Information support for distributed teamwork knowledge management

Olexander Shmatko
Doctor of Philosophy in Technical Science, Associate Professor
Department of Software Engineering and Management Information Technologies
National Technical University "Kharkiv Polytechnic Institute"
ORCID: https://orcid.org/0000-0002-2426-900X

Mariia Bilova
Doctor of Philosophy in Technical Sciences, Associate Professor
Department of Software Engineering and Management Information Technologies
National Technical University "Kharkiv Polytechnic Institute"
ORCID: https://orcid.org/0000-0001-7002-4698

Abstract. *Several concepts about knowledge management are proposed in the work: knowledge type (tacit knowledge and explicit knowledge, formal knowledge and informal knowledge); knowledge management in software development methods; knowledge management process (knowledge creation, knowledge storage and retrieve), knowledge transferring and knowledge sharing. The framework and implementation of knowledge management system are described. The framework of knowledge management system is illustrated by four elements: roles, product development, structure and forum.*

Keywords: *distributed teamwork, knowledge distribution, knowledge flow, knowledge management, knowledge sharing, software development*

Introduction

Nowadays the distributed software development becomes more and more popular. Not only for the efficiency and effectiveness the distributed teams provide, but also the distributed teams can supply a whole-day works with lower costs. Knowledge management, as a vital component in distributed software development, is responsible to resources creation, storage and retrieval, knowledge distribution and knowledge sharing during the software development process.

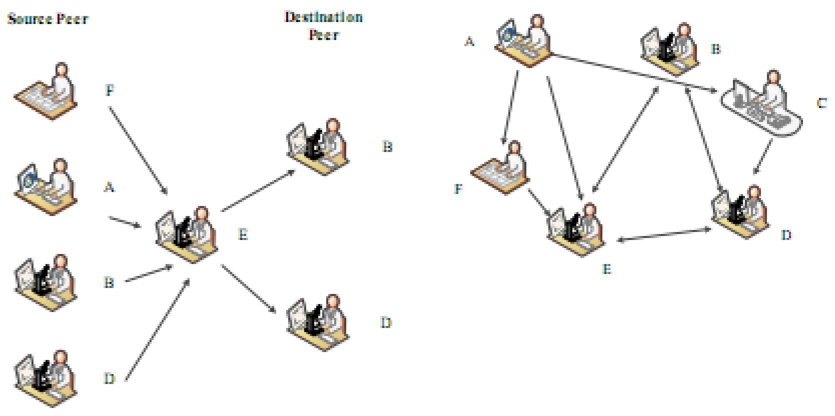
Knowledge sharing model

The process of knowledge sharing involves knowledge creation and import, knowledge capture, knowledge search, and knowledge use, which are equal to the progress of knowledge management. As above debating of knowledge management, in this article both the peer-to-peer (P2P) and C/S (client and server) based knowledge sharing model is considered and an open forum is chose as the description of knowledge contents and sharing.

C/S based architecture is one of the most applied architecture for knowledge management. Under the C/S based architecture, team members as client upload their explicit knowledge to server. Meanwhile, team members can retrieve the knowledge as their requirement from server. C/S architecture belongs to one-to-more or more-to-one mode (more clients correspond to one server), which helps team members obtain various kinds of knowledge from server. For the whole distributed software development team, C/S architecture provides both a safe and convenient centralized knowledge management.

Peer-to-peer architecture is a kind of one-to-one network, which is composed by a series of knowledge nodes. Every knowledge node corresponds to each team member that is the sender or receiver of knowledge flow (or both sender and receiver). Knowledge node generates and requires knowledge in the process of work implementation, which is shown as (Fig. 1).

From the architecture, knowledge sharing is more convenient that every knowledge node can receive knowledge from source peer and send knowledge to destination peer directly. But the peer-to-peer based architecture also brings some troubles: more difficult in management including knowledge retrieval and knowledge selection; more complex in structure of knowledge sharing model. Peer-to-peer based knowledge sharing model is a kind of network topology. Team members with specialization in their domains have different levels and types of knowledge.



1) the knowledge flow of knowledge node (Team member) E

2) knowledge sharing in peer-to-peer model

Figure 1. – Knowledge management under P2P architecture

In the following figure, the peer-to-peer based knowledge sharing model is composed by six knowledge nodes and several knowledge connections with their relevant directions. The knowledge connections with arrows illustrate the direction of knowledge flow. The connection with unidirectional arrow represents that the knowledge node can just be knowledge sender or knowledge receiver. The connection with bidirectional arrow represents the knowledge node can be both knowledge sender and knowledge receiver. For a knowledge node, there are three modes to demonstrate the correspondence among nodes: one-to-one, a knowledge node corresponds (knowledge sender or knowledge receiver) to a knowledge node (knowledge sender or knowledge receiver); one-to-more, a knowledge node (knowledge sender) corresponds to more knowledge nodes (knowledge receivers); more-to-one, more knowledge nodes (knowledge senders) correspond to a knowledge node (knowledge receiver). For knowledge node named E, it has source peer set (the set of knowledge sender) and destination peer set (the set of knowledge receiver). In this mode, knowledge node B and knowledge node D are both source peer and destination peer, for that the connection has bidirectional arrows.

Knowledge flow out from knowledge node is more abundant than the input from source peer. That is why peer-to-peer based knowledge model is more active than the traditional C/S based knowledge model. Also because of geographical distance, the authority to visit the knowledge server may be limited under the C/S based architecture for distributed team of software development. The peer-to-peer based knowledge management model is more suitable to distributed team of software development than C/S based knowledge management model.

Weighted-similarity calculation among knowledge nodes

Each knowledge node has their profile information that reflects the knowledge resource. The knowledge resource is consisted of a series of features. For distributed team of software development, these features reflect the elementary information of team members including: development experience, skills, ability of system analysis, and model of mentality and so on. The profile information or attributes is described as: $(a_1, a_2, a_3, \ldots, a_n)$, where n is the total number of attributes. For a knowledge node A_i it exists $A_i = \left(a_{i1}, a_{i2}, \ldots, a_{in_i}\right)$, where $i \in [1, m]$, m is the number of knowledge source (equal to team members); a_{ij} represents the selection j in j -th feature of A_i. For a feature a_i in $(a_1, a_2, a_3, \ldots, a_n)$ it have several selections as $a_i = (a_{i1}, a_{i2}, a_{i3}, \ldots, a_{in_i})$ where n_i is the amount of selections. For every feature, the amount of related selections is identical. The symbol a_{mn} represents the n -th selection in the m -th feature. The set of knowledge network's knowledge resources is described as $KN = \{A_1, A_2, \ldots, A_m\}$. In knowledge network, knowledge node does not need to connect with all of the other nodes. It only should to connect with the knowledge nodes which behavior similarly to it.

