An XMI-Based Repository for Software Process Meta-modeling

Francisco Ruiz, Mario Piattini, Félix García, and Macario Polo

Alarcos Research Group, University of Castilla-La Mancha
13071 Ciudad Real, Spain
{Francisco.RuizG,Mario.Piattini,Felix.Garcia,
Macario.Polo}@uclm.es
http://alarcos.inf-cr.uclm.es/english/

Abstract. In order to be able to work correctly with all the concepts handled in software process improvement it is useful to establish different abstraction levels that help to manage the complexity. Correct use of all the data and metadata (models and meta-models) handled in the different abstraction levels is necessary. This paper proposes a tool, based on MOF (Meta-Object Facility) conceptual architecture, for the management of these models and meta-models, that are stored in a repository in the form of XMI (XML Metadata Interchange) documents. This tool can be used as an integrated vertical component in other horizontal tools oriented to software process improvement and management. As example, we present its vertical integration with MANTIS, an environment for software maintenance management.

1 Introduction

In the field of software engineering, achieving high quality products is a common objective. A factor which is fundamental to and greatly influences the final quality of a software product is precisely the process which has been followed in order to develop or maintain the product. As a result, in a Software Engineering Environment (SEE) it is important that all the software processes should be managed in an integrated fashion, taking into account both the way in which software processes are developed (defining the required methodologies and tools) and their execution (especially its control).

A software process consists of a set of concurrent and cooperative activities that are related to the development and maintenance of the software as well as to the management of the project and the quality of the product. Software processes are inherently complex as they involve a lot of people, with different responsibilities and skills, and they produce or modify a wide range of elements [1]. In order to manage and improve software processes, it is necessary to establish abstraction levels that enable us to reduce this complexity, dealing satisfactorily with all their different aspects. Metadata constitute the key element for the integration of all the concepts involved in software

M. Oivo and S. Komi-Sirviö (Eds.): PROFES 2002, LNCS 2559, pp. 546-558, 2002.

process management. Metadata are descriptive information about the structure and meaning of the data, and of the applications and processes that manipulate them [6]. When different abstraction levels are defined (multilevel conceptual architectures), correct management of the data and metadata and the relationship between these becomes an essential aspect. All the information carried by the data and metadata must be stored in repositories in order to deal with the diversity of the data sources.

The aim of this paper is to present RepManager, a software component for managing metadata repositories that deals with all the information necessary for the managing and improving of software processes. In the following paragraphs we will describe the functions carried out by the RepManager tool, and explain the importance of each of them. We will then look at a practical application of this component in its integration with MANTIS [16], a SEE oriented to software maintenance management. At last, we will comment some related works, which are the future lines of this work, and the main conclusions.

2 Description of the Repository Manager

The first consideration to be taken into account when designing a repository is in which format the data and metadata are going to be stored. In order to have an open format for the storage of the data and meta-data at our disposal, the RepManager tool uses *XML Metadata Interchange* (XMI), [9]. It is essential for the repository manager to guarantee the possibility of interchange between the models and meta-models defined, so that both the meta-models defined with the tool and the meta-models defined with other tools -that support XMI- are usable. The principal objective of XMI is to facilitate the interchange of metadata between modeling tools and between metadata tools and repositories in distributed heterogeneous environments. XMI constitutes the integrating element for metadata originating from different sources, as it represents a common specification. Therefore, to a great extent the RepManager component provides a tool with support enabling it to interchange metadata with other repositories or tools that support XMI.

Another important aspect that should be taken into consideration is that the information of the repository should be accessible to all those that need it. For this reason, the metadata repository manager is a specialized software component which is designed to provide the necessary infrastructure and support for the storage of interrelated information components [7]. Therefore the functions of RepManager are similar to those provided by a database management system, but with the special characteristic that the information is stored as XMI documents. These functions will be supported via a set of calls to the system and are basically (see figure 1):

1. *Storage* of models and meta-models defined in the tool in a local metadata repository represented in XMI (for exportation).
2. *Importation/exportation* of models and meta-models.

