**Abstract (Summary)**

The purpose of this paper is to explore the use of knowledge management (KM) principles and technologies to improve the outcomes of software acquisition projects. Software acquisition organizations typically contract-out their software projects to reduce the risks associated with developing the software internally and to control their cost. This paper is based on a study of contracted software projects. Research results by the authors emphasize the need for institutionalizing processes for the collection of data about contracting costs within an organization so that databases of metrics about completed projects can be built and later used to forecast costs for future projects to improve the decision-making processes about outsourcing. A study of two-dozen contracted projects indicates that such organizations face unique risks and hidden costs that are particular to software acquisitions. KM models, practices, and tools are potentially valuable for improving software outsourcing activities. KM can be useful for identifying the organizational structures, processes and informational technologies for measuring, collecting, and analyzing costs and risks incurred before, during and after the contract award.

**»**  [Jump to indexing (document details)](http://proquest.umi.com.ezproxy.apollolibrary.com/pqdweb?did=1332903561&sid=3&Fmt=3&clientId=13118&RQT=309&VName=PQD" \l "indexing)

|  |
| --- |
| **Full Text**   (6862  words) |

Copyright Emerald Group Publishing Limited 2007

**Introduction**

Knowledge management (KM)[1] is being successfully applied in many and different types of organizations to gain a competitive advantage by reducing costs, improving relations with customers, increasing sales, reducing time to market products, facilitating communication and collaboration and by fostering innovation. The literature is replete with success stories in which companies are using KM to work "smarter" by leveraging their intellectual assets to improve their organizational performance. These companies capture part of the knowledge of their knowledge workers and make it available and shareable across the organization. As more organizations are outsourcing their software development projects, KM can be applied to improve their acquisition management processes and reduce their risks by capturing the knowledge and experience gained on past contracted software development projects. By doing so, they will be in a better position to manage the acquisition process "smarter" and forecast more accurately the actual cost and schedule for future projects.

Acquisition organizations contract their software development projects to outside contractors for a variety of reasons. Some of these reasons are to reduce the risk of failure due to continuously evolving technologies and lack of expertise in-house. This is particularly true for government agencies where technical salaries have not kept pace with salaries offered by business resulting in an inability to develop internal expertise. Studies show that between one fourth and one third of US software projects involve outside contractors. However, outsourcing introduces additional cost factors and risks attributed to the interactions between the contractor and the customer. A major hidden cost is incurred due to the contracting organization's personnel being involved in planning, acquiring, and managing the project, and user involvement and participation in the project's development. Such costs are considered hidden as they are incurred but not managed. This paper describes the source and magnitude of the hidden cost, the sources and types of risks to the outsourcing organization and addresses the need for improvements in existing software acquisition[2] processes to mitigate the risks involved through the use of KM principles and technologies.

**Background: the hidden cost survey**

There are a number of widely used software cost estimation processes, methods and tools. Most estimate the cost of the technical and some management and support resources needed for a software project. However, none of them estimates the contracting organization resources that are incurred before, during and after the contract award, in particular the user involvement dealing with the contractor during development processes. Such costs are considered hidden as they are incurred but not accounted for. The results of a recent survey by the authors indicate the need for improvements in existing software cost estimation processes by making the hidden costs of a contracting organization visible. The results of this research are significant to contracting organizations as they show that the hidden costs are incurred and are significant, but not managed. Failure to plan and schedule critical resources, such as user involvement, might result in introducing risks to the project as these resources might not be available when needed. Incorporating the hidden costs into software economics analysis could improve the decision-making process.

