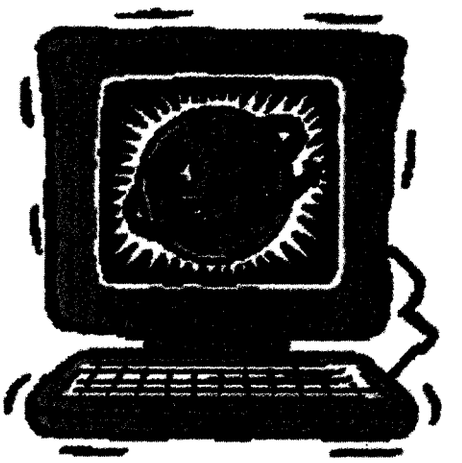
Proceedings

4th IEEE International Software Engineering Standards Symposium and Forum (ISESS'99)

"Best Software Practices for the Internet Age"

May 17-21, 1999
Curitiba, Brazil



IEEE COMPUTER SOCIETY

Sponsored by
IEEE Computer Society Technical Council on Software Engineering
In association with
IEEE Software Engineering Standards Committee (SESC)
The Institute of Electrical Engineers (IEE) UK
American Society for Quality (ASQ) Software Division
The International Center for Software Quality (CITS), Curitiba, Brazil

SESC
IEEE Software Engineering Standards Committee

ISESS'99

Best Software Practices for the Internet Age

Proceedings

Fourth IEEE International Symposium and Forum on

Software Engineering Standards

Curitiba, Brazil, May 17 – 21, 1999

Sponsored by

IEEE Computer Society Technical Council on Software Engineering

In Association with

IEEE Software Engineering Standards Committee
The Institute of Electrical Engineers (IEE) UK
American Society for Quality (ASQ) Software Division

In Cooperation with

International Center for Software Technology (CITS), Brazil

Los Alamitos, California

Washington • Brussels • Tokyo

IEEE Computer Society Order Number PR00068
ISBN 0-7695-0068-4
ISBN 0-7695-0069-2 (case)
ISBN 0-7695-0070-6 (microfiche)
ISSN Number 1082-3670

Additional copies may be ordered from:

IEEE Computer Society
Customer Service Center
10662 Los Vaqueros Circle
P.O. Box 3014
Los Alamitos, CA 90720-1314
Tel: + 1-714-821-8380
Fax: + 1-714-821-4641
E-mail: cs.books@computer.org

IEEE Service Center
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331
Tel: + 1-732-981-0060
Fax: + 1-732-981-9667
mis.custserv@computer.org

IEEE Computer Society
Asia/Pacific Office
Watanabe Bldg., 1-4-2
Minami-Aoyama
Minato-ku, Tokyo 107-0062
JAPAN
Tel: + 81-3-3408-3118
Fax: + 81-3-3408-3553
tokyo.ofc@computer.org

Editorial production by Bob Werner

Cover art production by Joe Daigle/Studio Productions

Printed in the United States of America by The Printing House

Table of Contents

Fourth International Symposium and Forum on Software Engineering Standards — ISESS'99

Session 9: Measurement and Evaluation

Panels

Workshops

Tutorials

Mini-Tutorials

Special Presentation

Foreword

Welcome to the Fourth IEEE International Symposium and Forum on Software Engineering Standards (ISESS'99) held in conjunction with the International Conference on Software Technology: Software Quality sponsored by the International Center for Software Technology (CITS). ISESS'99 precedes the 1999 ISO/IEC JTC1/SC7 annual meeting on Software Engineering Standards hosted by the Brazilian national JTC1/SC7 body. This year's location in Curitiba, a major South American software center, is a reminder of the worldwide cooperation taking place to make a universally valuable set of software engineering standards.

The goal of ISESS'99 is to capture users' experience to improve standards for the benefit of all users, whether in industry, government or academic institution. Its aim is to promote active international cooperation among practitioners, educators, standards developers, regulatory organizations, industrial users and software engineering researchers. ISESS'99 is taking place at a time when the challenges in software development and maintenance have never been greater, and at a time when public awareness is at a new high. ISESS'99 will address some of these challenges.

