Ernestina Menasalvas
Javier Segovia
Piotr S. Szczepaniak (Eds.)

LNAI 2663

Advances in Web Intelligence

First International Atlantic Web Intelligence Conference
AWIC 2003, Madrid, Spain, May 2003
Proceedings

Springer

Springer
Berlin
Heidelberg
New York
Hong Kong
London
Milan
Paris
Tokyo

Ernestina Menasalvas Javier Segovia
Piotr S. Szczepaniak (Eds.)

Advances in Web Intelligence

First International Atlantic Web Intelligence Conference
AWIC 2003
Madrid, Spain, May 5-6, 2003
Proceedings

Springer

Series Editors

Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA
Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Volume Editors

Ernestina Menasalvas
Javier Segovia
Universidad Politécnica de Madrid
Facultad de Informática
Campus de Montegancedo, Boadilla del Monte, 28660 Madrid, Spain
E-mail: {emenasalvas/fsegovia}@fi.upm.es

Piotr S. Szczepaniak
Technical University of Lodz
Institute of Computer Science
ul.Sterlinga 16/18, 90-217 Lodz, Poland
E-mail: office@ics.p.lodz.pl

Cataloging-in-Publication Data applied for

A catalog record for this book is available from the Library of Congress.

Bibliographic information published by Die Deutsche Bibliothek
Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data is available in the Internet at <http://dnb.ddb.de>.

CR Subject Classification (1998): I.2, H.3, H.4, C.2, H.5, J.1, J.2

ISSN 0302-9743
ISBN 3-540-40124-5 Springer-Verlag Berlin Heidelberg New York

Springer-Verlag Berlin Heidelberg New York
a member of BertelsmannSpringer Science+Business Media GmbH

http://www.springer.de

Printed in Germany

Typesetting: Camera-ready by author, data conversion by PTP-Berlin GmbH

Preface

We are pleased to present the proceedings of the 2003 Atlantic Web Intelligence Conference, AWIC 2003. The conference was located in Madrid, Spain during May 5–6, 2003, organized locally by the Technical University of Madrid.

AWIC 2003 aimed to be the first of a series of conferences on Web Intelligence, to be celebrated annually, alternatively in Europe and America, starting in Madrid. It was born as an activity of the recently created WIC-Poland Research Centre and the WIC-Spain Research Centre, both belonging to the Web Intelligence Consortium (WIC) (`http://wi-consortium.org`). AWIC 2003 was supported with grants from the Spanish Ministry for Science and Technology and the European Network of Excellence in Knowledge Discovery, KDNet.

AWIC 2003 brought together scientists, engineers, computer users, and students to exchange and share their experiences, new ideas, and research results about all aspects (theory, applications, and tools) of artificial intelligence techniques applied to Web-based systems, and to discuss the practical challenges encountered and the solutions adopted. Almost 70 contributions were submitted. After a preliminary evaluation, 60 of these papers were accepted to the conference and were assigned at least two reviewers from the international program committee. Out of this 60, 33 were conditionally accepted, and 32 of them were finally accepted after the conditions set by the reviewers had been met, which resulted in an acceptance ratio of 45%.

AWIC 2003 had a general session with 9 papers, and the others were organized into special sessions on areas such as:

- Web security,
- Semantic Web,
- Web authoring and design,
- Web information retrieval,
- Agents for the web, and
- User behavior

During AWIC 2003 we were pleased to have some important invited speakers. Prof. Lotfi Zadeh, who does not need any introduction, was the honorary chair and gave a talk about the transition of search engines to question-answering systems. This book contains what could be considered an extended abstract of his actual lecture. Prof. Ning Zhong, President of WIC, who can be considered as one of the fathers of the term "Web Intelligence," gave a talk about the past, present, and future of the area. Finally, Prof. Ryszard Tadeusiewicz spoke about the application of artificial intelligence techniques for information retrieval.

We thank all the authors for contributing to what in our minds was a more than interesting program in Web Intelligence. We also want to thank the remarkable effort made by Coro Perez, secretary of the conference, and the rest of the local committee. We hope to see all of them, and many others at AWIC 2004.

