IS "Maintainability": Should It Reduce the Maintenance Effort?

E. Burton Swanson
University of California, Los Angeles

Abstract

This paper reexamines information system (IS) maintainability *and takes a new and inclusive view of it and its effects. It proposes that an IS is maintainable to the extent that its maintenance, operation, and use is economical in its use of resources. A model is offered which puts IS maintainability in the larger developmental context and suggests a number of related propositions. To the question of whether increased maintainability should reduce an organization's maintenance effort, a straightforward analysis leads to what, for some, may be a surprising answer. Where IS maintainability is enhanced, organizations can sometimes be better off by sustaining, not decreasing, their overall maintenance efforts.*

ACM Categories: H.4.0, K.6.1, K.6.3

Keywords: IS maintenance, IS maintainability

Introduction

Information system (IS) maintenance — keeping an IS operational and responsive to users after it is installed and in production (Martin & Osborne, 1983) — is widely recognized as expensive. For years it has consumed more than half of application software development resources among organizations (Lientz & Swanson, 1980b; Guimaraes, 1993; Gallant, 1986; Nosek & Palvia, 1990; Hanna, 1993). It is viewed by many practitioners as a necessary evil (Couger & Culter, 1985). Significantly, it is charged with creating a "logjam" in new system development. It is seen as something to be improved by having less of it (so that we might have more new system development). How to reduce the "maintenance burden" is the common theme (*The Economist*, 1990).[1]

IS *maintainability* — broadly, the ease with which IS maintenance can be accomplished — is popularly seen as the key to the solution. If application software can be made more "maintainable," for example, by being made less complex (Banker et al., 1991; Kemerer, 1995; Banker et al., 1998), organizations should be able to reduce the bur-

[1]For a review of the "state of software maintenance," see Schneidewind (1987) and Jones (1986) for an examination of the development productivity dilemma.

Acknowledgments

The ideas in this paper were developed in part from seminars first presented at the Turku School of Business and Economics, Finland, and the University of Umeå, Sweden. Olaf Westheider provided research assistance, and Chris Kemerer offered helpful comments and suggestions on an early written version. Others have made useful comments supportive of subsequent revisions.

densome maintenance effort and free needed resources for more new system development. The rationale for IBM's repository-based AD/Cycle used just this argument (Mercurio et al., 1990); proponents of object-oriented technologies promise software reusability in much the same vein (*BusinessWeek*, 1991; Yourdon, 1993).

In this paper, IS maintainability is reexamined in a new light. A model is introduced which places maintainability in the larger developmental context and facilitates our reasoning about it and its effects. The question is posed: can — and should — increased IS maintainability reduce the maintenance effort? A straightforward analysis of the associated economics leads to what, for some, may be a surprising answer. Yes, increased maintainability *can* enable the maintenance effort to be reduced; but, no, reducing the maintenance effort should not always be the action chosen. Where IS maintainability is enhanced, organizations can sometimes be better off by sustaining, not decreasing, their overall maintenance efforts. This paper attempts to explain why and suggests some lessons for practice.

IS Maintainability

While "maintainability" is a concept broadly understood in its application to more traditional machinery, it is less well articulated when applied to software or information systems. There is no commonly accepted measure of software maintainability. While maintainability is asserted to be important, most organizations do not in fact monitor it (Banker & Kemerer, 1992).

Traditionally, a machine's maintainability refers to its ability to be retained in or restored to a specified working condition subject to physical deterioration (Pierskalla & Voelker, 1976). The maintenance of software poses a somewhat different challenge, as the condition subject to deterioration is logical, more than physical, with the software's application logic subject to becoming increasingly entangled and error-prone over time, and even eventually "wrong" for the task where it may have previously been "right." The specification of the proper working condition of software is, at any time, exceedingly complex and elusive. The specification is itself subject to deterioration, as the proper working condition is subject to redefinition through ongoing use. The maintenance challenge is magnified further when the software is incorporated within information systems, where it finds organizational application through human interaction, subject to its own forms of deterioration, for example, with turnover among a system's users which undermines required skills. In this broader context, how should system maintainability be appropriately conceived?

Adapting from our common-sense understanding reflecting ease of accomplishment, the following working definition is proposed. *An information system (IS) is maintainable to the extent that its maintenance, operation, and use is economical in its use of resources.* The definition has three important features. First, it focuses attention on the maintainability of not just software, but information systems more broadly (Edwards, 1984). Second, it broadens the domain of maintainability to include operation and use activities in addition to traditional maintenance work. Where an IS cannot be operated and used economically, it will not be considered maintainable in the organization whatever the narrower maintenance expense. Third, it goes beyond a purely technical perspective, to take a productivity-oriented, economic view of maintenance. For a more conventional treatment of software maintainability, see Kim and Weston (1988), who analyze professionals' assessments of twenty contributing factors (e.g., use of modular design, use of structured programming, use of in-source comments, use of formal software change procedures). The factors focus primarily, although not entirely, on technical matters. See also Martin and McClure (1983), who identify seven characteristics of software maintainability (including its usability) and further provides several practical checklists for assessing it.

