A Framework to Analyze Information Systems as Knowledge Flow Facilitators

Oscar M. Rodríguez-Elias[a,*], Ana I. Martínez-García[a], Aurora Vizcaíno[b], Jesús Favela[a], Mario Piattini[b]

[a] *CICESE-Computer Science Department, Km. 107 Carretera Tijuana-Ensenada, Ensenada, B.C., Mexico, 22860*

{orodrigu | maritinea | favela}@cicese.mx

[b] *University of Castilla-La Mancha, Escuela Superior de Informática, Paseo de la Universidad No. 4, Ciudad Real, Spain, 13071*

{aurora.vizcaino | mario.piattini}@uclm.es

Manuscript accepted to be published in the Information and Software Technology Journal

Available in: http://dx.doi.org/10.1016/j.infsof.2007.07.002

A Framework to Analyze Information Systems as Knowledge Flow Facilitators

Abstract

This paper presents a framework which can be used to analyze information systems as knowledge flow facilitators in organizational processes. This framework may be useful, particularly to small organizations, for two main reasons: it can help them to start seeing the implications of KM in their current technical infrastructure, and as a result, they should be in a better position to know how to include their current working tools in part of a KM strategy, thus facilitating the alignment of such a strategy to the daily work of the organization. Secondly, identifying the role that their current tools play in the flow of knowledge should help such organizations to identify means by which to improve such tools as KM enablers, before becoming engaged in costly KM efforts that could require the acquisition of new tools and often also big changes in their current work processes. The applicability of the framework is illustrated with a case study conducted in a software development environment in which it was successfully applied.

Keywords: *Knowledge Management Systems, Analysis of Technologies, Knowledge Management in Software Engineering, Knowledge Flow.*

1 Introduction

Knowledge Management (KM) provides methods and techniques which help organizations to process and re-use their knowledge. It can be defined as a systematic discipline and set of approaches to enable information and knowledge to grow, flow, and create value in an organization [60]. Although technology is not the main objective of KM initiatives, it is an important enabler for many of them [2], playing an important role for such initiatives to be successful [46]. Because of this, various frameworks exist which can be used to analyze and evaluate KM Systems (KMSs). Their goal is mainly to classify the KM tools which are available on the market.

Organizations frequently possess tools which have the potential to become important knowledge flow enablers [14]. In fact, many technologies or tools initially developed for other purposes, such as for supporting collaborative work, decision making, etc., have been adopted as KMSs [16, 52, 60]. However, often in organizations certain technologies are not used as knowledge flow facilitators because these organizations do not realise that these tools can be used to manage or disseminate their knowledge. Thus, analyzing whether a tool supports specific KM activities may be a starting point for integrating such a tool into a KM strategy.

On the other hand, for a KM initiative to be successful, it is important to take into account the work processes of the organizations and their current technical infrastructure [44, 51, 74]. This is particularly true for small and medium size enterprises which often do not have enough resources to be able to experiment with new tools and processes to integrate

KM into their activities. Among the factors that small companies should consider is that an inappropriate integration of KM initiatives into their current organizational culture may be a barrier to its adoption [85]. Small organizations should ensure that their KM initiative fits into their current organizational culture [24], since it is highly probable that they will not have the time or the resources to become engaged in a long term cultural change effort. Therefore, before developing or acquiring new KMSs, small companies should start seeing their current activities in terms of knowledge; they should start considering KM as a concept with implications in their current systems [72]. Additionally, KM projects are often more likely to succeed if organizations take advantage of the current infrastructure [24]. By using the above as a base, we have defined the following research question to guide our work: how can we help an organization to identify the role that their current working tools might play as KM facilitators? We are additionally interested in discovering whether showing an organization the role that their current tools play as KM facilitators can help them to start a KM initiative. We state that one way in which to convince small organizations to adopt KM activities in their processes is to help them to identify the role that their current tools play in such activities.

In this paper we propose a framework that can be used as a guideline to analyze information systems from the point of view of their contribution as KMSs. We also present a case study which exemplifies the application of the framework in the domain of software development. This case study shows the way in which a logbook (used by a software maintenance team to manage the modification requests sent by the end users of the systems they maintain) can be also used as a knowledge flow enabler for the software maintainers. The content of the paper is organized as follows: first, in Section Two we introduce current work in the field of KM in software organizations, in order to position our work within such efforts and to describe the goal of our research. Later, in Section Three we present a review of the characteristics which are used to classify and analyze KMSs. In Section Four we introduce the framework which was developed as a guideline to analyze tools and technology as knowledge flow facilitators. In Section Five we present an example through which to illustrate the use of the framework. Next, in Section Six some discussions and lessons learned from the application of the framework are shall described, and finally our conclusions are presented in Section Seven.

2 Knowledge management in software organizations

Traditional approaches for managing knowledge in software organizations require the use of staff who are in charge of packaging knowledge or experiences, maintaining an experience base, and supporting software projects in identifying and using appropriate experiences [7, 8]. Since such staff must be separated from the developers' staff, this may entail the need for resources that might not be easily available to small companies, although such approaches have been successfully applied in large software companies (see for instance [30, 69]). The latter point has motivated some researchers to propose lightweight approaches through which to help small organizations to adopt experience reuse. For instance, some approaches focus upon the use of postmortem analysis to capture experiences, either at the end of development projects [29, 31], or during maintenance projects [4, 26].

It is clear that small software companies are different to the large ones in many aspects [62]. For instance, they have fewer resources which are often not sufficient for them to be able to engage in a novel KM approach that requires new tools, processes, training, etc., or that require having staff in charge of creating, maintaining, capturing, etc. a base of experiences or knowledge. Mature KM initiatives for software organizations require a well established software process with a measurement program that enables the company to obtain data which is useful for measuring how they are working and how they can improve the quality of both the product and the process [7]. Unfortunately, many small software companies have not only not adopted standard processes and measurement programs, but have also not defined their own processes well [36], and these can sometimes seem chaotic [39]. All this makes it harder for such organizations to adopt traditional KM practices.

On the other hand, as has been observed by Land et al. [48], KM efforts in software organizations commonly focus upon managing explicit knowledge outside the development process, while important knowledge involved in the development processes is not explicitly managed. Thus, identifying the form in which tools used to support software process are involved in the flow of both explicit and tacit knowledge could help to identify means by which to make the management of the knowledge in such processes explicit. Some studies have shown that even in software companies which do not have explicit KM practices, software engineers implicitly apply some kind of KM in their daily work [54, 81]. Therefore, before engaging in a costly KM effort, small software companies should first become aware of the implications of KM in their current systems [72] by, for instance, knowing the role that these tools play in the KM activities. Our framework attempts to be of assistance in the latter aims, since it is orientated towards accomplishing the following four goals:

- Goal 1: the main goal is to identify the role that tools, currently used in a company's processes, play as knowledge flow facilitators in such processes. This is to help organizations to start seeing their current tools from a KM perspective, thus enabling them to initiate KM efforts by considering their current technical infrastructure as part of such efforts.
- Goal 2: to identify requirements to improve the use of such tools as knowledge flow enablers.
- Goal 3: to identify further usage of such tools in either other processes or activities.
- Goal 4: to compare tools with others which could support similar activities, from the point of view of their contribution to the management of knowledge.

These goals can help small software companies to recognize the importance of using KMSs and how everyday tools can be used as means of capturing or disseminating knowledge. Therefore, making their current KM practices explicit could contribute to the improvement of their processes [54]. The following section presents a review of the characteristics that were used as a basis for the development of the proposed framework, which is described in Section Four.

3 A review of the characteristics used to analyze KMSs

After having reviewed different types of KM frameworks, we have identified certain issues that should be considered when analyzing KMSs. This revision does not pretend to be a complete research survey, since we focus on technical and practical approaches, rather than on theoretical approaches. For a theoretical review of KM frameworks the reader can consult [43], and [34, 45, 71] for KM evaluation frameworks. Although we have principally used proposals for general KMSs to identify the characteristics that have been used to analyze KMSs in different domains, we have also chosen proposals from other perspectives, such as a classification of specific application domain tools, particularly for software development [68], and specific types of KM tools, such as groupware or collaboration technologies [59, 80], web technologies [77], and agent-based KMSs [79].

In order to develop our framework, we have identified four main aspects that should be considered when analyzing KM tools: 1) the application domain of the tools; 2) the structure or form of the knowledge managed; 3) the KM services, processes, or activities supported; and 4) the technical issues which might be considered important in particular situations. Table 1 presents the various frameworks that we have reviewed and considered in our work. The table shows the purpose for which each framework was proposed or used, the approach that the frameworks apply to that purpose, and which of the four aspects previously mentioned are covered (either explicitly or implicitly) in each framework.

[Table 1 around here]

The aspects covered by each framework are ordered according to the importance given to each one in such an approach. It is important to mention that each framework manages these aspects from different points of view and with different purposes. Moreover, in certain proposals some are only managed implicitly. The only proposal that is similar in most of the frameworks is that which is related to the KM activities. Moreover, the only work that partially covers the four is that of Woitsch and Karagiannis [84]. However, in this work all the information is managed in a single vector, and one issue is not distinguished from the rest in the same depth as in our work. The four aspects identified in this review led us to the definition of four main steps through which to analyze KMSs, which are described in the following subsections.

3.1 To define the application domain

KM technologies and tools must consider what is important for knowledge workers [23: p. 21], since KM strategies must be orientated towards facilitating the activities that such workers must perform [83]. KM strategies should identify where the knowledge is or where it will be applied, i.e. the process in which it is used or its users [3, 38]. The KMS which an organization needs depends on aspects related to the type of organization [46]. Therefore, the application domain of KMSs is a key characteristic in the analysis of such systems. Identifying the application domain of a KMS is also an important step towards its evaluation, which includes the identification of: how knowledge is being used, where, when, by whom, etc. [71]. Moreover, we consider that taking into account the work processes supported by and the people who benefit from or are affected by the KMSs may help to define the organizational context, which is an important aspect in the identification of whether a tool is aligned to the organization's strategies or processes [11].

