory of problem-solving skills such as a facility at decomposing a new problem into

0025-1909/98/4405/0629$05.00

subproblems that are similar to previously solved ones. The net effect is that problem-solvers with expertise in problems of a given type are able to solve that type of problem much more rapidly than can novices (Larkin et al. 1980). Thus, economies of specialization would tend to reward bringing all problems of any given type to appropriate specialists for a solution.

However, we propose that at least two important factors will tend drive the locus of problem-solving in the opposite direction—away from problem-solving by specialist suppliers, and towards those who directly benefit from a solution such as the direct users of a product or service. The first factor is generally understood, and involves various kinds of agency-related costs that might drive direct beneficiaries of a new product or service design to ''do it themselves.'' For example, direct beneficiaries will be motivated to create a solution that will be exactly right for their own very particular circumstances. In contrast, supplier agents may have an incentive to create solutions that are ''good enough'' for a wider range of potential users.

The second factor is less well understood and involves the impact of what we call ''sticky'' local information on the locus of problem-solving. Consider that to solve a problem, needed information and problem-solving capabilities (also a form of information) must be brought together at a single locus. The requirement to transfer information from its point of origin to a specified problem-solving site will not affect the locus of problem-solving activity when that information can be shifted at no or little cost. However, when it is costly to transfer from one site to another in useable form—is, in our terms, sticky—the distribution of problem-solving activities can be significantly affected.

We define the stickiness of a given unit of information in a given instance as the incremental expenditure required to transfer that unit of information to a specified locus in a form useable by a given information seeker. When this cost is low, information stickiness is low; when it is high, stickiness is high (von Hippel 1994). A number of researchers have both argued and shown that information required by technical problem-solvers is indeed often costly to transfer for a range of reasons. Information stickiness may be due to attributes of the information itself such as the way it is encoded (Nelson 1982, 1990; Pavitt 1987; Rosenberg 1982); and/or it may be due to attributes of information seekers or providers. For example, a particular information seeker may be less able in acquiring information because of a lack of certain tools or complementary information—a lack of ''absorptive capacity'' in the terminology of Cohen and Levinthal (1990). Also, specialized personnel such as ''technological gatekeepers'' (Katz and Allen 1982, Katz and Tushman 1980) and specialized organizational structures such as transfer groups (Katz and Allen 1988) can significantly affect the information transfer costs between and within organizations. Recently, Szulanski (1996) explored the relative importance of many of these factors in a study of transfers of information associated with 38 ''best practices'' within firms. He found that the three largest contributors to information stickiness in that sample were a lack of absorptive capacity by the recipient, incomplete or poorly encoded information, and a laborious and distant relationship between the information source and recipient.

The link between information stickiness and the locus of problem-solving activities involves two elements. First, the stickiness of a given unit of information is not immutable. Rather, it can be reduced by investments made to that end. For example, firms may reduce the stickiness of a critical form of technical expertise by investing in converting some of that expertise from tacit knowledge to the more explicit and easily transferable form of a software ''expert system'' (Davis 1986). And/or they may invest in reducing the stickiness of information of interest to a particular group of users by encoding it in the form of a remotely accessible computer database. This is what the travel industry did, for example, when it invested substantial sums to put its various databases for airline schedules, hotel reservations, and car rentals ''on-line'' in a user-accessible form.

Second, an investment in unsticking a unit of information is a one-time investment that reduces the marginal cost of all succeeding transfers of that information. Therefore, the incentive to invest in reducing the stickiness of a given unit of information will vary *according to the number of times that one expects to transfer it*. As illustration, suppose that to solve a particular problem, two units of equally sticky local information are required: one from a user and one from a supplier. In that case, there will be an equal incentive operating to unstick either of these units of information in order to re-

duce the cost of transfer, other things (such as the cost of unsticking) being equal. But now suppose that there is reason to expect that one of the units of information, say the supplier's, will be a candidate for transfer n times in the future, while the user's unit of information will be of interest to problem solvers only once. For example, suppose that a supplier expects to have the same technical information called on repeatedly to solve n user product application problems, and that each such problem involves unique user information. In that case, the total incentive to unstick the supplier's information across the entire series of user problems is n times higher than the incentive for an individual user to unstick its problem-related information.

In the case of the problem-solving work of product and service development, the situation just described is the one often encountered in the real world. Manufacturers do tend to specialize in a given solution type, which they attempt to apply to the diverse application problems of many users. As we will see later, the local information required from a supplier to solve each novel application problem tends to be the same, while the local information required from the user tends to be novel or have novel components. Under such conditions, and for the reasons just described, we expect that sticky information transfer cost considerations will create an incentive to shift the locus of problem-solving activity to the locus of the less frequently called-upon information—in the case of our example, to the user.

In sum then, we propose that allocation of the *application-specific portion* of the problem-solving work of custom product and service design to users will be economically attractive for a supplier when: (1) the supplier faces heterogeneous demand for a given type of product or service (that is, many of the users served place a high value on custom solutions); (2) agency costs experienced by users who outsource design activities are high; (3) the stickiness of application-specific user information is high; and (4) the stickiness of information held by suppliers that is relevant to application-specific problem-solving is low.

2. Context for Empirical Inquiry: ''Mass Customized'' Products

We have elected to explore the proposals just described by examining the locus of product design activities in two industries devoted to the production of ''mass customized'' products. The first is the application-specific integrated circuit (ASIC) industry. This industry is of substantial size and growing rapidly, with worldwide sales in 1994 of $13.5 billion—about 15 percent of worldwide IC sales (McClean 1995, figures 4-13, 4-15). The second industry chosen for study is the computer telephony integration (CTI) industry. Also growing rapidly, this industry focuses on business computing systems that draw upon both computing and telephony functions to accomplish a task. The ordering of goods from a mail order firm via telephone is an example of a task typically accomplished with the aid of a CTI system today. Sales of CTI systems were about $1.2 billion in 1995 (Multimedia Telecommunications Association 1996).

These two industries were chosen as appropriate for our present purpose because both focus on the production of customized products rather than standard ones, and because both are of substantial size, with many custom product variations being designed and produced. As a consequence, we expected that industry participants would have invested significant resources to develop efficient methods for designing mass customized products. Data were collected on each of these industries through specialized literature as cited, and also via extensive semistructured interviews held with approximately 100 experts in user and supplier firms in the ASIC and CTI industries. Experts initially selected for interview were those mentioned in books and specialized trade literature as having played important roles in the technical or business aspects of the partitioning of product design activities between users and suppliers in their respective industries. Later interviewees were those identified by early interviewees as also very knowledgeable on those matters.

