A Multy-Agent Knowledge Management System for Software Maintenance

Aurora Vizcaíno[1] , Jesús Favela[2], Mario Piattini[1]

[1]Grupo Alarcos,Escuela Superior de Informática, Ciudad Real (Spain)
[2]Cicese, Ensenada (México)

1. Introduction

Knowledge is to the Information Age as oil was to the Industrial Age (Larson et al., 2001). Nowadays organisations consider knowledge, also called intellectual capital, to be more important than tangible capital. To understand the knowledge that an organisation has enables it to grow and survive (Gupta and Govindarajan, 2000). For this reason, organisations are currently researching techniques and methods to manage their knowledge systematically.

On the other hand, organisations have different types of knowledge that are often related to each other and which must be managed in a consistent way. In this paper we focus on the different types of knowledge that are related to software engineering, concretely, on the software maintenance process.

Software engineering involves the integration of different knowledge sources that are in constant change. The management of this knowledge and how it can be applied to software development efforts has received little attention in the software engineering research community so far (Henninger and Schlabanch, 2001). However, tools and techniques are necessary to capture and process knowledge in order to facilitate subsequent development efforts.

This paper presents a multi-agent system in charge of manage the knowledge that is produced during the software maintenance process in order to help the staff to make decisions. The contents of this article are organised as follows: section 2 describes the different types of knowledge that are generated during the software maintenance process. Section 3 outlines the roles that the agents play in the knowledge management system. Finally conclusions are presented in section 4.

2. Knowledge in Software Maintenance

Many studies (Card and Glass, 1990; Pigoski, 1997) have demonstrated that the majority of the overall expenses incurred during the development of a software product occur during the maintenance stages. Thus, in recent years researchers have focussed their attention on looking for techniques which help to increase the efficiency of this problem. Software maintenance is a knowledge intensive activity. This knowledge comes not only from the expertise of the professionals involved in the process, but also from the one intrinsic to the product being maintained, the reasons that motivate the maintenance (new requirements, user complains, etc) and processes, methodologies and tool used in the organization.

During software maintenance several characteristics change substantially. One such example are changes in the maintenance staff (which could mean that the people's expertise changes as well), or the frequency with which each type of maintenance (corrective, perceptive, adaptive or preventive) is carried out. It would be advisable to utilise a system in charge of controlling maintenance information. The system might produce new knowledge and obtain the maximum performance from the current information. By reusing information and producing knowledge the high costs of software maintenance could also be decreased (De Loof, 1990).

One of the most important elements in the software maintenance process are the members of the staff. They have skills that are necessary in the use of a methodology, which helps to construct a product, which conforms to standards and satisfies criteria. But what would happen if a person, who has a certain degree of expertise, plays a role in a project and who is part of a team abandons the project or, what would be worse, the company. There are several possible answers to this question. One is that the company would lose the expertise of the person and for this reason the product would perhaps not be deliberated on time therefore producing an increase in the costs. Another answer is that no problem would be apparent- at that moment but will rise when something that was done by this person needs to be modified. The most optimistic response might affirm that nothing would happen since the company has a knowledge management system where the employees knowledge has been codified with the finality that such knowledge was accessible to the wider organisation (Kogut and Zander, 1992).

Software maintenance involves many activities where different people intervene. Each person has partial information that is probably necessary for the rest of the employees. If the knowledge only exists in the workers and there is no a system in charge of transferring the tacit knowledge (contained in the employees) to explicit knowledge (stored on paper, in files, etc) when an employee abandons the organisation part of the intellectual capital goes with him/her. If this fact occurred in a software maintenance company it could mean the impossibility of continuing to manage a project or maintaining a software that would cause a huge loss of benefits and what it is more important, a loss of intellectual capital. Unfortunately, this is often the case.

Another issue that complicates the maintenance process is the scarce documentation that exist related to a specific software system. For example legacy software from other units often has not documentation which describes the features of the software.

For an organization that deals with software maintenance it is vital to have a knowledge maintenance system which stores explicit knowledge and enables the organisation to own of intellectual capital and share it with the sub-units. Otherwise, the employees own this information and the company depends on them. Another advantage of using a knowledge management system is that it reduces the time that a person needs to mature professionally because it favours the professional development of employees and in this way increases the intellectual capital. With the passing of time, workers acquire knowledge which they do not normally pass on to the rest of the workers in the same area (let alone to workers in different areas). For this reason different solutions are often used in order to solve the same problem. Using a knowledge management system which acquires workers' knowledge and transmits it, the above mentioned situation would decrease since all workers could benefit from other employees' experience and the organisation would increase its expertise and coherence of information.

Having a knowledge management system the staff may also be informed about where information is. It is critical for software maintenance workers to have access to the knowledge the organisation has. Szulanski (1994) carried out a study which found that the number one barrier to knowledge sharing was "ignorance": the sub-units are ignorant of the knowledge that exists in the organisations, or the sub-units possessing the knowledge are ignorant of the fact that another sub-unit needs such knowledge. Sometimes the organisation itself is not aware of the location of the pockets of knowledge or expertise (Nebus, 2001). This fact has been summarised by management practitioners as "the left hand not only does not know what the right hand is doing, but it may not even know there is a right hand" (O'Dell and Grayson, 1998: 157).

