Software Engineering Institute

Basic Principles and Concepts for Achieving Quality

Emanuel R. Baker, Process Strategies Inc.
Matthew J. Fisher and Wolfhart Goethert, SEI

Lisa Marino, Editor

December 2007

TECHNICAL NOTE
CMU/SEI-2007-TN-002

Software Engineering Process Management

Carnegie Mellon

This work is sponsored in part by Process Strategies Inc. and the U.S. Department of Defense.

The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

This work was created in the performance of Federal Government Contract Number F19628-00-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.

For information about purchasing paper copies of SEI reports, please visit the publications portion of our Web site (http://www.sei.cmu.edu/publications/pubweb.html).

Table of Contents

List of Figures

List of Tables

Abstract

This technical note extends the quality concepts first articulated in *A Software Quality Framework* (SQF) developed in the early 1980s for the Department of Defense (DoD) by Baker and colleagues. The original quality concepts of the SQF are extended beyond software to include products, services, and processes. This technical note also describes the conceptual elements necessary for building quality into systems, or any entity, and evaluating the quality actually achieved. This technical note presents definitions and conceptual elements within the context of Capability Maturity Model Integration (CMMI) to show how CMMI codifies the concepts. Another goal of including the CMMI context is to help CMMI implementers recognize the purpose of some CMMI components relative to quality concepts and principles, and to help ensure a CMMI implementation leading to quality products. The definitions provided are the structural members of this extended framework, which lay the basis for establishing quality requirements, methods to help satisfy these requirements, and quality evaluation.

1 Introduction

This technical note describes an Extended Quality Conceptual Framework (EQCF), which represents, as the name implies, an extension to the Software Quality Framework (SQF) developed initially in 1982 [Baker 1982a, Baker 1982b]. The goal of the SQF was to place quality in proper perspective in relation to the acquisition and development of computer software. The concepts embedded in the SQF are relevant today in that they still provide a basis on which process and quality models can be structured and implemented As an example, this technical note discusses how the quality concepts have been adopted and codified by the CMMI model (as an example, CMMI-DEV, V1.2), where the CMMI model provides a first-level implementation of the overall concepts, especially those associated with the definition of quality and what it means in different disciplines.[1] Another goal of including the CMMI context is to help CMMI implementers recognize the purpose of some of the CMMI components that they are implementing and help ensure a CMMI implementation leading to quality products.

A point we make immediately is that the success of a project (and the foundation for its ultimate success) is tied to achieving the requisite product quality. However, one must understand what product quality is and the technical aspects of specifying, developing, and evaluating it. We also note that product quality results from employing appropriate processes and methods in product development. Quality cannot be achieved by assuring and testing the product! We stress these two fundamental concepts, introduced in the SQF published in 1982, throughout this document. These concepts were true then and are still true today.

1.1 BACKGROUND AND MOTIVATION

The SQF was developed as part of DoD efforts to establish a consistent set of software standards (DoD-STD-2167 and DoD-STD-2168) that incorporated quality concepts. At the time that the SQF was developed, the authors recognized that the quality concepts promulgated by this original framework were applicable not only to the defense community, but to all software development efforts. They were, and still are, universally applicable concepts.

As noted above, the goal of the SQF was to place quality in proper perspective in relation to the acquisition and development of computer software. Such a focus was necessary because the acquisition and development communities often tried to interpret, adapt, or equate software quality and associated functions to hardware concepts. For example, there were numerous attempts to place software quality in the context of hardware reliability and force-fit the content of standards and specifications for software quality assurance functions into existing hardware quality assurance (QA) organizations. At that time, the SQF provided a fresh look at quality especially as applied to computer software.

[1] For the purposes of this technical note, the term “framework or “conceptual framework” is defined as a set or system of ideas in terms of which other ideas are interpreted or assigned meaning.

Over time many of the concepts and principles articulated in the SQF have been implemented in various forms, not only in venues such as the DoD standards, but also in process and maturity models of today, such as Capability Maturity Model Integration (CMMI®). In such manifestations, the framework concepts have been applied not only to software but to other types of products and to processes. Although the concept of relating process quality to product quality was first articulated in the SQF it was not until the development of the Process Maturity Model[2] in 1987, and eventually the SW-CMM [Paulk 1995] and CMMI [Chrissis 2006], that these principles finally were codified in a concrete manner, making them easier to implement. Some of the same principles were likewise codified by the International Organization for Standardization (ISO) [ISO 2004]. The original SQF enumerated the kinds of things that needed to be done for each element of the framework. Process and maturity models and the ISO standards took those concepts and organized them into a structure that could be used to measure how organizations implement them.

1.2 TECHNICAL NOTE ORGANIZATION

The intent of this technical note is to extend and amplify the original scope of the Software Quality Framework to an Extended Quality Conceptual Framework that encompasses the quality of entities such as products, systems, services, and processes, in general, not just software. The EQCF addresses the conceptual elements necessary for "building in" quality and evaluating the quality actually achieved.

The information in this technical note is organized as follows:

- a set of definitions, which constitutes the structural members of the EQCF, and their attendant concepts impacting quality
- three additional framework elements that, in combination with the definitions, comprise the EQCF
- several considerations implied by or derived from the extended framework

Throughout, we attempt to illustrate application of the extended framework concepts into elements in the CMMI. Note that CMMI references are not only to illustrate the codification of the quality concepts in one process model, but also to help CMMI implementers recognize the purpose of some of the CMMI components that they are implementing—this may help ensure a quality CMMI implementation. The EQCF is much broader in scope and application than CMMI and ISO standards.

Three elements of the EQCF and interactions with other development activities are depicted in Figure 1. These elements are:

- Establish Requirements and Control Changes
- Evaluate Process and Product Quality
- Establish and Implement Methods to Achieve Quality (methods to build quality ***into*** products or other entities)

® CMMI and CMM are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

[2] The Process Maturity Model, developed by the Software Engineering Institute (SEI) in 1987, is the forerunner of the SW-CMM.

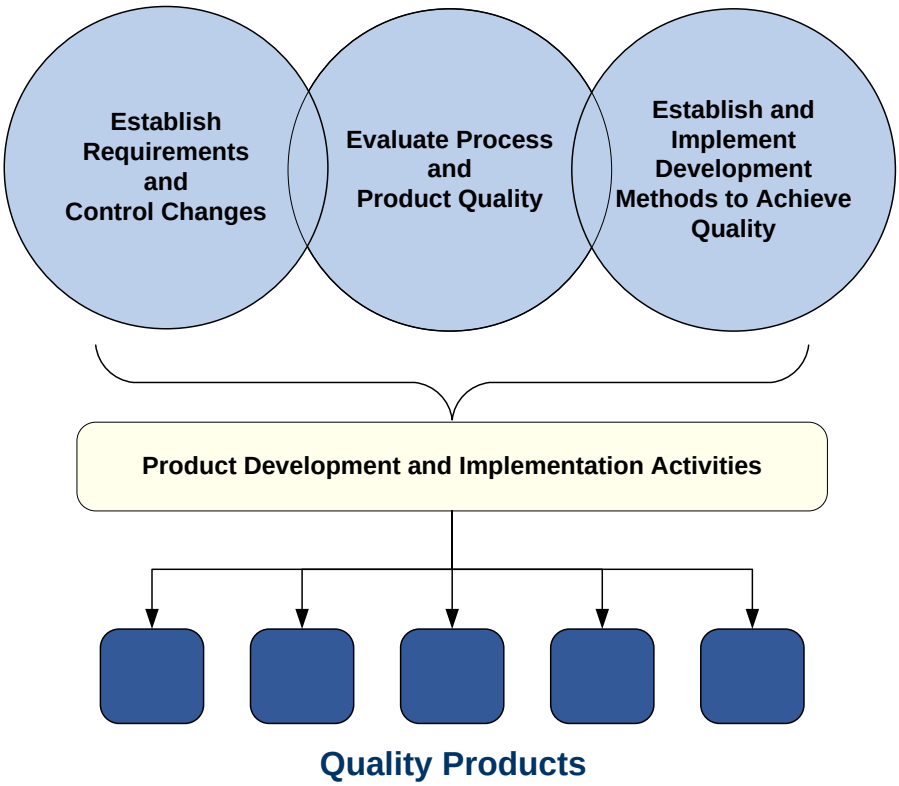

Figure 1: Interaction of Activities between Extended Quality Conceptual Framework and Development Project[3]

Note: While the EQCF can be applied to different disciplines, for example, development or acquisition, for conciseness this technical note will focus on product development. You can readily adapt the discussions to other disciplines and CMMI constellations.

[3] This figure is not intended to be a Venn type diagram, but simply a graphic to help visualize the fact that the three elements interact among themselves and with development activities leading to product quality.

2 A Conceptual Framework for Quality

To characterize the Extended Quality Conceptual Framework, we first establish a set of definitions and related concepts as the prime structural members of the framework. This is followed by presenting the application of these concepts to the construct of a Quality Framework that uses the definitions and implements the concepts of the EQCF.

2.1 DEFINITIONS AND CONCEPTS FOR QUALITY

The foundation of a Quality Framework is based on providing a comprehensive definition of quality; thus we present the following terms and definitions to establish a basis for the definition of quality and provide the foundation for a quality framework:

- Object (Entity)
- Process
- Requirements
- User
- Evaluation
- Measure and Measurement
- Quality
- Performance

In the following definitions, we avoid any specific organizational connotations and relate only to activities and interrelationships.

