IDEA GROUP PUBLISHING
701 E. Chocolate Avenue, Hershey PA 17033-1117, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com
ITP4131

An Empirical Study on the Software Process

Maria João Castro
Instituto Superior de Contabilidade e Administração do Porto, 4465 S.Mamede Infesta, Portugal
Tel: 351 22 9050000, mjcastro@mail.telepac.pt

Helena Mendes Moreira
Independent Research, Portugal, Tel: 351 22 6064008, Portugal, hmclmm2000@hotmail.com

ABSTRACT

Organizations have to track customer satisfaction and improve their software processes. Although a great deal of effort is expended in understanding what goes on within each of these areas, little effort has been applied to identifying and quantifying the relationships between the two. The objective of this research is to discover and establish potential relationships between the quality of the software process and the software quality, applying the Hypothetico-Dedutive method. The software quality is evaluated from customer satisfaction instead of consider all the associated attributes of a software product.

The hypothesis to be tested in this study is "the improvement of the software process increases the quality of the software product." In this article, we describe an ongoing research project conducted in a bank and a software house situated in Portugal. This study uses two years of data to measure the correlation between 18 software process variables and 7 software process attributes.

We used the SW-CMM software process model to evaluate the software process capability.

Some of the available results allow validating the hypothesis previously formulated.

INTRODUCTION

Despite the fact that the software has became one of this nation's major industries, difficulties still exist with delivering quality software within budget and on schedule. Improving software processes is a major focus for many organizations [Eman & Madhavji, 1999]. This was the *main motivation* for this study.

After, will be defined some terms used during the description of the paper and considered pertinent inside of the ambit of the software process assessment and improvement, by virtue of the great differences of existent concepts in the present literature.

Several definitions exist for the term *software*. As Macro referred, in 1990, it is of the common sense to consider the term "software" just applied to programs. This happens in the software maintenance professional's, consequence they be more in contact with programs than with documentation. Another possible definition of the term "software" it is given by McDermid (1991), that describes it consisting of the "group of programs, documentation and operational procedures that can be turned useful by the computers, for the men." This definition suggests that the term *software* is not just the source code. It also includes the whole documentation associated to a program, like the documentation of the analysis of the requirements, conception, user manuals, as well as the procedures used to maintain and operate the software system. It was this last definition that was adopted in this article.

The *software process* is the sequence of steps of the software development and maintenance processes [Humphrey, 1995]. When properly drawn, the description of the software process gives support to the software engineers in the development and maintenance of the software and work products associated (e.g., projects plans, drawing documents, code, tests and user's manuals). It is expected, that one team that follows a defined process can coordinate the work of individual members better, and tracking progress with more precision.

The *software process capability* describes the range of expected results that can be achieved by following a software process. The software process capability of an organization is a form of predicting the most likely outcome expected from the next software project that the organization has in hands [Eman & Madhavji, 1999].

The *software process performance* represents the actual results that can be obtained following a software process [Eman & Madhavji, 1999].

The *software process maturity* is the extension to which a specific process is explicitly defined, managed, measured, and controlled. Maturity, increases the potential to grow in capacity, indicates the richness of an organization's software process and the consistence it applied on projects throughout the organization [Eman & Madhavji, 1999].

A *software process model* supplies a technical and management platform for methods and tools application. Professionals that execute tasks associated to the software process [Humphrey, 1995]. The definition of the process identifies rules and specifies tasks. A definition properly drawn helps to assure that each work item is appropriately assigned, being your state registered. It also provides an orderly mechanism of learning. The organizations that communicate, develop and manage their processes more efficiently tend to have more success in the software process.

The SW-CMM (Capability Maturity Model for Software) was that chosen and used for the evaluation of the software process in these organizations. This model was developed by the Software Engineering Institute (SEI). The main focus of the SW-CMM model is of describing the sources and practices requested to obtain the software process maturity. The SW-CMM model defines an evolutionary road that will be followed by all organizations that seek the improvement of software process. The SW-CMM model also provides a platform to evaluate the software process actually in use as well as it identifies areas for improvement [Humphrey, 1989].

