Using Metamodels and Workflows in a Software Maintenance Environment [1]

Francisco Ruiz González, Mario Piattini Velthuis, Macario Polo Usaola
Departamento de Informática, Grupo Alarcos
Escuela Superior de Informática, Universidad de Castilla-La Mancha
Ronda de Calatrava, s/n. 13071, Ciudad Real (Spain)
{fruiz|mpiattin|mpolo}@inf-cr.uclm.es

Abstract

The objective of Software Engineering Environments (SEE) is to allow the integrated and automatic management and control of a specific process or group of processes of the software (ISO/IEC, 2000). In broader terms, the MANTIS project aims to define and construct an integral environment for the management of Software Maintenance Process (SMP). Due to the large number of different aspects that have to be considered, we have defined an architecture with 4 conceptual levels. Each of these levels incorporates concepts at a specific level of abstraction and generality. We present a proposal of a metamodel for the SMP based on the ontology formulated by Kitchenham et al.(1999) and in the Workflow Reference Model of the Workflow Management Coalition (WFMC, 1995). In so doing our aim is to incorporate in the said ontology, the aspects of process enaction that the workflow technology resolves in what we consider to be a satisfactory fashion.

Keywords:

Software Engineering, Metamodels, WorkFlows, Software Engineering Environments, Software Maintenance Management.

1. Introduction

Many studies have demonstrated that the majority of the overall expenses incurred by a software product throughout its lifecycle occur during the maintenance stage (Pigoski 1996) and that the characteristics specific to this stage which differentiate it clearly from the development stage, make it very useful to have specific methods, techniques and tools at one's disposal. For these reasons an effort has been made over recent years to deal with the problem of software maintenance taking into account its specific characteristics. Following this line there are specific maintenance methodologies such as MANTEMA (Polo et al. 1999) or proposals for the improvement of the SMP such as that of Niessink (2000). With the same objective and integrating these aspects and others in a more general framework, the MANTIS project aims to define and construct an integrated big-E environment for the management of the SMP. By using the nomenclature "*big-E Environment*" our intention is to emphasize the idea that MANTIS is broader than the concepts of:

[1] 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 Ministery of Spain (FIT-070000-2000-307).

- Methodology (in its usual sense), that is to say, a series of related methods and techniques), and
- Software Engineering Environment (SEE), that is to say, a collection of software tools used to support software engineering activities (ISO/IEC , 2000).

MANTIS includes the different aspects that must be taken into account when undertaking software maintenance projects. All these aspects are conceptually integrated in a similar way to that proposed by Cockburn (2000). The MANTIS big-E Environment for the integrated management of software maintenance projects aims to integrate the following aspects, amongst others:

- The people, with certain skills, carry out certain roles in the project, working together in different teams (groups of people).
- These people use methodologies to construct products that conform to certain standards (norms) and satisfy quality measurements (criteria). The processes must also satisfy quality criteria.
- The methodologies require certain skills and tools. The tools facilitate conforming with the standards.
- The teams participate in activities (included in methodologies) that belong to processes encompassed by the project. Each activity undertaken helps to reach a milestone which indicates the progress of the process.

The following paragraphs will describe the use of metamodels and workflow technology in MANTIS. To this aim we will speak about the general characteristics of MANTIS in the following section. Section 3 describes the conceptual levels that we have defined in order to be able to deal more easily with its complexity. In section 4 we focus on the level of the metamodel. Lastly, in section 5, we draw some conclusions and speak about several subjects for future work.