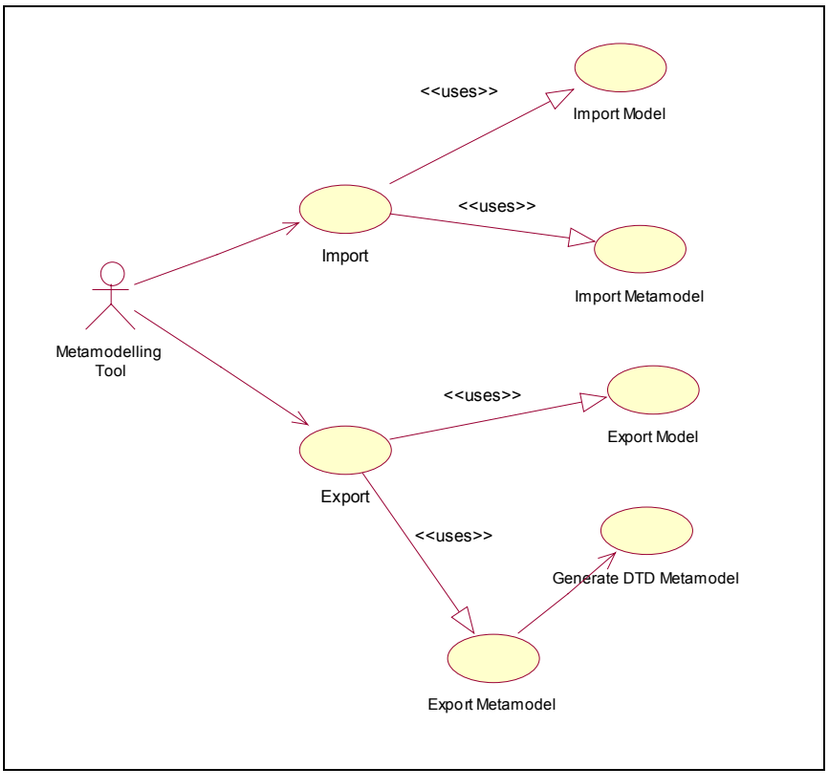

Fig. 1. RepManager imports/exports models and meta-models (UML use case diagram)

In order to gather the information on the models (at the different abstraction levels) that has been specified by the users or is contained in an XMI document, a class hierarchy has been defined that serves as a communication link between the control layer and the storage layer of RepManager. This class hierarchy is based on the specification of XMI, which establishes a series of production rules for the transformation of the meta-models into correct XML documents for the interchange. The storage function in the RepManager tool is supported by a specific service that generates the XMI document and the *Document Type Definition* (DTD) that represents it starting from a meta-model with its corresponding structure. When carrying out this operation the RepManager tool must guarantee that the updates are coherent with the syntax specified in the scheme associated with each type of XMI document.

As far as loading the meta-models is concerned RepManager provides another specific service which receives the document representing the meta-model as input and represents the information it contains according to the previously mentioned class hierarchy.

With the aim of managing the complexity, RepManager use 4 conceptual levels which are based on the MOF standard [8]. MOF is „a model for specifying, constructing, managing, interchanging and integrating metadata in software systems thus

allowing the flexible integration of the systems“ [8]. MOF describes an abstract modeling language aligned with the UML kernel. In table 1 we can see these four levels of the MOF architecture and their adaptation to RepManager.

Real specific software projects (with time, cost and resources restrictions) are found in level M0. The data dealt with at this level are instances of the concepts defined in the upper level M1. The specific models in level M1 represent the methods and techniques specific for each application domain. Level M2 corresponds to the software process (SP) generic meta-model. In M3, the last conceptual level, the SP meta-model of M2 level is represented as an MOF-model. An MOF-model basically comprises two types of objects: MOF-class and MOF-association (from our point of view these are the principal objects, although others do exist: packages for reuse purposes, types of data, etc.). Consequently, all the concepts represented in the level M2 are now considered examples of MOF-class and MOF-association. For example „Activity“, „Resource“ or „Artefact“ are examples of MOF-class and „Activity uses Resource“ or „Artefact is input of Activity“ are examples of MOF-association.

