Reviews, Walkthroughs, and Inspections

GERALD M. WEINBERG AND DANIEL P. FREEDMAN

Abstract—**Formal technical reviews supply the quality measurement to the "cost effectiveness" equation in a project management system. There are several unique formal technical review procedures, each applicable to particular types of technical material and to the particular mix of the Review Committee. All formal technical reviews produce reports on the overall quality for project management, and specific technical information for the producers. These reports also serve as an historic account of the systems development process. Historic origins and future trends of formal and informal technical reviews are discussed.**

Index Terms—**Project management, software development management, technical reviews.**

The Problem of Controlling Technical Information

ANY CONTROL system requires reliable information. A project management system normally obtains its information by two quite different routes, as indicated in Fig. 1. *Cost and schedule information* comes in channels relatively independent of the producing unit, and can thus be relied upon to detect cost overruns and schedule slippages. *Evaluation of technical output*, however, is often another matter.

If project management is not in a position to evaluate technical output directly, it must rely on the producing unit's own evaluation—a dangerous game if that unit is malfunctioning. If the unit is technically weak in a certain area, the unit's judgment will be weak in the same area. Just where the work is poorest, the evaluation sent to management will be least likely to show the weakness.

But even if the producing unit is not technically weak, the problem of unreliable information persists because of information overload. As a unit overloads, inadequate supervision may affect work quality—while at the same time affecting the quality of the evaluation. The unit *wants* to be done on schedule and *wants* the work to be correct. Under pressure, any human being will see what is wanted instead of what exists. Just when it is needed most, this control system utterly fails.

The Role of the Formal Technical Review

Formal technical reviews come in many variations, under many names, but all play the same role in project management, as indicated in Fig. 2. As in Fig. 1, the producing unit controls its own development work, perhaps even conducting informal reviews internally. At the level of the producing unit, in fact, the use of the formal technical review requires no change, which simplifies its introduction as a project management tool.

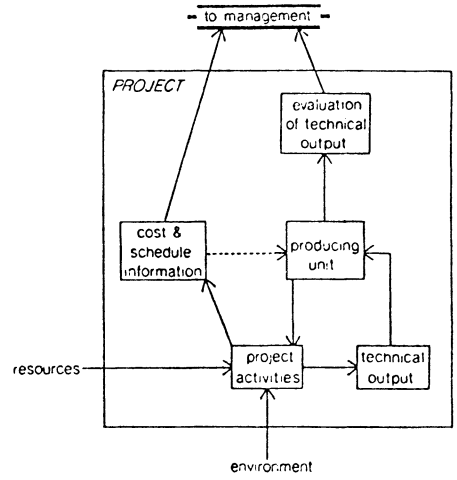

Fig. 1. Management's view of the output of a programming effort.

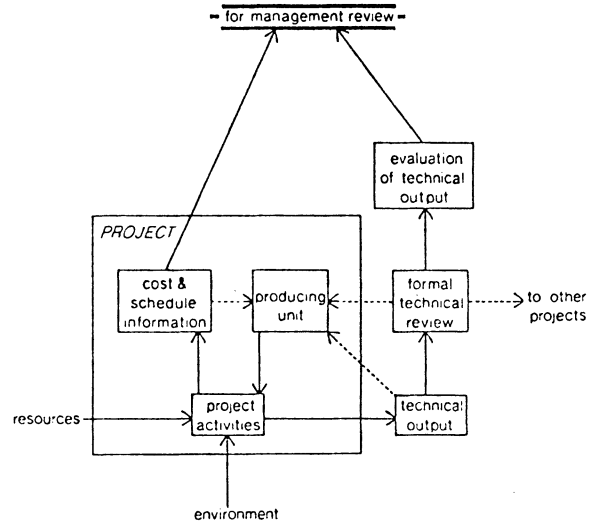

Fig. 2. The place of the formal technical review.

As the diagram shows, the formal review is conducted by people who are *not part of that producing unit*. Hopefully, these are people who have no conscious or unconscious reason for favoring or disfavoring the project's work. Moreover, their report—the technical review summary report—goes to management, thus providing *reliable information* to be used in *management reviews* of the project.

Management Reviews Versus Technical Reviews

Fig. 2 also illustrates the difference between a *technical* review and a *management* review, sometimes called a "project

Manuscript received January 5, 1983.

