APPLYING QUALITY COSTS IN A SOFTWARE DEVELOPMENT ENVIRONMENT

I.P. Hollingsworth+, W. Keogh*and M.H. Atkins%

+Department of Commerce,
University of Birmingham
B15 2TT
e-mail: i.p.hollingsworth@bham.ac.uk

*Aberdeen Business School
The Robert Gordon University
Viewfield Road
Aberdeen
AB15 7AW
e-mail: w.keogh@rgu.ac.uk

% (Correspondence author)
Department of Management Studies
University of Aberdeen
Old Aberdeen
AB24 3QY
e-mail: m.h.atkins@abdn.ac.uk

ABSTRACT

This paper shows how Quality Costs can be a measure of software quality. The relationship between Quality Costs and other software quality metrics is briefly explained, and software development oriented versions of the two principal Quality Cost models are described. Finally the paper discusses the major issues involved in setting up a software Quality Cost programme. The concepts are based on previous research on Quality Costs in manufacturing, coupled with work on software metrics and the work currently being undertaken by the authors in a number of industries.

Key Words: Quality Costs, software quality, quality metrics

AIM OF THE PAPER

This paper illustrates the potential use of Quality Costs as a measure of software quality, and outlines how this could be done. Based on ideas established in manufacturing, it presents the possibilities for Quality Costs to complement other measures of software quality. Quality Costs are not seen as better absolute measures of software quality than any other metric, nor is it the intention of the paper to establish a pecking order of software quality metrics.

The main concepts discussed are based on manufacturing's 40 or so years' experience of Quality Costs and other measures of quality which are combined with more recent work on software quality and metrics. Previous research has identified two basic models to identify Quality Costs and these are described and applied to a generic software development environment. The potential difficulties encountered in establishing a Quality Cost programme are also outlined, and ways to avoid them suggested.

INTRODUCTION

There is an established view of software development which sees it as an engineering discipline, in that it occurs in a planned and systematic manner, and with predictable results. One outcome of this is the increasing interest in planning and controlling software development through measurement. This trend is particularly noticeable in the field of software quality management.

This view that quality is measurable has been embodied in the International Standards Organisation's guide to quality systems for software development, ISO 9000-3 (BSI, 1991), which makes reference to the use of both product and process metrics as necessary elements of any quality system intended to meet the standard. This interest in the measurement of software quality has also lead to several major research projects within the UK including ESPRIT project EP-5425 PYRAMID (PYRAMID Consortium, 1991) and the Department of Trade and Industry's Quantum project (Praxis, 1992). The two main difficulties are deciding what 'software quality' actually means, and defining practical and meaningful ways by which to measure it. As ISO 9000-3 states, 'There are currently no universally accepted measures of software quality.' (clause 6.4 BSI, 1991b)

Introducing Quality Costs

First it is necessary to consider the nature of Quality Costs. Juran (1951) suggests that they could be identified by asking the question 'What present costs would disappear if all defects disappeared?' This seems to imply that the cost of quality is simply the cost incurred by making errors. Some might interpret this to mean that reducing Quality Costs would simply require more resources to be expended in removing errors. In the context of software development, the logical conclusion of this approach is to prove mathematically all software, taking into consideration all environmental factors. This is largely impractical, and makes this interpretation of Quality Costs of limited use in the context. However, the original question requires little modification to address this problem. If it is rephrased as 'What present costs would disappear if we *could be assured that* all defects *had* disappeared?', the costs of *error prevention* and *error identification* become part of the definition. In fact this definition is in line with the standard definition of quality related costs as given in BS4778 Part 2 (BSI, 1991a), namely the 'Cost in ensuring and assuring quality as well as loss incurred when quality is not achieved'.

Quality Cost Models

Two very different approaches to the modelling of Quality Costs have been developed over the years, and are outlined in BS6143, Parts 1 (BSI, 1992) and 2 (BSI, 1990). This section of the paper describes the Prevention, Appraisal & Failure (PAF) and Process models. The PAF model is restricted to consideration of quality-related costs while the Process model also considers the actual cost of the basic processes themselves.
It is important that Quality Costs are distinguished from the process costs which would normally be anticipated, for example, the cost of a planned level of post-implementation support for users. The other test for a Quality Cost, defined by Juran (1951) is that it should be an avoidable cost, based on the previously mentioned idea of being assured that no defects exist in the product. Taking the example of post-implementation support, even if the product is defect-free and meets user requirements, users are still likely to need some support for the new product. Similarly with maintenance work on the product, even if the software is defect-free and meets initial requirements, and thus requires no corrective maintenance, adaptive maintenance is still likely and therefore should not be regarded as a Quality Cost.

The Prevention Appraisal & Failure (PAF) Model.

The PAF model was developed through work done in the 1940s and 1950s (Juran, 1951; Feigenbaum, 1956), and is used as the basis for the British Standard (BSI, 1990). The thinking behind this model can be summarised neatly by two statements from the Standard:

> "failures, however caused, reduce profits"
> and
> "preventative quality control activities and the appraisal of quality standards cost money to operate".