The period since the first ISESS Symposium in 1993 has seen both a growth in, and a consolidation of, software engineering standards from IEEE and ISO/IEC JTC1/SC7, the two leading publishers of these standards. Today there are approximately 75 standards published by both organizations, over double the 35 standards available in 1993. More significant than growth alone is the work done to consolidate the standards into a more consistent set. A key driver for this initiative was the 1995 publication of *ISO/IEC 12207, Information Technology — Software Life Cycle Processes.* It was adopted in 1998 as IEEE/EIA Standard 12207.0, the first of a 3-part standard that also includes additional information related to 12207. 12207 has proved to be the cornerstone for bringing greater uniformity among all process-related standards. The new 1998 collection of IEEE software engineering standards reflects this consolidation.

ISESS'99 is especially exciting, with a rich program offering insights into new standards, standards experiences and on selected technology topics. ISESS'99 has invited three industry leaders to help define some of the challenges facing us:

- Lloyd Mosemann II will talk about these issues in his talk "The Software Challenge Continues!"
- David Farber will address the future of the industry.
- Harold 'Bud' Lawson will address some of today's issues in his talk "From Busyware to Stableware, The Essential Transformation."

Starting with three tutorials on Monday (May 17, 1999), the ISESS'99 program contains a full technical symposium of panels, paper sessions and mini-tutorials. In addition, five workshops provide a forum where delegates can participate:

- Two of these workshops represent the Brazilian view on software quality.
- A third workshop will review the status of the Software Engineering Body of Knowledge activities being coordinated by SESC (Software Engineering Standards Committee) of the IEEE Computer Society.
- A workshop on Users of Software Standards is a continuation of the popular ISESS'97 workshop covering the results of the Software Engineering Standards users survey.
- Another workshop covers safety-related issues in software engineering and in current standards.

Panel sessions will cover three technical topics: object technology, Commercial-off-the-Shelf (COTS) software and formal methods. Delegates will have the opportunity to hear the status of standards under development, notably the work on System Engineering and the evolving ISO/IEC standard 15288. A mini-tutorial on the software measurement process will review the progress of another new ISO/IEC standard under development.

Through the documentation of best practices in wide use today, software engineering standards form the basis for a maturing discipline. These standards also form the basis of ongoing work on a new Body of Knowledge. These events hold the promise for the establishment and recognition of a software engineering discipline early in the next Millennium and benefits from discussions about best practices in symposiums such as ISESS'99.

Planning and bringing the ISESS'99 program and activities together required the effort of many individuals and organizations. A special thank you to the authors, presenters and speakers; as well as the panel and workshop chairs; and members of the Steering Committee, Program Management Committee and Program Committee. George Jackelen's handling of the logistics and circulation of the many contributed papers is very much appreciated. Twenty-eight papers were selected to be part of ISESS'99. A special thanks also goes to Anne Marie Kelly and her staff at the IEEE Computer Society Volunteer Services organization, and to Roberto Almeida and Marcia Barbosa of CITS, our host, for their tireless work to bring the ISESS'99 arrangements together.

Peter Voldner, ISESS '99 General Chair
p.voldner@computer.org

General Symposium Committee

General Chair
Peter Voldner
Peregrine Software Inc., Canada

National Chair
Roberto A.R. Almeida
CITS, Brazil

Program Chair
Paul Croll
Computer Sciences Corporation, USA

Program Vice-Chair
Peter Poon
Jet Propulsion Laboratory, California Institute of Technology, USA