There were a number of goals of the research survey that gathered data from over two-dozen organizations about out sourced software projects. Among the practices and project outcomes explored were: the nature of contracting organizations' cost estimation processes, whether hidden costs were incurred due to outsourcing, if so, the magnitude of the hidden cost and the impact of not managing hidden costs, and the areas of risks that predominate in outsourcing domains. The focus of the study ([16] Haddad, 1999) was to determine the:

- cost incurred by the procuring organization for resources needed before and during the system development lifecycle to acquire, manage, coordinate, control, and support the software project;

- effort expended by management, users, support and other personnel with the required skills and expertise, and hardware and software tools required to support such efforts;

- cost of management of the contract, the contractor, the users and the quality of the product for the duration of the project;

- involvement of the users, user management and the user functional experts who have to participate in most activities in the development life cycle including requirement definition, product reviews, document reviews, and testing activities throughout the lifecycle;

- cost of quality assurance activities by quality assurance personnel; and

- cost of software tools, hardware tools, travel expenses, user training, acceptance testing, management of deliverables, and management of the user and other support and miscellaneous expense items in support of the various activities.

The survey showed that few organizations involved in software outsourcing have formal processes in place that will lead to insights into the real resource costs incurred during the outsourcing life cycle. These study outcomes are depicted in Table I [Figure omitted. See Article Image.].

The lack of formal processes for planning resources introduces cost and risks to the project as some of the critical resources are not committed formally to the project and might not be available when needed, in particular those of management and the users. And, for over 50 percent of the organizations studied, hidden cost estimation is an *ad hoc* process, an indication of a low-organizational maturity level.

The survey also tried to identify the source of funding of resources such as management, users, and others (other than contractors) working on software development projects and whether the cost of such resources are included in the final project cost. The results indicated that cost of a resource, such as a project manager or a user, does not come from a project budget. This means that the price of an outsourced software development project's contract does not reflect the actual cost. In addition, there will be a lack of commitment of a resource whose function is considered critical for project success and therefore risks are introduced.

**Additional/hidden costs: phases and labor category distribution**

In addition to revealing the hidden outsourcing costs, the study also explored the magnitude and distribution (over the projects' life cycles) of the hidden costs. The findings from this part of the study also exceeded the expectations of what the value of the hidden costs would be. Initially, the expectation was that the hidden costs would be about 15 percent of the project development cost. However, the study showed that the hidden costs are quite substantial, the mean value being 190 percent of the total development cost of the system, almost twice as much as the contract cost. The value of the hidden cost is significant enough to motivate organizations to have formal procedures to account for it and include it in project plans. The value of the hidden cost must also be included in estimation models used for economic analysis of software projects and must be taken into account in the decision-making processes.

The distribution of the hidden costs throughout the project life cycles was not unexpected the:

- analysis, testing and implementation phases were reasonably uniform, each consumed between 18 and 20 percent of the hidden costs;

- design phase consumed over 14 percent;

- acquisition and training phases consumed between 7 and 9 percent; and

- programming phase, as would be expected, about 6 percent.

Without further exploration, it is difficult to draw precise conclusions about the implications of this distribution other than to have them serve as a warning to project managers. For example, the involvement of client-personnel during the design and programming phases might reflect problems that were not detected during earlier phases and required intervention of users or other personnel even after implementation was begun.

The result of the distribution of the hidden cost by labor categories was also as expected:

- project management personnel induced over 35 percent of the hidden costs;

- users induced almost 50 percent of the hidden costs;

- quality assurance and consultant involvement approximately 6 percent; and

- other personnel about 9 percent.

This result is important in the sense that it draws attention to the need to plan and schedule the procuring organization's resources. The distribution of the hidden costs by labor category and by phase draws the attention to the needed resources and when they should be planned and scheduled during the various phases of the software project along with their expected involvement. For many of the organizations that participated in the survey, realizing the actual cost of a project was an eye opener - particularly, when respondents realized the magnitude of their hidden costs and their failure to manage this aspect of their outsourced project. Organizations that understand and collect data about costs of contracting software are better positioned to estimate costs of future projects and also improve decision-making processes associated with software contract feasibility and oversight.