2.1.1 Object (Entity)

In this technical note, the type of objects (entities) that quality can be applied to are

- product
- process
- service
- resource
- agent
- artifact
- activity
- measure or metric
- environment
- collection of entities or objects

However, for conciseness in this technical note, we focus on quality associated with a product, where a product is defined as such:

> *A product is any tangible output or service that is a result of a process [Chrissis 2006, Cooper 2002].*

A product itself may include hardware, software, documentation, or a combination of these; also note that a service is included in the definition. Accordingly, even though the discussion focuses on products, it is important to remember that the same principles apply equally as well to a service, or to anything else included under our definition of object.

2.1.2 Process

Ultimately, what one is interested in is the quality of the delivered product or service. The quality of a product or service is dependent on the quality of the process used to create it; consequently, we need to establish a definition of process, enabling us to develop a definition of quality. As part of this development, we view process as

> *A set of activities performed for a given purpose, for example, a software acquisition process [Cooper 2002].*

This definition is simple and accepted by a large part of the community. Another slightly more complicated definition is that a process is

> *The logical organization of people, materials, energy, equipment, and procedures into work activities designed to produce a specified end result [Pall 1987].*

This definition brings in more of the implementation aspects of a process, for example the skills of the people implementing the process and the application of tools, methods, etc.

CMMI and ISO Context

The definitions we use, that is, product and process definition, are fundamentally the same as codified in CMMI and the ISO standards; i.e., a useful result of a process that is intended for delivery to a customer or end user, and a process is a set of activities that transform inputs to outputs [ISO 15939]. In addition, the ISO standard categorizes products in four groups: services, software, hardware, processed material.

The CMMI describes work products as well as products, where work products contribute to the final product delivered to a customer but themselves may not be deliverable. Also, the CMMI uses the term product components. Depending on one's perspective, a product component could be a product, in which case the above definition can be applied to the product component as well [Chrissis 2006].

The CMMI includes an additional stipulation that a product is something delivered to a customer. A "customer" is the party (individual, project, or organization) responsible for accepting the product or for authorizing payment. The customer is external to the project, but not necessarily external to the organization; however, the basic definition is the same as stated here and in the ISO standard.

2.1.3 Requirement

In characterizing the elements of quality, a definition of requirement is also needed. There are various definitions of a requirement. For the remainder of this document, we use a general definition of requirement:

> *A requirement is a needed capability, condition, or a property [attribute] that must be possessed by an entity to satisfy a contract, standard, specification, or other formally imposed documents [Cooper 2002, Chrissis 2006].*

Simply put, a requirement is a way of characterizing a need that allows a development team to implement that need in the product in a concrete way. A user, for example an airline, may need an aircraft that is capable of flying 600 passengers 10,000 miles non-stop with high fuel economy. To actually develop the aircraft, more specific characterizations of the need must be expressed. For instance, the aircraft must be supplied with four engines each with 110,000 pounds of thrust.

We must also be aware that as in the CMMI for Acquisition and Software Acquisition Capability Maturity Model (SA-CMM) [SEI 2007, Cooper 2002], there are several types of requirements such as technical, non-technical, product, allocated, user, development, and so on. As a caution, we must be cognizant of the type of requirements being discussed or specified. For example, fixed regulations may also be requirements and may be interpreted as a contract provision, an applicable standard or specification, or other contractual document. Thus, as defined by the Institute of Electrical and Electronics Engineers, a product requirement is a condition or capability that must be met or possessed by a product or product component in order to satisfy a condition or capability needed by the user to solve a problem [IEEE-STD-610]. As noted above, these conditions may consist of a number of diverse types of attributes.

CMMI and ISO Context

While using slightly different wording, both the CMMI and ISO standards definitions of requirement are similar to the one provided in the extended framework. Both stress the documented aspects of a requirement and that it expresses a need to be fulfilled.

The CMMI adds to the above definition of requirement: "A condition or capability needed by a user to solve a problem or achieve an objective."

2.1.4 User

For our purposes, we define user as either the customer or the end user. Typically, three kinds of situations may be encountered in a development or maintenance effort. In the first situation, Figure 1, the customer (either internal or external) and the end user are one and the same.

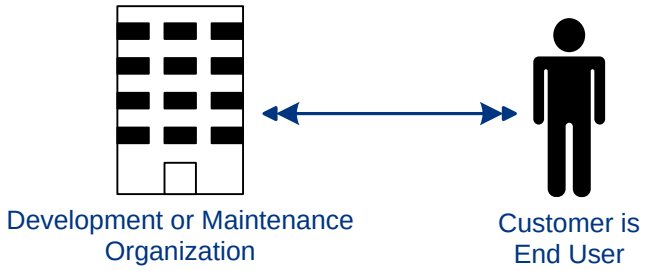

Figure 2: Customer and End User are the Same in Interactions With Organization

In the second situation, Figure 2, the end user is represented by a buyer, and all contact with the client organization is through the buyer. In this case, the buyer represents the user. The face presented by the buyer, therefore, is that of both buyer and user.

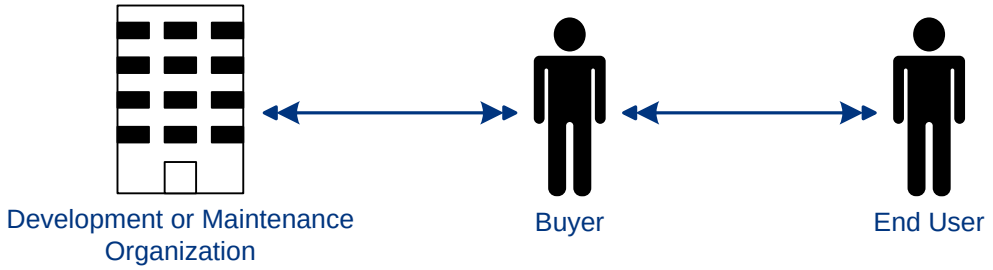

Figure 3: Buyer Represents the End User in Interactions With Organization

The third situation, Figure 3, is where both the buyer and the user community are accessible to the development or maintenance organization.

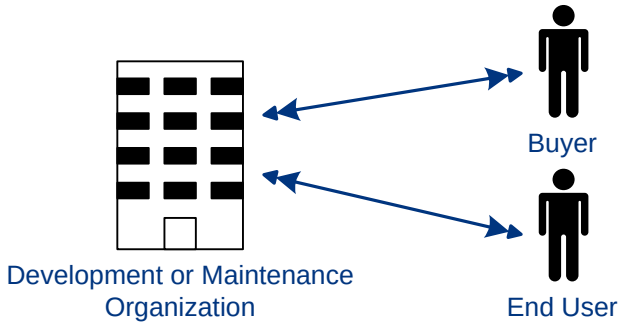

Figure 4: Buyer and End User Both Interact With Organization

For ease of reference, we will use the term "user" in this document to represent all three situations.

The considerations for these three user situations become prominent when we discuss the first element of the framework, *Establish Requirements.*

CMMI Context

The CMMI recognizes that the end user is not always the party involved in establishing the requirements. In the amplifications for Specific Goal 1 of the Requirements Development process area, it points out that

> *"Since stakeholder needs, expectations, constraints, and limitations should be clearly identified and understood, an iterative process is used throughout the life of the project to accomplish this objective. To facilitate the required interaction, a surrogate for the end user or customer is frequently involved to represent their needs and help resolve conflicts. The customer relations or marketing part of the organization as well as members of the development team from disciplines such as human engineering or support can be used as surrogates" [Chrissis 2006].*

2.1.5 Evaluation

As part of the framework, which we discuss below, the concept and definition of evaluation is critical, particularly that of product quality evaluation. As with requirements, the definitions of evaluation are varied. To place evaluation in the context of quality and the framework, evaluation is defined as:

> *Evaluation is defined as the process of determining satisfaction of requirements [Cooper 2002].*

Kenett and Baker give a more precise definition stemming from this basic definition:

> *Evaluation is defined as "a [process] to evaluate the quality of products and to evaluate associated documentation, processes, and activities that impact the quality of the products" [Kenett 1999].*

According to both definitions, evaluations may include methods such as analyses, inspections, reviews, and tests. In this context, evaluation can be applied to product or service development, performing services, producing documentation, creating a process, or any product or work product resulting from a process. In this technical note we are interested in evaluation of quality, specifically, determining process and product quality.

CMMI Context

In the CMMI context, evaluation is covered by Verification, Validation, and Process and Product Quality Assurance process areas while advanced forms of evaluation are covered by the maturity level 4 and 5 process areas.

(With CMMI at maturity levels 4 and 5, product quality, service quality, and process performance objectives are established at the organizational level; then further refined at the project level. Such objectives are then allocated to [derived for] particular product components, subprocesses, interim

life-cycle stages, and suppliers as appropriate; and then the project is managed to ensure it stays on track to achieving these. Evaluations occur in this context as a way of informing how well the project [or individual subprocesses or suppliers] are doing relative to its objectives.)

Keep in mind that at first verification and validation seem quite similar in the CMMI models, but they addresses different issues.