2. Main characteristics of MANTIS

For the design and development of MANTIS we have defined the following general requirements:

a) Tool Oriented: the services of MANTIS are supplied through software tools according to the philosophy of the integrated SEE. The integrated use of all these possible tools is advantageous as it increases productivity, reduces the possibility of errors and facilitates management, supervision and control. Although various proposals with these objectives in mind have been made[2] (PCTE, ECMA/NIST, STARS…) and their generic contributions and architectural models can be applied to the SMP, none of them take into account the special characteristics of software maintenance.

b) Process Driven: assuring consistency with the processes and activities described in the standards ISO 12207, referring to the software lifecycle processes (ISO/IEC,1995) and 14764, referring to the SMP (ISO/IEC, 1998). The orientation to processes is a very important vehicle for integration (Randall and Ett, 1995). In MANTIS two types of integration can be dealt with simultaneously: that of the processes of the organization with the Big-E Environment and that of the tools and artefacts with the processes. This orientation is known as *Process Sensitive Software Engineering Environment* (PSEE).

[2] PCTE "Portable Common Tool Environment" (Long & Morris, 1993),
"The Reference Model for Frameworks of SEE" (ECMA/NIST, 1993), and
STARS "Software Technology for Adaptable, Reliable Systems" (USAF, 1996).

c) Scalability: adaptation to the necessities of organizations of any size. The tailoring process of ISO 12207 is used for this (Polo et al, 1999b).

d) Selection of a specific methodology for the SMP, in this case MANTEMA (Polo et al, 1999).

e) Support of the technical work (more precisely the SMP) and the organizational and management activities (management processes). We consider that like a manufacturing project, a software maintenance project really consists of two types of processes: the SMP and the management processes that provide the resources needed by the SMP and control it.

f) Use the technology based on the XML standard (W3C, 2000) and its derivatives (XMI,…) for the data and metadata repository.

3. Conceptual Levels in MANTIS

An important principle of modern software engineering is the separation of a system in encapsulation layers which can mostly be specified, designed and constructed independently. Following this philosophy, 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 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) etc. Level M2 corresponds to the SMP metamodel[3] which we will discuss in detail in the following section. 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 nº 36 of the PATON project*" (see fig. 1).

In the last conceptual level of MANTIS, M3, the SMP metamodel is represented in a MOF-model. A MOF-model is composed basically of two types of objects: MOF-class and MOF-association[4]. 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

[3] 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.

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

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.