**Risk identification and management in software acquisition**

In addition to the identifiable hidden costs, the customer's assessment and management of risks associated with outsourced projects is equally important. At a minimum, failure to plan and schedule critical resources such as users, project managers, domain experts, management software, and other resources may pose a risk to an organization and to the project itself. Some risks have significant side effects that impact on other projects within the customer's organization. Outsourcing relationships magnifies some risks. This phase of the research explores the risks associated with sub-contracting with the expectation that recommendations can be devised that will help minimize those risks or at least help organizations design mitigation plans.

What are some of the sources of risk to the contracting organization? Risks are introduced in three primary areas: internally within the customer's organization, externally within the contractor's organization, and in the interfaces between the customer and the contractor. Some conditions that might be defined as potential risks from any of the three sources are not really risks in that there is a certainty of their occurrence but they are typically not identified by the customer or the contractor and they translate directly into cost or schedule overruns, in degradation of the customer-contractor relationship, or reduction in software quality. Table II [Figure omitted. See Article Image.] lists some of the risks in the three primary areas ([17] Haddad and La Salle, 2003).

The interfaces between the customer and the software contractor are the source of many risks that are typically not acknowledged and therefore not managed or mitigated. In the ideal world, customer and contractor would share processes and tools, their activities would be synchronized, and oversight would be welcomed. However, the number of failed contracted projects is testimony to the problems that arise in the relationship between the customer and contractor.

**Benefits in capturing the hidden cost and identifying risks**

During the data collection phase of the survey, top managers of organizations who were participating in the survey became very interested in the study and the findings. The process of engaging in the survey was consciousness-raising as they realized the magnitude of the hidden cost, which they might not have considered before. Their interest can also be attributed to the following benefits that they can achieve in estimating hidden costs:

- Including the hidden cost in any strategic decisions to be made by executive management can improve the accuracy of estimates and the decision making process before undertaking a software project. A decision is usually based on an economic analysis, which requires an estimate of the money, resources, and time required to complete the project. By including the hidden cost in cost-benefit analysis, breakeven analysis or make-buy decisions, they can improve their decisions.

- Using the estimates of the hidden cost in project management to plan, monitor and control the development of a project can improve the management process. Good planning and effective control require an estimate of the activities required to complete a project and the resources required for each activity for monitoring progress.

- The costs estimates can also be used for the allocations of adequate funds for projects over time. The relationship between cost estimation and controlling projects are detailed further in [12] DeMarco (1982).

- Using the estimates of the hidden cost in work breakdown structure to allocate resources and establish their commitments.

- Planning the hidden costs can result in improved relations and communications between the procuring organization and the contractor by allocating the needed user and management resources. For members of a project team to work together more efficiently on a project, it is necessary that each member understand his/her role in the project and the overall activities of the project.

- Including the critical resources in the risk-prone activities can result in improved risk management.

- Improve the chances for successful software projects: estimating and providing the necessary resources for a software project may help in avoiding the negative impacts identified in the findings and may improve the chances for developing better quality systems; systems within budget, on schedule, that meet the user requirements.

In the survey that supports the authors' research, it was evident that most organizations had no formal, institutionalized acquisition planning and tracking processes for software; contractor selection processes (if formalized) tended to follow traditional selection processes for products and services; software requirements documents lacked precision and specificity; few projects had formalized acquisition project management, contract management, or configuration management plans for tracking project activities and artifacts, overseeing internal and contractor personnel, evaluating progress and performance against requirements, or tracking costs; customer inspections of defined milestones were not part of the acquisition processes; risk identification and management processes were missing; and, data collection and archiving, if done at all, was the consequence of individual efforts rather than institutionalized requirements for software acquisition. Few projects involved post-mortems of the project's lifecycle to improve processes in the future.

**Knowledge management concepts**

The previous sections set the context and present strong motivation for organizations to better manage their outsourced software projects. As organizations mature and develop processes to archive metrics about their outsourcing experiences, it is possible to apply formal techniques that use those metrics for decision support and other critical aspects of outsourcing. One such technique is KM.

The integration of KM programs across organizations is growing. According to [13] *Enterprise Intelligence* magazine (2000) "Today, 25 percent of all organizations employ a CKO (Chief Knowledge Officer), or high-level KM position. By 2005, that number is expected to grow to 80 percent."

