PROCEEDINGS

INTERNATIONAL SYMPOSIUM ON SYSTEMS INTEGRATION

in conjunction with

INTERSYMP 2001

The 13th International Conference on System Research Informatics and Cybernetics

July 31 to August 4, 2000, Baden-Baden
Germany

Edited by
G.E. Lasker, University of Windsor, Canada
A. Dahanayake, Delft University of Technology, The Netherlands

Published by

THE INTERNATIONAL INSTITUTE FOR ADVANCED STUDIES
IN SYSTEMS RESEARCH AND CYBERNETICS

PROCEEDINGS

INTERNATIONAL SYMPOSIUM ON SYSTEMS INTEGRATION

Edited by
A. G.E. Lasker, University of Windsor, Canada
A. Dahanayake, Delft University of Technology, The Netherlands

Papers are reproduced here from camera-ready copies prepared by the authors.
Canadian Cataloguing in Publication Data

Main entry under title:
International Symposium on Systems Integration

Proceedings of the Symposium in conjunction with INTERSYMP'2001,
13th International Conference held in Baden-Baden, Germany, July 31, 4 Aug. 2001.

ISBN 1894613-25-2

Publisher: The International Institut for Advanced Studies in System Research and Cybernetics
c/o Dr. George E. Lasker
School of Computer Science
University of Windsor
Windsor, Ontario
Canada N9B3P4

Organizing and Program Chairs

Waltraud Gerhardt, Delft University of Technology, The Netherlands
Ajantha Dahanayake, Delft University of Technology, The Netherlands
Chris Zhang, University of Saskatchewan, Canada
Bogdan Czedjo, Loyola University, USA

International Program Committee Members

Pat Martin, Queen's University of Kingston, Canada
Sherman Lang, National Research Council, Canada
Waltraud Gerhardt, Delft University of Technology, The Netherlands
Ajantha Dahanayake, Delft University of Technology, The Netherlands
Chris Zhang, University of Saskatchewan, Canada
Bogdan Czedjo, Loyola University, USA

Table of Contents

An Conceptual Architecture Proposal for Software Maintenance

Francisco Ruiz, Mario Piattini, Macario Polo

Department of Computer Science, Alarcos Research Group

University of Castilla-La Mancha, Spain

fruiz@inf-cr.uclm.es

Abstract:

An important principle of modern software engineering is the system split in encapsulation layers which can mostly be specified, designed and constructed independently. We have applied this philosophy in order to define a conceptual architecture that allows to embrace the implicit complexity of Software Maintenance (SM) management. Four conceptual levels that are based on the MOF (Meta-Object Facility) standard for object oriented modelling proposed - by the Object Management Group (OMG, 2000) - have been defined.

Keywords:

Conceptual Architecture, Software Maintenance, Software Engineering Environment, MOF, Process Metamodeling.

1. Introduction

Traditionally, the software engineering community (scientists and professionals) have considered the maintenance process to be less important than the development process. This has been due to a number of different factors: technological, social, psychological etc. In recent years, however, everything that occurs once a software product has been delivered to users and clients has been receiving much more attention owing to the significant economic importance that it has in the information technology industry. Proof of this are the recent cases of the year 2000 effect and Euro adaptation.

Unfortunately, the methodologies for the development process have difficulties of adaptation to the Software Maintenance Process (SMP) because both processes have different characteristics (Ruiz et al, 2000) and, therefore, the activities and tasks included may also be different and have a different instant or importance. Consequently, it is not unusual for the software industry to demand methodologies, techniques and tools for controlling and managing this long and difficult stage, especially if we take into account the large number of legacy systems still being maintained (Briand et al, 1998). For these reasons, specific methodologies taking SMP into account are needed in order to achieve effective management of SM projects (Polo et al, 1999).

