

Assessing the Quality of OpenVMS AXP:
Software Measurement Using Subjective Data

by

Robert G. Thomson

ABSTRACT

In the absence of a well-defined development process and a set of objective metrics, subjective data can be used to assess the quality of a software release. This assessment can identify and characterize development risk, focus testing and validation efforts, and indicate where and how process management should be improved. The OpenVMS Engineering organization has developed a questionnaire, a set of quality indicators, and a data reduction methodology that implement such an assessment. This assessment approach is flexible and can be applied generally to the measurement of software quality during the evolution of a repeatable development process.

INTRODUCTION

Porting the OpenVMS operating system from the VAX to the Alpha AXP architecture was a tremendous technical challenge for the OpenVMS Engineering organization. Part of this challenge was to achieve the high degree of quality that customers expect of the OpenVMS system and would require before migrating their mission-critical OpenVMS applications and operations to a new hardware platform.

To assure that this quality challenge was met before releasing the product, the engineers involved in the port needed to answer the intuitive question, How will we know that it's right? The quality assessment approach described in this paper was an integral part of the answer. Following an overview of the quality challenge and the assessment framework, the paper describes the quality indicators and assessment process used to measure software quality during the development of OpenVMS AXP versions 1.0 and 1.5.

QUALITY CHALLENGE

OpenVMS Engineering considered schedule, functionality, and quality all to be critical factors in successfully porting the OpenVMS system to the Alpha AXP platform. Although both aggressive and complex, the port had several characteristics that favored its success:

- o An established product with well-defined capabilities

- o Carefully controlled source code and build procedures for the system
- o A very experienced development team
- o A consistent project management system for managing progress against the schedule

What the port lacked was a uniform development process with a comprehensive set of objective metrics for measuring software quality. As the project progressed, engineers were added when their expertise became needed. But with the engineers came a variety of engineering processes. Given the size and complexity of just the initial release of the OpenVMS AXP system, this lack of process consistency represented a significant deficiency.

The version 1.0 development effort kept to a demanding schedule spanning more than two years. During that time, more than 170 engineers made approximately 68,000 separate modifications or additions to the source code in order to port, build, and test the OpenVMS AXP system. These modifications were integrated and tested in stages with weekly software builds that resulted in roughly 1,200 system base levels. At its release for customer shipment, the base system of OpenVMS AXP version 1.0 comprised an estimated 3,045,000 lines of noncomment source statements. Yet, the existing metrics for measuring software quality were limited primarily to weekly statistics on incremental test hours, source code modifications, and problem reports.

QUALITY ASSESSMENT FRAMEWORK

Despite its dearth of software metrics for the initial release, OpenVMS Engineering had the following clear goals for the quality of its version 1.0 and version 1.5 releases on the Alpha AXP platform:

- o Correctness goals, which focused on completing all critical functionality
- o Reliability goals, which focused on minimizing defect introduction, stabilizing the code base, resolving all significant defects, and meeting availability targets
- o Performance goals, which focused on meeting SPECmark and TPC Benchmark A (TPC-A) projections
- o Migration goals, which focused on supporting easy and reliable application porting or execution of translated images
- o Usability goals, which focused on providing reliable system installation, documentation, and tuning

guidelines

- o Maintainability goals, which focused on supporting easy problem diagnosis

Measuring progress against these goals with objective data would have required OpenVMS Engineering to define appropriate metrics, integrate procedures for collecting metric data into the existing development process, and accumulate sufficient data to validate the collection procedures and establish baselines. The aggressive OpenVMS AXP development schedule made this approach impracticable for version 1.0.

As an alternative, OpenVMS Engineering developed an approach for assessing release quality based on subjective data. This approach built on the organization's historic reliance on the technical expertise of its engineering teams for assuring quality. At the same time, the approach laid the foundation for defining a practical set of quantitative metrics guided by experiences with the subjective data. Over time, OpenVMS Engineering can implement these metrics as part of its Continuous Improvement effort for the OpenVMS development process.

QUALITY ASSESSMENT INDICATORS

Seven quality indicators provide the framework for the process of assessing quality in the OpenVMS AXP operating system. Each indicator is intended to show the presence or absence of a meaningful characteristic of software quality. These indicators correspond to seven sets of data provided by projects that constitute a particular software release. Table 1 lists these indicators together with a summary of the subjective data and objective metrics over which the indicators are defined. The table also shows the significance of each indicator with respect to the quality assessment process. This section presents a more detailed discussion of the data sets that define the indicators and the information that these indicators provide.