May 2003

E. Menasalvas
J. Segovia

AWIC 2003 Organization

Honorary Chair

Lotfi A. Zadeh (University of California, USA)

Conference Co-chairs

Javier Segovia (Universidad Politécnica de Madrid, Spain)
Piotr Szczepaniak (Technical University of Lodz, Poland)

Steering Committee

Ning Zhong (Maebashi Institute of Technology, Japan)
Enric Trillas (Universidad Politécnica de Madrid, Spain)
Jiming Liu (Hong Kong Baptist University)
Janusz Kacprzyk (Systems Research Institute, Polish Academy of Sciences, Warsaw, Poland)

Organizing Committee

Ernestina Menasalvas (Universidad Politécnica de Madrid, Spain)
Jesús Favela (Centro de Investigación Científica y de Educación Superior de Ensenada, México)

Local Committee

José M. Peña (Universidad Politécnica de Madrid, Spain)
Oscar Marbán (Universidad Carlos III de Madrid, Spain)
Ángel Lucas (Universidad Politécnica de Madrid, Spain)
Sonia Frutos (Universidad Politécnica de Madrid, Spain)
Javier Crespo (Universidad Carlos III de Madrid, Spain)
Eugenio Santos (Universidad Politécnica de Madrid, Spain)

Organization of the Special Track on Uncertainty, Hypermedia and the Web: Search, Personalization and Usability

José Angel Olivas (Universidad de Castilla-La Mancha, Spain)
Miguel-Ángel Sicilia (Universidad Carlos III, Spain)

Table of Contents

Invited Speakers

General Session

Special Session: Web Security

Special Session: Semantic Weg

ecial Session: Web Authoring and Design

Special Session: Web Information Retrieval

Special Session: Agents for the Web

Toward Web Intelligence

Ning Zhong

Department of Information Engineering
Maebashi Institute of Technology
460-1 Kamisadori-Cho, Maebashi-City 371-0816, Japan
zhong@maebashi-it.ac.jp

Abstract. Web Intelligence (WI) presents excellent opportunities and challenges for the research and development of new generation of Web-based information processing technology, as well as for exploiting Web-based advanced applications. Based on two perspectives of WI research: an intelligent Web-based business-centric schematic diagram and the conceptual levels of WI, we investigates various ways to study WI and potential applications.

1 Introduction

The concept of Web Intelligence (WI for short) was first introduced in our papers and books [19,35,38,41,45,46]. Broadly speaking, Web Intelligence is a new direction for scientific research and development that explores the fundamental roles as well as practical impacts of Artificial Intelligence (AI)[1] and advanced Information Technology (IT) on the next generation of Web-empowered systems, services, and environments. The WI technology revolutionizes the way in which information is gathered, stored, processed, presented, shared, and used by electoronization, virtualization, globalization, standardization, personalization, and portals.

With the rapid growth of the Web, research and development on WI have received much attention. There is great potential for WI to make useful contributions to e-business (include e-commerce), e-science, e-learning, e-government, and so on. Many specific applications and systems have been proposed and studied. In particular, the e-business activity that involves the end user is undergoing a significant revolution [29]. The ability to track users' browsing behavior down to individual mouse clicks has brought the vendor and end customer closer than ever before. It is now possible for a vendor to personalize his product message for individual customers at a massive scale. This is called *direct marketing* (or *targeted marketing*) [32,47]. Web mining and Web usage analysis play an important role in e-business for customer relationship management (CRM) and direct marketing. Web mining is the use of data mining techniques to automatically discover and extract information from Web documents and services [15,29,36,

[1] Here the term of AI includes classical AI, computational intelligence, and soft computing.

Supporting Software Maintenance in Web Repositories through a Multi-agent System

Aurora Vizcaino[1], Jesús Favela[2], Mario Piattini[1], and Felix García[1]

[1]Escuela Superior de Informática. Universidad de Castilla-La Mancha
13071, Paseo de la Universidad, 4, Ciudad Real, Spain
{avizcain,mpiattini,fgarcia}@inf-cr.uclm.es
[2]CICESE, Ensenada, México
favela@cicese.mx

Abstract. Software Maintenance (SM) is a knowledge-intensive activity. A suitable management of this knowledge could decrease the high costs (economic and in effort) of software maintenance tasks.
The challenge of managing this knowledge increases as the distributed development of software becomes more popular, and developers as well as knowledge are distributed worldwide. Increasingly web repositories are being used for the coordination of development tasks among software and maintenance engineers. Thus, an appropriate technical solution to this problem should be based on a web architecture and associated protocols.
On the other hand, in order to work with all the concepts related to SM is advisable to establish different levels of abstraction, thus the complexity of the concepts, and their management, are simplified. This work presents a system that, by storing information in XMI documents, manages the data and metadata generated during SM, facilitating the work of SM engineers.