Speaking more specifically about maintenance as being economical of resources requires us to specify the task in terms of its inputs and outputs. The task itself includes corrective, adaptive, and perfective components (Swanson, 1976). Corrective maintenance includes all kinds of "fix-it" work and involves human procedures in addition to software and data. Adaptive maintenance is dictated by impending changes in the system's data and processing environments. Perfective maintenance seeks to eliminate inefficiencies, enhance performance, and improve maintainability itself (Swanson, 1976; Lientz & Swanson, 1980b). While corrective and adaptive maintenance are commonly undertaken out of obligation and necessity, perfective maintenance is usually based on economic justification. In particular, users are provided with enhancements, where the anticipated benefits significantly outweigh the costs. Where costs are minimal, small enhancements are often provided without demanding that users justify the benefits. They are simply incorporated within regularly scheduled maintenance. Where costs are significant, users are likely to be asked to quantify the benefits and the enhancements may even be arbitrarily classified as "new development" (Swanson & Beath, 1989b). Because such enhancements often account for about half the total maintenance effort in keeping the system responsive to its users, the maintenance task can be said to consist in large part of the system's "continued development."

Maintenance accomplishment as output thus comprises the completion of a diversified set of corrective, adaptive, and perfective subtasks. Aggregating this accomplishment as a whole requires a common metric and, further, that the subtasks be controlled as to the quality of their completion. Where the focus is on software, it has been proposed that function points and source lines of codes changed may be appropriate metrics for aggregation; and that project tasks can be ranked as to quality subsequent to completion (Banker et al., 1991). Given our own broader definition of maintenance, we will simply assume, for present analytic convenience, that standard service units can be defined across the different subtasks. From a practical perspective, this is obviously a heroic assumption. Fortunately, while it is required for the analysis to follow, it is not needed to be implement in its particulars.

Maintenance effort as input consists of the resources allocated to and expended on the maintenance task. Such resources include machine resources, such as workbenches, in addition to staff resources. This paper, focuses principally on staff resources. As with the task itself, these staff resources may be diverse in terms of skills, experience, and motivation. Aggregating the maintenance effort accordingly requires controls for individual and team differences. Thus, one notable study (Banker et al., 1991) used performance appraisals and records of experience from a bank's personnel system to control for team differences. For this paper, we will simply assume that the total effort can be aggregated by some equivalent means, e.g., by weighting individual hours according to job class and salary.

Overall, then, maintenance can be said to be economical of staff resources to the extent that the maintenance effort is minimized for a given level of accomplishment. Every IS is further associated with a variety of *maintainability conditions* which makes this more or less possible. Among these conditions are (1) *compatibility*, the extent to which the system employs institutionalized data and technology; (2) *integrity*, the extent to which the system provides for reliable, error-free processing; (3) *simplicity*, the extent to which the system invokes relatively few, straightforward procedures; (4) *usability*, the extent to which the system offers convenience and functionality well suited to the organizational task; (5) *extensibility*, the extent to which the system may be extended to meet new requirements and needs; (6) *stability*, the extent to which the system can accommodate environmental change and adaptive interventions; and (7) *familiarity*, the extent to which the system is known by the people who work with it.

The assessment of these conditions for any IS helps to establish its current inherent maintainability. Table 1 elaborates these conditions by means of examples. This list of conditions is merely suggestive, not definitive. While a more extensive discussion is warranted, it is beyond the scope of this paper. Other factors and conditions may be proposed beyond those listed. In the case of a system's software, some, (e.g., Yau & Collofello, 1980) suggest *testability* as a factor, which primarily serves integrity. *Modularity*, which is an important aspect of simplicity and extensibility, might also be suggested. Research evidence is mixed (Vessey & Weber, 1983; Basili & Perricane, 1984; Banker et al., 1993), but modularity may apparently be either too little or too much for maintenance purposes (Kemerer & Ream, 1992).

Imagine, now, that the maintainability of a particular system can be increased, say, by migrating it to a standard platform, thus making it more compatible; or by restructuring its code, thus making it simpler; or through a retraining initiative aimed at making it more familiar. Clearly, the mainten-

Maintainability Conditions	Maintainability Indicator in Development	Maintainability Indicator in Operation	Maintainability Indicator in Use
Compatibility	Current development methods and tools are employed	Standard platform technology is used	Broadly-accepted user interface is employed
Integrity	Software adheres to quality standards and contains few bugs	Operational trouble reports are infrequent	User-entered data are of high quality
Simplicity	Software is structured and modularized	Processing requires minimal operator intervention	Required user routines and skills are easily learned
Usability	User requests for new functionality are well motivated and informed	User requests for operational assistance are minimal	Users employ few "work-arounds" to accomplish tasks
Extensibility	New modules require few changes to existing modules	Platform supports expanded scale of processing	System functionality can be leveraged in new uses
Stability	Software may be modified without introducing new faults	Operator interventions do not bring system down	New peak periods of use do not result in system failures
Familiarity	Software maintainers have been with system from the beginning	Operators have long experience with system	Users have substantial system-specific training and experience

Table 1. Illustrative Indicators of IS Maintainability

ance effort should now be reducible without necessarily reducing the resulting accomplishment. However, maintenance can also be said to be economical of the use of resources to the extent that the maintenance accomplishment is maximized for a given level of effort. Thus, with increased maintainability, it should be possible alternatively to sustain the maintenance effort and thereby accomplish more with it. Under what circumstances might this be desirable? Exploring the answer to this question requires that maintenance be put in a broader context in which new system development makes its own claims upon the same staff resources.

