How "Learning by Doing" is Done: Problem Identification in Novel Process Equipment

Eric von Hippel and Marcie Tyre

January, 1993 SSM WP# BPS 3521-93

We would like to express our gratitude to Professors Anne Carter and Shmuel Ellis, and to Dietmar Harhoff and Stefan Thomke for their very helpful comments and advice on the ideas embodied in this paper.

Abstract

The costs of producing goods and services has been shown to decline over time as a result of "learning by doing." In this paper we explore how learning by doing is done at the micro level via empirical study of a sample of problems affecting two novel process machines. These problems were created and/or identified by "doing" in the factories where the machines were used. Analysis shows two forms of learning by doing. The first enables the identification of problems through field use, and the second involves the creation of problems and related needs for improvement by problem solving in the field.

Examination of the role of doing in these two types of learning by doing allows one to understand why it would be very difficult to eliminate doing and still learn the same (important) things. It also suggests that, typically, one can't "get it right the first time" when introducing a new product or process to the field, and that it would be valuable to adapt the innovation process accordingly.

How "Learning by Doing" is Done: Problem Identification in Novel Process Equipment

1. Introduction

Beginning with Wright (1936) a number of studies have shown that the unit cost of producing manufactured goods 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." More recently, Rosenberg (1982) has shown that similar gains can accrue to the end users of a product as their skill in using grows through "learning by using." (For example (ibid p.131), after a given jet engine has been in use for a decade, the cost of maintenance may have declined to only 30% of the initial level due to learning by using.)

Although the economic significance of learning by doing and using has been made clear, the *process* by which these gains are achieved is still very unclear (Adler and Clark 1991). That is, we do not know the mechanisms by which such learning is achieved, nor do we know whether or why "doing" is actually essential to learning what is learned. In this paper we will explore these matters by means of an empirical study of a particular kind of doing - the identification through field use, of problems affecting novel process machines. The sample of problems we studied were not anticipated prior to actual use of the machines in the field, and so their discovery did represent instances of learning by doing.

We find that two different types of learning by doing are involved in the identification and, in some instances the creation of the field problems. In the first type, the use environment serves as a stable arena in which a product or

process or service is applied and tested. The arena might be complex, containing both things and people but, if users are present they are not actively problem-solving with respect to the test being conducted. (For example, the Tacoma Narrows bridge failed shortly after its completion in 1940 due to an unanticipated instability encountered in the use environment (Petroski 1992, 164). Bridge users were certainly part of that use environment, but they were not engaged in problem-solving with respect to the stability of the bridge.) The second type of learning by using differs from the first in that the use environment is changed by users or others who <u>are</u> actively problem-solving and changing the use environment - sometimes in ways that create unanticipated problems for products or processes or services that operate within it. In our sample, this second type of learning by doing was typically carried out by users of the process machines under study. As a result of their experience, they changed their views as to how the machine should best perform - and asked for changes. In such cases the use environment is much more than a passive test bed, and users were central to the learning process observed. In our study, the outcome of the former type of learning by doing was to identify problems that prevented the machine from functioning according to its original specification. The outcome of the latter type of learning by doing was to identify ways to make the machine or some other part of the use environment different and better than it was originally planned to be.

After we have characterized learning by doing in terms of the two types or mechanisms just described, we are able to apply tests of reason as to whether actual doing (application of the machines in their actual use environment) is necessary to achieve the learning that we observed taking place. We conclude that doing is sometimes the only practical way to succeed.

Innovation developers may be interested to note an implication of this finding: To the extent that learning by doing is necessary, one often cannot "get

it right the first time" when developing new products and services. This suggests, in turn, that it might be desirable to change from methods of managing the innovation process that involve a "once through" design, to methods that explicitly incorporate learning by doing as part of the innovation process. Extant methods that have this attribute include the "rapid prototyping" process sometimes used by software developers, and the "put a product on the market, learn from what happens, and then put out something better" approach that has been attributed to some Japanese consumer electronics firms.

2. Learning by Doing as Problem Solving

Learning by doing is simply a form of problem-solving that involves application of a production process (or product, service or technique) in its intended use environment. In order to understand this learning process better, a brief digression into the general nature of problem-solving will be useful.

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. 43-7.).[1] This finding is supported by empirical studies of problem-solving in the specific arena of product and process development (Marples 1961, Allen 1966). Such studies do show trial and error (or, more precisely, trial, failure, learning, revision and re-trial) as a prominent feature.

Trial and error procedures guarantee a problem solution only in the instance of "well structured" problems - which are defined as those for which one can precisely specify a process of trial and error that will lead to a desired solution in a practical amount of time (Reitman 1965, Simon 1973, Pople 1982).

