Technical Report

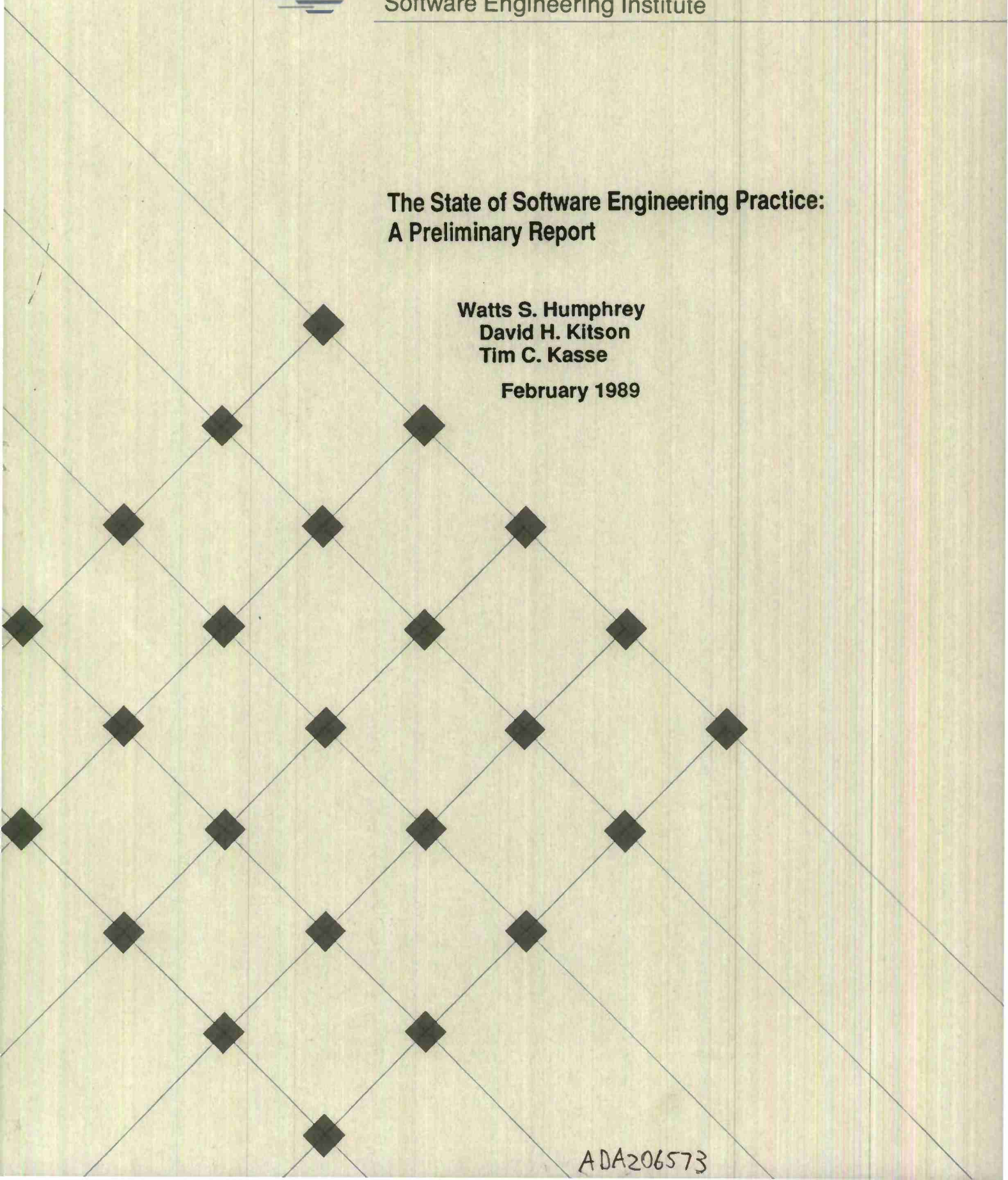
CMU/SEI-89-TR-1
ESD-TR-89-001

Carnegie-Mellon University
Software Engineering Institute

The State of Software Engineering Practice: A Preliminary Report

Watts S. Humphrey
David H. Kitson
Tim C. Kasse

February 1989



Technical Report
CMU/SEI-89-TR-1
ESD-TR-89- 001
February 1989

The State of Software Engineering Practice: A Preliminary Report

Watts S. Humphrey
David H. Kitson
Tim C. Kasse

Software Process Assessment Project

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

This technical report was prepared for the

SEI Joint Program Office
ESD/AVS
Hanscom AFB, MA 01731

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

Review and Approval

This report has been reviewed and is approved for publication.

FOR THE COMMANDER

Karl H. Shingler
SEI Joint Program Office

This work is sponsored by the U.S. Department of Defense.

Table of Contents

List of Tables

List of Figures

Acknowledgments

The authors acknowledge the efforts of those individuals (both within the SEI and from organizations which support the SEI) who contributed to the work described in this report. In particular, we acknowledge the contributions of Ken Dymond, who performed various analyses of the assessment data and provided the authors with relevant information and graphics. Also, we acknowledge Linda Pesante, who significantly improved the readability of the report through her technical editing skills. We extend a special thanks to all the individuals and sponsoring organizations that were involved in the various assessments, and to our colleagues at the SEI who reviewed the various drafts leading to this report. We particularly appreciate the efforts of the members of the final technical report review team: Ken Dymond, Ron Higuera, Mark Paulk, Rich Pethia, and Bill Sweet.

The State of Software Engineering Practice: A Preliminary Report

Abstract: This is the first in a series of SEI reports to provide periodic updates on the state of software engineering practice in the DoD software community. The SEI has developed, and is refining, a process framework and assessment methodology for characterizing the processes used by software organizations to develop and evolve software products. This report provides a brief overview of the process framework and assessment approach, describes assessment results obtained to date, and discusses implications of the current state of the practice for both customers and suppliers of DoD software.

Executive Summary

The Software Engineering Institute (SEI) was established by the U.S. Department of Defense (DoD) to transition improved software methods into general practice. As part of this mission, work is under way to characterize and report on the state of the practice of software engineering in the DoD software community.[1] Preliminary results of this work indicate that the majority of the software organizations in this community are operating at an immature level of software process maturity.[2]

In a mature software process, people, methods, techniques, and technology are effectively and efficiently coupled to consistently produce quality software within the constraints of cost and schedule requirements. In an immature software process, costs and schedules are largely unpredictable, quality is generally marginal, and technology is often used ineffectively. Specifically, organizations with immature processes are often deficient in one or more of the following areas:

- project planning
- project management
- configuration management
- software quality assurance

[1]By *DoD software community* we mean DoD agencies and DoD contractors engaged in the acquisition, production, or maintenance of software.

[2]Because of the way organizations were selected for inclusion in this study, the data upon which the report is based does not necessarily constitute a *statistically valid measure* of the state of DoD software community practice.

Software professionals generally need most help in controlling requirements, coordinating changes, managing (and making) plans, managing interdependencies, and getting help on systems design issues. Since these and similar problems generally consume much of every practitioner's time, this is where management can provide the most immediate help. *For low-maturity organizations, technical issues almost never appear at the top of key priority issue lists. This is not because technical issues are not important but simply because so many management problems must be handled first.*

A mature software process will not eliminate the ongoing need to understand the application, to deal with changing requirements, and to manage system design issues. However, organizations with more mature processes will be better positioned to address these issues effectively and avoid the unnecessary exacerbation of these and other problems.

According to the SEI five-level process maturity model,[3] current software engineering practice is largely at the initial level (level 1, or lowest level) of process maturity. There is a small number of level 2 organizations and a few level 3 projects in some organizations. No projects have been reported or assessed at level 4 or 5.

Nearly all level 1 software organizations urgently need to improve their management system for controlling their software process. Many managers need guidance on how to conduct project reviews, what key indicators to examine, and how to use basic management methods and tools. For project managers, this training should include the methods and procedures for estimating software size, estimating resource needs, and developing schedules. The second area requiring immediate attention for level 1 software organizations is Software Quality Assurance (SQA). SQA, while generally available, is not effectively performing its role because of inadequate resources, inadequate task definition, or inadequate management support. As organizations start to improve their software process, they should also begin gathering data on their code and test errors.