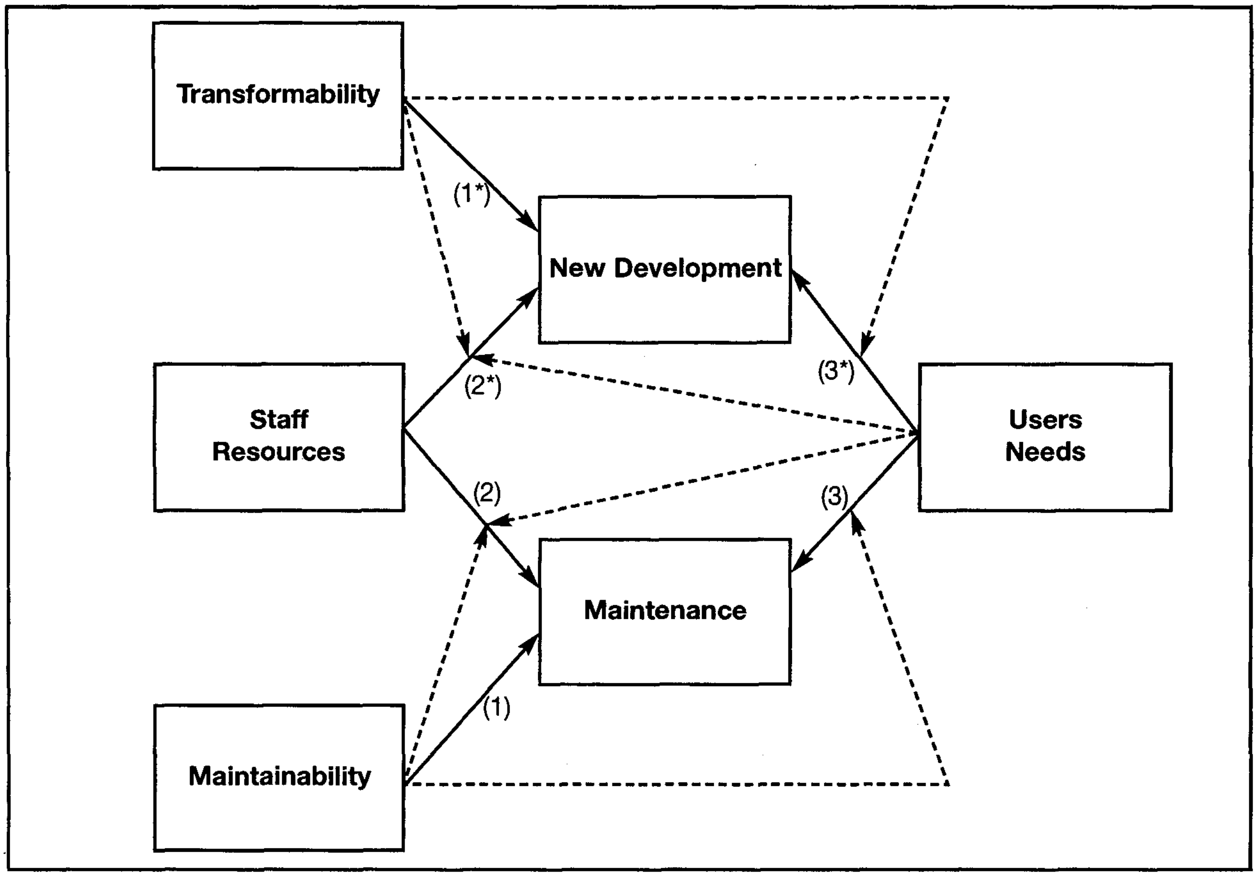
Exploring Effects

IS Maintainability in Context

Can — and should — increased IS maintainability reduce the organization's maintenance effort? Figure 1 presents a model which puts the question in the practical context in which the allocation of effort is characteristically decided. Here, user requests for IS service in the form of maintenance and new system development are understood to drive the process. And, we see that the maintenance effort can be understood only in tandem with the new development effort — the staff resources allocated to, and expended on, the development of new and replacement systems.

For analytic purposes, we distinguish here between users and the IS staff who serve them through their maintenance efforts. In practice, IS staff are often (but not always) organized separately from users. The analysis here requires no particular assumptions about organizational form, however. We further distinguish between user requests for maintenance and new system development. Again, in practice, the distinction may not be clearly drawn, especially at the outset. IS often undertakes its own analysis of service requests, most of which are for maintenance, but some of which after due consideration may lead to new system development initiatives. This complication is an interesting one, but beyond the scope of the present paper. It does not affect the basic argument here.

Maintenance and new development efforts are closely related. Both involve innovation on behalf of organizational users. Maintenance, as "continued development" beyond corrective work, aims in substantial part at incremental innovation and continuous improvement around an installed sys-



Notes:
(i) Solid-line arrows indicate direct actions and effects, while dotted-line arrows indicate informative effects.
(ii) The allocation of staff resources to new development and maintenance represents an informed choice and action, as is the expression of user needs in the form of new development and maintenance requests.
(iii) Maintainability informs both choices and directly affects maintenance and its accomplishment, while transformability also informs both choices and directly affects new development and its accomplishment.
(iv) The numbers refer to propositions associated with the direct actions and effects as they are discussed in the text.