Mass customization generally refers to the manufacture of one-of-a-kind, ''custom'' products via the use of flexible, computer-controlled mass-production machinery. Historically, individual products built to the specifications of a particular customer were made by using handwork-intensive processes—and were quite expensive. In contrast, many identical products could be ''mass produced'' on specialized production machinery at a much lower cost per unit. Relatively recently, the introduction of computer-instructed process equipment

has opened the way to producing one-of-a kind products at mass production prices. Such equipment follows software instructions and can be instantly adjusted to new specifications for production of each unit in a production run. As a consequence, each item in that run can be unique—''mass customized'' to the specifications of a particular customer (Pine 1993). One can also logically extend the concept of mass customization to the production of customized services. In that case, software-based instructions are used to instruct a service delivery system—say, an automated home banking system—rather than computerized production machinery.

Mass customization offers value when the demand for a final or intermediate good or service is heterogeneous. This is often the case. Many consumers would like goods and services, ranging from their clothing to their houses to their telephone answering services, to be in some way different from standard offerings. Similarly, engineers, who specify intermediate goods as components for more complex products that they are designing, often display heterogeneous needs within even quite narrow product categories. For example, a starter motor or engine controller chip that is well-suited for an auto engine of design A will not be quite right for many other engine designs.

The question we will explore in the case studies that follow is: Who *designs* the customized portion of the products that are built using mass customization production methods? The work of design involves collecting information on the unique needs of a customer; use of that information to create a customized product design; and conversion of the design information into a form suitable for driving a given manufacturer's computerized production machinery or service delivery system. In the case of mass customized production, manufacturers of a particular type of mass customized product and/or specialist design services are the locus of expertise with respect to problems common to many designs. In contrast, users or system designers possess the deepest understanding of a particular application. For example, a book publisher will be the locus of expertise as to how to edit, design, print, and distribute books. The author, on the other hand, will be the expert regarding the detailed content of his or her specific book.

In the cases we will study, single suppliers of ''mass customized'' products or services are in the business of applying a general solution to the unique needs of many users. Under these conditions, for reasons discussed earlier, we anticipate that we may find a user rather than a supplier locus for application-specific problem-solving. And, if this proves to be the case, we would expect to see problem-solving tools and components that are generally useful in solving *X*-type design problems being transferred from suppliers to user-based problem-solvers. What kinds of ''generally useful'' information do we expect to see being transferred from suppliers to users in such a case? First, we expect to see information being transferred regarding the *constraints* of the supplier's mass customization process. Second, as just noted, we expect to see *standard* tools and components being supplied to users that can help them in their *application-specific* design activities.

With respect to production process constraints, note that the economies of mass customized production are only achievable if and as a custom design falls within the preexisting capability and degrees of freedom built into a given mass customization system. We may term this the ''solution space'' offered by the system. For example, the solution space offered by a book publisher will typically enable the author to use any words he or she likes, arranged in any sequence. At the same time however, the solution space may only allow an author to specify those special symbols (say, mathematical symbols) that are ''in stock'' at the printer. Similarly, the solution space offered by an integrated circuit manufacturer may allow a customer a range of variation with respect to the size of silicon chip to be produced, the density with which electronic devices can be placed upon it, etc. However, it will also state limits on these variables. For example, ''Designers may only specify chip sizes no larger than *X* and no smaller than *Y*.'' The reason the manufacturer enforces such constraints is that the economies of mass customization require that a custom user design be implementable simply by making low-cost adjustments to the production process. This condition is met within the solution space on offer. However, responding to requests that fall outside of that space may require small or large additional investments by the manufacturer. For example, it may be relatively inexpensive for a printer to add an additional set

of symbols to his stock. On the other hand, an integrated circuit producer may have to invest many millions of dollars and rework an entire production process in order to respond to a customer request for a larger chip.

Further insight into the nature of the collection of tools and components that solvers of a particular problem of type *X* will find useful can be derived from the nature of problem-solving work. Research into problem-solving in general shows it to consist of trial and error, directed by some amount of insight as to the direction in which a solution might lie (Barron 1988, pp. 43–47). This finding is supported by empirical studies of problem-solving in the specific arena of product and process development (Marples 1961, Allen 1966, Habermeier 1990). Such studies do show trial and error (or, more precisely, trial, failure, learning, revision, and retrial) as a prominent feature. One may view the trial-and-error process as consisting of a four-step cycle: (1) one conceives of or designs an experiment; (2) one builds the (physical or virtual) apparatus needed to conduct that experiment; (3) one runs the experiment; (4) one analyzes the result. For example, one might (1) conceive of and design a new, more efficient air conditioner for a car; (2) build a prototype of key elements of that air conditioner as well as any special apparatus needed to test its efficiency of operation; (3) run the experiment to determine actual efficiency; and (4) analyze and learn from the result. If the results of a first experiment are satisfactory, one stops after step (4). If, however, as is usually the case, analysis shows that the results of the initial experiment are not satisfactory, one may apply what one has learned to modify one's experiment and then ''iterate''—try again.

To develop their custom design, developers will find it useful to have access to standard component parts and standard design tools that will help them to carry out the trial-and-error cycle of problem-solving work. Thus, a team of architects who are designing a custom office building will find it very useful to have access to a library of standard components, for example, a range of standard structural support columns with preanalyzed structural characteristics, that they can incorporate into their building design. They would also find it useful to have tools such as a structural analysis program that can help them to conduct trials of their evolving custom design to determine, for example, whether that design will be structurally safe. Similarly, users who are designing a document with the aid of a desktop publishing system will find it useful to have standard formats and standard ''clip art'' illustrations that they may choose to incorporate into their custom design. They will also value having a system capable of ''building'' a prototype of their design in the form of a simulation on their computer screen, as an aid to evaluating that design's fitness to their intended purpose.

3. Application-Specific Integrated Circuits (ASICs)

Application-specific integrated circuits (commonly referred to as ASICs) are integrated circuits that are designed and built for a specific application, and for a specific customer. For example, a maker of compact disk players—or autos or machine tools or dishwashers—might specify an ASIC to perform some or all of the product-specific electronic functions required in his unique design. In contrast, application-specific standard products (ASSPs) are integrated circuits that have a specific or narrow range of application, but that are developed for multiple users. For example, a ''chipset'' developed to implement major functions specific to personal computers and sold to many PC manufacturers is an ASSP. Also in contrast, a standard integrated circuit is one with a function useful in a wide array of applications that is sold to many customers. Examples are memory chips, flip-flops, and microprocessors (McClean 1995, p. 4-1).