G. M. Weinberg is with Weinberg and Weinberg, Rural Route Two, Lincoln, NE 68505.

D. P. Freedman is with Ethnotech, Inc., P. O. Box 6627, Lincoln, NE 68506.

review" or some similar name. The technical review committee is staffed by technical people and studies only technical issues. Its job is to put the evaluation of technical output on the same reliable basis as, say, cost and schedule information to management. Using both sorts of information, management can now make informed judgments of what is to be done in controlling the project.

It should also be noted that most "project control" systems do not concern themselves with the accurate and reliable evaluation of the quality of technical output. Instead, they concern themselves with measuring what can be measured *without* technical review, assuming, more or less, that one module of 300 lines of code is just like any other. If that assumption of quality is correct, then these systems can provide excellent management information for project control.

If that assumption is not correct, however, then only the "cost" side of the "cost effectiveness" ledger has any meaning. Under such conditions, even the best project control systems can provide only an illusion of control. The consequences are familiar enough—missed schedules, cost overruns, unmet specifications, inadequate performance, error-prone production, and huge and never ending maintenance.

Review Reports and Project Management

Whatever goes on *inside* it, the major project control function of *any* review is to provide *management* a reliable answer to the fundamental question:

Does this product do the job it is supposed to do?

Once any piece of work has been reviewed and accepted, it becomes part of the system. Subject to a very small risk factor, it is

1) complete,
2) correct,
3) dependable as a base for related work, and
4) measurable for purposes of tracking progress.

Without reviews, there are no *reliable* methods for measuring the progress of a project. *Sometimes* we get dependable reports from the producers themselves, but *sometimes* is not good enough. No matter how good their *intentions,* producers are simply not in a position to give consistently reliable reports on their own products.

For small, simple objects, with well-intentioned, competent producers, there is some chance of success without reliable progress measures. As projects grow larger and more complex, however, the chance of some self-report being overly optimistic becomes a certainty.

Whatever the system of formal reviews, the review reporting serves as a formal commitment by technically competent and unbiased people that a piece of work is complete, correct, and dependable. The review report states as accurately as possible the completeness and acceptability of a piece of software work, be it specifications, design, code, documentation, test plan, or whatever.

By themselves, these review reports do not guarantee that a project will not end up in crisis or failure. It is up to the management of the project to use the information in the review reports to make management decisions needed to keep the project on track. Well done review reports are not sufficient to make a project succeed, though poorly done reviews, or no reviews at all, are sure to get a project in trouble—no matter how skilled the management or how sophisticated the project management system.

Types of Technical Review Reports

The one report that is always generated by a *formal* review is the *technical review summary report.* This report carries the conclusions of the review to management, and thus is the fundamental link between the review process and the project management system.

Other reports *may* be generated. Issues raised that must be brought to the attention of the producers are placed on a *technical review issues list.*

If issues are raised about something other than the reviewed work itself, a *technical review related issues report* is created for each issue.

On occasion, an organization will institute some *research report,* such as a detailed breakdown of standards used and broken, or a report of hits and misses on a checklist.

Those cases where the review leader has to give a report of a failed review (not a failed product) lead to a *review process report,* the form and content of which will be unique to the situation and organization. For instance, on delicate matters the review process report may be verbal.

Other participants may also report on the process of the review itself. For instance, one or more participants might want to object to the behavior of the review leader.

The Review Summary Report

For effective project management, review summary reports must identify three items:

1) What was reviewed?
2) Who did the reviewing?
3) What was their conclusion?

Fig. 3 shows a widely used format containing these items. Although formats vary, the summary should generally be confined to a single page, lest its conclusion be lost in a forest of words.

The Technical Review Issues List

Whereas the summary report is primarily a report to management, the issues list is primarily a report to the producers. The issues list tells the producers *why* their work was not fully acceptable as is, hopefully in sufficient detail to enable them to remedy the situation.

The issues list is primarily a communication from one technical group to another. It is not intended for nontechnical readers and therefore need not be "translated" for their eyes. Moreover, it is a *transient* communication, in that once the issues are resolved, the list might as well disappear. (We exclude, for the moment, research use of the list). Therefore, the issues list need not be fancy, as long as it is clear.

Practices vary, but among our clients, management does not routinely get the issues list. The summary report already contains, in its assessment of the work, a weighted opinion of the seriousness of the issues, so management need not be burdened with extra paper and technical details.