Knowledge could be defined as:

... knowledge, while made up of data and information can be thought of as much greater understanding of a situation, relationship, causal phenomena, and the theories and rules that underline a given domain or problem ([15] Firestone and Mc Elroy, 2003)

There are many definitions of KM including the following:

- "Knowledge Management is leveraging relevant intellectual assets to improve organizational performance" ([36] Stankosky, 2000).

- "Knowledge management is the systematic, explicit, and deliberate building, renewal and application of knowledge to maximize an enterprise's knowledge related effectiveness and returns from its knowledge assets" ([40] Wiig, 1997).

- "KM is getting the right knowledge to the right people at the right time so they can make the best decision" ([31] Pettrash, 1996).

Knowledge is not only gained from employees' skills but also from all the organization's environmental elements and the understanding of their relationship which is referred to as the "knowledge landscape" ([33] Siemers and and Inc. Arc Partners, 2000).

Carla O'Dell of the American Productivity and Quality Center lists the following as the primary ways that knowledge adds value ([34] Skyrme and Amidon, 2000):

- KM as a business strategy - in products and processes;

- innovation and knowledge creation - new products, rapid commercialization, and renewing unique knowledge and expertise;

- transfer of knowledge and best practices - improving customer service, reduced cycle time or repair times;

- customer focused knowledge - building customer intimacy and working with them to make them successful;

- intellectual asset management - realizing the value in intellectual assets; and

- personal responsibility for knowledge - encouraging individual learning and development.

Most of these apply to organizations' direct business missions. However, it is clear that many can benefit internal business processes such as the management of software outsourcing.

While the literature is replete with the many benefits that KM can provide to an organization, initiating a KM program is not without its pitfalls. To be successful, KM programs require champions among upper management, middle management commitment, buy-in among users, and a solid technical infrastructure to support it. There are different approaches, models and methodologies for initiating and institutionalizing a successful KM program, but they all seem to agree on three interdependent core elements: people, processes, and technology. People acquire and hold the knowledge. Processes are the activities involved in generating, organizing and distributing knowledge. Technology consists of the information technology tools needed for the implementation of KM, the "enabler." Different models have been developed which graphically represent the complex operational and organizational aspects of KM. The models usually include the traditional aspects that need to be addressed such as the people, process, and technology and may include others aspects such as leadership and learning. Each aspect is usually further sub-divided. An example of such a model is the conceptual knowledge management framework developed by [6] Calabrese (2000) and [2] Baldanza and Stankosky (1999) shown in Figure 1 [Figure omitted. See Article Image.].

The following are descriptions of the interconnected factors or pillars as they are called:

- *Leadership.* Addresses and implements the business strategy, and climate which the strategy necessitates. Leadership interacts with the environment to position itself for success.

- *Organization.* The organizational structure that supports the strategy, the business processes, and performance management system to adapt to change.

- *Technology.* Technology consists of the tools needed to support the strategy.

- *Learning.* The lessons learned for improved effectiveness and/or efficiency. A continuous process built from managing information.

A KM initiative must include strategies that address all of these factors to be successful.

**KM in software acquisition**

Organizations involved in outsourcing their software projects can benefit from KM in managing many aspects of the software acquisitions: planning, managing software requirements, tracking project team and contractor team performance, managing the project's cost and schedule baselines, evaluating products and services, and transitioning the software to its support organization. The potential payoffs to an organization are compelling. According to a Washington Post article, the federal government spent $45 billion on IT products and services in fiscal 2001 ([3] Balluck, 2001). Defense giant Lockheed Martin estimated that "government IT spending will increase to $60 billion by 2006." At the same time, results of [25] KPMG Consulting (2003) Knowledge Management Survey 2002/2003 showed that 78 percent of companies believe they are currently missing out on business opportunities by failing to successfully exploiting available knowledge. Companies believe that an average 6 percent of revenue as percentage of total turnover or budget annually is being missed from failing to exploit knowledge effectively!