However, having said that, companies undertaking SM projects need a lot more than just a specific methodology. For software development projects, Cockburn (2000) included the following elements as minimum requirements: people, roles, skills, team, tools, techniques, processes, activities, milestones, work products, standards, quality measures and team values. With the same objective and integrating these aspects and others in a more general framework, the "*MANTIS Big-E environment*" aims to define and construct an integrated environment for management of SM projects (Ruiz et al, 2001a). By using the nomenclature "*Big-E environment*" our intention is to emphasize the idea that MANTIS is broader than the concepts of:

- *Methodology* in its usual sense, that is to say, a series of related methods and techniques.
- *Software Engineering Environment* (SEE), that is to say, a collection of software tools used to support software engineering activities.

We believe that the principal advantage of MANTIS is that it integrates practically all the aspects which must be taken into account for directing, controlling and managing SM projects into one conceptual framework. This advantage is built upon the following features:

- a conceptual architecture that facilitates working with the significant complexity inherent in the management of SM projects;
- the integration of methods and techniques that have been specially developed for SMP (such as the MANTEMA technology) or adapted from the development process (Polo et al, 1999);
- the integration of horizontal and vertical tools by means of orientation to processes, based on the proposal "Process Sensitive Software Engineering Environment" (PSEE), (Derniame et al, 1999), and the use of standards for storage and interchange of data and metadata like "XML Metadata Interchange" of OMG (1999); and
- a processes metamodel based on the international standards of ISO (12207 and 14764).

The following paragraphs will describe the conceptual architecture of MANTIS. To this aim we will speak about the conceptual levels in the following section. In section 3 we focus on the level of the metamodel and we present and we present a tool to store and to manage the models and metamodels of all the conceptual levels. Lastly, in section 4, we draw some conclusions.

2. Conceptual levels in MANTIS

In MANTIS we have defined 4 conceptual levels that are based on the MOF (*Meta-Object Facility*) standard for object oriented modelling proposed by the Object Management Group, OMG (2000). In table 1 we can see these 4 levels of the MOF architecture and its adaptation to MANTIS.

Level	MOF	MANTIS
M3	MOF-model (Meta-metamodel)	MOF-model of SMP
M2	Meta-model	SMP generic metamodel
M1	Model	MANTEMA & others techniques (SMP concrete model)
M0	Data	Instances of SMP (real-world concrete software maintenance projects)

Table 1. Conceptual levels in MOF & MANTIS.

Examples of real and specific software maintenance projects with time and cost restrictions are found in the level M0. The data handled at this level are instances of the concepts defined at the higher level M1. The specific model that we use at level M1 is based on the MANTEMA methodology and a group of techniques adapted to the special characteristics of maintenance: effort estimation, risk estimation, process auditing (Ruiz et al, 2000). Level M2 corresponds to the SMP generic metamodel[1] which we will discuss after. For example, the generic concept of "Maintenance Activity" used in M2 is present in the activities "Corrective Maintenance Activity" in M1 and these in turn appear in level M0 as "Corrective maintenance activity n° 36 of the PATON project" (see fig. 1).

In the last conceptual level of MANTIS, M3, the SMP generic metamodel is represented in a MOF-model. A MOF-model is composed basically of two types of objects: MOF-class and MOF-association[2]. Consequently, all the concepts represented in level M2 are now considered instances of MOF-class or MOF-association. For example, "Maintenance Activity", "Actor", or "Artefact" will be instances of MOF-class; and "Activity use Resource" or "Artefact is input of Activity" are instances of MOF-association. An MOF-model can be represented by UML diagrams or by MODL language (Meta Object Definition Language) but in order for it to be used automatically and to be portable amongst tools in a SEE, which is what interests MANTIS, it is much better to represent it by using one of the metadata exchange standards. For this reason, in MANTIS we use XMI (OMG, 1999) - a "dialect" of XML - for storing the metamodels.

The inclusion of level M3 allows us to work with different versions of SMP metamodel, which is a requirement in order to be able to manage the process improvement.

[1] Some authors do not distinguish clearly in their nomenclature between model and metamodel but refer to both that corresponding to level M2 and that of level M1 as model.

[2] As far as we are concerned these are the principal objects, although others also exist (packages for reuse, data types ...).