Since level 1 organizations are typically high-risk suppliers, we suggest that acquisition authorities that deal with level 1 organizations require aggressive action by these organizations to improve to level 2.

The relatively few level 2 organizations are currently among the most capable software groups in the DoD software community. But even though they have advanced substantially beyond level 1, they still have considerable room for improvement. Many of these organizations are equipped to

[3]The maturity model is discussed in Section 1.2.

advance rapidly; a full 69% have a Software Engineering Process Group (SEPG).[4] Organizations without a SEPG should promptly establish one.

Level 2 organizations typically do not adequately train their software people. A further deficiency in level 2 organizations is the lack of mechanisms to assure that SQA is evaluating representative samples of the software process. The lack of adequate regression testing is also a common problem. This generally leads to late discovery of problems, last-minute testing crises, and poor product quality. While level 2 organizations typically have their costs and schedules under reasonable control, they generally do not have orderly methods for tracking, controlling, and improving the quality of their software or of their software process. Further, few of these organizations have adequate resources or action plans directed at long-term software process improvement.

With the increasing reliance of critical defense systems on complex software, the necessary improvements require aggressive action. Thus, we suggest that acquisition authorities require level 2 organizations to dedicate resources to process improvement and to establish and report on the actions needed to progress to maturity level 3. An appropriate vehicle for doing so might be to provide for such improvement efforts as an allowable cost to the contract, or in the statement of work. We also suggest that acquisition authorities require that contractor SQA organizations be adequately staffed and effectively used.

Because there are only a few level 3 projects in organizations, our sample size is too small to draw meaningful conclusions about their improvement needs.

While SQA will continue to play an important and significant role in achieving the objectives of high-quality software delivered on time and within budget, we believe that SQA alone is not sufficient to meet these objectives. Ultimately, we need capable and motivated professionals, mature software processes, and commitment to *build quality into* every product; in short, we need the recognition that quality is everyone's job.

[4] An SEPG is a group of software professionals specifically chartered to focus on software process improvement.

1. Introduction

This report describes the initial results of a continuing Software Engineering Institute effort to characterize and report on the current state of software engineering practice.

Characterizing, understanding, and facilitating improvement in the practice of software engineering is important to the SEI, the Department of Defense, and the nation. The SEI Software Process Program has the goal of improving the process of developing and evolving software. Our approach emphasizes the following:

1. Developing and validating a software process framework and evaluation methodology for identifying capable contractors.

2. Transitioning the evaluation methodology to DoD software acquisition agencies and their prime contractors.

3. Developing and refining an associated assessment methodology for use by the DoD software community for internally assessing software engineering capability and determining improvement needs.

4. Characterizing and reporting on the state of software engineering practice in the DoD software community.

5. Facilitating software process improvement in the DoD software community.

The focus of this report is the current state of software engineering practice from a software process perspective; that is, the report will characterize the software processes currently used by software managers and practitioners in organizations doing DoD software work. The SEI has considered and our results are generally consistent with the results of a number of prior studies [BAS84, DRU82, REI88, THA82].

This report is organized in three parts. The first section provides the background and framework for collecting the data upon which the report is based. Section 2 describes the data collected and the analyses performed on this data. Section 3 discusses implications and recommendations for customers and suppliers of DoD software.

1.1. Software Process Focus

Since early 1987, the SEI Software Process Program has focused on *software process* as a means of improving the ability of software

organizations to produce software products according to plan while simultaneously improving the organization's ability to produce better products. This focus on software process is based on the premises that 1) the process of producing and evolving software products can be defined, managed, measured, and progressively improved and 2) the quality of a software product is largely governed by the quality of the process used to create and maintain it.

The software process is the set of activities, methods, and practices which guide people (with their software tools) in the production of software. An effective process must consider the relationships of the required tasks, the tools and methods, and the skills, training, and motivation of the people involved.

Software process management is the application of process engineering concepts, techniques, and practices to explicitly monitor, control, and improve the software process. It is only one of several activities which must be effectively performed for software-producing organizations to be consistently successful. Capable and motivated technical people are also needed. Knowledge of the ultimate application environment is critical also, as is detailed understanding of the end user's needs [CUR88]. Even with all these capabilities, however, inattention to the software management problems described in Section 2 will likely result in disappointing organizational performance. [KIT89] provides a more comprehensive discussion of the role and significance of software process and the discipline of software process management.

This view of process and process management has led to the creation of a process maturity model and a related software process assessment instrument, which are important elements of SEI methods for examining software processes. The remainder of Section 1 briefly discusses these elements and some methods of applying them to the software processes of organizations.

1.2. Software Process Maturity Model

The software engineering capability of an organization can be characterized with the aid of the software process maturity model shown in Figure 1.2.1. This model provides five maturity levels, identifies the key improvements required at each level, and establishes a priority order for moving to higher levels of process maturity.

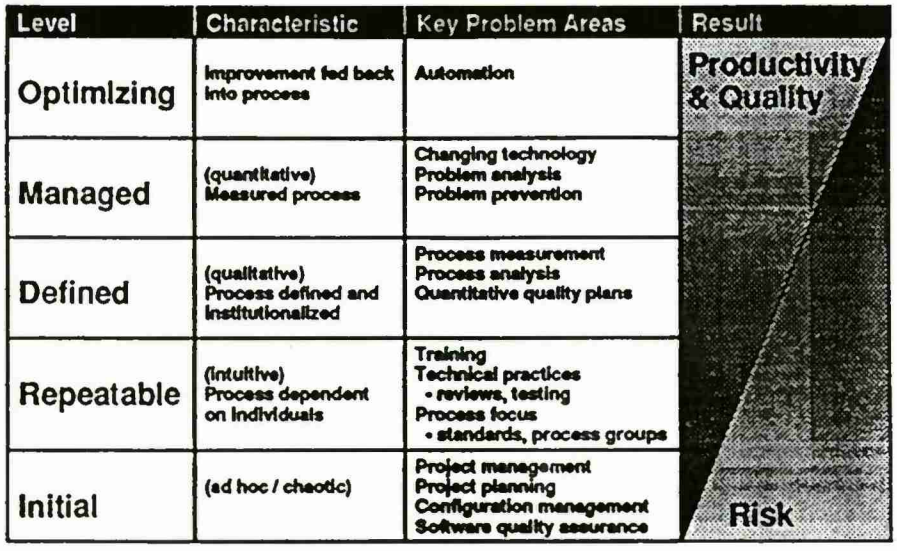


Figure 1.2.1: SEI Software Process Maturity Model

At the initial level (level 1), an organization can be characterized as having an ad hoc, or possibly chaotic, process. Typically, the organization operates without formalized procedures, cost estimates, and project plans. Even if formal project control procedures exist, there are no management mechanisms to ensure that they are followed. Tools are not well integrated with the process, nor are they uniformly applied. In addition, change control is lax, and senior management is not exposed to or does not understand the key software problems and issues. *When projects do succeed, it is generally because of the heroic efforts of a dedicated team rather than the capability of the organization.*

An organization at the repeatable level (level 2) has established basic project controls: project management, management oversight, product assurance, and change control. The strength of the organization stems from its experience at doing similar work, but it faces major risks when presented with new challenges. The organization has frequent quality problems and lacks an orderly framework for improvement.

At the defined level (level 3), the organization has laid the foundation for examining the process and deciding how to improve it. The key actions needed to move from the repeatable level to the defined level are to establish an SEPG within the organization, to establish a software process architecture that describes the technical and management activities required for proper execution of the process, and to introduce a family of software engineering methods and technologies.

The managed level (level 4) builds on the foundation established at the defined level. When the process is defined, it can be examined and improved but there is little data to indicate effectiveness. Thus, to advance to the managed level, an organization should establish a minimum set of measurements for the quality and productivity parameters of each key task. The organization should also establish a process database with resources to manage and maintain it, to analyze the data, and to advise project members on its meaning and use.

Two requirements are fundamental to advance from the managed to the optimizing level (level 5). Data gathering should be automated, and management should redirect its focus from the product to process analysis and improvement. At the optimizing level, the organization has the means to identify the weakest process elements and strengthen them, data are available to justify applying technology to various critical tasks, and numerical evidence is available on the effectiveness with which the process has been applied. The key additional activity at the optimizing level is rigorous defect cause analysis and defect prevention.

