IDEA GROUP PUBLISHING

701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

ITP4977

The Project Management in the Development Process

Luis Alberto Esteban Villamizar
Pamplona University of Columbia, Avda. De la Universidad, 30, 28911, Leganes (Madrid), Spain, lesteban@unipamplona.edu.co

Maribel Sanchez-Segura
Universidad Carlos III de Madrid, Escuela Politécnica Superior. Departamento de Informática, Avda. De la Universidad, 30, 28911 Leganés (Madrid), Spain, misanche@inf.uc3m.es

Antonio de Amescua
Universidad Carlos III de Madrid, Escuela Politécnica Superior. Departamento de Informática, Avda. De la Universidad, 30, 28911 Leganés, (Madrid), Spain, amescua@inf.uc3m.es

Luis García
Universidad Carlos III de Madrid, Escuela Politécnica Superior. Departamento de Informática, Avda. De la Universidad, 30, 28911 Leganés (Madrid), Spain, luisgar@inf.uc3m.es

ABSTRACT

This paper contains the results of the study of some software engineering pillars, from the point of view of the project management. The selected study pillars include standards (IEEE 1074, IEEE 12207), software engineering practices (SW-CMM) and methodologies (unified process and extreme programming). The analyzed project management processes include estimate, planning, monitoring and control, configuration management, quality management and human resources management. As a result of this study, we will offer an overview of the intensity of the processes inside the selected software engineering pillars. The results can be useful to decide the kind of software engineering methodology, process, etc, to be used in an organization or project, according with its management needs.

INTRODUCTION

The Software Engineering Body of Knowledge includes many elements defined by the main flow of knowledge, organizations and paradigms, and these have defined in a pragmatic way a division of concepts: technical and management area.

Some standards like IEEE 610.12-1990[5]: definition of terms for the software engineering, IEEE 1002-1987[6]: definition of the taxonomy of software engineering standards, IEEE 1074-1997[4]: standard for developing software life cycle processes, IEEE/EIA/ISO 12207.0[7]: standard for information technology software life cycle processes, and the IEEE SWEBOK[3]: guide to the software engineering body of knowledge; are considered methodological pillars for the software engineering and all these give importance to the management into the development processes and these allow a start point for the organization and analysis of software engineering as a science.

This paper finds the implicit relationships existent among different methods, standards and software engineering practices; and the software project management. We define the processes that determine the software project management and then assessment the management processes in each one of the pillars.

Section 2.1 describes briefly the selected pillars, and in section 2.2 we analyze how the pillars take into account the following project management processes: estimation, planning, monitoring and control, configuration management, quality management and human resources management

Figure 1: Structure for software engineering elements.

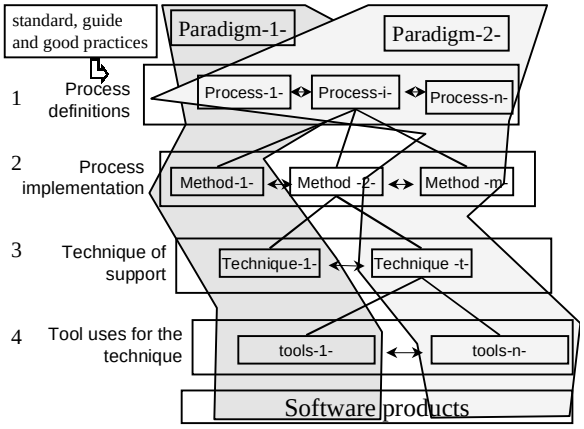
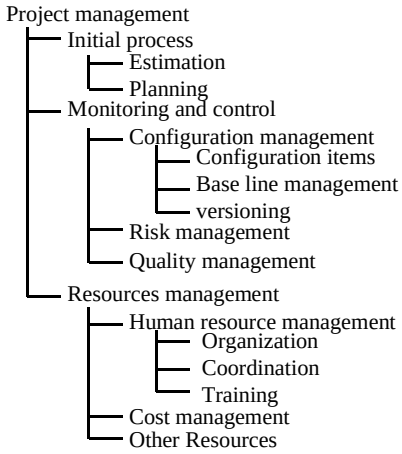


Figure 2: Factors for project management



ELEMENTS AND FACTORS OF STUDY

The definitions commonly used for the term "software engineering", are: "the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software", by IEEE [5], and a given in [14] as: "the software engineering is a discipline that integrates processes, methods and tools for the computers programs development".