TECHNICAL REVIEW SUMMARY REPORT

SAMPLE

REVIEW NUMBER 3.2.2.1 STARTING TIME 1000
DATE MAY 5, 1976 ENDING TIME 1050

WORK UNIT IDENTIFICATION EDIT-FILE-IB7 TRANSACTION EDITOR
PRODUCED BY PAOLILLO, NOWACKI, AND GARFIELD
BRIEF DESCRIPTION EDITS AND FORMATS INPUT TRANSACTIONS FOR UPDATING SX-PARTS CATALOG

MATERIALS USED IN THE REVIEW: IDENTIFICATION	(check here ___ if supplementary list) DESCRIPTION
COMPILED CODE LISTING	ALL SYNTAX ERRORS CORRECTED
TEST FILE LISTING	INCLUDING ERROR CASES
TEST RESULTS LISTING	SIMULATED BY DEVELOPER, NOT USER
CRITICAL ASSUMPTION LIST	ANY OF THESE WOULD REQUIRE MAJOR
	REVISION OF THE SYSTEM IF CHANGED,
	IN THE OPINION OF THE DEVELOPER.

PARTICIPANTS: NAME	SIGNATURE	I.D.
LEADER J. TAO	J. Tao	------
RECORDER A. MARIETTA	A. Marietta	------
3. P. SCHWARTZ	P. Schwartz	------
4. L. BARMAN	L. Barman	------
5. L. NOWACKI	L. Nowacki	------
6. P. THOMPSON	P. Thompson	------
7.		

APPRAISAL OF THE WORK UNIT:

ACCEPTED (no further review)	NOT ACCEPTED (new review required)
___ as is	✓ major revisions
___ with minor revisions	___ rebuild
	___ review not completed

SUPPLEMENTARY MATERIALS PRODUCED:	DESCRIPTION AND/OR IDENTIFICATION
✓ ISSUES LIST	3.2.2.1 - Wrong type used for price field edit
___ RELATED ISSUES LISTS	
___ OTHER	

Fig. 3. A technical review summary report.

The issues list need not be *concealed* from management, but when managers routinely receive lists of issues, they naturally try to use the information. For example, they may count issues as a means of evaluating producers or reviewers–a practice which tends to undermine the quality of future reviews.

Another common subversion of the review process is the attempt to make the issues list into a *solutions list*. The job of the review committee is to raise issues; the job of the producing unit is to resolve them. A review committee is generally no better at resolving issues than a producing unit is at raising them. Management may want to see issues lists from time to time to ensure that they are remaining issues lists, rather than solutions lists.

The Technical Review Related Issues Report

A related issue is something that comes up in the course of a review that does not happen to be the principal reason for the review. Examples of related issues might be:

1) a typographical error in a related document;

2) a hidden assumption in the specifications that makes part of one module obsolete;

3) a flaw in the original problem statement that makes the entire project plan invalid.

If an organization cannot handle issue 1) without alerting the management chain, it is probably in as bad a shape as an organization that handles issue 3) *without* alerting the management chain. The principal project management problem is with middle issues, such as 2). Such issues have always been troublesome to project control systems. When they are detected, the related issue report is a way of notifying *someone* who ought to be in a position to do something about them.

Because a related issue is, by definition, a deviation from smooth product development, there is really no way to develop a standard practice for handling all situations. A related issue report often descends like a bolt from the blue on some people who may not even know that a review is taking place. If it is not communicated in some standard, official form, people may not even recognize it. Therefore, if we want to keep related issue reports from passing directly into the wastebasket, we have got to give them *some* official status.

The mildest approach is to use a standard *transmittal sheet*, identifying the source of the material and attached to the actual communication, which may take any convenient form. Some organizations prefer a formal follow-up system that requires that each related issue report receive a reply within a few days. Another approach is to send the related issue report through the appropriate manager, leaving any action or follow-up decision on the managerial level.

Historical Analyses

Some of the information obtained from an historical analysis of review reports can be extremely specific. For instance, many organizations classify the types of problems turned up in each review and tabulate the frequency of each type. A similar tabulation is made of errors that slip through the review only to be caught at a later stage.

Comparison of these tabulations–in total, by review group, and by producer–provides clear guidance for future educational and reviewing practices. It is essential, however, that this information be used for improvement of project management, not for punishment of individuals, lest the whole scheme backfire and produce better methods of concealing errors and deficiencies.