These maturity levels have been selected because they do the following:

- Reasonably represent the historical phases of evolutionary improvement of actual software organizations.
- Represent a measure of improvement that is reasonable to achieve from the prior level.
- Suggest interim improvement goals and progress measures.
- Make obvious a set of immediate improvement priorities, once an organization's status in this framework is known.

While there are many aspects to the transition from one maturity level to another, the basic objective is to achieve a controlled and measured process as the scientific foundation for continuous improvement.

It has been our experience (based on ten SEI-assisted assessments conducted since February 1987) that when software organizations are assessed against this maturity framework, the assessment method has enabled us to accurately place them on the maturity scale and identify key

improvement needs. We believe software process maturity is a useful indicator of an organization's software engineering capability, e.g., its ability to produce quality software products on time and within budget. We also believe that while the use of tools and technology can enhance software engineering capability, their contribution is often of limited value for organizations with low-maturity software processes.

[HUM88] and [KIT89] provide more comprehensive descriptions of software process management and the maturity model.

1.3. Software Process Assessment Instrument

The assessment instrument is a structured set of yes-no questions which helps to facilitate the conduct of reasonably objective and consistent assessments of software organizations [HUM87]. It has also been designed to assist DoD acquisition organizations in identifying software contractors with acceptable software engineering capabilities. Since the instrument and method for applying it are publicly available, software contractors can use them to identify areas for improvement. The SEI provides training on how to conduct effective assessments for organizations interested in conducting their own.

The questions in the assessment instrument cover three areas:

1. Organization and resource management. This section deals with functional responsibilities, personnel, and other resources and facilities.

2. Software engineering process and its management. This section concerns the scope, depth, and completeness of the software engineering process and the way in which the process is measured, managed, and improved.

3. Tools and technology. This section deals with the tools and technologies used in the software engineering process. It helps determine the effectiveness with which the organization employs basic tools and methodologies.

Some sample questions from the assessment instrument are:

- Is there a software engineering process group or function?

- Is a formal procedure used to make estimates of software size?

- Are code and test errors projected and compared to actuals?

1.4. Examining Software Processes

There are a number of ways the software process framework (software process concepts and principles + maturity model + assessment instrument) can be applied; the SEI has developed, and has experience with, the following:

- SEI-assisted assessments
- Self-assessments
- Capability evaluations
- Workshop assessments

The paragraphs below briefly discuss each type of application. A more comprehensive discussion of how assessments are conducted and the role of assessment in improving software engineering capability is contained in [KIT89].

1.4.1. SEI-Assisted Assessments

An SEI-assisted assessment is an appraisal, by a trained team of experienced software professionals, of an organization's current de facto software process. Typically, a team is composed of four or five SEI professionals and one to three site professionals. A methodology for conducting assessments has been developed by the SEI [OLS89]. The assessment team receives training in the methodology prior to conducting the actual assessment. The goal for this type of assessment is to facilitate improvement of the organization's software process. The assessment team identifies the most important software process issues currently facing the organization and develops recommendations to deal with these issues. Since the objective is improvement within a given organization, validation of questionnaire responses (e.g., requesting substantiating documents) is limited to those having a direct bearing on transition to the next higher level of process maturity (contrast this with contractor capability evaluation as discussed in Section 1.4.3).

SEI-assisted assessments are conducted in accordance with an assessment agreement signed by the SEI and the organization being assessed. This written agreement contains provisions for senior management involvement, organizational representation on the assessment team, confidentiality of results, and follow-up actions.

The SEI has been conducting this type of assessment since February 1987 and is using the knowledge and information acquired to refine an emerging picture of the state of the practice of software engineering in the DoD software community.

1.4.2. Self-Assessments

Self-assessments are similar to SEI-assisted assessments, with the primary difference being assessment team composition. Self-assessment teams are composed primarily of software professionals from the organization being assessed, with one or two SEI software professionals optionally present. The context, objective, and degree of validation are the same as for SEI-assisted assessments.

The SEI offers self-assessment training on a limited basis for organizations committed to improving their software engineering capability. Organizations that participate in the SEI-provided training execute a written agreement with the SEI which provides for sharing of assessment results, integrity of the assessment methodology, and optional participation of SEI assessment team members.

1.4.3. Capability Evaluations

Capability evaluations, like SEI-assisted assessments and self-assessments, are appraisals of an organization's current software process; however, the context, purpose, and assessment team composition are different. The context of capability evaluation is the DoD acquisition process, and the purpose is to provide information concerning the organization's software engineering capabilities for the acquisition agency. This information is then considered, along with other relevant information, in the source selection decision. Hence, validation of assessment instrument responses is a greater consideration here than it is in assessments.

Capability evaluations are conducted by trained teams of evaluators from the acquisition agency. The SEI provides the necessary training for the evaluation teams using our methodology, but we do not participate in evaluations. The results of capability evaluations are supplied by the evaluation team to the acquisition agency. Non-attributed, "sanitized" results are provided to the SEI to help us refine the assessment instrument and evaluation methodology; they also contribute to our emerging picture of the status of DoD software engineering capability.

1.4.4. Workshop Assessments

At workshop assessments, professionals from various organizations learn about process management concepts, assessment techniques, and the SEI assessment methodology. They also complete an assessment instrument and supply demographic data based on a project with which they are familiar. This format is designed for people who wish to learn more about the SEI assessment methodology with minimal investment.

The data collected at workshop assessments is added to the SEI assessment database and is used for various analyses. Workshop assessments are typically conducted at conferences and symposia attended by DoD and DoD contractor software professionals (e.g., National Security Industrial Association, Electronic Industries Association, and the annual SEI affiliates symposium).

2. Data Collection and Analysis Methodology

This chapter provides a characterization of the data used in this report, identifies some of the considerations in using this data, and describes the analyses which were performed to derive the results presented.

2.1. Basis for the Report

This report is based on information of two types:

- Responses to the assessment instrument (the questions in the instrument are yes-no questions). The responses were collected from workshop assessments and SEI-assisted assessments.

- The collective knowledge and experience which the SEI has acquired as a result of our involvement in the development and application of the various assessment methods discussed in Section 1.4.

Assessment participants include software and hardware/software developers from DoD organizations, DoD contractors, and commercial enterprises. Ten organizations participated in SEI-assisted assessments (with 4 to 6 projects involved in each assessment), and over 70 organizations were represented in the workshop assessments, representing 168 data points[5] from assessments across the United States. In every assessment, the SEI signs an agreement that there will be no attribution of the results to a specific company. The implications and recommendations presented in Chapter 3 of this report, therefore, represent an aggregate view.

2.2. Data Usage Considerations

The results described in this report reflect the state of the software engineering practice based on the data, experience, and knowledge acquired by the SEI since February 1987. This section describes some methodological considerations which we feel are germane to readers of this report.

First, the sample population was not statistically selected. Most of the respondents came from organizations that are affiliated with the SEI. These respondents varied in the type and degree of involvement with the projects they reported on.

[5] A *data point* is one set of yes-no responses to the software process assessment instrument; the scope of these responses is a specific software project.

Another consideration is the degree of validation of the responses; the extent to which corroboration of responses was requested depended on the type of assessment being conducted. At this time, we have no way of determining the effect of this factor on the responses.

In comparing the question responses received from workshop assessments and from SEI-assisted assessments, several points should be noted:

1. The SEI-assisted assessments were conducted on-site by a trained team, with participation from knowledgeable project managers and technical professionals.

2. The workshop assessment respondents contained a mix of management and non-management professionals, some of whom likely had detailed knowledge of the technical points.

3. For SEI-assisted assessments, many threshold responses were verified; however, no workshop assessment responses were verified.

2.3. Data Analysis

Two views of the data were prepared and analyzed: (1) software process maturity level distribution and (2) percent negative response to key questions.[6] For the purposes of this report, we separated response data from SEI-assisted assessments and that from workshop assessments, treating them as two distinct data populations. Because of the considerations mentioned in Section 2.2, we do not believe that greater depth of analysis than that presented in this report is justified.