The host peer's source peers set should contain the peers that have the similar profile information with host peer [1]. The mathematical formulation to calculate the similarity between two knowledge nodes is:

$$KN_{sim}(X, Y) = \frac{\sum sim(a_{Xi}, a_{Yi})}{|X| \cdot |Y|}$$

where $KN_{sim}(X, Y)$ formulation is the similarity between knowledge node X and knowledge node Y;

$sim(a_{Xi},a_{Yi})$ formulation is the similarity of feature between knowledge node X and knowledge node Y;

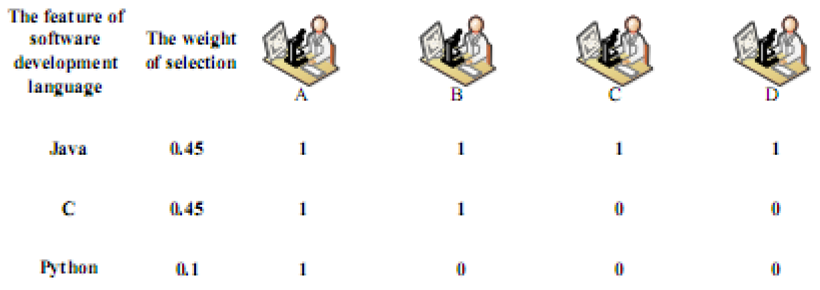
$|X|,|Y|$ are the numbers of features in X node and Y node respectively.

As the numbers of features in two knowledge nodes are different, the similarity needs the process of formalization, so the result is divided by $|X|\cdot|Y|$ in last. On the same time, $sim(a_{Xi},a_{Yi})$ is described as:

$$sim(a_{Xi},a_{Yi}) = \frac{\sum_j w_{ij} \cdot bina(a_{Xji},a_{Yji})}{\sum_j w_{ij}},$$

where w_{ij} is the weight of j-th selection in feature i; $bina(a_{Xji},a_{Yji})$ calculates the binary distance of knowledge features between two knowledge nodes.

The process is described as (Fig. 2).



The feature of software development language	The weight of selection	A	B	C	D
Java	0.45	1	1	1	1
C	0.45	1	1	0	0
Python	0.1	1	0	0	0

Figure 2. – Selections of four knowledge nodes (A, B, C, D) under the feature of software development language

From (Fig. 2), the feature of software development language for team members is different. "1" represents the state of selection is definition while "0" represents the state of selection is default. So for team member A possesses three software development languages: Java, C, and Python. Team member B possesses two software development languages: Java and C.

For software development, not all development languages are equally important. The software system may mainly depend on the Java and C. The weight of Java and C could large than the weight of Python. This regulation is also suitable to other kinds of features. If the similarity in the feature of software development language between A and B, B and C, C and D are:

$$sim(a_A,a_B) = \frac{0.45\cdot 1 + 0.45\cdot 1 + 0.1\cdot 0}{0.45+0.45+0.1} = 0.9$$

$$sim(a_B,a_C) = \frac{0.45\cdot 1 + 0.45\cdot 0 + 0.1\cdot 0}{0.45+0.45} = 0.5$$

$$sim(a_C,a_D) = \frac{0.45\cdot 1 + 0.45\cdot 0 + 0.1\cdot 0}{0.45} = 1$$

Thus the similarity between A and B is 0.9; C and D is 1.0; B and C is 0.5.

The similarity calculation between team members A and B shown as (Fig. 3).

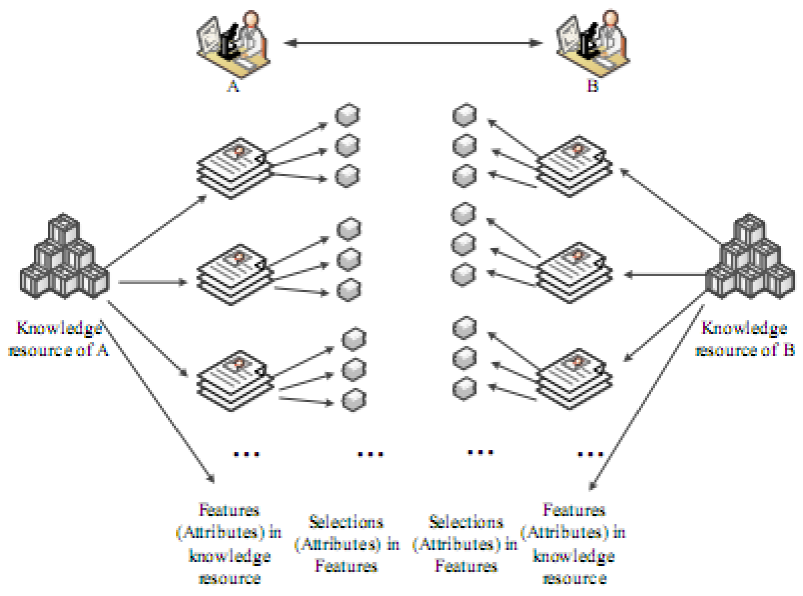

A
B
Knowledge resource of A
Knowledge resource of B
...
...
...
...
Features (Attributes) in knowledge resource
Selections (Attributes) in Features
Selections (Attributes) in Features
Features (Attributes) in knowledge resource

Figure 3. – The knowledge resources of two knowledge nodes (team members)

The process of similarity calculations includes two bottom-up steps: the first step is calculating the similarity of selections in features; the second step is calculating similarity of features. By summary of two steps, the similarity of knowledge resources is obtained. In procedure of knowledge management, the knowledge network initialization and the knowledge network formulation are involving with similarity calculation.

Knowledge network initialization

At the beginning of knowledge network initialization, there are knowledge nodes without connections. To construct the connections, the similarity among knowledge should be calculated at first. Besides the knowledge resource, a knowledge node should equip with source peers set. The source peers set includes the elementary information of source peers and the similarity value among source peers and this knowledge node. Not all team members have a direct relationship during the software development. Thus the size of source peers set is limited to a definite number D. The knowledge node calculates similarity value with other knowledge nodes and selects the D knowledge nodes with the largest value as source peers. There are two ways to initialize the source peers set. The one is global search, which means the knowledge node should search all other knowledge nodes. It costs a great amount of time when the size of network is large. Another is local search. In this search mode, knowledge node searches some of other peers within a definite scope. The other peers may have a similar work type with this knowledge node.

Knowledge network re-formulation

With the team members leaving and joining or information updating, the knowledge management process is dynamic. The knowledge network updates all the time. Authors in [1] advocated that every time knowledge network reformulation needs to search other peers again for finding the top D most similar peers in current states. Two strategies are proposed below to define the search space.

Equally expending. The search space is constructed by source peers. The knowledge peer extends to its source peers. As the procedure is endless, a fathoming depth is defines to limit the search depths.

Unequally expending. In this strategy, fathoming depths are different according the similarity value among source peers. The higher the similarity value is, the deeper the depth will be, vice versa.

In fact both two strategies has a complex calculation and time cost. Putting the similarity values in the sources peer set corresponding to its source peers is a good choice. When the knowledge node updates its information, it needs to recalculate the similarity with its source peers. If the new similarity rank of knowledge node does not fall into the last one, the formulation of knowledge network is not needed and the whole process' time complexity is linear. If the similarity rank of knowledge node falls into the last one, the knowledge network should re-formulation according to the unequally expending principle. When the knowledge node is deleted, which means the team member leaves the team, its destination peers only need to delete it from their source peers set and find a new source peer outside the source peers set. When the new knowledge node joins the knowledge network, it calculates the similarity with other peers to fill its source peers set at the same time other peers need to decide whether put this node to their source peers set or not by calculating the similarity.