1 Introduction

Many studies [7,15,18] provide evidence that the majority of the overall expenses incurred during the life-cycle of a software product occur during the maintenance stages. Thus, in recent years researchers have focused their attention on proposing methods and techniques which help increase the efficiency of the Software Maintenance Process (SMP).

One way to improve maintenance quality and decrease software maintenance costs is to reuse previous information and knowledge [10]. However, for information to be usable it needs to be modeled, structured, generalised and stored in a reusable form, to allow for effective retrieval [1].

In order to decrease the efforts and cost in the SMP we developed a knowledge management system called KM-MANTIS. The system is in charge of storing and reusing information, knowledge and expertise generated during the SMP. KM-MANTIS is based on the experience factory concept [4,5] since it is known that an organizational memory must be maintained by an organizational unit separate from the project organizations because it is mainly concerned to keeping schedules and cost constraints, and proving knowledge would imply an extra effort for the project organizations employees.

E. Menasalvas et al. (Eds.): AWIC 2003, LNAI 2663, pp. 307–317, 2003.

I -MANTIS is a multi-agent system where different types of agent manage the div types of information generated during SMP. Agents interchange data and take adv age of the information and experience acquired by other agents.

main feature of KM-MANTIS is that it stores data and metadata in XMI (XI metadata interchange) [13] documents. It is critical to manage diferent levels of raction to carry out an integral management of a software process. Moreover bei XMI an open format fosters the interchange of data between distributed and het geneous repositories which also use XMI.

contents of this paper are divided as follows. Section 2 discusses the impor tan of managing the knowledge generated during the software maintenance process. Sec n 3 describes the features of KM-MANTIS. Section 4 explains how and why inf ation is stored in a XMI repository, and shows, through an example, the use of the stem. Finally, conclusions are presented in Sect. 5.

2 he Need for Knowledge Management in Software Maintenance

Sof are maintenance is an activity where different kinds of knowledge are generated fro different sources. This knowledge comes not only from the expertise of the pro sionals involved in the process, but it is also intrinsic to the product being main tair , and to the reasons that motivate the maintenance (new requirements, user cor aints, etc). Moreover, the diverse types of knowledge are produced at different sta of the maintenance process. For instance, [6] claims that during Initial Devel opr t the maintenance staff acquires knowledge about the application domain, user req ements, roles of the application, solutions and algorithms, data formats, stre hs and weaknesses of the program architecture, operating environment, etc. In the ftware Evolution stage new users and environmental requirements arise and cha s are produced which generate new knowledge, and this process continues in the t of the stages.

thermore, this knowledge is used by different persons at different stages. Each per has partial information that is required by other members of the group. If the kn edge only exists in the software engineers and there is no system in charge of tra erring the tacit knowledge (contained in the employees) to explicit knowledge (st l on paper, in files, etc) when an employee abandons the organization a signifi can art of the intellectual capital goes with him/her.

of the five factors that have been identified as having a major impact on the pro ctivity of software maintenance is the expertise of the staff members [2]. It has bee found that systems maintained by relatively inexperienced programmers average sig cantly higher error rates [3].

other well-known issue that complicates the SMP is the scarce documentation tha xists related to a specific software system. Even if detailed documentation was pro ced when the original system was developed, it is seldom updated as the system evo s. For example, legacy software written by other units often has little or no doc entation describing the features of the software.

ddition, organizations still lack a culture of reuse and information sharing. As [12 aims, companies only use a quarter part of its intellectual capital.

the other hand, thanks to technology advances it is frequent for maintenance

- Coordination of distributed work.
- Providing access to knowledge to the different sub-units.
- Managing experience compiled from previous projects and making it available and reusable for new projects.

A knowledge management system could satisfy these needs so as to avoid some of the issues commented previously. For instance, if organizations store their information and knowledge in a KM system, they would own this intellectual capital. Therefore, even if experts left the organization their expertise would remain within the companies.

In addition, a suitable storage of information facilitates its reuse. For these reasons we decided to develop a KM system where the diverse kinds of knowledge generated from different stages of SMP were stored, analysed, shared and reused.