2.3.1. Software Process Maturity Level Distribution

The distribution of software process maturity level across the sample population provides a high-level view of the state of the practice; Figures 2.3.1.1 and 2.3.1.2 show the software process maturity distributions for workshop assessments and SEI-assisted assessments, respectively.

For both figures, the vertical axis represents the percentage of data points in the population; the horizontal axis represents the software process maturity scale--levels 1 through 5. In order to show additional fine structure, the maturity scale has been further divided into quartiles--four quartiles for each maturity level (for a total of 20 quartiles, or 20 vertical bars). The quartiles are identified in the charts using the notation x.y, where x is the maturity level (1-

[6]Key questions are those for which a high percentage of affirmative responses is required to qualify for a particular maturity level. See Section 2.3.2.

5), and y is the quartile (1-4). In Figure 2.3.1.1, for example, 2.4 refers to the fourth (and last) quartile for level 2 and contains approximately 13% of the sample population. Note that since no data points have been observed to date at level 4 or above, that portion of the graph has not been shown.

Each data point was placed in the maturity level distribution based upon a determination of how many additional affirmative responses would have been needed to rate the project at the next higher level of process maturity. The range of these values was then equally divided into four "buckets" or quartiles. Thus, the higher the quartile number, the closer the project is to being rated at the next higher maturity level.

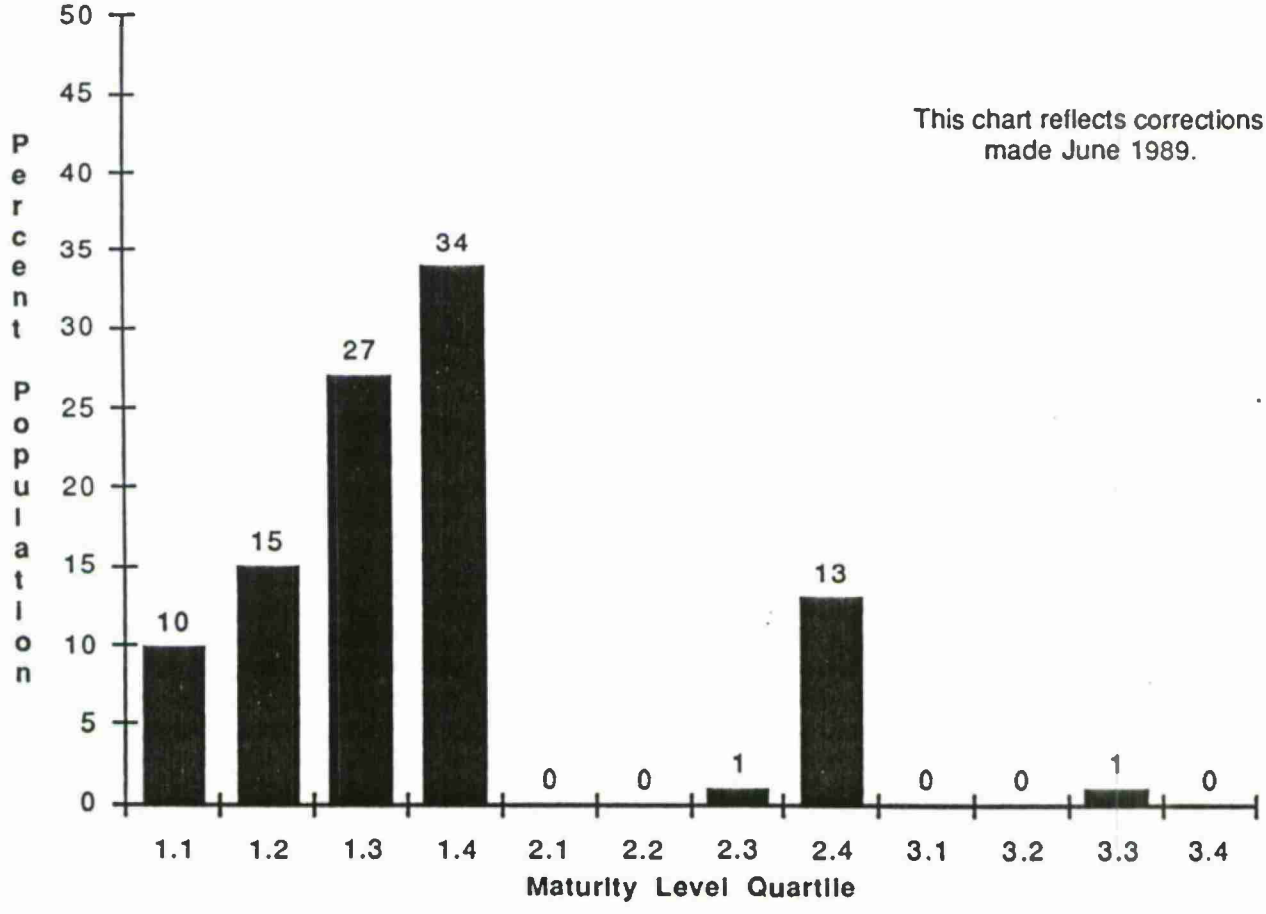


Figure 2.3.1.1: Software Process Maturity Level Distribution - Workshop Assessment Data (113 Data Points)[7]

[7]Note that the percentages may not total 100 due to rounding errors.

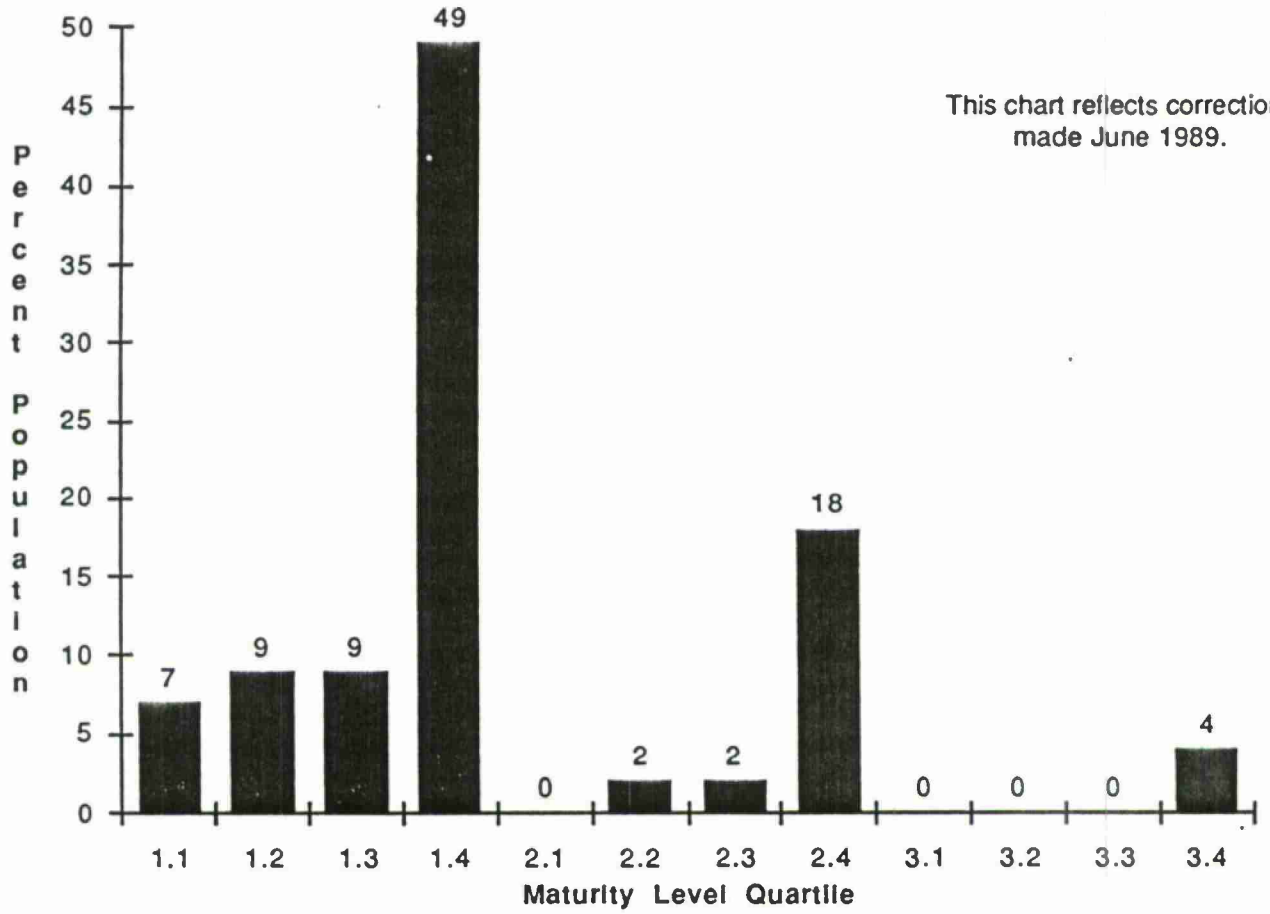


Figure 2.3.1.2: Software Process Maturity Level Distribution - SEI-Assisted Assessment Data (55 Data Points)[8]