These definitions do explicit a classification of the software engineering elements, (processes, methods, techniques and tools). With these concepts, we have defined a structure of the different software engineering elements, organized in levels. Figure 1.

On the other hand, in [13] is defined the project management as "application of knowledge, skill, tools and techniques to project activities to meet project requirements and assist or exceed necessities and expectations of the project stakeholder", so the project management consists basically of an initialization process, execution, control and termination of the project" [15]. The initiation includes the planning and estimate; the execution includes the organization, supervision and control and the termination includes performance evaluation [14].

Organizations like PMI (Project Management Institute) take charge of developing standards for the project management process, which offers a starting point to define a conceptual structure for the software project management. The basic areas of the project management, proposed by the PMI, are: integration, scope, time, cost, quality, human resources, communications, risk and procurement management.

Many project management techniques are applicable in the software development, but the product of this engineering has particular characteristics: invisibility, complexity and flexibility [2]. The invisibility difficulties to measuring the project progress; the complexity is superior than in others engineering and the flexibility should facilitate the changes that will be introduced during the software life[2].

According to the previous ideas about the project management we have grouped the concepts in the processes show in Figure 2.

We have selected the following project management factors in this study, because they are the more spreads:

1. Estimation: effort, cost, scope, time, etc.
2. Planning: activities definition and resources assignment (time, personal, etc.).
3. Monitoring and control: risk management, progress measures, report, etc.
4. Configuration management: definition and baselines control, versions handling, elements traceability, etc.
5. Quality management: measurement, defects control, improve ment process, etc.
6. Human resource management: staffing, work team structure, training, coordination, etc.

Software pillars to be analyzed

The five software engineering pillars selected to be studied are: IEEE 1074 Standard-1995 and IEEE/EIA 12207 Standard-1996 as representatives of the world efforts to standardize the software development process; software CMM as mechanism for process improvement; the unified software development process because it is frequently used in object-oriented development projects; and extreme programming as representative of new methodological tendencies.

IEEE 1074 Standard

The IEEE standard for developing software life cycle processes provides the set of activities that constitute the processes that are mandatory for software development and maintenance. It is used by the organizations to trace the activities specified in the standard, inside its own software life cycle model [4].

The standard defines 17 processes [4], as shows Figure 3.