In figure 2 we can see the basic operation scheme of the RepManager tool with its principal functions. As can be seen in this figure, RepManager provides the CASE tools for meta-modeling with two basic services: the storage and importation of meta-models. The CASE tool for meta-modeling calls up these services from its control layer and provides RepManager with all the necessary information (contained in a set of objects grouped in the specified class hierarchy) for the generation of the XMI documents that represent the correspondences between the different levels and the DTD that represent each type of XMI document generated. For the generation of XMI documents and their associated schemes the RepManager tool uses the generation rules specified in the XMI standard. Finally, the manipulation of the repository, that is to say, the storage and retrieval of information at document level, is carried out using the Document Object Model (DOM) [18]. This standard provides a collection of classes that represent the hierarchic structure on which all XML documents are based. The model provides classes that represent documents, nodes, list of nodes, etc., with the necessary properties and methods for the construction and editing of XML documents.

These services enable the CASE tools for meta-modeling to store the meta-models and models correctly by means of using a simple set of calls which constitute the repository interface.

Table 1. Correspondences between the conceptual levels of MOF and RepManager

Level	MOF	RepManager	Examples
M3	MOF-model (meta-meta-model)	MOF-model	MOF-class
M2	Meta-model	Generic SP meta-model	Activity
M1	Model	SP models (concrete methods and techniques)	Codification
M0	Data	SP instances	Codification of the PXB module

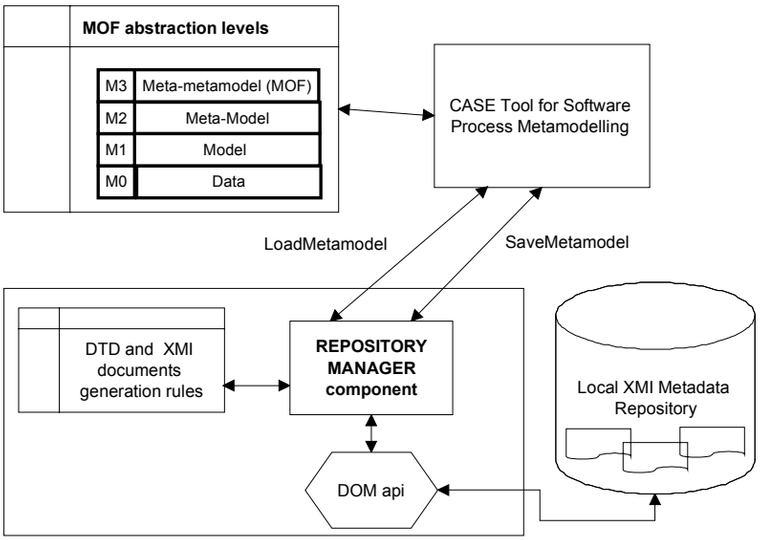

Fig. 2. RepManager interacts with other MOF-based tools

In the following sections we present a specific application of RepManager: the management of the repository of MANTIS, a SEE specially designed for the Software Maintenance Process (SMP) improvement and management.

3 One Use Case: MANTIS Environment

Software Maintenance (SM) represents the stage in the lifecycle of a software product that consumes most resources [11], and taking into account its special characteristics that clearly differentiate it from the development stage, it is very useful to have specific methods, techniques and tools at ones disposal. Moreover, to the improvement of this process, it is highly convenient to define and construct an environment for the integral management of the SM projects, due to the complex nature of this type of projects (as a result of the size and complexity of the product to be maintained and the difficulty of the task to be undertaken).

The MANTIS project aims to define and construct an integrated Environment for the management of SM projects [16]. By using the term Environment (with a capital „E“) MANTIS is considered as a broader concept than the concepts of:

- *methodology* (in the usual sense, meaning a series of related methods or techniques); and
- *software engineering environment*, meaning a collection of software tools used to support software engineering activities [4].