The workshop assessment results, shown in Figure 2.3.1.1, indicate that the majority of the respondents reported projects at the initial level of maturity. Figure 2.3.1.1 shows a large percentage of the respondents to be in the fourth quartile of level 1 (quartile 1.4); with minimal improvement, these projects could be classified as level 2. Fourteen percent of all the workshop respondents reported projects at the repeatable level, and only 1% of those respondents described projects at level 3, the defined level. No workshop respondents reported projects at either the managed or the optimizing level of software process maturity.

[8]Note that the percentages may not total 100 due to rounding errors.

The maturity level distribution for projects reviewed by SEI-assisted assessments, shown in Figure 2.3.1.2, is very similar to that for the workshop data. Although workshop participants were largely mid- to upper-level managers not currently managing a project (as opposed to the project managers who provided data for SEI-assisted assessments), the profiles of process maturity are surprisingly similar. Some key differences are apparent, however. First, Figure 2.3.1.2 shows that the sample population is skewed slightly towards higher levels of process maturity. Secondly, larger numbers of projects are in quartile 4 of maturity levels 1, 2, and 3, poised for moving to the next higher level of software process maturity.

2.3.2. Profiles of Negative Responses to Key Questions

For the purposes of this report, two attributes of the assessment instrument questions are germane. First, each question is associated with a particular maturity level; for example, the question "Is a formal procedure used to make estimates of software size?" is a level 2 question. This means that an organization that has all of the attributes of a level 2 software organization (with respect to the SEI process maturity model) would respond affirmatively to this question. Second, certain questions are designated as being *key*. In order to qualify at a given level of process maturity, an organization must respond affirmatively to 90% of the key questions for that level.

To analyze the responses to key questions, we determined the percentage of the population responding negatively to each key question for levels 2 and 3 and displayed the results in decreasing order. These computations were performed for both the workshop assessment data and for SEI-assisted assessment data. The results are provided in Appendix A and are referenced in appropriate parts of Section 3. An examination of the results in Appendix A shows a close, though not exact, correlation between the two data samples; for example, for both level 2 and level 3 profiles, four out of the top five questions for SEI-assisted assessments were among the top five questions for the workshop assessment profiles.

3. Implications and Recommendations

This section discusses the implications of the current state of software engineering practice and suggests improvement actions. We discuss implications and recommendations, first for level 1 organizations, then for level 2 organizations. Our views are offered for two audiences: software suppliers and acquisition authorities.

3.1. Level 1 Organizations

3.1.1. Software Suppliers

Nearly all level 1 software organizations urgently need to improve their project management methods (CN24, CN42, CN43 ,CN44, CN46, CN77, CN84).[9] Many managers need guidance on conducting project reviews, selecting key indicators to examine, and using basic management methods and tools. For project managers, training should include the methods and procedures for estimating software size, estimating resource needs, and developing schedules. While organizations in the highest quartile of level 1 typically have the ability to make resource and schedule projections, size estimating is a problem for fully 66% (CN42) of level 1 projects.[10] Software size tracking is also a problem for 64% (CN46) of this group. As a result, projects generally underestimate resources and rely on overly optimistic schedules. The introduction of more formal procedures for estimating and tracking software size will thus substantially contribute to improved project cost and schedule performance.

One of the first steps organizations must take when they start to seriously address software quality is to gather data on the errors found in the product. This area should receive early focus in any process improvement program as it is a prerequisite to significant improvements in overall process quality and productivity. Since the final code and test stages are generally the easiest to measure, this is where data gathering should start. Of the workshop assessment respondents, nearly 60% indicate that such data was not gathered (CN48).

Another area requiring immediate attention for level 1 software organizations concerns the role of Software Quality Assurance (SQA). While 70% of level 1

[9]CN = control number. The control number uniquely identifies a particular question and is invariant across versions of the assessment instrument. Where conclusions are directly supported by question responses, the relevant assessment instrument question control number is cited. The question text can be found in Appendix A.

[10]This, and subsequent, percent negative response values are taken from the workshop assessment charts provided in Appendix A.

organizations have reporting channels separate from development for their SQA groups (CN6), 56% of the organizations report that they do not have independent audits of each step in their software development process.[11] As a result, SQA, while generally available, is not effectively performing its role. The reasons may be a lack of adequate resources, a lack of adequate task definition, or inadequate management support. In any event, effective SQA is required to assure management that its established methods, standards, and procedures are being applied. SQA can only be effective, however, when it addresses clearly identified and stated objectives. Wherever SQA is established merely to meet a contractual provision, it is not likely to contribute significantly to overall performance and may, in fact, detract.

3.1.2. Acquisition Authorities

Since level 1 organizations are typically high-risk suppliers, we suggest that acquisition authorities who deal with level 1 organizations require aggressive action by these organizations to improve to level 2.

The key items to examine in determining whether an organization is at level 1 or level 2 are defined in the SEI software maturity model and the software process assessment instrument [HUM88, HUM87]. If a detailed review is impractical, however, a critical examination of current practices for size and resource estimating and scheduling should identify the most critical exposures. When these are not adequate, improvement commitments should include the establishment of a formal planning and review system as well as comprehensive management training in software project planning.

Acquisition agencies should also be particularly interested in contractor procedures for gathering code and test error statistics since this data provides a good indication of product quality. When the data is available and can be reviewed during the project, it provides early warning of quality problems. Without this data, quality problems are generally first detected in final test, when it is too late to recover without serious schedule and cost consequences. Thus, we suggest that the acquisition agency request code and test error statistics as part of its normal project review process.

We further suggest that software acquisition agencies require the contractor to establish and maintain an effective SQA organization with adequate resources to review the key steps in the process. This SQA role should be clearly defined and documented. SQA responsibilities should be focused on the policies, methods, procedures, and standards for making plans and tracking progress against them. Once these are in place and consistently

[11] The 56% figure is based on responses to question CN30 - a non-key question ("For each project, are independent audits conducted for each step of the software development process?").

followed, the SQA role should be expanded to include peer reviews[12] and test. If the measures above are coupled with a separate management reporting chain to assure that SQA nonconcurrences and issues are resolved, SQA is likely to quickly become effective.

3.2. Level 2 Organizations

3.2.1. Software Suppliers

Though organizations at level 2 have advanced substantially beyond level 1, they still have considerable room for improvement.

Across all organizations at all maturity levels, training was found to be the area most needing improvement. Fully 88% of the level 2 organizations in the workshop assessments did not have adequate training for review leaders (CN20) and over half the organizations did not have a required training course for software developers (CN19). Although a lack of training may be acceptable for simple or noncritical applications, training is crucial in organizations responsible for developing advanced software systems. Software practitioners need to be knowledgeable and skilled in the use of languages and organizational procedures, understand the project requirements and the application area, and have a common understanding of the system protocols and architectural design. Without adequate training, projects often have serious schedule and cost problems; and they have difficulty ensuring that the requirements and the system architecture are consistently implemented.

A full 69% of the level 2 organizations do have Software Engineering Process Groups (CN15) and, thus, are equipped to advance rapidly to a more mature status. Conversely, 31% of the level 2 organizations have not established SEPGs and, thus, are hindered in planning and implementing significant software process improvement actions. The lack of a process focus is demonstrated by the fact that 50% of level 2 organizations do not track software design errors (CN47).

