6

# Managing Knowledge in Collaborative Software Maintenance Environment

Mohd Zali Mohd Nor, Rusli Abdullah,
Masrah Azrifah Azmi Murad and Mohd Hasan Selamat
*Universiti Putra Malaysia*
*Malaysia*

## 1. Introduction

In recent years, many organizations consider knowledge management (KM) to be strategically important to their business. KM is envisaged to contribute to the organizations in the following manners (KPMG, 2003):

- Bring synergies among different teams, units or departments
- Accelerate innovation and boosting revenues for market development.
- Improve quality in operational and functional processes
- Reduce costs and exposure to business risks

With the above 'promises', KM is also enticing to software development and maintenance organizations, especially in managing software engineering activities. Within software engineering activities, software maintenance (SM) has yet to receive proper attention (SWEBOK, 2004). It is a costly process, where previous works (Fjeldstad & Hamlen, 1979; Lientz et al., 1981; Pigoski, 1997; Schach et al., 2003) estimated SM costs of between 60% to 90% of total software life cycle costs.

In Software Engineering area, KM have been studied mostly on Software Development environment, but in Software Maintenance (SM) environment, KM has not been widely used nor studied. Therefore, studies on KM in SM shall be beneficial to the SM communities to assist them to perform their daily activities .

The motivation to apply KM in SM in driven by the fact that the SM activities are knowledge-intensive (Rodriguez, 2004a), and depend largely on expertise of the maintainers. Most of the time, maintainers depend on experience and "hunches" when making decisions. Some of these expertise are documented as explicit knowledge, but more are hidden as tacit knowledge due to scarcity of documentation (Viscaino et al., 2003). As SM organizations grow and becoming more distributed, members shall need to collaborate and share these individual knowledge. Various artefacts are 'created' and shared during the SM activities. Among them are:

- Problem reports (PR) (a.k.a. incident report, call tickets) – recorded in helpdesk application, the PR shall remain open and notes are appended until the problem is resolved, or Maintenance Request (MR) is raised.

- MR – A request for maintenance task, either to fix production bug raised via PR, business enhancement, or perfective and preventive maintenance. Normally MR shall be registered in a Software Configuration Management (SCM) application. MR shall be versioned and remain open until the tasks are completed.
- Business Requirement Documents (BRD) (a.k.a. requirement specifications) - For business changes, a BRD is normally required to explain the current application and the intended changes
- Software Requirement Documents (SRD) (a.k.a. software specifications) – In addition to BRD, SRD details out the technical specifications to guide maintainers on the required changes
- Test Plan – a guideline for QA to perform testing on the MR
- Release Notes – a list of changes made for a specific release or version.
- Known Issues – a list of known issues for high-priority MR that could not be completed, with the workarounds, if applicable.

In many SM organizations, the above artefacts are kept in SCM, using various 'containers' such as MS-Word, MS- Excel, pdf and hence making searching difficult. As such, maintainers often have to resort to checking the code to derive the knowledge (Das et al.,2007). Notwithstanding, many other information that are important to maintainers resides somewhere else. For example, the domain knowledge often resides within users community, either explicit in form of best practices, policies and procedures, circulars and others, or implicit, in the mind and experience of the expert users. Are these knowledge important and pertinent to the other parties? In SM, the answer is a resounding yes. As expert users and maintainers leave the organization, the implicit knowledge are gone with them. This is where KM is useful to consolidate all these information together and allow users and maintainers to contribute, share and store knowledge.

In this chapter, we shall present the followings: review the definitions and concepts of KM, KMS and SM collaborative environment; propose a KMS framework for collaborative SM environment; present a Multi Agent System (MAS) tools to support users and maintainers in knowledge sharing; and an combined ontology to structure the required knowledge to be used by the MAS tool

## 2. Knowledge Management

As an overview, knowledge is defined as "*a fluid mix of framed experience, values, contextual information and expert insight that provides a framework for evaluating and incorporating new experiences and information. It originates and is applied in the mind of knowers. In organizations, it often becomes embedded not only in documents an repositories but also in organizational routines, processes, practices and norms.*" (Davenport & Prusak, 2000).

Meanwhile, KM, in technical perspective, is defined as the strategies and processes of identifying, understanding, capturing, sharing, and leveraging knowledge (Abdullah et al., 2006; Alavi & Leidner, 2000; Davenport & Prusak, 2000; Selamat et al., 2006). For individual knowledge creation cycle, Nonaka & Takeuchi SECI framework (Nonaka & Takeuchi, 1995), based on Polanyi's tacit and explicit knowledge, models the knowledge creation stages of socialization, internalization, combination and externalization. This SECI model has been used and synthesized by many others to model the KM for team and organization levels.

The knowledge creation cycle in SM environment and the collaboration technologies used are depicted in the following Fig. 1.

| **Tacit to tacit knowledge via Socialization** | **Tacit to explicit knowledge via Externalization** |
|---|---|
| SM knowledge are exchanged through experience sharing, brainstorming, observation and practice.<br><br>*Today technologies*:<br>Collaboration tools - teleconferencing, desktop video conferencing tools, live-meetings, village wells, synchronous collaboration | Articulate tacit knowledge into explicit via concepts, metaphor, or models. In SM cases, these could be in form of screenshots of errors, shadow sessions, emails, conversations<br><br>*Today technologies*:<br>Email, terminal sessions, chat |
| **Explicit to tacit knowledge via Internalization** | **Explicit to explicit knowledge via Combination** |
| Knowledge is documented or verbalized, to help maintainers internalize and transfer knowledge, and also help other maintainers to 're-experience' bug scenarios.<br><br>*Today technologies*:<br>Helpdesk and SCM applications are used to store bug reports and changes. Visualization tool to read or listen to success stories. | Knowledge are combined, sorted, added , exchanged and categorized, via specifications, SCM entries and error analysis<br><br>*Today's technologies*:<br>Collaboration tools - E-mail, GroupWare, Homepages, consolidates in SCM. Data mining to sort, and filter information. |

Fig. 1. SM Collaboration technologies in Knowledge Creation Cycle- Adapted from Nonaka & Takeuchi SECI Model