Essentially the PAF model breaks Quality Costs down into three main categories:

Prevention costs:	which arise in the course of preventing or reducing the risk of non-conformities or defects in the product;
Appraisal costs:	which are associated with the cost of checking conformance of the product to quality requirements;
Failure costs:	which arise as a result of identified non-conformities or defects in the product.

However, failure costs can be further sub-divided into internal and external failures:

Internal Failure costs:	which arise due to non-conformities or defects identified at any stage of the development process;
External Failure costs:	which arise due to non-conformities or defects identified at user acceptance testing or after delivery to the customer/user.

This gives a total of four categories of costs to consider. Hollocker (1989) also suggested that Prevention costs should be sub-divided into two separate categories (Prevention and Control). However, Hollocker's Prevention costs can be regarded as being beyond the scope of individual development projects. They are associated with non-routine activities such as training. On the other hand, control costs are associated with routine project

control issues such as project and configuration management. What separates the two categories is this idea of routinisation. Hollocker argues that if an activity is not regarded as a normal part of project control then its quality-related costs should be regarded as Prevention costs. While the addition of this extra category (i.e. Control) assists in identifying the impact activities such as training have on routine control costs, it does exacerbate the problem of categorising costs. Control issues are "apparently the hardest to manage", and so are worthy of being singled out, though adding the caveat that if a control function is not provided by "department charter" it should be placed in the Prevention category. However, the addition of an extra category exacerbates the problem of categorising costs. We have used the original PAF categories to illustrate the model, though some organisations might find that the extra category better serves their needs.

The limitations of the PAF model have been outlined by Porter and Rayner (1992). They identified difficulties and problem areas associated the application of the model such as the recording of quality cost information in a comprehensive programme; difficulties in deciding the accuracy levels which the organisation may require; categorising and placing the costs within the working model; and, of course, the difficulties of obtaining the basic data necessary for reporting the costs - which Feigenbaum has addressed from as early as the 1950s (Feigenbaum, 1956). Further criticisms of the PAF model are that it focuses on cost reduction rather than the positive impact of quality on sales revenue; that all successful applications of the model had hitherto been in large firms and, most importantly (they suggest), that it "implies an acceptable quality level as a result of the trade off between prevention and appraisal costs and failure costs." This, it is alleged, is incompatible with the continuous improvement philosophy of Total Quality Management.

Another aspect to bear in mind is that failure costs should include the cost of any additional appraisal or prevention activities necessary to fix an error, such as re-testing or any extra configuration management effort required. As a result of this it is necessary to take care, when collecting data, that a single cost is only collected under one category. Accordingly, care should be taken when dealing with the knock-on effects of errors which require new program versions to be developed and controlled, including their configuration management.

The Process Model of Quality costs

The second approach to Quality Costs is to link all costs to their associated process rather than split all activities into separate cost categories as in the PAF model . Within each process the quality-related costs are then split into costs of conformance or non-conformance:

> **Cost of Conformance** - the cost of achieving the required standard or specification for the product or service supplied by the organization,
>
> and
>
> **Cost of Non-Conformance** - the costs arising where specifications are not being met and failures occur at various points in the supply cycle.

These correspond approximately to the PAF categories, in that failure costs are equivalent to cost of non-conformance, and prevention and appraisal are equivalent to cost of conformance. Accordingly the final Quality Costs measured by the two models will not differ significantly but one important difference is that in the Process model normal process costs are also regarded as a cost of conformance (BSI, 1992; Porter and Rayner, 1992)

A drawback of the Process model is that it requires a well-defined, stable process. If the process is not clearly understood it is difficult to establish relevant costs. Secondly, if the process is not stable for some reason, such as changes in personnel, organisational structure, or the processes themselves, then the Quality Cost model may require extensive modification. This close link to the development process itself can also make it difficult effectively to fit in the costs of activities covering a wide area of the organisation, such as setting up a quality management system or TQM programme. A useful discussion on models including the use of process models in quality costs which can be used as a measure to improve effectiveness as well as efficiency of quality activities is given in Hwang and Aspinall (1996).

Hidden Quality Costs

Quality Costs can include costs which arise out of services provided beyond the bounds of the organisation, in either its suppliers or its customers. These may be reflected in the price paid for products/services and in the other costs incurred by the organisation such as rectification or downtime. These are Hidden Quality Costs in the sense that they are not measured by the organisation. Winchell (1987) has suggested that they can be divided into three parts:

- those that are incurred by the supplier within the organisation, and passed on to the buyer in terms of price;
- those that are incurred by the customer organisation in solving problems for the supplier, such as identifying faults in development tools;
- those that are not allocated to suppliers, but are incurred by the customer organisation as a result of potential or actual supplier problems; this would include the costs related to unexpected system behaviour caused by bought-in software, or extensive incoming acceptance-testing of software tools made necessary to assure their quality.