Integrated circuits in general, and ASICs in particular, are generally built upon the surface of a thin, flat wafer of silicon crystal by a process involving deposition of successive very thin layers of semiconducting and insulating materials in very precise patterns. Electronic components such as transistors and capacitors are formed via this process, and are then interconnected into a functioning circuit via the deposition of very thin lines of metal that serve as a form of electrical wiring. Integrated circuit components can be built using a number of different ''technologies'' such as bipolar and metal oxide semiconductor (MOS). Components of similar electronic function are designed differently in each of these technologies, and can have somewhat different

characteristics, such as lower power consumption or higher switching speeds.

In the early days of the integrated circuit industry, custom electronic circuitry was built by selecting a number of standard integrated circuits and other standard electronic components, and then connecting them together in a customized way on a printed circuit board. However, important technical considerations (reliability of interconnections, circuit speeds) and economic considerations (a potential for significantly lower manufacturing costs) provided strong incentives to move circuit customization down into the integrated circuit itself by creating customized ICs. ASICs are the realization of this goal. The earliest adopters of ASICs were manufacturers of high-speed computers, manufacturers of military equipment, and manufacturers of telecommunications equipment who had no choice but to utilize ASICs to meet their performance goals. Today, ASICs can be designed and built quite quickly, and so are also used by firms wishing to reduce development times for their products (Hilbert 1991, p. 4).

ASICs were first introduced in the late 1960s and early 1970s and have greatly increased in size and complexity since that time. In the late 1960s an ASIC with 100 ''gates'' (a basic digital logic function requiring a few electronic components to implement) was considered large. Today, an ASIC with 10,000 gates is considered small, an ASIC with 500,000 gates is considered large, and an ASIC with 1.3 million useable gates (such as one recently developed by IBM) is considered to be leading edge. A leading-edge ASIC is a very complex semiconductor product, and various of the means used to design and manufacture complex ASICs are considered to be at the leading edge of IC industry practice. These include design and simulation software, testing methods, and flexible manufacturing processes (McClean 1995, pp. 4-8 and 4-9).

Shift of ASIC Design Activities to Users and System Designers

The original method used to design integrated circuits is the so-called ''full-custom method.'' This method involves designing each transistor and interconnection on an integrated circuit ''from scratch,'' a relatively slow and expensive procedure. Nonetheless, it is still the method of choice today when an integrated circuit must perform at the very highest speed attainable with a given production process, and/or must be squeezed onto the smallest possible area of silicon so that it can be manufactured at the very lowest cost per unit. For example, leading-edge memory chips and microprocessors are generally designed using full-custom design methods.

Full customization enables designers to achieve high performance because it gives them the freedom to mutually adjust each circuit element and the demands that will be placed upon it. For example, if a circuit design requires a particular group of transistors to run especially rapidly, an engineer using full customization can modify the physical design of those particular transistors accordingly. And/or, the engineer can modify the design of the circuit to lessen the demands placed upon those transistors. This ability to make interdependent choices regarding physical device design and circuit design is the strength of full customization. However, this characteristic also represents a barrier to shifting full-custom ASIC design work to user sites, should anyone wish to do so. Design engineers working for firms that incorporate customized ASICs in the products they are developing to use or to sell were and are typically electrical engineers who understand circuit design but do *not* understand semiconductor device design. They would not find it easy to design ASICs via full customization methods without extensive specialized training.

From this initial state of technological affairs, two trends in the industry have combined to ''unstick'' the supplier-based information required by an engineer wishing to design customized ASIC chips to incorporate in a system he or she is designing. The first was the development of new ASIC architectures that reduced the amount of specialized, supplier-generated information that a designer must know to be able to design a custom ASIC circuit. The second was the encoding of the remaining information required by a circuit designer into easy-to-use software toolkits.

The reduction of the amount of supplier-based information required by an ASIC chip designer was achieved via the development of three new ASIC architectures that enabled designers to design a circuit *without* having to understand the physical design of semiconductor devices. These three architectures are gate arrays, standard cell ASICs, and field programmable logic devices (PLDs).

• Gate array ASICs are based on standard, ''semifinished'' chips that are then customized into finished ASICs. Specifically, in this approach an ASIC chip is designed and fabricated in a standard manner up to the point where the basic elements used in the circuit—an ''array'' of logic gates—have been completely fabricated but have not yet been interconnected into a functioning circuit. Custom circuit function is then achieved by designing and then fabricating one or two final interconnection layers that interlink these standard circuit elements into a special-purpose ASIC.

• Standard cell ASICs are designed from predesigned and pretested circuit modules contained in a ''library'' made available to circuit designers. Individual modules available in such a library (called cells or macrocells depending on size) range from analog to digital converters to complete microprocessors. To develop a standard cell ASIC, a designer draws predesigned cells from the library as needed and specifies how they should be interconnected to achieve the desired custom circuit functionality.

• Field programmable logic devices are built as standard semifinished chips that are then customized into a finished ASIC. As with gate arrays, these ''semifinished'' chips contain completely fabricated circuit elements. However, in field programmable logic devices, these elements emerge from the fabrication process totally interconnected via fusible conducting links, and are shipped to the customer in this nonfunctional form. The circuit designer then converts the chip to a customized functioning ASIC in the field by ''programming'' it using a desktop encoding device driven by a personal computer. This device applies precisely programmed electric pulses to the chip to melt and thereby eliminate all but the desired circuit interconnections. (Other versions of field programmable logic devices use an ''antifuse'' technology that allows users to selectively *create* desired connections.)

Each of the three chip architectures just described involves building custom ASICs from combinations of physical devices that have been predesigned by manufacturer experts. Each physical device is described to the ASIC circuit designer in terms of its logical functioning in a digital circuit rather than in terms of its physical nature. This, in turn, allows a circuit engineer to design an ASIC having a desired customized function simply by selecting and interconnecting digital logic elements. No understanding of the physical devices themselves is required of that designer.

