Transactions on Information and Communications Technologies vol 11, © 1995 WIT Press, www.witpress.com, ISSN 1743-3517

Implementing a quality management system using an incremental approach

S.A. Frangos
INTRASOFT, Adrianiou 2, 11525, Athens, Greece

Abstract

This paper analyses the steps which an organization may undertake so that it develops and maintains a QMS. An analysis of the various problems and issues which the author encountered while designing and implementing the QMS is presented as well as the experiences gained during various third-party certifications. The software development paradigm can be employed for the set-up of a QMS and such an approach is elaborated in this paper.

The set-up of any QMS requires activities to be undertaken in the planning, analysis, design, implementation and testing phases. During planning, the specific organizational matters are resolved, the quality policy is established and the scope of the QMS is clearly defined. An analysis of the business position of the organization with respect to quality as well as the establishment of clear and quantified quality related targets constitute actions in this phase. An account of what planning problems and issues Intrasoft was faced with is fully expanded in this paper.

The analysis phase entails the specification of the exact nature of the business moving along two basic directions, the technologies used and the business areas covered. A survey of all tools, techniques, metrics and methodologies employed or required so as to cover activities in each business area comprise the basic actions in this phase. The design of the QMS requires writing or gathering all QMS related documentation so as to address the entire scope of the QMS. Upon completion of the design phase, the QMS is put in effect and after a maturity period of several months, it is tested by independent "certification bodies". The attained experience from conducting such activities, an account of the specific tasks undertaken and the solutions established so as to address all problems listed constitute the core of this paper.

1 Introduction

Quality Management Systems (QMS) have become yet another computer buzz word of the 1990s. Many first heard this term only a couple of years ago, while others have not heard of it yet. This is because QMSs are something which are rapidly proliferating now and it is this decade where quality engineering is evolving as a new discipline within the IT industry.

Similarly, the ISO 9001 [5] standard has enjoyed much success in various industrial sectors.. It was only a matter of time for the benefits of compliance to this standard to become evident to the IT industry. As a result, an ever increasing number of IT firms are now seeking an ISO 9001 certification.

Implementing a QMS in compliance to the ISO 9001 standard is beginning to become a must for all IT firms. The Greek public sector and the CEU are increasingly seeking compliance to this standard with the private sector not lagging far behind in it's explicit quality demands. Firms which possess an ISO 9001 certification hold a significant business advantage over the firms which don't. It was both the business requirement as well as the strive for product and process excellence which dictated the need for the definition and deployment of a QMS in Intrasoft.

This paper looks into the various steps required so as to define a QMS. The problems and issues which Intrasoft was faced with will constitute the core of the presented material. The QMS life-cycle will be depicted first, from a macroscopic point of view. Subsequently, each QMS life-cycle phase will be scrutinized so as to analyze the underlying principles and details.

2 QMS Life-cycle

The waterfall model has enjoyed much success in software engineering. Its critics, however, point out that it is rigid, long and in some ways, unrealistic. Refraining from commenting on it's suitability for current software development in the large, this paradigm may be employed for the definition of a QMS. Minor enhancements must be made so as to account for the "salmon effect" and the paradigm offers a good starting point for any party interested in finding out what are the phases which must be undertaken so that a workable QMS is defined.

The basic concept behind the model is that a QMS must be established using an incremental approach. Phases containing specific inputs and outputs form the model skeleton. The model concentrates only on the QMS itself and not on it's effect on the software development and maintenance process. Having defined inputs and outputs for each phase, entry/exit criteria for each QMS life-cycle phase can be clearly defined. The following diagram depicts the QMS life-cycle.

3 QMS Planning

In classical Information Engineering, the planning phase is where important business related decisions take place with respect to the implementation planning for the required information systems. That is, one first assesses the current situation, performs the necessary business reengineering and then, outlines the information strategic plan.

