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

Eric von Hippel* and Marcie Tyre**

* Professor, Management of Technology
MIT Sloan School of Management
Room E52-556
50 Memorial Drive
Cambridge, MA 02139

** Associate Professor, Management of Technology
MIT Sloan School of Management
Room E52-564
50 Memorial Drive
Cambridge, MA 02139

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

1. Introduction

When products, processes and services are introduced for the first time to their intended use environments, things often go wrong: Software is found to have unanticipated "bugs"; process machines that worked very well in the development laboratory promptly fail when first installed in a factory, and so forth [1]. This paper is devoted to learning more about why some problems can be so hard to spot before a product or service is deployed in a field environment - and how early use aids in revealing them.

Our discussion is framed in terms of learning by doing. Beginning with Wright [2] 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 [3] has termed "learning by doing." More recently, Rosenberg has shown that similar gains can accrue to the end users of a product as their skill or understanding grows through "learning by using." (For example, 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 as a result of learning by using [4].

Although the economic significance of learning by doing and using has been made clear, the *process* by which these gains are achieved is still quite unclear [5]. That is, we do not know the micro-level mechanisms by which learning by doing is actually done, nor do we know whether or why doing is essential to such learning. In this paper we explore these matters.

2. A Study of Field Failures

Our explorations begin with a study of field problems encountered in the early field use of two novel process machines. The machines focused on were a solder paste profiler and a component placer. Both were developed to automate manual procedures previously used to attach large surface-mounted integrated circuits to circuit boards. The automated process begins with the application of a

tiny dot 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 dots can be as small as 25 thousands of an inch today, and each dot may be as small as the period at the end of this sentence.) Next is inspection (or profiling) of the solder paste - and this is where the first machine that we studied, the "solder 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 dot of solder paste applied to the board is as specified.

If all of the dots of solder paste on a board pass inspection, the board is passed on to the component placing machine - the second of the two machines that we studied. This machine is essentially a robot arm coupled with a machine vision system that picks up integrated circuits (which look like small plastic boxes with two or more rows of tiny metal legs protruding from the bottom) and places them on the circuit board at precisely the right locations, with each metal leg of each component resting exactly on one of the dots of solder paste previously applied to the board. Finally, when the component placing step has been 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.

Both of the machines just described were developed by a major computer manufacturer for use in its own factories, with the development of each machine being an independent event. That is, each was designed and built by different equipment development groups within the computer manufacturing firm, and each was first applied in a different factory of the firm by different process engineers and plant-based users. At the time of our study, both machine types had been installed in several factories. The first solder paste profiler had been placed into service 18 months before data collection began, and the first component placer 2 years before data collection began.

As was noted earlier, unexpected problems often emerge when novel products, processes or services are first introduced to the field, and this proved to be the case for the two machines we studied. Via interviews, we discovered that at the time of our study users had encountered a total of 27 problems (12 affecting the profiler and 15 affecting the component placer) that had: (1) been observed after the machines had been introduced to the field; (2) been considered sufficiently

serious to merit repair; (3) been diagnosed as to cause (although not always fixed) at the time of the study.

Further exploration showed that 22 of these problems (9 affecting the solder profiler, and 13 affecting the component placer) were true surprises to both users and developers - that is, they had only been identified in early field use of the machines. (An example of such a problem: After the component placing machine was installed in the 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.) In contrast, the remaining 5 problems had been known to users and/or developers prior to field introduction, but had not yet been resolved for some reason. (An example of such a problem: The initial specifications for the component placing machine called for it to be able to handle all boards to be processed without needing extra setup. As the 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. Problem Discovery via "Interference Finding"

To pursue our interest in the way field use aids in the discovery of unanticipated problems, we next explored in detail how the 22 unanticipated problems in our sample had been discovered in the factory. We found, first, that the problems were invariably first observed by factory personnel, who would report to the machine developers something like: "The machine stops working (or fails to perform as we want it to) under X conditions: Fix it!" Consider the following example drawn from our sample of cases:

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, users found that it

sometimes failed, and called this to the attention of the machine developers. Development engineers came to the field to investigate, and found 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 the machine developers. While factory personnel knew that the boards they processed varied in color, they had not volunteered the information because they did not know that the developers would be interested. Early in the machine development process, factory personnel had simply provided samples of boards used in the factory to the machine development group. 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.