The *processes* resemble each other to habits- they are difficult to establish and difficult to abandon. By virtue of the software process being extensive and complex, its definition, understanding and applicability becomes difficult. All these reasons led to the creation of a *software process maturity platform* [Humphrey, 1989]. This platform consists of a ordered group of five levels of process capacity, progressively more matured, which makes possible to the organizations to determine the processes capabilities that they perform and to establish improvement priorities (Table 1):

Initial: the software process is ad-hoc and considered occasionally chaotic**.** Few are the software processes defined and success depends on individual effort.

Repeatable: basic project management processes are established to track costs, schedule and functionality. It is guaranteed the necessary discipline in the process to assure the repetition of previous successes on projects with similar applications.

Defined: the software process is documented, standardized and integrated into a standard software process for the organization, as for the engineering and management activities. All the projects use a process approved by the organization's software standard, for software development and maintenance.

This paper appears in *Issues and Trends of Information Technology Management in Contemporary Organizations,* the proceedings of the Information Resources Management Association International Conference.

Managed: detailed measures of the software process and product quality are collected. Both the software process and products are quantitatively understood and controlled.

Optimizing: Continuous process improvement is enabled by quantitative feedback from the process and from piloting innovative ideas and technologies. Improvement occurs both by incremental advancements in the existing process and by innovations using new technologies and methods.

There are several software process models and each one has particular aspects of software process.

Table 1: Capability maturity model

SW-CMM Level	Key Process Areas
Level 1	None
Level 2 Repeatable	Requirements Management
	Software Project Planning
	Software Project Tracking and Oversight
	Software Subcontract management
	Software Quality Assurance
	Software Configuration Management
Level 3 Defined	Organization Process Focus
	Organization Process Definition
	Training Program
	Integrated Software Management
	Software Product Engineering
	Intergroup Coordination
	Peer Reviews
Level 4 Managed	Quantitative Process Management
	Software Quality Management
Level 5 Optimizing	Defect Prevention
	Technology Change Management
	Process Change Management

AIMS

The aim of the study described in this article was to identify and quantify the relationships between the data collected from the software process performance and customer satisfaction. Identifying and quantifying "what drives what" will lead to improved software quality and increased customer satisfaction. This research contributes to these benefits by answering a number of key questions, namely:

- What software process variables should one focus upon, to maximize improvements in software product quality and customer satisfaction?
- What software process variables can be "ignored" or discontinued?

The method that was used in this research is outlined in Section 3. The model, the results, the main findings and the key contributions are summarized in the next sections.

METHOD

The **first phase** of the method, *observation*, includes all aspects considered in the previous sections and that are the main motivation of this study, mainly the reality known in the study in [Castro & Moreira, 1998]; this enabled the acquisition of context for the next phase of research.

The **second phase**, *preliminary information acquisition*, involves capturing information of the area in study for acquisition of more knowledge concerning the identified problem.

We collected the data for the case study from site interviews and documentation supplied by each case-study site. Each initial site visit consisted of an overview of the two years project and an interview. The interview questions addressed the software process practices, software process improvement methods used and planned, and used measures.

Two months after the initial interview, we asked each site to obtain projects data to conduct an assessment to start a software process improvement program. Each organization experienced difficulty in generating data.

The **third phase**, *theoretical formulation*, is an attempt to integrate in a logical way information previously gathered to rework associations amongst critical variables as well as their contribution or influence on problem explanation and resolution. This phase consisted of model building (the model will be described in the following section); the model served as a basis to test the hypothesis at study. It was created a model (fig. A1) that would allow evaluating if the software process influences the quality of the generated product.

The **fourth phase** of this method, *the additional collection of scientific data*, consists in gathering more data relative to the defined variables to enable their characterization. All information obtained in this phase will be the basis to the next phase. In this phase different projects were appraised, and results documented.

The significative volume of appraised projects allowed the analysis and interpretation of the results.

Software Process Data

The choice of the model adopted to gather *software process data* for this study and that will be described later went by a consensus among the people involved in the current study. For a great majority it is the most complete and easy model to adopt and with easier access the most immediate documentation. The used model was the SW-CMM, described previously.

