**Inspection Benefits**

• There are many examples of inspection quality and productivity benefits, and there are no documented cases of poor experience. • This, of course, is partly due to people’s natural reluctance to write about project failure.

The cases in which inspections have not been effective have generally had errors in the way they were conducted. • Either the preparation was not adequate, too many people were involved, the wrong people attended, or too much material was covered at one time. • The biggest single problem is generally the combination of management inattention and schedule pressure. • It is becoming clear, however, that inspections can be highly effective and that they should be widely used in software development and maintenance. • The textbook includes many successful stories. • COBOL program quality before and after code inspection [Table 10.8] – A dramatic improvement that can be expected at the initial use of inspection • Error prevention or detection probabilities [Table 10.9] – What techniques could have been used to prevent or detect errors prior to program shipment? Clearly, inspections are an important way to find errors. • Not only are they more effective than testing for finding many types of problems, but they also find them earlier in the program when the cost of making the corrections is far less. • Inspection should be a required part of every well-run software process. – Inspections should be used for every software design, every program implementation, and every change made either during original development, in test, or in maintenance.

**Future Directions**

• While inspections are highly cost-effective with the quality of the programs generally produced today, they are also labor-intensive. – Each inspection requires the concentrated involvement of a few talented software professionals who together review each element of the product’s design and implementation. – While newer, more productive techniques will likely be found, some form of inspections will undoubtedly be needed if SE remains a human-intensive process.

**Inspection Costs**

• The effectiveness of inspections depends on the time and effort spent in preparing for and conducting them. • Optimum inspection rates depend on the type of product involved and the skill and experience of the people doing them. • Sample Rates for Software Inspections • Relative Inspection Rates with Experience

**The Conduct of Inspections**

• Inspection should be conducted at every point in the development or maintenance process at which intermediate products are produced. • Because they are time consuming, involve people from several groups, and use scarce resources, they must be planned well in advance. • To guarantee that they are done, inspections must be an explicit part of every project plan.

**Initiating an Inspection Program**

• Inspections have been installed successfully in many organizations with very positive results. • The way the inspection program is introduced, however, can have an important impact on its effectiveness.

• The AT&T Bell Laboratories introduced inspections in conjunction with an extensive program of education and consultation (reported in 1984, 1986). – Select a location and a key project for the initial effort. – Introduce the concept of inspections to the managers and key professionals in a three- to five-hour overview session. – Form a working team with one or two project members to determine training requirements, develop the needed forms and procedures, and establish the introduction plan. – If trained moderators are not available, conduct a two- to three-day moderator training class. – A two-day developer workshop is held to introduce the methods and obtain the support of the professional personnel who will use them. – After a couple of months’ experience in conducting inspections, a management seminar is held to outline the results and emphasize the need for continuing management support. – Periodically, the inspection program is assessed and any indicated changes made. After the initial project has obtained some early success, inspections are introduced throughout the organization. • The final implementation step is to incorporate the use of inspections into the organization’s official development process and establish an SQA monitoring program to advise management wheneve r the established procedures are not followed.

**Other Considerations**

Since well-run inspections require intense concentration from all participants, they can be very tiring. • As a result, inspection sessions should generally not exceed about two hours. • Inspections involving the same people should not be scheduled back-to-back since the participants will often be too tired to be fully productive.

• One inspection session a day is generally all that is advisable for any one individual. • The moderator should check on this at the time the inspection is scheduled.

• It is also helpful to assign some inspectors to specific product areas for the project duration. • If the assignment is done early in the design phase and maintained for the entire project, the inspector’s growing product knowledge will greatly facilitate inspection productivity and quality. • It is wise to include some new reviewers in each inspection, so that they may not lose their ability to see the problems.It is rarely desirable to cover two different designs at the same time. – The work should be split into two inspections, even if each will take less than one hour. • It is also wise to focus on the quality of the result being produced rather than on the number of errors being found. – Errors are a fact of life that must be expected. – The need is to improve the methods and tools the SW professionals use so the most prevalent error causes can be reduced or eliminated.