Regression testing helps to ensure that code changes in a product baseline which render previously implemented functions inoperable are identified. When regression testing is not adequately performed, such damage is generally not found until later in the process, when it is more expensive and time consuming to fix. Regression testing is a problem for nearly 80% (CN99) of the organizations in the workshop assessment population,

[12] By *peer review* we mean a review of a software product (specification, design, code, test plan, etc.) by peers of the producer(s) of the product for the purpose of identifying defects and improvements. Peer reviews range from walk-throughs to formal inspections, as described in IEEE standard 1028, "Standard for Software Reviews and Audits."

indicating that late problem discovery is a common problem. This situation can be substantially reduced with relatively simple regression test procedures.

A further serious need for level 2 organizations carries over from level 1. Almost one-third of these organizations do not have mechanisms in place to assure that SQA is evaluating representative samples of the software process (CN98). It has also been found that many SQA organizations are understaffed, or their role is ill-defined, or they are not adequately supported by management. The continuing lack of adequate SQA generally results in inconsistent use of established methods and procedures. Without effective SQA, organizations will find it difficult, if not impossible, to improve to level 3. Until the basic methods and procedures of level 2 are consistently and effectively applied, further process improvement efforts are likely to be ineffective.

3.2.2. Acquisition Authorities

Based on the SEI data and experience to date, the relatively few level 2 organizations are currently among the most capable software groups in the DoD software community. They typically have their costs and schedules under reasonable control; however, they generally do not have orderly methods for tracking, controlling, and improving the quality of either their software or their software process. Further, few of these organizations have adequate resources or action plans directed at long-term software process improvement.

Level 2 organizations should concentrate on establishing SEPGs as a focal point for process improvement. We suggest that acquisition authorities require organizations to dedicate resources to process improvement, including initiating and monitoring the actions needed to progress to maturity level 3. An appropriate vehicle for doing so might be to provide for such improvement efforts as an allowable cost to the contract, or in the statement of work. The key needs are for process standardization; improved methods for design, implementation, and test; and the identification and application of improved tools and technologies. Typically, the lack of an SEPG means that no one is responsible for defining metrics, installing an error tracking system, retaining and analyzing the resulting data, or reporting on progress in quality or process improvement. While the specific improvement priorities vary across organizations, the common need is for resources dedicated to process improvement.

Training is a particularly sensitive problem. Unless the contractor has an experienced team which is already familiar with the system and its application and is fully familiar with the languages and tools they are to use, some training programs are essential. Even with such an experienced team, some training is valuable. Though specific course needs vary among

organizations and training involves some expense, the costs are invariably less than the hidden costs of trial-and-error methods. *Training is expensive, but not nearly as expensive as not training.*

Regression testing is essential for any well-run software project. Without selective retesting of the system or component to verify that modifications have not caused unintended effects, there is no assurance that previously integrated functions still perform and that the system or components still comply with the specified requirements. Unless adequate regression testing is routinely performed as changes occur, large numbers of problems are likely to be found when the complete test suite is run at acceptance testing. The time used to fix defects and rerun the tests can be substantial when these activities occur during the final phase of testing. We suggest that acquisition agencies closely examine the regression test plans of their level 2 contractors.

We also suggest that acquisition authorities require their contractors to adequately staff SQA organizations and effectively use them. Although a high percentage of level 2 organizations have SQA organizations in place, only 31% of them have established methods for ensuring that SQA samples are appropriately selected. Thus, it is likely that many SQA groups represent a substantial expense but do not produce measurable benefits for the organization. If the contractor has an SQA group, the acquisition agency should require clear evidence that it is being used effectively. Such evidence should include: an SQA charter signed by a senior executive; approved standards against which SQA conducts audits; and a record of SQA nonconcurrences and the corrective actions taken. If an SQA group is not in place, its effective establishment should be a requirement in the contract.

References

1. [BAS84] Basili, V.R., Gannon, J.D., Hamlet, R.G., Yeh, R.T., Zelkowitz, M.V., "Software Engineering Practices in the US and Japan," *IEEE Computer,* 1984.

2. [CUR88] Curtis, B., Krasner, H., Iscoe, N., "A Field Study of the Software Design Process for Large Systems," *Communications of the ACM*, November 1988.

3. [DRU82] Druffel, L.E., Lt. Col. USAF, et al., *Report of the DoD Joint Service Task Force on Software Problems,* Department of Defense, July 1982.

4. [HUM87] Humphrey, W.S., Sweet, W., et al., *A Method for Assessing the Software Engineering Capability of Contractors*, Software Engineering Institute, (CMU/SEI-87-TR-23, ADA187230.), September 1987.

5. [HUM88] Humphrey, W.S., "Characterizing the Software Process: A Maturity Framework," *IEEE Software*, March 1988.

6. [KIT89] Kitson, D.H., Humphrey, W.S., *The Role of Assessment in Software Process Improvement*, Software Engineering Institute, (CMU/SEI-89-TR-3), March 1989.

7. [OLS89] Olson, T.G., Humphrey, W.S., Kitson, D.H., *Conducting SEI-Assisted Software Process Assessments,* Software Engineering Institute, (CMU/SEI-89-TR-7), February 1989.

8. [REI88] Reifer, D.J., *Final Report: Software Quality Survey*, American Society for Quality Control, 1988.

9. [THA82] Thayer, R.H., Pyster, A., Wood, R.C., "Validating Solutions to Major Problems in Software Engineering Project Management," *IEEE Computer*, August 1982.

Appendix A. Key Questions and Response Profiles

This section of the report provides a view of selected portions of the response data from workshop assessments and SEI-assisted assessments. Figures A.1 and A.2 show negative response profiles (with respect to those projects rated overall to be at level 1) for level 2 key assessment instrument questions. Table A.1 provides the text of the same key questions indexed by control number (CN). For example, Figure A.1 shows that of the workshop assessment projects reported to be at level 1 (96 out of a total of 113 projects), 64% responded negatively to question CN46 ("Are profiles of software size maintained for each software configuration item, over time?"). Question CN46 is a key question for advancing to level 2.

Figures A.3 and A.4 show negative response profiles (with respect to those projects rated overall to be at level 2) for level 3 key assessment instrument questions. Table A.2 provides the text of the same key questions indexed by control number.

Figure A.1: Percent Negative Response of Level 1 Projects to Level 2 Key Questions - Workshop Assessment Data (96 Data Points)