Another advantage of a KM system is that staff may also be informed about the location of information. A critical factor for maintenance engineers is to have access to the knowledge the organizations have. In [22] a study was conducted, which found that the number one barrier to knowledge sharing was "ignorance": the sub-units are ignorant of the knowledge that exists in the organizations, or the sub-units possessing the knowledge are ignorant of the fact that another sub-unit needs such knowledge. Sometimes the organization itself is not aware of the location of the pockets of knowledge or expertise [11].

3 The KM-MANTIS Multi-agent System

This section describes the features of KM-MANTIS and its architecture. However, before presenting the system we explain why a multi-agent architecture for the management of information was chosen.

3.1 Why Agents?

Previous works such as [21, 19] have used agents for knowledge management. There are several reasons why agents are recommendable for this task. First of all, agents can manage both distribute and local information. This is an important feature since, as it was explained in Sect. 2, the software maintenance information is generated by different sources and often from different places.

In a distributed architecture each agent knows about its local environment, and it is close to the place where the information is generated. When global knowledge is required the agents can act together and share knowledge. The alternative will be to have a centralized system that knows "everything". However, this is more difficult to implement and also less robust.

An aspect related to the previous one is that agents may cooperate and interchange information. Thus, each agent can share its knowledge with others or ask them advice, benefiting from the other agents' experience. Therefore, there is reuse and knowledge management in the architecture of the system itself.

A: :her important issue is that agents can learn from their own experience. Conse quen the system is expected to become more efficient with time since the agents have :rnt from their previous mistakes and successes.

O: ie other hand, agents may utilize different reasoning techniques depending on the s: ation. For instance, they can use ID3 algorithms to learn from previous experi ence: id use case-based reasoning to advise a client how to solve a problem.

3.2 :e KM-MANTIS Architecture

The : tem is formed of a set of agent communities which manage different types of know dge. When the ontology to represent the maintenance domain was developed diffe: it types of information were detected, thus the system has a community per type information detected since each one has its specific features. The three more relev. types of information identified were: information related to the products to be main ied; information referent to the activities to perform in order to maintain the prod :; and, peopleware involved during software maintenance process [23].

Tl :fore, KM-MANTIS has three communities: a community termed "products comr :ity", another called "activities community", and the last community denoted "peo; :ware community". In what follows, we describe each community in more detai

Pr :ucts Community: This community manages the information related to the prod : to be maintained. Since each product has its own features and follows a speci. evolution this community has one agent per product. The agents have infor matic :bout the initial requirements, changes made to the product, and about metrics that e :luate features related to the maintainability of the product (this information is obtai: i from different documents such as modification requests, perfective, correc tive c :reventive actions performed or product measurements). Therefore, the agents moni: the product's evolution in order to have up to date information about it at each mom: t.

Ea time an agent detects that information about its product is being introduced or modi: :l in KM-MANTIS (the agent detects it when the product identification num ber tl it represents is introduced or displayed in the interface of KM-MANTIS) the agent :rts to work analysing the new information (in the case of there being any), or comp ng it to that previously held in order to detect inconsistencies, or checking the differ :es and storing the relevant information in order to have up-to-date informa tion.

Be :es information relevant to each product, which is stored in a database that each : :nt has, general information about products such as type of products that the syster nanages should be considered. This information which can be considered as metac : is stored in a XMI repository common for the community, called "commu nity r :sitory" (see Fig. 1). The decision to use XMI documents based on the MOF (Met: bject Facility) standard (they are described in more detail in Sect. 4) makes it possil for agents of this community to have access to the different levels of infor matio hat they need to process and classify their information.

Th . products and activities communities have a common repository where gen eral i: :mation and metadata are stored. For instance, an agent can detect that when ever : hange labelled "A" is demanded another change called "B" is required. This

will be shared with the rest of the product agents. In a previous design of this system a single agent was in charge of this repository but this fact increased considerably the traffic of messages between agents, and so it was eliminated, producing thereby an improvement in the speed of the system.

Besides the repository that each community has, agents can also consult a general repository where the documents generated during the SMP are stored in format XMI. This repository is usually used when agents need additional information for answering queries or performing their tasks. Later, in Sect. 4, we provide examples of this.