---

[1] Some level of insight is necessary to problem-solve effectively. Imagine the cost if one used truly random trial and error: In order to discover how to exit a room, one would be as likely to try meditation or eye blinking as to try walking through a door.

For example, a traveling salesman problem can be well structured, because one can precisely specify a generator of alternative solutions and a solution testing procedure that are guaranteed to eventually identify the best solution. However, "In general, the problems presented to problem solvers by the world are best regarded as ill structured problems. They become well structured problems only in the process of being prepared for the problem-solvers. It is not exaggerating much to say that there are no well structured problems, only ill structured problems that have been formalized for problem-solvers." (Simon 1973 p. 186).

Ill structured problems may involve an unknown "solution space" (a precisely specifiable domain(s) in which the solution is known to lie). They may also involve unknown or uncertain alternative solution pathways, inexact or unknown connections between means and ends and/or other difficulties. Ill-structured problems are solved by a process of first generating one or more (typically several) alternative solutions. These may or may not be the best possible - one has no way of knowing. These alternatives are then tested against a whole array of requirements and constraints (Marples 1961, Simon 1969 p.149). Test outcomes are then used to revise and refine the solutions under development, and - generally - progress is made in this way towards an acceptable result.

In sum, learning by doing almost always addresses ill-structured problems. We can therefore anticipate that the problem-solving process associated with it will have the general characteristics described for this class of problems, plus some more particular attributes associated with "doing."

3. A Study of Unanticipated Field Problems

The particular kind of learning by doing that we decided to explore empirically was the identification in the field of unanticipated problems affecting novel process machines. Our sample was derived from field problems encountered by two specialized new process machines recently developed for use

in the factories of a large electronics manufacturer. Both of these machines, a solder paste profiler and a component placer, are used in the process of attaching surface-mounted integrated circuits to large, complex circuit boards.[2] Although carried out within a single firm, the development of each of the two machines was an independent event. Each was designed and built by different equipment development groups, and was first applied in a different factory of the firm by different process engineers and plant-based users. Both machines were (eventually) described as successful by users and developers. Both have been replicated for use in other factories, with later installations encompassing modifications developed during early field application.

Research into the introduction of new process equipment (e.g. Hayes and Clark 1985; Tyre and Hauptman 1992) has shown that problems continue to arise and get resolved for two years or more following first use. Therefore it was useful for our study purposes that the solder paste profiler had been placed into service 18 months before data collection began, and the component placer 2 years before data collection began. This meant that a relatively large number of unanticipated problems had surfaced in each machine.

---

[2]Brief Description of Solder Paste Profiling and Component Placing Machines

The board assembly process begins with the application of a tiny dab of "solder paste" - a form of solder which has the consistency of toothpaste at room temperature - to each location on a circuit board where an electrical connection must be made between the board and an attached component. (The spacing between dabs can be as small as 25 thousands of an inch today, and each dab may be as small as the period at the end of this sentence.) Next is inspection (or profiling) of the solder paste -- this is where the first machine that we studied (the "paste profiler") plays its role. The machine scans the board surface with a laser-based vision system and determines whether the location, amount and configuration of each dab of solder paste applied to the board is as specified. If all is correct, the board is then passed on to the next operation, where components are placed on the boards.

The next machine we studied was designed to automate the placement of complex components. It uses a vision system and a robot arm to pick up integrated circuits (which look like small plastic boxes with two or more rows of tiny metal legs protruding from the bottom) and place them on the circuit board at precisely the right locations - with each metal leg of each component resting exactly on one of the dabs of solder paste previously applied to the board. When this step is completed, the board is passed through an oven that heats the solder paste and converts it into liquid solder. When the board cools, the solder hardens into solid metal and the "placed" components have been permanently soldered onto the board.

3.1: Methods

After introduction into the factory use environment, each of the two machines that we studied was judged by users to have experienced several problems in the field. We began the process of selecting the sample for our study by identifying all such problems that were *first discovered after factory use began* and that had been diagnosed as to cause at the time of the study. We did this by first asking engineers involved in using and engineers involved in designing the novel equipment for an exhaustive list of all "significant" problems observed after factory use of each machine began that had subsequently been diagnosed. (Note that under this procedure factory machine users determine both what constitutes a problem and what constitutes a solution. Thus, a problem can entail a machine failure to perform as designed, or significant user dissatisfaction with a machine that is functioning as intended by its designers.)

Next, we took the list of observed-and-diagnosed problems to the machine developers and asked them to separate these into two categories: (1) problems the developers had *not* anticipated prior to field use of the machines; (2) problems that they had known about prior to field introduction, but had not yet fixed for some reason. The results were as shown in table 1.

Table 1: When were problems affecting the machines first recognized?