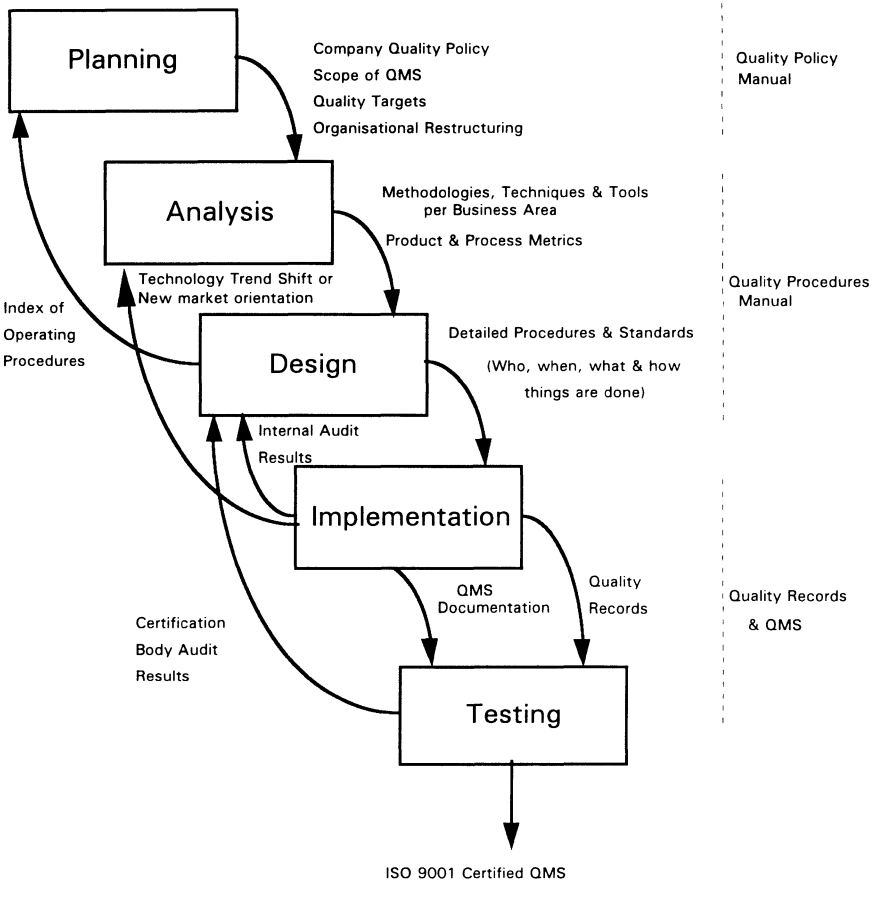
In a quite similar way, the QMS planning is conducted. The overall theme is to assess the current situation, to define targets and then, to establish the change process which will allow one to achieve his/her goals. Unfortunately, matters are not simple. Significant planning is required and above all, only when executive management commitment is assured can the quality program be launched. Otherwise, it is doomed to failure.

All quality programs commence when upper management realizes that it is to their company's best interest to establish a QMS. Usually, management representatives do not come to this decision on their own, but, they are guided to this direction by one or more of the following :

- A tender is issued which explicitly requires the existance of a QMS. Things become even more pressing when the tender requires ISO 9001 certification or in more lenient cases, ISO 9001 compliance.
- Customers start contacting upper management about the lack of testing and about "bug ridden" software. Project managers are then summoned to take action but they have no idea where to put a handle on the mess as they have "deadlines to meet" and "no time for testing".
- The lack of a *defined software development process* has a direct impact on the organization's quality and productivity. No two projects are doing things using the same tools, techniques and methodologies. Limited reuse is witnessed. The Technical Manager, responsible for coordinating/managing the entire development operation, is in total confusion about how things are being done.