Research results by the authors emphasized the need for institutionalizing processes for the collection of data about contracting costs within an organization, so that databases of metrics about completed projects can be built and later used to forecast costs for future projects and to improve the decision-making processes. This may effectively be accomplished through the application of the KM model which provides a framework for identifying the organizational structures, processes and informational technologies for collecting and measuring acquisition data incurred before, during and after the contract award. KM can also be used in risk management by providing means for proactive decision making to assess what can go wrong (risks), determine what risks are important to deal with, and implement strategies to deal with those risks.

The following is a list of some of the areas within software outsourcing management where KM can be applied:

- acquisition planning;

- request for proposal (RFP) preparation;

- contractor profiles management;

- contractor evaluation and selection;

- proposal evaluation;

- requirements management;

- project profiles management;

- determination of resource requirements;

- personnel skills and expertise evaluation and management;

- acquisition technologies (hardware and software tools required to support the acquisition effort);

- acquisition best practices;

- acquisition policy for software systems;

- deliverables and deliverable documents contractual obligations;

- quality evaluation criteria;

- actual project cost;

- project schedule;

- project technical assessments;

- program reviews guidelines;

- check lists for life cycle deliverables monitoring;

- contractual compliance;

- evaluation of software development artifacts;

- resources-risk management;

- project-risk management; and

- lessons learned.

Archives of knowledge about these areas along with KM tools can be used on future projects to improve acquisition processes and identify risks and mitigation strategies, all leading to a higher probability of project success.

**KM strategies for software acquisition**

Developing a strategy for a KM initiative is vital for a software acquisition organization. The strategy is a road map for the organization, determines whether the organization has committed to a program of KM improvement, and indicates how the organization plans to manage software outsourcing to ensure business benefit. The following strategies are critical for success.

**People strategy**

Stresses the important role of people in KM. The strategy must address executive commitment, leadership, organizational communication, employee motivation and incentives systems, and cultural changes needed to establish a knowledge-sharing environment.

**Process strategy**

Is concerned with activities involving continual innovation and implementation of ever-better process practices, which lead to improved organizational performance and capabilities. The strategy should address what kinds of knowledge is required, its sources, how it will be captured and shared by project managers, contractors, functional experts, employees and others. It must also address the operational KM needs and the plan to implement the strategy.

**Technology strategy**

Includes the strategy for using innovative information technology tools for performance improvements. This strategy should take into consideration the technical infrastructure that will enable workers to create and share knowledge.

**KM processes for software acquisitions**

The following section describes in more detail the stages of a strategy for KM processes and how they relate to software outsourcing.

**Knowledge identification**

This stage involves the identification of critical knowledge that the organization has and needs about software acquisition. This stage can employ knowledge mapping which is the ability to identify where expertise and knowledge reside throughout the organization and contractors base. It can also be used to highlight areas where there is a lack of knowledge. Knowledge about past details of acquisition projects, development cycles, contractors, resources, documents, best practices, and lessons learned will be the focus.

**Knowledge creation**

This stage involves creating repositories of data and information about previous acquisition projects. This knowledge is traditionally clustered in different formats and in different locations. Internally, knowledge is captured from all employees who were involved in acquisition projects such as project managers, technical leads, team members, users, testing and quality assurance personnel. Additional of knowledge can be obtained from previous projects documents, experience and practices. Other information that can be obtained from external sources include: best practices, input from consultants, technological trends, data from users, first impressions from new staff, collaborations with other organizations, and knowledge links with business partners.

**Knowledge retention**

This stage includes all activities that preserve knowledge artifacts and allow them to remain in the system once they have been introduced. It also includes "those activities that maintain the viability of artifacts within the system" ([29] Newman and Conrad, 1999). At this stage, knowledge can be retained in an information/KM system.