- Validation confirms that the product, as provided, will fulfill its intended use. It ensures "you built the right thing."
- Verification confirms that work products properly reflect the requirements specified for them. It ensures that "you built it right."

2.1.6 Measure and Measurement

As part of evaluation discussed above, we need to include in the Extended Quality Conceptual Framework the ability to measure the quality of processes and products; and, be able to assign actual quantitative values to an item's quality.

Toward this end, Cooper defines the concepts of measure and measurement:

> ***Measure*** *(v.) is to ascertain the characteristics or features (extent, dimension, quantity, capacity, and capability) of something, especially by comparing with a standard.*
>
> ***Measurement*** *(n.) is a dimension, capacity, quantity, or amount of something (e.g., 300 source lines of code or seven document pages of design [Cooper 2002].*

Other definitions for measure and measurement exist that convey similar meanings but reverse the definitions. That is, since the terms measure and measurement can be used as verbs and nouns, the community tends to use them interchangeably. For example, Fenton describes measurement as the process by which numbers or symbols are assigned to attributes of entities in the real world in such a way as to characterize the attributes by clearly defined rules [Fenton 1995]. This implies an action such as "to measure" in the above definitions.[4]

In addition, use of the terms metric and measure has become somewhat confused. In most cases they are used interchangeably, but be aware that in some cases various authors have made attempts to distinguish the two terms (and they have done so in different ways) [Melton 1996, Fenton 1991]. The IEEE Standard Glossary of Software Engineering Terms defines Metric: "A quantitative measure of the degree to which a system, component, or process possesses a given attribute." Note that the word metric is not used in ISO 15939 or in CMMI (except in one rare instance where the Goal Question Metric approach is mentioned) [ISO 15939].

Measurement plays an important role in the EQCF. As we will later see, an element of the EQCF is quality evaluation. We are interested in evaluating the compliance of the product and processes

[4] There are numerous reference works on the subject of measurement—[Krantz 1971], [Roberts 1979], and [Ghiselli 1981] for example. Some of these (for example, [Ghiselli 81]) provide excellent discussions of important issues such as measurement reliability and validity.

with applicable standards and requirements. One means of achieving that is through measurement. Consequently, we are concerned with the act of measuring, as well as the measure itself (both the verb and noun forms of the word). Anyone familiar with the discipline of measurement and analysis knows the importance of operational definitions for the measures selected. Without them, there is ambiguity about what is being measured and how it is being used. Likewise, an operational definition for measure and measurement is important in explicating the EQCF in order to avoid ambiguity about what it is that the EQCF is intending to accomplish.

CMMI Context

The CMMI uses the term measure both as a verb and as a noun, and the term measurement as a noun as described above. CMMI also uses the terms base measure and derived measure. A base measure is defined as "A distinct property or characteristic of an entity and the method for quantifying it." A derived measure is defined as "Data resulting from the mathematical function of two or more base measures" as shown in Table 1.

Examples of base measures	Examples of derived measures
SLOC	Productivity
Staff-hour	Miles/gallon

Table 1: Examples of Base and Derived Measures

Specifying the expected range and/or type of values of a base measure helps to verify the quality of the data collected.

2.1.7 Quality

A prime focus of the EQCF is, of course, the definition of quality and its implication for the implementation of the remainder of the EQCF elements. In the EQCF, quality is defined as in this reference:

> *Quality is the degree to which an object (entity) [e.g., process, product, or service] satisfies a specified set of attributes or requirements [Cooper 2002].*

However, it is important to point out that

> *A product possesses many quality attributes that are intrinsic to the product and which exist regardless of what is desired, specified, or measured, and only depend on the nature of the product [Baker 1982a].*

Thus, the definition of quality includes two aspects:

- the concept of attributes
- the satisfaction or degree of attainment of the attributes

2.1.7.1 Attributes

An attribute is a property or characteristic of an entity that can be distinguished quantitatively or qualitatively by human or automated means [ISO 15939]. The word attributes includes all specified requirements governing functional, performance, and other specified characteristics such as adaptability, maintainability, and correctness [Baker 1982a, Baker 1982b]. The attributes (that is requirements and other specified characteristics) are considered the determinants of product or process quality.

2.1.7.2 Specifying Product Quality Using Product Attributes[5]

The word "specified" implies that definitions of the needed quality attributes are documented and that the attributes themselves are assigned a value or range of values. Without clear articulation of the quality attributes, it is impossible to develop a product or determine whether the finished product has the needed quality. A specification is required to communicate to others which attributes and attribute values constitute the product's quality. Contractually, this specification is critical [Baker 1982a].

In addressing product or process quality, it is therefore necessary that the specification or definition of the attributes be expressed quantitatively. This quantitative expression allows a determination of the degree to which a product or process satisfies or attains the specified attributes. For example, saying that a hardware product has high reliability ("reliability" is the attribute) does not tell us how reliable the hardware is. Stating that the reliability of a product is 100 hours mean time between failure (MTBF) (providing both the definition and a value for an attribute) expresses a characteristic that can be measured and compared against, which means there is a method used to determine if the specified attribute has been attained.

Firesmith expresses the same concepts in terms of quality requirements. He states that quality requirements consist of three main parts, as shown in Table 2 [Firesmith 2006]:

Quality Requirement	Description
Condition	An optional condition that states under what conditions the quality requirements must be met. For example quality requirements specifying high performance and reliability may only hold during normal conditions, but not under degraded mode operations.
Quality Criterion	A quality criterion is a system-specific description that provides evidence either for or against the existence of a given quality factor or subfactor.
Quality Threshold	A quality threshold specifies a minimum level of quality along a quality measurement scale.

Table 2: Quality Requirements

[5] Some quality attributes typically called "ilities" that have been examined include without definition: availability, capacity, efficiency, interoperability, modifiability, performance, portability, produceability, reliability, reusability, robustness, safety, scalability, security, testability, and usability.

For implementation of a product, one selects those attributes most significant to the user community, assigns them values (to be used as criteria) in accordance with user needs, and then evaluates, rates, or measures the product's quality on how well, or to what degree, the selected attributes meet those criteria for excellence. Often, these attributes address only functionality and performance and ignore the other attributes, often referred to as "ilities." The "ilities" can be considered as attributes that address fitness for use. Conceivably, "if we look at the issue of the software meeting its requirements, and if those requirements are solely functional and prescribe no ilities, then clearly the software can meet the requirements but could be unable to fulfill any reasonable purpose" [Voas 2004].

Thus, we recognize that

> *Just as beauty is in the eye of the beholder, so is quality [Baker 1982b].*

Consequently, a set of attributes that one community deems important as a measure of quality may not be deemed important by another community. Rather, each community is likely to have its own set of attributes and attribute values with which to measure quality.

2.1.7.3 Considering User Needs

It is difficult to satisfy users if they cannot articulate what quality they are expecting. In many cases, users default to "give me something and I will tell you if I like it or not." Such a paradigm wastes time and resources in trying to satisfy the illusive (and elusive) user expectations. Clearly, vague notions such as "user needs," unless they are articulated, cannot be used to determine the quality actually achieved in a product. Something concrete, such as a user specifications document, can be used. Obviously, the requirement to accurately capture the user needs in such a document is crucial. Typically, user needs are captured in operational needs documents, which are then decomposed into system requirements documents and then further decomposed into software and hardware component requirements documents. All of these start from the documented user's needs. A description of this set of activities can be found in the Requirements Development process area of CMMI.

The fact that product quality requirements include the functionality and performance requirements and may include requirements for maintainability, portability, interoperability, and so on, leads us to the key point that

> *Product quality requirements stem from many sources, above all, from the stakeholders of the project.*

And, this leads us to the idea that

> *Quality is everybody's business [Baker 07].*

Or, if we consider how product development projects are organized, the implication is

> *Quality is affected by many, but effected by few [Baker 07].*

Finally, one must realize that the final quality of a product results from activities performed by the project developing the product. Everything that occurs within a project during development affects some attribute of the product and, therefore, the total product quality. However, all possible attributes may not be of equal relevance. Furthermore, all actions may not affect the specified attributes to the same extent and, therefore, the specified quality. In any event, quality is affected by activities such as requirements definition, design, coding, testing, and maintenance of the product—activities associated with the Extended Quality Conceptual Framework for the product, and the interaction of these activities as we will discuss below.

CMMI Context

In the introductory notes of the Requirements Development process area, we find

"The needs of stakeholders (e.g., customers, end users, suppliers, builders, and testers) are the basis for determining customer requirements. The stakeholder needs, expectations, constraints, interfaces, operational concepts, and product concepts are analyzed, harmonized, refined, and elaborated for translation into a set of customer requirements."

The link between quality requirements and user needs is contained in the CMMI definition of quality: "The ability of a set of inherent characteristics of a product, product component, or process to fulfill requirements of customers." The CMMI also addresses the issue of whether all attributes have equal relevance. In the Requirements Development process area, subpractice 4 of Specific Practice 3.3 states, "Identify key requirements that have a strong influence on cost, schedule, functionality, risk, or performance."

2.1.8 Performance

As with the word "quality" the word "performance" has been over used and misinterpreted. We note the use of such terms as project performance, process performance, and performance of the product.