Figure 3: Software Process by IEEE std1074

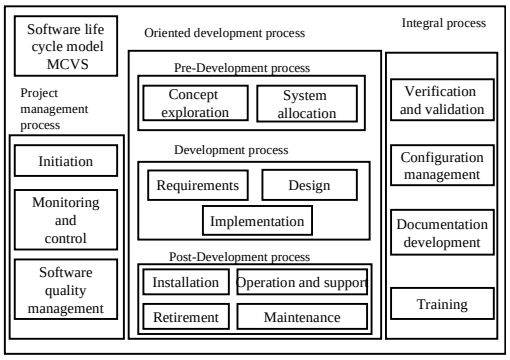


Figure 4: Software Process by IEEE std12207

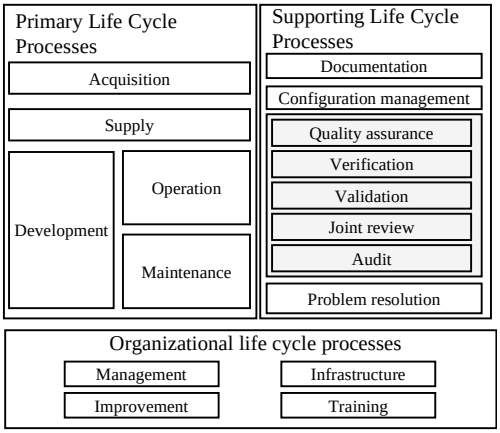
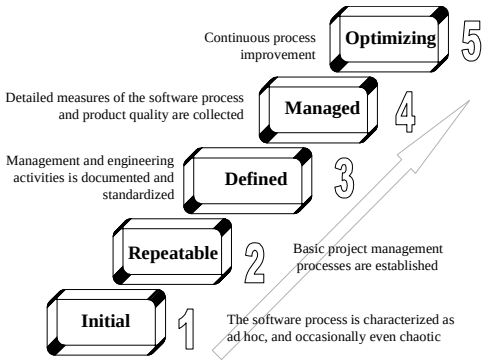


Figure 5: Software CMM levels



IEEE/EIA 12207 Standard

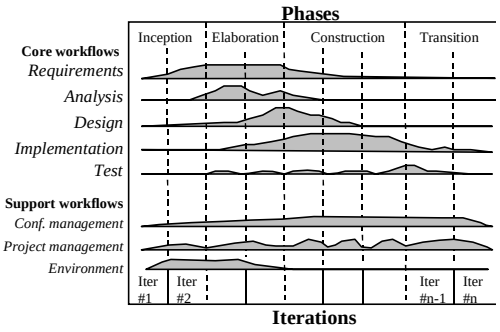
The industry implementation of international standard for information technology-software life cycle processes is a common model for management and software development [7].

The processes structure proposed by the standard is described in Figure 4.

Software CMM

The software CMM (Capability Maturity Model) is not a development process, it is not a development methodology, neither a particular technique inside software engineering, but it is an element of great importance, because it guides the definition of software development standard processes, according to particular organizational necessities.

Figure 6: The unified process



The software CMM pattern is progressive and permanent for which it has been defined five levels of software engineering process evolution, as shows Figure 5.

Each maturity level is defined according to some goals and purposes to find an appropriate standard process for the organization. It is as well as the TR-24 [9] and TR-25 [10] norms define and organize each one of the practices that conform the software CMM model.

The unified software development process

The diversity of software development methodologies and the growing acceptance of Unified Model Language (UML) as a tool for information systems description and specification have motivated the definition of a unified software development process (UP).

The UP defines a structure for the project activities and products, organized by phases, iterations and workflows (Figure 6).

The UP primordial features are summarized in three statements: use-case driven, architecture-centric, and iterative and incremental process.

Extreme programming

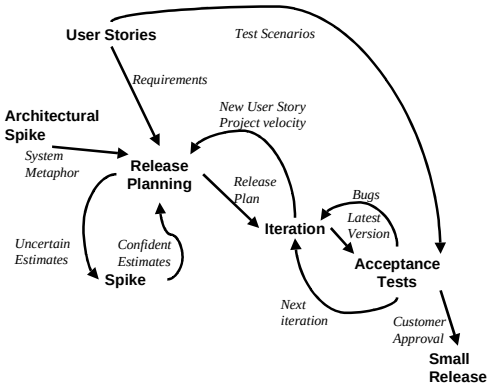
The software development methodologies impose a process disciplined with the purpose of making it more predictable, efficient, and with a strong emphasis in a strict planning; for this reason are considered bureaucratic and weight[11]. As a reaction to these methodologies, agile methodologies has arrived in the last years.

The agile methods are adaptive instead of predictive and people-driven instead of process-driven[11].

The extreme programming XP[1] is a agile methodology for the software development that emphasizes the client's satisfaction and authorizes the designers to respond confidently to the change requirements in any stage of a project.