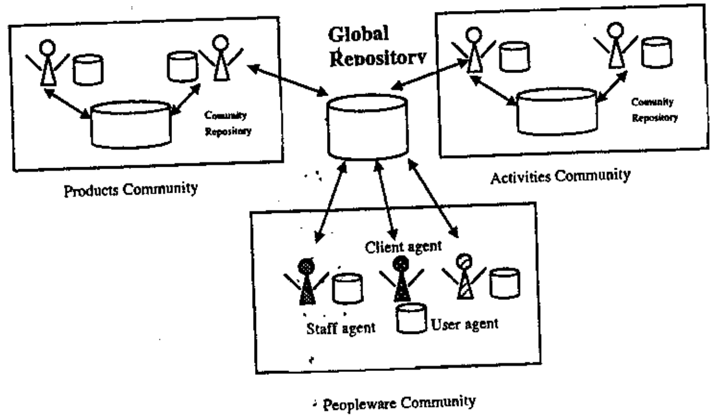

Fig. 1. KM-MANTIS architecture

Activities Community: Each new change demanded implies performing one or more activities. This community, which has one agent per activity, is in charge of managing the knowledge related to the different activities such as methods, techniques and resources used to perform an activity.

Activities agents can also obtain new knowledge from their experience and learning. For instance, an activity agent can learn what person usually carries out a specific activity.

As in the previous case each agent has a database to store its information, and a community repository to access it. This repository contains metadata and data important for all the community. For instance, a taxonomy of activities.

Peopleware Community: Three profiles of people can be clearly differentiated in MP [16]: the maintainer, the customer and the user. The Peopleware community has three types of agent, one per profile detected. One agent is in charge of the information related to staff (maintainers). This is the staff agent. Another manages information related to the clients (customers) and is called the client agent. The last one is in charge of the users and is termed the user agent.

The staff agent knows the personal data of the employees, in what activities they have worked, and what product they have maintained. Of course, the agent also has current information about each member of the staff. Therefore, it knows where each person is working at each moment.

Th(;gent can utilise the information that it has to generate new knowledge. For in-stance ;he staff agent may calculate statistics that indicate the time that an employee took t; :erform a task or calculate the performance graph of each member.

Th(:lient agent stores the information of each client, their requirements (even the initial ;quirements if they are available) and requests. The client agent also tries to obtain . w knowledge. For instance, it tries to guess future requirements depending on pre ous requirements or it estimates the costs of changes that the client wants to make, arning him for instance of the high costs associated with a specific change reques

The ;er agent is in charge of knowing the requirements of the users of each prod-uct, th background and also their complaints and comments about the products. New k ;wledge could be generated from this information, for example by testing to what d ;ee the users' characteristics influence the maintenance of the product.

Eac ;ype of agent has a database but in this case there is not a community reposi-tory bc ;use there is no data common to the three types of agents. However, as arrows indicat ;n Fig. 1 agents can access the global repository.

Agc exchange information by using ACL (Agent Communication Language). This l; ;uage provides agents with the means to share information and knowledge throug ; set of communicative acts.

The ;tform used is JADE (Java Agent Development Framework) since it simpli-fies the ;plementation of multi-agent systems through a set of tools that supports the debug; ;g and deployment phase.

4 W; Repositories of Information

The W ;d Wide Web has moved from its origins as a medium for the dissemination of info ;ation to a middleware platform for the development of sophisticated distrib-uted ap ;cations. This trend has given rise to a new branch of Software Engineering, named ;eb Engineering, concerned with establishing sound principles, techniques and toc ;or the development and maintenance of systems, services and applications over th ;eb.

As t; ;veb becomes the preferred medium for the deployment of software applica-tions sc ;al CASE tools have migrated to the web and several others have been de-velopec ;he use of the web as a platform for the support of software development offers s ;eral advantages:

- ;biquitous access. Web repositories can be accessed from any computer ;nnected to the Internet, including portable devices.
- ;imple integration of tools. The web is based on simple, open protocols that ;n be easily integrated in CASE tools to allow their interoperability. At the ;west level of integration a tool can export content to HTML or semanti-;lly richer XML documents.
- ;pport for distributed software development. Large-scale software systems ;e increasingly being developed by distributed teams of specialists that need ; communicate and coordinate their activities. The web offers a simple mid-;eware for the deployment of workflow and groupware tools that support ;llaboration.

Of particular interest to us is the use of web servers as repositories for product and process information. This information is not only essential to the production of software, but to its maintenance as well. The centralized storage of this information facilitates configuration management, quality control, project tracking, and the management of requirements.