We define performance in a general sense and as with all the definitions discussed here the general definition can be adapted to a particular situation by adding a modifier, e.g., project performance. Our definition is

> *The degree to which an entity accomplishes specified requirements within given constraints in the execution of work or implementation of a function*

As an aside, project performance is typically defined in the dimensions of cost, schedule, and quality of the entity being developed, e.g., a software development project. These dimensions are the specified requirements in the definition of performance above.

CMMI Context

CMMI uses the terminology of quality and performance explicitly at organizational maturity level 4 where it expects practices to establish the project's quality and process performance objectives and then manage to these objectives. CMMI notes that quality is considered part of process performance but is broken out for emphasis on quality [Chrissis 2006].

2.2 ELEMENTS OF THE EQCF

The previous definitions and concepts provide a foundation for the remainder of the framework and especially the concept that many people supporting a project affect the quality of the product.

> *Embedded in the implementation of these three elements is the overall approach to effect and determine the level of quality achieved in a product [Baker 1982b].*

The three elements of the framework cover the activities necessary to

Establish Requirements and Control Changes
Establish and specify requirements for the quality of a product
Establish and Implement Methods
Establish, implement, and put into practice methods, processes, and procedures to develop, operate, deploy, and maintain the product
Evaluate Process and Product Quality
Establish and implement methods, processes, and procedures to evaluate the quality of the product, as well as to evaluate associated documentation, processes, and activities that have an impact on the quality of the product

Figure 5 (which is the same as Figure 1 and repeated here for convenience) illustrates the interrelationships of these elements and the interrelationship of the EQCF with a product's design and implementation activities. We will discuss some of these relationships later. This interaction is continuous with the design and implementation activities affecting the EQCF activities. The EQCF addresses both technical and management activities. For instance, ensuring quality is built into a product is a management activity, while specifying the methods used to build in the quality is considered a technical activity.

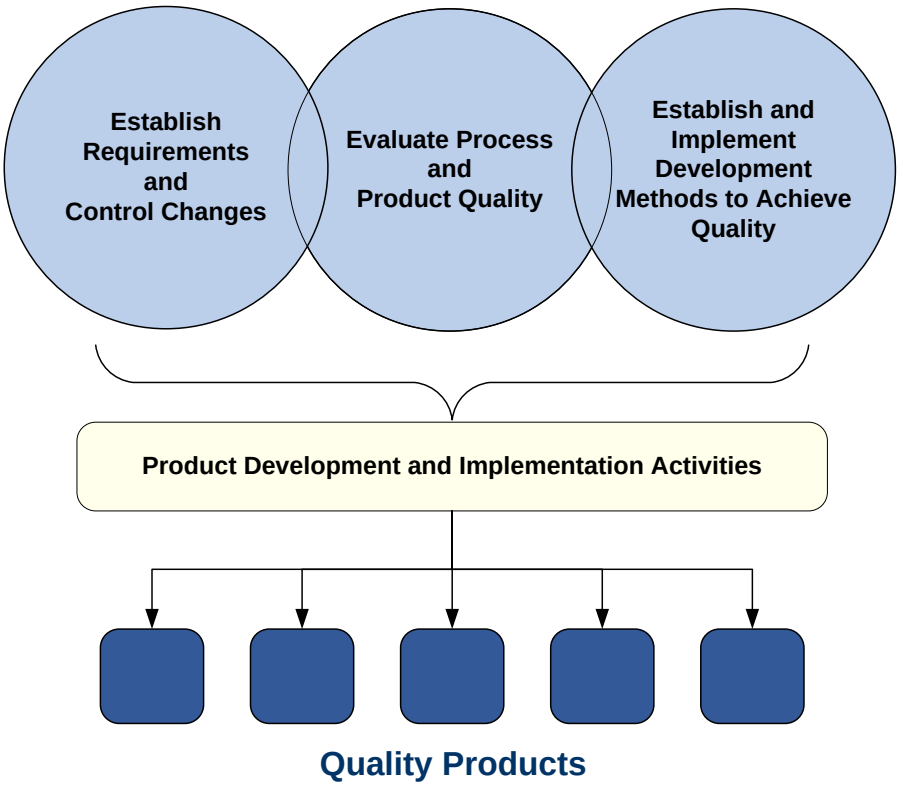

Figure 5: Interaction of Activities between Extended Quality Conceptual Framework and Development Project

One of the foundational aspects of the EQCF is how well quality can be built into a product, not how well one can evaluate product quality. While evaluation activities are essential activities, they alone will not achieve the specified quality. That is,

> *Product quality cannot be evaluated (tested, audited, analyzed, measured, or inspected) into the product. Quality can only be "built in" during the development process [Baker 2007].*

Once the quality has been built in, the deployment, operation, and maintenance processes must not degrade it. Unfortunately, the very nature of maintenance and bug fixes for software often degrades the quality of the code. What was once structured code becomes messy "spaghetti" code with all the modifications resulting from bug fixes and enhancements.

To elaborate further on the issue of ensuring that the deployment, maintenance, and operation processes do not degrade the quality built in, Figure 6 provides a notional example of when the quality of the software likely has been compromised and qualification test procedures should be re-executed or the software should be re-engineered as a result of maintenance. In the example in the figure, the organization set a threshold level of 30 percent of the modules changed per software maintenance request as the point at which re-qualification of the software should take place. (The 30 percent threshold is for illustrative purpose only and should not be considered a norm or standard.)

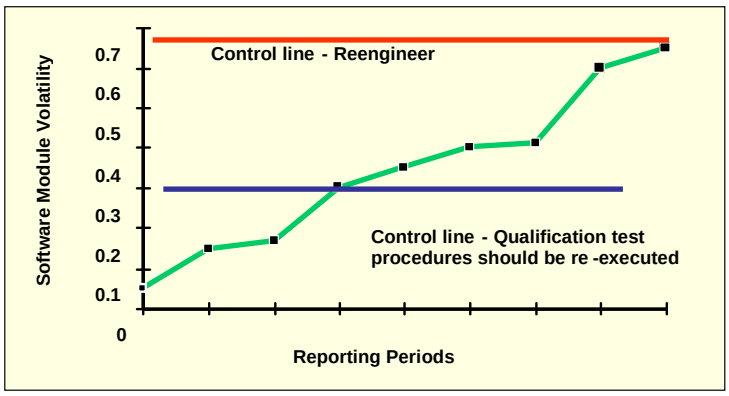

Figure 6: Software Module Volatility

Note that in Figure 6, the software is approaching the threshold level at which it should be re-engineered. Clearly, in this situation, there is a great potential for the quality of the software to be degraded as a result of maintenance activities. This quality degradation is shown in Figure 6 which is based on the following equation:

$$\textbf{Software Module Volatility} = \frac{\text{Number of modules changed due to software maintenance request}}{\text{Total number of modules in a release over time}}$$

The EQCF concept does not impose any organizational structure for performing the activities; however, organizations are responsible for assigning resources to accomplish these activities. We do suggest that organizations, especially at the corporate level, avoid assigning certain roles to carry out the activities without clearly understanding the concept of product and process quality and how those are affected and implemented.

The idea of many people affecting the product quality should be obvious from the fact that so many disciplines are involved in analyzing and working to achieve the product's quality requirements. Virtually everyone working on the project, from the project manager (PM)[6] to the most junior member of the staff, affects the quality of the product. However, only those actually producing the product (performing tasks such as requirements analyses, design, and manufacturing/coding) build quality into the product. For example, even though software projects code reviewers affect the quality of the resultant product, like testers, the reviewers do not actually develop the design or implement it.

Thus, it is important to understand that the person ultimately responsible for the quality of the product is the PM. It is the project manager's responsibility to integrate the efforts of "the many" to accomplish the project's quality objectives. The PM may, of course, may delegate authority for

[6] A project manager is an individual assigned the management responsibility for acquiring, developing, or providing a product or service. The PM has total business responsibility for an entire project and is the individual who directs, controls, administers, and regulates the project. The PM is the individual ultimately responsible to the customer or end user [Cooper 2002].

any part of this function, but ultimately he or she is responsible. This is reflected in the often-used phrase:

> *While authority may be delegated, responsibility cannot.*

CMMI Context

Delegation of authority is a critical point especially when it comes to buyers who typically assume that suppliers (developers) are totally responsible for the quality of products being procured.

The Extended Quality Conceptual Framework is multi-disciplinary. In organizations that have adopted the CMMI, coordinating the activities illustrated in Figure 2 above is often assigned to a centralized function, sometimes referred to as a Process Group (PG), Engineering Process Group (EPG), or in organizations that are primarily software development organizations, a Software Engineering Process Group (SEPG) [Fowler 1990]. A discussion of how the multi-disciplinary nature of the quality framework has been implemented in some organizations is provided in Fowler and Rifkin [Fowler 1990] and in the "Software Quality Program Organization" chapter [Baker 2007] of *The Handbook of Software Quality Assurance* [Schulmeyer 2007].

2.2.1 Establish Requirements and Control Changes

The first element or set of activities of the Extended Quality Conceptual Framework deals with establishing and controlling changes to requirements.

> *Product requirements must accurately reflect the product's desired overall quality, including functionality and performance, and must be documented and baselined (formalized).*

As noted previously, the requirements must accurately reflect the needs of the user community. The process for defining the requirements must ensure that the needs of all the stakeholders involved with the end product have been accurately captured. Thus, a process for establishing and controlling the requirements must be established. This indicates an interface with the second element of the EQCF: *Establish and Implement Methods*.