Steering Committee

John Harauz, Chair
Ontario Hydro, Canada

Tom Hannan
Thomas' Point, USA

Leonard Tripp
Boeing Commercial Airplane, USA

Kival C. Weber
POLO de Software de Curitiba, Brazil

Dennis Rilling
DOD, USA

Management Board

George Jackelen
EWA, USA

Kathy Land
BTG Inc., USA

Dennis Lawrence
Lawrence Livermore National Lab., USA

Jim Moore
The MITRE Corporation, USA

Chris Neubert
HQ Army Materiel Command, USA

Danilo Scalet
CITS, Brazil

Victoria Stavridou
SRI, USA

Publications Chair
George Jackelen, *EWA, Inc., USA*

Publicity Chair
Kathy Land, *BTG Delta Research Corporation, USA*

Program Committee

Alain Abran, *Canada*
Alain April, *Canada*
Angela Bailey, *USA*
Joao Batista, *Brazil*
Nigel Bevan, *UK*
Jonathan Billington, *Australia*
Jorgen Boegh, *Denmark*
Pierre Bourque, *Canada*
Robert Dupuis, *Canada*
Anna Gianetta, *Italy*
Janet Gill, *USA*
Tom Hannan, *USA*
Anatol Kark, *Canada*
Dan Huyng Lee, *Korea*
Kathy Liburdy, *USA*
Stan Magee, *USA*
Jose Carlos Maldonado, *Brazil*
Deependra Moitra, *India*
Carlos Moura, *Brazil*
Tomoo Matsubara, *Japan*
Shigeru Nishiyama, *Japan*
Stuart Nunns, *UK*
Sarala Ravishankar, *India*
Philippe Robert, *France*
Hugo Rehesaar, *Australia*
Paul Rogoway, *Israel*
Terry Rout, *Australia*
Basil Sherlund, *USA*
Itana Maria de Souza Gimenes, *Brazil*
Victoria Stavridou, *USA*
Tan Wan Seng, *Malaysia*
Igor Ushakov, *USA*
Anca Vermesan, *Norway*
Stephanie White, *USA*
Alistair Walker, *South Africa*

MANTEMA: a Software Maintenance Methodology Based on the ISO/IEC 12207 Standard[1]

Macario Polo, Mario Piattini, Francisco Ruiz, Coral Calero
Grupo ALARCOS
{mpolo, mpiattin, fruiz, ccalero}@inf-cr.uclm.es
Escuela Superior de Informática
Universidad de Castilla-La Mancha
Ronda de Calatrava, 5
13071-Ciudad Real (Spain)
Tel.: +34-926-295300 Ext. 3706
Fax: +34-926-295354

ABSTRACT

The maintenance of information systems is one of the greatest problems in the software life cycle: It is the most conflictive, costliest, less planificable; and the process requiring the most resources. In spite of this reality, most organizations do not possess methodologies for software maintenance. These facts, key to the imperative need of controlling the maintenance process, have carried us to propose a maintenance methodology. In this paper an adjustment of the ISO/IEC 12207 standard for the processes of the life cycle maintenance is presented. This methodology is being used by Atos ODS, one of the most important European consultants on software outsourcing and maintenance.

Keywords: *Life cycle, maintenance, software, software processes.*

Introduction

Software maintenance is the most costly stage in the software life cycle. Some authors [1] affirm it absorbs between 67% and 90% of the total life cycle costs. Although most organizations have a methodology for software development, they recognize they do not follow any particular methodology for software maintenance [2]. Consequently, organizations need a complete guide for software maintenance to help maintainers control maintenance activities.

To define a software maintenance methodology, it is useful to take into account more global frameworks, e.g., those defined for software life cycle processes [3] [4] [5]. These frameworks can also help organizations define procedures and methodologies to achieve ISO 9000 certification.

On the other side, as it is well known [6], outsourcing software maintenance is a fast growing activity.

This paper is organized as follows: we review ISO/IEC 12207 [5], as a starting point for MANTEMA methodology; we then present this new maintenance methodology, and our conclusions and future work outlines are then discussed.

A short description of ISO/IEC 12207

ISO/IEC 12207 is a reference framework covering all aspects of the software life cycle processes. This standard is applicable to the acquisition of software systems, products and services for the supply, development, operation and maintenance of software products, and to the software portion of firmware [5].

It describes the architecture of the software life cycle processes, but does not detail how to implement the activities and tasks included in such processes. This standard is tailorable for different life cycle models and its adaptation to the used life cycle is the responsibility of every party (in our case, the maintenance acquirer and the maintenance supplier).

In ISO/IEC 12207, the activities to be implemented during a software life cycle are separate in three groups (Figure 1); there is also a tailoring process for adapting ISO/IEC 12207 to specific cases.

[1] This work is part of MÁNTICA (CICYT/European Union 1FD-097-0168) and MANTEMA projects (carried out by Atos ODS and Universidad de Castilla-La Mancha; ATYCA, Dirección General de Tecnología y Seguridad Industrial of the MINER, Ministerio de Industria y Energía, Spain).

ISO describes the five primary processes as activities of the:

1) *Acquisition*: acquirer, the organization acquiring a system, software product or software service.
2) *Supply*: supplier, the organization providing the system, software product or software service to the acquirer.