In addition, gate array, standard cell, and field programmable ASIC chips can all be designed much more quickly and cheaply than fully customized chips (see Table 1). This is because, in contrast with full-custom design, the physical devices incorporated on the chips have all been predesigned and pretested in these architectures. A field programmable chip (PLD) is generally the ASIC technology of choice when the number of chips required is low, and when it is important to have functioning chips very quickly. As production volumes rise, the least costly choice becomes successively gate

Table 1 Attributes Affecting Customer Choices Among ASIC Technologies

ASIC Type	Full Custom	Standard Cell	Gate Array	PLDs
Time to Design ASIC	52–104 weeks	12–52 weeks	4–26 weeks	<2 weeks
Time to Build Prototype	8–12 weeks	6–10 weeks	1–3 weeks	<10 minutes
Typical Development Fees Charged by Supplier	$50K–500K	$20K–200K	$10K–100K	$0
Maximum Density of Gates on Chip/cm*	<350K	<250K	<100K	<10K
Unit Manufacturing Cost	Lowest	Medium	High	Highest

* Interviewees report that many of the time and cost figures shown here have been reduced significantly since 1991. (Thus, most full-custom ASICs can now be designed in a year or less.) However, the relative position of the various types of ASICs with respect to cost and time expenditures is unchanged.

Source: Chakravarty (1991, Table 1, p. 31).

arrays, standard cell designs, and full-custom designs—although these options also involve progressively longer and more costly design work. As can be seen in Table 2, there has been a major shift over time from full-custom ASICs to gate array, standard cell, and PLD ASICs.

The unsticking of the remaining supplier-based information required to design an ASIC circuit has been achieved by embodying it in software-based design tools that can be economically transferred from suppliers to circuit designers (Mathur 1996, pp. 1–6). In the very early days of the industry, these computer-aided design tools were developed by ASIC vendors only for the use of their own, in-house designers. Fairchild Semiconductor was a pioneer in this field in 1967–1970, and was followed by other major manufacturers and major manufacturer-users such as IBM (Walker and Tersini 1992, Chapter 2). However, in 1980–1981 the founders of LSI Logic, a startup manufacturer of custom ASICs, changed the pattern. They developed a suite of ASIC design tools and an extensive library of cell designs and made them available to their customers, so that *customer* engineers could design the ASICs that LSI would then manufacture.

The advantages ASIC manufacturers might gain by switching from a manufacturer-based design model to a user-based design model for ASIC design were initially not clear to other ASICs manufacturers with whom LSI founders discussed their planned approach. Thus, Wilf Corrigan, a founder of LSI Logic, reports the following conversation. ''When I talked to Yasufuku [a senior manager] at Fujitsu and told him that our plan was to put the software in the hands of the customers, he said, 'That is a brilliant strategy. If you do that and the software is good, you will win.' 'Why don't you do that?' I asked. 'Our software is so valuable that if we expose it to outsiders they will steal it.' In fact, [Fujitsu] had been unwilling to transfer the software even to their U.S. subsidiary because they were convinced that once they let the genie out of the bottle, they would never get it back in again.'' However, LSI Logic's idea was found to be strongly preferred by ASIC customers, and eventually other ASIC manufacturers and independent vendors of ASIC design software were driven to follow LSI Logic's lead (Walker and Tersini 1992, p. 80).

Initial CAD tools developed by LSI and other suppliers for customer use were not very user friendly in the sense that they took a lot of programming skill and specialized expertise to operate. As a consequence, design centers were established by ASIC manufacturers and also by independent entrepreneurs who would buy a set of tools from one or more manufacturers and develop the expertise to use them well. Engineers from ASIC customer firms that did not have the tools and/or the expertise would go to a design center and get help with implementing their designs on the specialized design software. Today, greater user experience and more user-friendly ASIC design software tools enable user engineers with ordinary skill to design ASICs entirely on their own.

A schematic overview of the sequence of problem-solving tasks typically undertaken by ASIC designers, and the functions of the major tools they use today is as follows. ASIC designers begin by creating a functional description of the circuit they desire and enter that information on a software design tool. This tool converts

Table 2 Change in Market Shares of Four ASIC Types Over Time

ASIC Type	Full Custom	Standard Cell	Gate Array*	PLDs	Total Market Size
	(%)	(%)	(%)	(%)	(Billion)
Market Share					
1986	52%	11%	30%	7%	$4.7
1994	20%	30%	40%	10%	$13.5
1999**	12%	40%	38%	10%	$23.6

* Includes linear arrays.

** Industry estimates.

Data source: McClean (1995, Figure 4–13).

the information provided by the designer into a description of a network of interconnected logic elements that will provide the function specified. This design software contains information on the nature and limits of the solution space made available by a given ASIC manufacturer. For example, it will model the circuit in terms of logic functions that can actually be delivered by the types of ASIC components manufactured by that manufacturer. Next, users ''run'' the model of their circuit design on a simulation tool. Any errors in the design logic will cause the ASIC simulation to not perform as intended, and the designer will use these results to detect and then correct such errors using capabilities contained in the simulation and design tools. Multiple run, diagnose, and repair cycles are typically needed before all or most of these errors have been eliminated (Thomke 1996).

Next, the designer transfers the corrected logical description of his circuit to other software tools that are generally located at the specific vendor selected by the designer to build the ASIC. This tool actually ''lays out'' the physical geometry of the cells and interconnections of the actual ASIC chip in a manner that is compatible with that manufacturer's production process. Information from this step is sent back to the designer who resimulates the design looking for and correcting any errors (typically, so-called ''timing errors'') created by the conversion of the ASIC design from a symbolic representation into the design of an actual physical chip to be produced by a specific vendor and process. The revised design is then sent back to the manufacturer and is used to drive the computerized equipment at the fabrication plant that creates the customized mask used to produce the finished ASICs (Haskard 1990, pp. 77–80).

Taken together, the development of the new ASIC architectures and the development of software design toolkits have unstuck the manufacturer-based knowledge required to design a custom ASIC. This unsticking has been accompanied by a general shift of the application-specific portion of custom ASIC design activities from manufacturer-based designers to user-based designers. Recall that this shift is what we would predict when the advantages associated with greater manufacturer expertise in customized ASIC chip designing are outweighed by the costs of transferring application-specific sticky information from user to manufacturer, other things being equal.

Data on industry structure in the ASICs field support the view that manufacturers should have significantly greater incentive to unstick and transfer manufacturer-related information needed by every ASIC designer than would users to unstick and transfer information related to a specific application. The top four ASIC manufacturers and their 1994 market shares were NEC with 9.3 percent of ASIC sales volume, Fujitsu with 9.0 percent, Toshiba with 7.5 percent, and LSI Logic with 7.2 percent (McClean 1995, Figure 4-22). Each of these vendors manufactures thousands of custom ASIC designs yearly. The top three ASIC design tool suppliers and their approximate 1996 market shares were Cadence Design Systems with 40 percent of sales volume, Mentor Graphics with 20 percent, and Synopsys with 20 percent. These firms sell hundreds of software packages per year, each presumably used for a number of design projects per year.