3.2 Defining the type and structure of knowledge

Defining the type and structure of knowledge refers to the identification of the class of knowledge managed and the way in which it is structured. Knowledge in organizations can be classified from different points of view [2]. That which is most frequently mentioned in KM research considers knowledge as being both tacit and explicit [58]. Most KMSs focus upon managing explicit knowledge [5, 16, 60, 76]. This type of knowledge can be structured as a variety of levels of formality, in which the more formal levels are easier to manage by automatic means [78]. Therefore, considering the type and structure of knowledge is an important characteristic when analyzing a tool as a KMS [78].

3.3 Defining the KM activities or knowledge flow stages which are supported

Since the main objective of KM is to facilitate the flow of knowledge from where it is created or stored, to where it is applied [16], in this paper we consider the KM activities as being the different stages required to facilitate this knowledge flow such as, for instance, capturing knowledge to assist in its future usage, or providing methods through which to facilitate the application of the organizational knowledge. The main characteristic used to analyze KMSs is that of checking to what degree systems support specific KM activities [2]. In fact, most of the frameworks consider this dimension as the main characteristic through which to analyze or evaluate KMSs. However, most of the approaches differ on which of the KM activities should be considered.

Certain other issues are also important to specific KM activities such as, in the case of knowledge transfer, considering a spatio-temporal dimension [6]. This means, specifying whether the KMSs consider the location or distribution of givers and takers of knowledge, and whether those KMSs enable the takers of knowledge to access the givers of knowledge in a synchronous or asynchronous manner [6]. Rao [61] proposes a matrix with which to classify KMSs from these two dimensions.

3.4 Defining the main technical issues

Certain technical issues should be taken into account, since traditional KMSs have limitations that prevent their use in some domains in which these aspects are not considered [13, 79]. For example, in software organizations, one of the problems that prevents the use of traditional KMSs is that they require users to spend much time capturing or searching for knowledge [27, 50]. Therefore, once the way in which a tool fits to a particular application domain, the type of knowledge the tool manages, and the manner in which it supports specific KM activities have been identified, the next step is to identify the main technical issues to be considered, and the form in which the tool supports them. In the following section we present guidelines which may be used to perform these four steps.

4 A framework to analyze tools and technologies as knowledge flow facilitators

Using the four aspects described in the previous section as a base, we have defined a framework with which to analyze support tools and technologies used in organizational processes, from the point of view of their contribution as knowledge management facilitators. The framework is summarized in Figure 1. As can be observed, it is based on four stages: 1) the identification of the application domain of the tools being analyzed; 2) the identification of the structure of the knowledge managed by the tools; 3) the KM activities supported by the tools; and 4) the identification of main technical issues. We shall now describe each stage.

[Figure 1 around here]

4.1 Application domain

The definition of the application domain of the tools being analyzed is orientated towards three aspects: 1) the use of knowledge, which is related to the organization's activities or processes in which the tool is used, 2) the scope of knowledge, which defines the people that might benefit from using the tool, and 3), the domain of the knowledge managed.

4.1.1 The use of knowledge

An information system can support different activities, some of which are common to every organization such as customer relationships, skill and competence management, etc. Others are specific to certain types of organizations, such as CASE tools or software configuration management which are designed for software organizations. In order to describe the process in which a KM tool will be used, we have defined certain categories which are shown in Table 2. However, we recommend the explicit description of which processes and activities are supported by the tool being analyzed.

[Table 2 around here]

4.1.2 The scope of knowledge

The scope of knowledge refers to the range of people that use or might benefit from using the knowledge managed by the tool. Some tools manage knowledge for personal use [75]. Some knowledge is useful to a specific organizational group, such as a department, or for groups formed of people inside and outside the organization. Finally, there is knowledge that is useful to the whole organization, for many organizations, or to industry in general [68]. Table 3 presents a set of levels that can be used; these are an extension of the scope dimension used by Rus et al. in [68].

[Table 3 around here]

4.1.3 The domain of knowledge

The domain of knowledge managed by a specific tool may be directly related to the processes and activities for which the tool is used. However, it is important to give an explicit description of the domain of the knowledge or information; for example, whether the tool manages information about the products or services provided by the organization, about its employees, its customers, etc. Table 4 shows an example of a classification schema that can be used here. Nevertheless, once the domain of knowledge has been classified, it is importanat to describe it explicitly.

[Table 4 around here]

4.2 The structure of knowledge

In order to define the structure of the knowledge managed, we have followed an approach which extends that defined by Valente and Housel [78]; this extension is presented in Table 5. In this table, knowledge is divided into different forms, based on tacit and explicit dimensions. Tacit knowledge corresponds with personal knowledge, that which is used by individuals to perform their activities (technical dimension), or to perceive their world (cognitive dimension) [57]. In this paper we focus upon the technical part of tacit knowledge, that which can be grouped as "know how". Technical skills are those that make a person an expert in a specific domain in an organization. Although a part of such technical knowledge could be articulated and made explicit relatively easily (for instance procedures, techniques, etc.), there is another part which depends upon the individual, and is therefore difficult, and some times impossible, to formulize. For instance, a person might know which procedure to use to perform a specific activity, but this is not the same as having the experience which helps an expert to know how to adapt such knowledge to exceptional situations that have not been previously faced. We have divided these two forms of technical knowledge into: "know-how", and skills and abilities respectively.

On the other hand, explicit knowledge is that which has been formulized and stored in formal means, and may have different levels of structure, from unstructured, such as free text, to highly structured, such as information represented in a mathematically rigorous form. In Table 5 we have made a distinction between the two levels of explicit knowledge: unstructured, and structured. To better define the degree to which explicit knowledge is structured, the representation format of that knowledge can be used. Table 5 also presents some examples of formats that could possibly be used; however, these can be extended or modified depending upon needs.

[Table 5 around here]

It is important to take into account the fact that some tools use more than one of the knowledge structure representations and formats which are defined in Table 5. For example, some XML based content management systems manage documents by storing information (metadata) about them in XML, and use this representation to classify them in a document repository. These systems can also provide search engines which index those documents by using both free text and metadata [37].

4.3 Knowledge flow support

In this phase the KM activities supported by the tools being analyzed are identified, along with certain characteristics related to each stage. Many models of KM activities, processes, or lifecycles have been proposed (see Table 6). Some of them (Holsapple and Joshi [43], Nissen [56], Qureschi et al. [59], and Dalkir [23]) are based on comparisons of various KM lifecycles. These lifecycles are composed of different stages, some of which are more detailed than others. Most of these models, with the exception of that of Dalkir

[23], present the KM activities in a form in which they are separated from each other. Holsapple and Joshi [43] argue that to have a common understanding of KM, it is necessary to see the different KM activities as interrelated entities, and not as entities that are independent and isolated from each other. This theory has prompted us to analyze the proposals shown in Table 6, and to define a model that shows the relationships between the different KM activities, as part of a continuous flow, shown in Figure 2.

[Table 6 around here]

The model of KM activities shown in Figure 2 is based on the knowledge creation model proposed by Nonaka [57], which shows the mechanisms used to convert knowledge from tacit to tacit (socialization), from tacit to explicit (externalization), from explicit to explicit (combination), and from explicit to tacit (internalization). The model in Figure 2 is presented to show the interconnections between the various KM activities, and particularly the manner in which those activities contribute to the knowledge application and transferring stages, since these could be considered to be the most important KM activities. For organizations, the value of knowledge is in its application [3], and KM systems and strategies must facilitate the flow of knowledge to where it needs to be applied [15]. For this reason, we use six aspects to analyze the role that a specific tool plays in the knowledge flow. These are: knowledge creation, transference, storage, and application. We also use externalization and internalization to group the activities that are involved in the conversion of tacit knowledge into explicit knowledge, and vice versa.

[Figure 2 around here]

4.3.1 Knowledge creation

This process consists of the activities that permit the generation of knowledge by individuals or by the entire organization, or its acquisition from external sources. To analyze technologies which support knowledge creation, we suggest taking both generation and acquisition as separated entities. It will therefore, be possible to explicitly define tools that support knowledge generation by, for example enabling brainstorming, and those that support knowledge acquisition, such as, for example, an online technical course system.

4.3.2 Knowledge externalization

Knowledge externalization or formalization includes activities which are focused on articulating, structuring, representing, codifying, organizing, etc. knowledge, to facilitate its management by expressing it in formal sources that can later be stored in repositories, databases, etc. This stage is often critical in organizations, since employees frequently do not use KM tools because they spend too much time attempting to make their knowledge explicit [27, 50]. Moreover, it can be difficult for some employees to know how to express their ideas. It is thus convenient to identify tools that may facilitate the externalization of employees' knowledge.

4.3.3 Knowledge storage

Once knowledge has been externalized, it must be stored. At this stage, it is extremely important that the scope of the stored knowledge, along with its type and structure, are appropriately and explicitly defined. The maintenance of stored knowledge is an important factor to take into account at this stage. Knowledge should be updated to avoid confusions caused by using obsolete knowledge. This may be static or dynamic [84]. There is knowledge that does not change over a long period of time, but there is also knowledge that is constantly changing. Moreover, if we have distributed and replicated knowledge, it is important to know where the most updated version is, and to gain control of the changes made to the different versions. Aditionally, there is knowledge with temporal relevancy, which once used may no longer be useful, but there is also knowledge that can be useful over long periods of time [46]. It is therefore important to define whether a tool provides characteristics to support knowledge maintenance, perhaps by identifying its temporal usability, and by facilitating its updating, the identification of obsolete knowledge, the management of its versions, etc. [60].