One problem associated with specifying a product's quality requirements is the inaccurate perception that the quality cannot be stated quantitatively or in a way that permits objective evaluation. As noted earlier, communicating the quality of an entity to others becomes difficult because people tend to interpret the quality of the same entity from their own perspective. The result is that verifying achievement of the desired quality can be quite subjective. Consequently, the methodology established must ensure that ambiguity is reduced and verifiable quality criteria are specified.

Simply defining and formalizing product requirements are insufficient. The baseline product requirements must be strictly adhered to and fully implemented. Failure to implement the requirements as specified can result in products that do not meet user needs and derived requirements. The resultant impact on product quality, such as functionality and performance, will range from

negligible to severe. It follows that any changes to the product requirements must be controlled and documented, and the affects of those changes understood.

The activities of defining and establishing the requirements and controlling changes to them necessarily involve interfaces with the other two elements of the Extended Quality Conceptual Framework: *Establish and Implement Methods* and *Evaluation of Process and Product Quality*.

To illustrate the interface between the first and latter two elements of the Extended Quality Conceptual Framework, an organization may establish, for example, methods specifying the use of data flow analysis, use cases, or object-oriented analysis for performing requirements analysis. Whatever method is selected and used must provide high confidence that the users' needs have been captured accurately; consequently, the evaluation of the requirements analysis process must demonstrate that it was followed and is effective in capturing the users' needs, and the evaluation of the requirements must indicate that the users' needs were captured correctly for the instances examined.

As another example, when establishing baseline requirements and controlling changes, a configuration management method must be selected and later implemented to

- establish a baseline as a reasonably firm point of departure from which to proceed into the other phases of project activity knowing that there is some reasonable degree of stability in at least the set or subset of requirements that were established
- prevent uncontrolled changes to the product baseline
- improve the likelihood that the development effort results in a quality product and that processes in subsequent life-cycle phases will not degrade it

Again, the process evaluation must demonstrate that the process was followed and is effective, and the product evaluations must demonstrate the correctness of the outputs.

A second interface between elements of the Extended Quality Conceptual Framework exists. It is between the *Establish Requirements and Control Changes* element and the *Evaluation of Process and Product Quality* element. It is concerned with two things: the evaluation of the product against the requirements, (as accomplished, for example, in the Verification and Validation process areas of the CMMI [Chrissis 06]) and the determination that the process was adhered to for defining requirements (the purpose of GP 2.9 of the Requirements Definition process area [Chrissis 06]). Total compliance with requirements does not guarantee a quality product if the requirements are not properly defined and errors exist in them. Then compliance with requirements produces a product that does not satisfy the intended end use. Clearly, evaluations/audits for compliance with the process for establishing requirements must be performed. Furthermore, the process and method by which the requirements are developed must be evaluated for adequacy during the development or maintenance process.

CMMI Context

These concepts from the Extended Quality Conceptual Framework are captured in the CMMI in the Requirements Management, Requirements Development, Process and Product Quality Assurance (PPQA), and Organization Process Focus (OPF) process areas. In the CMMI, the Require-

ments Management process area involves understanding requirements and then managing changes to them. The Requirements Development process area involves actively eliciting requirements from users or customers of the development project and then analyzing them for feasibility, conflicts, and balance, as well as refining, documenting, and baselining them. This particular process area also takes into account other stakeholders. Typically these other stakeholders add constraints and quality requirements the user or customer may not articulate. PPQA involves performing the necessary reviews or audits to ensure that the required process is followed and that the resultant work product (for example, a specification) complies with pre-established standards. OPF describes a process for collecting and reviewing recommended process improvements resulting from appraisals of the process and from internally generated suggestions.

All of these process activities within CMMI reflect the concepts and principles of the Extended Quality Conceptual Framework.

2.2.2 Establish and Implement Methods

The second element or set of activities of the Extended Quality Conceptual Framework involves selecting, implementing, and putting into practice the appropriate processes, practices, and methods to build quality into the product and achieve the specified quality requirements. This is typically accomplished by codifying these processes, practices, and methods as standards and training the organization and project teams to use them. These standards may be tailored to meet the unique needs of the project in accordance with established tailoring guidelines (or as defined processes, in the context of the CMMI).

The act of getting these standards into use is accomplished by corporate management, who can consistently mandate the application of the selected methods from project to project even under conditions of schedule pressure. The enforcement can be through various means, for example, assignment of appropriately trained personnel, or monitoring and controlling the project against a project plan, or both. The important point is that requiring compliance with standards is the responsibility of management and not that of some other organizational entity, for example, a quality assurance group. This does not preclude tailoring the methods to be consistent with the unique characteristics of the various projects. Tailoring should be permitted to account for the fact that there may be considerations that would mitigate against full compliance with the organization's standard process for some projects, or that perhaps might require additional, more stringent process steps. Guidelines should exist to cover those cases, and the resultant tailoring should be subject to review and approval. Nor does compliance with the organization's standard process preclude the acquisition, development, or use of different methods, but the appropriate approvals for their introduction must be observed. Reviews and audits of project activities for adherence with the standard processes are performed to provide visibility or insight to management. Reviews or audits for adherence do not necessarily constitute enforcement; they can only determine if compliance has occurred. Management should use the result of audits to exercise its leadership responsibilities by active monitoring and control of project activities.

Generally, the basis for determining which processes to address when establishing standards is to look to past history. If a process produced "high-quality products" on some past projects, it is believed that properly implementing the process on an organizational basis should result in a high-quality product now. However, establishing standards in this manner can be somewhat presump-

tuous. The link between methods selected for product development and the resultant product quality has not been demonstrated quantitatively because establishing links is more heuristic or intuitive than analytical. For example, Ada was touted as promoting information hiding, which, in turn, should make the product more adaptable. But, to our knowledge, the actual quantitative cause and effect link has never been documented.

CMMI Context

The CMMI has two process areas that are specifically focused on process definition and process improvement. These are OPF and Organizational Process Definition (OPD). OPF determines the organization's process needs and objectives; appraises the organization's standard processes; identifies potential improvements; and effectively deploys selected process improvements across the organization. OPD focuses on establishing process assets, e.g., standard processes, procedures, templates, life cycle models, measurement repository, tailoring guidelines, and the like. Another process area, Organizational Innovation and Deployment (OID), is a refinement to the activities of OPF and OPD in that its focus is on achieving quantitative objectives for quality and process performance, quantitative assessments of proposed process and supporting technology changes to determine their efficacy, systematic deploying selected changes, and follow-up quantitative assessments to determine if the changes are producing the desired results in practice.

2.2.3 Evaluation of Product and Process Quality

The third element or set of activities of the Extended Quality Conceptual Framework involves evaluating the implementation of processes and evaluating the quality of the resulting product(s). Evaluations are used to assess

- the quality of the product
- the adequacy of the processes and activities responsible for product quality
- compliance with established processes

2.2.3.1 Evaluation of Product Quality

Quality evaluations help to determine the "health" of the product and hence the project. Through evaluations, the project can determine whether its product satisfies the specified quality requirements within cost and schedule constraints. Because of the number of organizational groups or functions typically involved in quality evaluation activities, coordinating the results of this process should be performed by the PM. Regardless of which groups are involved, management must be sure that all quality evaluation activities are assigned to competent, unbiased reviewers.

Evaluation activities include reviews, assessments, tests, analyses, inspections, and so on. Depending on the action taken and the processes or products being evaluated, the results may be qualitative or quantitative in nature.

Any evaluation that requires reasoning or subjective judgment is referred to as an assessment. Assessments include analyses, audits, surveys, and both document and project performance reviews. On the other hand, measurement activities are the basis for quantitative evaluations such as tests, demonstrations, metrics, and, in some cases, inspections using checklists (although many checklists only allow a subjective evaluation). Accordingly, quantitative evaluations can include

tests for unit level, integration, and product or application level performance, as well as the output of a compare or a path analyzer program.

Evaluation activities will vary with each phase of the development cycle. Furthermore, they can be performed by individuals independent of the project, or one or several independent organizational units. (We discuss the concept of independence in more detail in Section 3.1). Evaluation activities to be performed and responsibility for them are generally defined and documented in project management plans, product development plans, project-specific product quality evaluation procedures, and/or company quality plans and related product quality evaluation procedures.

2.2.3.2 Evaluation of Established Processes

Another form of process quality evaluation is doing reviews and audits for compliance to or adherence with the process. It is one thing to specify a process, but if that process is not being followed, the quality of the resultant product can be adversely affected. Periodic audits for compliance with the process need to be performed to ensure that the established process is being implemented. For example, external appraisals, such as a Standard CMMI Appraisal Method for Process Improvement (SCAMPI[SM]) B or C appraisal, are helpful in this regard.

It is also important to evaluate the processes used to determine if these processes are producing products that yield the required quality. Using a concept from the CMMI to explain how processes are evaluated, low maturity organizations will do qualitative evaluations supported, in some cases, by rudimentary quantitative methods. High maturity organizations will use measurement and statistical analysis to determine how effective the processes have been. In the low maturity organization case, the evaluation will in most cases be experiential. For example, did the process implementers commit mistakes and thus have to perform rework as a result of following the process? In other cases, the organization might collect simple data, for example, defect data. If the number of defects appear to be large (a subjective determination at lower CMMI maturity levels), then investigations will be performed to figure out where in the overall scheme of things the defects are being introduced (e.g., by performing a root causal analysis as described in the Causal Analysis and Resolution (CAR) process area). In the high maturity cases, quality and process performance objectives are established, processes will be quantitatively monitored (using statistical analysis for the most critical subprocesses) and corrective action implemented when process performance fails to achieve the established objectives.