Figure 1. Maintainability in Context

tem. It provides users with enhancements which increase the system's value to them and extend its likely useful life. New development, in contrast, often aims at radical innovation, where the current system (or non-system) will be given up and a new system will enable major change in the way the organization's work is done. The situation is essentially to be transformed, for example, through reengineering. New, and especially replacement, systems development can therefore displace maintenance work; but so too can maintenance enable the organization to forego new development. The model accordingly reflects this basic symmetry.

In one prior study (Swanson & Beath, 1989a), replacement systems are found to be the majority of new systems developed in twelve organizations. Other researchers (Gode et al., 1990) propose a model to determine when to rewrite and replace a system's software. Another author (Chan, 1997) observes that replacement can reduce the maintenance backlog, essentially by absorbing it.

The allocation of staff resources to maintenance and new development is portrayed in Figure 1 as an informed choice. So too is the expression of user needs in the form of maintenance or new development requests. Maintainability informs both types of choices and directly affects maintenance and its accomplishment. Similarly, what may be termed the "transformability" of the current situation informs both and directly affects new development and its accomplishment. The concept of transformability is similar to that of maintainability, except that its application is to radical innovation marked by discontinuity. Although it is not a principal focus of this paper, it is discussed further below.

From Figure 1, we see that the maintenance effort and its accomplishment are accounted for by three major variables: the staff resources available; the system's maintainability; and the users' needs and requests for service. But the maintenance effort is also accounted for by the corresponding effort given to new system development. The allocation of staff to the two efforts is a joint decision.

The model shown in Figure 1 suggests several propositions which bear upon the question of whether increased IS maintainability can and should reduce the effort in maintenance. The first two propositions are basic building blocks and integral to the concept of maintainability as already discussed. First, *Proposition 1: the greater the maintainability of an IS, the more efficient and effective will be the accomplishment of the maintenance task; for any given allocation and expenditure of staff resources, more maintenance will be accomplished.* More maintenance tasks will be completed, increasing the overall level of service. Note that when we say that IS maintainability is a "good thing," this is what we mean by it. Maintainability is fundamentally supportive of maintenance. Because maintenance is made more efficient, the effort allocated to it could in fact now be reduced without reducing the prior accomplishment. But it does not follow that this should be done, as will be seen.

Second, and by implication from the first proposition, *Proposition 2: the greater the staff resources allocated to and expended on maintenance, the more maintenance will be accomplished, for any given level of IS maintainability.* Again, this is straightforward. More maintenance effort gets more done, presumably. But when should more or less effort be exerted? As already noted, this should be an informed choice.

Third, *Proposition 3: the greater the maintainability of an IS, the greater will be the proportion of user needs articulated incrementally through maintenance requests, rather than systemically through proposals for new system development.* This says users appreciate better maintainability too! From the same staff effort, they should be able to get more enhancements accomplished and hence more value added from their requests. Thus, like the consumers of any service where the price is effectively lowered, they are likely to demand more of it. In particular, they are more likely to originate lower-priority service requests which before would have been queued indefinitely, but might now be favorably received. Bear in mind too that maintainability applies not only to the application software, but to system use itself. Thus, users may leverage both the system's usability and their familiarity with the system through useful enhancements.

Transformability and its Effects

Figure 1 shows that a parallel line of reasoning should apply to the question of how transformability can and should affect the effort in new system development, with implications for maintenance. "Transformability" is now defined to complement our earlier definition of maintainability; specifically, *a current situation and its IS are transformable to the extent that new or replacement system development and implementation will be an economical use of the organization's resources.* As seen in Figure 1, transformability affects new development much like maintainability affects maintenance; and both may be expected to inform the allocation of staff resources and the expression of user needs.

Broadly, IS transformability can be understood to shape the opportunity costs of maintenance, just as maintainability shapes the opportunity costs of new system development. Where an IS is easily transformable, for instance, the opportunity costs of continued maintenance may be seen as relatively high. Similarly, where an IS is easily maintainable, the opportunity costs of new system development may appear to be comparatively large.

Following Figure 1, the suggested propositions mirror those above. First, *Proposition 1*: the greater the transformability of an IS, the more efficient and effective will be the accomplishment of the new development task; for any given allocation and expenditure of staff resources, more new development will be accomplished.* Second, *Proposition 2**: *the greater the staff resources allocated to and expended on new development, the more new development will be accomplished, for any given level of IS transformability.* And third, *Proposition 3**: *the greater the transformability of an IS, the greater will be the proportion of user needs articulated through proposals for new development, rather than incrementally through maintenance requests.*

A more probing discussion of transformability is beyond the scope of this paper. However, transformability may be enhanced for a firm by innovation convergence in the marketplace, where packaged software emerges as an attractive alternative to expensive custom-built code, especially where extant systems are to be replaced (Joseph & Swanson, 1998). Transforming local work practices to exploit the opportunities provided by commercial packages and their global business logic has been increasingly irresistible to many firms. The economics of transformability have looked increasingly appealing, and the opportunity costs of continued maintenance have apparently risen for many businesses.