4. Computer Telephony Integration (CTI) Systems

Computer telephony integration (CTI) refers to a field of specialized computing applications that draw upon both computing and telephony functions to accomplish a task. The ordering of goods from a mail order firm via telephone is an example of a task typically accomplished with the aid of a CTI system today; the management of one's stock brokerage account from home by pressing digits on a home telephone handset is another. Users typically require customized CTI systems because these must be closely integrated with user firm's—typically nonstandard—business practices and computerized business systems.

The first commercially supplied CTI systems were developed in the late 1960s and were capable of integrating computing and telephony in relatively simple applications only. Thus, in the late 1960s IBM supplied a custom CTI system to a book store chain that wished to transmit book orders from branch store computers to a supply house computer via telephone (Walters 1993, p. 25). During the same period other pioneer CTI suppliers such as Collins Radio, Rockwell International, and Datapoint commercialized ''automatic call director'' sys-

tems that were used by telephone sales centers in airline and car rental and hotel firms to queue incoming calls and allocate them to available sales agents in a systematic manner.

From these early beginnings, CTI applications have steadily grown in sophistication and complexity, benefiting both from an improved understanding of CTI application possibilities by both users and suppliers, and from major improvements in the power and sophistication of computer hardware and software available to build the systems. Current systems can assist in or carry out complex transactions which may involve multiple interactions among computing and telephony systems. Consider, for example, the capabilities of a current CTI system used by a stock brokerage firm. When a brokerage customer contacts the system by telephone, the system can offer that customer near instant access to a range of information upon request (e.g., account balance, stock performance), even though it might have to retrieve various of the requested items from a number of computer data bases located in different sites. The system can also execute and document complex transactions. If, for example, the customer decides to buy a stock, the system can switch the telephone call to the appropriate stock trader, aid in implementing the trade, and then properly update customer records to reflect the trade, transfer funds among accounts appropriately, etc.

Shift of CTI Design Activities to System Users

From the beginnings of the field until about five years ago, a CTI system consisted of two major subsystems: (1) CTI server software; (2) customer-specific application programs used to implement a customized CTI system. CTI server software aids application developers in two major ways. First, it implements an application programming interface (API) that ''virtualizes the underlying [telephone] network so that applications can perform meaningful work without needing to be tailored to the specific behaviors of the switch or other equipment'' (Nixon 1996, p. 44). Early application programming interfaces were designed to interface with the equipment of a single telephone switch manufacturer only, and often were implemented by means of special-purpose computing hardware. More recent APIs, such as Microsoft's TAPI and Novell's TSAPI, are designed to interface properly with telephony equipment supplied by many manufacturers, and have been incorporated into CTI servers that run on ordinary personal computers. Second, CTI server software enables applications developers using a programming language such as 'C' to incorporate basic telephony functions such as ''answer phone'' or ''transfer call'' in their programs in the same way that they incorporate traditional computing functions such as ''add'' or ''create a file.'' However, CTI servers do *not* modularize higher-level functions (for example, ''transfer this call and all the data we have collected that are related to it to location X'') for the programmer. Nor do they provide the other types of tools to aid user-based design that we described earlier, such as the ability to test a program's functionality via simulation during development.

Custom CTI programs have historically been developed by specialist CTI applications groups located in independent firms, or in firms that are major users of sophisticated CTI. Such groups develop a custom CTI system by first visiting the client and studying the existing and planned business systems that the client wishes to enhance via CTI. They come away with a schematic that describes the sequence of functions to be carried out by the CTI system, and then turn the description over to coding specialists, who convert it into functioning software. The experts rely on their accumulated experience in understanding basic telephony functions and designing CTI systems, and on field tests and correction of the completed system to produce a good result for their clients.

System descriptions generated by CTI applications specialists typically consist of a private language of interconnected functional modules that make unambiguous sense to a particular firm's group of expert programmers—but that do not necessarily make sense to users or to other expert CTI programmers. For example, one of the specialist firms we interviewed uses a ''list aging'' function as a basic system module. The role of this function is to keep track of a list of items, and to signal when any given item has been on the list for *X* period of time. The firm had programmed the function as a prebuilt software object, and found it to be a useful component in the development of many very different custom CTI systems. Thus, it is useful for tracking the length of time an incoming telephone call

has been unanswered in a telephone sales system; for monitoring the checkout time of hotel guests; and for tracking whether bank clerks process mortgage applications in a timely manner. Other firms we interviewed had no functional equivalent of the list aging module; however, they could achieve the same system functions by subdividing their functional descriptions and systems along other boundaries.

Within the last five years, a new type of CTI software product called a CTI applications generator has come to market. CTI applications generators work in conjunction with CTI server software, and contain at least primitive versions of all of the aids for user-based design that we described earlier in this paper. That is, they contain design tools, prebuilt program modules, and some form of simulation capability that allows programmers to give their programs somewhat realistic functional tests during the course of development. The early suppliers of CTI applications generators did not, according to our interviews, develop these toolkits with the end user in mind. Rather, they intended them for specialist programmers working for CTI applications development firms and for VARs, ''value-added resellers,'' who specialize in installing and maintaining custom CTI systems for users. These suppliers then discovered that many of the earlier buyers were in fact sophisticated end users. Less sophisticated end users then asked the suppliers for more ''user friendly'' versions of the products, and the suppliers saw profit in responding to these requests. (CTI interviewees estimate the total market for applications generators sold to specialist system customizers to be in the range of $10–20 million annually. In contrast, they estimate that the potential market for application generators sold to end users is in the range of $200–300 million annually.)

Accordingly, suppliers then designed application generators more suitable for nonspecialist end users. These have been made easy for nonspecialists to use in three important ways. First, they are designed to be run on ordinary personal computers. Second, they incorporate object oriented programming and graphical user interfaces. Object oriented programming allows CTI specialists to offer CTI system designers design toolkits consisting of software ''objects'' that can be linked together in unique configurations to create a customized CTI system. Graphical user interfaces display these objects on a designer's computer screen in the form of icons which can be placed and moved and interconnected to represent a graphical representation of the user's desired system. The application generator then actually creates the working software that will implement the functionality displayed on the screen. Third, the *functions* of the objects used in these applications generators have been selected to mirror activity modules that are familiar to end users. Thus, instead of objects such as ''list aging,'' users are provided with objects that implement familiar functional routines such as ''get data about this caller,'' or ''fax document to this caller.''