		# of problems affecting Profiler	Placer	Total
(1)	Only after machine was first installed in field *- Example: After the component placing machine was installed in field, users noticed that it was unable to pick up parts that had "tilted" heat sinks on top. This problem was a surprise to developers. They had not known that such parts existed, and had not designed the machine to handle them.*	9	13	22
(2)	Before machine was first installed in field . *- Example: Specifications called for machine to handle all boards to be processed without needing extra setup. Developers couldn't find a way to do this during the development time frame; users and developers agreed that this problem would be resolved after machine introduction.*	3	2	5
	Total	12	15	27

As can be seen in table 1, of a total of 27 machine problems fixed after the machines had been installed in a plant, 22 had first been identified via field use. The distribution of our sample with respect to this matter is similar for the two machines studied ($p < 0.27$, Fisher exact test).

Data on machine problems and related problem-solving activities were collected through interviews with both the users of each machine (the process engineers at the factories where they were installed) and developers (key people in the process equipment development teams). Most interviewees had been with the projects studied from their inception to the present. Initial interviews were conducted on-site where respondents could refer to contemporary logbooks and could demonstrate the problems they described on the actual equipment. Interviews lasted from three to six hours, including plant tours. Respondents were interviewed both separately and, to the extent possible, together. Follow-up questions were discussed in additional face-to-face meetings, and by telephone and electronic mail.

3.2: Patterns in Unanticipated Field Problems

When a problem in machine performance is observed and deemed worth correcting, a diagnostic process is conducted by plant or laboratory personnel to trace the failure symptom back to an underlying cause so that the problem can be fixed. (Patterns in the diagnostic procedures used are explored in Tyre and von Hippel (1993).) Since our study of problems first recognized in field use was retrospective in nature, we had the luxury of knowing both the initial symptom and the cause eventually diagnosed for each case in our sample.

In our analysis, we used these diagnoses to identify information that would have allowed engineers to eliminate the cause of each problem prior to field use - if only that information had been incorporated into the machine as originally designed. (Problems can be understood and "solved" at many levels. For example, if machine operators find they must make frequent machine adjustments and find this troublesome, one level of solution would involve making the adjustment process easier. A solution at a deeper level would involve reducing or eliminating the need for adjustment. In our analyses we focused on the level of diagnosis and solution actually selected and implemented by the problem solvers studied.) Next, rather than attempting to test pre-established hypotheses, we used a grounded research approach (Glaser and Strauss 1967) to examine the information that had been identified by using, to identify patterns in the learning by doing process. The most important patterns we found had to do with the "state" of problem-related information, and table 2 summarizes what we found on this matter.

Table 2: At the time the machine was designed, what was the "state" of the information which could have been used to avoid an unanticipated field problem?

"State" of problem-related information	# of problems affecting Profiler	Placer	Total
(1) Problem-related information existed in use environment when machine was designed, but:			
- (a) was not known to machine designers.	2	3	5
- (b) was known but not used by designers	5	5	10
(2) Problem-related information was created after machine was introduced to field by problem solvers outside of the design lab who were:			
- (a) users working directly with machine	1	4	5
- (b) problem solvers working on other aspects of the production process	1	1	2
Totals	9	13	22

We next describe each category in table 2 in a little more detail, and provide examples that will help to convey the flavor.

In cases tabulated under 1(a) in table 2, the information needed to understand or predict problems did exist in the intended use environment during the development of the machine, but lab personnel did not obtain that information, and so did not identify the related problem until the machine was actually placed into use in the factory. In each of the instances in this category, interviewees told us that the information could easily have been provided to the lab - had the developers thought to ask and/or had users thought to volunteer it. The following will illustrate:

Example: Yellow Circuit Board Problem

The component-placing machine uses a small vision system incorporating a TV camera to locate specific metalized patterns on the surface of each circuit board being processed. To function, the system must be able to "see" these metalized patterns clearly against the background color of the board surface itself.

The vision system developed by the machine development group functioned properly in the lab when tested with sample boards from the user plant. However, when it was introduced into the factory it sometimes failed. Field investigation by development engineers showed that the failures were occurring when boards that were light yellow in color were being processed.

The fact that boards being processed *were* sometimes light yellow was a surprise to lab personnel. While factory personnel knew that the boards they processed varied in color, they had not volunteered the information to the lab because they did not know that the designers would be interested. Early in the machine development process, factory personnel had simply provided samples of boards used in the factory to the lab. And, as it happened, these samples were green in color. On the basis of the samples, developers had then (implicitly) assumed that all boards processed in the field were green. It had not occurred to them to ask users, "how much variation in board color do you generally experience?" Thus, they had designed the vision system to work successfully with boards that were green.

In cases coded under 1(b) in table 2, the lab had in its possession <u>all</u> information needed to anticipate and avoid a field problem - but nonetheless failed to identify it prior to field use. Consider the following example.