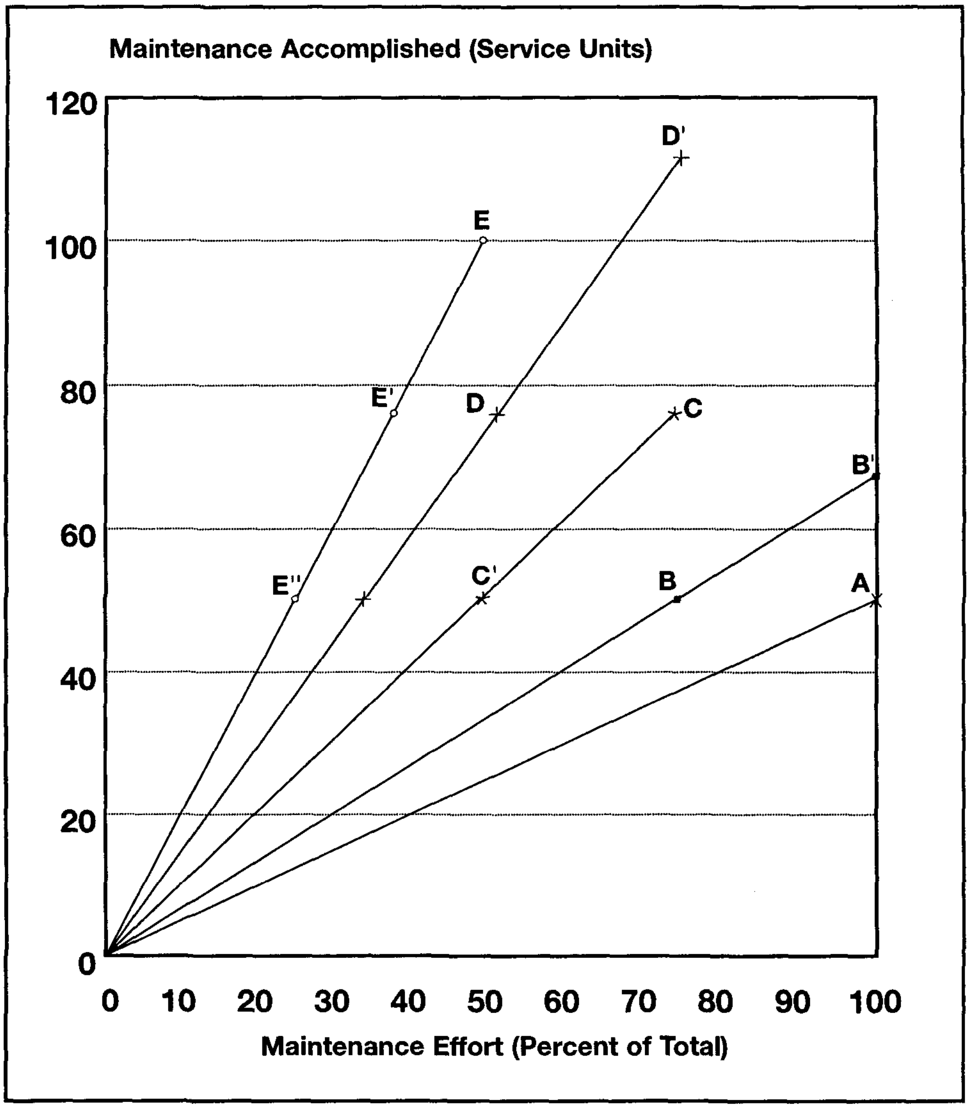
Effects on the Maintenance Task Mix

Having put the issue in context, it is now time ready to address the key question of whether increased IS maintainability should lead to a reduced maintenance effort. Remember that maintenance is a diversified task. Its components are likely to be differently affected. Therefore *Proposition 4**: *the greater the maintainability of an IS, the fewer will be the staff resources allocated to and expended on corrective and adaptive maintenance (because fewer resources can accomplish the same task as before), but the greater will be the staff resources allocated to and expended on enhancements within perfective maintenance (because additional maintenance tasks are now more cost effective).* Thus, the mix of the maintenance task should change, but the overall effort might conceivably increase or decrease or stay the same.

This insight has roots in empirical research. Lientz and Swanson (1980a) first observed from their survey data that the use of development productivity aids was not associated with a reduced level of effort in maintenance. They conjectured that resources freed from corrective maintenance might be redirected toward adaptive and perfective maintenance. A similar finding emerges from an exploratory field study (Dekleva, 1992), where the use of "modern methodologies," contrary to expectations, was not associated with a reduced level of maintenance effort in implementing functional enhancements. The choice among corrective, adaptive, and perfective rests not only on the costs, but on the benefits which accrue to users from the prospective reallocation of effort. In practice, the amount of perfective maintenance is often equal to that of corrective and adaptive maintenance put together, and that the bulk of perfective work goes toward providing enhancements for users (Lientz & Swanson, 1980b; Moad, 1990; Hanna, 1993).

In Figure 2, the consequences of increased maintainability are explored with a graphical illustration. Maintenance accomplished in a hypothetical organization is plotted against maintenance effort for five levels of maintainability, where total staff resources are assumed fixed. Imagine that the organization is initially in condition A, where maintainability is minimal. Imagine further that here one hundred percent of staff effort is allocated to corrective and adaptive maintenance and there is no new development. Suppose maintainability is improved in several steps (such that more may be accomplished from a given level of effort). At each step, it is decided whether to reduce the maintenance effort by reallocating staff to new development, or, alternatively, continue the same level of effort. Why might the organization proceed from condition A to B to C to D to E? In condition E, maintainability is quadrupled from condition A. The overall maintenance effort is halved, but the maintenance accomplishment is doubled. The option of reducing the maintenance effort to 25% (condition E'') is foregone. Instead maintenance is sustained at 50% of the total effort (typical), where enhancements account for half of the maintenance accomplished (also typical).