Application generators of the type just described enable users without traditional programming skills, but *with* a good understanding of the functional makeup of the system they are trying to create, to develop relatively simple custom CTI systems, primarily of the type known as interactive voice response systems. Development of more complex custom systems tends to still require the help of specialists during the design phase, after which users might be able to maintain and improve them on their own. However, the solution spaces provided by suppliers to users are steadily expanding, as more complex capabilities are steadily moved ''down'' into user-friendly applications generators. In the late 1980s, for example, the programming of fax capabilities into a CTI system could only be done by sophisticated applications development firms. In 1992–1993 such a capability became available as an object in application generators suitable for programming by end users. Speech recognition is taking a similar pathway. In 1993–1994 this capability became available in toolkits designed for specialist CTI system designers. It is clearly heading for inclusion in application generators for end users, but it is not there yet.

In sum we see in CTI as we did in ASICs, that the application-specific portion of the problem-solving work of custom system design is shifting from supplier to user, as supplier-based expertise is progressively embodied in CTI application generator toolkits intended to for user-based designers. As was the case with ASICs, this shift is what we would predict if the advantages associated with greater supplier expertise in CTI system design are outweighed by the costs of transferring

application-specific sticky information from user to supplier, other things being equal.

A number of CTI software vendors do offer application generator toolkits suitable for use by sophisticated end users. Published data on market shares held by these firms do not presently exist. However, industry interviewees often nominate Artisoft, Brooktrout, and Parity Software as being among the larger suppliers as of this writing, each having market shares guesstimated to be in the range of 10–20 percent. Interviewees also agree that the firms with the larger market shares each currently sell at least hundreds and perhaps in the low thousands of such toolkits annually. This supports the view that suppliers should have significantly greater incentive to unstick and transfer supplier-related information needed by CTI system designers than would users to unstick and transfer information related to a specific application.

5. Discussion

We have now examined the locus of problem-solving work related to the design of customized products in two fields, ASICs and CTI. In both, we have found an identical pattern: The application-specific portions of customized products are increasingly being designed by users, with the aid of standard components and design toolkits provided by specialist suppliers. At the start of the paper, we observed that one would *not* expect design-related problem-solving to be partitioned between users and suppliers on the basis of economies associated with specialization in problem-solving. On the other hand we argued that such a pattern could emerge if economies of specialization were outweighed by differentials in sticky information transfer costs, and/or by costs associated with delegating the customization task to a specialist supplier ''agent.'' We think that the allocation of design-related tasks we have observed makes sense from this latter point of view. With respect to sticky information transfer costs, note that only the problem-solving work associated with chip customization requires access to application-specific ''sticky'' information in each case—and this is the precisely the portion of the design work that we find has been shifted to users. With respect to agency considerations, note that chances for opportunism are reduced, and so agency costs are reduced, if the party that invests in a task is the one that is more certain of the nature and reliability of the information being acquired from others. Clearly, in both ASICs and CTI, the portion of the problem-solving information provided by suppliers (standard, well-specified components and tools) is better understood than the portion held by users (information related to a novel application under development). Therefore, agency costs will be reduced if the application-specific work of product customization is invested in and conducted by the user—and this is the pattern we see.

While our observation of a shift of the problem-solving work of product customization to users in two industries is consistent with the reasoning we put forward earlier, we obviously cannot rule out alternative explanations on the basis of the data in hand. However, we can anecdotally report that experts in these two industries whom we have interviewed do tend to spontaneously explain the shift of product customization to users in terms of both agency costs and sticky information transfer costs. Examples:

(1) Sticky information effects in ASICs are reported to be strongly present in, for example, the design of high-end computer workstations. In a leading company in this field, designers are striving to optimize the whole system functionality in terms of data flow patterns that they know will be encountered when a workstation computer is applied to leading-edge graphical applications which are an important market for their firm. System designers are doing the ASICs designs themselves because an optimal design for any state of technology consists of creatively applying the solution space available in ASICs with the demands of the leading-edge application. When a designer is asked why he does not assign the ASIC design work to a specialist he answers: ''For an ASIC design specialist to design this ASIC as well as I can, he would have to know everything I know about the system—he would have to be me!''

(2) As an example of an agency effect in the ASICs field, leading-edge users report that silicon foundries tend to rate the solution space they offer somewhat conservatively. For example, they may conceal information from user-designers as to the very narrowest line widths they can deposit on a silicon wafer when their process is tuned to its best possible state. Instead, they

will tell the users that their process limits are what they can be sure of providing when the process is in an average state of tune. The incentive driving the silicon foundries to behave in this manner is that they want to be able to produce the user-developed designs at higher yields and therefore lower costs. Users who discover during the course of their design work that they need access to the extreme limits of the manufacturers' actual solution space must contact the manufacturers and negotiate a different tradeoff. If, in contrast, the design work were being done by the manufacturer ''agent,'' the users would probably never become aware of the tradeoff the manufacturer was making—possibly against their best interests.

(3) Sticky information effects are manifested in CTI when users say they prefer to design and make changes to their CTI systems themselves because they are unable to describe what they want to suppliers accurately and completely—so a supplier-developed system will predictably require multiple revisions before it fits the need. Users who cannot describe their need precisely are in this position for one or both of two reasons: they know what they want but cannot encode it precisely, and/or they themselves do not know what the ''right'' solution is prior to trial-and-error experimentation. For example, ''When I am not in the office, roll over my urgent calls to George'' may seem like a good idea in concept. Experimentation may reveal, however, that George is in fact unavailable at certain times of day, and/or that he cannot effectively address certain types of urgent calls that he initially thought he could handle, etc. (Hauschildt 1986, von Hippel and Tyre 1995).

(4) Agency effects are typically manifested in CTI in the form of lags by suppliers in responding to user requests for system changes. Two reasons are commonly given for specialist developers responding slowly to a given user request for a system modification. First, the supplier has an incentive to staff for average work flows, and so does not have enough personnel to respond to all requests in a timely manner during peak periods. During such periods the supplier has an incentive to respond to what he views as his ''best'' customers first. Second, the supplier has an incentive to economize on resources expended in responding to user requests (many suppliers provide service based on a flat annual fee) by waiting for several requests to accumulate from a given user in order to be able to ''do them all at once.'' Slow responses by suppliers can be quite costly for users however, because CTI systems are increasingly intertwined with basic business functions: For example, when an insurance firm wants to make a change to its CTI-based sales system to respond to a sales campaign by a competitor, the cost of delay can be high.