QUALITY COSTS IN MANUFACTURING

The importance of Quality Costs was first recognised in the context of Manufacturing Industry and there is an established body of research undertaken and evidence available. In the UK, various studies including 'Standards in Specifications in the Engineering Industry' (Warner, 1977), indicated that quality costs comprise very large proportions of manufacturing costs. It was estimated that the cost of achieving or not achieving the required quality standard of goods produced by UK establishments cost the UK approximately 10% of the £105,000 million total sales of goods in 1976, or £10 billion. These figures were reported in "A National Strategy for Quality", a consultative document produced by the UK Government (DPCP, 1978), and originated from the 1976 Census of Production. Quality costs were taken as the costs of defective quality together with the costs of attaining quality, and were estimated as lying in the range of 4 to 15 per cent of turnover.

The importance of addressing quality costs was further emphasised in 1985 in the National Economic Development Council paper "Quality and Value for Money" (NEDC 1985), which not only discussed the benefits of identifying and dealing with quality costs, but made a firm recommendation that "Accountants and quality specialists in industry should examine the scope for developing management information systems to identify quality costs (based on BS 6143), with the objective of reducing these costs." The paper estimated that between 10 and 20 percent of sales value could be accounted for by quality related costs: taking the 10 per cent figure meant that there was a potential cost of £6 billion in manufacturing industry alone (1985 data). It has also been indicated from other, more recent, studies that Quality Costs can be as high as 25% of sales turnover (Dale and Plunkett,1992 and 1995). This compares with the figure of 30% for software Quality Costs in the U.S., quoted by A. Smith of Coopers & Lybrand Deloitte (Smith, 1991).

The background to the design and use of Quality Cost Programmes lies firmly in manufacturing. An early example from H.G. Crockett (1935) outlines how a hosiery mill reduced quality costs by improving workmanship in all operations and, not only identified key cost saving areas, but ultimately resulted in the reduction of the number of quality control inspectors from 25 to five. Joseph Juran outlined key areas (including elements of quality costs such as scrap and rework) dealing with management problems in the field of inspection and quality control based on his experiences as Chief of Inspection, Control Division, in the Western Electric Company (Juran, 1945).

As previously mentioned, the PAF model has been used in one form or another since the 1950s (Juran, 1951; Feigenbaum, 1956; Masser, 1957). In 'The Quality Manager and Quality Costs' (1957), Masser makes it clear that there are four major results which can be obtained from accurate quality costs:

1. a tool is provided for **measurement** of overall business quality performance
2. quality costs are an **analysis** tool which can be used to indicate where quality money is being spent
3. a tool is provided for **programming** the when, the where, and how of quality improvements
4. a fine **budget** tool can be made which enables the forecasting of realistic needs.

The American Society for Quality Control (ASQC) has issued material to assist with quality costs based on the very early work. In 1967, the Quality Costs Committee of the ASQC issued 'Quality Costs: What and How' which outlined ways of achieving management commitment as well as establishing a programme. A key area

which was addressed in this publication, and its subsequent revisions in 1971 and 1974, was that quality costs had to be sold to senior management: the bottom line of potential dollar savings was the key to gain management commitment. Juran's 'gold in the mine' analogy (Juran, 1951) can even today bring it home in a very simple and easily understandable manner to senior executives.

The aim of a quality cost improvement programme is to shift the costs from the failure category to prevention, thus leading to cost savings on overheads as well as expense items. For example, if fewer customer complaints and warranty claims are received, then there will be less administration required and subsequently lower costs. As a basis for quality improvement, a quality cost system can play a significant role. Potentially, a quality cost system, can become an important tool to be used to assist the management of the organisation (Campanella, 1990), for example by providing a tool for facilitating quality programmes, and quality improvement activities. However, as Keogh and others (Keogh, 1996) have shown, even in advanced industries, with a well educated work-force and a great deal of data available, problems of definition and categorisation can still occur in developing quality cost systems.

Therefore, by utilising quality costs properly, not only can cost reductions be made but the competitive position of the organisation can be strengthened, and these can be combined to improve profitability. Masser's work was carried out in the General Electric Company, New York, and the results and outcomes are just as relevant today as they were in the 1950s.

The 'Guide for Reducing Quality Costs' was published in 1977 by the ASQC. The task group comprised members from different backgrounds and organisations, including General Electric company, Digital Equipment Corp., and International Telephone and Telegraph Corporation. As well as dealing with the quality cost improvement philosophy, setting up a system was addressed and ways of measuring improvements were examined. Supplier quality costs were also examined by the ASQC and, in 1980, the "Guide for Managing supplier Quality Costs" was published to aid both purchasers and suppliers by suggesting methods by which supplier quality costs could be managed, with the second edition following in 1987 (Winchell, 1987).