MANTIS includes different aspects that must be taken into account when undertaking SM projects. For SMP management and improvement, MANTIS integrates the following:

- the *personnel* (with specific skills and specific roles to carry out in the project).
- the *techniques* (methodologies) used by these persons.
- the *tools* (that help to comply with the standards).
- the *activities* (in which the teams participate and which help important objectives to be reached).

The adaptation of the conceptual architecture showed in table 1 to this application domain is summarized next. In level M1, the MANTEMA methodology [12] and a set of techniques adapted to the special characteristics of SM (effort estimation, risk estimation, process improvement [13], audit [15], etc…) are proposed. Our generic SMP meta-model of M2 level is based on the informal ontology for SM proposed by Kitchenham et al [5], the „*Workflow Reference Model*" proposed by the Workflow Management Coalition [21], and the schema for software process modeling and software measurement proposed by Becker-Kornstaedt and Webby [1]. For example, the generic object type of „Maintenance Activity" used in M2 is instanced in the activity types "Analysis of the Modification Request" or „Urgent Corrective Intervention" in M1, and these in turn, in instances of level M0 as „Intervention nº 36 in the project PATON".

3.1 MANTIS-Metamod

MANTIS-Metamod is a tool for the meta-modeling of software processes in general, although constructed with the SM in mind (by means of its integration in the MANTIS environment) [16]. This tool can be integrated into horizontal tools based on the Meta Object Facility (MOF) standard. With MANTIS-Metamod the modeler can specify models and meta-models at the different levels of the MOF architecture.

The meta-models and models defined for the user by means of MANTIS-Metamod are validated and internally represented using the RepManager component. MANTIS-Metamod manages the correspondences between the levels M3-M2 and M2-M1. The correspondences between the levels M1 and M0 have not been included because they are related to the enactment of real projects instead of to process modeling.

The internal software architecture of this tool is based on a three layer model (presentation, control and storage) with the aim of reducing complexity and providing a certain degree of encapsulation. For the input of user data, the application comprises a meta-model administrator as its principal component and a windows system that permits a visual description of the classes that make up the models and meta-models (Package, Class, Data Type, Attribute, Operation, Reference, AssociationBegin, AssociationEnd and Constraint). In the meta-model administrator, as in the MOF model, the information is structured in a hierarchical fashion: a package contains classes and associations, a class contains attributes and operations, an association contains restrictions, etc. (see figure 3).

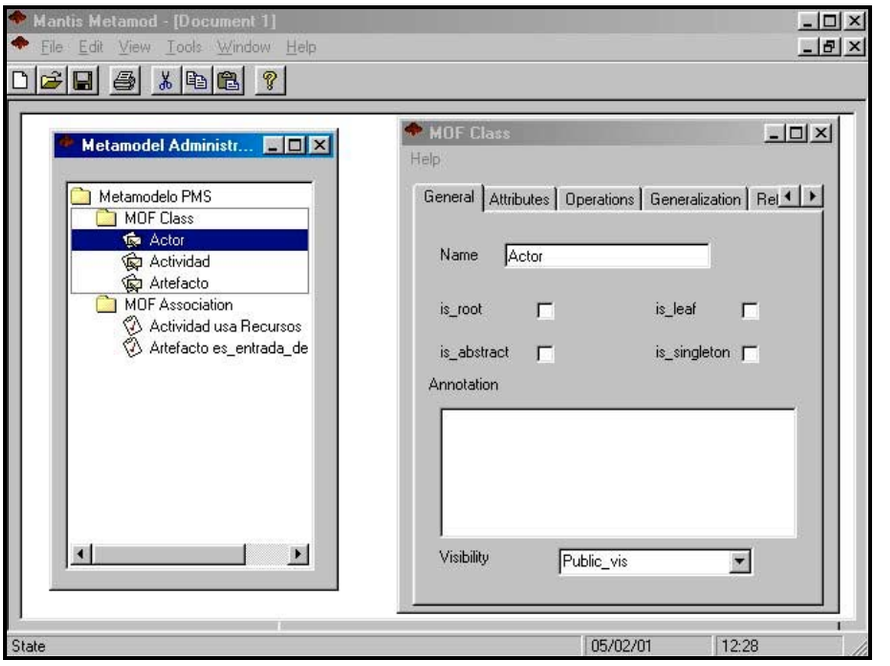

Fig. 3. Users can view and edit models and meta-models in an integrated way using MANTIS-Metamod