Although we have to this point documented the pattern in only two fields of mass-customized products, we propose that product and service customization by users will be found to be a very general phenomenon. For example, user-based design is common experience with respect to the combination of standard products into larger, customized systems. To understand this point, consider first that individual products, process equipment, and services commonly function as components in larger systems. This is clearly visible in the instance of processing machines (which fit into larger processing systems) and in the instance of industrial components (which perform functions within larger products or services). It is also true, but perhaps less intuitively obvious, in the instance of consumer goods and services (Boyd and Levy 1963). For example, a fork is a component part of a user's system for eating, and a component as well of systems for conveying signals on social status and other matters. Similarly, a telephone-answering service or machine is a component of many consumers' complex personal systems for receiving and storing data.

Users can and commonly do create customized end effects for themselves by combining standard products and services to create a customized system. For example, users design their own custom foods by combining standard food ingredients into unique recipes. Also, if the taste of a standard purchased food item is not quite what is wanted, users may customize it by adding spices to that item, and/or may adjust the impact of the standard item by the choices of other dishes being served in the particular meal ''system'' being devised. Similarly, if a standard production machine is not quite what is wanted, production engineers commonly modify it by adding customized tooling, and/or by combining that machine with other machines to create a customized production

system that will as a whole provide the unique function desired (Arora et al. 1997).[1]

Some factors that are important to determining the locus of design may shift back and forth over time. Others, however, appear to us to be driving markets for custom products and services irreversibly towards a pattern of user-based design—or, stated more generally, to a pattern in which problem-solving is carried out at the site of sticky local information that is less frequently drawn upon by problem-solvers. With respect to variable factors, consider that the heterogeneity of demand for a given type of product is not necessarily fixed. For example, heterogeneity may decrease if standards are tightened within a field, and increase if they are loosened again. Less heterogeneous demand, in turn, would tend to lessen the force with which the costs of transferring sticky local information are driving product design to user sites, because the uniqueness of each user's information with respect to the desired product would decrease.

The primary irreversible factor that we speculate is making user-based design an increasingly attractive option is technological advance. Specifically, improvements in computer hardware and software are allowing ''unstuck'' supplier information to be shifted to users in increasingly user-friendly and more capable ways. Consider, for example, that it has always been possible for a integrated circuit manufacturer to unstick key process information and transfer it to user-based designers. In earlier days, however, that information would have been unstuck by encoding it in a process specification sheet or booklet, and it would have been up to the user-designer to know when a particular bit of information was relevant to his or her design, find the booklet, and look it up. Today, process information can be embedded in a computerized design tool, which can be programmed to offer the user items of process information only if and as the design being worked upon makes them relevant. For example, a simulation tool can be programmed to tell a designer that ''your design is getting too big to process on a single chip'' only if and as the user is approaching that particular limit to the available solution space. More generally, the ability to encode unstuck problem-solving expertise in user-relevant language may not have changed over time, but the ability to offer this translated information conveniently and appropriately connected to the design work itself certainly has been greatly improved as a result of technological advance.

Suppliers of products and services who do wish to switch to a pattern in which their customers design the application-specific portion of a mass-customized product or service will confront important issues that differ from ''business as usual.'' For example, they may have to learn to modify their strategies for appropriation of innovation-related benefit. And, they will certainly need to learn to develop or acquire tools and components that will collectively provide the solution space needed by their customer-designers.

However, saying that suppliers must ''provide'' toolkits is not the same thing as saying that supppliers must develop all of the standard toolkit elements by themselves. Studies of the innovation histories of standard products have shown that the sources of innovation vary as a function of expected innovation-related benefit (summarized in von Hippel 1988). Also, as was discussed earlier in this paper, we may expect variation in the locus of innovation as a function of the location of sticky information required by the product developers. Incentives to develop individual toolkit items will vary with respect to these dimensions. Therefore we would expect that the innovation histories of toolkits will reveal that some toolkit elements are designed by ''lead'' users who have an especially high need for those particular elements coupled with especially rich information regarding them, and others by toolkit suppliers. (As anecdotal support for this expectation we note that, while toolkit innovation histories were not the subject of this study, we did during our data collection work observe instances of both user-developed and supplier-developed toolkit components in both ASICs and CTI.) As a consequence, suppliers transferring customization activities to their users must learn to search both lead

[1] A system can be seen as having many nested levels. Within each level, many components may be linked to form the next higher level system. For example, a computer hard disk drive is a system assembled from components. In turn, such a disk drive is a component in a computer system, which in turn is a component in, for example, a telephone switching system, which in turn is a component in a telecommunications system, etc. The point being made here is applicable independent of system ''level.''

user and supplier locations for the innovations that will allow them to compile effective and complete toolkits for their own users.

With respect to appropriability issues, consider that many suppliers of products and services appropriate benefit from both their design capabilities and their production capabilities. A switch to user-based customization can affect their ability to do this. For example, if a supplier develops an advanced toolset and keeps it in-house for the exclusive use of his own designers, customers can only benefit from that toolset if they also employ the supplier's production process—the two are effectively tied. However, when toolsets are made available to customer designers, this tie often weakens over time. Customers and independent tool developers can learn from process-specific toolkits about the input information a particular supplier's process requires, and then can use that information to design toolkits applicable to the processes of several suppliers. (This is precisely what has happened in the ASICs industry. The initial toolsets revealed to users by LSI and rival ASIC producers were producer-specific. Over time however, specialist tool design firms such as Cadance developed toolsets that could be used to make designs producible by a number of vendors. The end result was that many ASIC suppliers that previously established marketplace advantage on the basis of both product design skills and production skills were forced to a position of appropriating benefit from production skills only.)

When suppliers adopt a strategy of explicitly assigning the application-specific portion of customized product or service designs to their customers, the homogeneous items they produce are no longer standard end products or services. Instead, they are the standard tools and design components that user-based designers can draw upon to create customized end products or services. Suppliers must therefore learn enough about the activities and requirements of their users and enough about their design skills in order to ensure that those users are provided with the tools and components they need to perform the problem-solving work of application-specific design. Suppliers must also learn how ''much'' of a customized product should be provided in the form of standard components. Recall from the ASICs and CTI case studies that user-based customization of mass-customized products involves partitioning the design of a mass-customized product into segments, with expert suppliers providing the standard portions of the design, and users providing the application-specific portions. We can reason that firms can to an increasing extent capture the benefits associated with specialization in problem-solving *and* user-based design if this partitioning is done in such a manner as to make the application-specific portion of the problem ''as small as possible,'' while still providing the user-based designer with the degrees of freedom he or she needs to achieve the desired customized design.[2]

[2] Special thanks for very insightful discussion and contributions to the ideas presented in this paper are due to my colleagues Professors Ashish Arora, Anne Carter, and Scott Stern. I also thank Kwang Hui, MIT Sloan School Doctoral student, for major assistance in the collection of data for the ASICs case study reported upon here, and I thank Ali Pinar, Master's student in the MIT Technology and Society Program, for major assistance in the collection of data for the CTI case study reported upon here.