Once the decision is made to *enter the quality era*, upper management establishes a quality unit and appoints a Quality Manager to head the unit. The scope of the quality unit is certainly a matter of discussion. There are several options available to the executive management. Two options are the most popular. The first option is to address the entire company's quality needs. That is, to undertake a Total Quality Management (TQM) approach. The second option is to focus on the "production line" first with a vision to expand to the rest of the organization once the production line is streamlined. In either case, the Quality Manager and his staff must be fully authorized *to halt the production line* when things go wrong. Therefore, the choice of the Quality Manager and of his/her staff must certainly be prudent. Here are some of the criteria for selecting the right person for heading the quality unit as well as for the selection of the subordinate *quality engineering* staff :

Figure 1. QMS Lifecycle



- Candidates must enjoy credibility throughout the organization. The quality unit staff will be dealing with people from upper management to shop floor.
- To possess a persuasive nature. He/She must be capable of selling the QMS even to cynics.
- Project management, organizational and communication skills.
- Imagination and self-confidence.

Having the quality unit in place, the first and possibly most crucial task for the newly appointed staff is the establishment of the overall quality policy and the definition of targets. The firm should operate within the framework of a *mission statement* and that is where one begins. The mission statement is usually a concise paragraph explaining what the organization is all about. If the firm does not have a mission statement, then the definition of a QMS is of limited value.

The quality policy statement focuses on the organization's customers and how the firm's products and services will meet their requirements. There is certainly some overlap between the mission statement and the quality policy, however, the emphasis of the quality policy is in quality related issues. The following are examples from Intrasoft's QMS.

Mission Statement

Intrasoft, as a whole, can be characterized as a system integrator operating in adherence to the following three-fold mission :

- To deliver integrated information systems and to produce software products.
- To offer sound technical and data processing services.
- To achieve utmost customer satisfaction by providing quality services and deliverables.

Quality Policy Statement

Intrasoft has as it's primary goal to design and implement systems which meet the customer's stated and implied needs. In order to achieve this goal, Intrasoft makes best use of the human, technological and material resources available to the company.

The mission statement and quality policy form a transparent boundary within which the organization operates. The QMS is also confined to these boundaries. However, up until this point, the entire QMS definition is floating somewhere in the stratosphere. What needs to be done in order to make it mundane is the establishment of some specific targets. This is a quite challenging task which requires a lot of literature review as well as a survey of current practices in similar businesses. The way to go about carrying out this task is to proceed either "formally", using a model, or "informally", using sound judgement and business sense.

The *formal* approach requires the firm to carry out an assessment of the organization using some type of an assessment model [4], [8]. The Capability Maturity Model (CMM) developed by the Software Engineering Institutes (SEI) or the European counterpart BOOTSTRAP may be a good place to start . By carrying out the assessment, the organization will know where it is situated with respect to such models. The organization can then establish targets in terms of where it wants to be, with respect to the *maturity levels*, within a specified time-frame. Various highly qualified organizations carry out such activities for any requesting IT firm.

A more "informal" way to tackle this task is to reassess the reasons why the organization is seeking to establish a QMS. Was it severe market pressure, was it product quality or was it the chaotic development process which triggered the organization to react ? A comprehensive answer to all

of these issues may be the best solution. Table 1 contains examples of problems and solutions which Intrasoft had to deal with while planning a QMS. Also, it offers just some indicative problems and solutions the QMS must address. It is now the right time to examine each and every one of the above targets in greater detail so as to outline a precise solution.

Table 1. Problems and Solutions of QMS planning

Problems within the organization	Solutions quantified as targets
Market demand for ISO 9001	Launch a quality program which will result in an ISO 9001 certification within a specified number of months (e.g. 18). Establish that all pertinent ISO 9001 clauses will be addressed by the QMS.
Bad product quality	Establish a target concerning product defects. State, for example, that within the next two years the organization will witness an 100% decrease in defect density. Establish the need for systematic testing and state when the new testing approach will be in effect. Testing approach should include the required testing levels.
Bad software costing *and/or* Delivery dates never met	Establish a target concerning the cost model. State, for example, that during the next year historical data from completed projects will be collected and that a cost model will be available within a year.
Limited reuse and low productivity *or* Development process is chaotic	Establish a specific software development approach which fits the needs of the firm. The software development and maintenance life-cycle must be generic so as to address all development project needs.