Knowledge flow operations

The knowledge network has been constructed and the most important method to manage the knowledge is by knowledge flow. Team members are linked by all kinds of knowledge flow. Team members can get their desired knowledge by knowledge flow and share their own knowledge to others. In [2], the knowledge flow is regarded as a two dimensional region in knowledge space as type-filed (TF) and level filed (LF). The process among knowledge fields is summarized as:

$$< TF_1, LF_1 > \cup < TF_2, LF_2 > = < TF_1 \cup TF_2, LF_1 \cup LF_2 >$$
$$< TF_1, LF_1 > \cap < TF_2, LF_2 > = < TF_1 \cap TF_2, LF_1 \cap LF_2 >$$
$$< TF_1, LF_1 > - < TF_2, LF_2 > = < TF_1 - TF_2, LF_1 - LF_2 >$$
$$< TF_1, LF_1 > \subseteq < TF_2, LF_2 > \ if \ < TF_1 \subseteq TF_2, LF_1 \subseteq LF_2 >$$

For multiple knowledge flow operations, it exists:

$$< TF_1, LF_1 > \cup < TF_2, LF_2 > \cup ... \cup < TF_n, LF_n > = < TF_1 \cup TF_2 \cup ... \cup LF_n, LF_1 \cup LF_2 \cup ... \cup LF_n >$$

$$< TF_1, LF_1 > \cap < TF_2, LF_2 > \cap ... \cap < TF_n, LF_n > = < TF_1 \cap TF_2 \cap ... \cap LF_n, LF_1 \cap LF_2 \cap ... \cap LF_n >$$

$$< TF_1, LF_1 > - < TF_2, LF_2 > - ... - < TF_n, LF_n > = < TF_1 - TF_2 - ... - LF_n, LF_1 - LF_2 - ... - LF_n >$$

Knowledge flows with one knowledge node is shown in (Fig. 4).

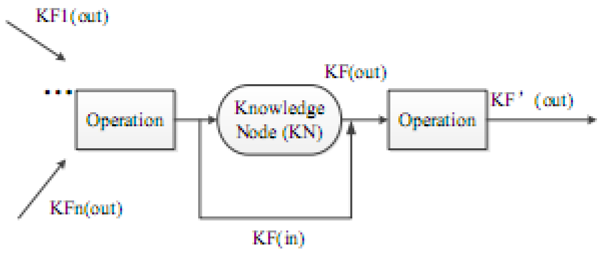


Figure 4. – Knowledge flows with one knowledge node

By logical operation, the input knowledge flows will be integrated into a formalized knowledge flow. The formalize knowledge flow will be input into the knowledge node. The operation within the knowledge node is unclear and new information will be put into the knowledge flow as the output. The operation of knowledge flow also could be not performed.

Framework and structure of knowledge management system

On above sections, it has discussed about characteristics of distributed software development, relationship between knowledge management and distributed software development, types of knowledge, knowledge management process, aims of knowledge management as well as algorithms and models for knowledge sharing and distribution. In this section, it will be introduced a designation of framework and structure for knowledge management system.

The framework of knowledge management system comes along with four processes: knowledge creation (knowledge production and knowledge update), knowledge deposition and retrieve, knowledge distribution (knowledge sharing) and knowledge application (knowledge integration). All four parts is associated with functions (what kind of works the knowledge management undertakes?), roles (what kind of people involves in this process and what kind of reputation they bear?), and events (what kind of achievements the knowledge management can realize?).

The structure of knowledge management system includes function structure (a large function can be divided into several small functions), data structure (attributes of roles and events). Structure of knowledge management system indicates the form of data which is deposited in the database. A complete knowledge management system should satisfy the elementary requirement of distributed software development. It can edit, deposit, search and integrate knowledge. Besides, a practical system ought to certainty obey the software development modes (such as Agile, Waterfall, RUP etc.). But for a knowledge management system in modern software development mode, these functions are not enough. To enhance the competitiveness of company, a knowledge sharing culture should be constructed within the organization. The knowledge management system should act as a convenient and direct knowledge sharing medium. Team members exchange their thoughts and learn the new knowledge through the system. In view of characteristics of distributed software development, the communications among team members are not ideal. For that different location and time zone, the communications are often asynchronous. The system designs a synchronous mechanism that team members can contact with others by sending instant messages. This mechanism can highly reduce the communication cost and save the time expense.

Framework of knowledge management system

The framework of knowledge management system is shown as (Fig. 5).

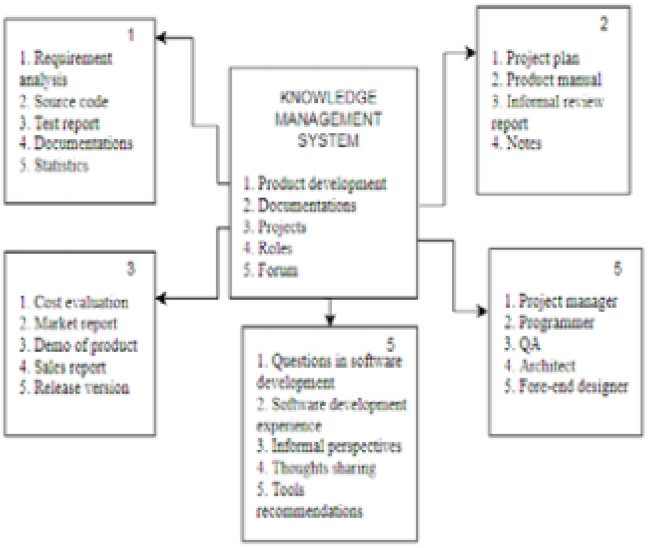


Figure 5. – The framework of knowledge management system

From (Fig. 5), it is clear that knowledge management system is composed to five parts: product development, documentations, project, roles and forum. The details of designs are described as follows.

Product development

Product development is same to the software development, which includes explicit knowledge as: requirement analysis, source code, test report, documentations and statistics etc.

All these explicit knowledge is deposited in different folders. In order to remain old knowledge for revision in the future, the knowledge management system introduces version control. When the team member first uploads files, the system will judge whether the same file has existed. If not, the file will be upload as original; if already existed the same file, the file will be signed with time stamp to distinguish the old version, as Fig. 6 shows.

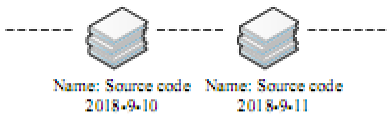


Figure 6. – Documents with version control (Documents with same name and different time stamp)

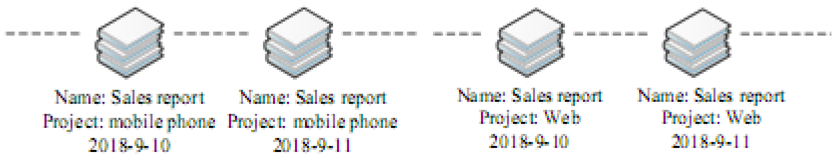
So, in this part only explicit knowledge involves and all explicit knowledge is organized by timeline, which makes framework clear.