Example: Component Slippage Problem

Just before the component placing machine places components on a board, little dabs of solder-containing paste are applied to the board - one

at each spot where an electrical connection is to be made between a component leg (a wire protruding from the base of the component) and the board. The machine designers knew about this, but chose to use adhesive tape instead of solder in their laboratory simulation of the use environment. (Use of solder would have required setting up the lab to comply with rules regarding the handling of hazardous materials - a costly matter).

When the component placer was installed in the field, it was noticed that components unexpectedly slipped sideways to an unacceptable degree when the robot arm was pressing them onto the board. Investigation showed that the mound-shaped dabs of solder paste were firm enough to push the component sideways if the legs touched down on their sides instead of directly on their tops. This effect did not occur in the lab because the lab had not used solder in its tests.

Three of the cases coded under 1b deserve special mention. In these, unanticipated field problems were caused by the premature failure of machine parts due to design error (for example, an inappropriately small bearing was designed into the machine - and quickly failed). It seems to us reasonable to classify these under "information known by lab but not used" because the problems could have been anticipated and avoided prior to field use by either following generally known principles of good design or subjecting the machine to longer life tests in the lab. (The intended field operating life of the machine was known to the lab.) If the attributes of the use situation causing the failure had not been known to the lab in cases of premature parts failure (e.g.. "We didn't know that you were going to process such heavy parts"), we would have coded the cases under category 1a in table 2.

In the second category of table 2, the information that might have allowed designers to anticipate and forestall a field problem was created after field introduction of the machine by problem solvers outside of the development lab. In most instances (category 2a) these problem solvers were machine users who, as a result of their field experience with the machine, had decided that he or she

wanted something different from or better than the original specification. Consider the following example.

Example: Location Adjustment Problem

> Each time a new board design was processed by the component placing machine, operators had to tell the machine where to put each of the components to be placed on the new board. They did this by entering the X and Y coordinates of each part location in the machine's computer memory. In case these coordinates required later adjustment, operators and machine designers both assumed that the operators would re-enter new X and Y coordinates.
>
> After the machine was installed in the plant, users discovered that they had to adjust X and Y coordinates very frequently. They also found that it was very cumbersome to do this by reentering new coordinates. Instead, they learned to make the needed adjustments via an obscure "move it over by X amount" command that was buried several layers down in a software menu on the machine's control panel. The problem that users then brought to the attention of machine designers was: The "move it over by X amount command" is very hard to reach and use. Make a more convenient one!

In two instances (category 2b), the problem solvers that changed the use environment in ways that caused machine problems were outside of the use environment most immediately surrounding that machine, working on other aspects of the printed circuit board production process. Consider the following example:

Example: Solder Mask Problem

> Some months after the solder paste profiling machine was introduced to the field, engineers working on the printed circuit board production process decided to slightly reduce the thickness of the plastic film (called a solder mask) which served as the topmost coating of the printed circuit

boards being processed. This was done to solve a problem unrelated to the profiler - the engineers wanted to improve the uniformity with which solder flux was being applied to the board. However, as an unanticipated side effect, the profiling machine's measurements suddenly became unreliable.

When engineers responsible for the profiling machine investigated the sudden rash of failures, they eventually found that the thinner solder mask was the root cause. The profiler was designed to identify the top surface of the board to be measured by reflecting a laser beam from that surface. Introduction of the thinner solder mask resulted in greater amounts of laser light passing *through* the film and reflecting off layers of metal located inside the circuit board. As a consequence, the machine sometimes judged these lower layers to represent the surface of the solder mask film - which in turn led to incorrect measurements.

4. Mechanisms Underlying Learning by Doing

From our table 2 data, we can see that all of the problems in our sample were identified after field use began, and that some were created after field use began. We now propose that problem identification while doing, and problem creation while doing involve two different types of learning by doing. We will explore each in turn.

4.1: Problem Identification by Doing

How was field experience or "doing" involved in the identification of the problems in our sample? We propose that the learning mechanism at work is a form of pattern matching which we term "interference finding." A pattern is a set of features or characteristics that describes an object (or event, stimulus, etc.). In the most basic form of pattern matching, one looks for similarities between patterns. In "subtractive pattern matching" one looks for differences between patterns. (For example, astronomers may compare two star maps of the same area of sky taken at two different times in order to "subtract" everything that is the same and highlight only what is changing, such as rapidly moving comets.)

When one introduces a machine into the field for the first time, one can also think of the machine and the use environment as patterns - with failures being the means of identifying problematic interferences between them.

