Constructing a Knowledge Map for a Software Maintenance Organization

Oscar M. Rodríguez-Elias¹, Ana I. Martínez-García¹, Aurora Vizcaíno², Jesús Favela¹, Mario Piattini²

¹CICESE, Computer Science Department, Ensenada, B.C., Mexico {orodrigu | martinea | favela}@cicese.mx
²University of Castilla-La Mancha, Escuela Superior de Informática, Ciudad Real, Spain {Aurora.Vizcaíno | Mario.Piattini}@uclm.es

1. Introduction

Software maintenance requires lots of knowledge. Maintainers must know what changes should do to the software, where to do those changes and how those can affect other modules of the system. Frequently they do not have enough knowledge to make the best decision and must consult other information sources, but these are often unknown or difficult to locate. A knowledge map can help to easily find sources that can be used to obtain the information or knowledge required to perform a specific task; since these maps can be used to point to the sources of specific information or knowledge [1]. This paper presents a work where, qualitative and theoretical research has been applied to develop a knowledge map for a software maintenance organization. This map has been used in the development of a knowledge management prototype that could help software maintainers searching for knowledge and information sources to do their jobs.

2. Identifying the knowledge required by maintainers, and its sources

To build the knowledge map, we investigated and identified the main topics of knowledge that software maintainers require to do their jobs. First we carried out a case study in a software maintenance group to understand the processes and activities performed by the group, the knowledge they require to do their activities, and the sources they use to obtain that knowledge. Then, we performed a bibliographic research to compare our findings to define a more general classification schema for types of knowledge and its sources. The literature review was based on research papers, such as case studies, software maintenance ontologies, standards for software engineering and maintenance, and the SEWBOK [2]. However, we used the case study to focus and establish the basis of a practical classification for the group studied. Next, we describe how the knowledge map was developed.

3. Constructing the knowledge map

To construct the knowledge map we first defined a classification schema and structure the types of knowledge and knowledge sources. The classification schema was used to define a metamodel which describes the relationships of the knowledge subjects and its sources. Then we defined a template to describe specific knowledge subjects and sources. These descriptions were used to construct the knowledge map by representing knowledge subjects and sources into a XML format. Next, we present this development.

3.1 Classification of the types of knowledge and their sources

Knowledge subject classification

The classification of knowledge subjects was done following a schema that consists of three levels of abstraction: categories, knowledge areas and knowledge subjects. At the first level are categories, which are structural elements of high level abstraction used to classify related areas of knowledge. A category can also contain more specialized categories. In the second level are the knowledge areas, which are subdivisions of the categories that are logically related with them, for example by aggregation or composition. An area can contain more specialized sub-areas or knowledge subjects. The subjects represent basic concepts with an explicit and well defined description. They are used to describe knowledge about a set of elements that can be considered as a unit. However, a subject can also be composed by more detailed subjects. Following the schema just described, we defined some general areas
of knowledge grouped into three main categories: (I) the knowledge related to the software maintenance process; (II) the knowledge required for the organization’s life; (III) and the general knowledge that is not part of the other two categories.

The knowledge category of the maintenance activities is the most important, and most of the areas of knowledge are grouped here. This category is composed by three subcategories: 1) Computing fundamentals contains areas of general knowledge about computing, such as operative systems, programming languages, etc.; 2) Software engineering considers the knowledge related with the phases of the software development life cycle, such as project management, analysis and design, etc.; and 3) Application knowledge category, groups the knowledge related with the applications maintained by the team, such as specific knowledge about the products, for instance the architecture, structure, functionality, history, etc.; and the knowledge of the domain that the applications support.

The organization’s life knowledge category considers the knowledge that is not directly related to the activities of software maintenance, but that all employees must know, such as the structure, norms and policies, goals, etc. of the organization; and knowledge about other processes followed by the organization.

Finally, in the general knowledge category, the knowledge and skills that are not part of the daily work, but can be useful for special purposes, are considered. For example, foreign languages speaking and writing, group work coordination, leadership, etc.

Knowledge sources classification

A schema composed of categories and types of sources was used for classifying the sources of information and knowledge.

Sources of knowledge were divided into four categories: (I) documentation, (II) people, (III) the maintained systems’ elements, and (IV) support tools. Next we describe each of these categories.

Documentation category groups all the kinds of documents that can be used by the maintenance group. These documents were classified into six main types: 1) System documentation, 2) Technical documentation, 3) User documentation, 4) Organizational documentation, 5) Maintenance process documentation, and 6) Other documents.

The people category refers to all the persons that are consulted by the members of the maintenance group. This category has been divided into three: 1) Users/Clients (Even though users and clients play different roles, we have decided to take them as a single category since in the group studied there is not a clear separation between the roles played by users and clients [5]). 2) Staff members are all the persons working in the maintenance group; and 3) Other experts refers to all the persons that are not staff members or users, but that are consulted by maintainers to obtain specialized knowledge, such as knowledge about the application domain, a specific programming language, etc. These experts can be either internal or external to the organization. For example, some maintainers consulted friends that are not in the organization, or consulted experts through internet newsgroup, email lists, etc.

The system category refers to all the elements that constitute the products that are being maintained and that can be sources of information and knowledge. These elements have been divided into three types: 1) Executable system, 2) Source code, and 3) Data bases of the systems maintained.

Finally, the support tools category is concerned with all the tools used by maintainers to obtain information or knowledge. These tools have been divided in two types: 1) Maintenance activities support tools are those used for supporting activities of the maintenance process; and 2) General support tools are those that are not directly related to the maintenance activities. For example, organizational memories or portals, content management systems, document repositories, etc.

