Payoff

Improvements in information systems quality are tied to a total quality management (TQM) approach to software development and maintenance. To achieve software TQM, the development process must be continuously improved. A process improvement program begins with process assessment, a systematic evaluation of software development procedures and practices. This article considers the options and opportunities that are available to organizations that intend to conduct a software process assessment.

Problems Addressed

Managers and technical staff in most companies are all too quick to select new methods and tools and proceed toward modern software engineering practice. The problem is that many of these same managers and technical people have a weak understanding of the development and maintenance process that is currently being applied within their organizations. They proceed without a firm foundation or an understanding of where they are. As a result, new technologies sometimes fail to provide the benefits that are expected.

Companies struggle with software engineering because managers fail to understand that a software engineering approach is one part of a broader Total Quality Management philosophy. Even when this fact is understood, some managers never connect the concept of kaizen, or continuous process improvement, to software development activities.

W. Edwards Deming defined quality as striving for excellence in reliability and functions by continuous (process) improvement, supported by statistical analysis of the causes of failure. If an organization wants to improve the quality of its software, thereby enabling information technology to better serve the business, it must focus its attention on improving the process through which software is developed. The starting point is assessment—a look-in-the-mirror approach that enables managers and technical staff to better understand their software development strengths and weaknesses. Process assessment is a first step toward the creation of a viable strategy that will serve as a road map for continuous software process improvement.

A Commonsense Process Improvement Strategy

Process assessment is the initial step in a technology transition cycle that spans many process improvement activities. The cycle begins with assessment and encompasses several other activities, as illustrated in Exhibit 1.
A Technology Transition Cycle for Process Improvement

**Education.**
Most software managers and developers know relatively little about software engineering. To increase the level of software engineering knowledge, an organization must develop an effective education strategy that is tied to the results of the process assessment and that coordinates training content and timing with immediate project needs so that maximum benefit can be attained.

**Selection.**
Selection defines specific goals and criteria for choosing software engineering procedures, methods, and Computer-Aided Software Engineering tools; it leads to the development of a rational mechanism for costing, justifying, and acquiring these important elements of software engineering technology.

**Justification.**
Expenditures for software engineering procedures, methods, education, CASE tools, and associated support activities must be shown to provide a return on investment before money is committed. A justification model is used to demonstrate the bottom-line benefits of process improvement.

**Installation.**
To install software engineering technologies successfully, a transition plan must be devised and executed. The plan defines tasks, responsibilities, milestones, and deliverables and specifies a schedule for getting the work done.

**Evaluation.**
Some managers make changes to improve the development process, select and install new technology, and then stick their heads in the sand, dedicating little time to evaluating whether the technology is working. The evaluation step initiates an ongoing assessment of the CASE/software engineering installation process.

All these steps define transition strategy, and they all depend on a successful process assessment. Information sources that address software engineering technology transition are listed in this article's Appendix. In the remainder of the article, the first step—process assessment—is considered in greater detail.

**Objectives of a Process Assessment**
Although informal software process audits have been conducted for many years, the use of a formal process assessment is relatively new, and it was not until process assessment was endorsed by the Software Engineering Institute (SEI) that major corporations and government agencies began to adopt the practice.

The term *process assessment* refers to both qualitative and quantitative information gathering. When process assessment is properly conducted, it satisfies its objectives by:

- Providing a framework for an objective examination of an organization's software development practices.
· Indicating technical and management strengths and weaknesses in a way that allows for comparison to industry norms.
· Indicating the relative software development maturity of an organization.
· Leading to a strategy for process improvement and indirectly, to the improvement of software quality.

**Process Attributes**

To accomplish these objectives, the process assessment approach should be designed in a way that probes each of the following process attributes:

· Organizational policies that guide the use of software engineering practices.
· Training that supports the use of procedures, methods, and tools.
· The framework (procedural model) that has been established to define a software engineering process.
· Quality assurance (QA) activities for software.
· Project management tasks that plan, control, and monitor software work.
· Software engineering methods that allow technical staff to build high-quality applications.
· CASE tools that support the methods.
· Software metrics and measurement that provide insight into the process and its product.