XP improves a software development with four essential values: communication, simplicity, feedback, and courage; and by means of the extreme application of well-know software engineering practices[1].

Figure 7: XP process



A XP software project (Figure 7)[1] is composed of several releases developed in several iterations. The functional requirements are obtained by means of the client's participation and the non-functional requirements are obtained by means of a metaphor system that defines the system general architecture.

Analysis of project management processes into software pillars

We analyzed in group meetings the degree of importance of the project management into the several software engineering pillars and we identified the way in which each of them take into account the project management processes. This section present a summary about this analysis.

IEEE 1074 Standard

An organization does not have defined a hundred percent of the project management processes when adopts this standard, because these factors depend in a large part of a particular software life cycle selected for a software development project. However, this standard demands the traceability of relationships among the software life cycle and the processes and activities defined by the standard. We deduced that the standard determines the process general structure of the project management, but it does not define the details.

The standard does not make direct reference to the estimation process, however implicitly estimation activities are intimately linked to the planning processes.

The planning activities can be seen in the standard at different levels: 1) at project level: the responsibility is of the project management process (project initiation process). 2) at processes level: the standard defines the necessity to carry out specific plans for the following processes: quality management, installation, retirement, verification and validation, configuration management, documentation development, and training. In anyone of the cases the detail planning will be in terms of the processes defined by the standard and for the selected software life cycle model.

The monitoring and control is based on costs, calendar, problems, project performance management and reports. It is sustained in the qualitative risk analysis, contingency plans definition and project historical data registration.

The configuration management is defined inside the integral processes group, by means of activities of elaboration and execution configuration management plan, the configuration identification, execution and configuration control and project status computation. The standard does not define particular configuration items, these depend of the software life cycle, methods, techniques, tools and procedures used in each one of the processes.

The quality management is classified inside the project management processes and it has a strong relationship with the verification and validation process. The quality assurance process goal drives to the other processes in term of the client's satisfaction and the quality improvement internal programs. The fundamental workload is centered in the metric definition and necessities identification for process and product quality improvement.

The standard does not define a organizational structure, however, in general terms we can speculate about an organization (Figure 8) according to the defined roles for each one of the processes.

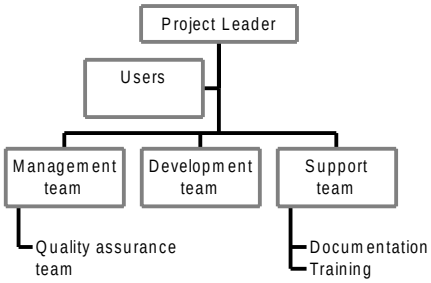
IEEE/EIA 12207 Standard

The adoption of this standard implies for the project management, the distribution of its activities inside different primary processes and mainly inside the organizational processes.

The estimation is considered explicitly inside the planning activity. However we identified the estimate necessity in each one of primary, support and organizational processes, under the title "process implementation", but these activities do not make explicit the methods or techniques to use.

The planning is considered by the standard at different levels: general planning, project planning and planning on each one of the processes.

Figure 8: Human resources Structure by IEEE 1074



The monitoring and control is responsibility of the management process and it is defined as only one of their activities.

The configuration management is defined inside the support processes, however it does not define neither the configuration management procedures, and configuration items structure, because the standard does not define the techniques to use in each one of the processes, and also the artifact kind to develop.

The quality management is treated by means of a quality assurance process, inside the support processes group and it is supplemented by other processes as: validation, verification, joint review, audit, and the training process. The quality is considered at three levels: product, process and system levels.

The standard does not specify any particular technique on the personal management, neither we can infer some organizational structure. The only aspect related to human resource is defined in the training process, classified inside the organizational processes group.