The application of the SW-CMM model presupposes the completion of a questionnaire of software process maturity [CMU/SEI 94-SR-7, 1994]. The questionnaire doesn't include an assessment method; it is a component that is used in several assessment methods. In a subsequent phase, it was applied an adaptation of the CBA IPI (CMM Based Appraisal to Internal Process Improvement) method [CMU/SEI 96-TR-007, 1996].

This way the software process variables (Table B1) were evaluated and the organization software process maturity level estimated.

Customer Satisfaction Survey Data

The *customer satisfaction data* came from a survey that is done when the product is delivered, to assess customer satisfaction with various attributes of each software product project. Relatively to all the analyzed projects we have the customer survey, so the response rate is 100%.

The survey is done per project and includes questions that apply to key product attributes, namely: Functionality, Usability, Performance, Maintainability, Security, Documentation and Overall.

The customer is asked to provide a satisfaction rating for each attribute and project, using the scale shown in customer survey. The customer has the opportunity to provide text comments when answering each question, in addition to providing a satisfaction rating.

The customer satisfaction data has a number of characteristics that the analyst should keep in mind. The first is that the data is subjective in nature. That is, it is qualitative rather than quantitative. This means that one would not expect to obtain as high a value for the correlation coefficients, as one would from purely quantitative data.

The second point to note is that customer satisfaction may be influenced by a wide variety of factors, including some that are outside the scope of this survey. For example, price, product availability and easy of ordering. These influences are assessed using other surveys and tools, since no single instrument could assess all the variables. This research is aimed at product level characteristics and thus the results presented here are limited to those entities.

Although not being part of the product characteristics nor the process, the customers' involvement in the final phase of product analysis, proved to be reinforcement for some of the conclusions.

The **fifth phase**, *data analysis*, is described in detail later. In this phase it was used a number of years of real data collected on the two organizations. The software process and customer satisfaction data was extracted from various databases and the correlation coefficients between 18 software process variables and 7 customer satisfaction survey attributes were computed.

The data from the survey is input to, and archived in, SPSS database. The data is then analyzed extensively by a group of dedicated specialists and statisticians.

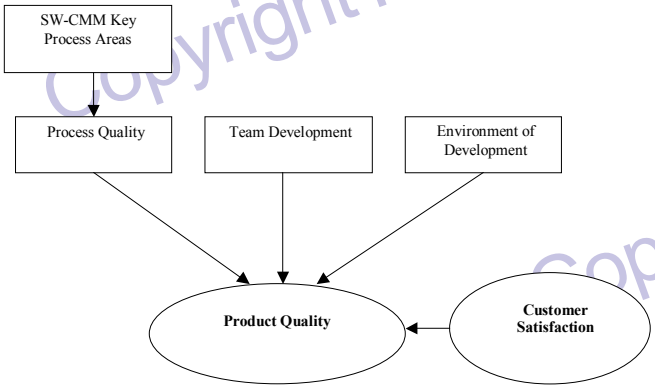
The **sixth phase**, *deductions*, consists of the process to reach conclusions based on the results of data analysis.

MODEL

Before the definition of the model to be adopted a revision of the state of the art on empirical studies on software process was performed (Basili & Green, 1994; Billings et al., 1994; Curtis et al., 1998; Curtis & Statz, 1996; Herbsleb & Zubrow, 1994; Lawlis et al., 1995; McGarry et al., 1994; Zahran, 1998). As a result of this, there is an initial conceptual model (fig A1). However, due to the dimension and the complexity of the starting model, data analysis only took into account the characterization of the software process- the 18 key process areas of the SW-CMM model (table B1) and the characteristics of the software product (table B2). For the association of these characteristics to the components of the model, bibliography on field studies in this area was previously reviewed.

We describe in detail each item that characterizes the software process and software product characteristics in Tabel B1 and Table B2.

Figure 1: Generic research model



ANALYSIS PROCEDURE AND INTERPRETATION OF DATA

For all analysis and inferences made after this point we must keep in mind that the universe we are working with does not include all organizations or projects although, sometimes, we use expressions in that sense. The results of empirical research carried out previously reflect the way reality was captured, thought and judged by those who definitely influence the aims and the options of the processes related with software process.