Table 1 Summary of Quality Assessment Indicators

Quality Indicator Metrics	Significance	Subjective Data	Objective
Explicit Statement	Judgment from engineering team that release requirements are met	Implementation quality; outstanding risks; completeness	Source code change rate; problem report rate
Element Expertise	More accuracy in quality judgments; less likelihood of introducing	Experience with OpenVMS and with project elements	

defects

Technical Ease	Less susceptibility to defect introduction; less need for element expertise	Quality requirements; portability; maintainability	Structural complexity
Process Consistency	Less quality variation within and across development phases	Coherence of requirements, design, reviews, and testing	
Engineered Changes	Better defect prevention; less reliance on methodical testing	Use of specifications and inspections in development	
Methodical Testing	Better defect detection; less reliance on well-engineered changes	Testing effort, regularity, variety, and code coverage	
Defect Detection	Indicates progress where change and testing processes are strong; indicates risk where they are weak	Percent of detected defects being logged; percent of logged problems that describe defects	Defect counts

Explicit Statement

A project most clearly indicates quality through explicitly stated judgments from the engineering team that the software elements

- o Possess all planned functionality
- o Currently pose little technical risk to the release
- o Embody equal or superior implementation on the Alpha AXP platform as compared to the VAX platform
- o Meet the project's criteria for release readiness

Because it most fully reflects a project's overall quality, explicit statement is the most important indicator of quality.

Element Expertise

The accuracy of a subjective measure of quality is a function of a team's expertise regarding the implementation of their

project's elements. Moreover, lack of expertise may indicate a higher likelihood of introducing defects during implementation. Such expertise is based on the team's knowledge of how the project's elements were implemented and behaved on the VAX platform. The expertise is bounded by areas where a team perceives difficulty in working with the elements on the Alpha AXP platform. A project indicates high element expertise when it involves engineers who

- o Have significant experience with the OpenVMS system
- o Are already familiar with the elements involved in the project
- o Encounter little technical difficulty in modifying project elements

Technical Ease

Project elements that are technically easier to maintain are also less vulnerable to the introduction of defects during changes. The less element expertise possessed by the project team, the more significant technical ease becomes as an indicator of quality. A project indicates technical ease if the team judges that their project has

- o A relatively low priority on technical quality
- o Simple functionality, code, and data structures
- o Little vulnerability to instruction atomicity or memory granularity problems

Process Consistency

The usefulness of a process-related indicator of project quality depends on the consistency of the software development process that a project team employs. This consistency encompasses the team's understanding as well as their implementation of good software engineering process. A project indicates process consistency when software delivery involves

- o Rating product suitability based on a good understanding of customer expectations
- o Removing technical and operating risks as a precursor to release readiness
- o Defining an effective development process based on requirements, design, specification, inspection, and testing

- o Using tests with good code coverage for methodical testing
- o Reviewing or inspecting the code developed in one-person projects

Engineered Changes

Careful engineering of changes to a project's source code can catch defects before its elements are integrated into a running system. A project indicates the quality of code ports, modifications, fixes, or additions through the extent of

- o Expenditures of engineering resources on design
- o Functional or design specification completeness
- o Inspections or reviews of code changes

Methodical Testing

Regular and deliberate ad hoc, regression, and stress testing is needed to find the defects introduced into a project's elements through additions or modifications to its source code. The less effectively a team engineers changes to the elements to prevent defects, the more significant testing becomes as an indicator of quality. Methodical testing of a project's elements is indicated where tests

- o Run each week and on each software base level
- o Involve ad hoc, regression, and stress tests
- o Cover a significant portion of main program code and error-handling code
- o Use a significant portion of a project's total engineering resources

Defect Detection

When compared against the number of defects detected in prior releases, the number detected within a project's elements for the current release provides an indication of its current quality. A low ratio of the current defect count to the past defect count may indicate either an improved development process or inadequate detection; a high ratio may indicate the reverse. The more effectively a team engineers changes to an element and performs the element's tests, the more reliable the defect detection indicator becomes as a measure of quality.