To complement this tool, other tools of visual modeling can be used. This functionality is possible owing to the fact, as already mentioned, that MANTIS-Metamod has the capacity to interchange meta-models and models in XMI format using the services of the RepManager tool. For example, you can use Rational Rose to draw the class diagrams of a model (level M1) and of a meta-model (level M2), and export both to MOF/XMI format. Using MANTIS-Metamod (and therefore, RepManager for the storage function) you will be able to edit both in a combined way; compare one with other, check the adaptation of the model to the meta-model, or visualize the model objects associated with a certain meta-model object.

We believe that the main utility of MANTIS-Metamod (and RepManager) is, in fact, to be able to manage and edit a model jointly with its associated meta-model.

4 Using RepManager in MANTIS

One of the objectives of MANTIS is the integration of the concepts on which it is based and more importantly the correct management of all the metadata and data at the different conceptual levels previously mentioned. In order to achieve this, the models and meta-models of the different levels can be stored in a repository -of XMI documents- managed by the RepManager tool. Each XMI document stored in the

repository represents a correspondence between the M_i level and M_{i-1} level, as each XMI document will contain the metadata (level M_i) that describe the corresponding data (M_{i-1} level instances). As a result, and bearing in mind that we work with four conceptual levels in MANTIS, three types of XMI documents will be stored in the repository:

- XMI documents that represent the correspondences between the levels M2-M3, such as the correspondence between the MOF-model and the SMP meta-model. The DTD or scheme that will represent this type of XMI documents is unique and will be the DTD of the MOF-model (only the MOF-model is present at level M3 in order to integrate the possible meta-models of the lower level).
- XMI documents that represent the correspondences between the levels M2-M1, such as the correspondence between the SMP generic meta-model with the specific MANTEMA methodology. In this case the DTD represents the meta-model of level M2, for example, it would be necessary to store the DTD of the SMP generic meta-model.
- XMI documents that represent the correspondences between the levels M1-M0 such as the correspondence between MANTEMA methodology and a specific project enactment applied in a company that is using that methodology. In this case the DTD represents specific models of level M1. In the MANTIS environment case it would be necessary to store the DTD corresponding to the MANTEMA methodology.

In figure 4 we can see the document types stored in the repository which give support to MANTIS and the correspondences between them. As can be seen, the data of a level M_i are transformed into metadata of level M_{i-1} which is immediately below. In this way „Maintenance Activity“, which is a piece of data (associated with the MOF-class M3 element) in the XMI document that represents the correspondence M3-M2, is converted into a label (metadata) in the XMI document that represents the correspondence between the levels M2 and M1. Two instances of „Maintenance Activity“ are defined in this M2-M1 mapping document: „Urgent Corrective Intervention“ and „Modification Request Analysis“. In turn, in the M1-M0 mapping document, „Urgent Corrective Intervention“ is a new element label with one new instance named „Intervention number 36“ corresponding to one real task of real SM project enactment.

4.1 Applying to the Assessment Process

In MANTIS, an important application of the RepManager tool is to integrate the different concepts involved in the assessment and improvement of the SMP [23]. Given the importance of improvement, and hence of the assessment, of software processes, it finds important to be able to treat all the concepts involved in these processes in an integrated way. To achieve this integration, we have extended the already commented 4 level architecture of MANTIS with a concrete model of assessment. This way, at level M0 are represented the results of the application of an assessment process to a maintenance project. That is to say, at this level there would be the results of a process from which it would be possible to establish its strong and weak points. The specific

model used at level M1 represents the assessment model proposed in part 5 of ISO 15504 standard [22]. In this level we focus on the guidance that ISO provides with regard to the assessment process.

The main activities that ISO 15504 identifies to assess a software process are Planning, Collecting Data, Validating Data, Process Rating and Reporting. The assessment process output is formed by the process profile which records all data related to the assessment of a process. In table 2 the main correspondences between the elements of the generic SMP meta-model and the main elements of the assessment process are summarized.