Hypothesis testing for this research amounts to determining if a coefficient is non-zero. A non-zero coefficient would show that the related predictor variable (e.g. process maturity level, key process area) does affect software quality. The objective of the analysis is to reject the null hypothesis (Ho) at the given confidence level thereby showing that the predictor does affect software quality.

First we point out the low percentage of organizations that adhered to the elaboration of the study, 2 in 73. Another aspect is the level of, almost, total ignorance with relationship to the existence of software assessment models.

Table 1: Software process variables

Independents Variables	Symbol	Description
Process Maturity	MPROC	It is the measure of the maturity level of a project level of a project's software process. It is the average of 18 KPA ratings used to assess a process's maturity [Boehm et al., 1995].
Requirements Management	KPA1	Management of requirements allocated to software to resolve issues before they are incorporated into the software project [Paulk et al., 1995].
Software Project Planning	KPA2	Developing estimates for the work to be performed, establishing the necessary commitments, and defining the plan to perform the work [Paulk et al., 1995].
Software Project Tracking and Oversight	KPA3	Tracking and reviewing the software accomplishments and results against documented estimates, commitments, and plans, and adjusting these plans based on the actual accomplishments and results [Paulk et al., 1995].
Software Subcontract Management	KPA4	Selecting a software subcontractor, establishing commitments with the subcontractor, and tracking and reviewing the subcontractor'' performance and results [Paulk et al., 1995].
Software Quality Assurance	KPA5	Reviewing and auditing the software products and activities to verify that they comply with the applicable procedures and standards and providing the software project and other appropriate managers with the results of these reviews and audits [Paulk et al., 1995].
Software Configuration Management	KPA6	Identifying the configuration of selected software work products at giving points in time, systematically controlling changes to the configuration, and maintaining the integrity and traceability of the configuration throughout the software life cycle [Paulk et al., 1995].
Organization Process Focus	KPA7	Developing and maintaining an understanding of the organization's and projects' software processes and coordinating the activities to assess, develop, maintain, and improve these processes [Paulk et al., 1995].
Organization Process Definition	KPA8	Develop and maintain a usable set of software process assets that improve process performance across the projects and provide a basis for cumulative, long-term benefits to the organization [Paulk et al., 1995].
Training Program	KPA9	Identifying the training needed by the organization, projects, and individuals, then developing or procuring training to address the identified needs [Paulk et al., 1995].
Integrated Software Management	KPA10	Integrate the software engineering and management activities into a coherent, defined software process that is tailored from the organization's standard software process and related process assets [Paulk et al., 1995].
Software Product Engineering	KPA11	Integrates all the software engineering activities: analyzing the system requirements allocated to software, developing the software architecture, designing the software, implementing the software in the code, and testing the software to verify that it satisfies the specified requirements. This way we produce and support correct consistent software products effectively and efficiently [Paulk et al., 1995].
Intergroup Coordination	KPA12	Participation with other project engineering groups to address system-level requirements,

All the respondents of the organizations we sent an e-mail to classified themselves at level 0 of the SW-CMM model.

Assessing Product Quality

The evaluation of product quality determines the capacity of the software, or corresponding documentation, to meet user's needs. Often no attempt is made to document and identify of the cause of a failure, and correcting the software becomes difficult.

Several measures can be applied for assessing product quality. Measures we included in this study gathered staff consensus among the software project managers. By virtue of not being an usual practice we selected those easier with management (Table B2).

It is of enhancing the significant increase with relationship to the software product quality along the 18 months of project analysis. We

Table 2: Software product variables

Dependents Variables	Symbol	Description
Functionality	FUNC	The product has necessary function to accomplish the user's task [Fenton & Pfleeger, 1997].
Usability	USAB	The product is easy to use in terms of accomplishing its desired task. It is easy to learn. The user can interact effectively with the product to enhance productivity. Effective training and documentation is available [Fenton & Pfleeger, 1997].
Performance	PERF	This relates to efficiency, i.e., the speed with which the product executes its functions. Included are overall throughput, memory, utilization and response time [Fenton & Pfleeger, 1997].
Maintainability	MAINT	This relates to easiness a software system or component can be modified to correct faults, improve performance or others attributes, or adapt to a changed environment [IEEE Standards Collection, 1993].
Documentation	DOC	This relates to the usalility, accuracy and understandability of software documentation [Fenton & Pfleeger, 1997].
Serviceability	SERV	This relates to techical support, response time and quality of corrections; easiness of installation procedures [Kitchenhan &Pfleeger, 1996]
Overall	OVER	This relates to the average of the six variables ratings used to assess software product quality.