In condition A, where maintainability is increased in the first step, the organization chooses B over B'; it prefers the benefits of the highest priority new development project to the benefits of the highest priority enhancements to current systems. At the second step, however, the organization chooses C over C'; it prefers the benefits of these same (and additional) enhancements to the benefits of a second new development project. At the



Note: This is an example of improvements in maintainability and their alternative effects. Maintainability in condition A enables 50 service units to be accomplished with 100% effort. Improved maintainability in condition B provides for the same accomplishment with 75% effort, or, alternatively, at B', 67 service units with the 100% effort retained. Choosing between the two alternatives involves weighing their relative benefits. And so on, for further improved maintainability. Note that in condition E, maintainability is quadrupled from condition A. The overall maintenance effort is halved, but the maintenance accomplishment is doubled.

Figure 2. Exploring the Effects of Maintainability

third step, the organization chooses D over D', now preferring the benefits of this second development project to those of further enhancements. But at the fourth step, the organization chooses E over E', preferring the benefits of a second set of enhancements to those of still another development project. Note that the path followed from A to B to C to D to E is merely illustrative and depends wholly on the relative benefits and choices at each step. The suggestion, however, is that the benefits of increased maintainability will, in the long run, be translated into benefits of both new development and maintenance accomplishment. The overall maintenance effort decreases, but the enhancements provided users increase. The magnitudes in each case depend upon the individual situation.

When Staff Resources Are Inadequate

A final issue is how IS maintainability is itself maintained. Recall that tending to the maintainability of a system is itself a form of perfective maintenance (Swanson, 1976). Although, for simplicity, it was not shown in Figure 1, there is a feedback loop from the maintenance task to IS maintainability. What happens when staff resources are not allocated to maintenance in accordance with user needs and requests? When such needed resources are withheld, slack in the maintenance task will be taken up and thus maintainability is likely to suffer.

"Slack" is defined here as resources beyond those needed to accomplish the more immediate and unavoidable aspects of a task. In the case of maintenance, these immediate aspects would include emergency corrective fixes, adaptive changes required in the near term, and enhancements which meet the most pressing needs of system users. Less immediate aspects would include recoding software to improve its processing efficiency or its control structure, and rewriting documentation for better readability by users and analysts. Slack resources, while costly, are thus not necessarily to be minimized; on the contrary, they may be important to delivering the longer-term benefits from a task (Galbraith, 1973).

They may also be usefully reallocated to meet short-term demands. Thus, when user requests for maintenance increase and additional IS staff resources are not made available, the organization can often meet its more immediate needs by tapping its slack resources. These more immediate needs are, by their nature, likely to be compelling. Hence, *Proposition 5*: *the greater the immediate needs of users expressed in their maintenance requests, the greater will be the proportion of the maintenance effort allocated to meet these needs, for any fixed allocation of staff resources.* Maintenance slack will be taken up. An attempt will be made to make do with given resources.

Unfortunately, *Proposition 6*: *The greater the proportion of the maintenance effort allocated to meeting immediate user needs, given a fixed allocation of staff resources, the less may be the proportion of the effort allocated to maintaining IS maintainability.* Where maintenance slack is thus taken up, IS maintainability is likely to suffer in the longer run. Tending to comprehensive maintainability is likely to be neglected. Architectural integrity, in particular, may be compromised through lack of sufficient attention. Users may receive their enhancements, but the longer term viability of the IS may be undermined. Maintainability may therefore be adversely affected, where the maintenance effort is not extended to meet the maintenance demand. Because software maintainability is not easily monitored, in particular (Banker & Kemerer, 1992), more observable accomplishments such as the response to user requests, are likely to dominate, where resources are pinched.

Of course, even where resources are plentiful, maintainability is difficult to maintain. User enhancements usually offer additional functionality, which tends to increase a system's size and, often, its complexity. Especially where enhancements are simply slapped one on top of the other without attention to architectural integration, system structure may crumble and maintainability may decline. In the unhappy extreme, the end result may be contemplated by tracing the reverse path in Figure 2 from E to D to C to B to A. Here users receive no enhancements and there is no new development. The only apparent recourse is to increase the total staff. Thus IS maintainability may deteriorate and work against the organization's interests. It may be much easier to get into position A in Figure 2, than to get out of it – especially where the view of maintenance as a "necessary evil" prevails.

In summary, IS maintainability is seen to be a good thing which, if increased, enables greater accomplishment in both maintenance and new system development. However, it can also attract demands for maintenance which, if met, take up slack in the task where staff resources are limited. In the long run, overall maintainability itself is subject to being neglected and undermined. IS managers thus need to tend to it purposefully, consistent with the availability of staff resources over the longer term.

Tending To Maintainability

While IS maintainability provides short and long term benefits, tending to it requires resources, as with other efforts. In Figure 1, we see that the costs of these resources extend beyond maintenance itself, to include the opportunity costs associated with foregoing new and replacement system development. Tending to IS maintainability therefore involves a systemic decision in resource allocation. Bearing this in mind, there are some practical suggestions for tending to maintainability and securing its benefits; in particular they focus on how managers can better inform themselves about IS maintainability. Thus we conclude on a practical note. However, from a research perspective, there remains much to be accomplished in developing models to enable managers better to tend to IS maintainability. Taking an investment approach, which seeks to establish likely returns to maintainability improvements, would seem to be an especially promising modeling direction, in particular, (See, e.g., Slaughter et al., 1998).