Documentations

For distributed software development, documentation includes project plan, product manual, and informal review report and so on. These documentations seem to be useless once it lost effectiveness for a given period of time. However, these documentations still have its effect to be a reference for future works. So the way of deposition for documentations is the same as product development.

Projects

Projects part is the result of product development and documentation. It has cost evaluation, demo of product, market report, sales report, and release version and so on. But the explicit knowledge may belong to different projects. In order to distinct different explicit knowledge from different projects, explicit knowledge in Projects part should have additional remark that points out attributes (Fig. 7).



a) sales report in "mobile phone" project b) sales report in "Web" project

Figure 7. – Explicit knowledge in Project part

Within the same project, explicit knowledge is also remarked with time stamp for distinction. Thus file in Projects parts have two-tuples attributes, which is enough to distinguish every file. This kind of naming mode has many benefits. For new team member, when he/she entered the organization first time, these files with specific identified information can help them learning knowledge precisely. For other team members, when they want to revise the whole process of project, they can easily and quickly locate the files.

Roles

Roles in knowledge management system indicate people who participate in the distributed software development. Roles include project manager, programmer, architect, quality assurance and fore-end designer etc. Project manager works for communicating with customers, analyzing requirements and then assign tasks to other team members. Project manager can be regard as the medium between customers and project programmers. Architect designs the architecture of project. Fore-end designer contributes to user interface design. Then project programmers are responsible for achieving modules of software project and integrate these modules into a complete project. Once the project is finished, quality assurance begins to make tests (like white-box test and black-box test etc.) of software product and writes the test report.

It seems that roles in the same position (as project programmer to project programmer) have stronger links and more chances to meet other project programmers over the whole distributed software development process. But for roles in different positions, like project manager and project programmers, there is less chance to contact with each other. Once the requirement is defined and tasks are distributed to corresponding project programmers, it is no need to make frequent meet. All in all, knowledge management system constructs profile for each team role. The profile contains personal information, skills, position. In the process of knowledge distribution, sender wants to find appropriate receivers and verse vice. Under this circumstances, profile as a strong tool indicates the similarities and difference among team roles.

Forum

Forum is a kind of electronic community, which is desired to encourage knowledge sharing and transformation from tacit knowledge to explicit knowledge [3]. Like "Quora", it provides a bulletin for team members to online edit; modify their questions. It is a kind of knowledge market letting team members sharing, learning and presenting.

When team member has question about software development, he/she can use online editor describing his/her questions and upload it to bulletin. If someone experts in this aspect, the question can be solved by effective solutions. In this part, questioner solves the problem with the help of expert. On the same time, the expert gains reputation from other team members. This is a win-win mode and motivates the knowledge sharing and knowledge distribution.

Team members can also share their experience in forum about distributed software development. Some team members with similar experience may be triggered and easily solve the similar question. For other purposes, team members could express their own informal perspectives. Different perspectives among team members mix and it is more likely for new creativity creation.

The forum part is also used to recommend software development tools for efficient developing. Team members can share the software development tools that they think is effective and efficient on the forum.

In the conclusion, considered the high value of tacit knowledge, which is not belonging to organization but to individuals, the knowledge management should encourage team members transforming their tacit knowledge to explicit knowledge. By writing a post on the forum, team members communicate with each other by their own experience, activities and skills. It is a process with internalization (from explicit knowledge to tacit knowledge: team members read the post on the forum) and externalization (from tacit knowledge to explicit knowledge: team members write down a new post on the forum). Forum fosters the message exchange among team members. Tacit knowledge, as one of the most valuable fortune, is well exploited through the forum. On the one hand, forum encourages team members to transform individual's knowledge to social knowledge (collective knowledge) as company's explicit fortune; on the other hand, the relationship among team members can be strengthened through the informal conversation.

For distributed software development, team members are in different sites and time zones. Forum is designed to increase the motivation for expressing perspectives of distributed team members. Thus, as consideration of these benefits, knowledge management should provide a space as a forum for team members sharing their perspectives and experiences. Also, it provides the function to comment on the post with sentences and even upload files.

Structure of knowledge management system

Structure of knowledge management system includes two aspects: one is deposition structure (for uploaded files in Product development part) and another is the data structure for elements (profiles of roles, post in the forum and all kinds of uploaded files).

Deposition structure

As above mentioned, when team members upload their files, unstructured principles cause the knowledge management chaos. It is needful to make a clear structure of Project development part in knowledge management system. To clearly manage the knowledge in project development, a tree structure is designed (Fig. 8).

For the project, every kind of part has its own folder to deposit source codes, reports, documentation and statistics. The folder named Source code is the warehouse depositing the source code of the project.

As for a complete project, it has several modules, such as database design, user interface design, and algorithm design and so on. For different module, its explicit knowledge is deposited separately. For example, the source code of the project is deposited in the module called Source code while test reports are stored in the Test report module.

Within the same module, all explicit knowledge is deposit together. In order to discriminate them, a version control mechanism is introduced. Within the same module, all file can have the same name, but they will have different time stamp for distinction.

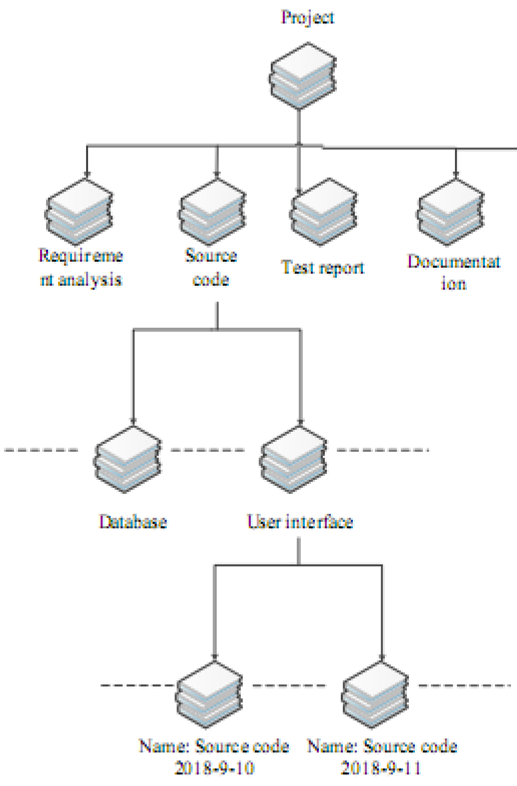
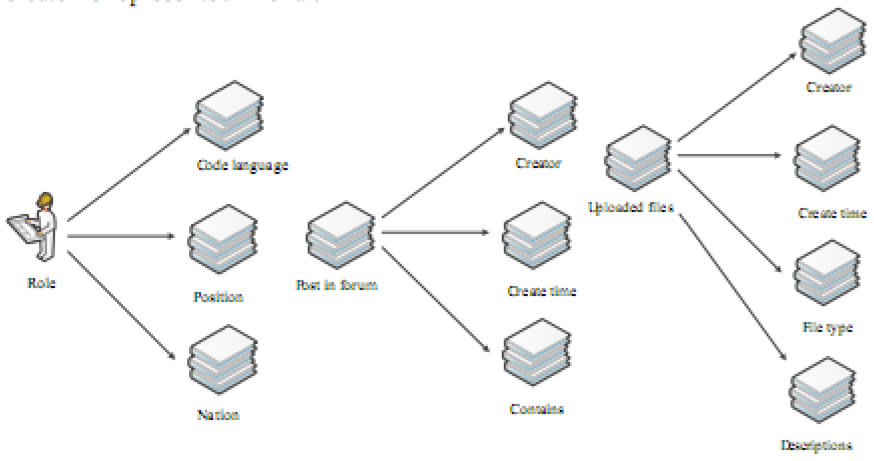