### 2.1 Knowledge Management Framework

KM frameworks for modeling organization knowledge cycles are useful to understand strategies and processes of identifying, understanding, capturing, sharing, and leveraging knowledge within the teams, departmental units and organizations. Among few are frameworks by Szulanski's model of knowledge transfer (Szulanski, 1996), APQC's organizational KM model (Arthur Anderson and APQC, 1996), Choo's model of knowing organization (Choo, 1996), Selamat et al.'s KM framework with feedback loop (Selamat et al., 2006) and Holsapple and Joshi's 3-fold collaborative KM framework (Holsapple & Joshi, 2002). This framework synthesizes the knowledge resources from Leonard-Barton, and Petrach and Sveiby models; KM activities from Nonaka, APQC, Wiig, Van der Spek and Alavi's models, and KM influences from Wiig, APQC, Van der Speck, Szulanski and Leonard-Barton models. The summary of the above frameworks are listed in the following Table 1:

| Dimension/ Framework Model | KM Activities | | Strategy | Enabler/ Enabling condition |
|---|---|---|---|---|
| | **Activities** | **Process** | | |
| **Wiig (1993) 3 pillars of KM** | Creation, Manifestation, Use, Transfer | Pillar 1 - Survey and categorize, Analyze knowledge and activities, Elicit, coding and organize<br>Pillar 2 - Appraise and evaluate, Action<br>Pillar 3 - Synthesize, Handle, use and control, Leverage, distribute and automate | | |
| **Nonaka & Takeuchi (1995) Knowledge creation** | Socialization, Externalization, Combination, Internalization | 5-phase model of K-creation process: Sharing tacit knowledge,Concept creation,Concept justification,Archetype building,Cross-levelling | | Intention, Autonomy, Creative Chaos, Redundancy, Requisite Variety |
| **Szulanski (1996) Knowledge Transfer model** | Knowledge transfer - Initiation, Implementation, Ramp-up, Integration | | | K-transfer influences - Characteristics of k-transfer, k-sources, recipient, context |
| **APQC (1996) Organizational KM model** | Share, Create, Identify, Collect, Adapt, Organize, Apply | | | Leadership, Culture, Technology, Measurement |
| **Van der Spek & Spijkervet (1997) Four-Cycle KM stage** | | Conceptualize, Reflect, Act, Retrospect | | |
| **Choo (1998) The Knowing Organization** | Sense making, K-creation, Decision making | | | |
| **Davenport & Prusak (2000) Working Knowledge** | Knowledge generation-Acquisition, Rental, Dedicated resources, Fusion, Adaptation, Networks<br>Knowledge codification and coordination-Mapping and modeling knowledge, Capturing tacit knowledge, Codifying knowledge | | Monopolies, Incompleteness of information, Asymmetry of knowledge, Localness, Artificial scarcity, Trade barriers | Price system - reciprocity, repute, altruism, trust<br>Knowledge market - buyer, seller, brokers |
| **Hansen, Nohvia & Tiernes (1999) KM Strategy** | | | Codification, Personalization | |
| **Australia KM Standards (2001) Integrated KM framework** | Knowledge process - Sharing, Acquisition, Creation | | Knowledge alignment - Context, Analysis, Planning | Knowledge Foundations - Culture, Technology, Sustaining systems |
| **Holsapple and Joshi (2002) 3-fold collaborative KM framework** | Acquiring, Selecting, Internalizing, Using | | | KM Influences - Resource, Managerial,Enviromental influences<br>Knowledge Resources - Participants knowledge, Culture, |

| | | | | |
|---|---|---|---|---|
| | | | | Infrastructure, Purpose, Strategies |
| **Handzig & Hasan (2003) Integrated Organizational KM framework** | | | | Enablers - Knowledge process, Knowledge stock, External environment<br>Organizational factors - Organizational environment, Technological infrastructure |

Table 1. KM Frameworks - Theoretical Construct

### 2.2 Knowledge Management System Framework

To conceptualize the KM frameworks into a Knowledge Management System (KMS ), a KMS framework shall need to be defined for collaborative SM environment. KMS is defined as "*I.T-based system developed to support and augment the organizational process of knowledge creation, storage, retrieval, transfer and application*" (Alavi & Leidner, 2000). In general, a KMS framework consists of influential factors of KMS initiatives and their interdependent relationships and a model of KMS implementation (Foo et al., 2006). However, systems and technology alone does not create knowledge (Davenport & Prusak, 2000), various other social "incentives" and organizational strategy and culture are often required to stimulate use of technology to share knowledge.

In this chapter, we review some of the related KMS frameworks and identified the components that could be synthesized for knowledge-based collaborative SM framework.

| **Dimension Framework Model** | **Activities & process** | **Functionality** | **Technology** | |
|---|---|---|---|---|
| | | | **Tools** | **Architecture** |
| **Meso & Smith (2000) Technical perspective of KMS architecture** | Using, finding, creating, packaging<br><br>Know how, know what, know why, Self-motivated creativity, Personal tacit, Cultural tacit, Organizational tacit, regulatory assets | | Computer-mediated collaboration, Electronic task management, Messaging, Video conferencing, GDSS, Web browser, Data Mining, Search and retrieval, Intelligent Agent, Document Management | |
| **Natarajan & Shekar (2000) Dual-KM Framework** | Generation,<br>storage,<br>application | | | Knowledge enterprise - OSI 7-layer model |
| **Hahn & Subramani (2000) Framework of KMS** | classifying KMS based on the locus of the knowledge and the a priori structuring of contents | | Document repository,<br>Data warehousing,<br>Yellow pages of experts,<br>Electronic discussion forum,<br>collaborative filtering,<br>Intranets & search engine | |

| | | | | |
|---|---|---|---|---|
| **Alavi & Leidner (2000) KMS Process framework** | Creation, Storage, Retrieval ,Transfer, Application | Coding and sharing best practices, Corporate K-directories, Knowledge network | | |
| **Rao (2005) 8'Cs audit framework** | | Connectivity, content, community, culture, capacity, cooperation, commerce, capital | | |
| **Abdullah et al. (2008) Collaborative KMS framework** | Acquisition, store, disseminate, use. Soft Components - Awareness, Reward, Motivation, Culture, Strategy, beliefs, values, experience | | Portal, EDMS, Workflow, OLAP, Agent | Infrastructure, technology, protocol, repository |
| **Deraman(1998) KMS model for SM** | Software knowledge, Change Request knowledge | | | |
| **Rus and Lindval (2001) KMS framework for SE** | 3 levels of KM Support in SE -<br>1st Level: Document mgmt, competence mgmt.<br>2nd Level: Store organizational memory, design rationale, SCM.<br>3rd Level: Packaged knowledge | | | |
| **Dingsoyr & Conradi (2002) Knowledge management "system"** | Method to manage tacit knowledge, explicit knowledge | | Infrastructure, Software systems, Experience management system | |
| **Rodriguez et al. (2004b) KMS in SM** | Collecting, distributing knowledge | | Active tools, passive tools | |
| **De Souza et al. (2006) KM framework in global software development** | Organizational Focus, Degree of structure, Knowledge repositories in place | | Client-server, Peer-to-peer (P2P), Hybrid | |

Table 2. KMS Frameworks - Theoretical Construct

## 3. Collaborative Software Maintenance Environment

As an overview, software maintenance (SM) is defined as "*The totality of activities required to provide cost-effective support to software system. Activities are performed during the pre-delivery stage as well as the post-delivery stage*" (IEEE 14764, 2006; SWEBOK, 2004). SM activities are complex, knowledge-intensive (Rodriguez et al., 2004a), and depend largely on expertise of the maintainers and expert users, as depicted in Fig. 1. Software maintenance processes and activities have been largely standardized. Standard organizations such as ISO, IEEE, and CMMI have detailed the activities to be carried-out by software maintainers (April et al., 2005; IEEE 14764, 2006). At a very minimum, the activities include process implementation, problem and modification analysis, modification implementation, maintenance review and acceptance, migration and software retirements.