Crosby (1979) argues that "Quality is Free" and what this means to practitioners is that the costs of achieving quality are offset by the savings in quality improvements. Regardless of whether the organisation is in the manufacturing, service, or public sectors, by analysing processes, involving people, and making improvements, then problems can be addressed and savings made (BSI, 1992).

QUALITY COSTS IN SOFTWARE DEVELOPMENT

We have already noted that evidence (Smith, 1991) suggests that in the US software Quality Costs may reach 30 per cent of turnover. Given this, possibly alarmist, figure, it is appropriate now to apply the previous analysis to the software development process.

Measuring Software Quality

There are many aspects of the software development process that can be measured (Fenton, 1991) including cost, size, complexity and quality. Irrespective of which of these aspects of software development is being measured there are several issues related to how measurement can be applied. Fenton (op. cit.) distinguishes between *measuring for prediction,* for example, estimating project costs during planning, and *measuring for assessment*, for example, establishing the actual cost of the project. Fenton and Ashley (1993) make a further distinction between:

resource measures	- which relate to the resources used as inputs to the development process such as cost or effort,
process measures -	- which relate to the actual software development process itself such as the potential for trapping/avoiding errors,
product measures -	- which relate to the deliverables produced as output by the development process.

A third aspect is the distinction between internal and external attributes of a resource/process/product. *Internal attributes* are those which can be measured objectively in terms of the resource/process/product itself, and are generally directly measurable. *External attributes* are those which can only be measured in situ in terms of how the resource/process/product relates to the environment in which it is to be used. Fenton (op.cit.) argues that the latter cannot be measured directly, and therefore require some form of substitute measure.

These three characteristics of software metrics need to be borne in mind whatever aspect of the software development process is being considered and not just in the case of software quality. As far as the customer is concerned, it is arguable that the *product measures* of software quality are paramount as these immediately relate

to the usefulness (or otherwise!) of the product or system in question. In the longer run, however, both *resource* and *process measures* of software quality will reflect on the individual supplier's efficiency and effectiveness.

In terms of the characteristics of measures given above, Quality Costs have a well-identified position. They are primarily an assessment metric, aimed at measuring process quality. Measurement is done indirectly, by measuring the cost of achieving a certain specified level of quality, rather than trying to measure the quality of the product directly, for example, the number of errors per thousand lines of code.

It should be clear from this that Quality Costs are not suitable as product metrics. The information that a piece of software has a Quality Cost of £x, or that Quality Costs were a given proportion of its total development costs, tells nothing about the actual quality of the software product itself. Accordingly, in order to establish that a piece of software has achieved a specified level of quality, it is necessary to use more well established software product metrics. Quality Costs would then be a measure of the effectiveness of the techniques and tools used to achieve that quality; in other words, the development process itself.

Another characteristic of a metric is the scope of the activity covered by it (PYRAMID Consortium, 1991). By their very nature Quality Costs can be used as a global measure (or indicator) of the effectiveness of the development process, or as a phase specific metric focused on one particular area. Indeed, the potential scope of a Quality Cost programme is so broad that it could be regarded as a form of *business* metric, by reflecting the effect quality has on the commercial success of the organisation as a whole, not just the development process.

The PAF Model

Applying the PAF model to software engineering gives examples of costs that may be included in each category (shown in Table 1). These are not intended to be definitive but to act as a guide to likely costs. It may be possible to challenge some of the categorisations such as implementation costs, but this reflects the difficulty which can arise with the categorisation process. The important thing is that the categorisation should be consistent within the Quality Cost programme and suit the needs of the organisation.

The Process Model

Table 2 identifies possible costs when applying the Process model of Quality Costs to the Software Development Process. This shows the generic sources of cost which can be applied to all the processes in the software development life cycle. The obvious advantage of this approach is that it is consistent with the process view of system development seen in project planning and control, making allocation of costs much easier than with the PAF model. This table has been based upon a simplified top-level process model for software development using the IDEF process-based system modelling technology.[21]

[21]Part 1 of the British Standard (BSI,1992) is based on the TQM process modelling approach using a variation on the Structured Analysis and Design Technique developed by SofTech Inc. in the early 1970s (Marca and McGowan, 1987). SADT activity modelling was subsequently revised by SofTech Inc. and published as IDEF0, as part of the US Air Force's ICAM programme (Colquhoun, 1993).

IDEF0 is a top-down, hierarchical process-based system modelling methodology originally designed for modelling the processes involved in aircraft manufacture. The basic notation is quite simple, with boxes showing activities (or processes) and arrows showing the different data constraints on those activities. These constraints are split into controls, inputs, outputs and mechanisms. Controls represent inputs which determine what the activity is to do, for example "test plan" which would determine what testing would take place. Mechanisms represent the resources which can be applied to the activity in order to perform it, i.e. how the activity is actually physically done (SofTech Inc., 1980).

Table 1.
Possible Quality Costs for PAF model of software development.