Figure 8. – The tree structure of Project development part

The tree structure makes Project development part clear which can avoid chaos caused by upload files. Additionally, when all the modules is completed, project programmers can easily integrate these modules into a whole project.

Data structure

In knowledge management system, there three elements: role, post in forum and uploaded files. All elements have data structure. The relationship among attributes and elements is described (Fig. 9). File possesses four attributes: upload time, creator, file type and descriptions. To identify a role, the personal profile is needed, which includes: personal information (name, id, create time, create IP and so on), skills (expert in what kind of aspects including program language etc) and position (position in the company, such as project manager, front-end project programmer, back-end project programmer etc.). The space of database prepared for role is limited. Personal information, skills and position can be described in char. For the post in forum, there also three parts: creator (the name of creator), create time (when the post is created) and contains. Like the role, the attributes of creator is represented in char.



a) role with attributes b) post in forum with attributes c) updated files with attributes attributes

Figure 9. – Data structures of three elements (role, post in forum and upload file)

The create time is described in Date Time type (YYYY-MM-DD, 8 bytes). As the long length of contains, it is indicated by Text type.

The implementation of knowledge management system

This section describes the progress of design and implementation on knowledge management system according to proposed model and framework in last section. At first, the requirement analysis is discussed. Then the details on every module and work flow of knowledge management system are illustrated. After that the structure and tables of system dataset are demonstrated. At last, the implementation and test on knowledge system are shown.

There is a clear requirement that develop an integral knowledge system providing knowledge storage, knowledge retrieval, knowledge sharing and updating, team member matching with high efficiency. Besides, the knowledge management system should give a chance to team members for asking, debating and answering question so as to encourage the knowledge sharing. The details of requirement analysis are described as follows.

The elementary functions of knowledge management system includes: knowledge deposit, knowledge sharing and updating;

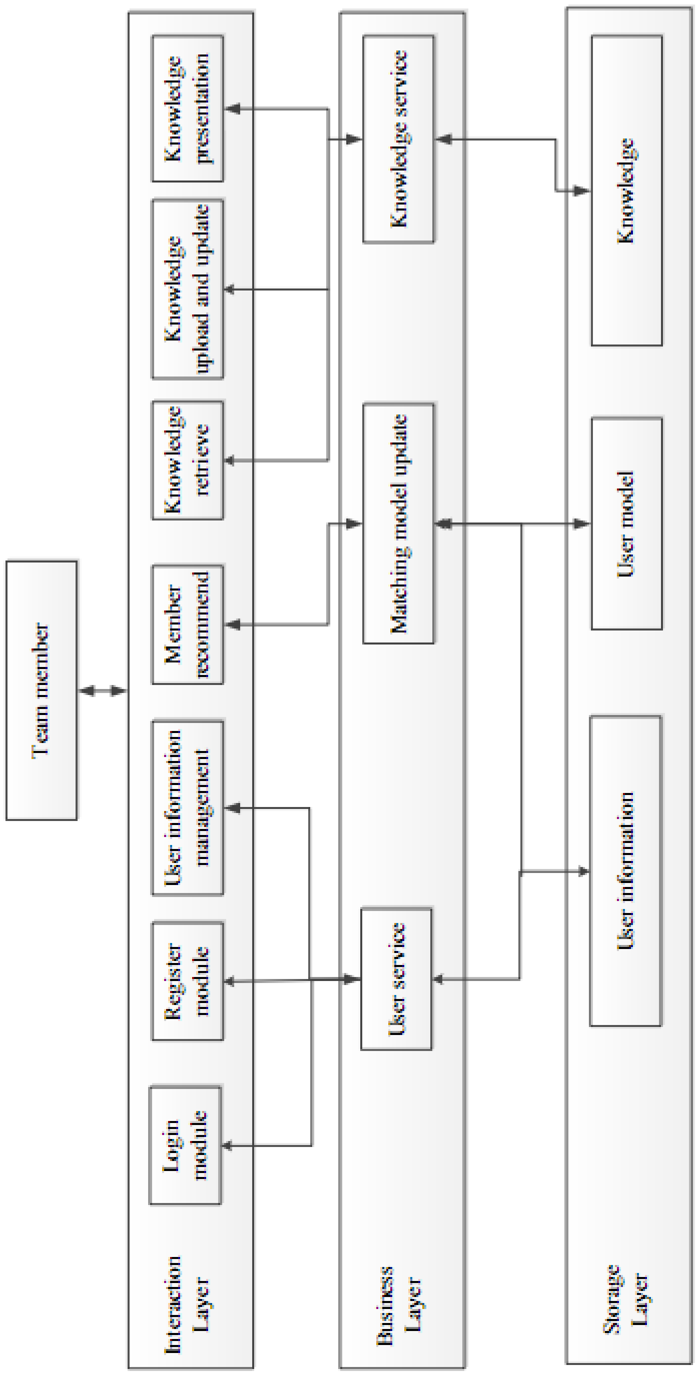


Figure 10.– The structure of knowledge management system

The information of team members is not unchanged. Based on user information the knowledge management system can precisely recommend team members with similar interests for team member matching;

The knowledge management system should provide a function of knowledge retrieval. Team members can find knowledge which they are interested in by using key words;

Because of the distributed software development, the communication between team members is difficult. Thus an effective and convenient medium is necessary.

The knowledge management system is designed by multi-layer and modules, which mainly includes: interaction layer, business layer as well as storage layer. Each layer is consisted of several modules (Fig. 10). The interaction layer provides the interface between user and knowledge management system. Through simple and intuitive interface, the knowledge management collects user information and knowledge. The interactive layer in this system includes: user login module, user register module, user information management module, member recommend module, knowledge retrieve module, knowledge retrieve module, knowledge upload and update module as well as knowledge presentation module.

User login module. Before user enters the knowledge management system, the module will verify the effectiveness of username and password. Only when both username and password is correct, the user is permitted to enter the system and make other operations.

User register module. If the user enters the system at the first time, the user must make the registration. The information of registration includes: user ID, user group ID, email, username and password, created date, position of user, code language and nation, where position, code language and nation are used to make member matching and recommend.

User information management module. The information can be updated by admin so as to make the member recommend precisely.