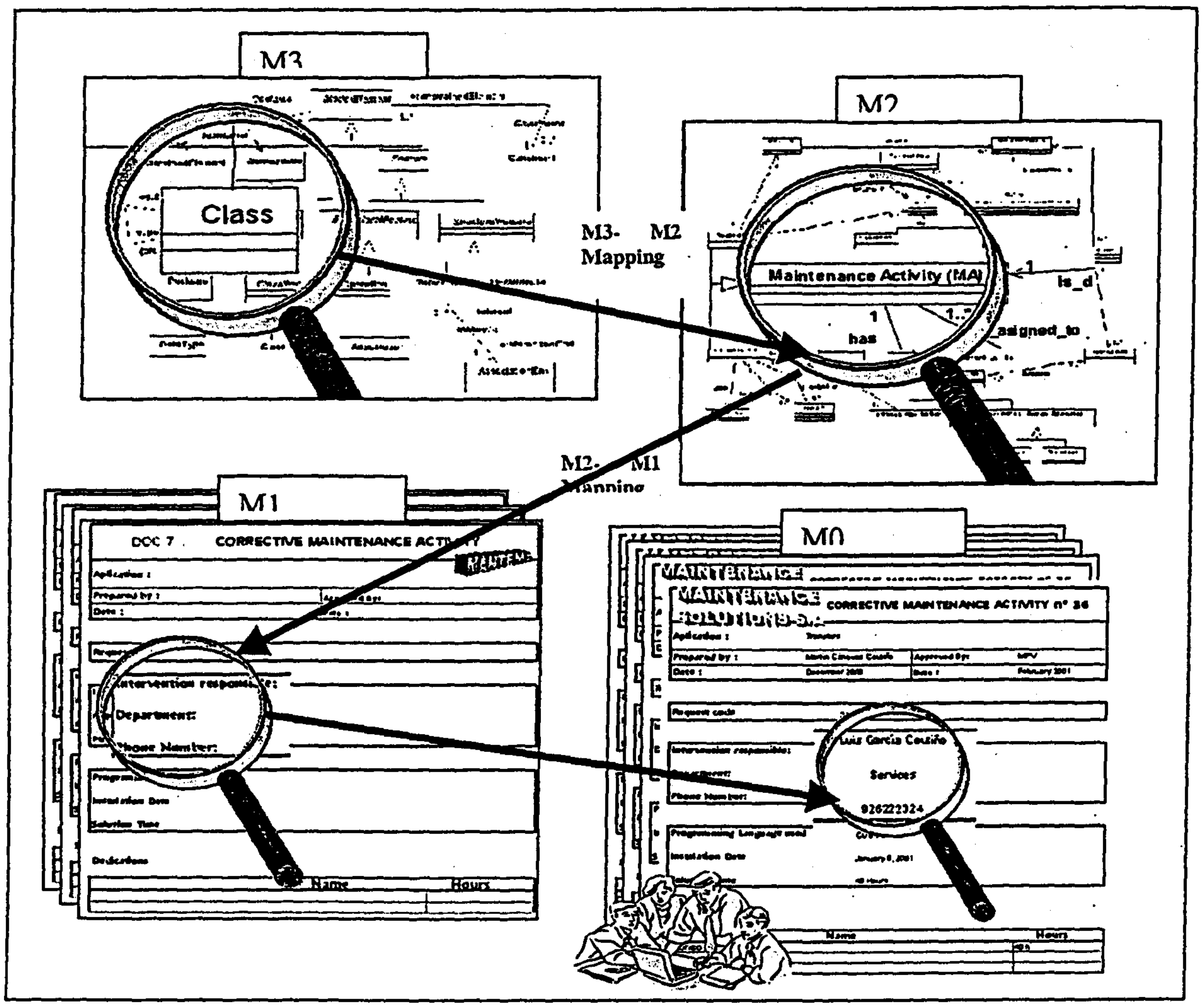


Figure 1 Example of conceptual levels in MANTIS

The SMP generic metamodel that we use in level M2 of MANTIS is based on the informal ontology proposal for software maintenance formulated by Kitchenham et al (1999). This global metamodel is comprised of the union of four partial metamodels focused on the "Activities", the "Products", the "Peopleware" and the "Process Organization". In short, the metamodel represents:

a) how to organize activities for maintaining software,

b) what kinds of activities they may be,

c) how the methods and tools (either specific or shared with the development process) can be applied to the activities, and

d) what skills and roles are necessary in order to carry out the activities.