**Knowledge transfer**

This stage involves "leveraging the many ways that knowledge migrates into the organization and strengthens business performance" ([9] Cross and Baird, 2000). Once knowledge is captured and stored, it must be shared and made available to anyone who needs it. Communications architectures must be in place (e.g. intranets) to allow users across all branches to have access to any piece of the organizational knowledge.

**Knowledge utilization**

Knowledge utilization refers to the application of the knowledge to solve problems, adapt the knowledge to new situations (i.e. new acquisition projects) and create new knowledge or update existing knowledge.

**Recommended KM approaches for software acquisition**

What is inferred in the above discussions is the process of codification and collaboration (also called personalization). Codification and collaboration are practical and intuitive approaches for implementing a KM program for organizations involved in software acquisitions.

**Codification**

Once knowledge related to acquisition is identified, it must be organized in a form that can be easily used and shared. Knowledge can be explicit (expressed, codified, formalized) or tacit (not easily expressed, codified and formalized). The problem with applying KM to software acquisition is that knowledge, like all KM applications, must be formalized, or made explicit in order to be "electronically" available ([1] Beckman, 1997). Interviews, observation, after action reviews and knowledge elicitation can be used in order to convert tacit knowledge into explicit knowledge ([9] Cross and Baird, 2000).

The codification approach can be used to format the knowledge about projects, contractors, resources, costs and other characteristics that are pertinent to software acquisition. This "codification approach" is intended to collect, codify and disseminate information. One of the benefits of the codification approach is the reuse of knowledge:

The aim of codification is to put organizational knowledge into a form that makes it accessible to those who need it. It literally turns knowledge into a code (though not necessarily a computer code) to make it as organized, explicit, portable, and easy to understand as possible ([11] Davenport and Prusak, 1998).

Knowledge is codified and stored in databases, where it can be accessed and used easily by anyone in the company. Knowledge is codified using a *people-to-documents approach;* it is extracted from the person who developed it, made independent of that person, and reused for various purposes ([21] Hansen *et al.* , 1999).

Codification relies heavily on IT, an electronic document system maybe developed that codifies, stores, disseminates, and allows reuse of knowledge. Best practices and lessons learned can be tied to specific types of projects, areas of interest, or processes. It is critical that this knowledge be readily available to all members of the acquisition community. At a minimum, in the absence of KM tools and structured repositories, some organizations publish experiences and lessons learned in unformatted databases that are made available on organizational intranets. This approach will have to be complemented by the collaboration approach in order to become a fruitful KM activity for software acquisition projects, it is at least a start that can be easily launched for those who intend to implement a full-blown KM system.

**Collaboration**

Involves developing networks for linking people so that tacit knowledge can be shared. Collaboration requires moderate investment in information technology since the goal of a KM activity is to facilitate interaction and the exchange of tacit knowledge. Collaboration involves encouraging communities of practice; bringing together people with similar interests to share and exchange their ideas, experiences, solutions, and knowledge. Personnel involved in current and past acquisition project are key participates who can share and access experiences, lessons learned, and best practices. Processes and tools must be used to try to capture as much as possible the exchange of tacit knowledge that happens during these human interactions. Archiving e-mail communications, instant messaging sessions, video conferencing sessions, and using online/electronic communities and discussion groups is a first step in capturing such knowledge exchanges.

In order to manage better and to systematically achieve levels of proficiency, software acquisition projects and system life cycle processes and practices like the Software Acquisition Capability Maturity Model (SA-CMM ® ) can be applied (Table III [Figure omitted. See Article Image.]).

Currently, a lot of software acquisition practices focus mainly on the codification approach where heavy documentation is required for maintenance and traceability issues. Through such processes mainly explicit and some tacit knowledge are captured but a need for better communication and collaboration remains to leverage the knowledge associated with the various phases of the SA projects. A stronger emphasis on the collaboration approach is needed and it will also result in improving innovation.

Owing to space limitation in this paper we will not be able to describe in details how KM could be used in every stage of the software acquisition process but we selected a very specific example associated with the development phase and that reflects how the lack of KM at this stage could impact future phases of the project.