Knowledge retrieve module. User can find the knowledge in which they are interested by knowledge retrieve module.

Knowledge uploaded and updated module. User can upload their explicit to knowledge management system and update the knowledge in time.

Knowledge presentation module. The knowledge is presented to user. And user can read and download relative knowledge.

The business layer is a connection between interaction layer and storage layer, which satisfy the requirement of users by operating the data from storage. The business layer is consisted of: user service module, matching update module and knowledge service module.

User service module. User service module is applied in user login, user registration and user information management;

Matching update module. This module receives the information from user. Then the similarities between team members are calculated the threshold of similarity is predefined. Only the similarity is larger than the threshold, the corresponding team member can be recommended to user (Fig. 11).

Knowledge service module. This module is responsible for knowledge management as knowledge upload and update (knowledge creation), knowledge retrieve and knowledge representation (knowledge sharing).

The storage layer is responsible for the persistent storage of system data, providing data support for each module of the business logic layer. The data of storage layer includes: user information and knowledge. The knowledge is classified into two classes: formal knowledge and informal knowledge.

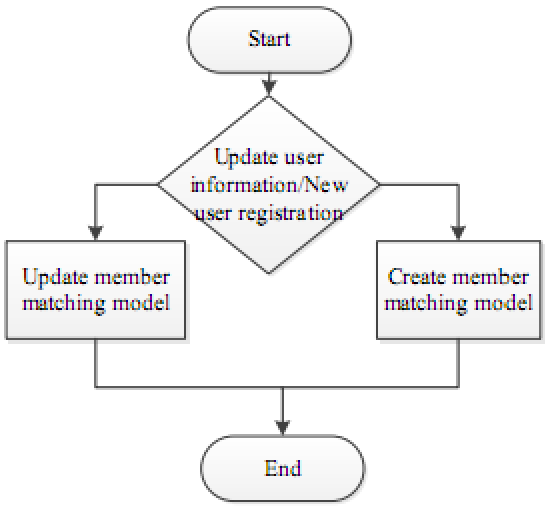

Start
Update user information/New user registration
Update member matching model
Create member matching model
End

Figure 11. – The work flow of matching update module

For formal knowledge, it includes project codes and relative documents. The informal knowledge is other information in threads and posts. Informal knowledge does not have a fixed format which is more flexible and casual.

The existence of informal knowledge aims to encourage team members to share their software development experience, questions and answers in software development progress or some development tools recommendation etc. In knowledge management system, the informal knowledge has two forms: thread and post. Posts are built based on the threads. The relationship between posts and thread is many-to-one: one thread corresponding to many posts. The contents of posts are associated with the thread.

For each kind of formal knowledge, it has an identical standard and fixed format. For distributed software development, the formal knowledge consists of: cost evaluation, market research report, demo of product, sales report, release version, requirement analysis report, test report, statistics and the source code at last but not least. The format of formal knowledge is various which depends on the code language (i.e., php file is created by PHP language; c file is created by C language etc) and instrument (i.e., vsd file is produce by Visio; doc file is built on Word etc.).

In knowledge management system, the formal knowledge is expressed in a unified form. It is regarded as the attachment of thread or post. The formal knowledge is created offline by specific language and instrumentation; then it is uploaded to the knowledge management system.

In order to provide an effective approach of communication for distributed software development team, the knowledge management system is implemented based on web. The architecture of web-based knowledge management system is described as (Fig. 12).

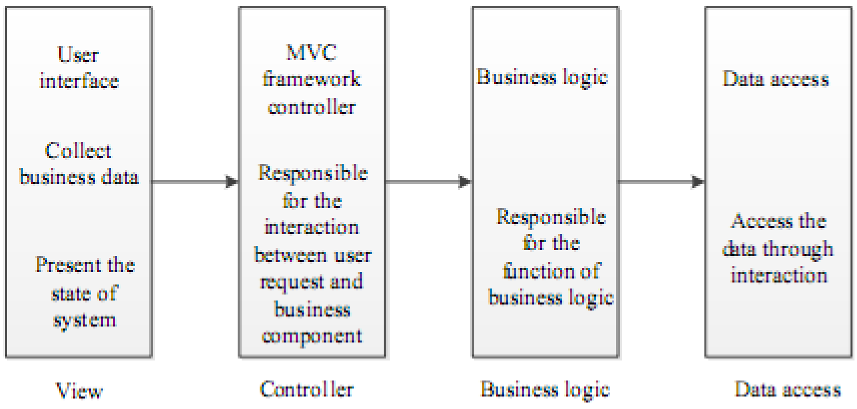

User interface
Collect business data
Present the state of system
MVC framework controller
Responsible for the interaction between user request and business component
Business logic
Responsible for the function of business logic
Data access
Access the data through interaction
View
Controller
Business logic
Data access

Figure 12. – Architecture of web-based knowledge management system

The first part is responsible for collecting the requirement information from users and presenting business data to user. The function in this part includes: data collection, data representation and data verification. The second part is about the interaction between user requirement and business component. The third part is associated with processing specific user requests and consists of a series of business logic stripping. These business logic objects implement the business logic methods required by the system. The implementation of the business logic methods depends on the data access services provided by the data access layer. The fourth part operates for data access. Separating the number of access operations from other business logics allows the business logic components to focus on the implementation of business logic without having to care about the underlying database access details, making the system more maintainable and portable. The work flow of web-based knowledge management system is described as: in knowledge management system, there are two kinds of users: general user and administrator. For general users, they can login the system, upload their knowledge to system, sharing knowledge to other team member, get recommendation on similar team members, search for other users and knowledge etc.

For administrator, it possesses all of the same rights as the general user. Besides, administrator can manage the information of users including user registration, user deletion, user information update, distribution of permission, knowledge management and so on. The concrete procedure is shown in (Fig. 13).

Administrator has a higher authority than general user. Besides the rights shown in Fig. 13, the administrator can manage information of users, like user registration, user inform, distribution of permission (Fig. 14).

In order to protect the security of information, the knowledge management system restricts the access of visitors and only the administrator have the right to register a new general user. In addition to other functions administrator can restrict and moderate the right of general user on reading, updating, deleting, uploading, writing, and downloading knowledge in the system. The front-end of web-based knowledge management system adopts BootStrap 4 + JQuery 3 framework while the back-end is designed by the combination of PHP 7 and MySQL 5.6.