Defect counts are available from the defect tracking system; however, defects that are readily resolved are frequently not logged. Therefore, defect counts across a release are normalized by having project engineers estimate the percentage of defects identified during inspections, debugging, and testing that they actually log in the defect tracking system.

QUALITY ASSESSMENT PROCESS

The assessment process applies these quality indicators to data gathered primarily through a questionnaire, which is administered to a subset of the projects included in a software release. Applying the quality indicators to questionnaire data yields a set of quality profiles. The usefulness of these profiles for assessing quality depends both on the accuracy of the data and on the ability of the targeted projects to represent the quality of the overall release. This section describes the quality assessment process in terms of our experiences across two releases of the OpenVMS AXP system, versions 1.0 and 1.5.

Select Assessment Targets

The assessment process begins by selecting a set of projects within the software release to serve as targets for measuring the release's quality. We made this selection for a particular OpenVMS AXP release by ranking the projects based on the following factors:

- o The functional areas where the project manager believed quality was critically important to the success of the release
- o Whether a project provided latent, limited, or full support of ported or new functionality for the release
- o The number of problem reports filed in prior releases against the elements of the project

Because the version 1.0 development effort was quite large, we focused the assessment on 57 projects, which constituted the top 17 percent of the resulting ranked list. Those projects accounted for 74 percent of the total source code involved in the release. Because the version 1.5 development effort was smaller, we targeted only 38 projects and yet encompassed more of those projects that dictated the release's quality.

Administer an Assessment Questionnaire

The assessment process uses a questionnaire to measure the quality of the targeted projects. Because all answers to the

questionnaire are assumed to be subjective, its effectiveness relies more on the completeness of the responses than on their accuracy. With this in mind, we designed the question set for each OpenVMS AXP release to be large and varied, yet easy to answer.

For the version 1.5 release, 29 questions, some multipart, provided 75 data values for each project. The version 1.0 questionnaire was slightly smaller. Most questions could be answered by indicating on a graduated scale either a percentage value or a qualitative judgment (such as easy versus hard, or low versus high). Typically, respondents were able to complete the version 1.5 questionnaire in less than 15 minutes.

Figure 1 shows the steps involved in deriving an individual quality score and a composite quality score using a questionnaire. The three questions from the OpenVMS AXP version 1.5 questionnaire illustrated in Step 1 of the figure form the question set that provides the data for assessing element expertise. The example shows the questions as completed for Project_20.

To mitigate bias and uncover inconsistency within the questionnaire data, we selected a broad range of questions that measured progress against quality goals from three perspectives:

- o A process perspective, which covered design, specification, coding, inspection, and testing. These process elements were measured with respect to project resource expenditures and product element coverage.
- o A product perspective, which covered element size, complexity, technical risks, implementation quality, completeness, release readiness, and suitability relative to customer expectations.
- o A project perspective, which covered priorities, difficulty, team size, and engineering experience.

For both releases of the OpenVMS AXP system, participation in the assessment survey was high. More than 90 percent of the project teams returned questionnaires with an average of more than 90 percent of the questions answered.

Apply the Quality Indicators

The purpose of applying quality indicators to questionnaire responses is to convert qualitative judgments into quantitative measures of quality. To facilitate database entry and quantitative analysis, we first normalized the questionnaire responses, using a scale of 0 to 10, where 10 generally represented greater contribution to product quality. Numeric answers were entered directly into the database without scaling;

unanswered questions were assigned the value of -1.

Given this scale, responses of 3 or less represented low (weak) assessments and responses of 7 or more represented high (strong) assessments. A response of 5 represented an implicit norm among the development teams for what constituted an acceptable process, product, or project. All assessments were interpreted in light of how this norm related to organizational goals or prevailing industry practices.

Step 2 of Figure 1 shows the normalized and averaged data used to assess element expertise for Project_20 and also for Project_36. Note that dividing by 10 normalized the responses to questions 2 and 3. For question 4, the five gradations from easy to hard were normalized by mapping them onto the values 0, 3, 5, 7, and 10. Easy completion with respect to technical difficulties indicated greater element expertise and hence received the higher value. Averaging the normalized data across the question set yielded the element expertise quality score for each of the two projects. Note that for the process consistency indicator, this averaging occurs not over the sum of all responses in the question set but over the differences between pairs of responses that should be close in value to be consistent. The resulting average is then subtracted from 10. For example, a project that rates its ability to meet customer expectations as 9 but its understanding of those expectations as 5 would score $10 - (9 - 5)$ or 6 with respect to this pair of responses.