We have extended this SMP generic metamodel by adding a fifth partial metamodel, called the "*Metamodel of the Workflows*" (see fig. 2), which incorporates aspects corresponding to the two following issues:

1) Specification of the structure of activities and subactivities and their relations.
2) Information for support, administration and control of the process enactment.

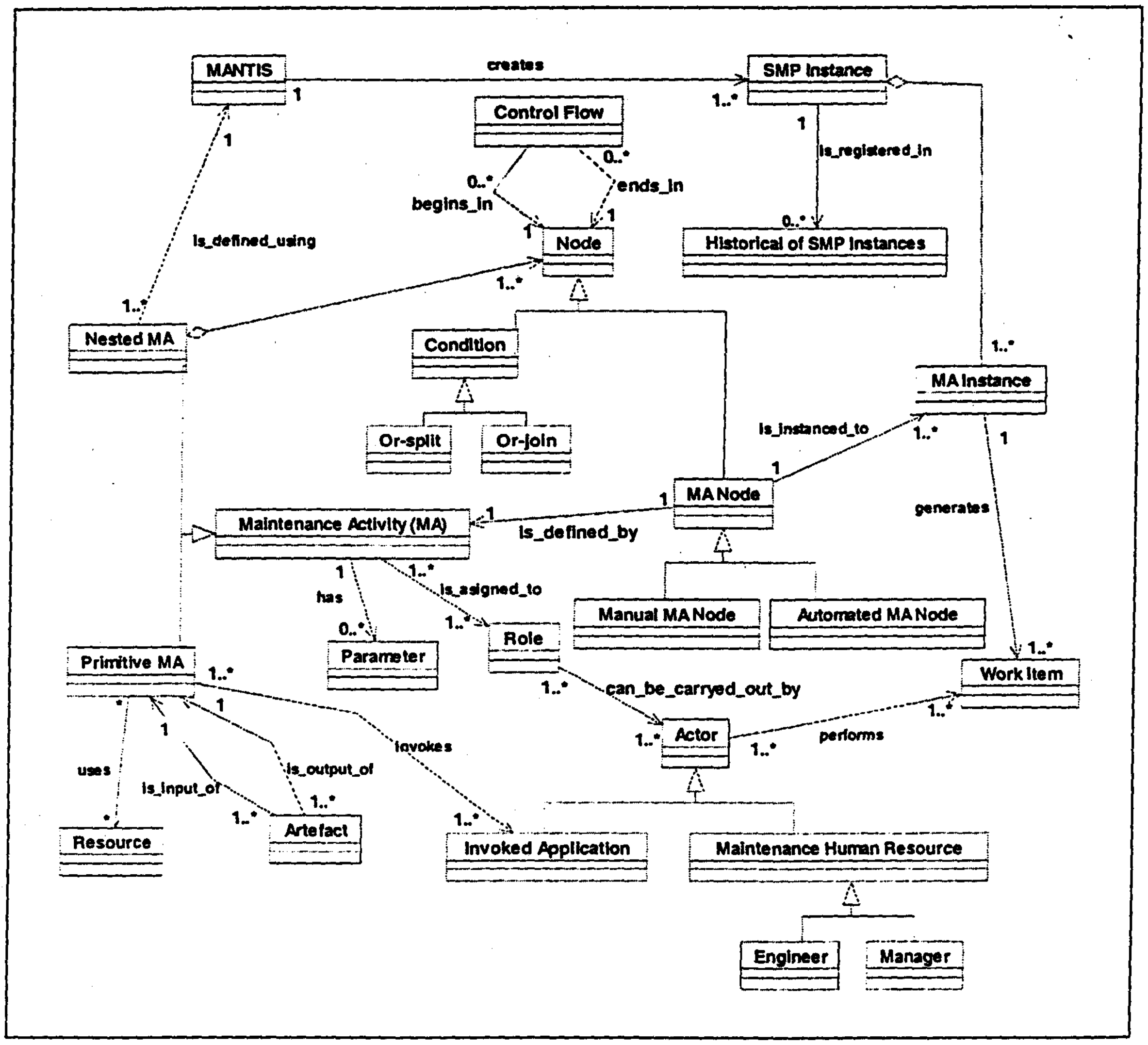


Figure 2. The metamodel of the Workflows.