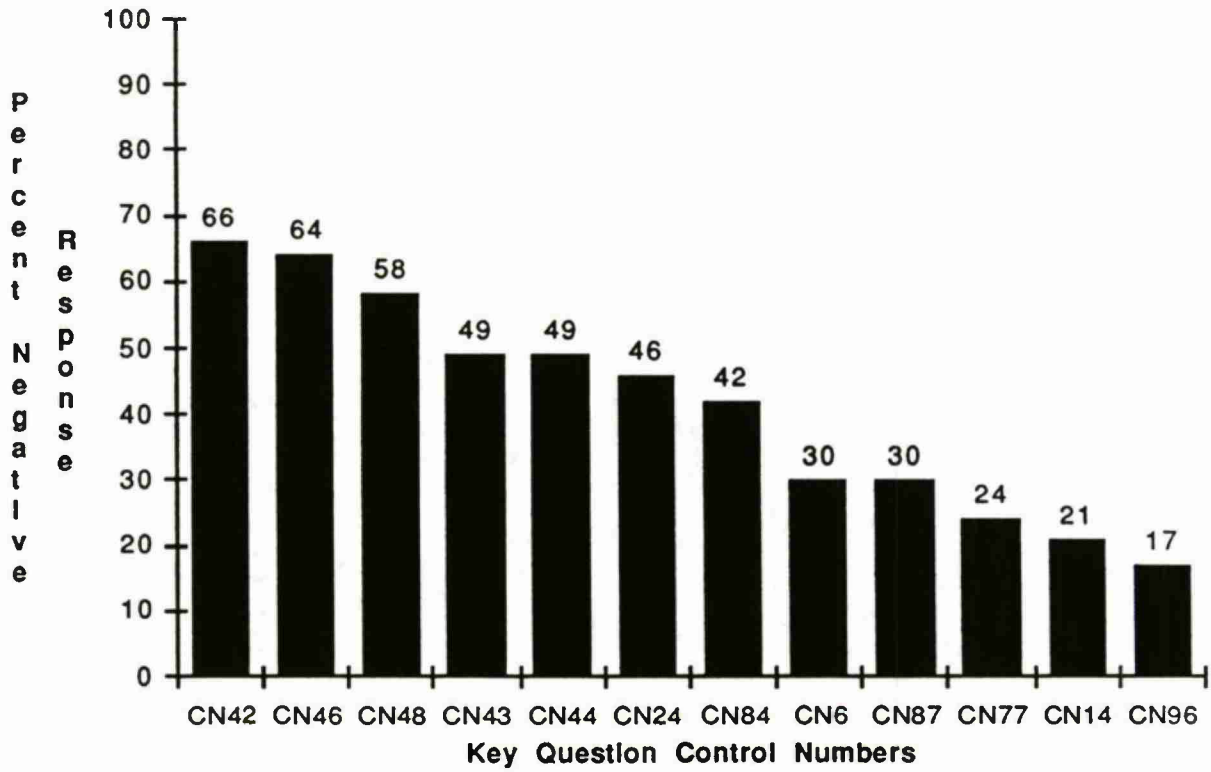


Figure A.2: Percent Negative Response of Level 1 Projects to Level 2 Key Questions - SEI-Assisted Assessment Data (41 Data Points)

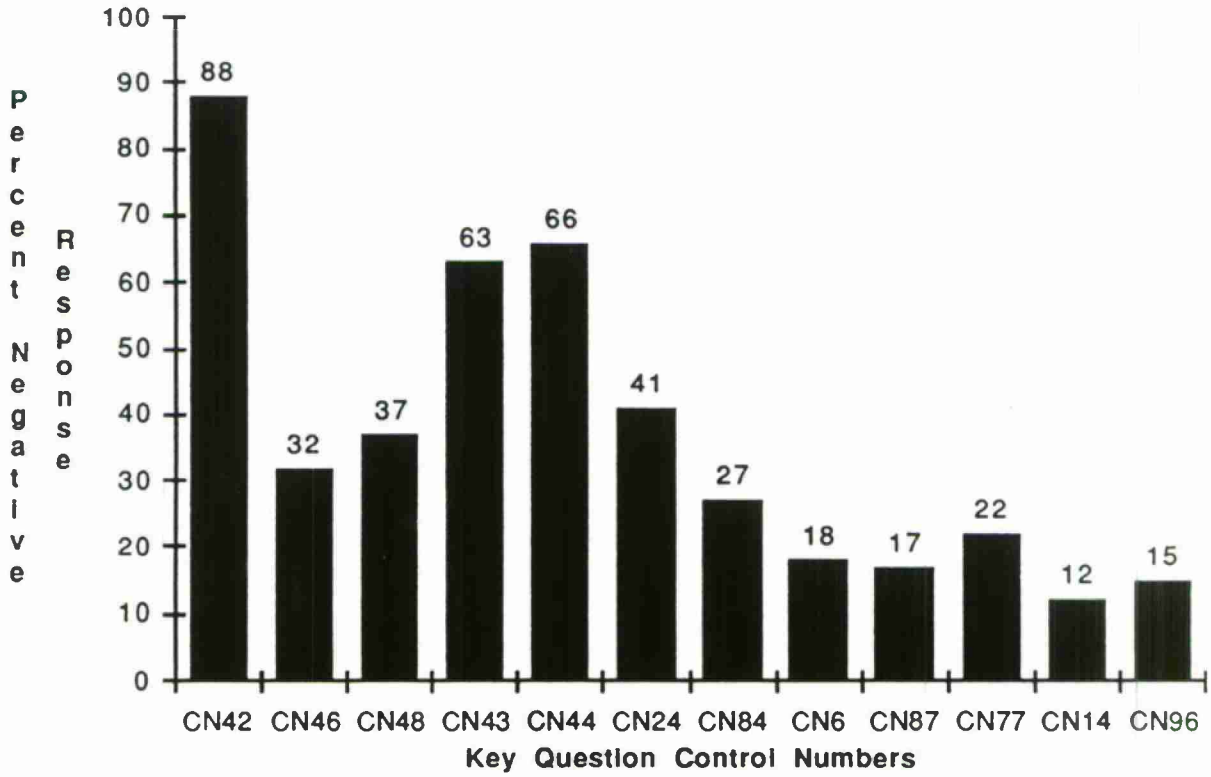


Table A.1: Key Questions for Level 2

CN[13]	Question
6	Does the Software Quality Assurance (SQA) function have a management reporting channel separate from the software development project management?
14	Is there a software configuration control function for each project that involves software development?
24	Is a formal procedure used in the management review of each software development prior to making contractual commitments?
42	Is a formal procedure used to make estimates of software size?
43	Is a formal procedure used to produce software development schedules?
44	Are formal procedures applied to estimating software development cost?
46	Are profiles of software size maintained for each software configuration item, over time?
48	Are statistics on software code and test errors gathered?
77	Does senior management have a mechanism for the regular review of the status of software development projects?
84	Do software development first-line managers sign off on their schedules and cost estimates?
87	Is a mechanism used for controlling changes to the software requirements?
96	Is a mechanism used for controlling changes to the code? (Who can make changes and under which circumstances?)

[13]CN = *control number* - the control number uniquely identifies a particular question and is invariant across versions of the assessment instrument. These questions are presented here in control number order.

Figure A.3: Percent Negative Response of Level 2 Projects to Level 3 Key Questions - Workshop Assessment Data (16 Data Points)

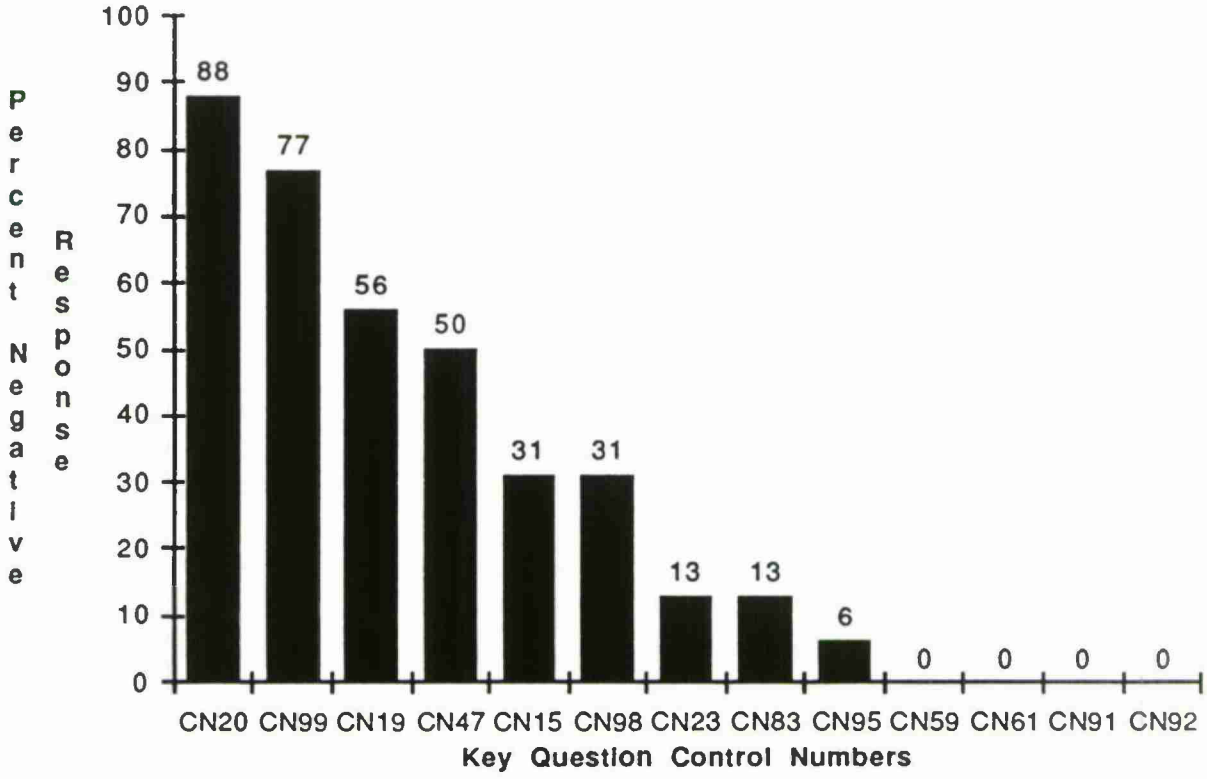


Figure A.4: Percent Negative Response of Level 2 Projects to Level 3 Key Questions - SEI-Assisted Assessment Data (12 Data Points)