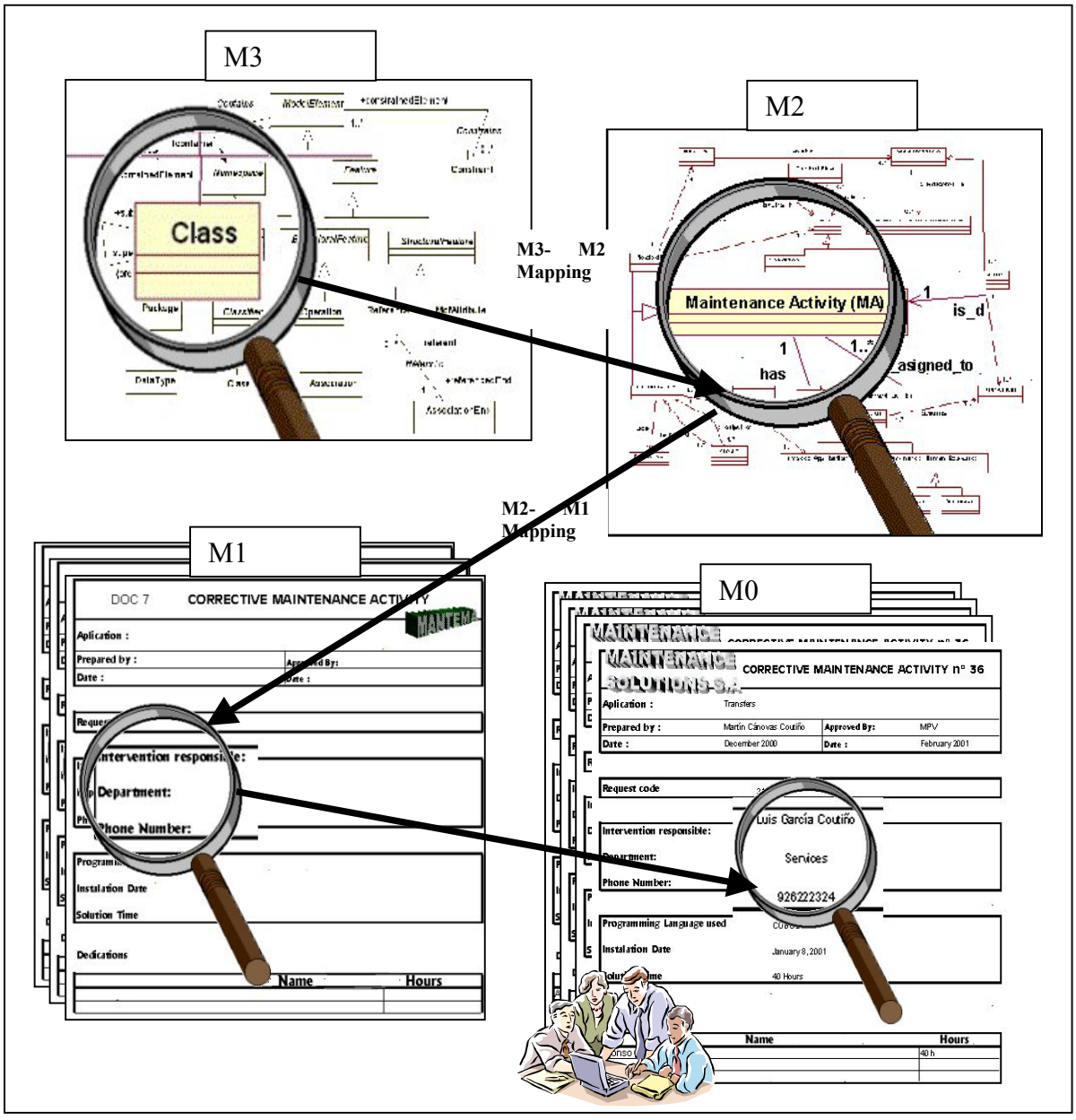

Figure 1. Example of conceptual levels in MANTIS.

By incorporating this last conceptual 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. 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.

4. Workflows for Software Maintenance Process Metamodelling

The SMP 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:

- how to organize activities for maintaining software,
- what kinds of activities they may be,
- how the methods and tools (either specific or shared with the development process) can be applied to the activities, and
- what skills and roles are necessary in order to carry out the activities.

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. The usefulness of Workflow Management Systems (WFMS) in the automation of business processes has been clearly demonstrated and given that SMP can be considered as part of a wider business process[5], it is reasonable to consider that workflow technology will be able to contribute a broader perspective to SMP (which we could call Process Technology) in line with the objectives of our MANTIS Big-E Environment.

The previously mentioned reasons have led us to integrate the workflow technology in the SMP metamodel. In level M2 of MANTIS we have incorporated aspects of the *Workflow Reference Model* of the Workflow Management Coalition (WFMC, 1995) and aspects of other proposals of workflow metamodels (Liu et al,1999).
We have extended the SMP metamodel by adding a fifth partial metamodel, called the Workflow metamodel, which incorporates aspects corresponding to the two following issues:

a) Specification of the structure of activities and subactivities and their relations.

b) Information for support, administration and control of the process enactment.

[5] In both cases we refer to activities that make up the process, to artefacts used, modified or produced by these activities (products, documents or data), to people who participate carrying out a certain role, to tools used (for example, software applications), to aspects of collaboration in work groups, etc.