An important effort in this area is the IETF's DeltaV project [24], an effort to extend the WebDAV and HTTP protocols to support document versioning and configuration control for shared web documents and resources.

On the other hand, applications manipulating Web data require both, documents or information retrieved from the Web, and metadata about this information [8]. In order to specify and manage meta-data we used the standard MOF proposed by the OMG [13].

This standard defines four conceptual levels (see Table 1): the base level, called M0, is where the instances or real data are. For example in the KM-MANTIS repository at this level we find the concrete changes that have been performed. Data managed in this level are instances of the concepts defined in the superior level called M1. The specific model that we used in level M1 is based on the MANTEMA methodology [17] and on a set of techniques adapted to maintenance, such as effort or risk estimations [20]. An example of information at this level are the different types of *support activities* that exist according to MANTEMA. The next level is the M2, which corresponds to the meta-models, in our case to the software maintenance process meta-model. And the last level, M3, is where the meta-meta model is represented, this is the MOF model itself.

Table 1. Conceptual levels in MOF and KM-MANTIS

Level	MOF	KM-MANTIS
M3	MOF model (Meta-metamodel)	MOF Model
M2	Meta-model	SMP metamodel
M1	Model	MANTEMA
M0	Data	SMP instances (products, activities)

An example will help illustrate this classification. The generic concept "Maintenance Activity" represents the correspondence between M3 and M2 (see Fig. 2). This concept is instantiated in a specific type of activity in the correspondence M2 and M1. For instance, in "Urgent Corrective Intervention" activity. When we have real instances of this type of activity it will be stored in the level M0, for example "Intervention number 3 performed in the project Counts.

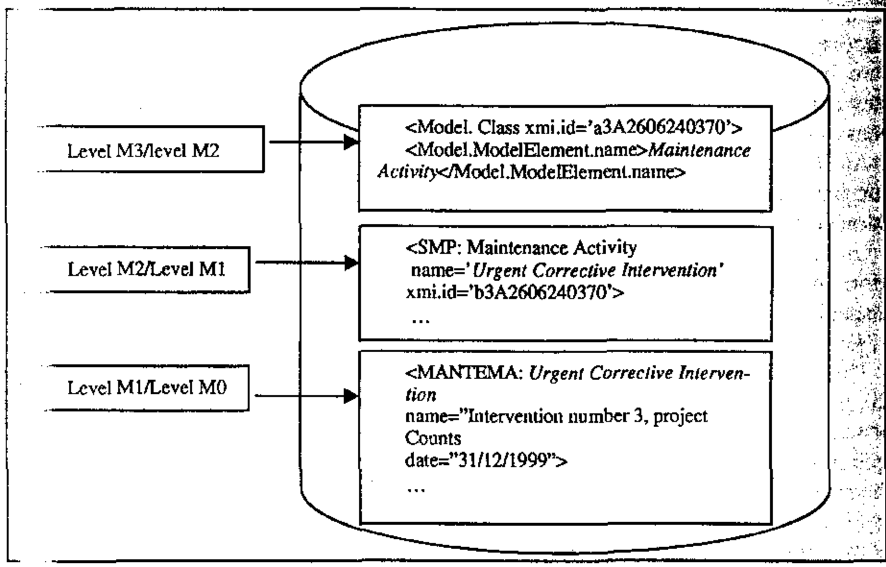

Fig. 2. Example of the XMI repository

As [Fi]g. 2 shows, data from an Mi level are converted into metadata in the inferior lev[el] Mi-1. Thus, "Maintenance Activity", that is a piece of data in the XMI docume[nt] which represents the correspondence between M3-M2, is converted into a label (m[eta]data) in the XMI document which represents the correspondence between the lev[els] M2 and M1.

[Usi]ng XMI enables us to represent different levels of abstraction and their relationshi[ps]. Knowing the relationship between these levels helps to deduce new knowledge fro[m] the previous one. Moreover, XMI permits to obtain information from other repo[sito]ries, systems or information sources which support XMI [14]. And it also facilitat[e] the interchange of information between agents since all use the same information rep[rese]ntation.