Prevention Costs

Cost of defining standards, e.g. coding, documentation, testing
Cost of initial project planning activities
Cost of progress monitoring and control
Cost of configuration management activities
Cost of quality-related[1] training, both generic and project specific
Cost of quality improvement programmes, including data collection[2]

Appraisal Costs

Cost of design reviews
Cost of system modelling, including prototypes[3]
Cost of test planning, execution, and documentation up to and including acceptance
Cost of checking delivery media
Cost of managing/maintaining test environments/tools/equipment
Cost of documentation reviews

Internal Failure Costs

Cost of problem identification during design and development
Cost of problem reporting and initiation of corrective action
Cost of rework of specifications/design
Cost of code rewrites
Cost of necessary rework of previous development phases, e.g. requirements analysis[4]
Cost of reworking project plans
Cost of appraisal of rework
Cost of additional administration due to rework, e.g. configuration management
Opportunity costs, e.g. losses due to delays in implementation[5]
Cost of development system down-time/performance problems

External Failure Costs

Excessive implementation costs
Unplanned customer support costs
Costs of post-implementation problem identification
Costs of problem reporting and related change control
Costs of corrective maintenance, including field updates
Liability, warranty and concession costs
Intangible costs, e.g. loss of goodwill

Notes to Table One

1 *This includes any training which will have an impact on the quality of the finished software product, e.g. training in specific methodologies, technologies, or quality assurance procedures.*
2 *This includes the cost of the Quality Cost programme itself.*
3 *Assuming that the prototype is not intended to be part of the final delivered product, but is only to be used a model with which to test requirements or design.*
4 *If during subsequent development stages it is found necessary to re-analyse part of the problem, possibly as the result of errors found at acceptance testing, then this becomes a Quality Cost.*
5 *This would be the case for a software house or similar supplier of software. For in-house IT departments such delays may be regarded as an external failure, as the business as a whole (their customer) will suffer the loss of opportunity.*

Table 2.
Possible Quality Costs for Process model of software development.

For all processes:

Costs of Conformance:

Cost of designing/defining the process as part of any wider initiative
Cost of carrying out the process without error, including all planning and checking internal to the process

Costs of Non-conformance:

Cost of problem identification
Cost of problem reporting and initiation of corrective action
Cost of in-process reworkCost of rework caused by this process but carried out in another
Cost of appraisal of rework
Cost of additional administration due to rework, e.g. configuration management
Cost of unavailability of necessary equipment or resources e.g. system downtime.
Opportunity costs, e.g. losses due to knock-on effect of delays in this process

QUALITY COST PROGRAMMES

There are several potential applications for Quality Cost programmes in software development environments. The scope of application can obviously be varied to suit the individual needs of an organisation, but it is useful to think on three levels:

- department/company-wide programmes which encompass more than just the development process;
- development process programmes which encompass the whole of the software development process;
- phase-specific programmes which are focused on specific phases of the software development life cycle.

Broad scope programmes are aimed at overall organisational improvement, with the software development process as part of a wider programme. This approach is particularly relevant to organisations such as software houses and consultancies, where software development is a significant part of their business.
Narrower (project) scope programmes are geared more towards development process improvement, rather than organisational improvement. There are a number of ways in which Quality Costs can be applied at this level. The first is process improvement, using Quality Costs to analyse the performance of development projects over a period of time. If development is carried out to a consistent methodology, then by monitoring the level of Quality Costs for each part of the methodology the programme can highlight possible weaknesses in the methodology which can then be targeted for improvement. If, on the other hand, development is not carried out in a consistent way, by comparing the levels of Quality Costs across projects the programme can be used to identify which methods are most effective. One benefit of Quality Costs in this context is that by comparing Quality Costs as a proportion of total costs, it is possible to allow for the effects of different development technologies. So rather than trying to establish if one defect per thousand lines of COBOL is the equivalent of one defect per form of a 4GL in order to make a reasonable comparison of quality, the issue would be which approach has higher relative quality-related costs.
The second potential application is as a means of justifying investment in new techniques or technology, such as the use of automated testing tools or the developing of a software quality management system, by estimating or measuring the effect they have on the level of Quality Costs.

Setting up a Quality Cost programme

There are a number of issues to consider concerning the setting up of a Quality Cost programme. These are outlined in BS 6143 Part 2 (BSI,1990) and in Feigenbaum's 'Total Quality Control' (1991). Further guidance is given by Evans and Lindsay (1993), who reproduce a summary of the twelve steps recommended by the U.S. National Association of Accountants (now the Institute of Management Accountants). Most guidance material is based on the PAF model, as this has the longer history of application. However, many of the points are equally applicable to the use of the Process model.

The following points are based on the guidance given in the Quality Costing literature. These are also consistent with the guidance given for establishing other types of software metrics programmes, as set out by the PYRAMID project (PYRAMID Consortium,1991), Fenton (1991) and the work of AMI (Application of Metrics in Industry). However, it is most important to recognise the second dimension that a software Quality Costs programme introduces. In addition to the technical aspects of quality with which practitioners will be familiar, any Quality Costs programme introduces financial and commercial aspects with which practitioners may well be much less familiar. Accordingly the principal differences between a Quality Costs programme and a conventional software metrics programme arise out of the wider scope of those involved; for example, accountants are likely to be included where they would not be in a conventional software metrics programme.