The yellow board problem illustrates recognition of an unanticipated problem as a consequence of "doing" - operating a machine in its actual use environment. But *how* does field use aid in problem discovery? The question is especially interesting because, given the additional complexity of using equipment in an actual factory environment rather than in a lab, one might expect that the difficulty of problem discovery would increase, not decrease with field introduction. In examining the process of problem discovery, we found a form of learning we call "interference finding" was present in all 22 of our cases of problem discovery through field use.

Interference finding can be described as a form of pattern recognition. A pattern is essentially a set of features or characteristics that describes an object (or event, stimulus, etc.). This bundle of features then may be used as a standard against which one may compare new objects. Thus, one may wish to focus on the *similarities* between patterns in a process called pattern matching. (Systems designed to recognize objects with known characteristics ranging from handwriting to military targets often use algorithms based on pattern-matching.) Or, one may wish to use subtractive pattern-matching to highlight the *differences* between two or more 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 - rapidly moving comets for example.

Interference finding is a form of pattern-matching which is sensitive to the interferences among objects (such as a process machine and a plant environment) that may have very different features or functions. Alexander [6] describes the essence of interference finding when he discusses 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."

The process of interference finding we observed in our sample of process machine problems is a more complex version of the process just described. Here, two very different and highly complex patterns, the new machine and the plant context, are brought in close juxtaposition during field use - "doing." As a result, previously unsuspected and often subtle interferences are discovered because they evoke an obvious symptom - poor machine performance. Thus, in the case of the yellow board problem described earlier, an obvious symptom (machine failure) led developers to discover that they had not properly adapted the machine to the color of circuit boards being processed in the plant.

In problem identification by doing, therefore, we find that the unique contribution of "doing" to problem discovery in the field environment is precisely the precipitation of obvious symptoms. These are then traced via diagnosis [7] to previously unrecognized interferences between machine and use environment.

4. Information Availability and Unanticipated Problems

We next attempted to understand why the 22 problems discovered as a result of field use had not been anticipated earlier. Since the causes of all of the problems in our sample had been diagnosed, we were able to approach this task

knowing both the initial symptom and the "cause" of each problem. (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.) We drew on the diagnosis of each problem to identify the information that would have allowed engineers to resolve each prior to field use - if only that information had been incorporated into the machine as originally designed.

We found (table 1) that the information associated with a problem fell into two major classes with respect to its potential availability to machine developers during the design process. In 15 cases, the information existed in the use environment prior to and during the period that the machines were being designed, and so was potentially available to the machine developers for use in problem avoidance. In the remaining 7 cases, the information that proved problematic was only introduced into the use environment after the machine had been designed and installed in the field.

Table 1: At the time the machine was designed, what was the availability of the information which could have been used to avoid an unanticipated field problem?

Availability 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:			

- (a) users working directly with machine	14	5	
- (b) problem solvers working on other aspects of the production process	11	2	
Totals	9	13	22

Category 1: Problem-identification in stable environments

In cases tabulated under 1(a) in table 1, the information needed to understand or predict problems did exist in the intended use environment during the development of the machine. Indeed 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. But, in the actual evolution of events, the relevance of the information was overlooked until it was made clear by interference finding during use of the machine in the field. The yellow circuit board case example presented earlier illustrates this category of problem.

In cases coded under 1(b) in table 1, the information needed to understand or predict problems was actually present in the machine design lab but, again, its relevance was not seen until made clear by field failure. This was often understandable: "having all the information" did not mean that it was easy to predict the often subtle chain of cause and effect that eventually resulted in an unanticipated field problem. Consider the following example:

Example: Component Slippage Problem

Just before the component placing machine places components on a circuit 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, users 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.

Category 2: Problem-identification in *changing* environments

In the second category of table 1, the information that might have allowed designers to anticipate and forestall a field problem was introduced to the use environment after field introduction of the machine by problem solvers who were not machine developers. In most instances (category 2a) these problem solvers were machine users who, in the course of their field experience with the machine, decided that they wanted something different from the originally specified performance. In many of these cases users experimented with changes to the use environment and/or to the machine itself [8] in order to develop their suggested improvements. 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 who created field problems after the machine was introduced were not machine users - they were individuals 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 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.