4 QMS Analysis

The precise definition of *what* a system will offer to it's end-users is something undertaken in the analysis phase using any software development methodology.
For the definition of the QMS, similar considerations take place. Typical issues which puzzle the entire quality unit as well as the *entire* organization are :

- Are software products being produced for one business sector? If not, are alternative development approaches required?
- What methodology fits the organization's development process best? Are more than one methodologies required? Why?
- What cost estimation method is best for the organization? Is an algorithmic model having Lines of Code as the driving force appropriate or are function points a better selection? Why?
- What tools and techniques are required so as to support the methodology(ies) and cost model selected? How will the selection process be carried out?
- What metrics should the organization employ so as to assess progress with respect to the planned targets? What tools are required so as to support the selected metrics? Is a metrics repository required and if so, what will it contain?

Needless to say that these are just a subset of the issues the organization will be faced with. Endless meetings will take place as it is important to receive input from everyone. That is, from upper management to the shop floor. No QMS can be established unless the quality unit is sufficiently confident that is has received adequate input from all key organizational roles and that upper management is fully committed in supporting the decisions taken. A classification of the issues to be addressed during the analysis phase allows for a better comprehension concerning what needs to be done.

4.1 Methodologies, Tools & Techniques

Browsing through any software related journal or attending any IT conference one invariably hears a lot about software development methodologies. The plethora of methodologies available to the software development firms, all of which claim to be a panacea to chronic software development problems, only confuse the perspective buyer. Information Engineering and Object Orientation are the two most popular development methodologies in the IT industry. What must be emphasized is that a careful analysis for this issue is warranted as the selection will probably dictate the success or failure of the entire QMS program. The place to begin is certainly the development methodology.

A mistake that is often made is that the distinction between CASE tools and methodologies is not always clear to the perspective buyer. A

methodology is a set of techniques covering one or more phases of the software development life-cycle whereas a CASE tool simply supports, either fully or partially, the corresponding methodology. Therefore, it only makes sense to zero in on the methodology first and then address the issue of the CASE tool selection. A number of articles exist which outline the criteria for selecting methodologies and CASE tools [9].

CASE tools are not the only tools which are suggested to be acquired. Other supporting tools are testing tools, configuration management tools, cost estimation tools, etc. *For each and every one of these categories of tools, the planned QMS targets and requirements must dictate the choice of the tool and not the other way around.* The right way to proceed is to purchase a function point tool, for example, once the QMS requirement is set that the function point methodology will be employed for cost estimation, while the wrong way to go about defining the QMS is to introduce function points to the QMS at the point in time when a persuasive vendor "lured" the organization into buying a function point tool.

To conclude on this topic, tools are not the **only** answer to chronic quality related problems. This is to say that their contribution to the resolution of quality related problems should not be overrated. If their scope is well-defined and if the tool users are well-trained, tools can offer solutions to a variety of problems ranging from configuration management to code generation. But only if tools are employed as part of a comprehensive QMS will their effectiveness be apparent in terms of quality and productivity increases.

4.2 Product and Process Metrics

The quote "you can not control what you can not measure" is certainly another *cliché* which has infiltrated the software engineering profession. And justifiably so! The fact that software project management was carried out for years using limited or no metrics could offer an explanation as to why so many products are bug ridden, late to the market and over cost [2].

Several models exist for launching a metrics program. Typical ones are the "Goal, Question, Metric" (GQM) paradigm[1] and the "Input, Output, Results" (IOR) model [3]. Intrasoft, after studying the above two models, adopted the latter one. Intrasoft found out in practice, what Bill Hetzel mentions in his book [3] concerning the basic metrics principle, namely *measurements come first, not last.* In a labor intensive software world, it is hard to allocate a "time slice" for the organization's management to contemplate on "goals" and "questions" which the top-down GQM paradigm dictates. Using the IOR model, metrics in the following classifications must be defined and subsequently, extracted so as to assist in the decision making process.

Input

Information about the resources (people, computer tools, other work products, ect.) applied and the process steps or activities carried out.

Output

Information about the deliverables and work products that are created.

Results

Information about the usage and effectiveness of the deliverables and work products in fulfilling their requirements. Measures that quantify the experience and satisfaction with the work product and how good it is in terms of satisfying the people using it.