• Some types of errors cannot be found very efficiently by inspections. • If, for example, a large number of minor errors were found during preparation, it is wise to discontinue the inspection and have the producers desk check their work more carefully before starting again. – It is often helpful to ask one or more of the more experienced reviewers to assist the producers while they recheck. – When reviewers become embroiled in minor details, they often overlook more important problems.

**Reports and Tracking**

• There are several reasons for gathering data and making reports on the inspection process. – It is essential to track inspection completions to ensure they are done as required. – Much can be learned about inspection effectiveness from a brief study of the data gathered. – Inspection metrics and data gathering are discussed in more detail in Chapter 15. – Inspection Report [Table 10.4] – Inspection Summary [Table 10.5]

Inspection Report [Table 10.4] – Project – System name – Moderator – Meeting type (overview, reinspection, requirements, design, code) – Number of inspections, inspection duration – Total number of reviewers, inspection prep time – Total lines inspected, pages of diagrams – Disposition (accept, conditional, reinspect) – Reviewers – Producers – Recorder.

• Inspection Summary [Table 10.5] – Project, System name – Moderator – Meeting type (overview, reinspection, requirements, design, code) – Major errors, minor errors (function, interface, data, logic, i/o, performance, maintenance, standards, documentation, human factors, syntax, other) – Distribution (project manager, QA, process group, producers, review coordinator)

**Inspection Training**

• Moderator training courses are essential. • The moderators need a complete grounding in the principles and methods of inspection before they can do a competent job. • This gives them the basic skills and helps to provide the self-confidence needed to lead such a potentially contentious activity.

• As for the participants, training is also highly desirable. • If a competent moderator is available, however, the software professionals can often learn how to be inspectors by participating in well-run inspections.

• Courses should teach inspection principles and provide practice sessions with the checklists and methods involved. • After the initial moderator training needs have been met, it is then desirable to broaden the training program to include all potential inspection participants.

**The Inspection and Post-Inspection Actions**

• In conducting the inspection meeting, the moderator first checks to see if all participants are prepared and obtains copies of any preparation reports not already submitted. • The producer(s) then review each major error either to clarify why it is not an error, to understand what the reviewer(s) meant, or to accept it. • Pertinent data on each error is recorded.

• After discussing all major errors, the product is briefly reviewed to identify any other areas of confusion or concern. • Based on the inspection results, and after asking the reviewers for their views, the moderator decides whether a re-inspection is required. – Sample Re-inspection Criteria

• Sample Re-inspection Criteria [Table 10.3] – Inspection rates unusual: • Inspection time per LOC too short • Inspection time per LOC too long • Too many errors per programmer hour • Too few errors per programmer hour – Error data out of line: • Too many minor errors, and too few major errors (preoccupied with details) • Too many major errors • Unusual error distribution • Too low a percent of errors found during preparation – Other: • Any module with more than N errors (N set in project plan) • Any module with persistently high error rates • The reviewers suggest a reinspection • The moderator suggest a reinspection • The testers suggest a reinspection • The module contains uninspected changes

• Following the inspection, the producer(s) fixes the identified problems and either reviews the corrections with the moderator or in a reinspection. • As the final inspection action, the moderator makes sure that the inspection results and data are inserted in the process database and that management is informed that the inspection has been successfully completed.

**Preparing for the Inspection**

• The full inspection process consists of a preparation phase, the inspection itself, and some post-inspection activity. – The producers and their manager decide that the product is ready for inspection. – The inspection participants are identified, the inspection entry criteria are prepared, and the supporting materials are produced for the opening meeting with the entire inspection group.

The moderator opens this meeting with a brief statement of the subject to be inspected, the inspection objectives, and, if needed, an overview of the inspection process. – The moderator then provides a copy of the inspection package to each of the participants. – Following this introductory meeting the reviewers individually prepare for the inspection. – During preparation the reviewers record their time and the errors identified on the error log form.

**Inspection Participants**

• The moderator (or inspection leader) • The producers – The person(s) responsible for doing the work being inspected • The reviewers (or inspectors) • The recorder (or scribe)

• While many more people may be interested in the inspection results, the purpose of the inspection is to assist the producers in improving their work. • This can best be done by limiting attendance to five or six reviewers. • The key point of this attendance list is that only technical peers attend.