Setting objectives & scope

As with any management initiative, it is important to establish the objective(s) of a Quality Costs programme before it starts. This helps to ensure that the programme will actually provide meaningful information to support the decisions the organisation wants to make. There is a wide variety of potential reasons for setting up a Quality Costs programme including

- identifying high cost problems;
- setting cost reduction targets, and measuring progress against them;
- justifying the implementation of a specific quality management/assurance measure;
- measuring the performance of projects;

or some combination of these, either as a general business parameter, or as part of a specific quality improvement programme. No matter what the other requirements of an organisation may be, one objective that should always be included if the Quality Cost programme is to succeed is that it should provide a basis for action.

The measurement of Quality Costs should always be used as *a basis for action.* In other words, any Quality Costs programme is likely to have the dual purposes of measurement of costs (providing information) *and* the identification of means of reducing those costs. After identifying which Quality Costs are most important, it is essential that their value is measured. This will permit cost reduction targets to be set and progress against them measured. There is currently a lack of information available here and issues are being reassessed and readdressed although Pursgrove and Dale (1996) offer an example which, whilst dealing with PVC coating, provides many parallels with other organisations.

Linked to the establishment of *objectives* is the need to clarify the *scope* of the programme in terms of what parts of the organisation's activities are to be covered. It is important that this is defined clearly in order to avoid relevant costs being omitted, or irrelevant costs being included. It is also a useful check on the objectives, which should be capable of being met within the scope; for example, if the objectives include being able to compare project teams but the scope only includes one team, the whole exercise becomes a waste of time and effort. The narrower the scope, the more likely it is that costs will simply be moved outside the scope by any 'improvement' programme, rather than actually being reduced. The wider the scope, the more difficult it is to hide costs. On the other hand, if the programme attempts too much, it actually becomes very difficult to achieve anything.

Justifying the implementation of a specific quality management or quality assurance measure can, in itself, lead to the identification of other areas which should be investigated. Useful and meaningful results from measuring the performance of projects, products or services really only occur when the measurement parameters have been carefully chosen, methodically monitored and analysed.

Defining reporting procedures

A number of difficulties arise in reporting Quality Costs and perhaps the key issue is "what" to report to the users of the information. This will depend to a large extent on the reasons for establishing the programme, as these dictate the target audience for Quality Cost data. There are a number of issues which must be borne in mind irrespective of how the data is used, the most significant of which include:

- the need for regular reports as part of overall management reporting;
- the need to keep Quality Costs in context;
- the way to report Quality Costs as part of the accounts for a project, or business.

It is useful that reports detailing Quality Costs are presented in a similar manner to other management accounts and the findings should be supported using ratios and/or trend analysis. These reports can then be used within the

context of overall cost reports. To facilitate trend analysis a number of comparative bases can be used to represent the changes occurring over time. Examples include:

- labour base such as internal failure costs related to total labour or effort
- cost base such as total failure costs related to total development costs
- sales base such as total quality cost related to net sales turnover

These are based on the use of Quality Costs over a broad corporate scope. The comparative bases when the scope is project or phase specific are more likely to be those of total project development cost or effort.
The need for regular reporting also presents problems in situations where the processes are not continuous and this is likely to be the case in software development. Quality Cost reporting can be integrated with project control reporting within each project, but at a corporate level the issues become more difficult, as the characteristics of low quality may be different for different phases of the development life cycle.
The other important decision is the *reporting format.* This will obviously depend on the use to which the information will be put. Some common formats from manufacturing include:

- by PAF category;
- as a Process Cost Report;
- as a Quality Cost balancing tool for the QA/QM department;
- by management level, based on responsibility;
- by key non-conformances.

Hollocker (1989) suggests that, for software, reporting by project and by calendar period are suitable additional formats.
Given that a Quality Cost programme should be linked to improving quality, it is helpful if costs are tabulated in such a way that they provide information to the users and provide useful 'benchmarks' for the continuous improvement effort. If areas of management concern relating to business performance are to be monitored using the Quality Costs programme, critical areas must be identified, cost comparisons made and costs ranked (usually in order of importance). Therefore, the reporting format should facilitate this. BS 6143 Part 2 (BSI,1990) includes examples of report forms and quality cost/accounting ratios.