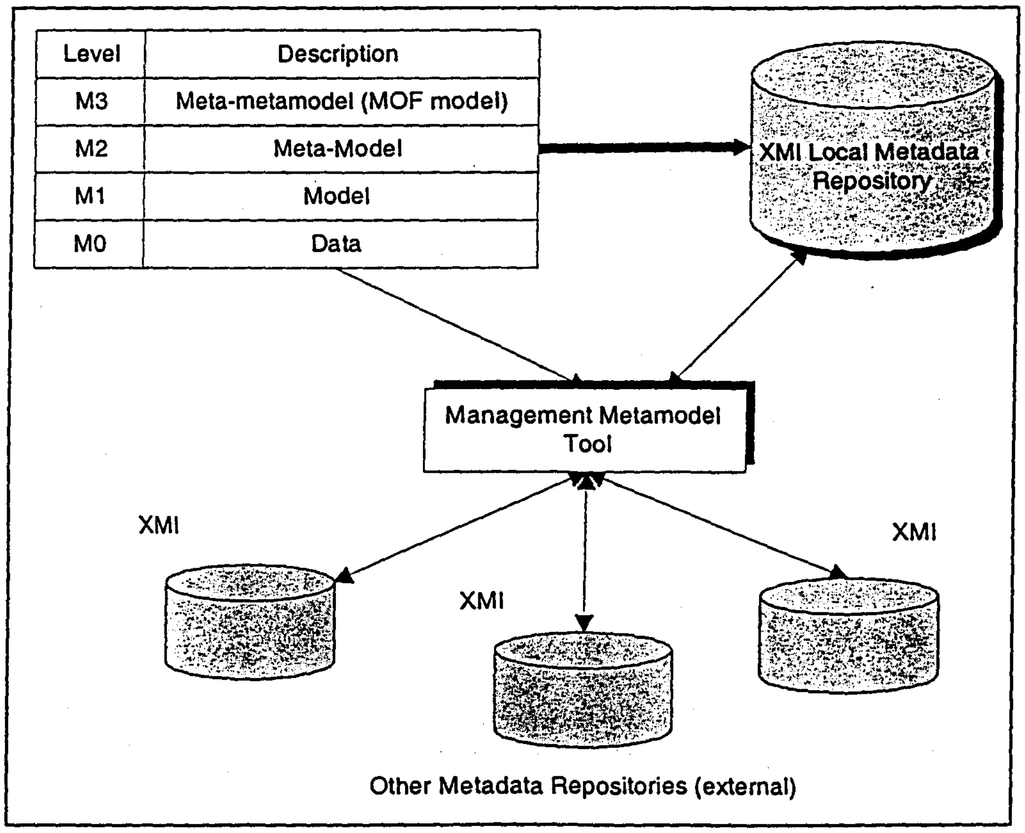
Recently, some authors (Ocampo and Botella, 1998) have suggested the possibility of using workflows for dealing with software processes, taking advantage of the similarity that exists between the two technologies. For it we believe reasonable to consider that workflow technology will be able to contribute a broader perspective to SMP in line with the objectives of our "MANTIS Big-E Environment".

3. Working with all the conceptual levels

By incorporating the M3 level and using standards for metamodelling (MOF) and for metadata interchange (XMI) we can achieve a flexible as possible environment for defining and sharing models and metamodels. MANTIS-Metamod is a software tool for this aim (Ruiz et al, 2001b).