**Structure of a Process Assessment**

Although there are many different process assessment approaches, all have the same basic structure. First, a set of questions that probe process maturity are asked and answered. The questions may focus solely on procedural issues or may delve into the application of software engineering technology. Responses to the assessment questions are evaluated and a process maturity level is computed. The maturity level represents an organization’s commitment and adherence to sound software engineering and QA practices. Finally, the results of the assessment are interpreted and used to develop a process improvement strategy. Interpretation may be global or may target specific process attributes.

**Assessment Questions**

Assessment questions are designed to enable an assessor (who may be an outside consultant or staff members drawn from the organization undergoing assessment) to gather enough information to understand the software organization, the application of technology within it, and the relative sophistication of the project management framework
for applying the technology. An effective software engineering process assessment approach uses three types of questions: qualitative, Boolean, and quantitative.

**Qualitative Questions.**

Questions in this category require a narrative explanation. Some qualitative questions are:

- Describe how project teams are formed. Is a functional or matrix organization used?
- Describe or identify the customers for software within the organization.
- Describe the relationship between the customer and the people who develop software. Who initially specifies products with software content? To what degree are software development practices understood by the customer? What communication problems occur between customers and the software engineering organization?
- Describe the role of quality assurance, manufacturing, and service organizations with regard to software.
- Describe individual software development tools (available as operating system features and as standalone functions) that are used during software development.

**Boolean Questions.**

Questions in this category elicit a yes or no response. Boolean questions are used to assess the following three areas:

- **Design.** Does the software development organization use a specific method for data design? For architectural design? Is procedural design constrained to the use of the structured programming constructs? Is there a defined method for human-computer interface design?

- **Programming and coding.** Is more than 90% of code written in a high-order language? Are specific conventions for code documentation defined and used?

- **Testing.** Are specific methods used for test case design? Does test planning begin before code is written? Are the results of testing stored for historical reference? Are there mechanisms for routinely performing regression testing and for ensuring that testing covers all software requirements?

**Quantitative Questions.**

Questions in this category enable an organization to obtain numerical information that can be used in conjunction with software metrics to compute costs and potential payback for new technology. The following information is representative:

- Annual revenue reported by your component.
- Annual budget for data processing or IS.
- Annual budget for engineering/product-oriented software development.
- Annual budget for software-related training.
- Annual budget for computer hardware.
- Annual budget for software tools (differentiate between hardware and software).
- Number of systems and software practitioners in all application areas.
- Number of IS people by job category.
- Number of software people working on engineered products and systems.
- Current number of outside contractors working on software in-house.
- Percentage of software people working on maintenance.
- Projected growth or decrease for each aforementioned item.

Most assessment questionnaires are organized in a way that probes specific process attributes (e.g., Software Quality Assurance or project management approach); most suggest a grading scheme for responses so that relative strengths and weaknesses can be ascertained; and most address both management and technical topics. The structure of the questionnaire, the types of questions asked, the grading scheme that is proposed, and the usefulness of the results are determined by the overall process assessment model (examples of which are discussed in the section headed “Process Assessment Models”).

Response Evaluation

The responses to the assessment questionnaire are evaluated to determine the process maturity level. Although specific evaluation approaches vary, the following steps are common:

- Responses to Boolean questions are used to derive a maturity value. Maturity values may be based on a simple count of yes/no responses, on a specific set of questions that must be answered positively to achieve a given maturity level, or on a weighting scheme that defines the maturity level of an organization that answers a specific question positively.

- Responses to quantitative questions are compared to industry averages, when available. Both quality and productivity data are collected, and compared averages are published in the technical literature.

- Responses to qualitative questions are used to derive additional insight into the current process. By documenting local conditions and constraints, qualitative responses establish a baseline for interpretation.
Interpreting the Results

The maturity values computed from the responses to Boolean assessment questions can provide a means for developing a transition plan for process improvement. Ideally, maturity values are assigned to one of the several process attributes. On the basis of each maturity value, an organization can rank process attributes according to their importance and impact on local efforts to improve process. After priorities have been assigned for process attribute areas, interpretation begins with the goal of developing an organizationally specific set of findings and recommendations. Findings describe specific areas of strength or weakness; recommendations define the actions required to improve the software development process.