[In] order to demonstrate how agents use the repositories we describe a scenario. Su[ppo]se a user consults in KM-MANTIS, when and by whom the intervention number[,] which was an Urgent Corrective Intervention, was performed. In this case the age[nt] in charge of this type of activity should act. As Fig. 3 shows, the urgent correctiv[e in]tervention agent will consult in its database the date and name of the person wh[o c]arried out the intervention number 3. If a second query was performed asking the [na]me of the maintenance activities the agent would have to consult the community rep[osi]tory where the information requested is. This type of information is in a higher lev[el i]n the MOF standard, which corresponds to M2-M1, therefore XMI documents of [this] representation would be consulted in order to find all the names of maintenan[ce] activities.

[To] show an access to the global repository, imagine that a user needs to see the interv[ent]ion number 3 request document. In this case the agent will access the global rep[osit]ory in order to get the XMI document that represents that request document.

A last example showing collaboration between agents is exposed. Once it is known who performed the intervention number 3 the user asks for more information about this person. The agent would contact the staff agent which will answer this request providing all information related to John Cruise such as telephone number, e-mail, his office location, etc.

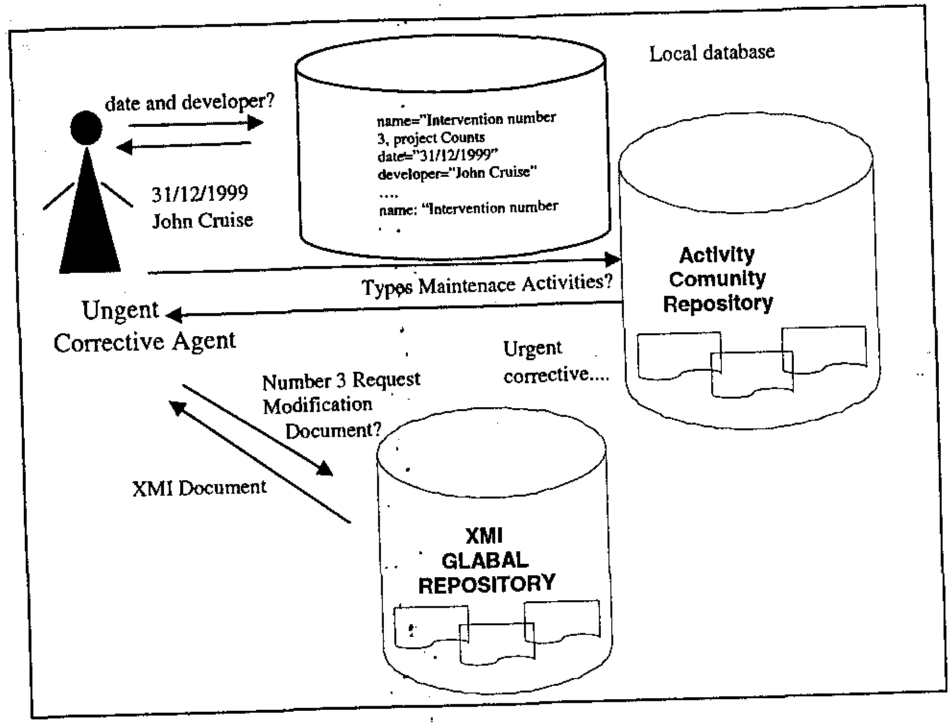

Fig. 3. Example of consults in KM-MANTIS

5 Conclusions

The complexity of the SM process demands a model of the information with a suitable level of abstractation for dealing with the different concepts that the process involves. XMI and MOF enable us to define and store a model with different levels of metadata and data. Several examples have clarified how agents utilise the levels and how diverse abstraction queries can be solved.

A future work will be the evaluation of KM-MANTIS following the DESMET [9] method. DESMET proposes a technique of evaluation divided into two types: Evaluation focused on the measurable effects of using a method or tool and the second evaluation centred on establishing criteria to measure in what degree a method or tool is suitable for a orgnisation taking into account its necesities and its culture.