In sum, then, we see from table 1 that some information associated with field problems existed at the time the machine was developed, while in other cases the information was created after the introduction of the machine, usually as a result of user learning associated with using the machine in the field.

5. Does Learning Require Doing?

We have seen that problems are discovered in the course of doing as a result of interference finding. In order to understand the need for "doing" in this process, we next apply tests of reason to explore whether it would be possible to obtain the

same learning without actually using process machines in their intended field environments. Rosenberg and also Habermeier [9] have argued that doing or using is required because the possible interactions between products and their use environments are sometimes too complex to be predicted. In what follows, we offer support for this idea and develop it further. We distinguish between situations in which problem-related information is available at the start of a machine (or product or service) design project, and situations in which the problem-related information is only introduced after the machine is in use.

Learning without doing in stable use environments

As noted above, in many cases the information needed to predict field problems exists during the design process. Even in these cases, however, engineers who wish to predict all potential field problems face a difficult task, for two reasons. First, the use environment and the machine that will interact with it contain a myriad of highly specific attributes that could potentially interact to cause field problems. Second, which items among these will actually be associated with problems 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. Since a circuit board - indeed, most components - have many attributes in addition to color (shape, size, weight, chemical composition, resonant frequency, dielectric constant, flexibility, etc., etc.) it is likely that problem solvers seeking to avoid all field failures would have to analyze a very large (perhaps unfeasibly large) number of potentially problematic items and interactions to achieve this.

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 [10], 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 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.). Also, one can use fault trees and other analytical procedures to make the search for possible causes of failure more systematic. Further, 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 [11]. One can also try to incorporate subsystems in one's design which have already been tested under field conditions. Also, 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 use of a new product differ from project to project. Learning by doing is the default strategy - other approaches are simply attempts to anticipate and prevent problems that will otherwise make themselves known through interference finding in the field. Thus, one can expect that designers will invest more or less heavily in the fault anticipation 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 [12].

Learning without doing in *changing* use environments

The problems coded in the second category of Table 1 were created by changes in machine uses or the use environment that occurred after the machine was installed in the field. These changes were 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 is a very significant matter to the designer who is attempting to anticipate and resolve potential field problems without "doing." When, as in the cases discussed just above, 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 in 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 use environment and thus the nature of the desirable solution that the designer is seeking to provide may well change during or after completion of the design process.

When problems are created by autonomous problem-solvers, designers are very unlikely to be able to generate the same information by other means, thus avoiding related field failures and requests for improvement. The autonomous problem-solvers are both posing hard-to-anticipate problems, and are generating an unpredictable set of proposed alternative solutions. Some of these may well involve changes in the machine (or product or service) provided by a particular manufacturer.

Neither game theorists' models of cooperative games (e.g., Axelrod [13]) nor psychologists' models of "mutual adaptation" [14] 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 ill structured. Therefore, the problem-solving path that will be taken by user problem-solvers cannot be predicted by the designers with certainty.

7. Discussion

The approach we have taken to studying learning by doing involved conducting grounded research [18] on multiple instances of a single type of learning by doing event, the identification of an unanticipated problem in a factory. We identified interference finding as a learning mechanism associated with this type of event. We also found that problem identified by learning by doing in our sample were associated with (1) information that existed in the use environment but was "lost in complexity," (2) information that was newly introduced to the use environment.

We have observed interference finding only within a very specific context. Nonetheless, it appears to be quite general, and may therefore be a useful way to describe the process of learning by doing and using in a range of contexts. Interference finding may also prove to underlie a significant proportion of the gains associated with learning by doing. Thus, Mishina [20] analyzed the learning curves associated with the production of the B-17 airplane, and found that learning in production is more closely associated with changes to the production process than with the number of units produced over time. This finding is congruent with a central role for interference finding in learning by doing, because that mechanism applies specifically to adapting to novelty in the production process.