Process Assessment Models

A process assessment model defines the overall structure and logistics of the process assessment, the organization and application of assessment questions, the process attributes that are considered during the assessment, and the manner in which process maturity is determined. Assessment models can be broadly categorized as follows:

- Models developed by large companies and originally intended for internal use. Examples are Hewlett-Packard's Software Quality and Productivity Analysis (SQPA) and Bell Canada's Software Development Capability Assessment Method.
- Models developed as an adjunct to consulting services, such as Howard Rubin Associates, R.S. Pressman & Associates, Inc., Software Productivity Research, Inc., and many others.
- Models developed by government/industry consortiums. The Software Engineering Institute Capability Maturity Model is the best-known of these.
- Models packaged as do-it-yourself products for use by any software development organization.

In addition, the International Standards Organization (ISO) is currently at work on a standard for software engineering process assessment to ensure compliance to ISO 9000 quality standards. At present, no assessment model meets all of the proposed requirements for the ISO assessment standard.

A detailed discussion of all types of assessment models is beyond the scope of this article. However, to provide a further understanding of the assessment approach, two representative assessment models are considered in the following sections.

The SEI Assessment Model

The Software Engineering Institute's comprehensive assessment model is predicated on a set of software engineering capabilities that should be present as organizations reach different levels of process maturity. To determine an organization's current state of process maturity, the SEI uses an assessment questionnaire and a five-point grading scheme. The grading scheme provides a measure of the global effectiveness of a company's software engineering practices and establishes five process maturity levels:
Level 1: Initial. The software process is characterized as ad hoc. Few processes are defined, and success depends on individual effort.

Level 2: Repeatable. Basic project management processes are established to track cost, schedule, and functionality. The necessary process discipline is in place to repeat earlier successes on projects with similar applications.

Level 3: Defined. The software process for both management and engineering activities is documented, standardized, and integrated into an organizationwide software process. This level includes all characteristics defined for level 2.

Level 4: Managed. Detailed measures of the software process and product quality are collected so that both the software process and products are quantitatively controlled. This level includes all characteristics defined for level 3.

Level 5: Optimizing. Continuous process improvement is enabled by quantitative feedback from the process and from testing innovative ideas and technologies. This level includes all characteristics defined for level 4.

To achieve specific levels of process maturity, selected questions from the SEI questionnaire must be answered positively. The SEI has associated key process areas (KPAs) with each of the maturity levels. KPAs describe those software engineering functions that must be present to constitute good practice at a particular level. Eighteen KPAs are defined across the maturity model and are mapped into different levels of process maturity. Assessment questions are designed to probe for the existence (or lack thereof) of key practices that reveal whether the goals of a KPA have been achieved.

The SEI approach represents a significant achievement in process assessment, but it has some drawbacks. Although detailed analysis of the assessment questionnaire can lead to an assessment of the efficacy of key process areas and related key practices, the maturity level alone tells little about individual KPAs. The process maturity level is computed in a way that causes a low grade if specific questions are answered negatively, even if other questions that represent reasonable sophistication are answered with a yes. The SEI questionnaire is sometimes criticized for underemphasizing the importance of technology and overemphasizing the importance of policies and standards. Consultants who are accredited assessors are usually needed to provide the additional detail and insight that is missing with the SEI questionnaire alone.

The assessment model proposed by the SEI represents the industry's most comprehensive look in the mirror. It requires broad-based organizational commitment, an assessment budget in the thousands of dollars, and the presence of accredited assessors to do the work.

The Process Advisor Assessment Model

The Process Advisor assessment model enables self-directed assessment for those organizations that want to begin software engineering technology transition activities without incurring substantial initial expense. Unlike the SEI assessment questionnaire (which contains only Boolean questions), the Process Advisor model incorporates qualitative, quantitative, and Boolean questions. The qualitative and quantitative assessment questions follow the structure discussed earlier in this article. Responses to these questions are assessed using a quasi-expert system that is built into the
model. Each response to the questionnaire is compared to a set of typical responses. The quasi-expert system provides a set of inferences that help an organization to develop findings and recommendations based on the response.