• The moderator is not the manager of the work being reviewed, and neither are any of the other participants. • The inclusion of managers changes the inspection process and distorts the participants’ objectivity. • Regardless of the manager’s behavior, the participants will feel that it is they who are being reviewed rather than the product.

• Inspection data is gathered to see how well the project is progressing, not to evaluate the people. • Since reviewers are human, however, they are subject to error, and managers need to study the inspection data to see where improvements are needed.

**Basic Inspection Principles**

• The inspection process follows certain basic principles: – The inspection is a formal, structured process with a system of checklist and defined roles for the participants. – Generic checklists and standards are developed for each inspection type and, where appropriate, they are tailored to specific project needs. – The reviewers are prepared in advance and have identified their concerns and questions before the inspection starts. Advanced Soft.

• The focus of the inspection is on identifying problems, not resolving them. • An inspection is conducted by technical people for technical people. • The inspection data is entered in the process database and used both to monitor inspection effectiveness and to track and manage product quality

**Examples of Some Major Standards**

• IBM’s Federal Systems Division (FSD) Software Design Practices

– Systematic programming practices • Logical expression • Program expression • Program design • Program design verification

– Systematic design practices • Data design • Modular design

– Advanced design practices • Software system specification • Real-time design

**Inspection Objectives**

• To find errors at the earliest possible point in the development cycle • To ensure that the appropriate parties technically agree on the work • To verify that the work meets predefined criteria • To formally complete a technical task • To provide data on the product and the inspection process.

• Inspections also provide a host of secondary benefits: – Inspections ensure that associated workers are technically aware of the product. – Inspections help to build an effective technical team. – Inspections help to utilize the best talents in the organization. – Inspections provide people with a sense of achievement and participation. – Inspections help the participants develop their skills as reviewers.

**Types of Reviews**

There are many kinds of reviews and many different names for them, so it is important to draw some distinction.

• Types of Reviews – Management review – Technical review – Software inspection – Walkthrough

• Management and technical reviews are generally conducted for management and typically provide information for management action. • Inspections and walkthroughs, on the other hand, are peer examinations aimed at assisting the procedures in improving their work.

• Management review – Ensure progress – Recommend corrective action – Ensure proper allocation of resources • Technical review – Evaluate conformance to specifications and plans – Ensure change integrity.

• Software inspection – Detect and identify defects – Verify resolution • Walkthrough – Detect defects – Examine alternatives – Forum for learning

• Inspection is to examine work technically and provide the producers with an independent assessment of those product areas where improvements are needed. • Walkthroughs are generally less formal and are often conducted in an educational format. • Inspections, on the other hand, generally have a formal format, attendance is specified, and data is reported on the results.

• Types of work products for inspections – Requirements – High-level design – Detailed design – Implementation – Test cases – Documentation.

While there is almost no limit on what can be inspected, there is a question of cost. • Where the cost of inspections does not seem warranted, a less formal walkthrough process is generally adequate. • Technical reviews can also be used for such items as development and test plans.

**Software Inspections**

• Software inspections provide a powerful way to improve the quality and productivity of the SW process. • This chapter provides overview and additional details are included in Appendix C. • The software inspection is a peer review of a programmer’s work to find problems and to improve quality.

• The fundamental objective of inspections is to improve the quality of programs by assisting programmers to recognize and fix their own errors early in the software engineering process. • With large-scale, complex programs, a brief inspection by competent co-workers invariably turns up mistakes the programmers could not have found by themselves. • An error often starts with an early misconception that is repeated in the design, the code, the documentation, and even the testing.

• Inspections help to motivate better work. • When programmers know their work will be critically examined by their peers in an inspection, they are encouraged to work more carefully either to avoid being embarrassed by sloppy mistakes or through the pride of exhibiting a quality work product. • By enlisting others in identifying their errors, programmers end up doing better work themselves.

• Inspections are not magic, and they should not be considered a replacement for testing. • But all software organizations should use inspections or similar technical review methods in all major aspects of their work. – Requirements, design, implementation, test, maintenance, and documentation.