4.3.4 Knowledge sharing and transferring

Knowledge sharing and transferring consists of the activities which focus on transferring, disseminating, deploying, diffusing, etc. knowledge between individuals, groups, or organizations. This stage should be seen from the two perspectives of explicit and tacit knowledge sharing. Explicit knowledge can be shared by storing it in shared repositories, databases, etc. From these media, explicit knowledge can be used by different people either within or even outside the organization. Explicit knowledge sharing may also consider mechanisms through which to provide information about the knowledge that is being stored, thus enabling the diffusion of new knowledge. On the other hand, tacit knowledge sharing takes place in socialization processes, and can therefore be supported by communication and collaborative technologies. Through socialization, people can both increase their own experience and share that experience with others. Socialization can be promoted by, for example, creating communities of practice [82].

4.3.5 Knowledge internalization

Internalization of knowledge can be achieved through various activities. First, explicit knowledge must be retrieved from where it is stored. The mechanisms that support this process include search engines to retrieve documents in repositories, database query systems, etc. Information retrieval systems can help to obtain information from local sources, such as the user's workspace, from central repositories or databases, or from distributed sources inside or outside the organization. Once the knowledge has been retrieved, its users must interpret it and filter out what is important for them. This process facilitates the internalization of the explicit knowledge or information available in formal sources. However, providing technologies to assist the users in this process can be a challenging job [52]. If the user has too much information or too many available sources, identifying that which is the most useful for a particular purpose may be a complex task. Therefore, providing tools to assist the user to carry out these tasks can be very valuable. This can be achieved by, for example, providing personalized knowledge or information, based on the interests, profile, etc. of the users of such knowledge.

4.3.6 Knowledge application

Knowledge application is integrated into the other stages, because when people perform an activity, make a decision, etc., they must obtain or create the knowledge they require by, for example, retrieving it from a database, a document, etc.; or by consulting colleagues or other experts. On the other hand, the application of knowledge can generate new knowledge which is valuable to the organization and which should be formalized and stored by, for instance, writing a report about the lessons learned while a person was solving a problem. Additionally, some of the most valuable knowledge can be generated during this stage, because it is by applying knowledge that people obtain experience and become expert. Thus, by its application, knowledge can evolve and grow. Tools that support the knowledge application stage should also support other stages of the KM lifecycle.

4.4 Technical issues

Technical aspects frequently lead to the situation of KMSs being unused in organizations [74]. Therefore, it is important to identify such aspects in order to analyze whether a specific KM tool supports them. In our case we have identified two main aspects which are of special interest if we are to address some of the problems in traditional KMSs. These are: to reduce the users' work by providing automatic support at certain stages of the KM lifecycle [52, 79], and to facilitate the management of distributed knowledge [6, 13, 21, 79]. Frequently, important knowledge distributed exists in an organization that is not well managed because it is not stored in central repositories, for instance documents stored in employees' workspaces, thus making it difficult for others to access or find them. As it has been pointed out in [13] and in [79], traditional KMSs are not designed by taking the distributed nature of knowledge in organizations into consideration. Moreover, traditional KMSs often require much work on the part of the users, perhaps because it is necessary for users to input their knowledge into the system, or because it is difficult for users to search for specific knowledge. We encountered these problems in a software development group studied [66], and this seems to be common problems in the field [27, 49, 50, 70]. Having taken all this into consideration, we agree that it is important to consider whether a tool supports these two aspects. We have therefore defined a set of values with which to analyze the level of autonomy of tools, and the distribution of the knowledge managed, see Table 7. These two aspects are described in the following subsections.

[Table 7 around here]

4.4.1 Level of autonomy

An autonomous system is one capable of sensing the environment in which it is embedded, and acting on this in pursuit of its own goals. Such types of systems are often called autonomous agents [35]. The definition of agents' autonomy is too general and makes it possible for these types of systems to have different types and levels of autonomy [9, 17, 19, 42]. In our case we are interested in user-autonomy which is to say that a system is autonomous if it can decide which actions to perform without the user's intervention [19]. However, although some activities may be carried out without the user's intervention, others may require the user's action. Therefore, this type of autonomy can change from a complete dependency, to a complete independency from the user [19]. Hexmoor, in [41], has proposed five levels of autonomy that can be used to measure user-autonomy:

1) *Not-autonomy* implies that the system does not have any type of autonomy.
2) *Semi-autonomy* means that the system's autonomy depends on external constraints such as the user's intervention if it is to act, i.e. whether the user must indicate to the system exactly what to do and when to do it, or provide all the information required for a specific activity.
3) *Shared-autonomy* means that there is a co-participation of the system and the user for the achievement of a goal. For instance, the system may assist the user by suggesting information to be considered or actions to be carried out, but the user decides whether or not to use that information or perform those actions.
4) *Delegated-autonomy* means that the system directs the achievement of a goal, but may delegate some decisions to the user.
5) *Full-autonomy*. This was originally called self-autonomy in [41]. We have, however, called it full-autonomy in order to differentiate it from the self-autonomy concept used in [19]. This means that the system has self-control, and performs all or most of the actions required to accomplish a specific goal.

To illustrate how the above levels can be used to define the autonomy of a tool, we now present a sample case, which is divided into various possibilities for a search system which are focused on helping to obtain information or knowledge which may be useful in accomplishing a particular task. A traditional *non-autonomous* search system requires the user to decide the time, the place, and the knowledge to search for by, for instance, introducing keywords or queries related to the information or knowledge s/he wishes to find. A *semi-autonomous* system may be one which is capable of suggesting some keywords or queries that can be related to those introduced by the user, to help her/him to redefine the search parameters in order to find more accurate information or knowledge. Let us now suppose that the user requests the system to start searching for information related to the activities being carried out, but that the system is capable of knowing the context of the user (for example the task s/he is doing) to formulate the queries or keywords in order to be able to start searching; this case could be defined as a *shared-autonomous* behavior of the system. Now let us suppose that once the search has started, the system identifies that there are different types of sources (types of documents, people, lessons learned reports, etc.) or that they proceed from different places, and decides to ask the user which sources are more important; this case could be considered as a *delegated-autonomy*, since the system delegates some decisions to the user to improve the results of the search. Finally, if the system is capable of identifying the knowledge needed by the user and starts searching before this is requested, and it is also capable of managing the user's history and profile in order to infer which sources might be more important for her/him, then that system might automatically rank the sources and present those that may be more relevant to the user, performing all the search without the user's intervention and this system could therefore be ranked as *full-autonomous*.

4.4.2 Distribution of knowledge

Distributed KM is a research area which has emerged to deal with the problem of managing the process of creating and exchanging local knowledge within autonomous groups [13]. Knowledge in organizations is distributed by its very nature; it resides in diverse sources that may be distributed throughout the entire organization. Moreover, some of these sources may be located outside the organization, or be geographically disperse throughout the world, for example in the case of transnational companies [25]. Traditional KMSs tend to use centralized repositories of all kinds of knowledge sources [47] to manage and control knowledge in a centralized manner [13]. However, not all knowledge can be managed in a centralized way, and not considering the distributed nature of knowledge might cause users to consider that a system is useless [13, 79]. Knowledge usefulness may depend upon where it was created or obtained or where it will be applied [13, 25: p. 41]. It may, therefore, be important to know whether a KMS enables the management of knowledge in a centralized or a distributed form.

The meaning of distributed knowledge depends upon the definition made for local knowledge. The latter may be knowledge which belongs to a single person, a group or unit of a company, an entire company, and so on. Therefore, in order to define whether or not a KMS manages centralized or distributed knowledge, we must first define the limits of the *knowledge nodes* embedded in the environment in which the system is situated or will be used [21]. "*A knowledge node represents a knowledge owner within the organization, namely an entity (individual or collective) which has the capability of managing its own knowledge both from a conceptual and a technological point of view*" [12]. Once the boundaries of the knowledge nodes of the organization have been defined, we can then identify whether the knowledge is managed in a centralized or distributed form from the point of view of the knowledge nodes. To do this, we propose a set of five values:

1. *Local*. The knowledge of each knowledge node is created, stored, managed, or

used locally to that node.

2. *Centralized-internally*. The knowledge is created, stored, managed, or used in a centralized location inside the organization of the knowledge nodes.
3. *Centralized-externally*. The knowledge is created, stored, managed, or used in a centralized location outside the organization of the knowledge nodes.
4. *Distributed-internally*. The knowledge is created, stored, managed, or used in a distributed manner throughout the organization of the knowledge nodes.
5. *Distributed-externally*. The knowledge is created, stored, managed, or used in a distributed manner throughout knowledge nodes both outside and inside the organization.

We shall now present a sample case through which to illustrate the use of distributed knowledge levels. This case deals with the distribution of the knowledge needed by the software engineers in a software development company. The company has different units throughout the world, and each unit has its own software development department. Each department in the company is considered as a knowledge node, and we wish to provide KM support to a specific software development department in a specific unit, in which the set of knowledge nodes in each unit is considered as an organization. Therefore, if the knowledge used by the software development department is stored in a knowledge repository controlled by the members of that department, then it is *local*. If the knowledge is managed by a server outside the development department, for example by a central server of the company unit in that department, it is *centralized-internally*. If the knowledge is managed by one of the company's central servers which is in a different unit to that of the software development department, then it is *centralized-externally*. If the knowledge is managed by different departments of the same unit, then it is *distributed-internally*. And finally, if the knowledge is managed by different departments of different units (for example, if each development department has its own lessons learned repository, but they are shared with other software development departments), then it is *distributed-externally*.

It is clear that the two technical factors defined in this section are not the only ones that could be used. An organization should define its own technical issues according to the needs or the focus of the tools that they wish to analyze. For instance, Tiwana and Ramesh [77] have analyzed web-based KM systems through characteristics such as robustness, security, or scalability, amongst others. Another example is the spatio-temporal dimension used by Banerjee [6] and Rao [61], in which the place (co-located/ remote) that givers and takers of knowledge are in, and the time (synchronous/ asynchronous) in which they share knowledge are considered.