Boolean questions address eight process attributes: organizational policies, training, software development process, quality assurance, project management, software engineering methods, Computer-Aided Software Engineering tools, and software metrics and measurement. Responses to the Boolean questions generate process attribute grades for each of the eight attributes. These form a process maturity footprint for a software organization.

The process maturity footprint (Exhibit 2) indicates the relative strengths and weaknesses in the process attribute areas and enables an organization to compare itself with common software engineering practice and best practice, which is the level of software engineering practice found in the top 10% to 15% of all software development organizations. Process Advisor also provides guidance for establishing an education strategy, creating technology selection and justification models, and building an effective transition plan.

**A Process Attribute Footprint**

**Process Maturity**

The majority of assessment models (including the two described in the preceding sections) enable an organization to compute its process maturity. Management must then know how to use this number.

All too often, a senior manager decides that a specific process maturity level should become an organization goal. That is, an organization that currently has a process maturity of 1.0 is chartered with becoming a 3.0 organization within 24 months. Although there is nothing inherently wrong with setting process maturity goals, focusing solely on improving the process maturity value misses the point. The goal of every software development organization should be to improve the quality of the applications it builds, to satisfy its customers and users, and to accomplish work on time. Improving process maturity helps in achieving these goals, but it should not become the goal.

In general, process maturity (and process attribute grades) should be used in the following ways:

- To target areas of strength and weakness.
- To raise management's consciousness.
- To define areas in which further investigation (e.g., assessment meetings with relevant staff) may be needed.
- To provide a comparison to industry common and best practice.
- To serve as a baseline for reassessment later in the transition life cycle.

Using process maturity in these ways, an organization can establish a foundation on which the technology transition plan is built.
Developing Findings and Recommendations That Improve Process Maturity

Findings and recommendations are derived from the results of the assessment. However, it is sometimes difficult to interpret the assessment results in a manner that leads to pragmatic recommendations for change. A self-directed assessment approach must provide a set of inference-based guidelines that are tied to different maturity levels for each of the process attributes under assessment. Once the assessment has been completed, the maturity grade for each process attribute is determined. The grade range provides a solid indication of both finding and recommendations. To illustrate inference-based guidelines, the sample findings and recommendations are reproduced from the Process Advisor workbook.

The Software Development Process

Questions in the software engineering process section of the questionnaire focus on standards as a way to determine whether an organization has codified its approach. Examine the grade and place it in the context of the grade ranges:

<table>
<thead>
<tr>
<th>Grade Range</th>
<th>Identifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Below 1.65</td>
<td>E</td>
</tr>
<tr>
<td>1.65 to 2.25</td>
<td>D</td>
</tr>
<tr>
<td>2.26 to 2.75</td>
<td>C</td>
</tr>
<tr>
<td>2.76 to 3.25</td>
<td>B</td>
</tr>
<tr>
<td>Above 3.26</td>
<td>A</td>
</tr>
</tbody>
</table>

Interpretation

**Grades E and D.**

It is unlikely that the organization has developed a written description of its process or defined a process in any explicit manner.

**Action.**

The organization should create a skeletal framework for software engineering—that is, a set of activities, deliverables, milestones, and QA actions that can be applied as software is being developed. A description of the framework is then needed to solicit comments and recommendations from managers and technical staff. Over time, the framework should be reworked and more detail added until it evolves into a standard.

**Grades C and B.**

The organization has codified many of the activities associated with software development. It is likely that the same approach is applied across different projects and that project planning, control, and Software Quality Assurance are easier to achieve as a result. However, just because standards exist does not mean that the process is effective or properly characterized.
Action.

Each of the standards should be reviewed to determine whether it reflects modern software engineering practice and whether there are aspects that can be streamlined or that do not work well. Time should be spent polling development staff to determine whether the standards are being used as widely as these grade ranges imply. Specific technical areas without standards can be determined by reviewing responses to individual questions. It may be worthwhile to develop a framework approach for a specific technical area (e.g., testing) in a manner similar to that described in the action paragraph for E and D ranges.