During the development phase programmers are faced with a lot of issues that are solved and that are not often documented or poorly documented. These issues might not only be related to the application developed or customized but also to the environment in which it operates (operating system, database, API, etc.). Capturing and storing these solutions will be a great value for the deployment, maintenance and re-uses phases. The ways that problems are solved depend sometimes on changing settings/configuration parameters of the IT environment that cannot be documented in a program source comment file. These changes can sometimes be considered as "tricks" exchanged between developers or that have been acquired through experience (workaround features/shortcuts). They often belong/remain in the head of few members of the development team which is likely to disband after the end of the project. The need to capture this tacit knowledge is critical. [27] Makkinejad (2005) proposes to create an application specific knowledge base that will capture the following knowledge artifacts:

- undocumented features used in developing the system;

- poorly documented features used in developing the system;

- defect work-around features used in developing the system;

- version/patch information of the building blocks of the system;

- implemented kludges;

- discussion of the technical reasons for using these features;

- things that could go wrong if these features cease to work in a future release;

- recommendations for replacing these features in future releases;

- date and time stamp for all entries in the knowledge base;

- preferred methods for incorporating features used in developing the system; and

- standards and guidelines that should be used to govern system development.

This knowledge base will be successful only if the developers accept it and use it. Very often, developers are under time pressure and focus their efforts on developing the product, the surrounding activities being seen as accessories (low-value added). The need for a culture change supported by strong leadership is required to make this shift happen. As [27] Makkinejad (2005) suggests, the knowledge base should be application specific and should be though as an undocumented corner magazine column, a pitfall list or a compilation of programmer's shortcuts. We agree on this approach and we think that this application specific approach should be part of a broader system acquisition knowledge base that could be built around a community of practices (CoP) environment. In such environment, developers will not only be able to codify how they solved particular issues, but it could also become a place where they will be able to learn and to search for solutions on how to solve problems (lessons learned, best practices, tools, etc.). Such knowledge base will require to be initially populated with a critical mass of knowledge (based on historical cases) to be recognized as useful by developers. The knowledge in such base will have to be maintained and validated and security features should be available to limit/restrict access to some confidential content. Developers will need to clearly see the benefits from getting involved in such communities (what's in it for them?) and the answer to this question should be: to learn, to be rewarded for their contribution and to be recognized among their peers.

Such communities of practice could be extended to other aspects of the software acquisition process. Some of them could focus on metrics issues, risks mitigations and costs. Such communities could become a valuable complement to databases of historical projects data since they will contain valuable knowledge and not just static data and information.

**Conclusions**

Organizations that outsource many of their software projects may benefit greatly from a KM program for software acquisition. Such a program may help the organization to: plan all aspects of the acquisition, manage software requirements, track project team and contractor team performance, manage the project's cost and schedule baselines, evaluate the products and services, manage risks, and successfully transition the software to its host organization. The utilization of KM by the acquisition community can have a profound impact on their ability to conduct software acquisitions and utilize their resources effectively. Acquisition based on a sound KM implementation can influence costs of ownership, improve schedules, improve economic analysis and decision making, enable the integration of innovative technologies, share best practices and manage risks - i.e. organizations can work smarter! Both KM approaches (codification and collaboration) should be used concurrently to better manage the knowledge associated with the various phases of acquisition projects. In the past a lot of emphasis was put on the codification approach focusing mainly on the explicit knowledge but organizations are slowly realizing the value associated with complementing such approach with a collaboration approach in order to leverage knowledge and to foster innovation. We believe that implementing communities of practice including the various phases of the software acquisition process could be used as a starter. But, as we previously stated, IT is only the enabler and without a culture change/evolution tacit knowledge will not be easily shared. Every actor in the SA process should become involved in one or many community (ies). These communities should be transversal meaning that they should not be centered around a particular phase of the SA project but they should be centered around a topic/issue that may involve or interest participants of various SA phases. From this diversity and from these interactions will emerge some valuable knowledge artifacts and innovation will flourish.