5 Use of the framework: a case study

In this section, we present a real scenario to illustrate the use of the framework in analyzing a tool in a company. The scenario was obtained from a case study conducted in a software development group [65, 66] where it was discovered that there were information systems which were used by the members of the group to create, capture, store, or retrieve valuable knowledge. To explore the way in which one of these systems could be integrated in the process as a KMS, we applied the framework described in this paper. The goal was to detect whether a specific tool supported certain KM activities and how this was done, in order to later decide whether it was convenient to improve the use of the tool as a knowledge flow facilitator, or to develop or acquire a new system through which to better support the knowledge flow.

5.1 Context of the study

The particular case that we are addressing in this section is focused upon an information system used to store the modification requests and problem reports made by the users of the systems maintained by the Informatics Department in a research centre. At the time of the study this department was made up of fourteen people: the head of the department, a secretary, six software engineers, and six assistant programmers. The group was in charge of dealing with applications in 5 domain areas: finances, human and material resources, academic productivity, and student services. The applications dealt with could be considered as medium sized applications (between 50 and 100 different modules per application, taking into account reports, views, forms, menus, etc., - the informatics department does not have an exact measurement of their current size-).

The tool analyzed in this case study is a logbook in which all the modification requests sent to the informatics department are stored in order to be able to follow their progress as they are addressed. The system was designed to provide end users and the head of the informatics department with an awareness of the status of the requests. This system can be accessed by all the members of the department, who can also add problem reports and modification requests, and by all the users of the different systems being maintained, such as secretaries, the centre's heads of divisions and departments, its researchers, and other employees. The exact quantity of people that have access to the system is unknown, but, it is highly probably that there are more than 500 people. Nevertheless, the people that actually access the system might be between 100 and 150. However, in this work we will analyze the logbook from the point of view of the members of the informatics department, not of the end users, since the system is part of the maintenance process of this department and the maintainers use it on virtually a daily basis.

It is important to comment on the fact that the present case was derived from a study in which the development process of the informatics department was analyzed to identify knowledge flows in order to obtain requirements through which to develop a KM system [65]. This previous study was carried out by following a process engineering based methodology [64]. The study was performed through the modelling of the process to identify the way in which knowledge flows through the process; examples of these models can be found in [63]. These models were used to obtain information in four phases: first, the main knowledge topics or areas involved in the process were identified; secondly, the main sources from which those knowledge topics were obtained were identified and classified; then, the main knowledge flows, which is to say the transfers of knowledge between activities and/or sources were modelled and analyzed. Finally, the knowledge flows were studied to identify the problems that were affecting them. In the present study, we used the models and information obtained from the application of the method described to focus on those processes and activities in which the logbook was involved. This information was analyzed by following the framework being proposed.

In the study, it was observed that the software engineers were using certain tools for different purposes than those for which they had been designed. We were particularly interested in the logbook, because we observed that some engineers in the informatics department were using it as a lessons learned repository, and not only as a simple requests tracking system. For instance, when they remembered that they had previously solved problems that were similar to those that they were solving at a specific time, they accessed the logbook to retrieve those previous problems in order to look for the comments or observations that they had written in those requests. Some engineers used it to write comments describing the changes made, such as the files that needed to be changed [65]. The following subsections illustrate the manner in which the framework was used to analyze the use of the logbook as a KM facilitator.

5.2 Defining the application domain and knowledge structure

The application domain of the logbook is summarized in Table 8 and is presented from the point of view of the informatics department software engineers. Our main interest was in identifying how the logbook could be used to help the informatics department engineers to obtain useful knowledge. The main activity in this department is the maintenance of the software systems used in the centre. Therefore, the logbook supports the domain process of the informatics department. Since, in this case, we are considering the department as the organization to which the tool is providing support, and the engineers are a group inside this organization, the scope of the logbook is intra-organizational. However, as we have previously mentioned, the department’s clients also have access to information that is managed in the logbook, but at this point we are only interested in how the engineers' activities can be supported by the tool.

[Table 8 around here]

5.3 Defining the knowledge flow support and technical issues

As is summarized in Table 9, the logbook can be characterized as a storage tool, since it is mainly used as an information repository. It does not support the maintenance stage. The logbook provides a graphical interface in which engineers introduce the data required to track the status of each request, or to describe the solution given to each request so it therefore also supports the externalization stage to a small extent. However, this description is written in a free text format and it is, therefore, difficult to obtain specific information in an autonomous way, since it is necessary to read the description of the solutions to discover what changes were made, the place in which those changes were made, the source files that were modified, etc. Moreover, each engineer writes these reports in a different way, and provides different levels of detail. To retrieve the information stored, the logbook provides a graphical interface which the engineers can search for modification requests by date, status, the engineer in charge, the client, or by the system to be changed. It is not easy to retrieve similar requests, since there is no mechanism with which to measure the similarities between them. If an engineer searches for a previous request which may be useful in solving a current problem, it is because s/he knows that the request exists, and has previous knowledge, such as an approximate date, or the client that requested the previous changes. Finally, the logbook is not autonomous, and it is internally centralized, since data is stored in a database server within the informatics department.

[Table 9 around here]

5.4 Requirements to improve the KM support

Once the informatics department observed the role that the logbook was currently playing as a KM facilitator, they become aware that the functionality of the logbook could be the basis of their KM strategy. They therefore decided to improve this functionality, either by improving the logbook, or by developing or buying a similar tool with better characteristics.

In the study we observed certain concerns of the informatics department that were related to the KM support and technical issues. The department was not interested in modifying the application domain of the logbook, but in improving the support in certain KM activities. First, the informatics department was interested in extending the access to knowledge sources associated with a specific request. There were frequently important sources of knowledge related to a request that were difficult to find; this was because the engineers did not know about them, or did not remember where they were. The solution proposed for this problem was to link each request to other documents or other types of sources that could be related to them. It would therefore be possible to access those sources through the requests to which they were linked. Thus, the first requirement was that **the system should facilitate the application of knowledge from different types of sources** (people, documents, source files, etc.), and not just previous requests. Therefore, the system should enable an improvement in the application of the knowledge that the informatics department already has.

The second problem observed was that the use of knowledge of previous requests through which to solve new problems was made exclusively by the engineers who already knew of the existence of those previous requests because they had dealt with them. It was important for the department to provide means by which to help the engineers to obtain knowledge even when they were ignorant of the existence of the sources of that knowledge. Therefore, the second requirement was that **the system should have a higher level of autonomy** (shared, delegated, or full autonomy). It should be capable of searching for useful knowledge, even if the engineers did not request specific topics or sources. This is because engineers do not always know what to search for, or where to search.

The third problem observed was that the system did not allow the software engineers to capture the information related to the solution of a request, since the request form in the logbook was not designed for that purpose. Therefore, in order to permit the engineers to externalize their experiences in solving a specific request, a third requirement was defined: **to include a structured form through which to capture information about the solution given to each request**, thus facilitating the externalization of the knowledge associated with those solutions, such as the cause of the problem, the change made, or the secondary effects that the changes may have caused.

It can be observed that by analyzing the logbook from the point of view of the characteristics proposed in the framework, it was easier for the informatics department to focus on the main aspects in which the logbook could be improved in order to extend the KM support in their process, particularly, in improving the support of the application and the externalization of knowledge and to increase system's the level of autonomy. Once these requirements had been defined, the next step was to decide whether to improve the logbook, to develop a new system, or to buy one that could address those requirements. This was done by comparing other similar tools, using the characteristics covered by the logbook and the new requirements as a base.

5.5 Analyzing tools to achieve the requirements

Once the requirements to improve KM support have been identified, a company may have three options: to acquire a new system that fulfills the requirements, to develop a new one, or to improve current tools already in use and that have been found to be useful in managing knowledge. In this section, we show the manner in which a comparison of the logbook to others similar tools was performed in order to decide which option to choose.

Commercial workflow and task manager systems exist which partially satisfy the requirements defined. For example, HelpStar [40] is a service manager system that can be used to manage requests, as is done by the logbook. This tool also manages a knowledge base which enables the users to obtain statistical information easily, such as the most common problems reported, the average time required to solve specific types of problems, etc. Nevertheless, since these types of systems are designed to be used by many different types of companies, they do not achieve some of the particular needs of the informatics department. For instance, it is not easy to track the files that were modified to solve each request. Although the system provides facilities to attach or link documents related to a specific request, such as the files changed, this must be done entirely by the users.

On the other hand, tools exist for software development and maintenance that may address the problem of tracking the changes of source files derived from a specific request. For instance, Bugzilla [18] is a tool which is widely used by software development companies to manage their requests. This tool has facilities to link the requests with the versions of the source files that were modified, by integrating a source version control tool, such as CVS [22]. Therefore, it would be possible to use Bugzilla to obtain knowledge such as the dependencies between software artifacts [33].

As is shown, we found commercial and even open source and freely available software that might support part of the requirements defined. However, we did not know of any available software that would support the application of knowledge at the level of autonomy that was needed by the informatics department. There are nevertheless some research tools that may address this requirement. For instance, Rodriguez et al, in [65] use software agents to identify knowledge needs by retrieving information from a problem report being analyzed by a software engineer and, based on that information, the system starts searching for knowledge sources that may have knowledge which is useful to solve that problem report.

After the analysis of the tools previously mentioned, the informatics department has decided to improve the logbook rather than buying or developing a new system. This is for three reasons: first, they have not found a tool that can fulfill all the requirements; second, the department does not want to lose the information that is already stored in the logbook; and thirdly, the improvement of the logbook, along with the development of the tools that may be required to satisfy all the requirements, can be done in-house. From this section, we can note that the framework provided the informatics department with information that helped it to make an informed decision about the way in which to choose their KM support.