First, managers should take steps to undertake an ongoing systematic assessment of IS maintainability in their organizations. At the most basic level, this requires that the maintenance effort and accomplishment be made more visible so that they may be monitored over time. Making the maintenance effort visible requires, first of all, that staff resources allocated to maintenance be carefully distinguished from those allocated to new system development. This is perhaps most easily accomplished where maintenance is organized separately from new development and staff cannot blur the distinction by arbitrarily charging their times to one or the other (Swanson & Beath, 1990). Also, because maintenance staff themselves will vary in terms of skills and experience as described above, they may be further differentiated, e.g., according to job class and associated salary, for purposes of accumulating and reporting the total maintenance effort.

Making the maintenance accomplishment visible requires the development and use of a system for classifying the component tasks. In the longer run, a fine-grained taxonomy of maintenance work is probably needed (Kemerer & Ream, 1992). Ideally, such a taxonomy would support the development of standard "service units" of maintenance accomplished, so that overall maintainability might be assessed along the lines portrayed in Figure 2. In the shorter run, it may suffice for immediate practical purposes simply to accumulate the completion of subtasks within each category, without attempting to aggregate across subtasks. The accomplishment may be reported on multiple dimensions.[2]

[2]See also Gracy (1987) for a broad, practical approach to measuring and managing maintenance, and Neil et al. (1990) for a case example.

Beyond these first steps, managers might also experiment with alternative "maintenance environments" for supporting maintainability. In the analysis above, it assumed, for simplicity, that maintenance technology is largely fixed. But maintainability is substantially achieved through technology and machine resources. Managers must therefore seek to understand the technological frontiers at which maintenance work can and should be done. To the extent multiple maintenance environments are already in use within the organization, these may be studied and compared and weighed in terms of their costs and benefits for assessing their respective futures. Where new technology and its adoption are contemplated, it may be introduced selectively and studied as a "quasi-experiment" to assess its likely efficacy if more widely deployed.

Lastly, managers might further consider the adoption of an IS health assessment program, whereby individual systems undergo regular examinations of their "health," much as people do (Swanson, 1997). Such examinations can be the vehicle for assessing and giving organizational visibility and importance to maintainability. Through their diagnoses and recommended interventions, they can further be the basis for remedial actions. Finally, they can be used to put maintainability in appropriate comparative context with transformability as discussed above. Current systems can be assessed in comparison to new and replacement system alternatives. Orderly retirement and replacement can thus also be planned.

References

Banker, R. D., Datar, S. M., and Kemerer, C., A. (1991). "Model to Evaluate Variables Impacting the Productivity of Software Maintenance Projects," *Management Science,* Vol. 37, No. 1, pp. 1-18.

Banker, R. D., Datar, S. M., Kemerer, C. F., and Zweig, D. (1993). "Software Complexity and Software Maintenance Costs," *Communications of the ACM,* Vol. 36, No. 11, pp. 81-94.

Banker, R. D., Davis, G. B., and Slaughter, S. A. (1998). "Software Development Practices, Software Complexity, and Software Maintenance Performance: A Field Study," *Management Science*, Vol. 44, No. 4, pp. 433-450.

Banker, R. D., and Kemerer, C. F. (1992). "Performance Evaluation Metrics for Information Systems Development: A Principal-Agent Model," *Information Systems Research*, Vol. 3, No. 4, pp. 379-400.

Basili, V. R., and Perricone, B. (1984). "Software Errors and Complexity: An Empirical Investigation," *Communications of the ACM*, Vol. 27, No. 1, pp. 42-52.

Chan, T. (1997). *Three Essays on the Economics of Software Maintenance*, unpublished doctoral dissertation, National University of Singapore.

Couger, J. D., and Colter, M. A. (1985). *Maintenance Programming: Improving Productivity Through Motivation*, Englewood Cliffs, NJ: Prentice-Hall.

Dekleva, S. M. (1992). "The Influence of the Information Systems Development Approach on Maintenance," *MIS Quarterly*, Vol. 16, No. 3, pp. 355-372.

Edwards, C. (1984). "Information Systems Maintenance: An Integrated Perspective," *MIS Quarterly*, Vol. 8, No. 4, pp. 237-256.

Galbraith, J. (1973). *Designing Complex Organizations,* Reading, MA: Addison-Wesley.

Gallant, J. (1986). "Survey Finds Maintenance Problem Still Escalating," *Computerworld*, Vol. 20, No. 4, p. 31.

Gibson, V. R., and Senn, J. A. (1989). "System Structure and Software Maintenance Performance," *Communications of the ACM,* Vol. 32, No. 3, pp. 347-358.

Gode, D. K., Barua, A., and Mukhopadhyay, T. (1990). "On the Economics of the Software Replacement Problem," *Proceedings of the 11th International Conference on Information Systems,* Copenhagen, pp. 159-170.

Grady, R. B. (1987). "Measuring and Managing Software Maintenance," *IEEE Software,* September, pp. 35-45.