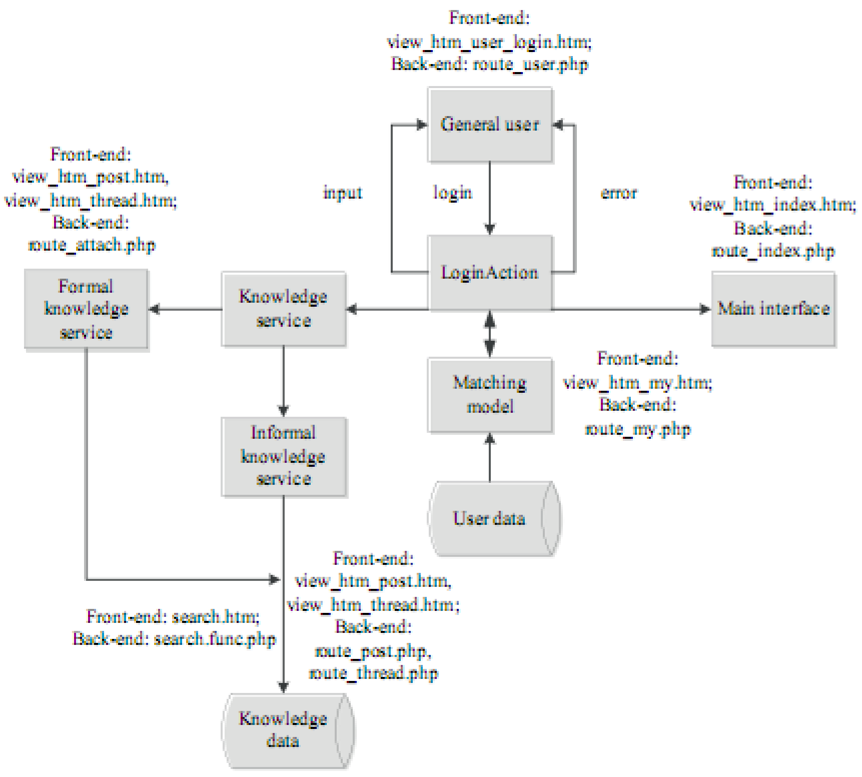

Front-end: view_htm_user_login.htm; Back-end: route_user.php
General user
input
login
error
LoginAction
Front-end: view_htm_post.htm, view_htm_thread.htm; Back-end: route_attach.php
Formal knowledge service
Knowledge service
Front-end: view_htm_index.htm; Back-end: route_index.php
Main interface
Matching model
Front-end: view_htm_my.htm; Back-end: route_my.php
Informal knowledge service
User data
Front-end: view_htm_post.htm, view_htm_thread.htm; Back-end: route_post.php, route_thread.php
Front-end: search.htm; Back-end: search.func.php
Knowledge data

Figure 13. – The concrete work procedure of general user

Test on knowledge management system

This section will execute the test on the elementary function of web-based knowledge management system. The test functions are: login function, registration function, formal knowledge management function, informal knowledge management function, similar team member recommendation function, user/knowledge search function, and permission setting function.

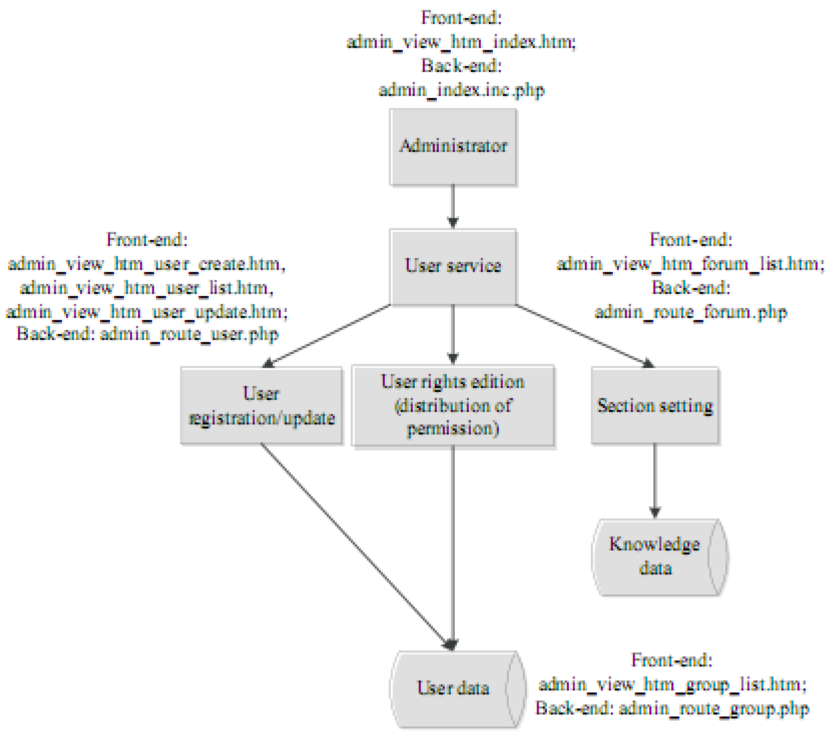

Front-end:
admin_view_htm_index.htm;
Back-end:
admin_index.inc.php
Administrator
User service
Front-end:
admin_view_htm_user_create.htm,
admin_view_htm_user_list.htm,
admin_view_htm_user_update.htm;
Back-end: admin_route_user.php
Front-end:
admin_view_htm_forum_list.htm;
Back-end:
admin_route_forum.php
User registration/update
User rights edition (distribution of permission)
Section setting
Knowledge data
User data
Front-end:
admin_view_htm_group_list.htm;
Back-end: admin_route_group.php

Figure 14. – The specific work procedure of administrator

Table 1. – Login function testing

Input	Input username="Coder_6", password="123456";
	Input username="Coder_5", password="1234";
	Input username="Coder_5", password="123456".

Output	Login failed, prompt "User does not exist"; Login failed, prompt "Password Incorrect"; Login successful, jumps to user main interface.
Result	Pass

For login function of the system, design 3 sets of test cases, test cases and test results are shown in Table 1 and the execution procedure is shown in (Fig. 15).

For registration function of the system, design 3 sets of test cases, test cases and test results are shown in (Table 2) and the execution procedure is shown in (Fig. 16).

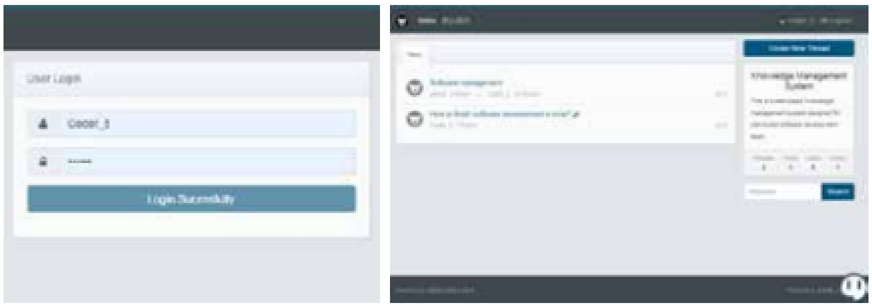


Figure 15. – Login function testing

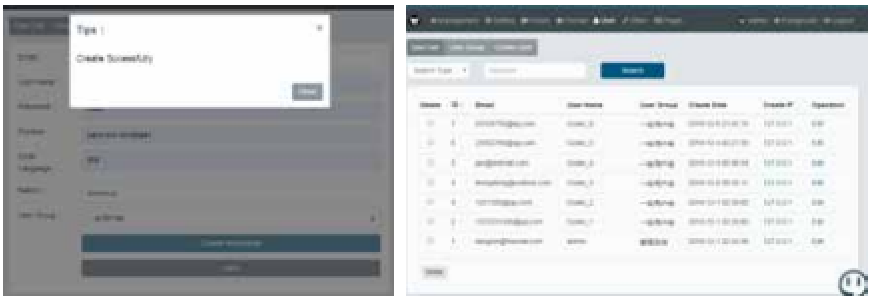


Figure 16. – Registration function testing

Table 2. – Registration function testing

Input	1 Input email = "69126755" username = "Coder_6", password = "123456" position = "back-end developer" code language = "php" nation = "American";
	2 Input email = 69126755@qq.com username = "Coder_5", password = "123456" position = "back-end developer" code language = "php" nation = "American";
	3 Input email = 69126755@qq.com username = "Coder_6", password = "123456" position = "back-end developer" code language = "php" nation = "American".