Software CMM

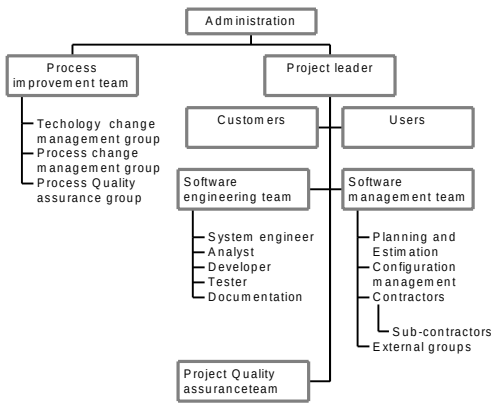
The CMM levels allow a general classification of the software engineering key practices, according to the emphasis in particular aspects of the software development. For this reason, the levels 2 and 4 emphasize the software management process.

The CMM adoption implies procedures definition and documentation, for estimating and planning at different levels. These procedures should define the methods, techniques and tools to use in the tasks execution.

The monitoring and control process is a learning mechanism for development process improvement, therefore the CMM demands the definition of procedures and strategies for the project status valuation, its deviations versus the plans, risks permanent valuation and the contingency plans disposition.

The quality management for the CMM is carried out in two levels, mainly at process level where it defines metrics to identify the process strengths and weaknesses. At project level, the quality is accountability of the audit permanent activities.

Figure 9: Human resources Structure by CMM



The personal management in the CMM is based in four basic concepts: training, communication, coordination and commitment. The CMM does not suggest any work team structure, but we deduce the following organizational structure (Figure 9).

This organizational structure surpasses the project boundary and it goes until the organization boundary, it includes the clients and the subcontract organizations by project.

The unified software development process

In general terms the unified process (UP) adoption as development process for a project, require: to define the plans in terms of the phases, iterations, workflows and artifacts; to use the UML artifacts as configuration items; and to assign responsibilities in terms of the organizational structure deduced of the processes. But there are some independent factors like the estimate process (it is not defined UP) of which depends in great measure the definition of project primary variables such as: scope, time, cost and product quality.

The estimation is not defined by the UP but it should be based on the artifact concept as measure unit, for this reason it is necessary the construction of classification models and to assign weight of importance on each artifact kind in terms of the artifact internal structure (UML diagrams, documents, etc.). This way, we suggest to use the proposal gives by[12], which proposes that the software size of the "s" is a function "fs" of the longitude, functionality and complexity: s::=fs(l,f,c) where: "l" represents the number of entities in a system, "f" represents the number of the functions provide the system, "c" it represents the problem of the complexity.

The UP suggests that the parameter "l" can be obtained from initial classes, objects and components diagrams, the parameter "f" obtained from the use-case diagrams and the parameter "c" obtained from the relationships between classes, use-cases, components and artifacts in general.

The planning should be kept in mind the three UP features: 1) use-case driven, 2) architecture-centric and 3) iterative and incremental. The planning process consists on identifying the iterations number, the use-cases to develop in each iteration and the manner as it was integrated according to the defined architecture. Therefore the UP suggests to carry out several kind of plans: project and iteration plan [8].

The main activities for monitoring and control are centered in the progress measure, contrasting these with the plan goals. The UP defines the termination criteria as the main progress measure[8] and the iterations as mechanisms to reduce the risks.

The project baseline should be defined and structured by [8]: technical and non-technical artifacts, development phases, workflows and iterations, and people roles. The UP artifacts kinds are clearly defined and structured as configuration items, because it uses UML.

The quality management activities are not explicit inside the UP, but these are related with the tests at product level.

We deduced an organizational structure in terms of the technical roles, the tasks inside the different workflows, development phases and the iteration order (Figure 10).