Guimaraes, T. (1983). "Managing Application Program Maintenance Expenditures," *Communications of the ACM*, Vol. 26, No. 10, pp. 739-746.

Hanna, M. (1993). "Maintenance Burden Begging for Remedy," *Software Magazine,* April, p. 53 ff.

Hanna, M. (1990). "How Computers Can Choke Companies," *The Economist*, June 9, pp. 65-66.

Humphrey, W. S. (1989). *Managing the Software Process,* Reading, MS: Addison-Wesley.

Jones, T.C. (1986). *Programming Productivity*, New York: McGraw-Hill.

Joseph, T., and Swanson, E. B. (1998). "The Package Alternative in System Replacement: Evidence for Innovation Convergence," in *Larsen*, T. J., and McGuire, E. Eds. *Information Systems Innovation and Diffusion: Issues and Directions*, Idea Group Publishing, pp. 375-389.

Kemerer, C. F. (1995). "Software Complexity and Software Maintenance: A Survey of Empirical Research," *Annals of Software Engineering*, Vol. 1, pp. 1-22.

Kemerer, C. F., and Ream, A. K. (1992). *Empirical Research on Software Maintenance: 1981-1990*, unpublished working paper, Massachusetts Institute of Technology, May.

Kim, C., and Weston, S. (1988). "Software Maintainability: Perceptions of EDP Professionals," *MIS Quarterly*, Vol. 12, No. 2, pp. 167-185.

Lientz, B. P., and Swanson, E. B. (1980a). "Impact of Development Productivity Aids on Application System Maintenance," *DATA BASE*, Winter-Spring, pp. 114-120.

Lientz, B. P., and Swanson, E. B. (1980b). *Software Maintenance Management*, Reading, MA: Addison-Wesley.

Martin, J., and McClure, C. (1983). *Software Maintenance*, Englewood Cliffs, NJ: Prentice-Hall.

Martin, R. J., and Osborne, W. M. (1983). *Guidance on Software Maintenance*, NBS Special Publication, 500-106, U.S. Department of Commerce.

Mercurio, V. J., Meyers, B. F., Nisbet, A. M., and Radin, G. (1990). "AD/Cycle Strategy and Architecture," *IBM Systems Journal,* Vol. 29, No. 2, pp. 170-188.

Moad, J. (1990). "Maintaining the Competitive Edge, *Datamation*, Vol. 36, February 15, pp. 61-66.

Neil, M., Cole, R. J., and Slater, D. (1990). "Measures for Maintenance Management: A Case Study," *Journal of Software Maintenance,* Vol. 2, pp. 223-240.

Nosek, J. T., and Palvia, P. (1990). "Software Maintenance Management: Changes in the Last Decade," *Journal of Software Maintenance,* Vol. 2, pp. 157-174.

Pierskalla, W. P., and Voelker, J. A. (1976). "A Survey of Maintenance Models: The Control and Surveillance of Deteriorating Systems," *Naval Research Logistics*, Vol. 23, No. 3, pp. 353-388.

Slaughter, S. A., Harter, D. E., and Krishnan, M. S. (1998). "Evaluating the Cost of Software Quality," *Communications of the ACM*, Vol. 41, No. 8, August, pp. 67-73.

Schneidewind, N. F. (1987). "The State of Software Maintenance," *IEEE Transactions on Software Engineering*, Vol. SE-13, No. 3, pp. 303-310.

"Software Made Simple," *BusinessWeek*, September 30, 1991, pp. 92ff.

Swanson, E. B. (1976). "The Dimensions of Maintenance," *Proceedings of the Second International Conference on Software Engineering,* San Francisco, pp. 492-497.

Swanson, E. B. (1997). "Toward Information System Health," in Ruohonen, M. Ed. *Visioiva Valmentaja Verkostoissa*, Turku School of Economics and Business Administration, pp. 107-115.

Swanson, E. B., and Beath, C. M. (1989a). *Maintaining Information Systems in Organizations,* Chichester, England: Wiley.

Swanson, E. B., and Beath, C. M. (1989b). "Reconstructing the Systems Development Organization," *MIS Quarterly*, Vol. 13, No. 3, pp. 293-307.

Swanson, E. B., and Beath, C. M. (1990). "Departmentalization in Software Development and Maintenance," *Communications of the ACM,* Vol. 33, No. 6, pp. 658-667.

Vessey, I., and Weber, R. (1983). "Some Factors Affecting Program Repair Maintenance: An Empirical Study," *Communications of the ACM,* Vol. 26, No. 2, pp. 128-134.

Yau, S. S., and Collofello, J. S. (1980). "Some Stability Measures for Software Maintenance," *IEEE Transactions on Software Engineering,* Vol. SE-6, No. 6, pp. 545-552.

Yourdon, E. (1993). *Decline and Fall of the American Programmer,* Englewood Cliffs, NJ: Prentice-Hall.

About the Author

E. Burton Swanson is a professor of information systems at the Anderson Graduate School of Management at the University of California, Los Angeles. He earned his bachelor's degree at San Jose State, his M.B.A. at the University of Michigan, and his Ph.D. (under C. West Churchman) at the University of California, Berkeley. Dr. Swanson has studied the area of software maintenance for over two decades and is the co-author of the pioneering book, *Software Maintenance Management*.

E-mail: burt.swanson@anderson.ucla.edu