Alexander (1964, 19) describes a basic version of interference finding in his discussion of a means for characterizing the fit between form and context:

> "It is common practice in engineering, if we wish to make a metal face perfectly smooth and level, to fit it against the surface of a standard steel block, which is level within finer limits than those we are aiming at, by inking the surface of this standard block and rubbing our metal face against the inked surface. If our metal face is not quite level, ink marks appear on it at those points which are higher than the rest. We grind away these high spots, and try to fit it against the block again. The face is level when it fits the block perfectly, so that there are no high spots which stand out any more."

In the case described by Alexander, the link between the observed symptom of interference - a black mark on the metal face being smoothed - and the cause - a physical location at which the two metal plates are making contact - is simple and direct. Also in this case the fault for ink marks observed is clearly assigned to only one of the two objects placed into contact: the "standard steel block" serves as a template, and the other metal surface is the one to undergo any needed correction. In the instance of a machine and its use environment the patterns being juxtaposed are much more complex; both or either are potential candidates for adjustment; and the link between the symptom of interference and the underlying cause may be obscure and difficult to diagnose. The principle, however, remains the same.

In problem identification by doing, therefore, the unique contribution of the field environment is precisely the precipitation of the failures. By means of interference finding, a formerly obscure interaction between machine and use environment is flagged by a very visible symptom: unacceptable performance in

the field. This mechanism operates in all our cases - whether the aspect of the user environment involved in the interference is pre-existing or newly created. Thus, in the case of the "yellow board" problem described earlier, interference finding pointed out the obscure matter of variation in the color of circuit boards used in the factory by associating that fact with a very salient failure of the component placing machine in the field. Similarly, in instance of the "location adjustment" problem, an interference between newly created user expectations and actual machine performance lead to a request for an improved "move it over by X command."

Who is learning by doing in the case of identifying unanticipated field failures by interference finding? As just noted, the field environment itself is only serving as a complex pattern that precipitates failures in these instances. The failure can be sensed and diagnosed by anyone (or any thing, in the case of self-monitoring machines) who is interested in and in a position to do so. In our sample, all of the failures were first sensed by users, and the follow-on diagnostic work was conducted by users and/or machine designers depending on the nature of the information and skills required (Tyre and von Hippel, 1992).

4.2: Improvement Creation by Doing

The problems coded in the second category of table 2 were created by a change in the use environment that occurred after the machine was installed in the field. The learning by doing mechanism at work in such instances was problem-solving affecting the use environment that was carried out by users (category 2a) or others associated with the production process (category 2b) rather than by machine designers.

The possibility that the use environment might change makes a very significant difference from the point of view of the designer's problem-solving task. When the designer is the only problem-solver active on a problem, he or

she is in the same position as a scientist or engineer asking a question of "nature." These problem solvers know that the answer they seek may be complex and hard to puzzle out. But they also know that it is not being changed as they work due to the actions of other problems solvers. For example, engineers building the first supersonic plane did not know all they needed to know about the stresses the airplane would encounter is supersonic flight. But they knew that nature would remain stable as they learned more, and that the correct answer would not change half way through the project. In contrast, a use environment populated by and/or affected by autonomous problem solvers offers no such assurance. Under such conditions the environment and thus the answer that the designer is seeking may change.

Who does the learning in this second type of learning by doing? In our sample, it was engineers and operators working on the process - process "users" - who created the improvements that changed the use environment in the ways we observed. In principle, one might argue that either the user or the developer could do this second type of learning by doing, but responsibility for overall process improvement was assigned to users in the situation we studied, and so it is logical that they were the innovators here.

4.3: Time Sequence in Learning by Doing

We have seen that two types of learning by doing, interference finding and improvement creation, are initiated at the time a machine (or product or service) is first introduced to the use environment. On the face of it, it seems reasonable that users or others will generally discover interferences with the existing use environment sooner than they will engage in devising improvements and, as table 3 shows, we do see a significant tendency in this direction in our data. ($P < 0.02$, Fisher exact test, that existing interferences are noticed more quickly (within one month of machine introduction) than those due to improvement creation.)

Table 3: How soon after machine was introduced to the field was the problem symptom noticed?

"State" of problem-related information	# of Months after machine installed that problem symptom first noticed				
	<=1	1-2	>2	NA	Total
(1) Problem-related information existed in use environment when machine was designed:	11	1	3	0	15
(2) Problem-related information was created after machine was introduced to field by problem solvers outside of the design lab[a]:	0	2	2	1	5[a]

a The sample in table 3 is the same as in table 2 except that the two cases in table 2's category 2b are excluded from category 2 in table 3. The reason: In category 2b, the creation of problem-related information was independent of the machine under study. (Inclusion of these cases would have strengthened rather than weakened the statistical finding reported here.)