Figure 10: Human resources structure by unified process

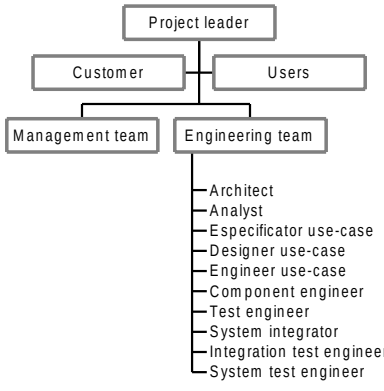
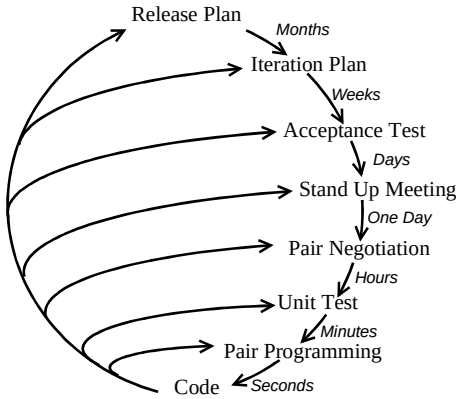


Figure 11: XP planning game



Extreme programming

The XP leaves in a hidden way the project management process, however for its philosophy is necessary that this process assists to its principles of simplicity, feedback, communication and courage. The simplicity makes that the management factors are governable and understandable elements by the whole development team, the feedback facilitates to adjust the plans, allowing to adapt quickly to the suggested changes as much for client as for development team.

Contrary to others methodologies, in the XP, personal management is of vital importance, because some of its practices (e.g pair programming, client participation, at all) and values (as the communication) are significant challenge for the human resource management.

The estimation is treated as control mechanism of the development speed, time, scope and priorities in terms of user histories.

The XP planning game is a negotiation activity between the project external forces (customers and project leaders) and the developers in terms of the four project control variables: cost, time, quality, and scope.

Figure 11 shows the planning process and feedback at different levels.

The monitoring and control depends on planning process at different levels (release, iteration and day to day). The feedback activities in XP are the best monitoring and control mechanism. The main milestones for the project control are the version delivery, culmination of iteration and stand-up meetings.

The configuration management process defines a few set of configuration items (user histories, plans, tests, tasks, code and bugs), according to the simplicity value.

The quality assurance interpreted as the client's satisfaction is one of the XP foundations. The main project quality indicator is the percentage of successful acceptance tests during an iteration.

The organization structure in XP is flat and it includes the client in the teamwork (Figure 12). The main strategies of personnel's management is the face-to-face communication among different participants in the process, the pair programming and the move people around practices.

Figure 12: Human resources structure by XP

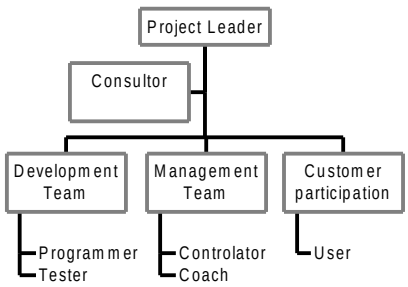


Table 1: Degree of detail of project management processes vs software engineering pillars.

	PROJECT MANAGEMENT FACTORS					
SOFTWARE ENGINEERING PILLARS	Estimation	Planning	Monitoring and control	Configuration. management	Quality management	Human resources management
IEEE std 1074-1995						
IEEE/EIA12207.0-1996						
Software CMM						
Unified process						
Extreme programming						

CONCLUSIONS

The project management is a permanent process during the whole software life cycle and it is indispensable support for the others technical processes defined by the methodology.

Beyond the software process as central point of this study the techniques, methods and tools used in a particular project and these are decisive part of the different project management processes.

Methodologies and software development standards distribute the project management tasks in several processes defined as necessary for the software development; however, a tendency exists to separate the technical processes from the management processes and to define the communication interfaces among these processes, in terms of input and output information.

Planning is the main factor inside the project management, however a good planning depends on a good estimation, an appropriate personal management and an appropriate monitoring and control strategies.

Planning is executed in several levels: mainly at project level, at phase level and process level, always in terms of the activities, resources, time, costs, scope and product quality.