CMMI Context

In terms of evaluation of both product and process quality, these evaluation concepts are articulated in several CMMI process areas, such as Process and Product Quality Assurance (PPQA), Verification (VER), Validation (VAL), and CAR.

The primary product evaluation process areas are VER and VAL. The purpose of VER is "to ensure that selected work products meet their specified requirements" [Chrissis 2006]. An example of this would be testing a circuit board model to ensure that it is designed properly, or examining code to ensure that the design is implemented properly in the software. The latter might be per-

[SM] SCAMPI is a service mark of Carnegie Mellon University.

formed as a peer review. On the other hand, the purpose of VAL is "to demonstrate that a product or product component fulfills its intended use when placed in its intended environment" [Chrissis 2006]. This may involve a test program executed in the target environment or a simulation to determine whether the product or product component will adequately address stakeholder needs. Clearly, the determination that the "product or product component fulfills its intended use" is highly dependent on how well the requirements are defined. (Note: Validation is generally applied throughout a development project, not only at the end of the project.)

Other CMMI process area indirectly associated with product quality are Measurement and Analysis, Quantitative Project Management (QPM), CAR, and Organizational Process Performance (OPP). Measurement and Analysis addresses "information needs" relative to product quality (and other product and process attributes) by establishing appropriate measurement and analysis activities. The purpose of QPM is "to quantitatively manage the project's defined process to achieve the project's established quality and process-performance objectives" [Chrissis 2006]. The focus is on projects, using quantitative and statistical methods to measure quality and process performance to ensure that the end product meets quality and process performance objectives. The CMMI discusses the use of statistical methods to monitor and control the most critical processes from a business perspective. Here, both the process and the product are often being measured to determine satisfaction of quality and process performance objectives, not just directly but also in that product quality measures are often a component of the measures used to evaluate process performance. The purpose of CAR is to identify causes of defects and other problems and take corrective action to prevent then from occurring in the future. In many cases CAR is applied to systematically address patterns of product defects and other problems through a root causal analysis.

The primary process evaluation process area is PPQA. A focus of this process area is to "ensure that planned processes are implemented" [Chrissis 2006]. This is accomplished in conjunction with Generic Practice (GP) 2.9, which appears in all process areas. GP 2.9 establishes the need to "objectively evaluate adherence" [Chrissis 2006] for each of the processes. In a sense, Specific Practice (SP) 1.1 of PPQA is a collection point for all of the GP 2.9 evaluations. The full text of SP 1.1 reads, "Objectively evaluate the designated performed processes against the applicable process descriptions, standards, and procedures" [Chrissis 2006].

In terms of determining the adequacy of the implemented processes to produce quality products, the CMMI establishes a focal point for accomplishing this with the Organizational Process Focus and OPP process areas. Included in the activities under OPF are responsibility for accomplishing periodic appraisals of the implementation of the processes and collecting feedback from the projects on the effectiveness of the established processes. Included in OPP are activities related to determining the process performance of the organization's standard processes (and in particular that of selected subprocesses) in order to better ascertain the ability of the organization and its projects to achieve organizational objectives for quality and process performance. As indicated earlier, the PPQA process area is the focal point for performing reviews and audits for process adherence.

3 Considerations

In the previous sections, we discussed the concepts that comprise the Extended Quality Conceptual Framework. The following sections consider factors important to implementing the Extended Quality Conceptual Framework.

3.1 THE CONCEPT OF INDEPENDENCE

As part of quality evaluation activities, we discuss the concept of independence.

> *Relative to quality evaluation, independence implies performing product quality evaluations by an "outside" organization (or individuals).*

In this case, "outside" means different from those that produced the product, or those that executed the processes and activities being evaluated. The concept of independence relates not only to performing the evaluation, but extends to establishing the evaluation criteria. Such criteria should be established during Project Planning (a real "best practice") so that there is agreement on the scope and character of the evaluations and the right expectations are set "up front" among involved parties.

The need for independence arises because the persons performing the process or creating the product may have a conscious or unconscious need to make the process or product look good. They might also have a biased expectation of what the result should be and consequently would fail to perform certain checks or could miss anomalies because of that expectation. Such evaluators can hardly be considered independent. By removing from them the responsibility for establishing the evaluation criteria and performing the evaluations, such problems will be substantially reduced.

Independence, as a concept, has two aspects:

- Independence exercised within an organization, such as the use of a test team composed of individuals different from those who designed and developed the product.
- Independence exercised by utilizing a group from outside the organization, such as an independent verification and validation agent. This form of independence is perhaps the most stringent.

Either way, the notion of independence is applied to reduce errors resulting from extensive familiarity with the product being evaluated. Decisions as to the application of independence, the degree of independence to apply, and the types of independent agencies to employ are a function of a number of variables, such as size and complexity of the product/project, corporate policy, available funds, and criticality of the product to its end use (human safety, destruction of equipment, severe financial loss, and so on).

CMMI Context

The CMMI supports this notion of independence in stating that "Objectivity [in process and product quality assurance evaluation] is achieved by … independence … ." It may be appropriate in some organizations to implement process and product quality assurance independence by having a separate quality assurance organization that reports independently to top management. In other organizations, that role may be implemented without that kind of independence. For example, in an organization with an open, quality-oriented culture, the process and product quality assurance role may be performed, partially or completely, by peers and the quality assurance function may be embedded in the process. As may be seen in the Process and Product Quality Assurance process area, there is no need for a separate, enterprise-defined QA group.

If the quality assurance function is embedded in the process, several issues must be addressed to ensure objectivity.

- Everyone performing quality assurance activities should be trained in quality assurance.
- Those performing quality assurance activities for a work product should be separate from those directly involved in developing or maintaining the work product.
- Those performing reviews or audits for process compliance should be separate from those who actually performed the process.
- An independent reporting channel to the appropriate level of organizational management must be available so that noncompliance issues may be escalated as necessary.

3.2 QUALITY EVALUATION VERSUS QUALITY ASSURANCE

A point of confusion, especially related to organizational aspects of the quality framework is the role of quality assurance groups and the quality evaluation activities those groups may perform. Part of the confusion stems from a misunderstanding by many project teams about the word quality and the belief that anything to do with product or process quality is the purview of a QA organization. The belief is that if a Quality Assurance organization exists and is assigned to the project, that this will ensure quality in the product. This belief contradicts the principle that "Quality is everybody's business." Quality does not occur from after the fact inspections, audits, or tests. As once noted by Harold Dodge of Bell Telephone Labs, "You cannot inspect quality into a product" [Schulmeyer 2007]. Put another way, quality evaluation only results in a determination of the quality or level of quality that has been built in.

Further confusion results from a poor understanding of the distinction between the concept of quality and the quality functions associated with development and maintenance activities. For example, there is a blurred understanding of the difference between QA functions and QA groups. For example, if a QA group exists in a corporate structure and only performs checklist types of audits, a project manager may mistakenly conclude that this group is sufficient to satisfy all the quality evaluation needs of the project. In performing checklist activities, the group may not go into enough depth to ascertain whether product quality or process adherence is being achieved. This is especially true if QA personnel are not well versed in the disciplines being reviewed or audited. This situation may preclude vital measurements (or tests) on critical parts of the product or process.

Compounding this problem is the difference in approaches to quality assurance from organization to organization. In some organizations, the QA group may have a considerable amount of technical expertise and perform some of the necessary quality evaluations. In other organizations, it may be merely a checklist activity involving little more than asking a performer if he or she has performed the required activity—a technique which produces questionable results. Consequently, we see that there is little in the way of a consistent definition in the industry of what constitutes the discipline of quality assurance.

The fact is, quality evaluation activities often are performed by many groups, not only quality assurance groups. Typically, these activities are conducted both formally and informally and include

- peer reviews involving other developers to determine code quality (for software)
- bench tests to evaluate the design of a hardware component
- process effectiveness reviews conducted by the multi-disciplinary members of the Engineering Process Group
- document reviews conducted by subject matter experts
- functional- and system-level testing conducted by development team members and testing team members
- integration testing conducted by testing team members, other development groups, and business partners
- usability testing conducted at many development milestones to determine quality related to user interaction

Some of the above quality evaluations may also be conducted by QA groups depending on their level of expertise.

Regardless of which group (or groups) actually performs the evaluations, it is necessary to assess and measure not only product quality but also the processes used to achieve it. This is the only way to make sure enterprise, project, and the specified user requirements are met. Once the quality evaluation activities are defined, management can assign them to the appropriate group or groups.