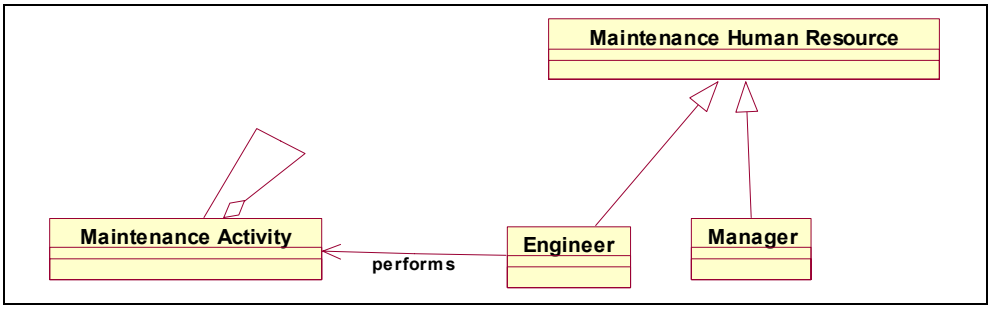

Figure 2. Part of SMP metamodel without activities detailed specification.

4.1 Activities Specification.

The central part of the SMP metamodel that represents the existing activities and their performers (figure 2) must be extended in order to show in more detail how it can broken down into simpler activities, their relationships and control flow and the possibility of automatic execution. In MANTIS, the resulting metamodel represents the following characteristics:

- Each *Maintenance Activity* (that is to say, its specification) and the node that represents it in the associated workflow (*MA Node*) is differentiated. The class *Maintenance Activity* contains properties that can be abstracted from an activity and which a result are independent of any specific process, that is to say, it includes the general properties of a group of activities of the same type.
- The possibility of automated enactment activities is contemplated, for which two specializations are created, called *Manual MA Node* and *Automated MA Node*. The only thing that differentiates these two specializations are the characteristics of the run-time phase. An *MA Node* may belong simultaneously to both the subclasses. This possibility allows the existence of so-called mixed activities which are activities that are carried out partially manually and partially automatically.
- In order to make it possible for certain activities to be executed either wholly or partially by automatically invoked external applications, the class *Actor* is specialized in *Invoked Application* (new) and *Maintenance Human Resource* (already existing).
- The class *Parameter* allows each activity to have several associated parameters. The purpose of these parameters is to manage the information that is received by or leaves an activity.
- Each activity can be assigned to several different roles. Each role can be carried out by several actors. The new class *Role* makes these associations possible.
- In order to represent the activities hierarchy, that is to say, to demonstrate that an activity can be comprised of other activities, and these activities, in turn, of others etc, we use a recursive representation. The class *Maintenance Activity* specializes in two new classes: *Primitive MA* for the atomic classes that do not include other simpler activities and *Nested MA* for the classes with a complex structure, meaning those with an associated workflow including other simpler activities. Moreover, the recursivity allows the use of the class *Nested MA* to represent the full specification of an SMP using the MANTIS tools (in a similar way to an WFMS).

- The management of the resources used in the activities and of the artefacts used or generated by them is undertaken at a primitive activity level. For this reason, the classes *Resource* and *Artefact* are associated with *Primitive MA* instead of with *Maintenance MA*. The primitive activities are also those which can invoke external applications.
- Lastly, it is necessary to use a workflow model to represent the internal structure of the nested activities. MANTIS uses the representation proposed by Sadiq and Orlowska (1999). The classes *Node*, *Control Flow*, *Condition*, *Or-split*, *Or-join* and *MA Node* can be used for this.