We can generate automatically the DTD associated to the assessment process model of ISO 15504 (level M1) using the RepManager tool. This way, the tools used to perform specific assessment processes (level M0) could exchange their data efficiently (in XML documents adapted to the previous DTD).

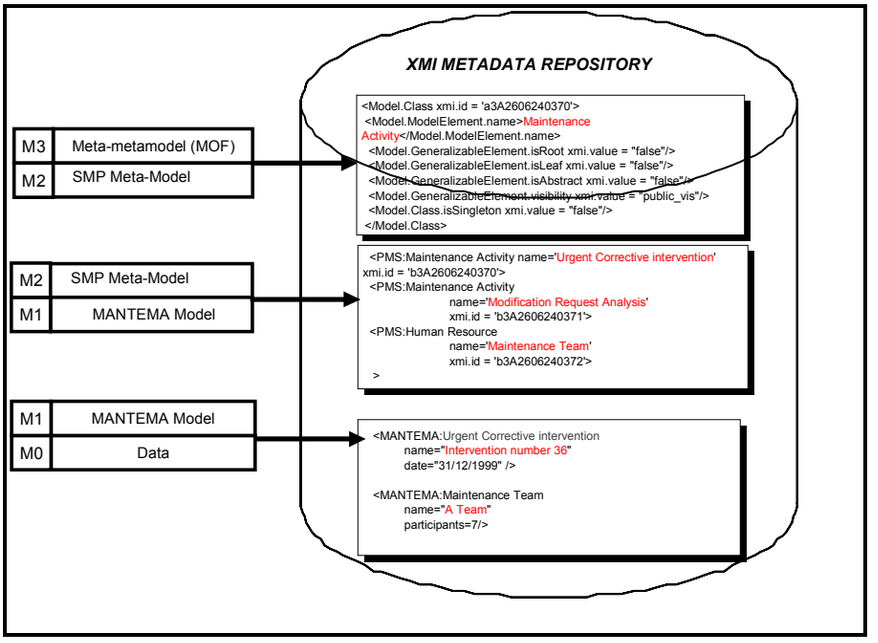

Fig. 4. Using RepManager with the three mapping layers of MANTIS

5 Related and Future Works

There are several examples of recent literature that demonstrate the advantages of MOF and XMI for the construction of CASE tool and SEE repositories. The project FAMOOS (Framework-based Approach for Mastering Object-Oriented Software Evolution) is a good example of this [2]. The goal of this project is to build a framework to support the evolution and reengineering of object-oriented software systems.

With this aim in mind, FAMIX, a language-independent meta-model (in M2 level of MOF) for describing object-oriented source codes at the program entity level, has been developed [17]. In this case RepManager could be used as a repository manager by storing the meta-model FAMIX (together with its associated DTD) and a model (in M1 level) for each of the programming languages used.

In [10], SPAIL (Software Process Assessment Interchange Language) is presented. It is a XML-based language oriented to representing and exchanging descriptions of process assessment results. SPAIL gives rise to a DTD that defines the grammar of this language, taking as its base a specific model of the assessment process, proposed in the standard SPICE [14]. An XML document based on the DTD of SPAIL, containing the results of a real-world process assessment, is an instance of correspondence M1-M0 (relationships between model objects and data objects). Using RepManager it is possible to store and manage this model along with any other model of the same or related processes. In fact, the DTD of SPAIL can be automatically generated by RepManager. To achieve this, all you have to do is define the process meta-model (level M2) using an MOF-based meta-modeling tool (for example, with Mantis-Metamod, although another option would be to use a CASE tool that allows MOF to be exported e.g. Rational Rose for the graphic design of this meta-model) and invoke the GenerateDTDMetamodel use case of RepManager. Although RepManager does not include a use case for generating DTD models, it is not in fact necessary, as by defining the model of the level M1 in the same way as the meta-model of the level M2, their DTDs will also be the same.