Figure 2 presents and example of the knowledge sources classification schema; where two types of documents with information directed to the users of the applications maintained (the user and installation manuals) are classified as user documentation; which is a subdivision of the documentation category.
3.2 Describing knowledge subjects and their sources

To describe knowledge subjects we followed the template exemplified in Table 1. In this template, each subject is identified by a name and a short description. Then, the main cognitive and technical knowledge related to the subject are defined. Cognitive knowledge refers to know what, for example, which activities we must do, what information is required to do these, where we can find that information, etc. Declarative or cognitive knowledge can be divided in two types [4]: 1) topic knowledge, that refers to knowledge about concepts, their definitions, properties, and relationships; and 2) episodic knowledge, that represents the experiences on the use of knowledge. Finally, procedural or technical knowledge helps to know how an activity should be done.

The example of Table 1 refers to the knowledge related to an activity. The topic knowledge items can be used to identify sources that can help to obtain information about the topics defined. The episodic knowledge items establish the situations that can cause generation of knowledge related to the subject. These definitions can be used to identify people that have been involved in one or more of these situations. Procedural knowledge definitions can be used to identify sources of knowledge that can be useful to obtain information about how to do something related to the subject.

Table 1. Example of a knowledge subject description.

<table>
<thead>
<tr>
<th>Knowledge subject</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performing modifications to the check bills elaboration module in the finances system</td>
<td>Knowledge about the check bills elaboration process in the finances system, and about how to modify the module</td>
</tr>
<tr>
<td>Topic Knowledge</td>
<td>Which is the module of check bills elaboration</td>
</tr>
<tr>
<td>Which are the source files of the module</td>
<td></td>
</tr>
<tr>
<td>Related modules</td>
<td></td>
</tr>
<tr>
<td>Which is the programming language used</td>
<td></td>
</tr>
<tr>
<td>Episodic Knowledge</td>
<td>Experience using the check bills elaboration module</td>
</tr>
<tr>
<td>Experience developing the check bills elaboration module</td>
<td></td>
</tr>
<tr>
<td>Experience modifying the check bills elaboration module</td>
<td></td>
</tr>
<tr>
<td>Procedural Knowledge</td>
<td>How to access the module</td>
</tr>
<tr>
<td>How to make changes in the module</td>
<td></td>
</tr>
<tr>
<td>How to identify problems in the module</td>
<td></td>
</tr>
<tr>
<td>How to correct problems in the module</td>
<td></td>
</tr>
</tbody>
</table>

The sources of information and knowledge are described by templates as the showed in Table 2. Each source has a unique identification (id), a short description, and it is classified in a category and a kind. Each source can be consulted, at least, in one location. Loca-
The systems are defined by its type and a description that is used to provide specific access data. Depending on the location, the source can have a physical support (such as paper, video tape, CD, etc.) and a format; for example, the source described in Table 2 is an electronic file, a word 2000 document. Finally, the main information and knowledge subjects that can be obtained from the source can be specified into the “knows about” list. This information is later used to identify in which activities the source can be helpful.

<table>
<thead>
<tr>
<th>Source id:</th>
<th>p1230_requerimientos.doc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>Documentation</td>
</tr>
<tr>
<td>Kind</td>
<td>System documentation / Requirements</td>
</tr>
<tr>
<td>Description</td>
<td>Document containing the requirements specification of the finances system SIREFI</td>
</tr>
</tbody>
</table>

The prototype of the KM system is based on a multi-agent architecture where there is a staff agent that plays the role of assistant of a member of the maintenance team [5]. When the staff agent detects that the maintainer is performing an activity, it tries to identify the knowledge required by that activity. For example, if the maintainer wants to solve a problem reported, the agent obtains information from the problem report, such as the system and the module where the problem appeared, the type of problem, etc.; then, it tries to infer what knowledge can be required to solve that problem; for example, which are the source code files of the module, where they are, etc.

Once the agent finishes defining the list of subjects of knowledge, it starts searching for knowledge sources that could have information about the subjects defined. When the search is finished, the agent informs the user that there are sources of information that can be relevant to the activity being done. If the user decides to consult those sources, the system shows a window with the sources found, grouped by types. When the user chooses one of the sources, the system shows information such as how or where that source can be consulted, and the main subjects of knowledge related to the activity, that can be obtained from it.

The prototype and the preliminary knowledge map where tested following scenarios obtained from the case study carried out [5]. The knowledge map was developed with information obtained from the group studied, and represents real situations.

### 5. Conclusions and future work

Finding methods and tools that help software maintainers reduce the time needed to do their jobs can provide major benefits to software organizations; for example, by helping maintainers reducing the time they spend searching for sources of information to obtain the knowledge they need to perform their jobs. In this paper we presented how we developed a knowledge map for a software maintenance team, by classifying the main knowledge required by the members of the team and the sources of information available. This map also helps to identify where that knowledge and sources can be required by defining the relationships between the types of knowledge, the sources, and the main activities performed by the team.

We have used the knowledge map in a prototype of a KM system. However, more research must be done to measure how useful the map could be in a real environment. In order to make that research, a more complete knowledge map should be developed and adapted to the maintainers’ work environment.

### Acknowledgements

This work is partially supported by CONACYT under grant C01-40799 and the scholarship 164739 provided to the first author, and the MAS project (grant number TIC2003-02737-C02-02), Ministerio de Ciencia y Tecnologia, SPAIN.

### References