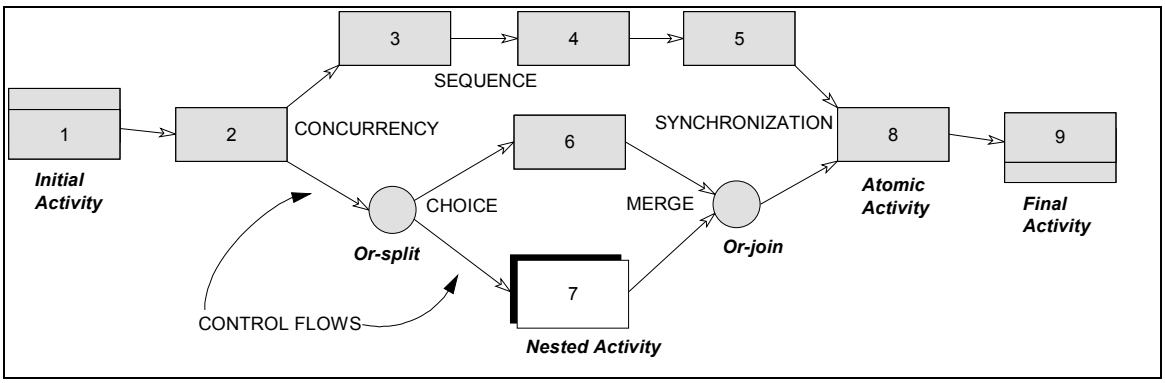

Figure 3. Nested activities representation using workflow structures.

According to the previous model (see fig. 3) a nested activity is represented by a graph whose nodes are activities (rectangles) or conditions (circles) interconnected by control flows (arrows). In MANTIS, the nested activities are represented using the *Node* and *Control Flow* classes of objects. *Node* is classified into two subclasses: *MA Node* (maintenance activity node) and *Condition*. A *Condition* is used to construct choice structures (*Or-Split* class) and merge structures (*Or-join* class). A *Control Flow* links two nodes in the graph.

4.2 Process Enactment Specification.

In order to include the aspects related with the *Run-Time* phase in the SMP metamodel, we must take the following into account:

- the tools of the MANTIS Big-E Environment play a similar role to a WFMS:
  a) they allow the definition of the SMP metamodel (using the *Nested MA* class in a recursive manner), and
  b) they allow the creation of SMP run-time instances (*SMP Instance* class).
- The SMP run-time instances must be recorded in a historical repository (*Historical of SMP Instances* class).
- Each SMP run-time instance is made up of run-time activity instances (*MA Instance* class) which generate work items (*Work Item* class). A *Work Item* is the smallest unit of a job which is undertaken and that is controlled, managed and assigned to an actor to carry it out. This method for controlling the execution of the jobs is based on the norm of the Project Management Institute (PMI, 2000).

4.3 Integration of Design-Time and Run-Time Aspects.

The previous subsections deal with aspects that correspond to the phases of *Design-Time* and *Run-Time* in Workflows. Their integration in a single metamodel is represented by two associations:

a) The association "*is-instanced-to*" between the classes *MA Node* and *MA Instance* for representing the run-time instances generated in each SMP run-time instance. The same activity may have several different run-time instances for the same SMP run-time instance (for example, those activities included within an iteration structure).

b) The association "*performs*" between the classes *Actor* and *Work Item* to determine which actors (humans or software applications) should carry out each work item.

The SMP metamodel of figure 2 is transformed into the diagram of figure 4 as a result of the considerations discussed in the three previous subsections.