Quality Assurance Activities

Questions in the QA activities section of the process assessment questionnaire explore an organization's emphasis on documentation, reviews, and other QA functions. Examine the grade and place it in the context of the grade ranges (listed in the previous section). If one or more of the subsection grades are significantly different than the overall section grade, further investigation into that area is warranted.

Interpretation

Grades E and D.

Software quality and the activities needed to ensure it are not a primary focus within the software development organization. Documentation is probably weak, because there are no standard formats to guide developers. Effective reviews are not being conducted and the results of reviews are not applied to improve the process. Software Quality Assurance is not a formally defined activity.

Action.

The organization should develop a plan to improve documentation, reviews, and software quality assurance. Beginning with documents and reviews, the first action is to pick one or two documents and develop a standard format (being brief is best) and then develop a set of review guidelines for them. Over time, the actions can be broadened until most important documents are defined, are being produced, and are being reviewed.

Grade C.

The organization's approach to predictable documentation, effective reviews, and basic quality assurance activities is coming together.

Action.

The organization should review responses to each of the subsections to determine which areas need the most improvement. Quality assurance functions are likely to need further improvement; if so, focus on establishing mechanisms for ensuring compliance with documentation and process standards. The organization might also broaden its review approach, if this can be done cost-effectively. At the same time, Computer-Aided Software Engineering tools should be employed to create effective documentation in a more productive manner.
Grade B.

The organization is at the state of practice in the QA area. However, it may not be using quantitative data to analyze the software engineering process.

Action.

One idea to consider is a fledgling program in statistical QA for software. By first collecting data on defects uncovered through other QA activities, the organization can work to improve methods to reduce defects. It can then acquire tools that will enable the organization to build quality software more effectively.

Managing Process Assessment

The process assessment may be conducted by an internal consulting organization, by outsiders, or in a self-directed fashion. Regardless of the approach that is chosen, the assessment must be coordinated by IS management.

It is vital that management prepare an organization for the process assessment activity before it begins. The intent of the assessment, and the benefits to be derived from it, should be communicated to technical managers and staff. The types of assessment data to be collected and the use of assessment data should be thoroughly explained. The results of the assessment should be shared with all participants. All involved must understand that the purpose of the assessment is to establish a basis for process improvement, not to punish or judge technical proficiency.

Recommended Course of Action

If an organization is committed to software development process improvement, its first step is to assess the current state of software engineering practice. The IS manager is advised to:

- Set the stage for process improvement by providing preliminary training in software engineering for managers, technical staff, and users. Training should introduce process and technology options, emphasize the benefits of improved software quality, and explain the technology transition cycle.

- Select a local champion to manage the assessment approach from the inside. The local champion should drive the assessment process as well as facilitate those who are conducting the assessment and those who are taking part.

- Select an assessment approach that is appropriate for the organization, its budgets, and its management philosophy. If budgets are limited or outside consultants cannot be used, consider the use of a self-directed assessment model. If consulting budgets are available, hire a competent consulting firm to conduct the assessment. If customer requirements dictate a specific assessment approach (e.g., many government contractors are encouraged to use the Software Engineering Institute assessment), develop a plan that enables that approach to be conducted.

- Determine the software development process maturity level for the organization. The process maturity level makes it possible to compare the organization with common and
best practice in the industry. However, it should not be used to compare internal groups or to force technology transition.

- Examine findings and recommendations derived as a consequence of assessment results to be certain that they accurately reflect the organization. Recommendations must be realistic.

- Begin the development of a transition plan that will lead to software process improvement. The transition plan describes the education strategy, the approach to be taken for selecting process changes and technology upgrades, and the tasks, milestones, and deliverables associated with process and technology installation.

**Bibliography**


**Author Biographies**

Roger S. Pressman

Roger S. Pressman, PhD, is an internationally recognized consultant and author in software engineering. He has worked as a software engineer, a manager, a professor, and a consultant. He is president of R.S. Pressman & Associates, Inc., a consulting firm that specializes in helping companies establish effective software engineering practices and is the developer of Process Advisor, a self-directed software engineering technology transition product. He can be reached by phone at (203)795-5044 or by fax at (203) 799-1023.