**Software Inspections**

• Types of Reviews • Inspection Objectives • Basic Inspection Principles • The Conduct of Inspections • Inspection Training • Reports and Tracking • Other Considerations • Initiating an Inspection Program • Future Directions

**Standards Versus Guidelines**

• A standard is appropriate when no further judgment is needed. • Standardization makes sense when items are arbitrary and must be done uniformly or when there is one clearly best alternative. – The definition of coding or naming conventions – The selection of a programming language – The use of common design methods

• There are many possible selections, and there may be no clear right or wrong choice, but they must all be done the same way by everyone. – Appropriate subjects for standardization.

• Similarly, some standards are essential to the maintenance of business or technical control. – Standard cost categories – Reporting forms – Change approval procedures – Schedule checkpoints

• The choices are somewhat arbitrary, but standard methods are needed because the support of several different approaches would be expensive and confusing.

• There are also many cases in which standards are totally inappropriate. – Typically, these are cases involving technical judgment. The specification of limits on module size – Clearly it is not wise to establish a standard unless there is convincing evidence that it is always the right thing to do.

• In questionable cases, a guideline should be used, or the standard should have some well-known and practical escape provisions. • While such approaches have much the same effect as standards, they recognize that exceptions occasionally make sense

**Enforcing Standards**

• Standards enforcement is the basic role of the SQA organization. • They do this with a mix of reviews and tests. • Exhaustive reviews are most appropriate when automated tools can be used to support the monitoring process or when the standard is so critical that no single deviation is acceptable. • Otherwise, statistical samples are sufficient unless major problems are encountered.

• Statistical reviews are used for all other standards and procedures. • While the level of sampling should be determined for each case, it is often possible for SQA to provide useful help to development as part of the review. • The effective use of statistical reviews requires that SQA have control over which cases to review and management’s support in follow-up actions when the review finds problems.

**Maintaining Standards**

– Standards must be kept current.

– Standards should be modified and adjusted based on the experience in using and enforcing them, on the changes in available technology, and on the varying needs of the projects.

– If the standards are not maintained, they will gradually become less pertinent to working conditions and enforcement will become progressively less practical.

– If not corrected, the standard will ultimately become a bureaucratic procedure that takes time without adding value.

– The responsibility for maintaining each standard should be assigned to an individual or group.

– This could be the SW Engineering Process Group, or some other group that has the technical capability and the management charter to handle such technical functions.

**Benefits of Standards**

• While there is little quantitative evidence that supports the use of standards, most experienced software managers can cite at least one standard that was key to a program’s success. • While standards alone will not make the difference between project success and failure, they clearly help.

• Thayer (1982) has surveyed SW managers to determine their views on the key software problems and their most effective solutions. – Of all the solutions listed, the use of enforcement of standards and procedures ranked first.

**The Reasons for Software Standards**

• Standards are needed when many people, products, or tools must coexist. • Standards are essential for establishing common support environments, performing integration, or conducting system test. – The fact that everyone knows and understands a common way of doing the same tasks makes it easier for the professionals to move between projects, reduces the need for training, and permits a uniform method for reviewing the work and its status.

• Standards also promote the consistent use of better tools and methods. • When everyone uses common coding conventions and commenting guidelines, for example, it is practical for programmers to review each other’s work and easier for them to understand it. • This both facilitates design and code inspections and improves the maintainability of the finished product.

**Definitions**

• Software terms can be grouped into categories, with the meaning understood in the textbook, as follows:

• Authoritative direction on what is to be done: – Policy A governing principle, typically used as the basis for regulations, procedures, or standards, and generally stated by the highest authority in the organization – Regulation A rule, law, or instruction, typically established by some legislative or regulatory body – Specification The precise and verifiable description of the characteristics of a product A process specification define a method, procedure, or process to be used in performing a task. Specifications are produced by technical experts.

• The characterization of how a task is to be performed or the required characteristics of the result: – Guideline A suggested practice, method, or procedures – Procedure A defined way to do something – Standard A rule or basis for comparison that is used to assess size, content, or value, typically established by common practice or by a designated standards body

• The ways in which tasks are accomplished: – Convention A general agreement on practices, methods, or procedures – Method A regular, orderly procedure or process for performing a task – Practice A usual procedure or process, typically a matter of habit or tacit agreement - Process A defined way to perform some activity

• The focus in this chapter is on: – Guidelines A suggested practice, method, or procedure, typically issued by some authority. – Procedures A defined way to do something, generally embodied in a procedure’s manual. – Standards A rule or basis for comparison that is used to assess size, content, or value, typically established by common practice or by a designated standards body.