To finalize the description of the case study, we can state that it clearly illustrates the way in which the four main goals (explained in Section 2) for which the framework was developed are accomplished: first, the accomplishment of goal 1 was illustrated in Section 5.2 by identifying the role that a tool plays as a KM enabler, and by identifying the type of support provided by the tool to the flow of knowledge as is shown in Section 5.3. Secondly, goal 2 was illustrated in Section 5.4 through the identification of requirements through which to improve a tool as a KM facilitator. In the third place, goals 3 and 4 were illustrated in Section 5.5, in which the logbook was compared with other tools to decide whether to improve the current tool or to develop or acquire a new one; the final conclusion was that the best choice was to extend the logbook with new characteristics.

6 Discussion and lessons learned

A complete validation of the benefits of applying the proposed framework could be made by measuring the benefits obtained from the integration, as part of the KM strategy, of the tools analyzed. However, validation of KM approaches is a very difficult task, since results appear in the long run, and it is difficult to measure how much a KMS or strategy has contributed to those results, as has been pointed out in [4]. Perhaps this is why "KMS success is commonly evaluated at an abstract level that is influenced by an unmanageable and unstructured amount of factors" [34] citing [45]. In fact, it is not clear what kind of metrics should be used to evaluate specific KMSs, tools, strategies, or initiatives [34, 44, 45, 71]. Therefore, we have limited our evaluation to the testing of the framework's applicability to a real case, and to extracting experiences and lessons learned from this. This was done through the analysis of a tool used in a software development environment. From our experience, we can extract the following observations:

- The members of the informatics department are now aware that they had actually been applying KM in their processes even when they had no explicit KM strategy. This has helped them to understand the importance of KM in their activities, which has contributed to the fact that the informatics department has started applying KM in an explicit way. It was not necessary to engage in a cultural change to make the members of the department aware of the importance of KM in their activities. They have seen that they actually perform KM activities in their daily work and with their daily working tools.
- The informatics department now knows that they actually have KM systems even when they did not acquire them for KM purposes. Thus, they have found new value in their current tools, which has also contributed to the members of the informatics

department seeing their work environment from a KM perspective. The integration of working tools as KMSs can facilitate the integration of KM activities into part of the working processes, which is a key issue if a KM strategy is to succeed [73].

- Finally, we observed that one of the aspects that was considered most valuable by the informatics department whilst they were comparing tools, was a technical one, particularly the level of autonomy of the tools. This is interesting for us since most frameworks do not consider technical aspects, and none of them consider autonomy as an important characteristic. The case study has shown us that technical characteristics may be a very important issue in helping organizations to make decisions about the type of support they need.

7 Conclusions

Sometimes, organizations may not have enough resources to be able to apply KM strategies that require the acquisition of new KMSs, since these systems may be very expensive, or require too much extra work from users [85]. If we are to facilitate the adoption of KM strategies in organizations' processes, then it is important to help them to integrate their current technologies into their KM strategy. Another reason for doing this is that such strategies should support the real work being done in those organizations. The framework presented in this paper was designed to be used as a starting point in helping to accomplish this. Our experience in applying the framework has led us to believe that it is useful for this purpose. However, more research will be carried out by applying the framework to different domains and settings with the goal of continuing to evaluate its efficiency and limitations. Defining a quantitative approach through which to evaluate the contribution of frameworks such as ours in order to facilitate their comparison with others is also a challenge that opens directions for further work.

Acknowledgments

This work is partially supported by CONACyT under grant C01-40799 and scholarship 164739 provided to the first author in Mexico; and the ESFINGE project (TIN2006-15175-C05-05) Ministerio de Educación y Ciencia (Dirección General de Investigación)/ Fondos Europeos de Desarrollo Regional (FEDER) in Spain.

References

[1] M. Alavi, *KMPG Peat Marwick U.S.: One Giant Brian*. 1997, Harvard Business School: Boston, MA. p. 21.

[2] M. Alavi and D.E. Leidner, Knowledge Management and Knowledge Management Systems: Conceptual Foundations and Research Issues, MIS Quarterly, **25** (2001) 107-136.

[3] M. Alavi and A. Tiwana, Knowledge Integration in Virtual Teams: The Potential Role of KMS, Journal of the American Society for Information Science and Technology, **53** (2002) 1029-1037.

[4] N. Anquetil, K.M. de Oliveira, K.D. de Sousa, and M.G. Batista Dias, Software maintenance seen as a knowledge management issue, Information and Software Technology, **49** (2006) 515-529.

[5] A. Aurum, R. Jeffery, C. Wohlin, and M. Handzic, Eds. Managing Software Engineering Knowledge, Springer, Berlin, Germany, 2003.

[6] R. Banerjee, A Fool with a Tool Is Still a Fool, in: M. Rao, (Ed.), Knowledge Management Tools and Techniques, Elsevier, Amsterdam, 2005, pp. 283-292.

[7] V.R. Basili, G. Caldiera, and H.D. Rombach, The Experience Factory, in: J.J.

Marciniak, (Ed.), Encyclopedia of Software Engineering, John Willey & Sons, 1994, pp. 469-476.

[8] V.R. Basili and C.B. Seaman, The Experience Factory Organization, IEEE Software, **19** (2002) 30-31.

[9] G. Beavers and H. Hexmoor. Types and Limits of Agent Autonomy, in: Proceedings of the Agents and Computational Autonomy at AAMAS 2003, Springer, Melbourne, Australia, 2003, pp. 95-102.

[10] D. Binney, The knowledge management spectrum -understanding the KM landscape, Journal of Knowledge Management, **5** (2001) 33-42.

[11] S.J. Bleistein, K. Cox, J. Verner, and K.T. Phalp, B-SCP: A requirements analysis framework for validating strategic alignment of organizational IT based on strategy, context, and process, Information and Software Technology, **48** (2006) 846-868.

[12] M. Bonifacio, P. Bouquet, and R. Cuel, Knowledge Nodes: the Building Blocks of a Distributed Approach to Knowledge Management, Journal of Universal Computer Science, **8** (2002) 652-661.

[13] M. Bonifacio, P. Bouquet, and P. Traverso, Enabling Distributed Knowledge Management: Managerial and Technological Implications, Novatica and Informatik/Informatique, **3** (2002) 23-29.

[14] N. Bontis, M. Fearon, and M. Hishon, The e-flow audit: and evaluation of knowledge flow within and outside a high tech firm, Journal of Knowledge Management, **7** (2003) 6-19.

[15] U.M. Borghoff and R. Pareschi, Information Technology for Knowledge Management, Journal of Universal Computer Science, **3** (1997) 835-842.

[16] U.M. Borghoff and R. Pareschi, Eds. Information Technology for Knowledge Management, Springer, Berlin, Germany, 1998.

[17] S. Brainov and H. Hexmoor. Quantifying Relative Autonomy in Multiagent Interaction, in: Proceedings of the IJCAI-01, Workshop on Autonomy, Delegation, and Control: Interacting with Autonomous Agents, Seattle, WA, 2001, pp. 27-35.

[18] Bugzilla. Bugzilla web site. Available from: http://www.bugzilla.org/.

[19] C. Carabelea, O. Boissier, and A. Florea. Autonomy in Multi-agent Systems: A

Classification Attempt, in: Proceedings of the Agents and Computational Autonomy at AAMAS 2003, Springer, Melbourne, Australia, 2003, pp. 103-113.

[20] K. Chang Lee, S. Lee, and I. Won Kang, KMPI: measuring knowledge management performance, Information Management, **42** (2005) 469-482.

[21] R. Cuel. A New Methodology for Distributed Knowledge Management Analysis, in: Proceedings of the International Symposium on Knowledge Management (I-KNOW '03), Graz, Austria, 2003, pp. 531-537.

[22] CVS. CVS web site. Available from: http://www.nongnu.org/cvs/.

[23] K. Dalkir, Knowledge Management in Theory and Practice, Elsevier, Amsterdam, 2005.

[24] T.H. Davenport, D.W. De Long, and M.C. Beers, Successful Knowledge Management Projects, Sloan Management Review, **39** (1998) 43-57.

[25] T.H. Davenport and L. Prusak, Working Knowledge: How Organizations Manage What they Know, Harvard Business School Press, Boston, Massachusetts, USA, 2000.

[26] K.D. de Sousa, N. Anquetil, and K.M.d. Oliveira. Learning Software Maintenance Organizations, in: Proceedings of the 6th International Workshop on Learning Software Organization (LSO 2004), Springer, Banff, Canada, 2004, pp. 67-77.

[27] K.C. Desouza, Barriers to Effective Use of Knowledge Management Systems in Software Engineering, Communications of the ACM, **46** (2003) 99-101.

[28] Y. Ding and S. Foo, Ontology research and development, Part 1 - A review of ontology generation, Journal of Information Science, **28** (2002) 123-136.

[29] T. Dingsøyr, Postmortem reviews: purpose and approaches in software engineering, Information and Software Technology, **47** (2005) 293-303.

[30] T. Dingsøyr and R. Conradi, A survey of case studies of the use of knowledge management in software engineering, International Journal of Software Engineering and Knowledge Engineering, **12** (2002) 391-414.

[31] T. Dingsøyr, N.B. Moe, and Ø. Nytrø, Augmenting Experience Reports with Lightweight Postmortem Reviews, Lecture Notes in Computer Science, **2188** (2001) 167-181.

[32] J.S. Edwards, Managing Software Engineers and Their Knowledge, in: A. Aurum, R. Jeffery, C. Wohlin, and M. Handzic, (Eds.), Managing Software Engineering Knowledge, Springer, Berlin, 2003, pp. 5-27.

[33] M. Fischer and H. Gall, Visualizing feature evolution of large-scale software based on problem and modification report data, Journal of Software Maintenance and Evolution: Research and Practice, **16** (2004) 385-403.