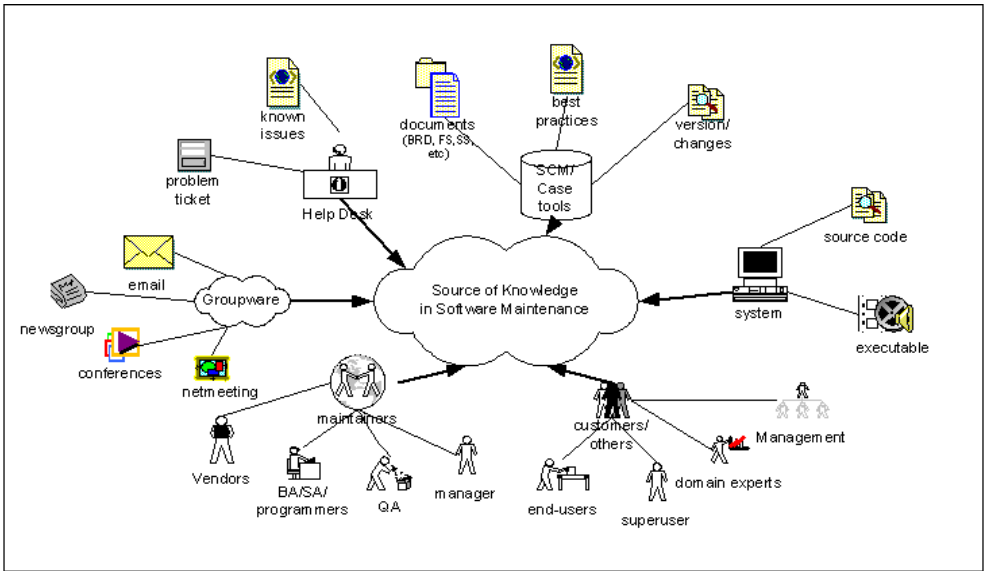


Fig. 2. Sources of Knowledge in SM

However, many software maintenance organizations may have their own best-practice processes and activities to suit the organizational and business practices. Detail activities may vary depending on the following setups:

- Types of maintenance organizations – such as in-house maintenance, vendor or outsourced maintenance, or commercial-of-the-shelf (COTS) application maintenance.
- Team setup – such as similar or separate development and maintenance team.
- Types of software or applications being maintained (Pressman, 2005)
- Maintenance approach or model – For example, those using Waterfall approach may differ from those using Agile approach.

As a broad example, the SM activities are depicted in Fig. 3 below:

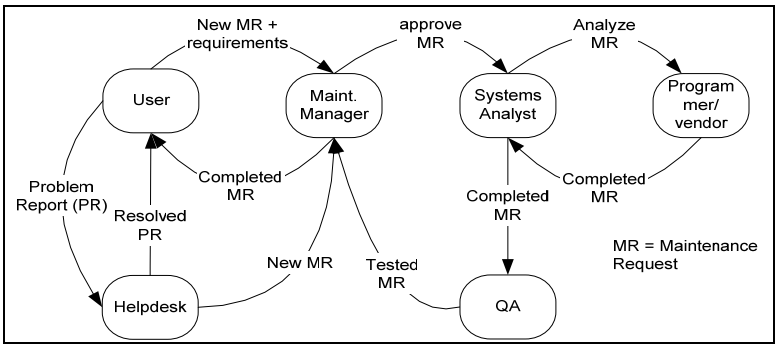


Fig. 3. Sample maintenance activities

Meanwhile, the knowledge needed in SM can be summarized as follows (Ghali, 1993; Rodriguez, 2004a; Rus & Lindvall, 2001):

- Organizational knowledge, such as roles and resources. The parties involved in software maintenance activities consist of various application users and software maintainers. The list may include end-user, superuser, maintenance manager, business analyst, systems analyst, project manager, QA personnel, build manager, implementation personnel and trainer. Attached to these roles are the area of expertise.
- Managerial knowledge - such as resource management, task and project tracking and management.
- Technical knowledge – such as requirement analysis, system analysis, development tools , testing and implementation. Critical to this is also the knowledge on supporting groupware and CASE tools such as Software Configuration Management (SCM), helpdesk and testing tools
- Domain knowledge – knowledge of the products and business processes.
- Knowledge on source of knowledge – where the knowledge resides, such as source codes, documentation, supporting CASE tools and more importantly, the where the experts are.

There are various issues associated with the above knowledge, which makes organizing, storing, sharing and disseminating knowledge difficult. Among the problems are:

- The 'containers' could be in different electronic forms, which sometimes need to be mined and manually extracted. Or worse, in paper form which require more effort to place it in KMS
- Documentation are most of the time not up-to-date. As mentioned earlier, Earlier studies indicates around 50% of efforts are spent on this activity and rely more on source code than any other source of information (Fjeldstad & Hamlen, 1979; Schach et al.,2003)
- Domain knowledge are becoming more important to software maintainers (Santos, 2005). However, this knowledge are often not available within the software maintenance CoP, especially in vendor and distributed environment. Changes to business processes and changes to application affects not only the business users, but also software maintainers
- Human experts hold most of the tacit knowledge that are not readily available to others. are the major source of knowledge. However, there are still reservation toward knowledge sharing. ''*If getting promotion, or holding your job, or finding a new one is based on the knowledge you possess - what incentive is there to reveal that knowledge and share it?*' (Wilson, 2002).
- The above problems are further exacerbated in distributed maintenance teams, where members resides in different location and time-zones. As such, face-to-face meetings are seldom and tacit knowledge transfer is difficult.

Managing knowledge in this area is therefore critical to ensure that both users and maintainers can perform SM activities properly and timely, by sharing and obtaining vital knowledge.