To illustrate appropriate use of such historical analyses, let us say that most of the flaws detected during code reviews centered around the module interfaces. If this deficiency was project-wide, the training department could set up special training for everyone, guided by the specific types of interfacing errors recorded in the review reports.

Or perhaps the interfacing errors, upon analysis, reveal a weakness in project standards concerning interfaces. Whatever the problem, the historical records should first make it visible, them make it measurable, and finally help narrow it down to its true source.

Reviews and Phases

Any time after a project begins, an accurate, complete review report history can be compared with the schedule projected at the beginning of the development cycle. In which phases did the estimated time match the actual time? Where did the deviations occur? Were the deviations caused by problems in development? Were they mistakes in the original estimate?

Such historical information is obviously essential if project

management is to improve from project to project. Yet such information will be meaningless if the "phases" of the project plan do not correspond to units of work marked at both beginning and end by reviews.

In order for any project control system to work, the system life cycle must be expressed in terms of measurable phases–some meaningful, reviewable product that represents the end of one phase and the beginning of the next. If there is nothing that can be reviewed, then nothing has been produced, and if nothing has been produced, how can it be controlled?

Varieties of Reviewed Materials

Much of the earliest public discussion of reviews focused on the varieties of *code reviews*, rather than reviews of other materials produced in the life cycle. In the early history of software development, we were primarily concerned with code accuracy, because the coding seemed to be the major stumbling block to reliable product development. As our coding improved, however, we began to see other problems that had been obscured by the tangle of coding errors.

At first we noticed that many of the difficulties were not coding errors but design errors, so more attention was devoted to *design reviews*. As these techniques begin to be effective at clearing up design problems, the whole cycle starts again, for we notice that design is no longer the major hurdle.

In many of these cases, we never clearly understood the problem the design was attempting to solve. We were solving a *situation*, not a problem. Currently, increased emphasis is being placed on the analysis process, which becomes the next area of application of technical reviews–*specification reviews*.

Other types of reviewed material include *documentation*, *test data* and *test plans*, *tools* and *packages*, *training materials*, *procedures* and *standards*, as well as any other "deliverable" used in a system.

Reviews of these materials are conducted not only during development, but also during operation and maintenance of the system.

Principal Varieties of Review Disciplines

It is possible to conduct a review without any particular discipline decided in advance, simply adjusting the course of the meeting to the demands of the product under review. Many reviews are conducted in just this way, but over time special disciplines tend to evolve which emphasize certain aspects of reviewing at the expense of others.

For instance, many of the best known review disciplines are attempts to "cover" a greater quantity of material in the review. The "inspection" approach tries to gain efficiency by focusing on a much narrower, much more sharply defined, set of questions. In some cases, an inspection consists of running through a checklist of faults, one after the other, over the entire product. Obviously, one danger of such an approach is from faults that do not appear on the checklist, so effective inspection systems generally evolve methods for augmenting checklists as experience grows.

Another way to try to cover more material is by having the product "walked through" by someone who is very familiar with it–even specially prepared with a more or less formal presentation. Walking through the product, a lot of detail can be skipped–which is good if you are just trying to verify an overall approach or bad if your object is to find errors of detail.

In some cases, the walkthrough is very close to a lecture about the product–which suggests another reason for varying the formal review approach. In some cases, rapid education of large numbers of people may suggest some variation of the formal review.

In a walkthrough, then, the process is driven by the *product being reviewed*. In an inspection, *the list of points to be inspected* determines the sequence. In a plain review, the order is determined by the *flow of the meeting as it unfolds*. In contrast to these types, the various kinds of "round-robin" reviews emphasize a *cycling through the various participants*, with each person taking an equal and similar share of the entire task.

Round-robin reviews are especially useful in situations where the participants are at the same level of knowledge, a level that may not be too high. It ensures that nobody will shrink from participation through lack of confidence, while at the same time guaranteeing a more detailed look at the product, part by part.

Real Versus Ideal Reviews

Although many "pure" review systems have been described, people who observe actual reviews will never find one following all the "rules." By examining some of the real advantages and disadvantages of one of these "pure" systems, we can understand why every real review system involves aspects of all the major varieties. We will use the walkthrough as our example, but any system could be used to illustrate the same points.