The main aim of the RepManager tool is to provide support in the form of a model and meta-model repository to any CASE tool that may need it. This functionality is particularly useful in the area of Process-centered Software Engineering Environments (PSEE), [3]. The MANTIS environment, which we have discussed previously, has an architecture of this type whose main feature is a central process concept. A basic aspect of software processes is that they are subject to changes both when being put into use in a specific real-world project and in relation to the associated process model. The improvement of a process also leads to both types of alterations. In order to manage these changes it is necessary to be able to represent the different models (a new version of a process model whilst similar to the original model is in reality a different process model) and associated meta-models. It would also be useful to enchance RepManager with support for facilitating the re-engineering of models using the common meta-model as an intermediate step.

A further improvement, of a technical type, is to facilitate partial access to the information contained in a model or meta-model by means of a query engine based on the use of the Xquery language [20], a standard proposal of the W3C for XML documents query. The use of XML schemas instead of DTD is also being studied in order to take advantage of their greater semantic capacity [19]. Moreover, XML schemas have XML syntax and as a result can be handled with DOM, the same API as that used for the XML documents themselves [18].

Table 2. Mapping between the generic SMP of MANTIS and the assessment process model of ISO 15504

M2 classes	**M2 instances** (M1 classes)
Activity	Defining the assessment input
	Perform the assessment process
	Planning
	Collect data
	Validate data
	Process rating
	Reporting
	Record the assessment output
Artifact	Process to assess
	Process profile
Activity contains Activity	„Perform …“ contains „Planning“
	„Perform …“ contains „Collect …“
	„Perform …“ contains „Validate …“
	„Perform …“ contains „Process …“
	„Perform …“ contains „Reporting“
Activity consumes Artifact	„Perform …“ consumes „Process to assess“
Activity produces Artifact	„Perform …“ produces „Process profile“

6 Conclusions

In this document we have presented RepManager, a component tool for the management of repositories of software process models and meta-models stored in XMI documents, and its vertical integration with MANTIS, an integrated environment for software maintenance management and improvement.

Nowadays the development and maintenance of software have become inherently overly complex processes due to the distributed, heterogeneous character and the size of the applications. The solution for dealing with this complexity is to be found in a classic but valuable concept of information system terminology: modeling. Models provide the abstraction that is necessary for dealing with this complexity. However, due to the great complexity of the applications, modeling architectures have been proposed that allow the models themselves to be handled in different levels.

The correct management of data and metadata represents the key to dealing with these architectures. Data and metadata repositories have become a key element for the management of software processes. Metadata enable the construction of easy-to-maintain applications and moreover the efficient management and improvement of software development and maintenance processes.

The tool that we have presented allows the management of data and metadata defined for software process modeling. The Repository Manager facilitates working with the models and meta-models stored in the repository. Moreover it provides the user with transparency when handling the data as the system conceals the details related to the way in which the data and metadata are stored, thus allowing these to be efficiently managed.

Any other modeling tool that uses the abstract language provided by MOF and XMI for the definition and storage of meta-models will be able to collaborate with RepManager tool, and in so doing would increase the efficiency of the software process management. As an example of this functionality, we have presented the integration of RepManager with the MANTIS environment and, as demonstration of its generality, we have commented the integration of the assessment process.

Acknowledgements

This work has been partially funded by the TAMANSI project financed by „Consejería de Ciencia y Tecnología, Junta de Comunidades de Castilla-La Mancha“ of Spain (project reference PBC-02-001).