Collecting Quality Cost data

Cost data varies from company to company. The required data may be not be readily available, or it may not be in the required form. The first question has to be *what* to collect? Just as with other metrics, it is important not to get bogged down in collecting too many different Quality Cost elements to start with. It is more important to collect data on a small number of the most significant relevant cost areas, on a regular and consistent basis. The more focused the objectives of the programme, the easier this will be. Once the initial programme is established, the Quality Costs can be refined by adding more elements, or splitting existing cost elements to give more detail.
When establishing what to collect, there are a number of steps which can be taken. Step one calculates the costs directly attributable to the 'quality' function, such as the Quality Manager's salary. Step two identifies costs that are not directly the responsibility of the 'quality' function but which should be counted as part of the total quality-related cost of the organisation; these include the cost of testing. In terms of the PAF model, these two steps cover the identification of *prevention* and *appraisal* costs. Step three identifies and enters into a memorandum account the internal costs of 'budgeted failures', such as Brooks' "Plan to throw one away" (1975). Step four identifies the internal costs of failures not allowed for in step three, such as all reworking. Step five identifies the cost of failures after change of ownership; in the software development context these would include field fixes. In terms of the PAF model, these last three steps cover *failure* costs, both *internal and external.*
Once it has been decided *what* costs are to be collected, it is necessary to establish *who* will collect them. This will depend very much on an organisation's own preferences. It may be left to a centralised unit, possibly the accounts department, or it may become the remit of the systems or project managers, as part of their project monitoring activities.
Finally, it is necessary to decide *how* to collect the Quality Costs. Although sources vary from company to company, most organizations have some sort of source documents which could be used to provide information. Examples include payroll analysis, project progress reports, rework or rectification authorizations or reports, field failure reports and replacement and warranty costs reports. In addition, activities may already be categorised and/or costed as part of other software metrics programmes, and this information will be documented somewhere. The two most obvious mechanisms for collecting Quality Costs are existing software quality metrics programmes, and timesheets, which then feed into a costing or monitoring system, where they can be combined with other costs such as capital expenditure. Obviously, the needs and requirements of the collection system

will vary with the company, and with the development methods used on individual projects, but there are some general issues.

Taking software metrics first, there can be a number of drawbacks to using these for Quality Cost information. These tend not to include measures of appraisal and prevention activities, and they tend to be simple counts of, for example, errors per thousand lines of code, not of the consequences of those errors. Thus, it is likely that even with an existing metrics programme, it will not provide data suitable for use as Quality Cost data. However, it will provide information on the numbers of errors, and thus provide a basis for following up the cost of correcting these errors.

On the other hand, timesheets could be invaluable, as all that is required is some means of splitting out Quality Cost elements from normal process costs. This will then provide data on the most significant cost item - labour. The main difficulty with timesheets is accuracy, and this would need to be addressed, depending on the level of accuracy required. It is dangerous, and unnecessary, to strive for absolute accuracy as this tends to lead to very bureaucratic time recording which is of doubtful veracity. Most inaccuracy is unintentional, but if there are concerns that timesheets may be deliberately completed inaccurately to hide Quality Costs, these can be cross checked against results of reviews and tests on a sampling basis. For example, if a test is failed there should be some visible effort expended in fixing the problem.

An important point is that identifying and monitoring appropriate quality-related costs, and the collecting and classifying of these costs should be consistent with accounting/costing practices within the organisation. The people responsible for analysing the costs should be given any necessary assistance or input from others (such as technical specialists), to assist with the cost allocation exercise. However, care is required when it comes to using the costing system which is fed by such data. Generally speaking, it is a fallacy that it is easy to extract information from an existing costing system, as accounting systems are not set up to give Quality Cost information. If a costing system is already in place, it is likely to be concerned only with total project phase costs, with normal process costs and Quality Costs lumped together. Therefore, it may be advisable to avoid the temptation to modify existing costing systems to extract Quality Cost data, as it is often easier to start afresh.

CONCLUSION

This paper has shown that Quality Costs can provide an important second dimension to the measurement of quality in the software development process. It presents a picture of where Quality Costs fit into the overall area of measuring software quality, and highlights the major issues concerned with trying to put them into practice. Given the relative lack of literature on Quality Costs in software development, especially empirical work, it is difficult to be precise about how the experience of manufacturing will translate into software development. However, given that the general trend in software development is towards an engineering view of the processes involved, and the fact that previous moves towards improving the management of software development (such as project planning and control) have come from the same area, it is likely that Quality Costing will become a serious issue particularly for the larger software houses in the near future.

REFERENCES

Advisory Council for Applied Research and Development (1982) **Command Paper 8621: Facing International Competition.** HMSO, London

Ashley, N. (1993) Measurement as a Management Tool in: **Management and Measurement of Software Quality** M. Kelly (ed.) pp31-49, Avebury Technical, Aldershot, Hants.