While implementing the organization's metrics program, it may be wise to create a repository containing all project related information. Intrasoft selected this route and developed a repository entitled ARIS (Activity Reporting Information System). Coincidently, in ancient Greece, ARIS was the God of War which is indicative of what is required to *combat* project and quality management problems in the IT industry. ARIS is a comprehensive and fully integrated by data repository containing measurements for the following :

- Project Effort classified by Work Breakdown Structure (WBS) and Software Configuration Item (SCI) codes.
- Function Points, using Albrecht's 1984 revision [6].
- Problem Notices, recording all customer change requests after system(s) is installed.
- Defects, attached to each SCI code, recorded during each software development and maintenance life-cycle phase.

It is important to stress that metrics will never provide absolute answers to software engineering problems. Yet, if analyzed properly, they will assist in guiding management in the right direction so as to make the right decisions and to ask the right questions.

5 QMS Design

Design in software development is the phase where one takes into account the overall operating environment characteristics, and based on the available options, provides a specification about *how* things are to be implemented. It should come as no surprise that the design of a QMS moves along the same lines.

During analysis, a significant amount of time and effort is expended for the specification of the required methodologies, tools and metrics so as to properly support the QMS. A grace period of a couple of months is

suggested so that the engineering staff becomes intimately familiar with the selected tools and metrics. In other words, the software development operating environment characteristics must be pretty much set and stable before the design of the QMS can commence.

It is important to adjust to the fact that no magic book can be purchased which will allow for the quality unit to define a QMS which fits the idiosyncrasy of the organization. Every company has a different organizational structure, has a different mission statement and employs different technologies to solve its corresponding business problems. Certainly some commonalities can be found, however, the unique manner by which the company in question operates is what must be organized in terms of a QMS. *Defining a QMS which does not correspond to the organization's software practices and trends is of no significance and, chances are, it will soon be abandoned.*

Therefore, one arrives at the fundamental QMS design principle which is to know your business.

Another important prerequisite for designing a QMS is to know the standard to which the QMS will conform to well and in every detail. Having the planned QMS scope in mind, the selected standard (e.g. ISO 9001) must be scrutinized so as to select all the applicable clauses which need to be addressed by the QMS. Only then can the tedious and laborious task of writing all the procedures and work instructions can take place.

The task of writing the procedures and standards was one of the most painful experiences Intrasoft had in the quest for the attainment of a QMS. It is absolutely necessary to carefully plan out all QMS design activities. Viewing the design of the QMS as a project having a specific roadmap certainly helps. The procedures must cover anything ranging from in-process deliverables and inspections to quality system auditing [10]. "A procedure for writing procedures" which outlines how procedures and standards are to be generated and approved can be seen as the first procedure which must be written. In short, procedures contain some or all of the following :

who	who is the procedure addressing ?
	who is responsible for what ?
what	what is the scope of the procedure ?
	what activity does it cover ?
	what references were used ?
	what are the inputs and the outputs ?

	what are the required quality records ?
where	where does the corresponding activity take place ?
when	when does the corresponding activity take place ?
how	how is the corresponding activity performed ?
	what are the required steps or work instructions ?

The question often posed is "where do we stop" ? The following two criteria may offer an answer to this question.

- All deliverables produced within the scope of the QMS are generated or updated in accordance to some procedure and/or standard. This includes the inspection or review of the deliverable as well as the approval mechanism. Deliverables can be viewed as anything being produced including project documents, designs, control reports, plans and above all, code.

- All pertinent clauses in the relevant standard (e.g. ISO 9001) are adequately covered.

It is important to note that the quality unit staff are not necessarily the authors for all the required procedures and standards. There is, unfortunately, a misconception as many believe that the quality assurance department is the "paper generator". The quality unit staff simply organizes and coordinates the entire effort so as result in a coherent QMS. The best company human resources must be employed in order to generate procedures and standards of utmost quality as their knowledge is what must be disseminated to all others. Besides, authors are usually the strongest advocates of their work, therefore, it is desirable to have the production staff write their own procedures. This will greatly facilitate the adoption of the QMS on the part of the shop floor staff.

