An Abstract Methodology for Developing Knowledge Management Systems

Razieh Dehghani and Raman Ramsin
Department of Computer Engineering
Sharif University of Technology
Tehran, Iran
rdehghani@ce.sharif.edu, ramsin@sharif.edu

Abstract—Powerful organizations are those that manage their power factors efficiently; organizational resources are considered vital power factors, and Knowledge is one of the most important resources to manage. There is no universally accepted Knowledge Management (KM) process, but it is known that establishing the appropriate knowledge flows in the organization is the main goal of organizational KM. A Knowledge Management System (KMS) is an information system which supports the KM process, mainly by providing the required knowledge and enhancing its flow. Organizations increasingly feel the need for appropriate methodologies for developing their target KMSs. However, existing KMS development methodologies are not comprehensive enough to satisfy all organizational needs. In this paper, we propose an abstract KMS development methodology which alleviates the weaknesses of existing methodologies while reusing their strengths. Method engineers can develop concrete methodologies by instantiating the proposed abstract methodology and adding the necessary detail, thus producing bespoke methodologies which are best suited to organizational needs.

Keywords—Knowledge Management; Knowledge Management System Development Methodology; Method Engineering; Abstract Methodology; Goal-Question-Metric (GQM) Method.

I. Introduction

Nowadays, continuous improvement of organizational capabilities is considered as one of the most important success factors in organizations, and improvement of organizational capabilities requires detailed and practical planning. On the other hand, fulfilling organizational plans requires an adequate supply of organizational resources. Due to the constraints typically imposed on organizational resources, managers prefer to maximize the usage of their available resources instead of employing new ones. To this aim, two major steps should be performed: Identification of organizational resources, and planning for their preservation and promotion.

Identification and promotion of organizational *knowledge* resources is specifically important yet complicated, mainly due to their intangibility, diversity, and complexity. Due to the generality of “knowledge” as a concept, different definitions have so far been proposed for it, but it is generally accepted that *knowledge* is more intangible and valuable than *information*. A Knowledge Management (KM) process supports the identification, preservation, and promotion of organizational knowledge resources, and a system which supports the KM process is called a KM System (KMS) [1]. Obtaining, preserving, and managing different types of organizational resources (Human, Organizational, and Customer Capital [2]) requires possessing and managing the relevant knowledge that the organization has obtained over the years; this pivotal role of knowledge has further increased the importance of KM processes. Although various KM processes have so far been proposed, the following steps should generally be included in any KM process: Identifying knowledge resources, and collecting, capturing, organizing, storing, sharing, using, and creating knowledge [3].

Due to their effective role in supporting KM processes, KMSs are increasingly gaining in popularity, and using capable methodologies for KMS development has become essential. A KMS development methodology can be seen as an engineering approach which directs managers on how to map their KM issues into a practicable solution (that is, a KMS). Such a methodology consists of two parts: A Process, and a Modeling Language [4]. The process part provides a multi-dimensional procedure for developing the KMS; i.e., it determines which activities should be performed and in what order, who is responsible for performing each activity, what artifacts are used/produced by each activity, and how the activities should be performed and monitored. On the other hand, the modeling language specifies the syntax and semantic rules which should be observed in expressing the artifacts of each activity.

Unfortunately, existing KMS development methodologies do not address all organizational needs. We have previously conducted a criteria-based evaluation of a number of prominent KMS development methodologies, the results of which have been analyzed to reveal the strengths and weaknesses of existing methodologies. In this paper, we propose an abstract high-level methodology for developing KMSs which addresses the weaknesses of existing methodologies while using their strengths, and which can be configured to produce bespoke methodologies. This abstract methodology has been developed in three major stages: 1) Developing an abstract KMS development *framework*; 2) Producing the target abstract *methodology*; and 3) Evaluating the proposed abstract methodology. In the first stage, a methodology framework has been produced by abstracting prominent KMS development methodologies. The second stage consists of the following steps: 1) Instantiating the abstract framework; 2) Adding the necessary detail based on the strengths and weaknesses of

978-1-4799-7211-1/14/$31.00 ©2014 IEEE

existing methodologies (so that weaknesses are addressed and strengths are built upon); and 3) categorizing the evaluation criteria employed in our previous research, and based on these categories, adding the features/activities/stages required for developing a successful KMS. In the third stage, the resulting methodology is evaluated based on the criteria proposed in our previous research, and also through a case study. The methodology thus produced is still abstract, and it can be configured based on the specific requirements of organizations to produce tailored-to-fit concrete methodologies.

The rest of this paper is structured as follows: A KMS development framework is proposed in Section II; our proposed abstract methodology is explained in Section III; Section IV provides an evaluation of the abstract methodology; and Section V presents the conclusions and suggests ways for furthering this research.

II. Proposed KMS Development Framework

In this section, a brief review of our previous research on evaluation of KMS development methodologies will be provided. Our proposed high-level framework for developing KMSs is then described.

A. Evaluation of Existing KMS Development Methodologies

Our previous research (to appear in a separate paper) focused on evaluating existing KMS development methodologies. For the purpose of this evaluation, only the most prominent, detailed, comprehensive, and innovative methodologies were targeted. The methodologies, which were evaluated based on a comprehensive set of evaluation criteria, are as follows: Rubenstein-Montano et al. [5], Smuts et al. [6], Sarnikar & Deokar [7], Amine & Ahmed-Nacer [8], Moteleb et al. [9], Chalmeta & Grangel [10], and Iglesias & Garijo [11].

B. Proposed KMS Development Framework

The proposed framework, shown in Fig. 1, consists of three phases: Initiation, Development, and Termination. This framework was produced through applying abstraction to the methodologies evaluated in our previous research (listed above), and identifying the architecture common to them all.

In the Initiation phase, the following activities are performed: Analysis of problem-domain characteristics, planning for the KMS development phases and the organizational KM process, analysis of KMS features (solution-domain), identification and elicitation of organizational knowledge, and identification of security levels for organizational knowledge. The Development phase is concerned with: Designing the KMS, determining appropriate technologies, implementing the system, and distribution of organizational knowledge. The Termination phase involves the following: Transition to user environment, system/process testing, training, maintenance, and evaluation of KM process and knowledge content. Furthermore, umbrella activities (shown within the arrow symbol in Fig. 1) are implicitly supported by the phases of the proposed framework.

III. Proposed Abstract KMS Development Methodology

In this section, our proposed abstract methodology will be described, and the logic behind its constitution will be explained. To this aim, we will first discuss the bases for the derivation of our proposed methodology, and will then describe the abstract methodology itself.

A. Bases for Derivation of Proposed Methodology

Our proposed methodology is built upon three bases, all of which are rooted in the results of our previous research: 1) The proposed KMS development framework (Fig. 1); 2) weaknesses and strengths identified in prominent KMS development methodologies; and 3) features/activities/stages deemed necessary for satisfying the requirements of an efficient KMS development methodology. The target abstract methodology was produced by utilizing the strengths, addressing the weaknesses, and adding the required features/activities/stages (discussed throughout the rest of this section) to the proposed KMS development framework (introduced in the previous section).

1) Strengths and Weaknesses of Prominent KMS Development Methodologies

As mentioned above, we have evaluated a number of prominent KMS development methodologies in our previous research, the results of which point to the following strengths among the methodologies: 1) attention to the discovery of knowledge sources, 2) provision of methods for accessing organizational knowledge sources, 3) periodical assessment of knowledge content, 4) attention to the discovery of organizational knowledge flows, 5) attention to the discovery of organizational knowledge taxonomy, and 6) special attention to prioritization of requirements. Also, the following weaknesses were identified: 1) lack of planning for the organizational KM process, 2) lack of attention to organizational policies and standards, 3) failure to determine managerial responsibilities and their assignment to the right individuals, 4) failure to address training, motivation, preservation and promotion of human resources, and the communicational strategies required, 5) lack of support for methodology-level training and learning, 6) poor support for documentation, 7) failure to support the basic requirements of KMSs, 8) lack of support for continuous and criteria-based evaluation of the satisfaction of system requirements, 9) failure to consider organizational structure, 10) failure to determine the degree of supervision required on user activities, 11) lack of periodical notifications, 12) failure to properly manage the financial resources, 13) failure to address oppositions/obstacles in KMS development, 14) failure to attract managerial support, 15) lack of attention to user requirements at different organizational levels, 16) failure to specify appropriate technologies and tools, 17) failure to provide an adequate vision on KMS capabilities, 18) failure to determine the responsibilities and authorities of users at various levels, 19) lack of attention to the various knowledge security levels required, 20) lack of attention to distinguishing tacit KM from explicit KM, 21) absence of periodical validation, 22) failure to gather knowledge based on knowledge requirements, and 23) lack of attention to long-term goals.

978-1-4799-7211-1/14/$31.00 ©2014 IEEE

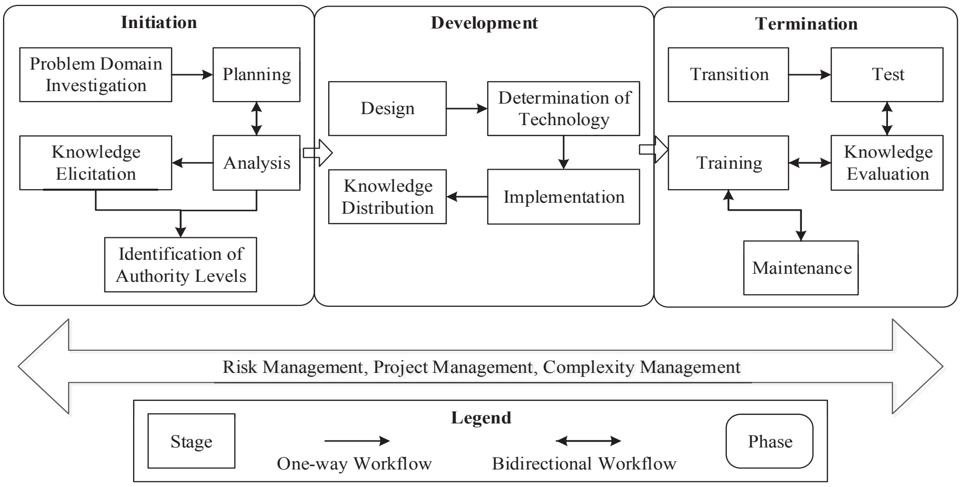

Fig. 1. Proposed KMS Development Framework

2) Elicitation of Features/Activities/Stages

This section explains how a set of overall features, fine-grained activities, and coarse-grained stages have been elicited for the target KMS development methodology. To this aim, we have used the evaluation criteria used in our previous research, which an efficient KMS development methodology is expected to satisfy. We have classified the evaluation criteria using the well-known Goal-Question-Metric (GQM) method [12]. For this purpose, the goals of an efficient KMS development methodology have been extracted based on the goals of an efficient KMS; the evaluation criteria are then classified based on the questions that may be asked in order to assess the satisfaction of goals. We have elicited the required features/activities/stages for fulfilling the goals of KMS development methodologies at the highest level. As an example, one set of elicited features/activities/stages has been shown in Table I; the complete collection is quite large and cannot be presented herein due to space limitations.

By using this method for eliciting the features/stages/activities of our proposed abstract methodology, we can claim that the proposed methodology is superior to existing methodologies from the following three aspects: 1) Comprehensiveness: Since we have used two sets of evaluation criteria (one for features common to all types of methodologies, and the other for features specific to KMS development methodologies), we can claim that our proposed methodology possesses both the general and specific features of an efficient KMS development methodology; 2) CMM Support: As we have elicited certain criteria based on features at different levels of the Capability Maturity Model (CMM), our proposed methodology helps managers to gradually improve their organizational CMM level; and 3) Practicability: Since we have elicited certain criteria based on the success/failure factors of KMS development projects, our proposed methodology is more likely to be practicable.

B. *Proposed Abstract KMS Development Methodology*

The process part of our proposed abstract methodology consists of eight phases (as shown in Fig. 2): Initiation, Analysis, Design, Implementation, Test, Deployment, Maintenance, and Death. The prescribed phases and their corresponding input and output products are explained in the following subsections; due to the generality of the process, the activities nested in each stage are just listed without prescribing any specific techniques for performing them. As shown in Fig. 2, four phases of the process (Analysis, Design, Implementation, and Test) are performed iteratively.

The activities that are shown on the left side of Fig. 2 (within the arrow) refer to umbrella activities. These activities are implicitly supported by the whole process; this means that all of the prescribed stages collaborate for fulfilling these umbrella activities. Due to the dynamicity of knowledge, the *Analysis of the Current Situation* and *Knowledge Acquisition* stages should be performed in parallel with the stages within the six phases of the prescribed process; this helps to continually update the available and required knowledge. It should be noted that these two stages are the main stages of the Analysis phase, and their outputs are detailed in this phase. The roles involved in the development teams are:

- Preliminary Analysis Team: Representative of the Communities of Practice–CoPs (including knowledge producers, system users, and decision makers), Customer (including the Organizational Manager and the Project Manager), Methodology Specialist, and Configuration Manager; it should be noted that CoPs are groups of specialists where members communicate with each other based on their common interests [13].

TABLE I. EXAMPLE OF FEATURES/STAGES/ACTIVITIES (GQM-BASED)

Goal: **Adaptivity to diverse/uncertain requirements**	
Questions: - Can the process be configured based on organizational requirements? - Is the methodology flexible during enactment?	
Criterion	***Elicited Features/Stages/Activities*** [a]
Configurability	*: Defining the methodology at a high level; +: Initial decision making; -: Configure the methodology; -: Update the methodology.
Flexibility	

[a] Legend: */+/- : Feature/Stage/Activity that should be supported

- Analysis Team: Requirements Engineer, Organizational Expert [14], Planner, Representative of CoPs, Customer, Knowledge Manager (the individual who integrates, finalizes, and updates the strategies and goals for developing KMSs) [14], System Analyzer, and Knowledge Engineer (the individual who analyzes the organizational knowledge sources/flows and monitors the KM process) [14].
- Design Team: Representative of CoPs, Customer, System Designer, and Knowledge Engineer.
- Implementation, Deployment, and Test Team: Representative of CoPs, Planner, Customer, Tester, Programmer, Technical Coordinator (the individual responsible for developing the system architecture and also for providing appropriate technologies and checking their consistency) [15], Knowledge Engineer, Tester, Technical Writer (the individual who should prepare the documents together with the related specialists) [15], and Configuration Manager.
- Maintenance and Death Team: Requirements Engineer, Methodology Specialist, Knowledge Engineer, Planner, and Technical Writer.

1) Initiation Phase

This phase is targeted at preliminary planning along with making the initial decisions. As shown in Fig. 2, this phase consists of the following serial stages, each of which includes a set of nested activities:

- Preliminary Analysis: Determine high-level goals for developing the KMS, Investigate the organizational business processes, Determine the strategies, Determine the policies, and Study the feasibility of developing the KMS.
- Initial Decision Making: Configure the methodology, Determine a timeframe for the iterations, Plan for managing the organizational KM process, Gain the managers' support, Determine and Assign managerial responsibilities, Determine authorization levels for different users, and Form the development teams and CoPs in order to facilitate knowledge sharing.

a) Input: Available Organizational Documents, Methodology Definition Document.

b) Output: Feasibility Study Document, Planning Document.

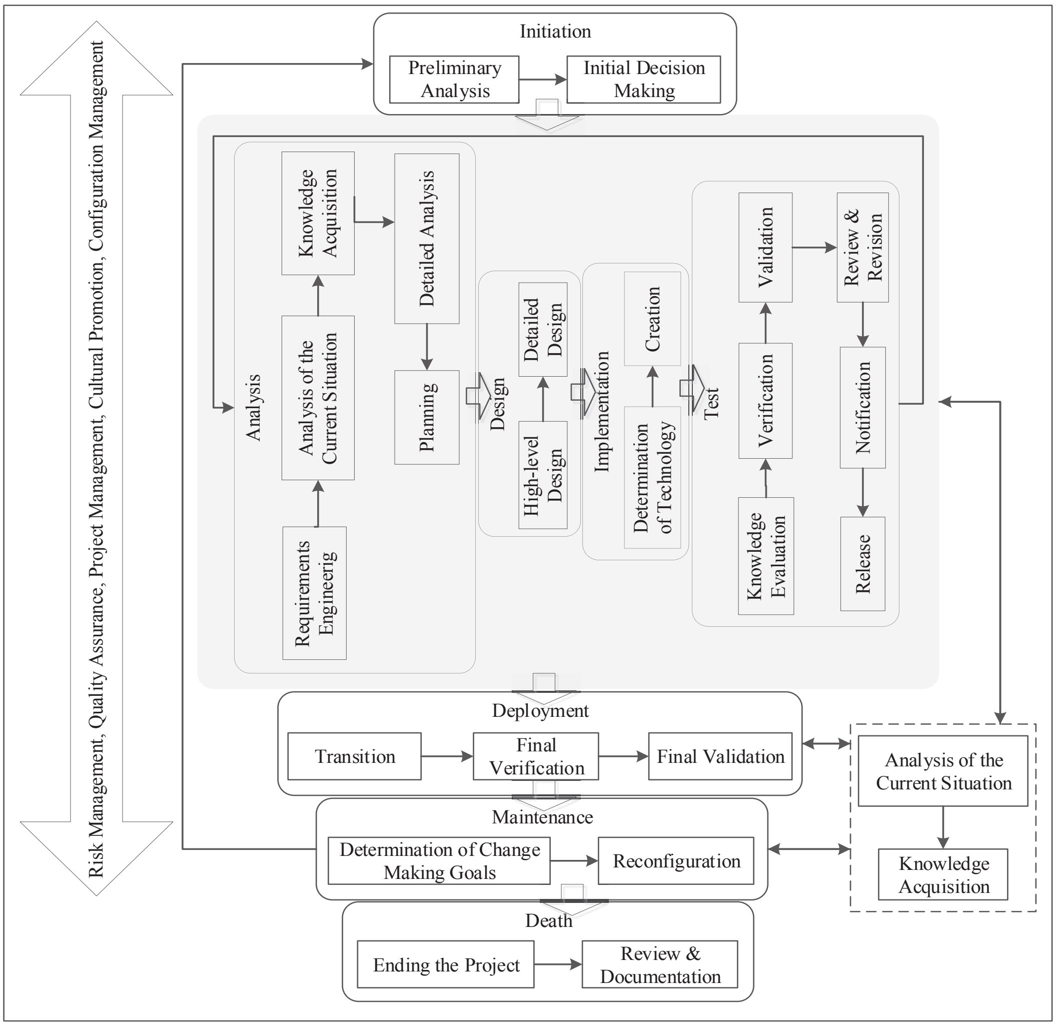

Fig. 2. Proposed Abstract Methodology for KMS Development

2) *Analysis Phase*

The aim of this phase is to gain detailed understanding of the problem- and solution domains. The stages are as follows:

- Requirements Engineering: Elicit, Analyze, and Validate the requirements.
- Analysis of Current Situation: Analyze the needed and available knowledge, and Investigate existing systems and their changes, methods to access knowledge sources, and current cultural status of the organization.
- Knowledge Acquisition: Elicit knowledge from the identified knowledge sources/flows, and Formalize and Label the elicited knowledge.
- Detailed Analysis: Analyze the structure, functionality, and behavior of the KMS.
- Planning: Plan and Schedule the system development effort, and Determine criteria for evaluating the success of the KM process and satisfaction of the requirements.

a) Input: Planning Document.

b) Output: Behavioral/Structural/Functional Analysis Models, Goal Refinement Tree (where goals are refined into sub-goals/requirements) [16], Requirements Specifications, Planning Document, Knowledge Acquisition Document.

3) *Design Phase*

The system is designed through the following two stages:

- High-level Design: Choose/Design an appropriate architecture for the KMS.
- Detailed Design: Design the knowledge map, system, test procedures (and cases), and the system interface.

a) Input: Structural/Behavioral/Functional Analysis Models, Planning Document, Knowledge Acquisition Document, Knowledge Map.

b) Output: Behavioral/Structural/Functional Design Models, Available and Required Knowledge Map, Planning Document, UI Modeling Diagram (optional).

4) *Implementation Phase*

The system is iteratively implemented in this phase. For this purpose, the system prototype will be continuously completed. Two stages are prescribed for this purpose:

- Determination of Appropriate Technologies: Choose appropriate technologies for implementing the system.
- Creation: Implement the available and also the needed knowledge map, and Code the designed system.

a) Input: Output artifacts of the previous phase.

b) Output: Technology Investigation Document, System Prototype (encompassing the knowledge map).

5) *Test Phase*

This phase is conducted for evaluating and improving the items that affect KMS efficiency. Furthermore, the new version of the KMS should be released in this phase. The following stages are prescribed for this purpose:

- Knowledge Evaluation: Evaluate the accuracy and validity of elicited knowledge.
- Verification: Verify the system.
- Validation: Validate the system from the users' point of view.
- Review and Revision: Update the development team members, CoPs, strategies, and the methodology, and Version the output artifacts of the iteration.
- Notification: Notify the representatives of users, developers, and managers about the successes/failures.
- Release: Release the current prototype of the system.

a) Input: Planning Document, System Prototype, Requirements Specifications, Knowledge Map, Configuration Management Document.

b) Output: Configuration Management Document, System Prototype, Planning Document, Test Document, Requirements Specifications, Knowledge Evaluation Document.

6) *Deployment Phase*

Preparation of the system for use in the user environment (the organization) is the aim of this phase. For this purpose, the system is deployed by performing the following stages:

- Transition: Prepare legacy systems, Install the system and apply initial settings, Provide essential training for users and maintenance/support personnel, and Provide the necessary documents (such as manuals).
- Final Verification: Verify the system in order to fix the problems that had not been identified up to this point.
- Final Validation: Conduct final acceptance test in order to validate the system from the users' point of view.

a) Input: Test Document, System.

b) Output: Test Document, Configuration Management Document, Training Resources, Postmortem Document.

7) *Maintenance Phase*

This phase aims to keep the system up and running by updating it as required. The following stages are prescribed:

- Determination of Update Goals: Determine the goals for updating the system.
- Reconfiguration: Reconfigure the methodology based on the update goals, and Execute the methodology in order to satisfy the newly identified goals.

a) Input: Methodology Definition Document, Requirements Specifications.

b) Output: Configuration Management Document, Requirements Specifications.

8) *Death Phase*

If the system can no longer be used or maintained (or be reengineered to become maintainable), the death phase should be performed. This phase consists of the following stages:

- Ending the Project: Declare the project as closed, and Free the allocated resources.
- Review & Documentation: Review the development process, and Prepare the lessons-learnt documents.

a) Input: All the artifacts produced in previous phases.

b) Output: Postmortem Document, Planning Document.

IV. Evaluation of Proposed Abstract Methodology

We have evaluated our proposed abstract methodology through applying it to a case study at a private Iranian insurance company; the methodology has also been assessed based on the evaluation criteria developed in our previous research. Due to space limitations, only a small subset of the evaluation results is provided in Table II (the +/- symbols denote the satisfaction/non-satisfaction of each criterion).

Since the evaluation criteria have also been applied to existing methodologies, the results provide a basis for comparative analysis. Based on the evaluation results, our proposed methodology satisfies most of the evaluation criteria, and is superior to the methodologies evaluated in our previous research; however, the following deficiencies were revealed:

- Failure to conduct context-spanning case studies: We have performed one case study in order to demonstrate the practicality of our proposed methodology; whereas ideally, this requires multiple case studies. However, since the features/stages/activities of the methodology have been elicited based on empirically justified criteria, this is not considered a major defect.
- Failure to prescribe tools and technologies: Prescribing tools/technologies is not expected from an abstract methodology; thus, this is not a major shortcoming.
- Failure to prescribe a modeling language: Ideally, a methodology should prescribe an integrated modeling language; however, reusing sets of artifacts from existing modeling languages is also acceptable; therefore, this is not considered a major weakness.

TABLE II. SAMPLE OF CRITERIA-BASED EVALUATION OF THE PROPOSED ABSTRACT METHODOLOGY (AS COMPARED TO EXISTING METHODOLOGIES)

Methodology \ **Criterion**	***Specification of Policies***	***Gaining Managerial Support***	***Monitoring the KM Process***	***Periodical Notification***	***Basis in Practical Experiences***	***Scalability***
Rubenstein-Montano et al. [5]	-	-	-	+	-	+
Smuts et al. [6]	-	+	-	+	+	+
Sarnikar & Deokar [7]	-	-	-	-	-	+
Amine & Ahmed-Nacer [8]	+	-	-	-	-	+
Moteleb et al. [9]	-	-	-	-	+	-
Chalmeta & Grangel [10]	+	-	-	-	-	+
Iglesias & Garijo [11]	-	-	-	-	-	+
Our Proposed Methodology	+	+	+	+	+	+

V. Conclusions and Future Work

Our proposed abstract methodology has been evaluated through a case study, which has demonstrated the practicability and practicality of the proposed methodology. Furthermore, the results of criteria-based evaluation have shown that the proposed methodology satisfies most of the criteria, and can be configured and used based on organizational requirements.

We aim to further this research by proposing new concrete KMS development methodologies. For this purpose, the proposed abstract methodology should be instantiated and the necessary detail (such as fine-grained techniques) should be added. We also aim to perform context-spanning case studies in order to demonstrate the practicality of our proposed methodology in different KMS development contexts.

References

[1] M. Alavi and D. E. Leidner, “Review: Knowledge management and knowledge management systems: Conceptual foundations and research issues,” MIS Quart., vol. 25, no. 1, pp. 107–136, Mar. 2001.

[2] G. Petrash, “Dow’s journey to a knowledge value management culture,” Eur. Manag. J., vol. 14, no. 4, pp. 365–373, Aug. 1996.

[3] H. Lai and T. Chu, “Knowledge management: A review of theoretical frameworks and industrial cases,” Proc. Hawaii International Conference on System Sciences, Jan. 2000, pp. 10–20.

[4] R. Ramsin and R. F. Paige, “Iterative criteria-based approach to engineering the requirements of software development methodologies,” IET Softw., vol. 4, no. 2, pp. 91–104, Apr. 2010.

[5] B. Rubenstein-Montano, J. Liebowitz, J. Buchwalter, D. McCaw, B. Newman, and K. Rebeck, “SMARTVision: A knowledge-management methodology,” J. Knowl. Manag., vol. 5, no. 4, pp. 300–310, 2001.

[6] H. Smuts, A. van der Merwe, M. Loock, and P. Kotzé, “A framework and methodology for knowledge management system implementation,” Proc. Research Conference of the South African Institute of Computer Scientists and Info. Technologists, Oct. 2009, pp. 70–79.

[7] S. Sarnikar and A. Deokar, “Knowledge management systems for knowledge-intensive processes: Design approach and an illustrative example,” Proc. Hawaii International Conference on System Sciences, Jan. 2010, pp. 1–10.

[8] M. Amine and M. Ahmed-Nacer, “An agile methodology for implementing knowledge management systems: A case study in component-based software engineering,” Softw. Eng. Appl., vol. 5, no. 4, pp. 159–170, Oct. 2011.

[9] A. A. Moteleb, M. Woodman, and P. Critten, “Towards a practical guide for developing knowledge management systems in small organizations,” Proc. Conference on Knowledge Management, Sep. 2009, pp. 559–570.

[10] R. Chalmeta and R. Grangel, “Methodology for the implementation of knowledge management systems,” J. Am. Soc. Inf. Sci. Technol., vol. 59, no. 5, pp. 742–755, Mar. 2008.

[11] C.A. Iglesias and G. Mercedes, “The Agent-Oriented Methodology MAS-CommonKADS," in Intelligent Information Technologies, V. Sugumaran, Ed. 2008, pp. 445–468.

[12] V. R. Basili, Software modeling and measurement: The Goal/Question/ Metric paradigm, Tech. Report, University of Maryland, 1992.

[13] E. Wenger, Communities of practice: Learning, meaning, and identity, Cambridge University Press, 1998.

[14] G. Schreiber, Knowledge engineering and management: The CommonKADS methodology, MIT Press, 2000.

[15] Z. S. H. Abad, M. H. Sadi, and R. Ramsin, “Towards tool support for situational engineering of agile methodologies,” Proc. Asia Pacific Software Engineering Conference, Nov. 2010, pp. 326–335.

[16] “Objectiver: KAOS,” available at: http://www.objectiver.com/index.php?id=25 (Accessed 1 Jan. 2014).