The mean value of all quality scores for a particular indicator reveals the engineering team's collective perception of how strong the overall release is with respect to the group norm for that indicator. Ranking the quality scores and then graphing them as variances from this mean facilitates Pareto analysis of the projects by indicator. This analysis reveals those projects with a particularly strong or weak score for a specific indicator.

Figures 2 and 3 show the quality scores for element expertise and technical ease that we derived for OpenVMS AXP version 1.5. These figures suggest a relatively high perception across the projects of overall element expertise contrasted by a lower and more varied perception of technical ease. Pareto analysis of these distributions highlights projects such as Project_36, whose quality scores were high for both indicators, and Project_20, whose scores were both low.

Synthesize Quality Profiles

Because our derivation of the indicators was based on engineering experience rather than on statistical modeling, no single indicator is a reliable predictor of overall project quality. Moreover, because the quality indicators are based on inexact data, the application of a particular quality indicator may be inconclusive with respect to some projects. To overcome these

obstacles to comparative assessment of project quality, we synthesized quality profiles using a composite of the quality scores for each project.

Repeating Step 2 of Figure 1 using the responses to other question sets yields normalized scores for each quality indicator. The table presented in Step 3 shows the quality profiles for Project_20 and Project_36. Also shown is the quality profile arrived at by averaging the quality scores across all the targeted projects in the version 1.5 release.

Figure 4 depicts the quality profiles of the projects targeted for OpenVMS AXP version 1.5. These composites use six of the seven quality indicators. Due to insufficient questionnaire data regarding defect detection and removal, the corresponding indicator was not employed in the assessment. Consequently, the identification of error-prone modules and the assessment of defect removal efficiency occurred separately within the ongoing verification efforts for that release.

To reflect the relative capacity of each indicator to independently provide meaningful information about project quality, we formed the composites by weighting the individual quality scores as follows:

- o Explicit statement has a weighting factor of 3.
- o Methodical testing, engineered changes, and element expertise have weighting factors of 2.
- o Technical ease and process consistency have a weighting factor of 1.

This weighting was based on OpenVMS Engineering experience and reflects relative contribution to the assurance of quality within the current development process. Because field data regarding the actual quality of the released product was unavailable during the assessment effort, statistical analysis of the questionnaire data was inconclusive.

Using this weighting, the resulting maximum score across all six indicators totaled 110. To make the range of values for the composite quality profiles more intuitive, we further scaled this aggregate by 0.91 (100 divided by 110) so that the maximum totaled 100. Multiplying the individual scores by the weighting and scaling factors yielded the second set of scores shown in Step 3 of Figure 1. For reference, an indicator composite that consists of the maximum possible scores for these weighted and scaled indicators appears at the bottom of Figure 4. A similar composite profile of the average project scores for the release also appears.

Interpret the Quality Profiles

Clustering the projects according to their composite quality profiles highlights relative product quality, project risk, and process deficiencies. For OpenVMS AXP version 1.5, we identified nine groups of quality profiles with similar distinguishing characteristics relative to the average profile. In Figure 4, braces delimit these groups.

The average composite score for the targeted projects in the version 1.5 release was 55 out of 100, with 76 percent of the projects scoring in the range of 45 to 65. Only Project_29 scored at or above the average for each indicator; only Project_33 and Project_38 scored at or above the norm for each. Consequently, most projects fell within the Needs Ongoing Validation region of Figure 4. Scoring in this region indicated that a project required some form of validation work to improve quality prior to beta testing and customer shipment of the release.

In several instances, the questionnaire data was sufficiently scant or the quality issues sufficiently numerous to suggest that additional data on a project's actual condition was needed before completing that project's quality assessment. Because a value of -1 was assigned to each unanswered question, projects for which such a value was assigned generally exhibited low indicator composites as depicted in Figure 4 by the bars ending in the Needs Further Investigation region. Project_01 and Project_09 are examples of projects in this category.