Last, but certainly not least, comes the task of documenting the QMS. The suggested approach is to generate two quality related documents [10]. These are:

Quality Policy Manual	An overview of the entire QMS.
Quality Procedures Manual	All the procedures and standards which comprise the QMS.

6 Implementation

Implementation in software engineering is the phase where the actual software is generated in accordance to the design specification. Similarly,

the QMS having been fully specified and documented is now ready to be fully deployed in the "production line".

The first, and possibly most crucial activity in this phase is training. The objective is to raise the level of awareness amongst the intended staff as well as to achieve "buy-in". The QMS must be presented to all concerned parties in a persuasive manner and the benefits of conforming to the QMS must be clearly stated. The quality policy manual and the quality procedures manual must become accessible to the applicable staff as they are to become intimately familiar with the pertinent procedures and standards.

The implementation of the QMS in the production line is affirmed by the corresponding quality records. Quality plans, test plans, inspection records, code review records are all considered as quality records. The objective of holding quality records is not simply to affirm conformance to the QMS. It is more an issue of understanding the past so as to be able to predict the future. In other words, all detected defects must be fully analysed and documented prior to undertaking any corrective action. By assessing the identified defects, appropriate preventive actions can be planned which will eliminate defects before they even occur. This is a substantial benefit of having a QMS in effect.

In contrast to the rigid waterfall model, the model presented in this paper (as it is depicted in fig. 1) envisages a significant amount of "recycling". The QMS must evolve as technology is advancing and as higher levels of software process maturity are sought. A fundamental principle of TQM is that of constant improvement. This translates into constantly assessing the QMS and making the corresponding improvements. The mechanism for implementing the QMS assessments and improvements is that of QMS auditing. Scheduled reviews should be carried out covering the entire scope of the QMS. QMS auditing allows for assessing the effectiveness of the QMS as well as for establishing the degree of conformance to the QMS. Audits are also useful for receiving precious feedback from the staff. All complaints and requests must be documented so that follow-up action can be planned.

At this point, the organization should be witnessing the positive effects of having a QMS in place. Intrasoft certainly showed improvements in all productivity, quality and time measurements recorded. Some of the positive effects noticed include :

- Deliveries are being made on time
- Productivity is increasing
- Reuse is increasing
- Proposals are being generated faster and with higher quality
- Defects are minimised
- Customer satisfaction is maximised

In short, quality systems are being delivered.

7 Testing

One of the major objectives for conducting testing in software engineering is to establish the required *level of confidence* before the software system becomes operational. Similar considerations certainly apply when a company is establishing a QMS.

Up until this point, the QMS is operational within it's scope and it surely is providing solutions to chronic software engineering problems. The quality unit can offer a *subjective* opinion as to whether the QMS is really what it should be and to report on it's effectiveness with respect to the outlined targets. However, just as a well-known software engineering practice dictates that coders must not test their own code, similarly, quality unit staff must not be the final assessors of the QMS.

What is required is the assessment of the QMS by an independent unit, namely, a **certification body**.

Certification bodies provide assessors who will carry out an assessment process to test the organization's readiness for registration to the selected standard. The assessment is carried out in two distinct phases.

The first phase involves the comprehensive review of all QMS related documentation. This includes, but is not limited to, the quality policy manual and the quality procedures manual. The objective of this phase is to assess conformance of the specified QMS with the standard of choice. Something which vastly facilitated this process for our firm was the establishment of ISO 9001 clause versus QMS element matrices. By providing such matrices the auditors were assisted in assessing both QMS completeness as well as proper interpretation of the selected standard.

Once the auditors are satisfied that the QMS is adequately documented and compliant to the selected standard (e.g. ISO 9001), the second phase can commence which is to have the auditors carry out a "spot check" at the auditee's premises. The objective of this phase is for the auditors to assess that all organizational units covered by the QMS are performing their job functions in accordance to the QMS. The important thing to keep in mind is that the role of the certification body is essentially to check that the auditees are controlling their business in the way they have deemed most suitable. This exercise was quite painful for our organization as the selected projects had to retrieve **all** the required *quality records* that were specified in the QMS and pertinent to the audited projects. This is another point where the internally developed activity reporting system (ARIS) was of great assistance.