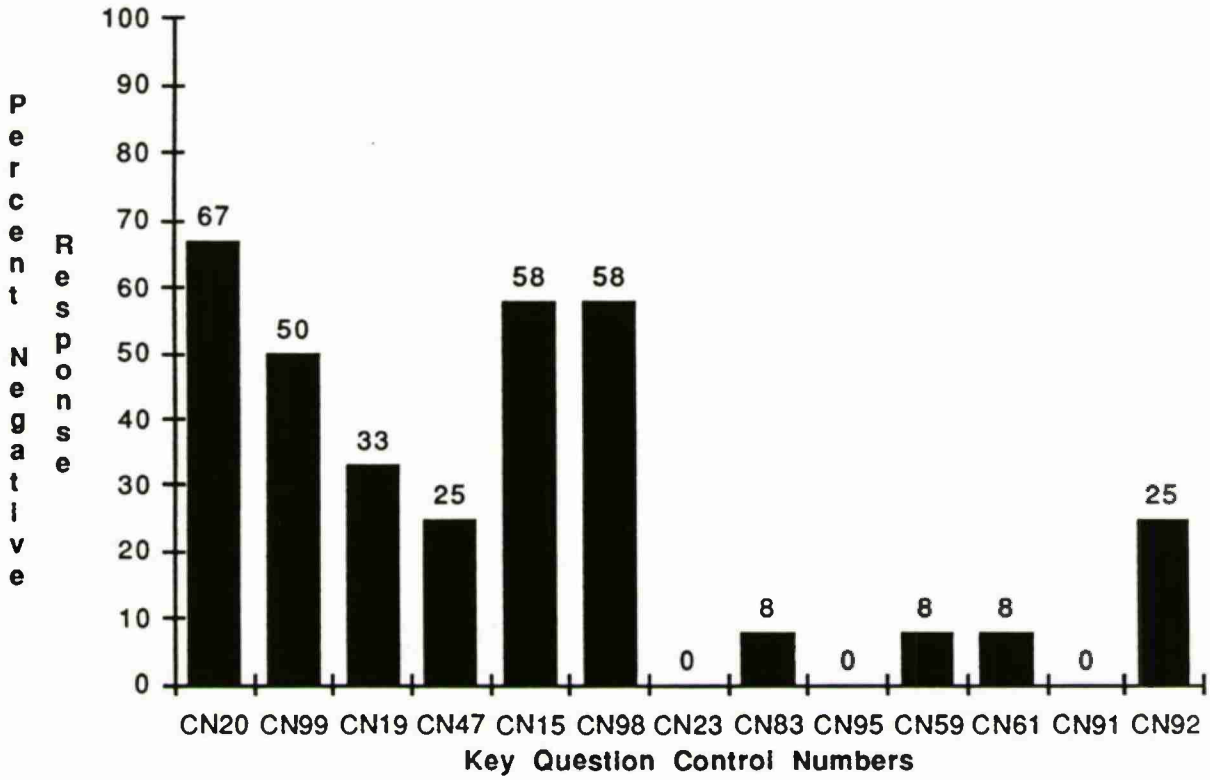


Table A.2: Key Questions for Level 3

CN	Question
15	Is there a software engineering process group function?
19	Is there a required software engineering training program for software developers?
20	Is a formal training program required for design and code review leaders?
23	Does the software organization use a standardized software development process?
23	Does the software organization use a standardized and documented software development process on each project?
47	Are statistics on software design errors gathered?
59	Are the action items resulting from design reviews tracked to closure?
61	Are the action items resulting from code reviews tracked to closure?
83	Is a mechanism used for ensuring compliance with the software engineering standards?
91	Are internal software design reviews conducted?
92	Is a mechanism used for controlling changes to the software design?
95	Are software code reviews conducted?
98	Is a mechanism used for verifying that the samples examined by Software Quality Assurance are truly representative of the work performed?
99	Is there a mechanism for assuring the adequacy of regression testing?

UNLIMITED, UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS NONE
2a. SECURITY CLASSIFICATION AUTHORITY N/A	3. DISTRIBUTION/AVAILABILITY OF REPORT APPROVED FOR PUBLIC RELEASE DISTRIBUTION UNLIMITED
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) CMU/SEI-89-TR-1	5. MONITORING ORGANIZATION REPORT NUMBER(S) ESD-TR-89- 001
6a. NAME OF PERFORMING ORGANIZATION SOFTWARE ENGINEERING INSTITUTE 6b. OFFICE SYMBOL (If applicable) SEI	7a. NAME OF MONITORING ORGANIZATION SEI JOINT PROGRAM OFFICE
6c. ADDRESS (City, State and ZIP Code) CARNEGIE MELLON UNIVERSITY PITTSBURGH, PA 15213	7b. ADDRESS (City, State and ZIP Code) ESD/XRS1 HANSCOM AIR FORCE BASE, MA 01731
8a. NAME OF FUNDING/SPONSORING ORGANIZATION SEI JOINT PROGRAM OFFICE 8b. OFFICE SYMBOL (If applicable) SEI JPO	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F1962885C0003
8c. ADDRESS (City, State and ZIP Code) CARNEGIE MELLON UNIVERSITY SOFTWARE ENGINEERING INSTITUTE JPO PITTSBURGH, PA 15213	10. SOURCE OF FUNDING NOS. PROGRAM ELEMENT NO.: PROJECT NO.: N/A TASK NO.: N/A WORK UNIT NO.: N/A
11. TITLE (Include Security Classification) THE STATE OF SOFTWARE ENGINEERING PRACTICE: A PRELIMINARY REPORT	
12. PERSONAL AUTHOR(S) Watts S. Humphrey, David H. Kitson, Tim C. Kasse	
13a. TYPE OF REPORT FINAL 13b. TIME COVERED FROM _____ TO _____	14. DATE OF REPORT (Yr., Mo., Day) February 1989 15. PAGE COUNT 35
16. SUPPLEMENTARY NOTATION	

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB. GR.	software assessment, capability evaluation, software process, maturity model, assessment instrument, software process maturity, self-assessment, software engineering capability

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This is the first in a series of SEI reports to provide periodic updates on the state of software engineering practice in the DoD software community. The SEI has developed, and is refining, a process framework and assessment methodology for characterizing the processes used by software organizations to develop and evolve software products. This report provides a brief overview of the process framework and assessment approach, describes assessment results obtained to date, and discusses implications of the current state of the practice for both customers and suppliers of DoD software.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ OTIC USERS ☒	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED, UNLIMITED
22a. NAME OF RESPONSIBLE INDIVIDUAL KARL SHINGLER	22b TELEPHONE NUMBER (Include Area Code) (412) 268-7630 22c OFFICE SYMBOL SEI JPO

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE UNLIMITED, UNCLASSIFIED

Software Engineering Institute

Our distribution records show that you received a copy of the Software Engineering Institute technical report entitled *The State of Software Engineering Practice: A Preliminary Report* and bearing the report numbers CMU/SEI–89–TR–1 and ESD–TR–89–~~01~~ 001

We recently discovered several data–entry errors in that technical report. Enclosed are corrected versions of the pages containing the errors. Please insert these change pages into your copy of the report.

Sincerely,

Purvis M. Jackson

Purvis M. Jackson
Information Management
Software Engineering Institute

Carnegie Mellon University
Pittsburgh, Pennsylvania 15213-3890
(412) 268-7700