**Software Standards**

• A standard is a rule or basis for comparison that is used to assess the size, content, value, or quality of an object or activity.

• In software, two kinds of standards are used: – Describes the nature of the object to be produced – Defines the way the work is to be performed

• Other standards – Languages, coding conventions, commenting, change flagging, error reporting

• Procedures are closely related to standards.

• For example, there are standards for software reviews and audits as well as procedures for conducting them. – A review standard specifies review contents, preparatory materials, participants, responsibilities, and the resulting data and reports. – The procedure for conducting the review describes how the work is to be done, by whom, when, and what is done with the results.

• **Representative Software Standards** – Software quality assurance plans – Software development notebooks – Software development plans – Software reviews and audits – Software requirements – Software design documentation – Software test plans – Software quality assurance reviews – Software configuration management – Problem reporting/corrective action – Software documentation

• **Representative Software Procedures** – Auditing Software development notebooks – Reviewing a software development plan – Conducting software reviews – Conducting software audits – Reviewing software requirements – Reviewing software design documents – Reviewing software test plans – Auditing the software testing process – Conducting SQA reviews – Performing Software Configuration Management (SCM) – Auditing Software Configuration Management systems – Handling problem reporting/corrective action – Auditing problem reporting/corrective action systems – Reviewing software documentation

• **TRW software development policies**, procedures, and standards– TRW has also published a guidebook of DoD regulations, specifications, and standards. – Although it is somewhat out of date, the guidebook does provide a helpful overview of a few military software standards

TRW software development policies, procedures, and standards

**Part III: The Defined Process**• Once organizations have mastered the basic capabilities described in Part II, the performance of their projects will have sufficiently stabilized to permit orderly process improvement.

• The priority needs are then to improve from Level 2 to Level 3.

• This improvement will establish a more consistent and uniform process across the organization and provide a coherent framework for organized learning.

• Part III describes the key topics that Level 2 organizations must address to advance to Level 3.

• The subjects requiring priority management attention at this point (from 2 to 3) are: – Standards – Software inspections – Testing – Advanced configuration management topics – Process models and architecture – Software Engineering Process Group (SEPG)

**The Standards Development Process**

• Standards development involves the following steps:

– Establish a standards strategy that defines priorities and recognizes prior work

– Distribute, review, and maintain this strategy

– Select individuals or small working groups to develop the top-priority standards

– This development effort should build on prior work where available, define the areas of applicability, specify the introduction strategy, and propose an enforcement plan.

– The draft standards should be widely distributed and reviewed.

– The standards should be revised to incorporate the review comments and then re-reviewed if the changes are extensive.

– The standards should initially be implemented in a limited test environment.

– Based on this test experience, the standards should again be reviewed and revised.

– Implement and enforce the standards across the defined areas of applicability.

– Evaluate the effectiveness of the standards in actual practice.

**Establishing Software Standards**

• Before establishing an aggressive standards development program, it is wise to formulate an overall plan that considers the available standards, the priority needs of the organization, the status of the projects, the available staff skills, and the means for standards enforcement.

• While it is important to establish standards, it is also important to concentrate on those standards that can be implemented in a reasonable period and that will provide the most immediate benefit to the organization.

• The establishment of standards starts with an examination of the organization’s standards and procedures needs.

• This should be considered in three categories, and an effort should be made to maintain a balance of emphasis among them: – Management and planning standards and procedures – Development process standards and methods – Tool and process standards

• Management and planning standards and procedures – Configuration management – Estimating and costing – Software Quality Assurance – Status reporting

• Development process standards and methods – Requirement – Design – Documentation – Coding – Integration and test – Reviews, walkthroughs, and inspections.

• The standard development processes

• Maintaining standards

• Enforcing standards

• Tool and process standards – Product naming – Size and cost measures – Defect counting and recording – Code entry and editing tools – Documentation systems – Languages – Library system