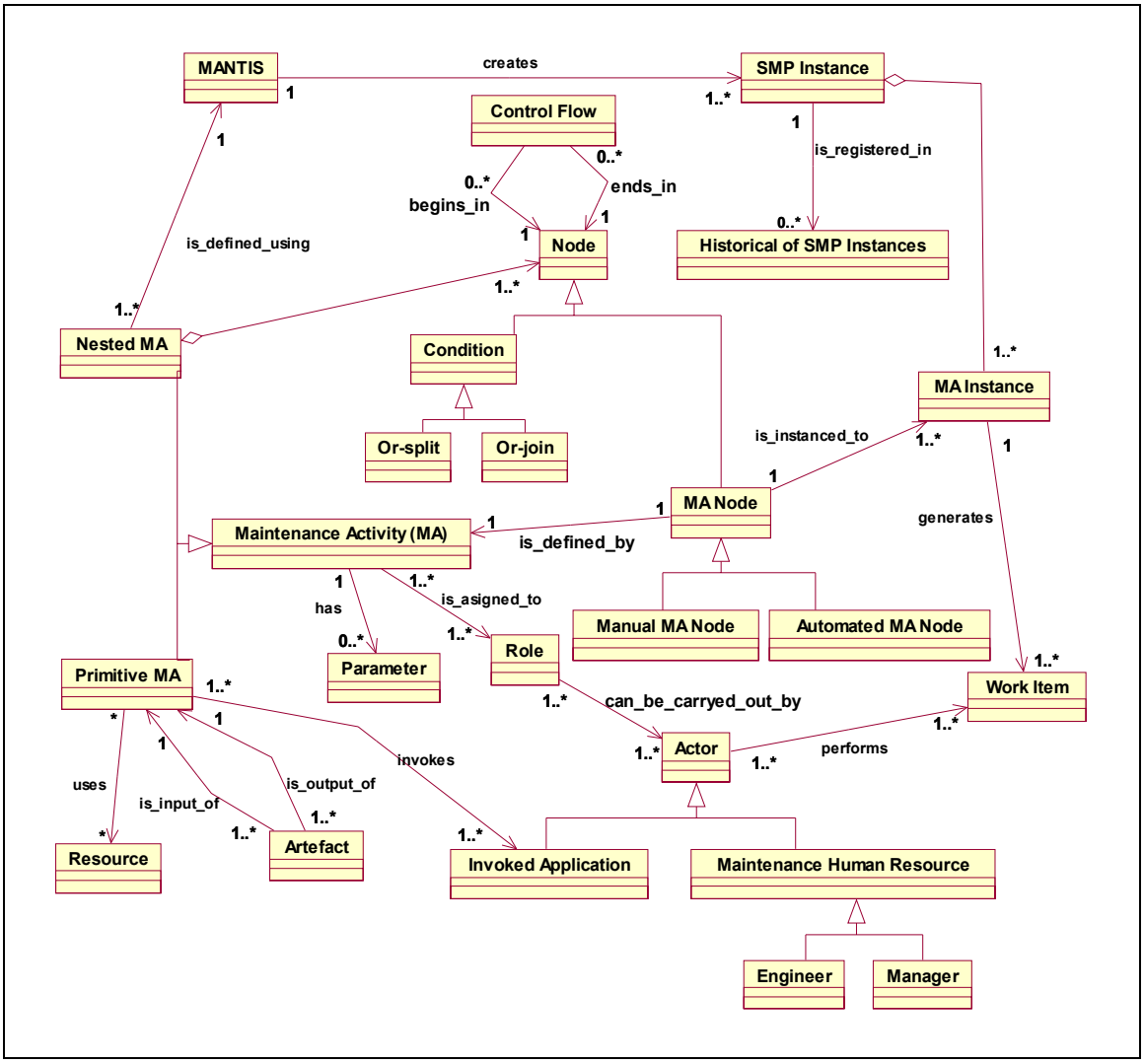

Figure 4. Extended SMP metamodel.

In the previous figure, the elements related to the run-time phase have been situated on the right.

5. Conclusions and future works

We have presented a metamodel for the SMP that integrates the ontology of Kitchenham et al, - the most important proposal of software maintenance conceptual modelling – with workflow technology. The principle advantages of this approximation are:

a) Integration between the static aspects (represented fundamentally in the ontology) and the dynamic aspects (using the idea of workflow enactment).

b) Integration between manual activities and those carried out automatically or semiautomatically.

c) Representation of the internal structure of the activities through workflow diagrams.

d) Incorporation of the principles of Process Sensitive Software Engineering Environments.

This proposal takes advantage of two technological fields, the software process and the workflow, which up until now have converged very little (Conradi and Jaccheri, 1999).

In order to deal with the great complexity inherent in the integral management of software maintenance projects, we propose a 4-level conceptual architecture with the following characteristics:

- the possibility of working with different models and metamodels (models of models),
- easier integration of all the aspects of the Big-E Environment, and
- the use of standards of metamodelling (MOF) and of metadata interchange (XMI).