As shown in figure 4 the tool maintains and manages a local repository of metadata for the storage of the MOF models. Basically this diagram represents the two services provided by the storage layer:

- Storage of MOF models in XMI format in order to facilitate their exportation.
- Importation of process metamodels (defined according to MOF levels), using the XMI format as a base in such a way that useable MOF models can be taken advantage of for the representation of software processes independent of the platform and the tool with which they have been defined.



Level	Description
M3	Meta-metamodel (MOF model)
M2	Meta-Model
M1	Model
M0	Data

XMI Local Metadata Repository
Management Metamodel Tool
XMI
XMI
XMI
Other Metadata Repositories (external)

Figure 3. Structure of MANTIS-Metamod.

4. Conclusions

In order to manage any software process - such as the SMP - in an integrated and structured way, it is useful to consider different levels of abstraction, both for modeling the process and managing the different instances of execution (specific projects).

In this paper we have presented the importance of establish a conceptual level architecture for software process metamodeling. As example of this we have considered MANTIS, an integrated environment for the management of the SM process. In MANTIS this integration is based on the use of a common terminology and on the abstraction mechanism provided by the architecture of conceptual levels described by the MOF standard.

In order to provide automated support for the software process metamodeling, and in particular for the SMP, we have made MANTIS-Metamod, a tool for the description, importation and exportation of software process metamodels. With this component of MANTIS our aim is to take advantage of the proven benefits that can be obtained by using metamodels for the integral maintenance of software processes.

Acknowledgements: This work has been undertaken in collaboration with the company Atos ODS subsidized by the MANTIS and MPM projects. MANTIS has been partially supported by the European Union and CICYT-Spain (1FD97-1608TIC). MPM has been partially supported by the Science and Technology Ministry of Spain (FIT-070000-2000-307).

References

Briand, L., Kim, Y., Melo, W., Seaman, C. & Basili, R. (1998): "Q-MOPP: Qualitative Evaluation of Maintenance Organizations, Processes and Products". *Journal of Software Maintenance*, n 10, pp. 249-278, 1998.

Cockburn, A. (2000): "Selecting a Project's Methodology". *IEEE Software*, July/August 2000, pp. 64-71.

Derniame, J.C., Kaba, B.A., and Wastell, D. (1999): "The Software Process: Modelling and Technology". In *Software Process: Principles, Methodology and Technology*. LNCS 1500, Springer-Verlag.

Kitchenham, B.A.; Travassos, G.H.; Mayrhauser, A. von; Niessink, F.; Schneidewind, N.F.; Singer, J.; Takada, S.; Vehvilainen, R. and Yang, H. (1999): "Towards an Ontology of Software Maintenance". *Journal of Software Maintenance: Research and Practice.* 11, 365-389.

Ocampo, C. and Botella, P. (1998): "Some Reflections on Applying Workflow Technology to Software Processes". Technical Report-LSI-98-5-R, UPC, Barcelona (Spain).

OMG (1999): XML Metadata Interchange (XMI), v. 1.1, Oct-1999.

OMG (2000): Meta Object Facility (MOF) Specification, v. 1.3 RTF, Mar-2000. In http://www.omg.org.

Polo, M., Piattini, M., Ruiz, F. and Calero, C. (1999): "MANTEMA: A Complete Rigourous Methodology for Supporting Maintenance based on the ISO/IEC 12207 Standard". *Third Euromicro Conference on Software Maintenance and Reengineering* (CSMR'99). IEEE Computer Society Press, Amsterdam (Netherland), pp. 178-181.

Ruiz, F.; Piattini, M.; Polo, M.; and Calero, C. (2000): "Audit of Software Maintenance". In "*Auditing Information Systems*". Idea Group Publishing, USA.

Ruiz, F., Piattini, M. and Polo, M. (2001a): "An Integrated Environment for Managing Software Maintenance Projects". In *World Class IT Service Management Guide* (2nd edition), Addison-Wesley (Juny-2001).

Ruiz, F., Garcia, F., Marquez, L., Piattini, M., and Polo, M. (2001b): "Tool based on MOF for software process metamodelling". *Business Information Technology Management Conference*. Cairo (Egypt).