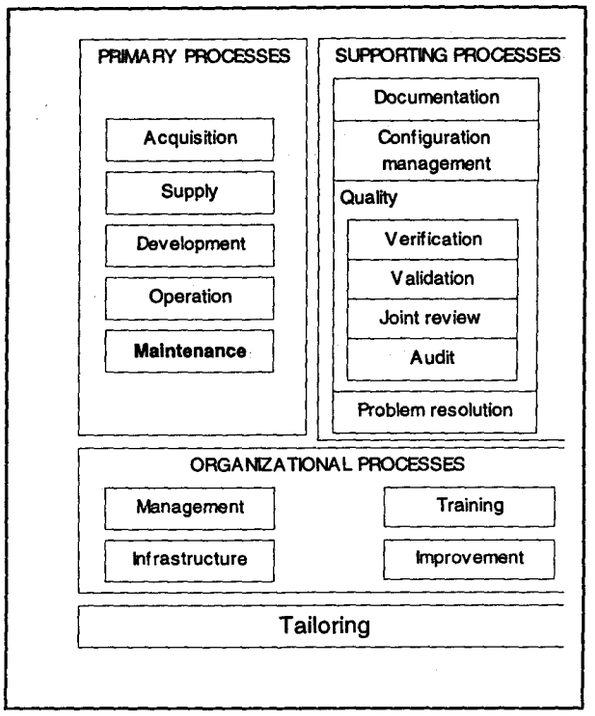


***Figure 1.* Life cycle processes in ISO/IEC**

3) *Development*: developer, the organization that defines and develops the software product.
4) *Operation*: operator, the organization providing the service of operating a computer system in its environment for its users.
5) *Maintenance*: maintainer, the organization providing the service of maintaining the software product; that is, managing modifications to the software product to keep it current and in operational fitness. This process includes the migration and retirement of the software product.

All the processes are formed by a list of activities and every activity by a set of tasks.

The maintenance process in ISO/IEC 12207

As seen in Figure 1, ISO considers maintenance one of the primary processes of software life cycle. The set of six activities and their respective tasks of the maintenance process is shown in Table 1 where we see that a Development Process is introduced in task 10 of the "Modification implementation" activity.

Activity	*Task no.*	*Task*		
Process implementation	1	Developing of maintenance plans		
	2	Establishing procedures for receiving and recording problems reports and modification requests		
	3	Implementing the Configuration Management Process (an organizational process)		
Process and modification analysis	4	Problem report or modification request analysis		
	5	Problem verification or replication		
	6	Developing options for implementing the modification		
	7	Problem documentation		
	8	Obtaining approval for the selected modification option		
Modification implementation	9	Conducting analysis and determining which documentation, software units and versions needing to be modified.		
	10	Entering at the Development Process to implement the modifications	10.1	Process implementation
			10.2	System requirement analysis
			10.3	System architectural design
			10.4	Software requirement analysis
			10.5	Software architectural design
			10.6	Software detailed design
			10.7	Software coding and testing
			10.8	Software integration
			10.9	Software qualification testing
			10.10	System integration
			10.11	System qualification testing
			10.12	Software installation
			10.13	Software acceptance support
Maintenance review/Acceptance	11	Conducting reviews, with the organization authorizing the modification, to determine the integrity of the modified system		
	12	Obtaining approval		
Migration	13	Accordance with ISO/IEC 12207		
	14	Developing a migration plan		
	15	Notifying users of the migration plans and activities		
	16	Parallel operations of the old and new environments		
	17	Notifying users of the arrival of the migration		
	18	Post-operations review		
	19	Saving old environment data		
Retirement	20	Retirement plan		
	21	Notifying users of the retirement plan and activities		
	22	Executing parallel operations		
	23	Notifying users of the retirement		
	24	Saving old environment data		

Table 1. Activities and tasks in the ISO/IEC 12207 maintenance process.

A strict follow-up of this maintenance process may im-

ply overlapping of the activities and tasks listed in Table 1 as, for example:

- The 10.13 task (*software acceptance support*) is also done subsequently with task 12 (*obtaining the approval*).
- The 10.10 task (*system integration*) is done as part of task 11 (*maintenance process*).
- The 10.1 task (*process implementation*) may include tasks previously executed in task 3 of the maintenance (*implementing the configuration management process*).