### 3.1 Knowledge-based Framework for Collaborative Software Maintenance

KMS for SM has been studied late 1980s by Jarke and Rose (1988), who introduced a prototype KMS to control database software development and maintenance, mainly to facilitate program comprehension. The KMS is a decision-based approach that facilitates communication across time and among multiple maintainers and users, thus improving maintenance support. However, facilitating program comprehension is not enough as SM is more than just understanding codes and extracting knowledge from codes.

Similarly, Deraman (1998) introduced an KMS model for SM which, albeit very simple, could provide us with the main essence of SM knowledge – the Software Knowledge, Change Request Knowledge and their functional interaction. However, these alone, are not enough for users and maintainers. Newer technologies such as software agents are used to capture SM process knowledge in researches conducted by Viscaino et al.(2004) and Rodriguez et al. (2004b). However, no KMS framework for SM was conceptualized by these studies.

Looking at the wider perspective of software engineering (SE), KMS in SE have been studied by many, including Santos et al. (2005), Rus and Lindval (2001) and Aurum et al.(2003). Rus and Lindval described the three main tasks of SE (individual, team and organization) and identified the three level of KM support for each task. The 1st level includes the core support for SE activities, document management and competence management. Meanwhile, the 2nd level incorporates methods to store organizational memory using method such as design rationale and tools such as source control and SCM. The 3rd KM support level includes packaged knowledge to support Knowledge definition, acquisition and organization. The above should describes the KMS framework for SE. However, this model do not consider the social, physiological and cultural aspects of KM, as identified by the previous other generic KMS frameworks.

In order to propose a suitable KMS framework for SM,, a review of current KMS framework for generic KMS, and related SE/SM KMS are conducted. The theoretical constructs for KM frameworks, KMS frameworks and knowledge components in SM are summarized, and components suitable for SM KMS framework are identified, as follows:

- Required knowledge, such as organizational knowledge, managerial knowledge, technical knowledge, enterprise business domain knowledge and knowledge on source of knowledge, are derived from Ghali (1993), Rus and Lindval (2001) and Rodriguez et al. (2004a)
- KM Activities are derived from Nonaka and Takeuchi (1995), Holsapple and Joshi (1998). This includes Acquiring knowledge, Selecting knowledge, using knowledge, Providing/ Creating knowledge and Storing knowledge.
- SM governance tools are from Rus and Lindval (2001), IEEE 14764 (2006) and Mohd Nor et al.(2008a). To support these flow of SM information, tools such as Helpdesk, Software Configuration Management (SCM), Source Control and Project Management (PM) are crucial to monitor MRs.
- KM Components and Infrastructure are derived from Abdullah et al. (2006), Meso & Smith (2000) and Rus and Lindval (2001) frameworks. The major components includes computer-mediated collaboration, Experience Mgmt System, Document Management, KM portal, EDMS, OLAP, and Middlewares tools.
- Automation and knowledge discovery tools are from Meso and Smith (2000), Abdullah et al. (2006), Rodriguez et al. (2004b) and new internet tools in the

market. Tools such as GDSS, Intelligent Agents, Data mining/warehouse, Expert system and Case-Based Reasoning (CBR). Active tools such as RSS are also useful to get the right knowledge to the right users at the right time.

- KM Influences are derive from Holsapple and Joshi (2002) and Abdullah (2006). Among these are the managerial influences and strategy, and psychological and cultural influences.

To summarize the collaborative SM in perspective of People, Process, Technology and Knowledge Content, the following dimensions are proposed:

| **Knowledge Dimension** | | **Relevance to SM** |
|---|---|---|
| People | Organization | Routine, rules, culture |
| | | Enterprise Domain knowledge |
| | Team | Knowledge on roles, expertise and their location |
| | Individual | Technical skills - requirement analysis, systems analysis, programming, testing and implementation |
| | | Managerial skills - MR management, resource planning |
| | | Domain expertise |
| Process | Organizational | Best practices, culture, strategy, psychological influences |
| | Regulatory | Audit, data security |
| | Best Practices | SM best practices, Software Configuration Management (SCM) process, Versioning process |
| Technology | SM tools | SCM, Version Control, Source Control, Project Management, Helpdesk tools |
| | KM tools | KMS portal, Search engine, Data warehouse, EDMS, OLAP, and Middlewares tools |
| | Collaboration | email, e-group, wikis, SMS, MMS, mobile technologies |
| | Automation & K- discovery | GDSS, Intelligent Agents, Data mining/warehouse, Expert system, RSS and Case-Based Reasoning (CBR) |
| Content | Domain knowledge | Products, Business rules |
| | Knowledge Map | Ontology, yellowpages |
| | Software artifacts | |

Table 3. People, Process, Technology and Content Model for Knowledge-based Collaborative SM

Based on the above, the model for knowledge-based collaborative SM framework is proposed, as per Fig. 4 below:

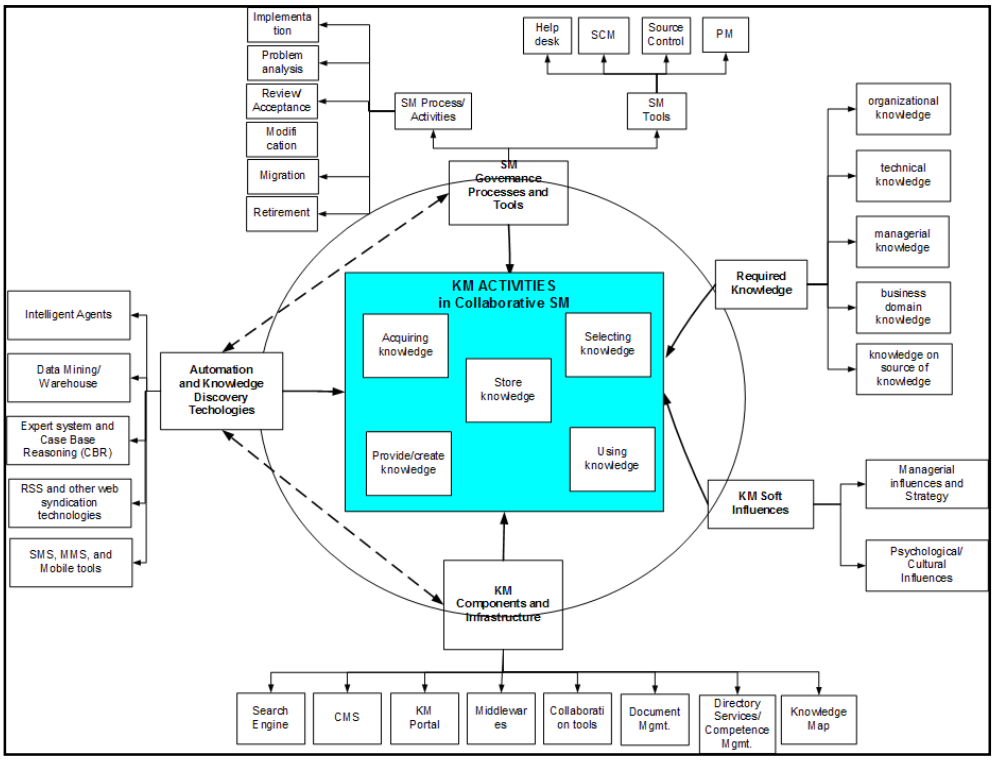

Fig. 4. Knowledge-based Collaborative SM Framework

### 3.2 Managing Knowledge in Collaborative SM

To provide for all the above components for knowledge-based collaborative SM system is the ultimate goal. However, this will require tremendous efforts and overall revamp of the SM process management tools. In our previous studies, much of the knowledge-based SM tools are siloed and not integrated to allow seamless knowledge combination, which hampers knowledge acquisition and sharing. This was further supported by a survey on managing knowledge of SM process in higher learning institutions (HLI) in Klang Valley, Malaysia, several major issues were identified as follows (Mohd Nor & Abdullah, 2008b):

- 80% of the surveyed SM organization do not use KMS to store knowledge acquired during the maintenance activities. Hence, this could contribute to problems and lateness in getting the information from other experts.
- In various aspects of SM activities (helpdesk, planning and analysis and coding and testing), between 60% to 80% of respondents consider domain knowledge important to assist them in daily SM activities. However, they admit that the knowledge generated from these activities are not stored in KMS or other electronic means, thus making them harder to extract, shared and turned explicit.
- Substantial efforts are spent collaborating with users, experts, SAs, and vendors to ascertain the problems and requirements. In the survey, 41% and 20% of helpdesk time are used to collaborate with maintenance team and users, respectively.

Meanwhile, in the planning and analysis, 22% of the time is spent discussing issues with users, 20% with colleagues and another 20% with developers. Without a systematic approach to these information acquisition and sharing, these efforts shall remain a major issue.

- Overall, in term of the perceived problems, quality of application documentation and inadequate system support remain as major issues.

One good way of solving the above issues is via automation. According to Davenport and Prusak (2001), one of the main goal of a KM system is to automate, as much as possible, the tasks of acquiring, disseminating and storing of knowledge. With all the sources of knowledge located and traversing in different repositories via different tools, keeping track of information useful for both users and maintainers could be a nightmare.

In this chapter, we shall introduce an automation mechanism to allow users and maintainers to acquire, share and use knowledge during software maintenance activities, vis-à-vis the following functionalities:

- Assist users in reporting errors, by checking for previously related reported errors, known issues and related enterprise business domain rules. This would help to reduce unnecessary duplicate errors that Helpdesk personnel need to handle.
- Assist Helpdesk personnel to monitor helpdesk call tickets, create Maintenance Request (MR) and assign it to the respective maintainers
- Assist Maintainers to check for the earlier reported MRs, Domain business rules and the domain experts, as well as monitoring the assigned MRs.
- Store the domain and SM knowledge created during maintenance process onto a repository.

## 4. Multi-Agent System

The agent-oriented approach is gaining acceptability in supporting maintainers in automating their daily activities (Dam and Winikoff, 2003; Viscaino et al., 2003; Rodriquez et al., 2004b). Intelligent software agent is a computer system capable of flexible autonomous action in some environments. Being flexible means that the agent is reactive (maintains an ongoing interaction with its environment, and responds to changes), proactive (taking initiatives) and social (interact with other agents) (Wooldridge,2002). Hence, a MAS is a system consisting of a number of agents, which interact with each others.

We propose a MAS tool to enable both users and software maintainers to automate some of the SM activities and capture and share the enterprise business domain knowledge and automatically link them to the application and SM process information.

Prometheus methodology was used to design the MAS and based on analyses of goals, data cohesion and agent interaction, the proposed MAS System Overview and architecture are depicted below in Fig. 5 and Fig. 6, respectively (Mohd Nor et al., 2008c).

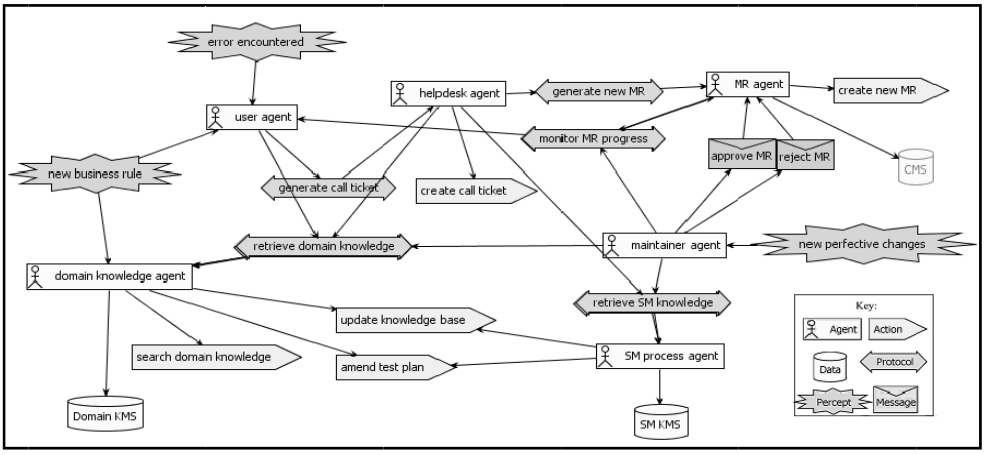


Fig. 5. MAS Systems Overview

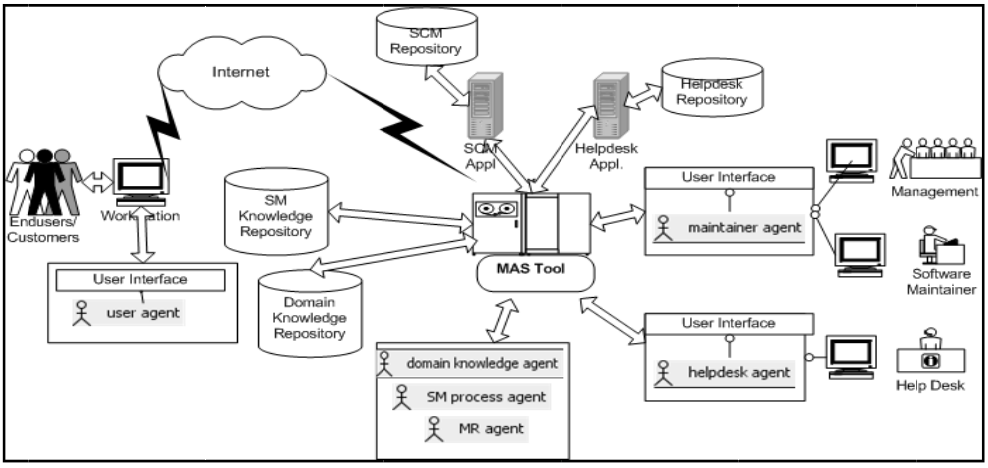


Fig. 6. Multi-Agent Architecture

As a result, six agent types are proposed: User Agent, Helpdesk Agent, Maintenance Request Agent, Maintainer Agent, SM Process Agent and Domain Knowledge Agent, as described below:

- User Agent – represents each user to file complaints, check best practices and application guides, as well as receive MR status from other agents.
- Domain Knowledge Agent – manages the domain knowledge ontology and data. When new knowledge is added, this agent shall inform relevant users and maintainers. When a new version is applied, the domain knowledge is updated with new best practices and old practices are deprecated.
- Helpdesk Agent – represents the role of helpdesk staff, by generating new call tickets based on error reported by User Agent, and assigning them to available

helpdesk personnel. If bugs is valid, Helpdesk Agent shall liaise with MR Agent to create new MR.

- MR Agent – Other than creating new MRs, this agent shall also assist planner to approve/reject MRs, monitor the progress of MRs and assign MRs to maintainers, via Maintainer Agent.
- Maintainer Agent – represents maintainers (analysts, programmers and testers) to monitor MR statuses and assign to maintainer groups for development. This agent also liaise with Domain Knowledge Agent and SM Knowledge Agent to obtain knowledge to assist analysts and tester in their works..
- SM Process Agent – For new artifacts and object changed, SM Knowledge Agent shall update the SM knowledge base, as well as the Domain knowledge. This agent also monitors the releases and versions, and provides maintainers with the information requested.

### 4.1 Combined Enterprise Domain and SM Ontology

The term ontology, in our context, can be best defined as a formal explicit description of concepts or entities, and their properties, relationships and constraints [Gruninger & Fox, 1995; Noy & McGuinness, 2001). The uses of ontology to support the agent-based tool, development of ontology is critical in the following ways:

- Agents use ontology to share common terms and to communicate to other agents (Wooldridge, 2002).
- Agent must understand the environment in which they operate. When agent retrieves or store the knowledge, it needs to know how the knowledge is structured. These semantics form the ontology of the knowledge (Yusko, 2005).

Critical to the previously identified MAS agents are the ontologies for domain and SM process knowledge. The above agents shall use ontology to make sense of the complex relations between reported errors, MRs, versions and releases, known issues, domain knowledge and users, experts and maintainers profiles.

Henceforth, we outline a combined ontology which links and extends the enterprise business domain to SM process ontology and model the combined ontology using Protégé ontology editor. For SM process ontology, the Ruiz ontology (Ruiz et al., 2004), which was based on Kitchenham et al. (1999) SM ontology, shall be used as the basis, due to similarity of the concepts in author's SM environment. For Domain business ontology, the hierarchical domain ontology proposed by Kabilan (2007), and business process metadata from Ulrich (2002) shall be used as the basis for our enterprise business domain ontology. In summary, the following sub-ontologies are proposed (Mohd Nor et al., 2008d):

- Product subontology – defines the software products that are maintained. These include the various artifacts (components, modules, versions, documents, etc.)
- Process subontology – includes the explicit processes used to carry out different activities. These processes defines the methods for problem reporting, problem identification and various maintenance activities
- Activity subontology – defines the various activities being performed by maintainers, such as support, managerial, maintenance and testing.
- Organization subontology – specifies the organizational units, the roles and the personnel involved.

- Enterprise business domain subontology – which includes:
  - Domain process type - top most layer which includes the generic high-level functionality.
  - Complex process and basic process – describes hierarchical business processes. A complex process may have several basic processes.
  - Process use – how process uses the application.

The redefined schema for the Activity and Product subontologies are drawn using OWL-Vis in Protégé ontology editor and are illustrated in Fig. 7 and Fig. 8, respectively. The linkages between the above SM sub-ontologies and Enterprise business domain sub-ontologies are depicted in Fig. 9.

Compared to other related SM ontologies, The strength of this ontology lies with the much needed details on the links between Domain sub-ontology and the SM sub-ontologies. With this linkage, changes to either Enterprise Domain knowledge or SM artifacts could be traversed and specific actions could be triggered. Also, the agents could relate the current reported errors with the previously reported errors via these ontological links.

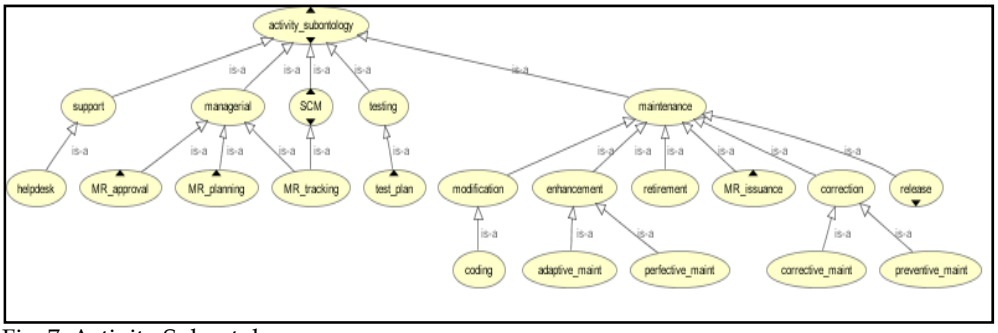


Fig. 7. Activity Subontology

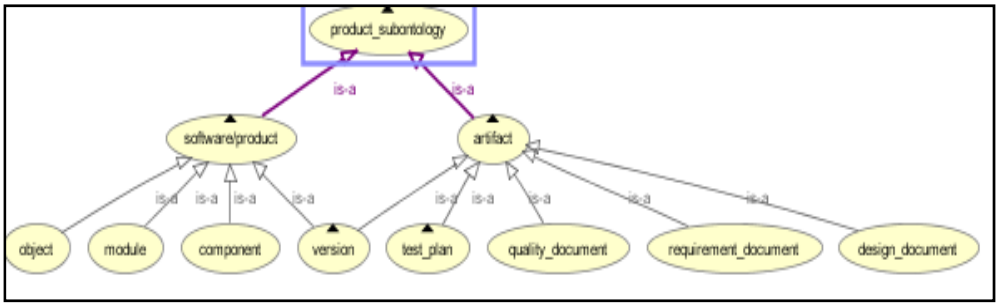


Fig. 8. Product Subontology

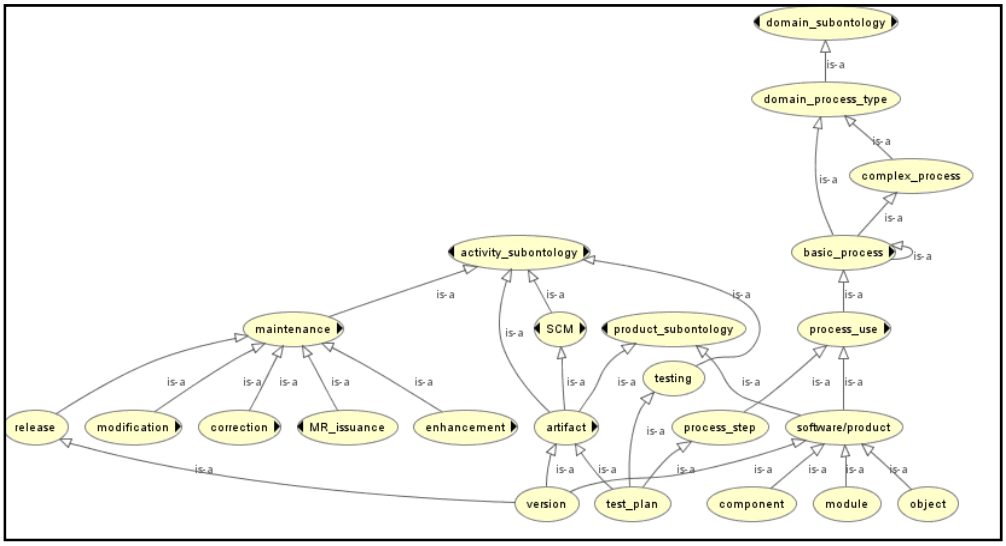


Fig. 9. Relations between Enterprise Business Domain Subontology and SM Ontology

## 5. Conclusion

In recent years, many organizations consider knowledge management (KM) to be strategically important to their business. In general, Knowledge Sharing (KS), Knowledge Transfer (KT) and Knowledge Management System (KMS) are among the themes to bring synergies among different teams, units or departments, to accelerate innovation, improve quality and reduce costs and exposure to business risks. In Software Engineering area, KM have been studied mostly on Software Development environment, but Software Maintenance (SM) environment are often neglected. SM environment is complex, knowledge-driven and highly collaborative and therefore, KM is critical to SM to provides an environment for creating and sharing knowledge.

One of the major challenges faced by software maintainers is inadequate knowledge to perform daily activities. Maintainers spent considerable efforts checking codes and collaborating with other parties to obtain information. In a survey in selected I.T. departments in higher learning institutions in Malaysia, inadequate enterprise business domain knowledge are deemed important but are seldom stored in KMS or other electronic means. Therefore, resolving these issues should be given high priority.

To overcome the problems associated with lack of knowledge in SM environment, we propose a MAS tool to enable both users and software maintainers to capture and share the enterprise business domain knowledge and automatically link them to the application and SM process information. Prometheus methodology is used to design the MAS and as a result, six agent types are proposed: User Agent, Helpdesk Agent, Maintenance Request Agent, Maintainer Agent, SM Process Agent and Domain Knowledge Agent. Critical to the systematic information organization is the ontology for domain and SM process knowledge, to allow software agents to communicate among each others, and to understand the information structure when retrieving or storing the knowledge. Henceforth, we outline a

combined ontology which links and extends the enterprise business domain to SM process ontology and model the combined ontology using Protégé ontology editor.
With this tool, users and maintainers shall benefits from systematic organization of domain and SM process knowledge, as well as ensuring that changes to either application or domain business knowledge are corroborated and shared among the business users and maintainers. The new tool shall also promote automation and integration of systems or tools to support maintenance processes

## 6. References

Abdullah, R. (2008). Knowledge Management System in a Collaborative Environment. University Putra Malaysia Press.

Abdullah, R., Sahibuddin, S., Alias, R., & Selamat, M. H. (2006). Knowledge Management System Architecture For Organizational

Alavi, M., & Leidner, D. (2000). Knowledge Management Systems: Issues, Challenges, and Benefits. Communication of AIS, 1.

April, A. (2005). Software Maintenance Maturity Model (SMmm): The Software Maintenance Process Model. Journal of Software Maintenance and Evolution: Research and Practice, 17(3).

Arthur Anderson, & APQC. (1996). The Knowledge Management Assessment Tool: External Benchmarking Version. Arthur Anderson/APQC.

Aurum, A., Jeffery, R., Wohlin, C., & Handzic, M. (2003). Managing Software Engineering Knowledge. Springer.

Choo, C. (1996). An Integrated Information Model of the Organization: The Knowing Organization.

Dam, K. H., & Winikoff, M. (2003). Comparing AgentOriented Methodologies. In 5 th International Workshop on Agent-Oriented Information Systems (AOIS'03).

Das, S., Lutters, W., & Seaman, C. (2007). Understanding Documentation Value in Software Maintenance. In Proceedings of the 2007 Symposium on Computer human interaction for the management of information technology.

Davenport, T., & Prusak, L. (2000). Working Knowledge: How Organization Manage What They Know. Harvard Business School Press.

Deraman, A. (1998). A Framework For Software Maintenance Model Development. Malaysian Journal of Computer Science, 11(2).

Desouza, K. C., Awazu, Y., & Baloh, P. (2006). Managing Knowledge in Global Software Development Efforts: Issues and Practices. Software, IEEE, 23(5), 30–37.

Dingsoyr, T., & Conradi, R. (2002). A Survey Of Case Studies Of The Use Of Knowledge Management In Software Engineering. International Journal of Software Engineering and Knowledge Engineering, 12(4).

Fjeldstad, R., & Hamlen, W. (1998). Application Program Maintenance Study: Report to Our Respondents. In Software Engineering- Theory and Practices. Prentice Hall.

Foo, S., Chua, A., & Sharma, R. (2006). Knowledge management Tools and Techniques. Singapore: Pearson Prentice Hll.

Ghali, N. (1993). Managing Software Development Knowledge: A Conceptually-Oriented Software Engineering Environment, PhD. Thesis. University of Ottawa, Canada.

Handzic, M. and Hasan, H. (2003), "The Search for an Integrated KM Framework", chapter 1 in Hasan H. and Handzic M. (eds.), Australian Studies. in Knowledge Management, UOW Press, Wollongong

Australia KM Standard (2001), Framework. In Australian Studies in Knowledge Management. UOW Press. Retrieved January 14, 2009, from http://www.worldscibooks.com/.

Hahn, J., & Subramani, M. (n.d.). A Framework Of Knowledge Management Systems: Issues And Challenges For Theory And Practice.

Hansen, M., Nohria, N., & Tierney, T. (1999). What's Your Strategy For Managing Knowledge? Harvard Business Review.

Hosapple, C., & Joshi, K. (2002). Understanding KM Solutions: The Evolution of Frameworks in Theory and Practice. In Knowledge Management Systems: Theory and Practice. Thomson Learning.

IEEE 14764 (2006). IEEE 14764-2006, Standard for Software Engineering - Software Life Cycle Process – Maintenance. The Institute of Electrical and Electronics Engineers, Inc.

Jarke, M., & Rose, T. (1988). Managing Knowledge about Information System Evolution. In Proceedings of the 1988 ACM SIGMOD International Conference.

Kabilan, V. (2007). Ontology for InformationSystems (O4IS) Design Methodology: Conceptualizing, Designing and Representing Domain Ontologies. PhD Thesis, The Royal Institute of Technology, Sweden.

Kitchenham, B., Travassos, G., Mayrhauser, A., Niessink, F., Schneidewind, N., Singer, J., et al. (1999). Towards an ontology of software maintenance. Journal of Software Maintenance: Research and Practice, 11(6).

KPMG. (n.d.). KPMG European KM Survey 2003. Retrieved from www.knowledgeboard.com.

KPMG. (n.d.). KPMG Knowledge Management Research Report 2000. Retrieved from www.knowledgeboard.com.

Lientz, B., & Swanson, E. (1981). Characteristics of Application Software Maintenance. Communications of the ACM, 24(11).

Meso, P., & Smith, R. (2000). A Resource-Based View Of Organizational Knowledge Management Systems. Journal of Knowledge Management, 4(3).

Mohd Nor, M. Z., & Abdullah, R. (2008a). A Technical Perspective of Knowledge Management in Collaborative Software Maintenance Environment. In Knowledge Management International Conference (KMICE).

Mohd Nor, M. Z., & Abdullah, R. (2008b). Survey on Software Maintenance Profile and Knowledge Requirement in Public Higher Learning Institutions. In Accepted for 3rd International Symposium on Information Technology Conference (ITSIM) .

Mohd Nor, M. Z., Abdullah, R., Selamat, M. H., & Ghazali, M. (2008c). Agent-based Tool To Support Collaborative KMS In Software Maintenance Process Environment. Proceeding of International Conference on Industrial Engineering and Engineering Management (IEEM) 2008.

Mohd Nor, M. Z., Abdullah, R., Selamat, M. H., & Ghazali, M. (2008d). Combining Business Domain and Software Maintenance Process Ontology, Proceeding of Malaysian Software Engineering Conference (MySEC) 2008.

Natarajan, G., & Shekar, S. (2001). Knowledge Management: Enable Business Growth. McGraw-Hill.

Nonaka, I., & Takeuchi, H. (1995). The Knowledge-Creating Company. New York: Oxford University Press, Inc.

Pigoski, T. (1997). Practical Software Maintenance:Best Practices for Managing your Software Investment. John Wiley & Sons.

Pressman, R. (2005). Software Engineering: A Practical Approach (6th ed.). McGraw Hill.

Rodriquez, O., Martinez, A., Favela, J., Viscaino, A., & Piattini, M. (2004a). Understanding and Supporting Knowledge Flows in a Community of Software Developers. Lecture Notes in Computer Science, 2198.

Rodriquez, O., Martinez, A., Favela, J., Viscaino, A., & Piattini, M. (2004b). How to Manage Knowledge in the Software Maintenance Process. Lecture Notes in Computer Science, 3096.

Ruiz, F., Viscaino, A., Piattini, M., & Garcia, F. (2004). An Ontology for The Management of Software Maintenance Projects. International Journal of Software Engineering and Knowledge Engineering.

Rus, I., & Lindvall, M. (2001). Knowledge Management in Software Engineering. IEEE Software, 19(3).

Santos, G., Vilela, K., Montoni, M., & Rocha, A. (2005). Knowledge Management in a Software Development Environment to Suport Software Process Deployment. Lecture Notes in Computer Science, 3782.

Schach, S., Jin, B., Yu, L., Heller, G., & Offutt, J. (2003). Determining the Distribution of Maintenance Categories: Survey versus Measurement. Journal of Empirical Software Engineering, 8.

Selamat, M., Abdullah, R., & Paul, C. (2006). Knowledge Management System Architecture For Organizational Learning With Collaborative Environment. International Journal of Computer Science and Network Security, 6(8a).

SWEBOK. (2004). Software Maintenance. In Guide to the Software Engineering Body of Knowledge (SWEBOK). The Institute of Electrical and Electronics Engineers, Inc.

Szulanski, G. (1996). Exploring Internal Stickiness: Impediments to the Transfer of Best Practice Within The Firm. Strategic Management Journal, 17.

Ulrich, F. (2002). A Multi-Layer Architecture for Knowledge Management Systems. In Knowledge Management Systems: Theory and Practice (pp. 97-111). Thomson Learning.

Van Der Speck, R., & Spijkervert, A. (1997). Knowledge Management: Dealing Intelligently with Knowledge. In Knowledge Management and its Integrative Elements. CRC Press.

Viscaino, A., Soto, J., & Piattini, M. (2003). Supporting Software Maintenance in Web Repository through a Multi-Agent System. Lecture Notes in Computer Science, 2663.

Vizcaíno, A., Soto, J., & Piattini, M. (2004). Supporting Knowledge Reuse During the Software Maintenance Process through Agents. Proceedings of the 6th International Conference on Enterprise Information Systems (ICEIS).

Wiig, K. (1993). Knowledge Management Foundation. Schema Press.

Wilson, T. (2002). The nonsense of knowledge management. Journal of Information Research, 8(1).

Wooldridge, M. (2002). An Introduction to Multiagent Systems. John Wiley and Sons.
Yusko, J. (2005). The Knowledge Collective: A Multi-Layer, Multi-Agent Framework for Information Management in an Intelligent Knowledge Base. PhD. Thesis, Illinois Institute of Technology.