With a walkthrough, because of the prior preparation of the presenter, a large amount of material can be moved through rather speedily. Moreover, since the reviews are far more passive than participating, larger numbers of people can become familiar with the walked through material. This larger audience can serve educational purposes, but it also can bring a great number of diverse viewpoints to bear on the presented material. If all in the audience are alert, and if they represent a broad cross section of skills and viewpoints, the walkthrough can give strong statistical assurance that no major oversight lies concealed in the material.

Another advantage of the walkthrough is that it does not make many demands on the participants for preparation in advance. Where there are large numbers of participants, or where the participants come from diverse organizations not under the same operational control, it may prove impossible to get everyone prepared for the review. In such cases, the walkthrough may be the only reasonable way to ensure that all those present have actually looked at the material.

The problems of the walkthrough spring rather directly from its unique advantages. Advance preparation is not required, so each participant may have a different depth of

understanding. Those close to the work may be bored and not pay attention. Those who are seeing the work for the first time may not be able to keep up with the pace of presentation. In either case, the ability to raise penetrating issues is lost.

Why There is so Much Variety in Reviews

Although all reviews occupy the same role in project management as a control system, managers are justifiably confused by the great variety found in technical review practices. The practice of technical review differs from place to place for a variety of reasons, the principal ones being:

1) different external requirements, such as government contract provisions;

2) different internal organizations, such as the use or nonuse of teams;

3) continuity with past practices.

Continuity is probably the strongest reason. When it comes to social behavior, people tend to resist changing what they already do, even if it does not seem exceptionally productive in today's environment. In many project management systems, formal technical reviews have been introduced as a new form of some old practice, perhaps because it was easier to introduce reviews in this way.

How Reviews Evolved

The idea of reviews of software is as old as software itself. Every early software developer quickly came to understand that writing completely accurate programs was too great a problem for the unaided human mind–even the mind of a genius. Babbage showed his programs to Ada Lovelace, or to anyone else who would review them. John von Neumann regularly submitted his programs to his colleagues for review.

These reviews, in our terms, were *informal* reviews, because they did not involve formal procedures for connecting the review reports to a project management system. Informal review procedures were passed on from person to person in the general culture of computing for many years before they were acknowledged in print. The need for reviewing was so obvious to the best programmers that they rarely mentioned it in print, while the worst programmers believed they were so good that *their* work did not need reviewing.

Around the end of the 1950's, the creation of some large software projects began to make the need for some form of technical reviewing obvious to management all over the world. Most large projects had some sort of reviewing procedures, which evolved through the 1960's into more formalized ideas.

In the 1970's, publication espousing various review forms began to appear in the literature. For those interested in a history of publication, a bibliography appears in Freedman and Weinberg [1]. Publications, however, tend to conceal the grass-roots origin of reviews, giving the impression that they were "invented" by some person or company at a certain time and place.

Where Reviews Are Going

Today, the evolution of reviewing procedures continues, primarily on an experiential basis within projects. Reviews are a partial formalization of a natural social process, arising from the superhuman need for extreme precision in software. Therefore, the "science" of reviewing is a *social* science, and it is difficult to make general, quantifiable statements that apply to all reviews.

Some experimental work has been done on reviews, but these experiments generally suffer from the following problems:

1) Only one or two narrowly defined review procedures are examined.

2) Reviewers are novices in the procedures used.

3) The environment is significantly different from that of a real software development or maintenance environment.

Field reports overcome items 2) and 3), but introduce the problem of experimental control. Nevertheless, many of these reports indicate that effective project management is not possible without the technical review, in one form or another. These reports are sometimes puzzling to managers in other organizations, who have "tried reviews," but who have failed to overcome some of the human problems of changing entrenched social practices.

The best evidence for the effectiveness of reviews is that their use continues to spread. A body of practical knowledge has grown with this spread, particularly concerning the problems associated with starting a system of reviews. We anticipate that most future development of review technology will arise from such on-the-job experiments, rather than any theoretical or laboratory work.

References

[1] D. P. Freedman and G. M. Weinberg, *Handbook of Walkthroughs, Inspections, and Technical Reviews: Evaluating Programs, Projects, and Products*, 3rd ed. Boston, MA: Little, Brown and Company, 1982. (Because this reference contains an extensive bibliography, we are omitting further references here.)

Gerald M. Weinberg, photograph and biography not available at the time of publication.

Daniel P. Freedman, photograph and biography not available at the time of publication.