Furthermore, the existence of one maintenance process for all possible types of maintenance involves excessive time in the tailoring process for ISO/IEC 12207's adaptation to a particular case. Then, we think it would be adequate to modify ISO/IEC 12207 to adapt it specifically for maintenance. In the next section we propose a methodology incorporating these ideas.

MANTEMA: a methodology for software maintenance

In this section we present a new maintenance methodology, "MANTEMA", built from ISO/IEC 12207. This standard has been chosen because "ISO/IEC 12207 will drive the world software trade and will impact maintenance" [7].

In this methodology we integrate some processes of ISO/IEC 12207 (acquisition, supply, development, documentation, verification, validation, joint review, audit, problem resolution and management). In this manner, a general view of our methodology processes is shown in Figure 2.

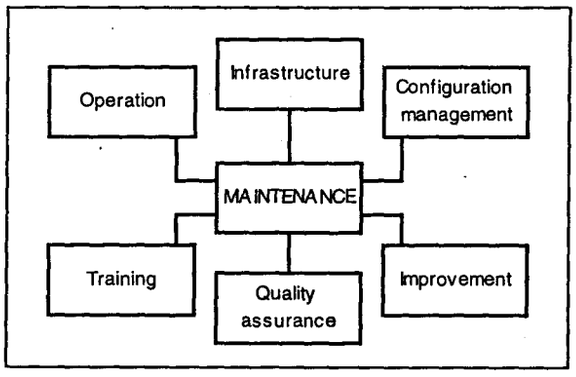


Figure 2. Processes in our methodology.

MANTEMA consists of some activities and tasks which vary with the maintenance type. The set of initial and final tasks and activities is common to all types. Figure 3 is a graph illustrating these ideas. Five types of maintenance are defined as follows:

1) *Urgent corrective*: a detected error prevents normal system operation and the solution time is critical.
2) *Not-urgent corrective*: a detected error does not block the normal operation of the system and the solution time is not critical.
3) *Perfective*: when new functionalities are added to the system.
4) *Preventive*: consists of the software modification to improve its maintainability and quality properties.
5) *Adaptive*: when the system will change its execution environment.

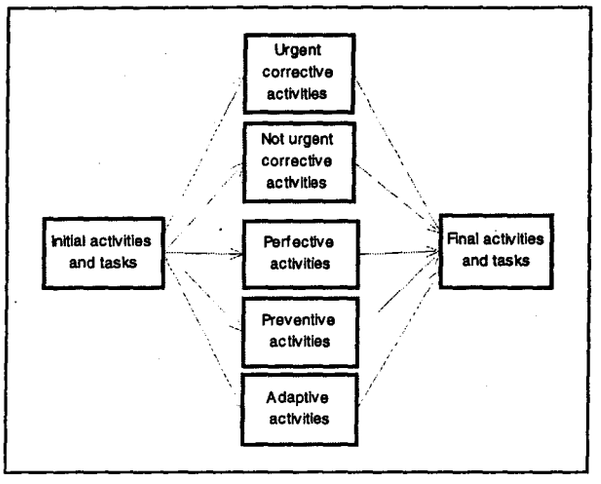


Figure 3. General view of MANTEMA.

Such definitions (Table 2) correspond with the definitions given in [8].

MANTEMA	*IEEE, 1992 [8]*
Urgent corrective	Emergency
Not-urgent correc-	Corrective
Perfective	---
Preventive	Perfective
Adaptive	Adaptive

Table 2. Equivalence of definitions

In the next pages, we detail (Tables 3, 4 and 5) the activities and tasks of the common initial, the urgent corrective and the common final activities and tasks. The rest of the table contents relate to the other types of maintenance are similar to urgent corrective. These tables are built with a similar style as those of [8].

In each table we have the list of activities composing it. Below every activity its respective set of tasks appears. Furthermore, for every task we have written, when it is needed, its inputs, outputs, responsibilities and interfaces with other processes. After the tables we have also explained briefly each task.