References

1. Becker-Kornstaedt, U. and Webby, R.: A Comprehensive Schema Integrating Software Process Modelling and Software Measurement. Fraunhofer Institute, IESE report Nº 047.99/E., v. 1.2 (1999)
2. Ducasse, S. and Demeyer, S.: The FAMOOS Object-Oriented Reengineering Handbook. University of Berne, October (1999). See http://www.iam.unibe.ch/~famoos/handbook
3. Fugetta, A., Godart, C. and Jahnke, J.: Arquitectural Views and Alternatives. In: Derniame, J-C., Kaba, B.A., and Wastell, D. (eds.): Software Process: Principles, Methodology and Technology. Lecture Notes on Computer Science, Vol. 1500. Springer-Verlag (1999), 95-116
4. ISO/IEC: JTC1/SC7/WG4 15940 working draft 5, Information Technology - Software Engineering Environment Services, June (2000)
5. Kitchenham, B.A., Travassos, G.H., Mayrhauser, A. von, Niessink, F., Schneidewind, N.F., Singer, J., Takada, S., Vehvilainen, R. and Yang, H.: Towards an Ontology of Software Maintenance. Journal of Software Maintenance: Research and Practice. 11 (1999), 365-389
6. MDC: Meta Data Coalition, Open Information Model, v.1.0, August (1999)
7. Morgenthal J.P. and Walms, P.: Mining for Metadata. In Software Magazine, Feb/Mar (2000); Wiesner Publishing. Available in http://www.softwaremag.com/archive/-2000feb/MiningMetadata.html
8. OMG: Object Management Group, Meta Object Facility (MOF) Specification, v. 1.3 RTF, September (1999). In http://www.omg.org
9. OMG: Object Management Group, XML Metadata Interchange (XMI), v. 1.1, November (2000). In http://www.omg.org
10. Park, J. and Lee, K.: A XML-based Approach to Software Process Improvement Environment on The Internet. Proceedings of the IASTED International Conference on Software Engineering and Applications. Anaheim-California, USA (2001), 76-80

11. Pigoski, T.M.: Practical Software Maintenance. Best Practices for Managing your Investment. John Wiley & Sons, USA (1996)
12. Polo, M., Piattini, M., Ruiz, F. and Calero, C.: MANTEMA: A complete rigorous methodology for supporting maintenance based on the ISO/IEC 12207 Standard. Third Euromicro Conference on Software Maintenance and Reengineering (CSMR'99). Amsterdam (Netherland), IEEE Computer Society (1999), 178-181
13. Polo, M., Piattini, M., Ruiz, F. and Jiménez, M.: Assessment of Maintenance Maturity in IT Departments of Public Entities: Two Case Studies. Proceedings of Third International Conference on Product Focused Software Process Improvement (PROFES'2001). Kaiserslautern, Germany, 86-97
14. Rout, T.: SPICE: A Framework for Software Process Assessment. Software Process Improvement and Practice. vol 1, no 1 (1995), 57-66
15. Ruiz, F., Piattini, M., Polo, M. and Calero, C.: Audit of Software Maintenance. In: Auditing Information Systems, USA, Idea Group Publishing (2000), 67-108
16. Ruiz, F., Piattini, M. and Polo, M.: An Conceptual Architecture Proposal for Software Maintenance. International Symposium on Systems Integration (ISSI, Intersymp'2001). Baden-Baden, Germany (2001), VIII:1-8
17. Tichelaar, S., Ducasse, S. and Demeyer, S.: FAMIX and XMI. Proceedings of the Seventh Working Conference on Reverse Engineering (WCRE'2000). Brisbane, Australia (2000), IEEE Computer Society Press, 296-299
18. W3C: World-Wide-Web Consortium, Document Object Model (DOM) level 1 specification, v. 1.0. October (1998). In http://www.w3.org/DOM/
19. W3C: World-Wide-Web Consortium, XML Schema Part 0: Primer, May (2001). In http://www.w3.org/TR/xmlschema-0/
20. W3C: World-Wide-Web Consortium, XQuery 1.0: An XML Query Language, working draft, June (2001). In http://www.w3.org/TR/xquery/
21. WfMC: Workflow Management Coalition TC00-1003 1.1, The Workflow Reference Model, January (1995)
22. ISO/IEC: ISO/IEC TR 15504-5: Information Technology – Software Process Assessment - part 5: An assessment model and indicator guidance, (1999)
23. García, F., Ruiz, F., Piattini, M. and Polo, M.: Conceptual Architecture for the Assessment and Improvement of Software Maintenance. 4th International Conference on Enterprise Information Systems (ICEIS'02). Ciudad Real, Spain, 610-617