The pattern that we show in table 3 may be commonly seen in learning by doing, but it cannot be taken as an iron rule. After all, improvements might sometimes be devised very rapidly, and/or the symptom of an existing interference between a machine and a use environment might not occur immediately when the machine (or product or service) is introduced. With respect to the latter, consider that the machine and/or the environment might not be immediately configured in a way that would cause an existing interference to be expressed. (For example, if an interference was associated with the "annual report" section of a software package, the user might not see a related symptom until that section of the package was activated.) Also, the symptom of an interference may not be manifested immediately, as in the case of premature wear failures in a machine. (We had three such cases in our problem sample, and two of these took many months to emerge.)

5. The Utility of Doing in Learning by Doing

Earlier investigators have shown that learning by doing can have significant effects on reducing costs of production. We have now identified two mechanisms that are involved in such learning. With these in mind, we can now think about how and whether the same learning might be achieved without the necessity of "doing" -that is, applying the innovation in the actual use environment.

Engineers who wish to avoid interference finding in the use environment face a difficult task. First, consider that the use environment and the machine contain a myriad of highly specific items of information that could potentially interfere. Second, which items among these are problem-related is contingent on the solution path taken by the engineer designing the product. We can illustrate both of these matters via the yellow circuit board problem described earlier.

With respect to the first point, note that the property of the board at issue in the yellow board case was problematic in a very narrow and specific way. That is, the problem with the board was not that it had "physical properties," nor that it had a color. The problem was precisely that the boards were yellow, and a particular shade of yellow at that. If one cannot radically simplify the problem by making general assumptions (e.g., we assume that the chemical composition of the board will not cause field problems - and so we will ignore it), one can see that problem solvers must analyze a very large number of potentially problematic items and interactions between items if they wish to avoid field failures.

With respect to the second point, note that the problem caused by the yellow color of the board was contingent on the design solution to the component placing problem selected by the engineer, and this was only done during the development process. That is, the color of printed circuit boards in the user factory became relevant only when engineers, during the course of their

development of the component placer, decided to use a vision system in the component-placing machine they were designing, and the fact that the boards were yellow only became relevant when the engineers chose a video camera and lighting that could not distinguish the metalized patterns on the board against a yellow background. Since engineers often change the alternatives they are developing during the course of their development work (Marples 1961, Allen 1961), the relevance of any particular item of information to potential field problems can also change frequently during the development process.

Of course, we do not intend to suggest by this litany of difficulties that one cannot anticipate and avoid a field failure when use environments are stable with respect to that problem's cause. It simply says that to do so can be complex[3] and costly. Methods for reducing the likelihood of unanticipated field problems include simulating the use environment in the lab more completely: If the simulation is totally complete and accurate, one can cause all unanticipated failures to occur in the test lab instead of in the field. (This is the approach taken by airlines which seek to train pilots in simulators that are so accurate that simulator time is counted as the equivalent of actual flight time.). And/or, one can use various analytical procedures such as "fault trees" (Henley and Kumamoto 1981) which can help make the search for possible causes of failure more systematic. And/or, one can hire very experienced engineers who have prior experience with failure modes on existing products, and so are more likely to anticipate them when designing similar new products (Larkin et al. 1980). One

---

[3] We do not use quantitative measures of complexity in this discussion because, in the instance of ill-structured problems with poorly specified and explored solution spaces, one does not have the data such measures call for. (For example, Steinmann (1976) specified seven sources of task complexity that involve the 'absolute amount of information involved in a task, the internal consistency of this information, and the variability and diversity of the information'.) An additional complication in measuring the complexity of problems is raised by researchers (e.g., Larkin et al 1980) who suggest that task complexity involves both the attributes of a task or problem and the capabilities of the person attempting it. Campbell (1988) offers a good overview of the research on task complexity.

can also try to incorporate subsystems in one's design which have already been tested under field conditions. And/or, one can try to make some of the subtasks in a design project well-structured so as to reduce the possibility of unanticipated field failure in these.[4] And/or, one can lessen the likelihood of failure by making the solution more robust - less dependent on possible variations in the use environment and/or more redundant. (The practice of incorporating safety margins into the design of bridges and buildings is an example of the first approach; the design of fault-tolerant computers an example of the latter.)

Both the costs and the benefits of identifying potential field failure prior to using differ from project to project. Since learning by doing is the default strategy - one is attempting to anticipate and prevent problems that will otherwise make themselves known through interference finding - one can expect that designers will invest more or less in some of the fault anticipation or fault mitigation strategies just listed depending upon the costs and benefits that they expect. For example, one would expect designers of nuclear power plants to invest a lot in attempting to anticipate and avoid potential field failures, and they do (Nuclear Regulatory Commission 1975).