	COMMON INITIAL ACTIVITIES AND TASKS										
	Initial study				Process implementation					Study of the modification request	
	0.1 Beginning and information recollection	0.2 Preparing maintenance proposal	0.3 Contract	0.4 Planning customer/ supplier relations	1.0 Knowledge acquisition	1.1 Developing plans	1.2 Defining modification request procedures	1.3 Implementing Configuration Management process	1.4 Preparing test environments	2.1 Reception of the modification request	2.2 Decision about the type of intervention
Inputs	Maintenance request	Questionnaire Interviews	Maintenance proposal	Maintnance contract	Software product in operation	Software product in operation Maintenance plan	Maintenance plan	Software product in operation Maintenance plan	Software product in operation	Modification request	Received modification request
Outputs	Questionnaire	Maintenance proposal	Maintenance contract	Customer/ supplier relations planning (responsibilities, meetings calendar, etc.)	Software product in operation Documents about such software	Maintenance plan	Document models and forms *Regulatory ways*	Configuration Management Process	Copies of the software elements in operation	Received modification request	Report about type of maintenance, criticise, etc. Decision about the type of maintenance to apply
Responsible	Customer Maintainer	Maintainer	Maintainer Customer	Maintainer Customer	Maintainer	Maintainer	Maintainer	Maintainer	Maintainer	Customer Maintainer	Maintainer
Interfaces with other processes				Tailoring				Configuration Management	Quality assurance		

Table 3. Common initial activities and tasks

0.1 Beginning and information recollection

In this task a questionnaire is filled in. Certain aspects of the software that will be maintained are put together (operating system, programming language or languages, number of programs and any other data of interest).

0.2 Preparing the maintenance proposal

The Maintainer constructs, from the data collected in the previous task, a software maintenance proposal.

0.3 Contract

A maintenance contract is drawn up, discussed and signed by the customer and the maintainer

0.4 Planning customer/supplier relations

A schedule of meetings is produced, and authorized representatives and people in charge from both parties to the contract are appointed.

If the maintainer coincides with the developer, this first activity may be omitted, since the information is already known by the maintainer; the contract is not needed and neither is the software maintenance proposal, which is a preliminary version of the contract.

1.0 Knowledge acquisition

In this task, the maintenance team obtains the specific information about the software to be maintained. The maintenance team must study existing documentation, code of programs, cross references and must interview the users and with current maintainer, etc.

During this task, the customer is responsible for the software maintenance, since the future maintainer acts as a mere observer. However, the maintenance team must provide a good set of documentation after this period including audit reports, possible improvements, etc.

1.1 Developing plans

A maintenance plan is built up.

1.2 Defining Modification Request procedures

The maintenance team generates document templates for the presentation of Modification Requests. In addition, procedures for presenting and deciding modification requests, who is in charge of their reception and study, etc. are established in this task.

1.3 Implementing Configuration Management process

This task makes an interface with the configuration management process (see figure 2) of the organization. This serves to manage efficiently the modifications made in the system.

1.4 Preparing test environments

The maintenance team prepares copies of the production software environment for implementing maintenance interventions on them.

2.1 Reception of the Modification Request

The maintainer receives a Modification Request from the customer, which must be registered.

2.2 Decision about the type of maintenance

From the Modification Request received and registered in the previous task, the maintainer decides which of the

five maintenance types must be applied. After this, the Modification Request is put into the queue, perhaps putting it ahead of other previous modification requests.

Table 4 corresponds to the urgent-corrective activities and tasks:

	Error analysis	Urgent-Corrective intervention			Intervention close		
	1.1 Investigating and analysing causes	2.1 Making corrective actions	2.2 Complimenting documentation	2.3 Verification of the modifications	3.1 Documenting correction	3.2 Putting the software product to production environment	3.3 Saving intervention
Inputs	Software product in operation Urgent error Modification request	Software to be corrected	Old software (with errors) New software (without visible errors)	Corrected software Unitary test cases Integration test cases	Documentation of: · Corrective actions · Tests · Customer conformity	Corrected software	Full document
Outputs	Software to be corrected	Corrected software	Documentation with the corrective actions made	Assurance of the correction of the error	Full document	Corrected software in operation	Stored full document
Responsible	Maintainer	Maintainer	Maintainer	Maintainer	Maintainer Customer	Maintainer	Maintainer
Interfaces with other processes						Configuration Management	

Table4 Urgent-corrective activities and tasks.

1.1 Investigating and analysing causes

Error must be reproduced and its causes must be found. Perhaps different implementation alternatives will be studied.

2.1 Making urgent-corrective actions

The maintenance team must perform the actions adequate to correct the error. We understand that both "Software product in operation" and "Corrected software" are formed in this way by the set of programs or databases to be modified and all the related documentation.

