So you want to save money during software development by using design Reviews? You have heard that reviews allow identification of errors early in the life cycle. You realize that finding errors early on reduces rework costs and increases the probability of meeting your schedule.

Many of us have considered software design reviews, but it is hard to get from the concept of the review to a profitable implementation. Let's take a first step along the road from concept to implementation by putting a name on the software evaluation method that you need (perhaps it is best called a design review).

Our industry abounds with names of evaluation methods: design reviews, technical reviews, inspections, audits, walkthroughs. Do these descriptors name the same method or are they differentiated identifiers for different methods? Once these methods are sorted out, it will become clear how we can use and supplement them in productivity programs.

The software industry traditionally has divided all these methods into two groups: reviews and audits (see Figure 1). If an evaluation is conducted by a technically separated person or group, it is categorized as some type of review.

Technically separated refers to being independent of the creation of the product or the performance of the process. A colleague on the same project can be technically separated. Design and technical reviews, inspections, and walkthroughs are types of reviews when they are performed by technically separated people.

If the evaluation is conducted by a financially separated person or group, it is categorized as an audit. Financially separated refers to the fact that those doing the audit will not receive benefit or penalty from the success or failure of the product or process being audited. Usually auditors come from outside one's company.

The word “audit” connotes both formality and credibility. It should not be allowed to include two members of the same organization looking over one another's work. When customers are told software has been audited, they have the right to expect that it has been quantitatively evaluated by experts who had no financial interest in the product.

**Reviews**


If the object of the review was a management plan or project progress report, the review was classified as management. If the object of the review was a software configuration element (document, source code, test case, and the like), the review was considered technical. As early as the 1960's, these same distinctions were made in the military community by the creators of MIL-STD 1521A, a standard for reviews.

The management and technical categories for reviews are still applicable. People running companies have long realized they have to review schedule and cost issues to survive—but on the technical side, we have been a little slower to incorporate reviews. Now that their value has been demonstrated, we are trying to define them more precisely to make them easier to use.

**Basic question.** To classify software technical reviews, we simply should ask, "What is going to be reviewed?"

ANSI/IEEE Std 730-1984, a standard for software quality assurance plans, calls for a software requirements review, a preliminary design review, a critical design review, and a software verification review to be performed on any pro-

---

**Figure 1. Evaluation methods.**

Robert M. Poston, *Software Standards Editor*

March 1986
By looking at these titles, we see what is going to be reviewed.

But, how should the review be performed? What review technique should be used? Should an inspection, a walkthrough, or some other combination of these review techniques be used? (Note that we can now define inspections and walkthroughs as techniques rather than methods because they describe how to do something instead of describing what should be done.)

**Primary purpose.** To decide what review technique to use, simply specify the primary purpose of the review. The primary purpose of an inspection is to isolate defects. The primary purpose of a walkthrough is to educate the participants about the subject under review. Inspections and walkthroughs may have more than one purpose, but we will use only the primary purpose to classify them.

If we want to hold a software requirements review to acquaint the development team with the requirements, we would use the walkthrough technique. If we want to hold a critical design review to eliminate as many faults as possible, we would use the inspections technique.

Some say the difference between a walkthrough and an inspection is that a walkthrough is formal while an inspection is informal. I do not find that distinction to hold true in the industry.

Frequently, I attend walkthroughs where every role is defined, formal reports are required, and error classification records are maintained. I find that inspections do start out to be very formal at the beginning of most projects, but usually they become less and less formal as the project progresses.

Rework control is often mentioned as another way to differentiate walkthroughs from inspections. In walkthroughs, rework control is said to be at the discretion of the individual doing the rework. In inspections, rework is supposed to be controlled by the review team. However, I have worked with several organizations where all rework control was a configuration management issue—this was true no matter what type of review technique was used.

We often hear of peer reviews in the software industry. A peer review is simply one professional informally checking another’s work. A mini version of either or both the inspection and walkthrough techniques may be applied.

**Audits**

We have already shown that most software evaluations are types of reviews. However, there are some special instances when a software audit may be required. ANSI/IEEE Std 730-1984 identifies three types of software audits: functional audits, physical audits, and in-process audits.

In a functional audit, the product is compared as it is implemented (what does it do?) with the original requirements specifications (what should it do?).

In a physical audit, all representations of the product (requirements documents, test input data, design documents, source code, and the like) are compared. The auditors must determine whether the code and documentation match.

The in-process audit is a consistency check made as the product is evolving.

Definitions of other types of software audits, such as the process quality audit and the quality systems audit, are still developing.

Now that we better understand names and decisions related to software evaluations, we can select those most appropriate for our needs (refer to the decision tree in Figure 2).

**New standard**

The IEEE Working Group on Project 1028, chaired by Charles Hollocker, is trying to go beyond classifying the different audits, reviews, and review techniques. The group hopes to establish an IEEE standard that will define procedures for each review technique.

The procedures they are defining include responsibilities of all the members in a review, entrance and exit criteria for the object of the review, and auditability. They are taking us in some new directions.

The first draft of a *Standard for Software Reviews and Audits* is available from the IEEE Computer Society Press, Draft Standards Dept., 1730 Massachusetts Ave. NW, Washington, DC 20036-1903.

**Prevention**

Technical reviews are valuable. They do find errors early on, and they should be considered in every software project. However, we should spend at least as much time and energy on error prevention as we do on early detection.

Here are four simple techniques that prevent faults from occurring in the first place.

*Fault-boundary problems.* Programmers typically make mistakes handling the...
De facto debit system standard

Applied Communications has agreed to develop software for the Visa consumer credit company that allows automated-teller and point-of-sale networks to link their debit card systems to other networks.

The agreement is intended to further the use of debit cards by making the Visa system the de facto standard. A debit card is like a plastic check — it is used like a credit card, but the cost of whatever is bought with it comes from a checking account instead of a credit line.

The software will be available to anyone, and links between regional and national transaction networks won’t need to be routed through Visa’s transaction network. In turn, Visa will provide its interface specifications to other software firms interested in developing similar software.

Industry-wide standards for debit transactions are being developed by Visa, MasterCard, Plus, and Cirrus under the auspices of the American Bankers Association. The Applied Communications software will be fully conformant to the ISO 8583 financial interchange message formats standard and will support the association’s standards, the software company said.

Applied Communications has agreed to develop software for the Visa consumer credit company that allows automated-teller and point-of-sale networks to link their debit card systems to other networks.

The agreement is intended to further the use of debit cards by making the Visa system the de facto standard. A debit card is like a plastic check — it is used like a credit card, but the cost of whatever is bought with it comes from a checking account instead of a credit line.

The software will be available to anyone, and links between regional and national transaction networks won’t need to be routed through Visa’s transaction network. In turn, Visa will provide its interface specifications to other software firms interested in developing similar software.

Industry-wide standards for debit transactions are being developed by Visa, MasterCard, Plus, and Cirrus under the auspices of the American Bankers Association. The Applied Communications software will be fully conformant to the ISO 8583 financial interchange message formats standard and will support the association’s standards, the software company said.