References

Allen, T. J., ''Studies of the Problem-Solving Process in Engineering Design,'' *IEEE Trans. Engineering Management*, EM-13, 2 (1966), 72–83.

Arora, A. and A. Gambardella, ''The Changing Technology of Technological Change: General and Abstract Knowledge and the Division of Innovative Labor,'' *Research Policy*, 23 (1994), 523–532.

——, ——, and E. Rullani, ''Division of Labour and the Locus of Inventive Activity.'' *J. Management and Governance*, 1 (1997), 123–140.

Arrow, K. J., ''The Economic Implications of Learning by Doing,'' *Rev. Economic Studies*, 29 (1962), 155–173.

Barron, J., *Thinking and Deciding*, Cambridge University Press, New York, 1988.

Boyd, H. and S. Levy, ''New Dimensions in Consumer Analysis,'' *Harvard Business Rev.*, 41 (1963), 129–140.

Chakravarty, D., ''Marketing ASICs,'' in N. G. Einspruch and J. Hilbert (Eds.) *Application Specific Integrated Circuit (ASIC) Technology*, Academic Press, San Diego, CA, 1991.

Cohen, W. M. and D. A. Levinthal, ''Absorptive Capacity: A New Perspective on Learning and Innovation,'' *Admin. Sci. Quarterly*, 35 (1990), 128–152.

Davis, R., ''Knowledge-Based Systems,'' *Science*, 231, 4741 (1986), 957–963.

Habermeier, K. F., ''Product use and product improvement,'' *Research Policy*, 19 (1990), 271–283.

Haskard, M. R., *An Introduction to Application Specific Integrated Circuits*, Prentice Hall, Sydney, Australia, 1990.

Hauschildt, J., ''Goals and Problem-Solving in Innovative Decisions,'' in E. Witte and H.-J. Zimmermann, *Empirical Research on Organizational Decision-Making*, Elsevier Science Publishers B. V., North-Holland, Amsterdam, 1986.

Hilbert, J. L., ''Introduction to ASIC Technology,'' in N. G. Einspruch and J. Hilbert (Eds.), *Application Specific Integrated Circuit* (*ASIC*) *Technology*, Academic Press, San Diego, CA, 1991.

Katz, R. and T. J. Allen, ''Investigating the Not Invented Here (NIH) Syndrome: A Look at the Performance, Tenure, and Communication Patterns of 50 R&D Project Groups,'' *R&D Management* 12, 1 (1982), 7–19.

—— and ——, ''Organizational Issues in the Introduction of New Technologies,'' in R. Katz (Ed.), *Managing Professionals in Innovative Organizations*, Ballinger, Cambridge, MA, 1988.

—— and M. L. Tushman, ''External Communication and Project Performance: An Investigation into the Role of Gatekeepers,'' *Management Sci*., 26, 11 (1980), 1071–1085.

King, A., ''Retrieving and Transferring Embodied Data: Implications for Self-Directed Management of Interdependence Within Organizations,'' NYU Stern School of Business Working Paper, New York, January, 1997.

Larkin, J., J. McDermott, D. P. Simon, and H. A. Simon, ''Expert and Novice Performance in Solving Physics Problems,'' *Science*, 208 (1980), 1335–1342.

McClean, W. J. (ed.), *Status 1995*, Integrated Circuit Engineering Corporation, Scottsdale, AZ, 1995.

Marples, D. L., ''The Decisions of Engineering Design,'' *IRE Trans. on Engineering Management*, June (1961), 55–71.

Mathur, G., Determinants and Outcomes of Interorganizational Interaction in Technological Innovation, Unpublished Doctoral Dissertation, Harvard Business School, Cambridge, MA, 1996.

Multimedia Telecommunications Association, 1996 *Multimedia Telecommunications Market Review and Forecast*, Multimedia Telecommunications Association, Washington DC, 1996.

Nelson, R. R., ''The Role of Knowledge in R&D Efficiency,'' *Quarterly J. Economics*, 97, 3 (1982), 453–470.

——, ''What is Public and What is Private About Technology?'' Consortium on Competitiveness and Cooperation, Working Paper No. 90-9, Center for Research in Management, University of California at Berkeley, CA, 1990.

Nixon, T., ''Design Considerations for Computer-Telephony Application Programming Interfaces and Related Components,'' *IEEE Comm. Magazine*, 34, 4 (1996), 43–47.

Pavitt, K., ''The Objectives of Technology Policy,'' *Science and Public Policy*, 14, 4 (1987), 182–188.

Pine, J. B. II, *Mass Customization: The New Frontier in Business Competition*, Harvard Business School Press, Cambridge, MA, 1993.

Pisano, G., ''Knowledge, Integration, and the Locus of Learning: An Empirical Analysis of Process Development,'' HBS Division of Research, Working Paper #95-006, Cambridge, MA, 1994.

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

Strathmeyer, C., ''An Introduction to Computer Telephony,'' *IEEE Comm. Magazine*, 34, 4 (1996), 2–7.

Szulanski, G., ''Exploring Internal Stickiness: Impediments to the Transfer of Best Practice Within the Firm,'' *Strategic Management J*., 17 (1996), 27–43.

Thomke, S., ''Managing Experimentation in the Design of New Products and Processes,'' Working Paper #96-037, Harvard Business School, Cambridge, MA, 1996.

von Hippel, E., *The Sources of Innovation*, Oxford University Press, New York, 1988.

——, ''Task Partitioning: An Innovation Process Variable,'' *Research Policy*, 19 (1990), 407–418.

——, '' 'Sticky Information' and the Locus of Problem Solving: Implications for Innovation,'' *Management Sci*., 40, 4 (1994), 429–439.

—— and M. Tyre, ''How 'Learning by Doing' is Done: Problem Identification in Novel Process Equipment,'' *Research Policy*, 24 (1995), 1–12.

Walker, R. and N. Tersini, *Silicon Destiny: The Story of Application-Specific Integrated Circuits and LSI Logic Corporation*, C.M.C. Publications, Milpitas, CA, 1992.

Walters, R., *Computer Telephone Integration*, Artech House, Boston, MA, 1993.

Wright, T. P., ''Factors Affecting the Cost of Airplanes,'' *J. Aeronautical Sci*., 3 (1936), 122–128.

Accepted by Ralph Katz; received May 1995. *This paper has been with the author* 8 *months for* 3 *revisions*.