2.2 Cumplimenting documentation

A document detailing the changes made on the software product must be filled in.

2.3 Verification of the modification

The correct operation of the software must be tested with unitary and integration tests.

3.1 Documenting correction

A document describing the test cases and results of the previous task must be filled in. This document could be used to make regression tests when we have to change this software again. Furthermore, perhaps the customer must sign a conformity document.

3.2 Putting the software product to production environment

The maintainer must put into production the changed software.

3.5 Saving intervention

All the documentation produced during the intervention is registered and saved.

The last table we show corresponds to the Common final activities and tasks. These activities are the same that ISO/IEC 12207 ones: migration and retirement. Their respective sets of tasks are also the same.

CONCLUSIONS AND FUTURE WORK

In this paper we have presented MANTEMA, a methodology for software maintenance built from ISO/IEC 12207. The MANTEMA methodology is being applied by this consultant and is in the first stage of a collaboration project between our university and Atos ODS.

We think one of the most positive contributions of this methodology is that it constitutes a quick reference guide for software maintenance due to its separation into several activity tables, i.e., one for every type of maintenance. This minimizes the effect of the continuous tailoring of a standard to a specific application case.

We are concluding the methodology and our new works are routed to the application of metrics to software maintenance, i.e., putting special attention to metrics for big systems and new environments [9].

Another future activity is the implementation of an evolved version of MANUTEN-CONDUCT, an auto-

	COMMON FINAL ACTIVITIES AND TASKS									
	Migration				Retirement					
	0.1 Developing migration plan	0.2 Notifying the future migration to the users	0.3 Executing parallel	0.4 Post-operation review	0.5 Storing old environment data	1.1 Developing retirement plan	1.2 Notifying future retirement	1.3 Executing parallel	1.4 Notifying retirement	1.5 Storing old environment data
Inputs	Old software environment in operation New software environment prepared	Migration plan	Migration plan Old software environment in operation New software environment prepared	New software environment in operation	Old environment data	Old software products in operation New software product prepared	Retirement plan	Retirement plan Old software products in operation New software product prepared	Retirement plan	Old environment data
Outputs	Migration plan	Note to the users with information about the migration (date, duration, etc.).	New software environment in operation	Results of the post-operation review	Old environment data stored	Retirement plan	Note to the users with information about the retirement (date, duration, etc.).	Old and new software products coexist	New software products in operation Old software products retired	Old environment data stored
Responsible	Maintainer	Maintainer	Maintainer	Maintainer	Maintainer	Maintainer	Maintainer	Maintainer	Maintainer	Maintainer
Interfaces with other processes			Training					Training		

Table 5. Common final activities and tasks.

matic tool for integral support of maintenance developed by Atos ODS.

REFERENCES

[1] Card, D.N. y Glass, R.L., Measuring Software Design Quality. Englewood Cliffs. USA.

[2] Piattini, M.G., Ruiz, F., Polo, M., Villalba, J., Bastanchury, T. and Martínez, M.A. Mantenimiento del software: conceptos, métodos, herramientas y outsourcing. Ed. RAMA. Madrid, Spain, 1998.

[3] IEEE Std 1074-1995. Standard for Developing Software Life Cycle Processes (ANSI).

[4] IEEE Std 1074.1-1995. Guide for Developing Software Life Cycle Processes (ANSI).

[5] ISO/IEC 12207. Information Technology. Software life cycle processes.

[6] Communications of the ACM (contains a monograph about outsourcing), vol. 39, no 7, 1996.

[7] Pigoski, T.M. Practical Software Maintenance. Wiley Computer Publishing. New York, USA, 1997.

[8] IEEE Std 1219-1992. Standard for Software Maintenance.

[9] Piattini, M.G., Calero, C., Polo, M. and Ruiz, F. Maintainability in object-relational databases. Proceedings of The European Software Measurement Conference, pp. 223-234. Antwerp, Belgium, 1998.

Published by the IEEE Computer Society
10662 Los Vaqueros Circle
P.O. Box 3014
Los Alamitos, CA 90720-1314

IEEE Computer Society Order Number PR00068
ISBN 0-7695-0068-4
ISSN 1082-3670

ISBN 0-7695-0068-4