conclude that in the 1st year the foreseen time and the actual time for project delivery tends to coincide, relatively to the bank organization. The presented results of the 34 analyzed projects refer medium times of delivered projects, per quarter.

This increase of delivery products under schedule, will probably be associated to the definition of several processes, although the level 1 of maturity is not yet obtained. The database created for the project characteristics is also probable to be in the origin of products completed on schedule, since similar projects are analyzed.

The only metric referenced in this article is Defect Density.

This metric assesses product quality by normalizing the number of defects in the software by the software size. Typical metrics are defects per thousand lines of source code (KSLOC) or per function points. Defect density accumulates the numbers of defects detected, as a result of design or code inspections conducted during specific phases of development effort. The basic measure is:

$$DD = \sum_{I=1}^{I} D_i/KSLOC,$$

Where

Di= Total number of unique defects during *i*th design or code inspection

I= Total number of inspections during that phase

KSLOC= Total number of source lines of executable code or non-executable data statements, in thousands (for code inspections, or where estimates of the lines of code are available on the basis of expansion from the known lines design statements).

The basis for evaluation is comparison against experience on past projects. There are no absolute values for this measure or industry standards to compare against.

From the analysis of the graph of the illustration C2 we conclude that the defects discovered in the software applications have been corrected from the 6th month onwards. This data also refers to the same organization, bank.

Concerning customers' satisfaction, it has been growing gradually in the two analyzed organizations. All the delivered applications are accompanied of a questionnaire that the customer will fill out.

DEDUCTIONS

We emphasize the low percentage of organizations that use other knowledge sources beyond source code. Probably, this is the only credible form and/or available to obtain updated information.

An analysis of the results reveals the inexistence of a set of rigorous tests to assure that a given change was adequately performed. This may be in the origin of a significant rate of errors that are added when a change is carried out; in these circumstances there is no reliability on the changes.

The majority of managers recognize the importance of using efficient processes and documentation update. In spite of the variety of available models for the software process they are not applied. The study revealed the ignorance of the available models.

FUTURE INVESTIGATION

The model presented in figure A1 offers a horizon for future research. New variables could be included in the model we have built and relevant hypotheses would be stated to be tested later on.

Further work is required for a more detailed characterization of the software process, in particular, the verification of stated deductions. Resuming, the model in figure A1 requires refinement based on questions raised by the results obtained from others studies. On the other hand, after this preliminary study, it is important to accompany the continuous software process improvement, based on quantitative measures.

CONCLUSIONS

This empirical study enabled the characterization of the *Software Process* in organizations situated in Portugal, to make simple deductions about the software process and to offer ideas for future research.

The lack of defined processes for software development and maintenance, the lack of updated documentation are big problems on software process in the Portuguese organizations. This practice does not allow managers to create teams for software process improvement.

We should point out that the stated results are related to particular applications. We believe that similar results could be obtained with different applications. However, different ones could emerge. As a consequence, it would be most interesting to extend this study to other applications in order to:

- validate these results;
- determine what specific connections are with other products.
- create of a common database, leaving all organizations take advantage of new ideas for improvement their software processes.

The contribution of this research is the discovery of quantified effects that Process Maturity can have on Software Quality. In addiction, and based in this study that will be concluded in March of 2002, a Program of Software Quality Improvement is being developed.

We conclude believing to have contributed for the enrichment of knowledge on software process more specifically in the software process improvement, aiming quality improvement and efficiency of software products.

REFERENCES

Basili, V. and Green, S., *Software Process Evolution at the SEL*, IEEE Software, Vol. 11, No. 4, July, (1994).

Billings, C., Clifton, J., Kolkhorst, B., Lee, E. and Wingert, W., *Journey to a Mature Software Process*, IBM Systems Journal, Vol. 33, No. 1, (1994), 46-61.