Problem creation was associated with the second learning by doing mechanism identified, and here designers are very unlikely to be able to generate the same information by other means, thus avoiding related field failures and requests for improvement. This mechanism involves problem-solving by an autonomous group - here, users - who are both posing hard-to-anticipate problems, and are generating an unpredictable set of proposed alternative

---

[4]Despite the restrictiveness of the criteria for well-structured problems, designers can often partition an overall design task in such a way as to create some well-structured subproblems. For example, Smith and Eppinger (1991) studied a sub-problem in the design of automobile brakes that seems to us so tightly constrained as to meet the criteria of a well-structured problem. The goal was that "the brakes on car model X should not squeal under these conditions when test A is applied". To achieve this goal, it was permissable to manipulate only three well-understood variables, such as the composition of the brake lining material, in precisely specified ways.

solutions - some of which involve changes in the machine (or product or service) provided by a particular manufacturer.

Neither game theorists' models of cooperative games (e.g., Axelrod 1984) nor psychologists' models of "mutual adaptation" (Lave and March 1975, p 248) offer us much help in predicting the path or the outcomes of this type of multi-party problem-solving. Although both developer and user are presumably motivated towards mutual adaptation (or, at least, the machine developers are motivated to adapt to their user-customers), the problems that machine users are framing and partially solving are, as noted earlier, ill structured. Therefore, as our section 2 discussion of ill structured problems indicates, the problem-solving path that will be taken by user problem solvers cannot be predicted by the designers with certainty.

Of course, field problems caused by this type of learning by doing can be stopped if one prevents users and others from engaging in the problem-solving that may create them. Thus, for example, the military or some other innovation using organizations might demand that the users under its control do everything "by the book." However, innovation users typically cannot be constrained in this way by innovation designers - they are, after all, the customer. And even if this were possible, there would likely be a heavy cost to pay for relief from some field problems. After all, learning by doing by users and others can be a significant source of innovations and innovation improvements (von Hippel 1988).

6. Discussion

The approach we have taken to studying learning by doing involved selecting multiple instances of a single type of learning by doing event, the identification of an unanticipated problem in the field. We found two learning by doing mechanisms associated with this type of event: Interference finding in the

instance of problem recognition, and user problem-solving in the creation of some of the problems studied. These two mechanisms seem quite general, and we suspect that they will be found to be associated with learning by doing in a range of contexts. Interference finding may be a useful way to characterize all instances of problem identification in learning by doing, and a use environment affected by problem-solvers who are autonomous from the point of view of the designer of a particular process component are likely to be found in any but the simplest and most constrained circumstances.

Of course, we do not argue that these are the only two mechanisms for learning by doing: research conducted on additional types of learning by doing events might reveal additional ones. Examination of contexts in which learning by using (Rosenberg 1982) rather than learning by doing is going on may also be fruitful in this regard, although our own expectation is that these two phenomena will be found to be identical with respect to underlying mechanisms.

The level of analysis we have performed here would probably not allow us to determine the shape of an overall learning curve for a given production process. Such curves generally integrate the effects of learning by doing associated with many changes that are introduced to a process over time. However, the mechanisms we have identified for learning by doing may be exploited to allow us to suggest a particular shape for a learning curve that will be induced by the introduction of a particular change into a use environment. For example, we found that most pre-existing interferences between the new machines we studied and the use environment were flagged within one month of the machines' installation, while improvements derived from learning by doing followed somewhat later. If a large proportion of the total problems flagged as worth working on were due to the identification and resolution of existing interferences, and if these were diligently diagnosed and solved - and they certainly would be if they caused grossly unacceptable performance - one would

then find a relatively high rate of learning by doing immediately after the introduction of the novel element, that would drop to a low level over time. (This fits the pattern shown by Tyre and Orlikowski (1993) in their study of the pattern over time of changes introduced to a new process machine, and is the conventional wisdom with respect to the "debugging" of new products such as a computer software packages.)

On a different matter, our assessment of possible substitutes for the "doing" of learning by doing mechanisms we have identified here suggests that it may be very costly to substitute for in the instance of interference finding failures, and that it is probably impossible to do so in the instance of problems created by users or others learning by doing in the field. Therefore, it is very likely that the innovation process will involve learning by doing, and that those who develop new products and services will have to deal with it.

The need for learning by doing indicates that the innovation process will often be iterative - and that developers typically can't "get it right the first time." For managers, this suggests the value of shifting from product and service development methods that assume that one can specify a user need and use environment accurately at the start of a project to methods that expect user trial and error and incorporate it into the development process. Rapid prototyping is an example of such a method. Used in software development, this method is explicitly designed to shuttle repeatedly between manufacturer and user in order to better determine the "real need" for a given software package. First, key functions of proposed software products are simulated (prototyped) and provided to users for trial. Users then experiment with these prototypes, and ask manufacturers for improvements based upon what they have learned. This back and forth process continues until users are satisfied. This approach has been found to be better at creating a good fit between need and solution than method that rely upon the accuracy of an initial statement of need by users (Gomaa 1983,

Boehm, Gray, and Seewaldt 1984).