Our findings may also 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. 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 user machine-related learning followed later. If a significant 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 lower level over time. A study by Tyre and Orlikowski [21] of the rate of adaptation of a particular process machine over time shows such a pattern. We propose that the type of micro-level understanding of learning by doing we have pursued in this paper can contribute to a better understanding of learning curves for entire production processes, since these are the aggregate of more micro-level changes.

Our discovery that some of the problems in our sample were caused by changes to the use environment introduced after introduction of the machine has an additional interesting implication for the innovation process. Stable problems with stable causes can eventually be gotten right - although, as we have seen, probably not without learning by doing. Dealing with this type of problem will involve viewing initial implementation as an extension of the innovation process [22]. For example, one might shift 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 such as rapid prototyping that incorporate trial and error in the use environment into the development process.

But problems caused by changes in the use environment after introduction of the machine - primarily due to user learning by doing - will presumably continue to arise. This suggests that one can never get it right, and that innovation may best be seen as a continuous process, with particular embodiments of a product, process or service simply being arbitrary points along the way.

References

[1] Hayes, Robert L. and Kim B. Clark (1985) "Exploring the sources of productivity differences at the factory level in K. B. Clark, R. L. Hayes and C. Lorenz (eds) The Uneasy Alliance, Boston, MA: HBS Press 425-458; Leonard-Barton, Dorothy (1988) "Implementation as mutual adaptation of technology and organization." Research Policy 17: 251-265; Tyre, Marcie and Oscar Hauptman, Oscar (1992), "Effectiveness of Organizational Response Mechanisms to Technological Change in the Production Process." Organization Science 3, 301-321.

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

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

[4] Rosenberg, Nathan.(1982) Inside the Black Box: Technology and Economics. New York: Cambridge University Press, p. 131.

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

[6] Alexander, Christopher (1964), Notes on the Synthesis of Form, Cambridge, MA: Harvard University Press, p. 19.

[7] Tyre, Marcie and Eric von Hippel (1993) "Locating Adaptive Learning: The Situated Nature of Adaptive Learning in Organizations" Sloan School of Management Working Paper #BPS 3568- 93, May.

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

[9] Rosenberg, Nathan.(1982) Inside the Black Box: Technology and Economics. New York: Cambridge University Press, p. 122; Habermeier, K. F. (1990) "Product use and product improvement" Research Policy 19, 271-283.

[10] Allen, Thomas J. (1966) "Studies of the Problem-Solving Process in Engineering Design." IEEE Transactions on Engineering Management EM-13, no.2 :72-83; Marples, David L. (1961) "The Decisions of Engineering Design." IRE Transactions on Engineering Management :55-71.

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

[12] Nuclear Regulatory Commission (1975) Report # 75/014, October.

[13] Axelrod, R. (1984) The Evolution of Cooperation. New York: Basic Books.

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

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

[16] Simon, H. A. (1973) "The Structure of Ill Structured Problems," Artificial Intelligence 4, p. 186.

[17] Marples, David L. (1961) "The Decisions of Engineering Design." IRE Transactions on Engineering Management :55-71; Simon, Herbert A.(1981) The Sciences of the Artificial, Second Edition. Cambridge: MIT Press p. 149.

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

[19] Rosenberg, Nathan.(1982) Inside the Black Box: Technology and Economics. New York: Cambridge University Press, p. 122

[20] Mishina, Kazuhiro, "Learning by New Experiences" Division of Research Working Paper 92-084, Harvard Business School, Boston, MA, May 1992.

[21] Tyre, Marcie and Wanda Orlikowski (1993) "Windows of Opportunity: Temporal Patterns of Technological Adaptation" MIT Sloan School of Management Working Paper.

[22] Leonard-Barton, Dorothy (1988) "Implementation as mutual adaptation of technology and organization." Research Policy 17: 251-265.

[23] Boehm, Barry W., Terence E. Gray, and Thomas Seewaldt (1984) "Prototyping Versus Specifying: A Multiproject Experiment." IEEE Transactions on Software Engineering SE-10, no. 3: 290-303; Gomaa, Hassan. (1983) "The Impact of Rapid Prototyping on Specifying User Requirements." ACM Sigsoft Software Engineering Notes 8, no.2 :17-28.