Boehm, B. et al., *Cost Models for Future Software life Cycle Processes: COCOMO 2.0*, Annals of Software Engineering, Science Publishers, Amsterdam, (1995).

Castro, M. J. and Moreira, H., *A Survey on the Software Maintenance Process*, Proc. of Conference on Software Maintenance- 1995, Computer Soc. Press, IEEE, Bethesda, Maryland, (1998), 265-274.

Curtis., B., Krasner, H. and Iscoe, N., *A Field Study of the Software Design Process for Large Systems*, Communications of the ACM, 31(11), (1988).

Curtis, B. and Statz, J., *Building the Cost-Benefit Case for SPI*, 1996 National SEPG Conference Tutorial, Atlantic City, NJ, May, (1996).

David, Z., et al., *Maturity Questionnaire*, Special Report, CMU/SEI 94-SR-7, June, (1994).

Dunaway, D. and Masters, S., *CMMSM-Based Appraisal for Internal Process Improvement (CBA IPI): Method Description*, Technical Report CMU/SEI-96-TR-007, April, (1996).

Eman, K. and Madhavji, N., *Elements of Software Process Assessment & Improvement*, Computer Soc. , IEEE, (1999).

Herbsleb, J. and Zubrow, D., *Software Process Improvement: An Analysis of Assessment*, also Technical Report CMU/SEI 94-TR-13, Pittsburgh, PA, August (1994).

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

Humphrey, W. S., *A Discipline for Software Engineering*, MA: Addison-Wesley, (1995).

IEEE, *IEEE Software Engineering Standards Collection, Institute of Electrical and Electronics Engineers*, New York, NY, (1993).

ISO 9000-3, *Quality Management and Quality Assurance Standards-Guidelines for Selection and Use*, (1991).

Kitchenham, B. and Pfleeger, S., *Software Quality: The Elusive Target*, IEEE Software, Vol. 12, No 1, (1996).

Lawlis, P., Flowe, R. and Thordahl, J., *A Correlational Study of the CMM and Software Development Performance*, Crosstalk, Vol. 8, No. 9, Sept (1995).

Macro, A., *Software Engineering: Concepts and Management,* Prentice-Hall International Ltd, hemel Hempstead, (1990).

McDermid, *Software Engineer's Reference Book*, Butterworth-Heinemann, (1991).

McGgarry, F., et al., *Software Process Improvement at the NASA SEL*, CMU/SEI 94-TR-22, Dec (1994).

Paulk, M., Weber, C., Curtis, B. and Chrissis, M., *The Capability Maturity Model: Guidelines for Improving the Software Process*, SEI Series on Software Engineering, Addison-Wesley Publishing Company, (1995).

Pressman, R. S., *Software Engineering- A Practitioner's Approach*, McGraw-Hill Book Company Europe, London, (1994).

Rout, T., *Software Process Assessment and Improvement*, Computational Mechanics Publications, (1998).

Zahran, S., *Software Process Improvement: practical Guidelines for business success*, Software Engineering Institute, Addison-Wesley, (1998).

CUSTOMER SURVEY

Please answer to the following subjects placing a circle in the appropriate number. The subjects are destined to the improvement of the customer's services. Any important comment can be placed in the available space.

Satisfaction Rating Scale
1 = very satisfed
2 = satisfied
3 = neither satisfied nor dissatisfied
4 = dissatisfied
5 = very dissatisfied
N = don't know or not opinon

Q1: What is your satisfaction with project XXX?

Functionality	1	2	3	4	5	N
Usability	1	2	3	4	5	N
Performance	1	2	3	4	5	N
Maintainability	1	2	3	4	5	N
Documentation	1	2	3	4	5	N
Serviceability	1	2	3	4	5	N
Overall	1	2	3	4	5	N

Q2: Was there a deadline for this project?

Yes [1] No [2] No comment [3]

Q3: If yes, did the team express confidence that this could be met?

Yes [1] No [2] No comment [3]

Q4: Was the project completed on time?

Very early [1] Early [2] On time [3]

Late [4] Very Late [5] Not completed [6]