A knowledge management system also help employees to have a shared vision, since the same codification is used and misunderstanding in staff communications may be avoided. Several studies have shown that a shared vision may hold together a loosely coupled system and promote the integration of an entire organisation (e. g., Orton and Weick, 1990)

The above explained issues motivated us to design a knowledge management system for obtaining, managing and transmitting knowledge in a software maintenance company, thus increasing the workers' expertise, and making easier their work since advises which decision must be made.

3. A Multi-agent System to Manage knowledge in Software Maintenance

The changeable character of the software maintenance process requires that the information generated to be controlled and stored. Thus it might be shared and besides inconsistency between different information could be detected. We propose in order to manage the knowledge generated during maintenance a multi-agent system where each agent is in charge of controlling a different kind of

knowledge such as, information about products, standards, criteria, processes, activities, tools, methodologies, projects and staffs' roles and skills.

The roles of the agent are:

- Capturing information related to the entities that they control
- Comparing new information with that which has already been stored in order to detect inconsistency between old and new information. If an inconsistency is detected the agent must inform the rest of the agents in order to discover why the inconsistency has occurred.
- Informing other agents about the new information received. Trying to ensure that all agents share their information and in this way avoiding inconsistencies and ensuring that information is always up to date.
- Predicting new client's demands. Similar software projects often require similar maintenance demands. What a company has done before tends to predict what it can do in the future (Gupta and Govindarajan, 2000).
- Predicting possible mistakes by using historic knowledge. Since, as Henninger and Schlabach (2001) claim, knowledge management avoids the repetition of common mistakes.
- Advising solutions to problems. Storing solutions that have worked correctly in previous maintenance situations helps to avoid the effect that Zell (2001) comments upon, indicating that due to the limited transfer of knowledge companies are forced to reinvent new practices, resulting in costly duplication of effort. The best practices often linger in companies for years unrecognised and unshared (Zell, 2001, pp 77).
- Helping to make decisions. For instance to evaluate whether it is convenient to outsource certain maintenance activities. When knowledge is enhanced it is easier to improve problem identification, development of alternative solutions and the selection of the best solution (Gnyawali, Stwart and Grant, 1997).
- Advising certain employee to do a specific job. The system has information about each employee's skills and about where and in what has each person worked. Agents may process this information to suggest which person is most suitable to carry out a task.
- Estimating the cost of future interventions. Information available may be used to make statistical analyses that help to predict maintenance effort and costs.

4. Conclusions

Software maintenance is one of the most important stage in the software life cycle. This process takes a lot time, effort and costs. Besides, it generates a huge amount of different kinds of knowledge that must be suitably managed. A system with different agents in charge of manage each kind of knowledge might

improve the process of maintenance since agents would help the staff to find information and solutions to problems and to make decisions, increasing in this way the organization competitive.

References

Card, D.M and Glass, R.L (1990) Measuring Software Design Quality. EE UU: Englewood Cliffs.

De Looff, L. Information Systems Outsourcing Decisión Making: a Managerial Approach. Hershey, PA: Idea Group Publishing, 1990.

Gnyawali, D.R., Stewart, A.C., and Grant J.H. (1997). Creating and Utilization of Organizational Knowledge: An Empirical Study of the Roles of Organizational Learning on Strategic Decision Making", Academy of Management Best Paper Proceedings, pp. 16-20.

Gupta, A., and Govindarajan, V. (2000). Knowledge Flows within Multinational Corporations. *Strategic Management Journal*, 21(4), pp. 473-496.

Henninger, S., and Schlabach, J. (2001). A Tool for Managing Software Development Knowledge, 3ª International Conference on Product Focused Software Process Improvement. PROFES 2001, Lecture Notes in Computer Science, Kaiserslautern, Germany, pp 182-195.

Kogut, B. and Zander, U. (1992) Knowledge Of The Firm, Combinative Capabilities, And The Replication Of Technology, Organization Science, Vol. 3; pp. 383-97.

Larson, L., Nidiffer, K.E., Rose, L.C., Small, R., Stankosky, M. (2001). Knowledge Management: Insights from the Trenches. IEEE Software, Vol. 18, No. 6, pp 66-68.

Nebus, J. (2001). Framing the Knowledge Search Problem: Whom Do We Contact, and Why Do We Contact Them? Academy of Management Best Papers Proceedings, pp h1-h7.

O'Dell, C., and Grayson, C.J. If Only We Knew What We Know: Identification and Transfer on Internal Best Practice. *California Management Review*, Vol. 40, No. 3, pp 154-174.

Orton, J.D., and Weick, K.E. (1990) Loosely coupled systems: A reconceputalization. Academy of Management Review, 15(2), pp 203-223.

Stewart, T. A., (1997) La nueva riqueza de las organizaciones: el capital intelectual, Ediciones Granica, Buenos Aires, 950-641-253-7.

Szulanski, G., (1994). Intra-Firm Transfer of Best Practices Project. American Productivity and Quality Centre, Houston, Texas.

Pigoski, T.M. (1997*). Practical Software Maintenance. Best Practices for Managing Your Investment*. Ed. John Wiley & Sons, USA, 1997.

Zell, D. (2001) Overcoming Barriers to Work Innovations: Lessons Learned at Hewlet-Packard. Organizational Dynamics, 30 (1), pp 77-86.