R· ·rences

1. ·lthoff, K-D, Birk, A., and Tautz, C. : The Experience Factory Approach: Realizing ·earning from Experience in Software Development Organizations. In proceedings of the ·)th German Workshop on Machine Learning (FGML 1997), University of Karlsruhe, ·997), 6–8
2. ·anker, R.D., Datar, S.M. and Kemerer, C.F. A model to evaluate variables impacting the ·roductivity of software Maintenance Project. Management Science 37 (1), (1991), 1–18
3. ·anker, R.D., Datar, S.M. and Kemerer, C.F., and Zweig, D. Software Errors and Software ·faintenance Management. Journal of Information Technology & Management. Pirkul, N., ·nd Vargheses S.J (Eds.). Vol. 03. ISSN 1385-951X. Kluwer Academic Publichers, (2002) ·5–41
4. ·asili, V.R., and Rombach, H.D. The TAME project: Towards improvement-oriented ·oftware environments. IEEE Transactions on Software Engineering, SE-14(6):(1988): ·8–773
5. ·asili, V.R., Caldiera, G., and Rombach, H.D.: Experience Factory. In John J. Marciniak ·d.) Encyclopedia of Software Engineering, v. 1, pp. 469-476. John Wiley & Sons.(1994)
6. ·nnet K.H., and Rajlich V.T. Software Maintenance and Evolution: a Roadmap, in ·nkelstein, A. (Ed.), The Future of Software Engineering, ICSE 2000, June 4–11, Limer-·k, Ireland, (2000) 75–87
7. ·ard, D.N., and Glass, R.L. Measuring Software Design Quality. Prentice Hall, Engle-·ood Cliffs, NJ (1990)
8. ·omes, D., Campos, J.P., Silva, M.J: Versus: A Web Repository. Workshop on Distrib-·ed Data and Structures (WDAS) (2002)
9. ·tchenham, B. Evaluating Software Methods and Tools Using the DESMET Methodol-·y. In Proc. Of Empirical Assesment in Software Engineering (EASE), University of ·ele, U.K. (1997).
10. · Looff, L.A., Information Systems Outsourcing Decision Making: a Managerial Ap-·oach. Hershey, PA: Idea Group Publishing, (1990)
11. ·bus, J. Framing the Knowledge Search Problem: Whom Do We Contact, and Why Do ·e Contact Them? Academy of Management Best Papers Proceedings, (2001) h1–h7
12. ·onaka, Ikujiro and Takeuchi, Hirotaka.; The knowledge creating company. Oxford Uni-·rsity Press. ISBN 0-19-509269-4. (1995)
13. ·IG Meta Object Facility (MOF) Specification, v. 1.3 RTF, Sep. 1999. In ·tp://www.omg.org.
14. ·IG XML Metadata Interchange (XMI), v. 1.1, Nov-2000
15. ·oski, T.M.: Practical Software Maintenance. Best Practices for Managing Your Invest-·nt. Ed. John Wiley & Sons, USA, (1997)
16. ·lo, M., Piattini, M., Ruiz, F. and Calero, C.: Roles in the Maintenance Process. Software ·gineering Notes; vol 24, no. 4, 84–86. ACM., (1999)
17. ·lo, M., Piattini, M., Ruiz, F. and Calero, C.: MANTEMA: A Complete Rigourous ·thodology for Supporting Maintenance based on the ISO/IEC 12207 Standard. Third ·romicro Conference on Software Maintenance and Reengineering (CSMR'99). Amster-·n (The Netherlands). IEEE Computer Society, (1999) 178–181
18. ·lo, M., Piattini, M., and Ruiz, F.: Using a Qualitative Research Method for Building A ·tware Maintenance Methodology. In Software Practice & Experience. John Wiley and ·ns. Vol. 32, No. 13, (2002): 1239–1260
19. ·smus, D.W. Mind Tools: Connecting to Groupware. PC AI, September/October, (1996) ·-36
20. ·z, F.; Piattini, M.; Polo, M.; Calero, C.; Audit of Software Maintenance. In "Auditing ·ormation Systems". Idea Group Publishing, USA (2000)
21. ·verman, B.G., Bedewi, N. and Morales, A. Intelligent Agents in Software Reuse Re-·itories. CIKM Workshop on Intelligent Information Agents. Baltimore, (1995)
22. Szulanski, G.,: Intra-Firm Transfer of Best Practices Project. American Productivity and Quality Centre, Houston, Texas, (1994) 2–19
23. Vizcaíno, A., Ruíz, F., Favela, J., and Piattini, M. A Multi-Agent Architecture for Knowledge Management in Software Maintenance. In Proceedings of International Workshop on Practical Applications of Agents and Multiagent Systems (IWPAAMS'02), Salamanca, Spain 23–25 October, (2002) 39–52
24. Whitehead, J. "The Future of Distributed Software Development on the Internet." Web Techniques, Vol. 4, No. 10, October, (1999), 57–63