Quality assurance is treated at two levels: process-oriented and product-oriented. The product-oriented quality management is directly related to a particular project. The process-oriented quality management looks for to improvement the software engineering practices and for this reason affects the principles and organizational structure for software development.

The monitoring and control is treated as responsible to maintain coherence between plans and project advances, in terms of the metric definition for visualize the project status and the an appropriate risk management.

The project management process becomes concrete and defined, when the software development methodology specifies the techniques, methods and tools to use during the project development.

In another way, each software engineering elements studied in this paper makes direct reference to the project management, however, some make more emphasis in some factors than in others. Table 1 shows this relationship by means of a gray scale on the cells that cross each one of the pillars (rows) and the project management factors (columns). The intensity represents the degree in that the project management factor is treated in the corresponding software engineering pillar (low intensity means a lower degree of detail).

Table 1 can be used by software engineers, organizations, etc., as a reference to select one or other standard, software process or methodology, according to the degree of intensity required for management processes in a particular project.

REFERENCES

[1] Beck, Kent Zapata Martínez, Francisco Javier, Una explicación de la programación extrema: aceptar el cambio, 1ª ed, Madrid : Addison-Wesley Iberoamericana España, S.A.

[2] Bob Hughes, Software Project Management, Mike Cotterell, International Thomson Computer Press, printer in the UK by The Alden Press, Oxford,1995.

[3] IEEE Guide to the Software Engineering Body of Knowledge-SWEBOK Trial Version. A Project of the Software Engineering Coordinating Committee, IEEE Trial Version 1.00. May 2001.

[4] IEEE Standard for Developing Software Life Cycle Processes; IEEE Std 1074-1995, sponsor Software Engineering Standards Committee of the IEEE Computer Society; Approved 9 December 1997.

[5] IEEE standard Glossary of Software Engineering Terminology, IEEE Std 610.12-1990, sponsor Standards Coordinating Committee of Computers Society of the IEEE, Approved September 28,1990.

[6] IEEE Standard Taxonomy for Software Engineering Standards, ANSI/IEEE Std 1002-1987, sponsor Software Engineering Subcommittee of the Technical Committee on Software Engineering of the IEEE Computer Society; Approved December 11.1986 IEEE Standards Board, Approved June 4 1987 American National Standards Institute.

[7] IEEE/EIA 12207.0-1996, Industry Implementation of International Standard ISO/IEC 12207: 1995 (ISO/IEC 12207) Standard for Information Technology, Software life cycle processes IEEE/EIA 12207.0-1996 (a joint standard developed by IEEE and EIA), March 1998.

[8] Ivar Jacobson, Grady Booch, James Rumbaugh. El proceso Unificado de desarrollo de Software. Addison Wesley, Pearson Educación S.A. Madrid 2000.

[9] Mark C. Paulk, Bill Curtis, Mary Beth Chrissis, Capability Maturity Model for Software TR-24. August 1991, CMU/SEI-91-TR-24, ESD-TR-91-24.

[10] Mark C. Paulk, Charles V. Weber, Suzanne M. Garcia, Marybeth Chrissis, Marilyn Bush, Key Practices of the Capability Maturity Model, version 1.1. February 1993, CMU/SEI-93-TR-25 ESC-TR-93-178.

[11] Martin Fowler, The New Methodology, Traducing: Alejandro Sierra, March /April 2003.

[12] N.E. Fenton, Software Metrics: A Rigorous Approach. London: Chapman and Hall, 1991.

[13] PMI Standards Committee. A guide to the project Management Body of Knowledge. Upper Darby, PA. PMI Management Institute. 1996.

[14] Roger S. Pressman, adapted by Darrel Ince, Ingeniería del software un enfoque práctico, quinta edición, McGraw Hill, Madrid 2002-11-27.

[15] Schwalbe Kathy, Information Technology project management. Course Technology. A division of Thomson Learning, Canada 2000.