CMMI Context

While not explicitly placing quality assurance within the organizational context, the CMMI clearly defines the quality assurance function (in the PPQA process area) as

- objectively evaluating performed processes, work products, and services against the applicable process descriptions, standards, and procedures
- identifying and documenting noncompliance issues
- providing feedback to project staff and managers on the results of quality assurance activities
- ensuring that non-compliance issues are addressed

In CMMI, the evaluation of the actual quality attained in the product or the evaluation of the adequacy of the process is not addressed within the PPQA process area. These activities are covered

by several other CMMI process areas (e.g., VER, VAL, and QPM). Whichever decision is made relative to who performs these evaluations, management must be sure that they are assigned to a competent party that can effectively perform those functions and, where independence is specified, to an organization with the proper degree of independence.

What is crucial to any development project is that the definition and implementation of the activities necessary to assess and measure the quality of products produced by that project, in accordance with the requirements established for that project, are carried out. When the quality assurance activities are defined, their assignment to a specific organization is a management prerogative. In summary, while quality assurance may be a functional entity within a corporate structure; it does not address the full range of quality evaluations needed for the project and does not release the PM from overall responsibility for quality.

3.3 PROCESS QUALITY VERSUS PRODUCT QUALITY

Intuitively, people cannot help but believe that implementing a structured process will improve system and software quality. To come to that conclusion, simply look at the results of many software development projects that have produced disastrous results because no process was in place. Based on what is for the most part anecdotal evidence, the maxim that "the quality of a product is highly influenced by the quality of the processes used to produce it" tends to be true [Humphrey 1989]. We know qualitatively that organizations implementing a disciplined process tend to have better, more consistent results. On the other hand, experience also says that this is not always the case. Because of this inconsistency, a question arises: *Will a quality process always, in fact, produce a quality product?*

This question comes to mind because of the phrase that the product quality is only "highly influenced" by the process quality. The implication is that there is no guarantee that a quality process will, in reality, produce a quality product. This uncertainty is supported by Jeffrey Voas in an article in IEEE Software [Voas 1997]. The article points out that the relationship between process quality, methods selected for product development, and quality of the delivered code has never been established *quantitatively*. He likens the situation to a belief that clean pipes will always deliver clean water. This belief disregards the fact that there are other ways that dirty water can be injected into the water supply, despite the clean pipes. Further, the article states that there is no guarantee that even the most superb software development processes will manufacture correct code.

However, there is empirical evidence supporting our intuitions that there is a connection, i.e., a cause and effect relationship [Gibson 2006]. In addition, while we can acknowledge that there is currently no guarantee, work in this area address the situation of a "mature supplier" working with a "low maturity customer" in a way that supports process improvement and use of mature proven processes. A work in progress studies the so-called problematic programs (mature programs not performing well) and discovered the water coming from clean pipes is much more likely to be clean than is the case for dirty pipes (less mature organizations), even in dirty homes (immature customers); i.e., while there is no guarantee, odds of successful programs with quality products are improved if the processes are mature.

The same theme is also expressed by Bazzana with respect to some findings from a study on the efficacy of ISO certification by itself [Bazzana 2099]. In his paper, he points that in a European-wide survey of IT representatives, the findings showed that approximately 65 percent of them felt that certification against ISO 9000 alone did not provide sufficient guarantees for the quality of the delivered product. They felt that an assessment or appraisal involving two (or more) process models produced greater confidence. Compounding this implication is of course the definition, or lack of a definition, of quality for the process and the definition, or lack of definition, of the quality of the resultant product.

In discussing product quality as it relates to process maturity models, Tully, in *Better Software Practice for Business Benefit*, discusses the results of a European-wide awareness survey of IT representatives concerning the relationship between product and process quality. The survey revealed that "process certification according to ISO 9001 as not enough to guarantee quality of a specific software product" [Bazzana 1999]. Of the responders, 64.62 percent answered NO to the question "Do you think that certification against ISO 9000 is enough to guarantee the quality of software products?" Also, a very low percentage of the interviewed declared that such certification can give a sufficient guarantee of the quality of the delivered products. Nearly 57 percent responded that the "Quality of a software product is not guarantied by a certificate on its production process." Another 38.46 percent responded that "it is more important the results of the process, i.e. the product, than the process itself." These observations lead to Tully pointing out that "process maturity models are unconcerned with cause-effect relationships. They are based on a very simple proposition, that software product quality is a function of software process quality. Such a statement of a static equality can be vastly misleading, because it ignores the real-world chains of events which are interposed between a process change and a product change, the speed at which they unfold, and the extent to which they are affected by other causes."

As noted above, one of the problems in making a connection between product and process quality is that of measuring the quality of a process.

- How do you define process quality?
- How do you measure (evaluate) process quality?
- How was that process implemented to produce the quality of the product, once the product quality is defined? This is addressing two things: (1) was the right kind of process implemented to achieve the requisite quality?; and (2) was that process correctly implemented?

As an example, if an organization is appraised with the CMMI as a reference model and achieves a certain maturity level, is this rating sufficient to indicate that the process is a quality process or that the processes being appraised actually will produce quality products?

Hypothetically, all organizations rated at Maturity Level 3 should have an organizationally consistent process, but that doesn't mean that two organizations rated at Level 3 have a set of processes that will produce products having the identical quality. One organization's process may be significantly better than the other's in producing quality products, although according to their maturity rating, the processes are both organizationally consistent and are consistently applied throughout the organization (but allowing for project tailoring of the organization's set of standard processes to better address specific customer needs and expectations). This illustrates a lack of "connection"

between maturity levels and what constitutes product quality. Just using the words product quality and process quality without really specifying the connections and measurements is not appropriate and gives a false sense of security.

What are missing are the definitions and the connections. Maturity levels alone only represent a dimension of process quality without linking the process maturity to overall process quality and then to the quality if the product's produced by the process. That is,

> *Implementing and executing standard organizational processes does not guarantee "high-quality" products.*

However, by standardizing proven activities of the organization and by improving the linkage between process and product that begins at CMMI ML3 and continues at the higher levels, the expectation is that the ability to improve product quality and the ability to handle complexity and higher risk should become more evident—but again not guaranteed.

4 Summary

The basic concepts associated with product quality, or the quality of any entity, have been presented and lead to the idea of a framework for quality consisting of

- definitions and attendant concepts
- activities to:
  - establish requirements and control changes
  - establish methods to implement and achieve specified product quality
  - evaluate process and product quality

These three elements were explored showing not only interaction with each other but also with all other development project activities. The extremely complex interactions occur at many levels within the development project and throughout the project's life cycle.

Baker was the first to articulate the framework with the definitions we discussed here [Baker 1982a]. These concepts were subsequently extended in other works, for example, the "Software Quality Program Organization" chapter [Baker 2007] of *The Handbook of Software Quality Assurance* [Schulmeyer 2007].)

One driving factor in creating the original Software Quality Framework [Baker 1982a] was the necessity to clarify the confusion about software quality and software quality assurance. In addition, then, as now, there were widespread attempts to equate software quality with hardware quality, for example, software reliability, as well as attempts to force-fit software quality into hardware quality concepts and systems. The same confusion between quality and quality assurance exists for the development and maintenance of systems and products. The success of a project (and the foundation for its existence) is tied to achieving the requisite product quality. However, one must understand what product quality is and the technical aspects of specifying, developing, and evaluating it. Product quality results from employing appropriate processes and methods in the project design and implementation. Quality cannot be achieved by inspecting and testing the product.

A typical misconception that exists is that any action or activity including the word "quality" becomes the responsibility of a quality assurance (QA) group. It recognized that the quality of anything (product, process, services,), while influenced by many groups, has to be first specified and then built in. It cannot be assured, audited, or tested into the entity. Even so, today, there are areas of the acquisition and development communities that relegate any aspect that deals with quality to QA groups (in some cases these are referred to as quality management departments). Organizations are using the idea of quality management to sustain QA groups. Clearly, this is not the case with this Extended Framework nor in codification of these concepts in ISO standards and the CMMI.

Work on the original Software Quality Framework was initiated by DoD efforts to develop standards such as DoD-STD-2167 and DoD-STD-2168. The framework is recognized as applicable to

entities and disciplines other than software. And, as previously noted, many of the concepts and principles have found their way into commercial standards and process models, such as the CMMI, that have codified them in ways that can be implemented. While in this technical note the Framework concepts were mapped to the CMMI to show how these provide a first-level implementation of the principles, the overall concepts (especially those associated with the definition of quality and what it means in different disciplines) can be seen in many other efforts. For example, the idea of organizational approaches to implementing the framework, including associated planning and management aspects [Baker 07, Schulmeyer 07]. The principles put forth in the EQCF are sound principles upon which any process or quality model can be based.

The idea of specifying and measuring quality attributes of software products in contracts, such as assessing quality of software architectures, can be found in *Software Quality Management* [Cooper 1979] and *Use of the Architecture Tradeoff Analysis Method (ATAM) in Source Selection of Software-Intensive Systems* [Bergey 2002]. These documents make significant use of the concepts discussed here.

Although there is general belief that the use of a structured, consistent process will lead to better quality products, a number of researchers point out that the cause and effect relationships have not been established quantitatively. Thus, the relationship between quantitative measures of process quality and their relationship to product quality has not been clearly demonstrated. However, there is empirical evidence supporting our intuitions that there is a connection, i.e., a cause and effect relationship [Gibson 2006]. In addition, while we can acknowledge that there is currently no guarantee, work in this area address the situation of a "mature supplier" working with a "low maturity customer" in a way that supports process improvement and use of mature proven processes. A work in progress studies the so-called problematic programs (mature programs not performing well) and discovered the water coming from clean pipes is much more likely to be clean than is the case for dirty pipes (less mature organizations), even in dirty homes (immature customers); i.e., while there is no guarantee, odds of successful programs with quality products are improved if the processes are mature.