ASQC (1967) **Quality Costs - What and How** ASQC, Milwaukee, Wisconsin

ASQC (1977) **Guide for Reducing Quality Costs** ASQC, Milwaukee, Wisconsin

British Standards Institution (1987) **BS 4778: Part 1. ISO 8402. Quality concepts and related definitions.** BSI, Milton Keynes

British Standards Institution (1990) **BS 6143: Part 2. Guide to the economics of quality - Prevention, appraisal and failure model.** BSI, Milton Keynes

British Standards Institution (1991) **BS 4778: Part 2. Quality concepts and related definitions.** BSI, Milton Keynes

British Standards Institution (1991) **BS 5750:Part 13: ISO 9000-3. Guide to the application of BS5750:Part 1 to the development, supply and maintenance of software.** BSI, Milton Keynes

British Standards Institution (1992) **BS 6143: Part 1. Guide to the economics of quality - Process cost model.** BSI, Milton Keynes

Brooks, F.P. jr. (1975) **The mythical man-month** Addison-Wesley, Reading, MA

Campanella Jack, ed. (1990) **Principles of quality costs, second edition.** ASQC Quality Press, Milwaukee, Wisconsin

Colquhoun G L, Baines R W and Crossley R (1993) "A state of the art review of IDEF0", **International Journal of Computer Integrated Manufacturing,** 6(4), 252-264

Cooper, Robin and Kaplan, Robert S. (1991) **The design of cost management systems.** Prentice-Hall International, New Jersey

Crockett, H G (1935) "Quality, but just enough", **Factory Management and Maintenance,** 93(6) 245-246

Crosby, Philip B. (1979) **Quality is free.** McGraw-Hill, New York

Dale, Barrie G. and Plunkett, James J. (1991) **Quality costing** Chapman & Hall, London.

Dale, Barrie G. and Plunkett, James J. (1992) **The case for costing quality.** UK Department of Trade and Industry, London

Dale, Barrie G. and Plunkett, James J. (1995) **Quality costing 2nd Edition,** Chapman & Hall, London.

DPCP (1978) **A national strategy for quality**, Department of Prices and Consumer Protection, London

Evans, James R. and Lindsay, William M. (1993) **The Management and Control of Quality, second edition.** West

Feigenbaum, Armand V. (1956) "Total quality control" **Harvard Business Review** 34(6) 93-101

Feigenbaum, Armand V. (1991) **Total quality control: 3rd edition revised.** McGraw-Hill, New York

Fenton, N.E. (1991) **Software Metrics. A Rigorous Approach** Chapman & Hall, London

Hollocker, Charles P (1989) "Finding the cost of software quality." In: **Quality Costs: Ideas & Applications, Volume 2. A Collection of Papers.** Campanella, J. (ed), pp 355-367, ASQC Quality Press, Milwaukee, Wisconsin

Hwang and E.M. Aspinall "Quality cost models and their applications: a review." **Total Quality Management** Vol 7 No3 June 1996 pp267-282

Johnson, Thomas H. and Kaplan, Robert S. (1987) **Relevance Lost: The rise and fall of management accounting.** Harvard Business School Press, Boston, MA

Juran, J.M. (1945) **Management of inspection and quality control.** Harper & Brothers Publishers, New York

Juran, J.M. (1951) **Quality Control Handbook** McGraw-Hill, New York

Juran, J.M. and Gryna, Frank M (eds) (1988) **Juran's quality control handbook, fourth edition.** McGraw-Hill, New York

Keogh, W. (1994) "Total quality management - the role of the professional in determining quality costs." **Managerial Auditing Journal** 9(4) pp 23-32

W.Keogh, P.Brown and S.McGoldrick "A pilot study of quality costs at Sun Microsystems." **Total Quality Management** Vol 7 No1 Jan 1996 pp29-38

Marca D A and McGowan C L (1987) **SADT Structured Analysis and Design Technique** McGraw-Hill, New York

Masser, W J (1957) "The quality manager and quality costs." **Industrial Quality Control** 14(6) 5-7.

Morse, Wayne J. (1993) "A handle on quality costs." **CMA Magazine,** Feb. 21-24.NEDC (1985) **Quality and value for money.** National Economic Development Council, London

Porter, Leslie J. and Rayner, Paul (1992) "Quality costing for total quality management." **International Journal of Production Economics** 27 69-81.

Praxis Study Team (1992) **QUANTUM. A measurement-based framework for the assurance of software quality** Department of Trade and Industry, London

Pursgrove and B.G. Dale "The influence of management information and quality management systems on the development of quality costing." **Total Quality Management** Vol 7 No4 Aug 1996 pp421-432

PYRAMID Consortium (1991) **Quantitative management: get a grip on software!** PYRAMID Consortium

Smith, A. (1991) "Quality - an integrated approach" In: **Software Quality and Reliability. Tools and Methods,** Ince, D. (ed.) pp 16-23 Chapman & Hall, London

SofTech Inc. (1980) **Architect's Manual. ICAM Definition Method "IDEF0" DR-80-ATPC-01** Computer Aided Manufacturing-International Inc., Arlington, Texas

Warner, Sir Frederick (1977) **Standards in specifications in the engineering industry.** NEDO, London

Winchell, William O. (ed) (1987) **Guide for managing supplier quality costs.** ASQC Quality Press, Milwaukee, Wisconsin