If the quality indicators were sufficiently strong, little further assessment or validation work appeared to be needed. Projects that exhibited high indicator composites are depicted by bars ending in the Needs Final Confirmation region. Only Project_33, Project_36, Project_37, and Project_38 fell into this category.

QUALITY ASSESSMENT RESULTS

Taken together, the composite quality profiles, the quality indicator distributions, and the project questionnaire data form an assessment continuum within which to measure progress against quality goals. From a release perspective, the composite quality profiles and the indicator distributions identify process deficiencies. They also characterize areas of risk for the product. From a project perspective, a comparison of quality profiles and scores focuses ongoing verification efforts where they can have the greatest impact on the overall quality of a release. The questionnaire data itself can help determine the form this verification work takes. The results from the assessment of data obtained from the alpha test of OpenVMS AXP version 1.5 illustrate these measurement perspectives.

Identification of Release Deficiencies

The projects that made up the version 1.5 release were known to have a widely varying and typically incomplete process for engineering changes in their code base. From the quality assessment administered when alpha testing began, we clarified the following deficiencies in the process and product for that release so that steps could be taken to ensure that the release was ready for customer shipment:

- o Sixteen percent of the projects had significant risk due to outstanding dependencies, unresolved technical problems, or operational instabilities.
- o Although 76 percent of the project teams rated their technical capacity as high, 71 percent reported having significant difficulty completing the project due to schedule, equipment, or personnel constraints.
- o Ad hoc, regression, and stress tests were regularly executed on the code of 34 percent of the projects.
- o Fifty-five percent of the projects had some portion of their code implementation described by a functional or design specification.
- o Thirty-seven percent of the projects were handled by just one engineer. Of these 14 projects, 5 had above-average technical difficulty and 5 expended no engineering resources on reviews or inspections.
- o Twenty-six percent of the projects lacked a strong understanding of customer expectations against which to evaluate product attributes.
- o Code reviews across the projects averaged only 30 percent coverage of ported source code, 40 percent coverage of rewritten or added source code, and 60 percent coverage of source code fixes.

Similar kinds of results from the quality assessment for the version 1.0 release led to the implementation of a process for enhancing product stability prior to customer shipment. The results also contributed to decisions within OpenVMS Engineering to establish more rigorous software metrics within the development process. Moreover, clarifying the process deficiencies for OpenVMS AXP versions 1.0 and 1.5 has contributed to an increased emphasis on defect prevention in the follow-on release.

Focus for Project Verification

In the context of the product risks and process deficiencies just summarized, the quality assessment results for version 1.5

provided the following framework for focusing the ongoing verification efforts:

- o Project_01 through Project_05 were missing more than 15 percent of the questionnaire data. (See Figure 4.) These projects required further investigation to determine the current condition of constituent elements as well as the form, focus, and priority of needed verification work.
- o Project_06 through Project_18 exhibited composite scores that were below average overall. Verification work that focused on compensating for the weak change and testing processes was a high priority for these projects.
- o Project_19 through Project_24 exhibited at least average values for engineered changes and methodical testing; these projects also exhibited significantly below average values for technical ease and, in most cases, element expertise. Verification work for these projects needed to focus on the functionality that posed the greatest technical difficulty or risk given schedule and resource constraints.
- o Project_25 through Project_29 exhibited average quality profiles. Their verification work needed to focus on specific portions of the code where defects may exist due to technical difficulty, inadequate changes processes, or poor test coverage or effectiveness.
- o Project_30 through Project_32 had strong processes. Because their technical ease or element expertise indicator values were below average, however, verification work needed to focus existing processes on mitigating current risks and improving the product's readiness to meet customer expectations.
- o Project_33 through Project_38 were evidently on-track to a high-quality release and therefore required only a confirmation of quality prior to customer shipment.

Given the limitations of the assessment data and its pervasive reliance upon engineering judgment, following all assessments with some form of verification work was important. In some cases, the data as provided and interpreted within the assessment indicated a level of quality that we knew was not actually present.

By removing defects from the product as projects completed their planned functionality, the ongoing verification effort for version 1.5 contributed to improved implementation quality relative to the VAX platform, mitigated risk due to technical or stability problems, and increased the satisfaction of release readiness criteria.

CONCLUSIONS

To assure the quality of its product while improving the quality of its development process, OpenVMS Engineering implemented a process for assessing the quality of its releases using subjective data. This assessment process has proven useful in characterizing product risks, focusing verification efforts, and identifying process deficiencies during the development of versions 1.0 and 1.5 of the OpenVMS AXP operating system. The assessment identified areas that needed attention; the resulting actions led to improved quality.

Using the Assessment Process

By focusing only on those projects key to a release's success, the assessment process described in this paper limits the cost and turnaround time for an assessment of quality without significantly diminishing its value. By focusing on subjective data, this process captures the judgment of engineers on the project teams regarding overall progress toward release readiness.

The OpenVMS AXP questionnaire covers various product, project, and process aspects of a release. The questions may be tailored for different software releases or even different software products.

Using seven quality indicators, which are defined over subsets of questions from the questionnaire, the assessment process synthesizes quality profiles for each project. These profiles are based on quality norms that are implicit within the development organization. By administering the assessment process as a release enters its alpha testing, these profiles can guide the project's movement toward its quality goals for the release.

Improving the Assessment Process

Several opportunities exist for improving the usefulness of this assessment process. As the process is repeated across successive software releases, the organization can

- o Validate the predictive value of the assessment process through statistical analysis of quality indicators and questionnaire data against selected quality results when a release begins shipping to customers
- o Refine the questionnaire to ensure that the questions remain relevant to the development process, unambiguous, and internally consistent
- o Complement the developer assessment administered during

alpha testing with a similar customer assessment during beta testing

As an organization's software measurement process matures, subjective measures should be replaced with objective metrics for which data can be economically and reliably collected. Such metrics should reduce reliance on the subjective data, but not eliminate it: the perceptions of an experienced engineer can usually add clarity to the assessment of release quality.

ACKNOWLEDGMENTS

Development of this assessment process was sparked, encouraged, and facilitated by the members of the Quality Assessment project for OpenVMS AXP version 1.0: Jeff Pilsmaker, Pam Levesque, Ralph Weber, and Bill Goleman. Its form was refined and its usefulness validated by the members of the OpenVMS AXP Verification Group. Curt Spacht and Tim Beaudin, the validation project leaders for versions 1.0 and 1.5, were particularly supportive during the implementation and repetition of this process.

GENERAL REFERENCES

As representative of current trends in the definition and deployment of software measurement processes, the following references proved particularly useful during the effort described in this paper:

W. Humphrey, *Managing the Software Process* (Reading, MA: Addison-Wesley, 1989).

C. Weber, M. Paulk, C. Wise, and J. Withey, "Key Practices of the Capability Maturity Model," Technical Report CMU/SEI-91-TR-25 (Pittsburgh, PA: Software Engineering Institute, Carnegie-Mellon University, 1991).

J. Baumert and M. McWhinney, "Software Measures and the Capability Maturity Model," Technical Report CMU/SEI-92-TR-25 (Pittsburgh, PA: Software Engineering Institute, Carnegie-Mellon University, 1992).

R. Grady and D. Caswell, *Software Metrics: Establishing a Company-Wide Program* (Englewood Cliffs, NJ: Prentice-Hall, 1987).

R. Grady, *Practical Software Metrics for Project Management and Process Improvement* (Englewood Cliffs, NJ: Prentice-Hall, 1992).

TRADEMARKS

The following are trademarks of Digital Equipment Corporation: Alpha AXP, AXP, OpenVMS, and VAX.

SPECmark is a registered trademark of the Standard Performance Evaluation Cooperative.

BIOGRAPHY

Robert G. Thomson A senior software engineer in the OpenVMS AXP Group, Robert leads the validation of Volume Shadowing's port to the Alpha AXP platform. During the OpenVMS port, he measured quality and contributed to functional verification, for which he was co-recipient of an Alpha AXP Achievement Award. Since joining Digital in 1986, Robert has also contributed to improvements in system availability measurement and in symmetric multiprocessing and Backup performance. He has a patent pending and two published papers based on this work. Robert holds an M.S. in computer engineering from Boston University.

=====
Copyright 1993 Digital Equipment Corporation. Forwarding and copying of this article is permitted for personal and educational purposes without fee provided that Digital Equipment Corporation's copyright is retained with the article and that the content is not modified. This article is not to be distributed for commercial advantage. Abstracting with credit of Digital Equipment Corporation's authorship is permitted. All rights reserved.
=====