The use of appraisals against a process model such as the CMMI only measure one dimension of process quality in terms of organizational maturity levels or process area capability levels. But such appraisals while providing a maturity or capability level does not guarantee good products from the process. There is ongoing work attempting to provide a better way to link processes and associated practices and the resulting product quality.

References/Bibliography

[Baker 1982a]
Baker, Emanuel R. & Fisher, Matthew J. "A Software Quality Framework." *Concepts—The Journal of Defense Systems Acquisition Management 5*, 4 (Autumn 1982).

[Baker 1982b]
Baker, Emanuel R. and Fisher, Matthew J. "A Software Quality Framework," Fourth International Conference of the Israel Society for Quality Assurance, October 18-20, 1982, Herzliyah, Israel.

[Baker 2007]
Baker, Emanuel R. & Fisher, Matthew J. "Organizational Aspects of the Software Quality Program," in Schulmeyer, G. Gordon & Mcmanus, James I., eds. *The Handbook of Software Quality Assurance*, 4th ed. Norwood, MA: Artech House Publishers, 2007 (in publication).

[Bazzanna 1999]
Bazzana, Gulatiero, "Process and Product Measurement," in Messnarz, Richard and Tully, Colin, eds., *Better Software Practice for Business and Benefit: Principles and Experience*, Los Alamitos, CA: IEEE Computer Society Press, 1999.

[Bergey 2002]
Bergey, J.; Fisher, M.; & Jones, L. *Use of the Architecture Tradeoff Analysis Method (ATAM) in Source Selection of Software-Intensive Systems* (CMU/SEI-2002-TN-010, ADA403813). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2002.
http://www.sei.cmu.edu/publications/documents/02.reports/02tn010.html

[Chrissis 2006]
Chrissis, M. B.; Konrad, M; & Shrum, S. *CMMI: Guidelines for Process Integration and Product Improvement*, 2nd ed. New York: Addison-Wesley, 2006.

[Cooper 2002]
Cooper, J. & Fisher, M., eds. *Software Acquisition Capability Maturity Model (SA-CMM) Version 1.03* (CMU/SEI-2002-TR-010, ADA399794). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2002.
http://www.sei.cmu.edu/publications/documents/02.reports/02tr010.html

[Cooper 1974]
Cooper, J. & Fisher, M., eds. *Software Quality Management.* Princeton, NJ: Petrocelli Books Inc., 1979.

[Fenton 1991]
Fenton, N. E. *Software Metrics: A Rigorous Approach.* New York, NY: Chapman & Hall, 1991.

[Fenton 1995]
Fenton, Norman E. & Whitty, Robin. "Introduction," 1-19. *Software Quality Assurance and Measurement, A Worldwide Perspective*, Norman Fenton, Robin Whitty, & Yoshinori Iizuka, ed. London: International Thomson Computer Press, 1995.

[Firesmith 2006]
Firesmith, D. *QUASAR: A Method for Quality Assessment of Software-Intensive System Architectures*. (CMU/SEI-2006-HB-001, ADA455116). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, July 2006.
http://www.sei.cmu.edu/publications/documents/06.reports/06hb001.html

[Fowler 1990]

Fowler, Priscilla, & Rifkin, S. *Software Engineering Process Group Guide* (CMU/SEI-90-TR-024, ADA235784). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1990. http://www.sei.cmu.edu/publications/documents/90.reports/90.tr.024.html.

[Ghiselli 1981]
Ghiselli, Edwin E.; Campbell, John P.; & Zedeck, Sheldon. *Measurement Theory for the Behavioral Sciences.* San Francisco, Calif.: W. H. Freeman and Company, 1981.

[Gibson 2006]
Gibson, Diane L.; Goldenson, Dennis R.; & Kost, Keith. *Performance Results of CMMI-Based Process Improvement.* (CMU/SEI-2006-TR-004, ADA454687). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, July 2006.
http://www.sei.cmu.edu/publications/documents/06.reports/06tr004.html

[Humphrey 1989]
Humphrey, Watts S. *Managing the Software Process*. New York: Addison-Wesley, 1989.

[IEEE-STD-610]
IEEE Standard Glossary of Software Engineering Terminology.

[ISO 15939]
International Organization for Standardization. Software engineering – Software Measurement Process, 2002-07-15.

[ISO 2004]
International Organization for Standardization. *ISO 9000*. 2004.
http://www.iso.org/iso/en/iso9000-14000

[Kenett 1999]
Kenett, Ron S. and Baker, Emanuel R. *Software Process Quality: Management and Control*. New York: Marcel Dekker, 1999.

[Krantz 1971]
Krantz, David H.; Luce, R. Duncan; Suppes, Patrick; & Tversky, Amos. *Foundations of Measurement, Vol.1*. New York, NY: Academic Press, 1971.

[Melton 1996]
Melton, A. *Software Measurement.* New York, NY: International Thomson Computer Press, 1996.

[Pall 1987]
Pall, Gabriel A. *Quality Process Management.* Englewood Cliffs, N.J.: Prentice Hall, 1987.

[Paulk 1995]
Paulk,, Mark C.; Weber, Charles V.; Curtis, Bill; & Chrissis, Mary Beth. *The Capability Maturity Model: Guidelines for Improving the Software Process*. Reading, MA: Addison-Wesley Publishing Co., 1995.

[Schulmeyer 2007]
Schulmeyer, G. Gordon. "Software Quality Lessons from the Quality Experts," in Schulmeyer, G. Gordon, *The Handbook of Software Quality Assurance*, 4th ed. Norwood, MA: Artech House Publishers, 2007 (in publication)

[SEI 2007]
CMMI Product Team. *CMMI for Acquisition, Version 1.2* (CMU/SEI-2007-TR-017). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, November 2007. http://www.sei.cmu.edu/publications/documents/07.reports/07tr017.html

[Stevens 1959]
Stevens, S. S. "Measurement, Psychophysics, and Utility," 18–63. *Measurement: Definitions and Theories,* C. West Churchman & Philburn Ratoosh, ed. New York, NY: John Wiley & Sons Inc., 1959.

[Tully 1999]
Tully, Colin, Kujava, Pasi, and Messnarz, Richard, "Software Process Analysis and Improvement: A Catalogue and Comparison of Models," in Messnarz, Richard and Tully, Colin, eds., *Better Software Practice for Business and Benefit: Principles and Experience*, Los Alamitos, CA: IEEE Computer Society Press, 1999.

[Voas 1997]
Voas, Jeffrey, "Can Clean Pipes Produce Dirty Water?" *IEEE Software 14,* 4 (July/August 1997): 93-95

[Voas 2004]
Voas, Jeffrey, "Software's Secret Sauce: The –ilities," *IEEE Software 21,* 6 (November/December 2004): 14-15

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
(Leave Blank)	November 2007	Final

4. TITLE AND SUBTITLE	5. FUNDING NUMBERS
Basic Principles and Concepts for Achieving Quality	FA8721-05-C-0003

6. AUTHOR(S)
Emanuel R. Baker, Matthew J. Fisher, Wolfhart Goethert

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213	CMU/SEI-2007-TN-002

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER
ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2100	

11. SUPPLEMENTARY NOTES

12A DISTRIBUTION/AVAILABILITY STATEMENT	12B DISTRIBUTION CODE
Unclassified/Unlimited, DTIC, NTIS	

13. abstract (maximum 200 words)

This technical note extends the quality concepts first articulated in A Software Quality Framework (SQF) developed in the early 1980's for the Department of Defense (DoD) by Baker and colleagues [Baker 82a, Baker 82b]. The goals of the SQF were to state essential concepts and principles about quality, address fundamental misconceptions, and place quality in a proper perspective in relation to the acquisition and development of computer software. This earlier work led to the inclusion of a more "enlightened view" of quality within military standards such as DoD- STD-2167 and DoD- STD-2168. What makes the SQF work relevant today is the fact that many of today's organizations harbor similar misconceptions about quality. In addition, these same concepts and principles need to be addressed within today's process and quality models. As an example, we discuss how the quality concepts have been adopted and codified by the CMMI model (specifically, CMMI-DEV, V1.2).

In this technical note we extend the original quality concepts of the SQF beyond software to products, services, and processes. We address the conceptual elements necessary for building quality into systems, or any entity, and evaluating the quality actually achieved. For the definitions and conceptual elements we include brief discussions of the CMMI context. In this way we show how CMMI codifies the concepts. Another goal of including the CMMI context is to help CMMI implementers recognize the purpose of some of the CMMI components relative to the quality concepts and principles that they are addressing and help ensure a quality CMMI implementation leading to quality products. Definitions and attendant concepts constituting the structural members of our extended framework are provided, laying the foundation for the remainder of the framework consisting of additional framework areas or elements for establishing quality requirements, establishing methods to help satisfy the requirements, and quality evaluation.

14. SUBJECT TERMS	15. NUMBER OF PAGES
software quality, product quality, process quality, quality products, quality methods, measuring quality, quality requirements, evaluating product quality, evaluating process quality, quality evaluation, quality assurance	44

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL