The Journal of Korean
Institute of CALS/EC
Vol. 5, No. 1, June 2000

UML의 "4+1" 뷰를 이용한 지식 관리 시스템의 개념적 모델

안성욱*, 이남용*, 류성열*

A Conceptual Model of Knowledge Management System by using "4+1" views of UML

Sung Wook Ahn, Nam Yong Lee, Sung Yul Rhew

Abstract

Over the past several decades, several architectural approaches have been applied to develop Information Systems. The software architectural model has been recognized as one of the most important methods for improving productivity. Most conceptual models are difficult to understand and interpret at various system models. As a consequence, conceptual models of many Information Systems fail to represent, exploit, and apply to various aspects, which is needed for reducing development step of the architecture. In this paper, we will explain the architectural model as the 4+1 View of UML. This model integrates the Knowledge Management System into five views: the Logical View, the Process View, the Deployment View, Implementation View, and the Use-Case View. Moreover, this paper will not only provide information on the application of the software architectural model by stakeholders, but also ultimately improve productivity.

Key Words: architectural model, conceptual model, software productivity, software architecture, UML, Knowledge Management System

---

* 숭실대학교 컴퓨터학부

1. Introduction

Among various Information Systems, the KMS (Knowledge Management System) has gained great attention from academic, government, and industrial circles over the last several years. Management professionals have recognized Knowledge Management as a powerful means of potentially improving decision-making and company productivity [Abecker, 1998], [Alavi, 1999], [Sumner, 1999], [Sumner, 1999], [Tuomi, 1998]. Among various kinds of Knowledge Management approaches [Alavi, 1999], [Edwards, 1997], [Gains, 1999], [Ginsburg, 1999], [O'Leary, 1998], [Robillard, 1999], [Vanhoenacker, 1999], the architectural model of KMS only related to this study. Knowledge Management is the formal management of knowledge for facilitating creation, access, and reuse of knowledge, typically using advanced technologies [O'Leary, 1998]. Though many people agree with the fact that knowledge is important, there is much confusion due to a misunderstanding of KMS. Several aspects of KMS appears to be indefinite and are difficult to understand.

The purpose of this study is to be able to describe the architectural model by using a "4+1" view, not to learn how to create in architecture. Also, this study is to understand and read the materials through the viewpoints of the stakeholders. This will ultimately prolong software life cycle by reusing architectural analysis strategy.

The authors assume that the readers of this paper have a working knowledge of architectural models, UML (Unified Modeling Language), and KMS analysis techniques. An introduction to these is given in [Booch, 1999], [Booch, 1999]a, [Eriksson, 1998], [Fowler, 1997], [Nesi, 1998]. It must be stated that those models have several limitations so that an understanding of them should precede it. First, these models are confined to web based KMS. Thus, it is impossible to generalize these models because organization difficulties limit them according to environment. These environments include Intranet, Internet, Extranet. The second limitation is XML (eXtensible Markup Language) based KMS. XML view must be provided in order to share with a conventional knowledge base. It provides ontologies based knowledge base [O'Leary, 1998]a, [Sumner, 1999].

In Section 2, the authors discuss the conceptual model on the basis of the results of the literature review of KMS. In Section 3, the authors present a research framework, which includes problem statements and research methodology. In Section 4, the authors present the conceptual model of KMS by using UML. In Section 5, the authors describe the useful findings of this study and discuss several implications of them. The authors believe that the results of this study will aid in reading and interpreting software architecture analysis among various aspects, and reveal areas in which further research is required.

2. Literature Review

The following five models for knowledge management were identified. These models differ not only in their focus, but also in their nature of knowledge management. These five models are presented in chronological order.

Arthur Andersen and APQC [Andersen, 1996] present a model of organizational knowledge management. The model is comprised of seven knowledge management processes: create, identify, collect, adapt, organize, apply, and share; and four organizational enablers: leadership, measurement, culture, and technology. Because this model does not detail the nature of the enablers [Holsapple, 1999], it consequently can be implemented incorrectly with KMS.

Choo [Choo, 1997] describes a model of the knowing organization. The model has three processes: sense making, knowledge creation, and decision making; These three processes are concerned with understanding the processing link between information and organization. However this model is abstract and makes it difficult to understand knowledge management and aspects of other models.

The Solicitor Processor Acceptor Resistor Trafficator Assessor [Vanhoenacker, 1999] provides some anchor points to relate knowledge management concepts with process change issues. This model does not interpret the views of different stakeholders, only presents and demonstrates the integrative character of it.

The Annotate: A Web-based Document Knowledge Management Support System [Ginsburg, 1999] presents two data stores: discussion data and session data. It does not describe users or various aspects, but explain technology points from the perspective of the developer.

Leonard-Barton [Holsapple, 1999] introduces a Knowledge Management model comprised of four core capabilities: physical systems, managerial systems, employee skills and knowledge, and values and norms; four knowledge building activities: problem solving, implementing and integrating, experimenting, and importing knowledge. This model cannot interpret Knowledge Management from multiple views, because of the detailed nature of activities.

The mentioned models and the conventional KMS are not represented. They only describe the study of Knowledge Management and Organizational Memory Process from a singular view. Stakeholders had no conceptual model in order to develop KMS. Views show different aspects of the system that is modeled. Each view is described in a number of diagrams that contain information that emphasize a particular aspect of the system [Eriksson, 1998].

Recently we have performed several case studies, conceptual model concerning Knowledge Management Systems by using UML. The UML includes a set of consistent diagrams that may be used to describe and communicate a software system's requirements, designs, and codes.

In this paper, we how effectively understand conceptual model with multiple stakeholders. We represent the "4+1" view of UML: Logical View, Process View, Deployment View, Implementation View, and Use Case View [Lee and Litecky, 1997], [Nesi, 1998], [Robillard, 1999].

3. Research Framework

KMS is described through a number of aspects; functionality, structure, scalability, delivery, software management, etc. Figure 1 represents the "4+1" views of UML to present system architecture [Booch, 1999]a, [Eriksson, 1998].

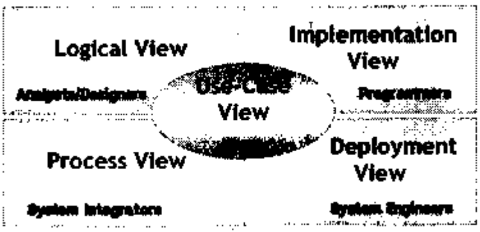

Figure 1. The "4+1"view Model of Architecture

The Use-Case View of a system describes the use cases that describe the behavior of the system as seen by its end users, analysts, and testers [Booch, 1999]a. With the UML, It is captured in use-case diagrams and occasionally in activity diagrams [Eriksson, 1998].

The Logical View of a system describes major design packages, subsystems, and classes. With the UML, it is captured in class, object, state, sequence, and activity diagrams.

The Process View of a system describes the design's concurrency and synchronization aspects. With the UML, it is captured in the same kinds of diagrams as the logical view.

The Deployment View of a system describes deployment, installation, and performance. With the UML, it is captured in deployment, interaction, and activity diagram.

The Implementation View of a system describes the source code, data files and components that are used to assemble and release the physical system. With the UML, it is captured in component, interaction, and activity diagrams.

Until now, we show the "4+1" view architecture model and briefly explain each view. And the conceptual KMS model being introduced in the following section will accompany our understandings.

4. Conceptual Model of KMS

Organizations use Knowledge Management for a number of reasons, including environmental pressure, technological advancements, and the ability to create valuable information. KMS has spread rapidly. Many companies and public organizations introduced or plan to do it. Eventually, KMS enables enterprise low cost and high efficiency.

The Knowledge Management System we suggest is an indispensable information infrastructure for knowledge management. KMS is a building tool that effectively implements and optimizes enterprise knowledge management processes. The following must be considered in the Knowledge Management System; to provide a representation and classification system for building knowledge assets in organization, to acquire and accumulate identified valuable distributed knowledge as organizational asset, and to provide knowledge management process that expand knowledge by sharing and reuse between members. The following four steps must be considered to build KMS.

The first step of building KMS is identifying knowledge: It is for establishing basic framework for obtaining and accumulating valuable knowledge. However, the valuable knowledge is distributed across humans, databases, and various external systems. Their type or structure also varies from simple bulletins to applications. Hence, the KMS this paper suggests a knowledge vessel that stores and represents various forms of knowledge, and knowledge classifying system for making easy to access to knowledge, a knowledge map.

The second step is accumulating knowledge: A process is required for obtaining knowledge and accumulating in to a database. That is for making distributed knowledge to a sharable, reusable organizational asset. For the purpose, KMS provides a way of inputting knowledge according to knowledge sources and collecting it from external systems. Moreover, it accumulates obtained knowledge-to-knowledge Database by information analysis and indexing associated with a knowledge map. This is done in order to search swiftly and correctly.

The third step is leveraging knowledge: It must provide a way to search for necessary knowledge quickly by sharing valuable information in order to enhancing work productivity. KMS provides integrated searching that is correct automatic. It reclassifies knowledge depending on a user's view. Through this step, users are able to apply valuable knowledge to decision making and learning.

The fourth step is refining knowledge: Knowledge has a dynamically changing nature due to environmental fluctuations. Hence, completeness of knowledge must be enhanced and updated in order to maintain its function as a knowledge asset. For this, KMS induces knowledge interchanges and utilizes the experiences to make current knowledge grow and evolve. Also freshness of knowledge is maintained by discarding low contributions of old knowledge. This can be used for changing organizational culture management.

Enterprise's knowledge is scattered over the core internal information systems such as e-mail, groupware, EDMS, MIS as well as employees and external information systems. Hence, integrating this knowledge and establishing organizational assets is necessary in order to build an enterprise-wide integrated KMS. KMS provides an information collector with the core information system, and integrates internal and external knowledge to provide transparent utilization environments to users.

Also, quick searching technology is essential. Quick knowledge searching is possible by XML as a data standard.

Figure 2 shows functionalities of KMS. Information can be collected through interfaces of the five core systems and external systems. These functionalities can be divided into the followings:

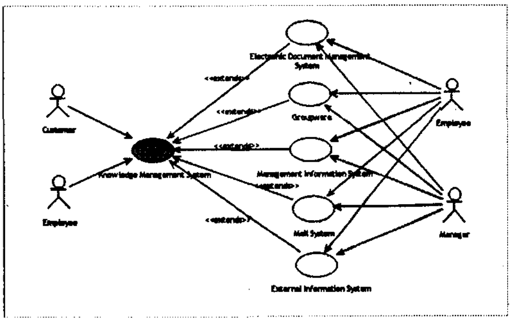

Figure 2. Use-case diagram of a general KMS

4.1. Use Case View

A use case is a typical interaction between a user and a KMS and focuses attention on the user's viewpoint about what KMS is supposed to give back in response to the user's input. A use case diagram illustrates a set of actors, use cases, the uses and extended relationships among use cases, and the relationship between actors and use cases. The purpose of the Use-case diagram is to show the functionality, understandability, and usability that the KMS provides. In this section, we present two Use-case diagrams. We show KMS by focusing on KMS's internal functionalities and association with it by five interfaces of core systems and external systems.

KMS has very grown and complicated according to globalization and dispersion. We classify the entire KMS to subsystems for comprehension and management. Subsystems may be defined and used for several purposes to organize a large IT system according to the organizational environment. In Object-Oriented projects, a subsystem can be a classical subsystem, identified by a structural composition-decomposition process. In Figure 3, the conceptual model divides KMS into five subsystems - Electronic Document Management System, Groupware, Management Information System, Mail System, and External Information System.

In Figure 4, we describe the use case diagram focus on the "Knowledge Management System" subsystem. As mentioned in the previous section, the most important function in KMS are ontologies that define knowledge base's characteristics and views. Hence we implement conceptual KMS model using XML and DTD. "*Knowledge Management System*" enables to the other systems by using core parts of conceptual KMS model. Moreover it can provide views enabling integration with other systems in organization.

As Figure 4, the "Knowledge Management System" is composed of functions: The *Make Path, Register Document, Select DTD, Attach File, Delete Document, Query Element, Read Document, Update Document, EDMS Information, Groupware Information, MIS Information, Mail Information, External Information, Management Document, Manage DTD*. For example, *Register Document* provides users with a document registering function. This use case uses a *Select DTD* use case in order to select a document form. As DTD must be selected when writing document, <<uses>> relationship made between *Register Document* and *Select DTD*. It then, sets a path for registering the document and <<extends>> the relationship made between *Register Document* and *Attach File*. It used only when defined in DTD and the Read Document must be done before creating on Update Document. An extended relationship is established. Also, the Query Element fetches data through information of the EDMS Information, Groupware Information, MIS Information, Mail Information, External Information, and interfaces. The manager oversees Manage Document and Manage DTD. It means the Knowledge Manager of information for managing intangible assets.

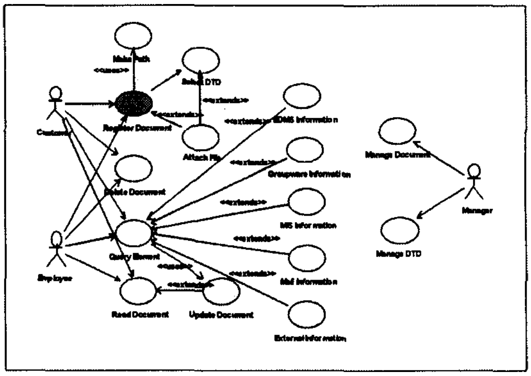

Figure 4. Use-Case Diagram of Knowledge Management System

4.2. Logical View

A class diagram describes a set of classes and their logical relationships with KMS: association, usage, composition, inheritance, and so on. This diagram shows the static structure of KMS from the analysts and designer view point. When we describe the class diagram of the "*Knowledge Management System*", we group five classes. Figure 5 shows " *Knowledge Management System* " subsystem as class diagram. There needs to be classes of *KMSDBManager*, *Document, DTD Form, List,and KMSController*. DTD Form class manages document structure, and Document is an abstract class for storing and managing common document contents. List class manages document form numbers and titles. *KMSDBManager* and XML class registers APIs of the XML Document framework, and plays a role in wrapping up related classes. Hence external workers can access the XML Document framework through the controlling class. Classes do not describe any attributes and operations that are not included. Because of an elaborate description of classes is of little importance in conceptual modeling. It is very important in the designing and implementation phase.

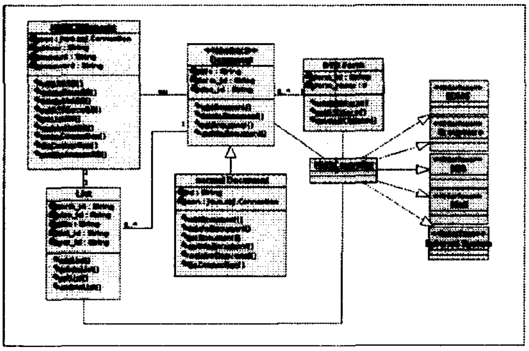

Figure 5. XML type Document Class Diagram

4.3. Process View

A sequence diagram shows how objects interact with each other. It takes into account some nonfunctional requirements, such as performance, system availability, and system fault tolerance at the system integrators level. And sequence diagrams are for expressing process flows between classes. The process flows help with clearly understanding controls between classes. It becomes the internal message flow in framework. In Figure 6, sequence diagram is shown as a process of "R*egister Document*" function in the "*Knowledge Management System*" use case. There is a menu for registering documents on the web in *Register Document*. The Register Document function includes this sequence diagram in the KMS use case. Selecting *register Document* are concurrent with the choice of DTD. The contents are filled out in XML document as selected DTD form. *Attach File* is optional function. File attaching information is stored in the database. *Select DTD* can repeatedly processed in the application level.

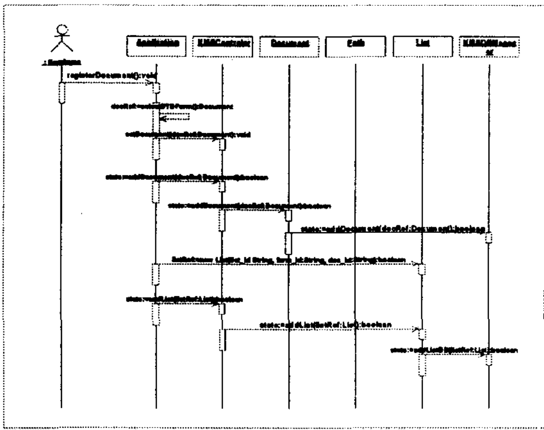

Figure 6. Sequence Diagram of "update" in XML Document Use case

4.4. Deployment View

A deployment diagram shows the physical architecture of the hardware and software in the KMS. It takes into account the system's nonfunctional requirements such as system availability, and reliability, at the system engineers view. We established a 3-tiered Client-Application-Database archit-ecture for KMS's hardware structure. The three-tier architecture provides extensibility, trust, flexibility, efficient management of the system, and it enables object based distributed computing through frequent business logic and technical changes.

In Figure 7, it includes three nodes; Client, Application Server, and the Application Server node. The KMS node connects to the XML Database, and shares information through a core internal system, Electronic Document Management System, Groupware, Management Information System, Mail System External Systems, and TCP/IP. The Web plays an important role in connecting clients and other systems. The XML Database manages integrated data related to all other systems. And the Web based KMS is very important in aspect of knowledge sharing and dissemination. Users can easily access KMS through the public nature of Internet, and interconnection among information sources is easy. Application servers including a web server enable business application developers to plug business logic components in to the application.

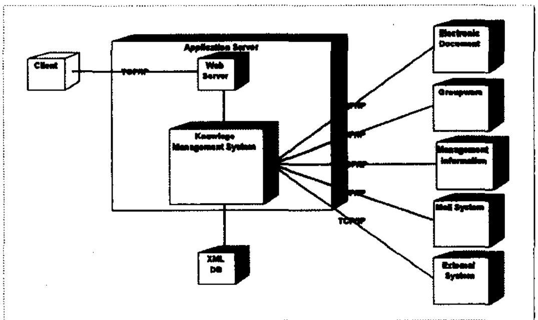

Figure 7. Deployment diagram of the physical architecture of a KMS

4.5. Implementation View

A component diagram shows software components and their dependencies with each other. It represents the structure of the code. It takes into account the reuse and software management of its organization in its implementation environment at the programmer's view.

This view is not founded in the analysis phase and only is developed during implementation. Thus it is considered beyond the scope of this analysis and design course. In other words, the conceptual model has the advantage of strongly emphasizing a focus on domain concepts, not software entities, such as component of file type. Until now, we have briefly shown several diagrams and explained each diagram, in brief.

5. Findings and Discussions

KMS aims at adapting to rapid changes of the external environment. In this paper, KMS enables sharing and managing information of external systems, as well as an inter-enterprise's Electronic Document Management System, Groupware, Management Information System, and Mail System. Two important functional features in KMS we proposed are that it is web based and XML based. This model can help designing not only KMS, but also other Information Systems. In this paper, we use the "4+1"view of UML based on object-oriented modeling technology [Booch, 1999], [Booch, 1999]a, [Eriksson, 1998], when we describe the conceptual KMS model. There are three reasons for using the "4+1"view when describing the conceptual KMS model. First, this model can facilitate designing the KMS system through related diagrams. In other words, we use this model during analysis to reduce the complexity of analysis, and to improve its consistency by providing designers with a shorthand representation for complex behavior.

However, it does not focus on functionality, but modeling methods for the object-oriented based KMS suggested in this paper. Second, it provides views of stakeholders that are different from each other according to perspectives of KMS. In other words, the "4+1"view of architecture is the best way to represent viewpoints of stakeholders. Third, The conceptual KMS model simplifies the functionality of KMS. Simplification of the system makes the KMS model easier to understand. The model is used as example when adopting KMS. Put simply, a better system is easier for understanding, implementation, and maintenance for users and developers. The conceptual KMS model can clearly interpret and develop KMS more easily. Ultimately, the KMS by the "4+1"view of UML can improve decision-making, productivity, and accomplishes a more innovative process.

참고 문헌

[Abecker, 1998] A. Abecker, A. Bernardi, and K. Hinkelmann, Otto Kuhn, and Michael Sintek German Research Center, "Toward a Technology for Organizational Memories", *IEEE Intelligent Systems*, 1998.

[Alavi, 1999] M. Alavi, D.Leidner, "Knowledge Management System: Emerging Views and Practices from the Field", *IEEE*, 1999.

[Andersen, 1996] A. Andersen and The American Productivity and Quality Ce- nter. The Knowledge Management Assessment Tool: External Benchmarking Version, Winter, 1996.

[Booch, 1999] G. Booch, J. Rumbaugh, I. Jacobson, The Unified Modeling Language User Guide, *ADDISON WESLEY*, 1999.

[Booch, 1999]a J. Rumbaugh, I. Jacobson, The Rational Unified Process an Introduction, ADDISON WESLEY, 1999.

[Choo, 1997] Choo, C. An Integrated Information Model of the Organizat- ion: The Knowing

Organization, at URL: http://choo.fis.utoronto.ca/fis/ko /#contents, 1997.

[Edwards, 1997] D.L. Edwards, D. E.Mahling, "Toward Knowledge Manageme- nt Systems in the Legal Domain", *ACM*, 1997.

[Eriksson, 1998] H.E. Eriksson, M. Penker, *UML Toolkit*, vol. , 1998.

[Fowler, 1997] M. Fowler and K.Scott, *UML Distilled*, Addison Weslety, 1997.

[Gains, 1999] B.R. Gains and M.L.G. Shaw, "Embedding Formal Knowledge Models in Active Documents", *COMMUNICATION OF THE ACM*, Vol.42, No.1, January 1999.

[Ginsburg, 1999] M. Ginsburg, A. Kambil, "A Web-based Knowledge Management Support System for Document Collections", *IEEE*, 1999.

[Holsapple, 1999] C.W. Holsapple, K.D. Joshi, "Description and Analysis of Existing Knowledge Management Frameworks", *IEEE*, 1999.

[Lee and Litecky, 1997] N.Y. Lee and C.R. Litecky, "An Empirical Study of Software Reuse with Special Attention to Ada", *IEEE trans.software eng.*, vol 23, no. 9, September 1997.

[Nesi, 1998] P. Nesi, "Managing OO Project Betters", *IEEE Software*, pp. 50-60, July/August, 1998.

[O'Leary, 1998] D.E. O'Leary, "Using AI in Knowledge Management: Knowledge Bases and Ontologies", *IEEE Intelligent Systems*, 1998.

[O'Leary, 1998]a D.E. O'Leary, "Knowledge Management System: Converting and Connecting", *IEEE Intelligent Systems*, 1998.

[Robillard, 1999] P.N. Robillard, "the Role of Knowledge in Software Development", *COMMUNICATIONS OF THE ACM*, January, Vol. 42, No. 1, 1999.

[Sumner, 1999] M. Sumner, "Knowledge Management: Theory and Practice", *ACM*, 1999.

[Tuomi, 1998] I. Tuomi, "Data Is More Than Knowledge: Implications of the Reversed Knowledge Hierarchy for Knowledge Management and Organizational Memory", *IEEE*, 1998.

[Vanhoenacker, 1999] J. Vanhoenacker, A. Bryant, G. Dedene, "Creating a Knowledge Management Architecture for Business Process Change", *ACM*, 1999.

저자 소개

안성욱

1999. 전산학 학사, 숭실 대학교
현재. 숭실대학교 컴퓨터학부 대학원 (석사과정)
관심분야 : 객체지향 기술, 웹 기반 모바일 컴퓨팅(Mobile Agent, XML)

이남용

1979, 전산학 학사, 숭실대학교
1983, 경영학 석사, 고려대학교
1993, 경영학 박사, 미시시피주립대학교
1993 - 1996 한국국방연구원(국방정보체계연구소) 군수체계 연구부장
1997 - 1999 한국국방연구원(국방정보체계연구소) 정보체계 연구부장
1999.9 - 현재 숭실대학교 컴퓨터학부 조교수
관심분야 : 전자상 거래, 객체지향 기술, 소프트웨어 재사용 기술

류성열

1976, 전산학 학사, 숭실대학교
1980, 전산학 석사, 연세대학교
1996, 전산학 박사, 아주대학교
1997 - 1998 George Mason University 교환교수
1981.3 - 현재 숭실대학교 컴퓨터학부 교수
1998.3 현재 숭실대학교 정보과학대학원 원장
관심분야 : 리엔지니어링, 분산 객체 컴퓨팅, 소프트웨어 재사용 기술