[34] F. Folkens and M. Spiliopoulou. Towards an Evaluation Framework for Knowledge Management Systems, in: Proceedings of the Practical Aspects of Knowledge Management (PAKM), Springer, 2004, pp. 23-34.

[35] S. Franklin and A. Graesser. Is it an Agent, or Just a Program? A Taxonomy for Autonomous Agents, in: Proceedings of the Workshop on Intelligent Agents III, Agent Theories, Architectures, and Languages, Springer, 1996, pp. 21-35.

[36] S. García, C. Graettinger, and K. Kost, *Proceedings of the First International Workshop for Process Improvement in Small Settings, 2005*. 2006, Carnegie Mellon, Software Engineering Institute: Pittsburgh, PA. p. 309.

[37] R. Grehan, Managing your content with XML, InfoWorld, **27** (2005) 16-18.

[38] K. Haggie and J. Kingston, Choosing Your Knowledge Management Strategy, Electronic Journal of Knowledge Management Practice, **4** (2003).

[39] M. Harris, K. Aebischer, and T. Klaus, The Whitewater Process: Software Product Development in Small IT Businesses, Communications of the ACM, **50** (2007) 89-93.

[40] Help Desk Technology. HelpStar web site. [accesed: 2006 09 february]. Available from: http://www.helpstar.com/.

[41] H. Hexmoor. A Cognitive Model of Situated Autonomy, in: Proceedings of the PRICAI 2000 Workshop Reader, Four Workshops held at PRICAI 2000; Advances in Artificial Intelligence, Springer, Melbourn, Australia, 2000, pp. 325-334.

[42] H. Hexmoor, Stages of Autonomy Determination, IEEE Transactions on Man, Machine, and Cybernetics, **31** (2001) 509-517.

[43] C.W. Holsapple and K.D. Joshi. Description and Analysis of Existing Knowledge Management Frameworks, in: Proceedings of the 32nd Hawaii Int. Conf. on Systems Sciences (HICSS), IEEE Computer Society, Maui, Hawaii, 1999.

[44] M.E. Jennex and L. Olfman, Assessing Knowledge Management Success, International Journal of Knowledge Management, **1** (2005) 33-49.

[45] A. Kankanhalli and B.C.Y. Tan. A Review of Metrics for Knowledge Management Systems and Knowledge Management Initiatives, in: Proceedings of the 37th Hawaii International Conference on Systems Science, IEEE Computer Society, 2004.

[46] A. Kankanhalli, F. Tanudidjaja, J. Sutanto, and B.C.Y. Tan, The Role of IT in Successful Knowledge Management Initiatives, Communications of the ACM, **46** (2003) 69-73.

[47] O. Kühn and A. Abecker, Corporate Memories for Knowledge Management in Industrial Practice: Prospects and Challenges, in: U.M. Borghoff and R. Pareschi, (Eds.), Information Technology for Knowledge Management, Springer, Berlin, 1998, pp. 183-206.

[48] L.P.W. Land, A. Aurum, and M. Handzic. Capturing Implicit Software Engineering Knowledge, in: Proceedings of the 13th Australian Software Engineering Conference (ASWEC'01), IEEE Computer Society Press, Sydney, NSW, Australia, 2001, pp. 108-114.

[49] T.C. Lethbridge, J. Singer, and A. Forward, How Software Engineers Use Documentation: The State of the Practice, IEEE Software, **20** (2003) 35- 39.

[50] M. Lindvall and I. Rus, Knowledge Management for Software Organizations, in: A. Aurum, R. Jeffery, C. Wohlin, and M. Handzic, (Eds.), Managing Software Engineering Knowledge, Springer, Berlin, 2003, pp. 73-94.

[51] R. Maier and U. Remus, Defining Process-oriented Knowledge Management Strategies, Knowledge and Process Management, **9** (2002) 103-118.

[52] A.D. Marwick, Knowledge management technology, IBM Systems Journal, **40** (2001) 814-830.

[53] M.W. McElroy, The New Knowledge Management, Knowledge and Innovation: Journal of the KMCI, **1** (2000) 43-67.

[54] B. Meehan and R. Richardson, Identification of Software Process Knowledge Management, Software Process Improvement and Practice, **7** (2002) 47-55.

[55] B.B. Newman and K.W. Conrad. A Framework for Characterizing Knowledge Management Methods, Practice, and Technologies, in: Proceedings of the

Practical Aspects of Knowledge Management (PAKM 2000), The Knowledge Management Forum, Basel, Switzerland, 2000, pp. 16.11-16.11.

[56] M.E. Nissen, An Extended Model of Knowledge-Flow Dynamics, Communications of the Association for Information Systems, **8** (2002) 251-266.

[57] I. Nonaka, The Knowledge-Creating Company, Harvard Business Review on Knowledge Management (1991) 26-46.

[58] I. Nonaka and H. Takeuchi, The Knowledge-Creation Company: How Japanese Companies Create the Dynamics of Innovation, Oxford University Press, 1995.

[59] S. Qureshi, V. Hlupic, and R.O. Briggs. On the Convergence of Knowledge Management and Groupware, in: Proceedings of the 10th International Workshop on Groupware (CRIWG'2004), Springer, San Carlos, Costa Rica, 2004, pp. 25-33.

[60] M. Rao, Ed. Knowledge Management Tools and Techniques: Practitioners and Experts Evaluate KM Solutions, Elsevier, Amsterdam, 2005.

[61] M. Rao, Overview: The Social Life of KM Tools, in: M. Rao, (Ed.), Knowledge Management Tools and Techniques, Elsevier, Amsterdam, 2005, pp. 1-73.

[62] I. Richardson and C.G. von Wangenheim, Why Are Small Software Organizations Different? IEEE Software, **24** (2007) 18-22.

[63] O.M. Rodríguez-Elias, A.I. Martínez-García, A. Vizcaíno, J. Favela, and M. Piattini. Organización de conocimientos en procesos de ingeniería de software por medio de modelado de procesos: una adaptación de SPEM, in: Proceedings of the VI Jornada Iberoamericana de Ingeniería del Software e Ingeniería del Conocimiento (JIISIC'07), IEEE Computer Society Press, Lima, Perú, 2007, pp. 257-265.

[64] O.M. Rodríguez Elias, A.I. Martínez García, J. Favela, A. Vizcaíno, and J.P. Soto. Knowledge flow analysis to identify knowledge needs for the design of knowledge management systems and strategies: a methodological approach, in: Proceedings of the 9th International Conference on Enterprise Information Systems (ICEIS): special session on Business Intelligence, Knowledge Management and Knowledge Management Systems, Funchal, Madeira - Portugal, 2007, pp. Accepted.

[65] O.M. Rodríguez, A.I. Martínez, J. Favela, A. Vizcaíno, and M. Piattini, Understanding and Supporting Knowledge Flows in a Community of Software Developers, Lecture Notes in Computer Science, **3198** (2004) 52-66.

[66] O.M. Rodríguez, A.I. Martínez, A. Vizcaíno, J. Favela, and M. Piattini. Identifying Knowledge Management Needs in Software Maintenance Groups: A qualitative approach, in: Proceedings of the Fifth Mexican International Conference on Computer Science (ENC'2004), IEEE Computer Society Press, Colima, México, 2004, pp. 72-79.

[67] I. Rus and M. Lindvall, Knowledge Management in Software Engineering, IEEE Software, **19** (2002) 26-38.

[68] I. Rus, M. Lindvall, and S.S. Sinha, *Knowledge Management in Software Engineering: A State of the Art Report*. 2001, Data & Analysis Center for Software: ITT Industries: Rome, NY. p. 53.

[69] K. Schneider, J.-P. von Hunnius, and V.R. Basili, Experience in Implementing a Learning Sofware Organization, IEEE Software, **19** (2002) 46-49.

[70] C. Seaman. The Information Gathering Strategies of Software Maintainers, in: Proceedings of the International Conference on Software Maintenance (ICSM'2002), IEEE Computer Society, Montreal, Canada, 2002, pp. 141-149.

[71] H.A. Smith and J.D. McKeen, Development in Practice XVII: A Framework for KM Evaluation, Communications of the Association for Information Systems, **16** (2005) 233-246.

[72] J. Sparrow, Knowledge Management in Small Firms, Knowledge and Process Management, **8** (2001) 3-16.

[73] D. Stenmark and R. Lindgren. Integrating Knowledge Management Systems with Everyday Work: Design Principles Leveraging User Practice, in: Proceedings of the Hawaii International Conference on System Science (HICSS), IEEE Computer Society Press, 2004, pp. 80245b.

[74] T.A. Stewart, The Case Against Knowledge Management, Business 2.0, **3** (2002) 80.

[75] J. Teevan, W. Jones, and B.B. Bederson, Personal Information Management, Communications of the ACM, **49** (2006) 40-43.

[76] A. Tiwana, The Knowledge Management Toolkit: Practical Techniques for Building a Knowledge Management System, Prentice Hall, USA, 2000.

[77] A. Tiwana and B. Ramesh, Integrating Knowledge on the Web, IEEE Internet Computing, **5** (2001) 32-39.

[78] A. Valente and T. Housel. A Framework to Analyze and Compare Knowledge Management Tools, in: Proceedings of the Knowledge-Based Intelligent Information Engineering and Allied Technologies (KES'2002), IOS Press, Crema, Italy, 2002, pp. 291-295.

[79] L. van Elst, V. Dignum, and A. Abecker. Towards Agent-Mediated Knowledge Management, in: Proceedings of the International Symposium AMKM 2003, Springer, Stanford, CA, USA, 2003, pp. 1-30.

[80] A. Vizcaíno, M. Piattini, M. Martínez, and G. Aranda. Evaluating Collaborative Applications from a Knowledge Management Approach, in: Proceedings of the Evaluation of Collaborative Information Systems and Support for Virtual Enterprises (ECE), Linköping, Sweden, 2005.