With this architecture, we believe the construction of a Big-E Environment for better management of the maintenance projects of an organization, is more feasible.

The main future work is to carry out a global evaluation of the MANTIS environment and another specific of each tool. To approach this task we will follow the recommendations of the DESMET proposal (Kitchenham et to the, 1997), a method for evaluating software engineering methods and tools.

References:

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

Conradi, R. and Jaccheri, M.L. (eds.) (1999): Process Modelling Languages. In Derniame, J-C.; Kaba, B.A.; & Wastell, D. (Eds.); "*Software Process: Principles, Methodology and Technology*". LNCS 1500, Springer-Verlag, 1999, pp. 27-52.

ECMA/NIST (1993): *Reference Model for Frameworks of Software Engineering Environments*, 3rd edition. TR-55, June 1993. In http://www.ecma.ch.

ISO/IEC 12207 (1995): *Information Technology – Software Life Cycle Processes*.

ISO/IEC FDIS 14764 (1998): *Software Engineering - Software Maintenance* (draft), Dec-1998.

ISO/IEC JTC1/SC7/WG4 15940 working draft 5 (2000): *Information Technology – Software Engineering Environment Services*, Juny-2000.

Kitchenham, B.A.; Linkman, S.G. and Law, D. (1997): DESMET: A methodology for evaluating software engineering methods and tools. *IEE Computing & Control Journal*, June 1997, pp120-126.

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 (1999).

Liu, C.; Lin, X.; Zhou, X.; Orlowska, M. (1999): Building a Repository for Workflow Systems, Proceedings of the *31st International Conference on Technology of Object-Oriented Language and Systems*. IEEE Computer Society Press, 1999, pp. 348-357.

Long, F., Morris, E. (1993): *An Overview of PCTE: A Basis for a Portable Common Tool Environment*. Technical Report CMU/SEI-93-TR-1, 1993.

Niessink, F. (2000): *Perspectives on Improving Software Maintenance*. PhD Thesis, Vrije Universiteit, Netherland. In http://www.opencontent.org/openpub/, 2000.

Ocampo, C.; Botella, P. (1998): *Some Reflections on Applying Workflow Technology to Software Processes*. TR-LSI-98-5-R, UPC, Barcelona (Spain), 1998.

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.

Pigoski, T.M. (1996): *Practical Software Maintenance. Best Practices for Managing your Investment*. Ed. John Wiley & Sons, USA 1996.

PMI (2000): *A Guide to the Project Management Body of Knowledge 2000 edition*, Project Management Institute Communications, USA 2000.

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), 1999, pp. 178-181.

Polo, M., Piattini, M., Ruiz, F. and Calero, C. (1999b): Using the ISO/IEC Tailoring Process for defining a Maintenance Process. *IEEE Conference on Standardization and Innovation on Information Technology*. IEEE Computer Society Press, Aachen (Germany) 1999, pp. 205-210..

Randall, R. and Ett, W. (1995): Using Process to Integrate Software Engineering Environments. *Proceedings of the Software Technology Conference*, Salt Lake City, USA, 1995. In http://www.asset.com/stars/loral/pubs/stc95/psee95/psee.htm.

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

Sadiq, W and Orlowska, M.E. (1999): On Capturing Process Requirements of Workflow Based Business Information Systems. *3rd International Conference on Business Information Systems*, Poznan, Poland, 1999, pp. 195-209.

USAF (1996): *Software Engineering Environment Integration Process. Summary-level Definition*. US Air Force Informal TR STARS-PV03-A032/001/00, 1996.

W3C (2000): *Extensible Markup Language (XML) 1.0* (second edition), oct-2000. In http://www.w3.org/.

WfMC (1995): TC00-1003 1.1: Workflow Management Coalition. *The Workflow Reference Model*, Jan-1995.