References

Adler, Paul S. and Kim B. Clark (1991), "Behind the Learning Curve: A Sketch of the Learning Process" Management Science V37, No. 3

Alexander, Christopher (1964), Notes on the Synthesis of Form, Cambridge, MA: Harvard University Press.

Allen, Thomas J. (1966) "Studies of the Problem-Solving Process in Engineering Design." IEEE Transactions on Engineering Management EM-13, no.2 :72-83.

Arrow, Kenneth J. (1962) "The Economic Implications of Learning by Doing." Review of Economic Studies 29 :155-73.

Axelrod, Robert. (1984) The Evolution of Cooperation. New York: Basic Books.

Baron, Jonathan. (1988) Thinking and Deciding. New York: Cambridge University Press.

Boehm, Barry W., Terence E. Gray, and Thomas Seewaldt. "Prototyping Versus Specifying: A Multiproject Experiment." IEEE Transactions on Software Engineering SE-10, no. 3 (May 1984): 290-303.

Campbell, Donald J. (1988), "Task Complexity: A Review and Analysis," Academy of Management Review, V. 13, No. 1, 40-52.

Glaser, Barney G, and Anselem L. Strauss (1967) The Discovery of Grounded Theory: Strategies for Qualitative Research New York Aldine De Gruyter.

Gomaa, Hassan. (1983) "The Impact of Rapid Prototyping on Specifying User Requirements." ACM Sigsoft Software Engineering Notes 8, no.2 :17-28.

Habermeier, K. F. (1989) "Product use and product improvement" Research Policy 19, 271-283

Henley, Ernest J. and Hiromitsu Kumamoto (1981) Reliability Engineering and Risk Assessment. Englewood Cliffs, New Jersey: Prentice Hall Chapters 2,3 and 7.

Larkin, Jill, John McDermott, Dorothea P. Simon, Herbert A. Simon (1980) "Expert and Novice Performance in Solving Physics Problems," Science vol 208, 20, pp 1335-1342.

Lave, Charles A. and James G. March (1975) An Introduction to Models in the Social Sciences, New York Harper and Row

Marples, David L. (1961) "The Decisions of Engineering Design." IRE Transactions on Engineering Management :55-71.

Newell, Allen, and Herbert A. Simon.(1972) Human Problem Solving. Englewood Cliffs, N.J.: Prentice-Hall, Inc.

Nuclear Regulatory Commission (1975) Report # 75/014, October

Petroski, Henry (1992) To Engineer is Human: The Role of Failure in Successful Design New York: Vintage

Pople, Harry E. Jr. (1982) "Heurisitic Methods for Imposing Structure on Ill-Structured Problems: The Structuring of Medical Diagnostics," Chapter 5 in Peter Szolovits, ed: Artificial Intelligence in Medicine Westview Press, Boulder, Colorado

Reitman, W. R. (1965) Cognition and Thought Wiley, New York

Rosenberg, Nathan.(1982) Inside the Black Box: Technology and Economics. New York: Cambridge University Press.

Schroder, H, Driver, M., & Streufert, S. (1967), Human Information Processing, New York: Holt, Rinehart & Winston.

Simon, H. A. (1973) "The Structure of Ill Structured Problems," Artificial Intelligence 4, 181-201

Simon, Herbert A.(1981) The Sciences of the Artificial, Second Edition. Cambridge: MIT Press, 1969.

Smith, Robert P. and Steven D. Eppinger (1991) "Identifying Controlling Features of Engineering Design Iteration," Sloan School of Management Working Paper # 3348-91-MS, December.

Steinman, D (1976) "The effects of cognitive feedback and task complexity in multiple-cue probability learning," Organizational Behavior and Human Performance, 15, 168-179.

Tyre, Marcie J. (1991a), "Managing the Introduction of New Process Technology:International Differences in a Multi-Plant Network, "Research Policy, 20, 1-21.

Tyre, Marcie and Oscar Hauptman, Oscar (1992), "Effectiveness of Organizational Response Mechanisms to Technological Change in the Production Process." Organization Science 3, 301-321.

Tyre, Marcie and Wanda Orlikowski (1993) "Windows of Opportunity: Temporal Patterns of Technological Adaptation" Organization Science , forthcoming.

Tyre, Marcie and Eric von Hippel (1993) "Situated Trial and Error Learning in Organizations" Sloan School of Management Working Paper, January.

von Hippel (1988) The Sources of Innovation New York: Oxford University Press.

Wright, T. P. (1936) "Factors Affecting the Cost of Airplanes," J. Aeronautical Science, 3, February 122-128.