[81] J. Ward and A. Aurum. Knowledge Management in Software Engineering - Describing the Process, in: Proceedings of the 15th Australian Software Engineering Conference (ASWEC 2004), IEEE Computer Society Press, Melbourne, Australia, 2004, pp. 137-146.

[82] E. Wenger, Communities of Practice: Learning, Meaning, and Identity, Cambridge University Press, Cambridge, U.K., 1998.

[83] K. Wiig, People-Focused Knowledge Management: How Effective Decision Making Leads to Corporate Success, Elsevier, Amsterdam, 2004.

[84] R. Woitsch and D. Karagiannis, Process-oriented Knowledge Management Systems based on KM-Services: The PROMOTE Approach, International Journal of Intelligent Systems in Accounting, Finance & Management, **11** (2002) 253-267.

[85] K.Y. Wong, Critical success factors for implementing knowledge management in small and medium enterprises, Industrial Management & Data Systems, **105** (2005) 261-279.

Table 1. Characteristics of some frameworks for comparison, characterization, or classification of KMSs. The table presents the purpose of the framework, the approach used, and the steps and characteristics used to classify KMSs.

Author/ Year	Purpose	Approach	Aspects covered
Borghoff and Pareschi/ 1998, [16]	To classify KM technologies	Authors use a four component corporate memory architecture, and classify technologies depending on the component that tools implement. The components considered are: the flow of knowledge, communities of knowledge workers, knowledge repositories and libraries, and knowledge cartography.	- KM activities - Structure of knowledge (tacit and explicit)
Newman and Conrad/ 1999, [55]	To organize and classify KM methods, practices and technologies	Authors propose a table that can be used as a checklist to specify some concepts related to a KM method, practice or technology. The concepts considered are: KM activity (creation, retention, transfer, utilization), work activity level (high-level process, mid-level process, decision or action), agent type (individual, organization, automatized), knowledge artifact type (explicit, implicit, tacit), and focus (agent, artifact, process).	- KM activities - Structure of knowledge (tacit, implicit, and explicit) - Application domain (use, scope, and domain of knowledge)
Alavi and Leidner/ 2001, [2]	To analyze the role of KMSs in organizational KM	Authors define a set of KM processes and classify KMSs depending on the KM process they support. The KM processes considered are: creation, storage/retrieval, transfer, and application.	- KM activities
Binney/ 2001, [10]	To classify KM applications mentioned in literature	Author defines a "KM spectrum" that establish a set of KM application' types and classifies each application depending on the KM activity in which it is most cited in literature. The spectrum considers the following KM application types: transactional, analytical, asset management, developmental, innovation and creation. This approach is also used and extended in [38].	- Application domain (use of knowledge) - KM activities
Marwick/ 2001, [52]	To describe characteristics of KM technologies	Author uses the knowledge conversion mechanisms proposed by Nonnaka [57], to classify and describe KM technologies. He describes whether specific KM technologies support socialization (conversion of tacit to tacit knowledge), externalization (tacit to explicit), combination (explicit to explicit), or internalization (explicit to tacit).	- KM activities - Structure of knowledge (tacit and explicit)
Rus et. al./ 2001, [68]	To classify KM tools used in software engineering	The authors' main focus is on the organizational processes and tasks that the tools support. They also consider KM strategies from the point of view of the scope of those strategies, particularly, if they are focused on individuals, projects, organizations, or industry.	- Application domain (use and scope of knowledge)
Tiwana and Ramesh/ 2001, [77]	To classify general KM technologies, and describe some web-	Authors define a "KM network" composed of a set of knowledge sources, tools, and tasks; and use them to classify KM technologies according to the organizational processes that support, or the KM objectives that enable. The authors also consider	- Application domain (use of knowledge) - KM activities - Technical issues (specific for web-based systems)

	based KMSs	some technical attributes that are important for web-based KM systems	
Valente and Housel/ 2002, [78]	To analyze and compare KM tools	Authors define a matrix of KM services and types of knowledge structures, to be used as a checklist to specify whether a tool supports specific services, and manages knowledge structured in specific types of structure. Knowledge structures are defined as a range of knowledge structure levels from unstructured to structured. Knowledge services consider services required to enable organizational processes (work and managerial processes), and KM processes.	- KM activities - Structure of knowledge (a set of levels from highly unstructured to formally structured knowledge) - Application domain (use of knowledge)
Woitsch and Karagiannis/ 2002, [84]	To define the required functionality of enterprise KMSs	Authors define a vector of "KM-Dimensions" where the dimensions are a set of properties or attributes of knowledge to be managed, such as: representation, medium, user, time, origin, sophistication, life cycle, relevance, applicability, level, dynamic, expression, service boundaries, abstraction, action, and structure.	- KM activities - Application domain (use, scope of knowledge) - Structure of knowledge (tacit, explicit / unstructured to structured) - Technical issues (a set of attributes for the specific type of knowledge being managed)
Kankanhalli et. al./ 2003, [46]	To analyze the role of IT in KM in a set of organizations.	Authors compare technologies following a four dimension framework. This framework considers if the systems are product or service based, and if they allow managing low or high volatile context, that means environments that change slowly or rapidly.	- Application domain (use of knowledge)
van Elst et. al./ 2003, [79]	To analyze and classify agent-based KM approaches	Authors define a three dimensional framework, where the first two consider technical aspects of agent based systems, while the third one focuses on the system's application area. In this third dimension, called KM application level, are considered the scope of the system, the KM processes or tasks supported, and whether the system is oriented to knowledge products or processes.	- KM activities - Application domain (use of knowledge)
Qureshi et. al./ 2004, [59]	To analyze groupware tools as KM technologies	Authors use a set of characteristics of groupware technologies to classify them, and illustrate the KM processes that different types of groupware tools support.	- KM activities - Application domain (use of knowledge) - Technical issues (related with groupware tools)
Banjerjee/ 2005, [6]	To analyze how KM tools are being used in organizations	Author uses the KM processes and the business scope supported to analyze KM tools. In the business scope dimension are considered the organizations context, the distribution of givers and takers of knowledge, and the temporal access of knowledge (synchronous, asynchronous).	- KM activities - Application domain (use of knowledge) - Technical issues (distribution of knowledge sources, temporality of the access to the knowledge)
Rao/ 2005, [61]	To classify and describe characteristics of KM technologies	Author uses different approaches for classifying KM tools and technologies: the spatio-temporal, the KM processes or activities supported, the knowledge creation process of Nonnaka [58], and the domain and scope of the tools.	- KM activities - Application domain (use of knowledge) - Technical issues (characteristics of knowledge, characteristics of specific types of tools, and the same of [6])
Vizcaíno et. al./ 2005, [80]	To evaluate whether collaborative systems support KM activities	Authors define a set of questions which answers can help to identify if a tool supports a specific KM activity.	- KM activities

Table 2. Schema used to classify the processes and activities supported by KM technologies.

Category	Description
Business strategies	Processes and activities that help accomplish the business strategies, such as tools to manage knowledge about the market, competition, etc.
Organization management	Processes and activities to manage the organization, such as the management of human resources, finances, etc.
Organizational life	Processes and activities related to the organization's daily work. For example, to manage the inventory, the organization's infrastructure, etc.
Domain processes	Processes and activities related to the work domain of the organization, such as those directly related to the products or services provided by the company.
Technical activities	Processes and activities related to specific aspects of the domain processes, such as the use of the tools used by the employees to develop a specific product.

Table 3. Structure of the levels of granularity used to define those people who benefit from KM technologies (an extension of the used by Rus et al. [68]).

Level	**Definition**
Personal	The knowledge is useful to specific people in a personal way. For example, a repository of personal documents in the workplace of a particular person.
Intra-Organizational group	The knowledge is useful to a well defined unit or group of people within the organization. For example, certain roles or position levels in the organization, a department, a division, etc.
Extra-Organizational group	The knowledge is useful to a group of people inside and outside the organization, for example a community of experts in which some employees participate.
Organization	The knowledge is useful to the entire organization.
Multiple organizations	The knowledge is useful to other organizations too. For example, the costumers or suppliers.
Industry wide	The knowledge managed by the tool can be used by all the industry.

Table 4. Scheme used to classify the domain of knowledge.

Type of knowledge	**Definition**
Business	Knowledge related to the business strategy, such as knowledge of the clients, the suppliers, the market, etc.
Organizational	Knowledge related to the operation of the company, such as its structure, human or material resources, etc.
Managerial	Knowledge related to aspects of the management of the company, such as planning and leading projects. For instance, knowledge of the processes followed in the company, such as standards to follow, etc.
Product-Service	Knowledge related to the specific aspects of the products or services supplied by the company. For instance, the production processes, intermediate products such as documents or components, etc.
Technical	Knowledge required for performing the productive activities of the company such as methodologies or usage of tools.

Table 5. Knowledge structure classification schema (an extension of the used by Valente and Housel [78]).