The selection of a certification body is surely an important task which must be undertaken by the quality unit [7]. It is beyond the scope of this paper to elaborate more on this issue. The only point which must be stressed is that the certification body must itself be certified to carry our such activities (e.g. ISO 9001 certification of an IT firm).

It is a common misconception to believe that once a QMS is defined, it is there to stay with no updates. Technology is advancing, software engineers have identified better solutions, new development techniques have emerged and suddenly, the organization enters the never ending spiral of constant quality improvement. This should be viewed positively as a system that is not evolving will eventually be replaced.

8 Conclusion

Big bang integration was never seen as a viable solution to the software engineering problem of integrating individual software units. Similarly, if one is to be successful in launching a quality program having as an overall intent to define a QMS, an incremental approach is sought. A step-by-step approach is recommended using distinct phases each of which has defined targets and deliverables. Most importantly, entry and exit criteria from each phase must be defined and met, before advancing to subsequent phases. Opportunities for "regressing" to previous phases must also be envisaged and offered so as to account for new business goals or technology trend shifts. This paper is based on the attained experience of successfully implementing a QMS using such an incremental approach.

The basic problem Intrasoft encountered while implementing the QMS was getting everyone involved. As a result of not getting some key staff involved from the initial stages, a significant amount of rework was required so as to rectify shortcomings. What must be stressed is that the authors of procedures become the procedure's best advocate, therefore, every attempt should be made to get the people doing the various organizational functions to undertake writing the corresponding procedures and work instructions.

Having an overall approach to attaining a QMS is a critical success factor of paramount importance. Other factors which were covered in this paper include management commitment, training, clear and defined targets and personnel involvement. Failure to secure any one of such critical success factors will most likely have a negative effect on the entire quality program. In the fortunate case, however, of success, the organization will enter an endless QMS "maintenance phase" so as to secure that the QMS is offering solutions to **current** production problems.

Acknowledgements

Clearly, the establishment of a QMS requires a contribution from all company personnel. Therefore, I would like to express my gratitude to all Intrasoft personnel for their assistance in this most difficult task. I am especially indebted to the Technical Division Manager, Mr. A. Koukouvinos, for his support for my department's efforts. I would also like to thank my colleagues A. Filindras and K. Soutou for their valuable contribution in the design of the QMS, which approach I documented in this paper. Finally, my appreciation to my eagle-eyed reviewer P. Fitsilis for his excellent review of this paper as his postgraduate experience in "paper writing" was certainly an asset to me.

References

[1] Basili, V. & Weiss, D. A Methodology for Collecting Valid Software Engineering Data. *IEEE Transactions on Software Engineering*, 1984, **6**, 728-738.

[2] Grady, R.B. *Practical Software Metrics for Project Management and Process Improvement*, Prentice Hall, 1992.

[3] Hetzel, B. *Making Software Measurement Work*, QED Publishing Group, 1993.

[4] Humphrey, W.S. & Seet, W.L. *A Method for Assessing the Software Engineering Capability of Contractors*, Software Engineering Institute, 1987.

[5] ISO 9001 Standard, *Specification for design/development, production, installation and servicing*, ISO, 1987.

[6] Jones, C. *Applied Software Measurement*, McGraw-Hill, 1991.

[7] Munro-Faure, L., Munro-Faure, M. & Bones, E. *Achieving Quality Standards*, Pitman Publishing, 1993.

[8] Paulk, M., Curtis, B., Chrissis, M. & Weber, C. *Capability Maturity Model for Software*, Software Engineering Institute, Carnagie Mellon University, CMM/SEI-93-TR-24, 1993.

[9] Simon, A. The Integrated CASE Manifesto, *American Programmer*, July 1994.

[10] Waller, J., Allen, D. & Burns, A. *The Quality Management Manual*, Kogan Page Ltd, 1993.