Knowledge structure	**Format**	**Description**
Tacit (T)	Skills / abilities	Refers to the technical knowledge that depends on the experiences of people, and therefore it is hard, if not impossible, to formulize and make it explicit.
	Know how	Refers to the technical knowledge that people might have, and that might be relatively easy to make it explicit, such as procedures, methods, techniques etc. As well, this type of knowledge could be implicitly contained in some formal sources of knowledge, such as a workflow or a process based system that helps people to know how to do something, such as which activities should be performed to accomplish a goal.
Explicit unstructured (EU)	Audio or video (A | V)	Media that can be used to store and share knowledge and information; however, due to its nature it is difficult to process it automatically through machines. Hence, people are generally required to interpret it.
	Images (I)	Images can be used to represent information visually. Although there are techniques to automatically obtain information from images (image processing algorithms), people could still be required to fully interpret some images.
	Free text (FT)	This category contains sources of knowledge expressed in free text, such as unstructured documents, for instance reports, memos, etc. There are tools that help manage these kinds of sources using the words that they contain, for example to index or classify them.
Explicit structured (ES)	Graphics (GR)	This category is used to specify knowledge represented graphically, and follows a well defined graphical language, in a manner in which any person who knows the language can understand the information represented in the graphic.
	Semi-structured text (SST)	This representation allows some characteristics to be automatically identified in the documents, such as titles, types of paragraphs, etc., due to the use of a semi-structured representation of the information. One example is HTML, which facilitates obtaining information about the structure of a document.
	Structured text (ST)	This representation stores more detailed information in the documents, which facilitates their management by computers. An example of this is XML, which allows one to obtain information about the content and structure of documents.
	Data (D)	Data alone could have no meaning, but grouped and taken into context they can provide information and knowledge. Moreover, there are techniques that can be used to discover knowledge in large amounts of data, such as data-mining.
	Metadata (MD)	Metadata allows one to obtain information to make the management of data or information by computers easier. Metadata can be seen as information about the data or other information. An example of this is ontologies used to describe the contexts of some data or a domain, such as the concepts, their attributes and relationships [28].
	Categorized knowledge or information (CK)	This category is used for systems that make use of well defined structures to classify documents or other information sources. Examples of this are the directories used in web search services such as Yahoo to search for web pages by categories.
	Mathematically structured knowledge (MSK)	This category considers the knowledge expressed in a mathematical form, with an explicit and rigorous definition of its semantics, in a manner which makes it possible to follow the rules for managing that knowledge without ambiguity. An example of this could be an expert system with a knowledge base created following a mathematical formulation, making it easy to obtain concrete answers to specific questions expressed with a formally defined language.

Table 6. Knowledge management life cycle models.

Model	**Stage 1**	**Stage 2**	**Stage 3**	**Stage 4**	**Stage 5**	**Stage 6**	**Stage 7**	**Stage 8**
Davenport & Prusak [25]	Generate	Codify/ Coordinate	Transfer					
Tiwana [76: p. 72]	Acquire	Share	Use					
McElroy [53]	Produce	Integrate	Diffusion	Apply				
Alavi & Leidner [2]	Create	Store/ Retrieve	Transfer	Apply				
Meehan & Richardson [54]	Create	Store	Share	Leverage				
Rus & Lindvall [67]	Originate/ Create	Capture/ Acquire	Transform/ Organize	Deploy/ Access	Apply			
Edwards [32]	Create/ Acquire	Retain	Share/ Transfer	Use	Refine/ Update			
Qureshi et al. [59]	Create	Collect	Organize	Deliver	Use			
Chang Lee et al. [20]	Create	Accumulate	Sharing	Internalize	Use			
KPMG [1]	Acquire	Indexing	Filter	Link	Distribute	Apply		
Nissen [56]	Create	Organize	Formalize	Distribute	Apply	Evolve		
Dalkir [23]	Capture/ Create	Assess	Share/ Disseminate	Contextualize	Acquire/ Apply	Update		
Rao [60]	Create	Codify	Retrieve	Apply	Distribute	Validate	Track	Personalize

Table 7. Parameters used to evaluate how the stages of the KM lifecycle are supported.

Parameter	**Description**	**Metric**
Autonomy	Indicates how autonomous the tool is when it supports a specific stage of the KM lifecycle. Whether it requires people to do everything or can act automatically for some purposes.	A value of the set: [not-autonomy, semi-autonomy, shared-autonomy, delegated-autonomy, full-autonomy]
Distribution	Indicates the distribution of the knowledge managed by the tool	A value of the set: [local, centralized internally, centralized externally, distributed internally, distributed externally]

Table 8. Definition of the application domain of the logbook.

Parameter	**Value**	**Detail**
Use of knowledge	Domain processes	Software maintenance process. The tool is used to help the software engineers obtain information useful to accomplish the modification requests or problems reported.
Scope of knowledge	Intra-organizational group	The software engineers of the ID are the ones who benefit from using the tool under this application domain.
Domain of knowledge	Domain knowledge	• Information of the modification requests, such as the requirements, or the problems reported by the clients. • Historical information that may be useful for the engineers to accomplish current requests, such as similar requests or problems reported previously, and their solutions. • Historical information of the systems that are being maintained by the ID that can be used to obtain statistical data, such as the most problematic systems, the most reported type of problems, and so forth.
Structure of knowledge	Explicit Structured / Data	The tool stores and manages data structured as registers of a relational database.
	Explicit Unstructured / Free text	The tool enables storing detailed descriptions of the requests and their solutions as free text.

Table 9. Definition of the knowledge flow support and technical issues.

Parameter	**Level**	**Detail**
Storage	-------------------	An information repository.
Externalization	Small support	Provides means for capturing information in both structured and unstructured forms.
Internalization	Small support	Provides means for retrieving information and organize it in different forms.
Autonomy	Non-Autonomous	Totally manual
Distribution of knowledge	Centralized-Internally	A data base server within the informatics department

Fig. 1. Stages of the framework for analyzing KM systems.

Fig. 2. Model of KM activities, and it's relationship with the Nonnaka's Knowledge Creation Process [58].

Figure 1

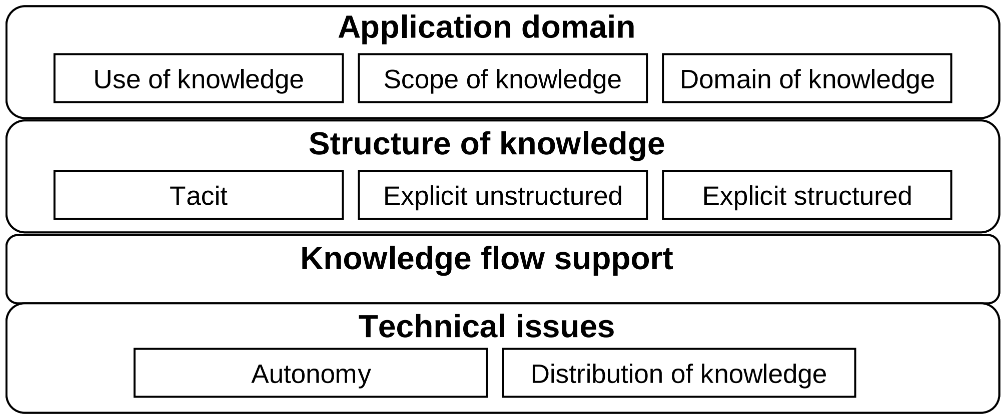
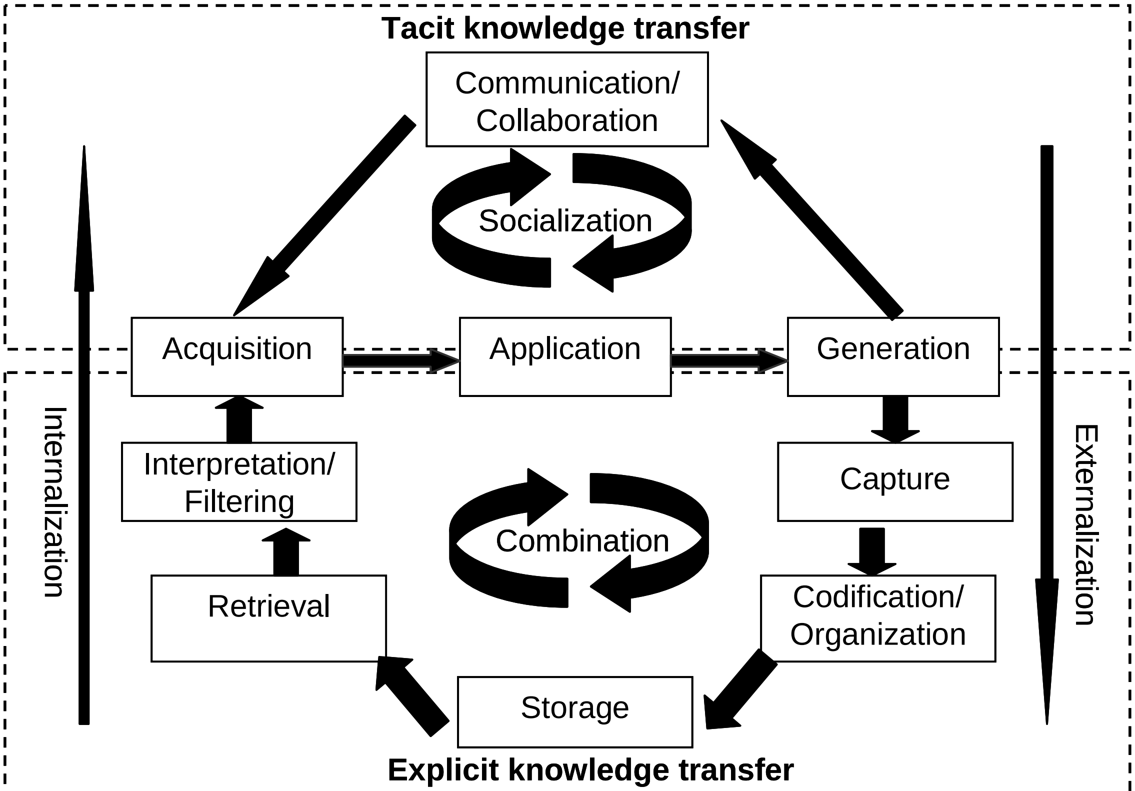

Application domain
Use of knowledge
Scope of knowledge
Domain of knowledge
Structure of knowledge
Tacit
Explicit unstructured
Explicit structured
Knowledge flow support
Technical issues
Autonomy
Distribution of knowledge

Figure 2


Tacit knowledge transfer
Communication/ Collaboration
Socialization
Acquisition
Application
Generation
Internalization
Externalization
Interpretation/ Filtering
Capture
Combination
Retrieval
Codification/ Organization
Storage
Explicit knowledge transfer