Output	1 Registration failed, prompt "Email format illegal";
	2 Registration failed, prompt "User already exists";
	3 Registration successful, prompt "Create Successfully" then jumps to user list interface.

Result	Pass

Formal knowledge management function includes knowledge upload, knowledge edit, knowledge delete in post and in thread. Test cases and test results are shown in (Table 3) and the execution procedure is shown in (Fig. 17–18).

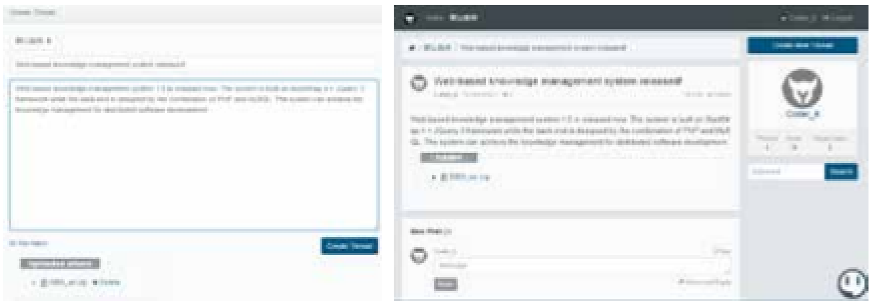


Figure 17. – Formal knowledge management function testing (part 1)

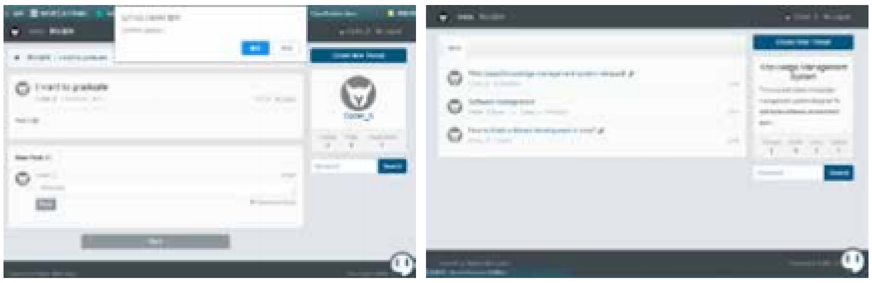


Figure 18. – Formal knowledge management function testing (part 2)

Table 3. – Formal knowledge management function testing

Input and step	1 Create a theme, write a description of the upload knowledge in the edit box, and click the "Add Attach" button; 2 Enter the theme, click the "Edit" button, modify the description of the uploaded knowledge in the edit box, then click the "Upload Attachment" button, delete or re-upload the file, and finally click the "Edit Post"; 3 Enter the theme, click the "Delete" button.
Output	1 Upload successfully, jumps to user main interface; 2 Update successfully, jumps to user main interface; 3 Delete successfully, jumps to user main interface.
Result	Pass

Informal knowledge management function includes knowledge creation as thread and post, knowledge edit, knowledge delete. Test cases and test results are shown in (Table 4) and the execution procedure is shown in (Fig. 19).

Similar member recommendation function can recommend similar team members to user based on personal information (position, code language and nation). Test cases and test results are shown in (Table 5) and the execution procedure is shown in (Fig. 20). User/knowledge search function is used to search relative user or knowledge by key words. Test cases and test results are shown in Table 6 and the execution procedure is shown in (Fig. 21).

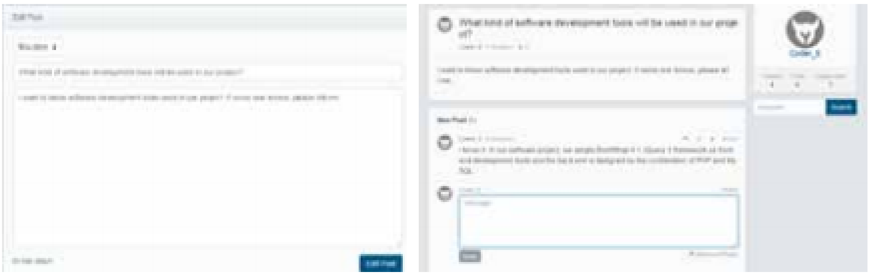


Figure 19. – Informal knowledge management function testing

Table 4. – Informal knowledge management function testing

Input and step	1. Create a thread, write a description in the edit box, and click the "Create Thread" button; 2. Create a post, write a description in the edit box, and click the "Reply" button; 3. Update a thread, click "edit" button, edit a description in the edit box, and click the "Edit Post" button; 4. Update a thread, click "delete" button.
Output	1. Create successfully, jumps to thread main interface; 2. Update successfully, jumps to thread main interface; Edit successfully, jumps to thread main interface; Delete successfully, jumps to thread main interface.
Result	Pass

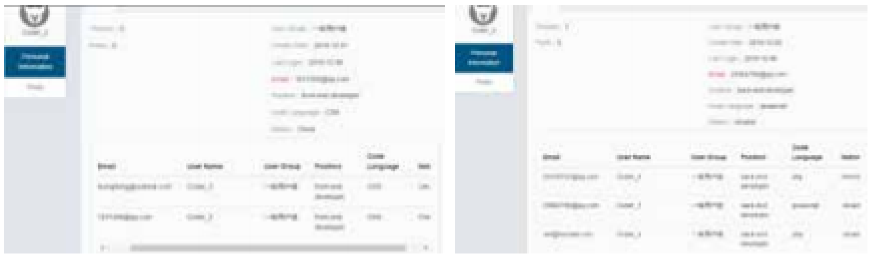


Figure 20. – Similar team member recommendation function testing

Table 5. – Similar team member recommendation function testing

Input and step	1. Login the system using username "Code_2", check the personal information; 2. Login the system using username "Code_5", check the personal information.
Output	1. Recommend the similar team members; 2. Recommend the similar team members.
Result	Pass

Table 6. – User/knowledge search function testing

Input and step	1. Input the uid "2", and click the "Search" button; 2. Input the uid "7", and click the "Search" button; 3. Input the word "knowledge", and click the "Search" button; 4. Input the word "university", and click the "Search" button.
Output	1. Search failed, prompt "No result"; 2. Search successfully, present the information of user with uid="7"; 3. Search successfully, present the knowledge associated with "knowledge"; 4. Search failed, prompt "No result".
Result	Pass

Permission setting function set the scope of operation for users. Test cases and test results are shown in (Table 7) and the execution procedure is shown in (Fig. 22).

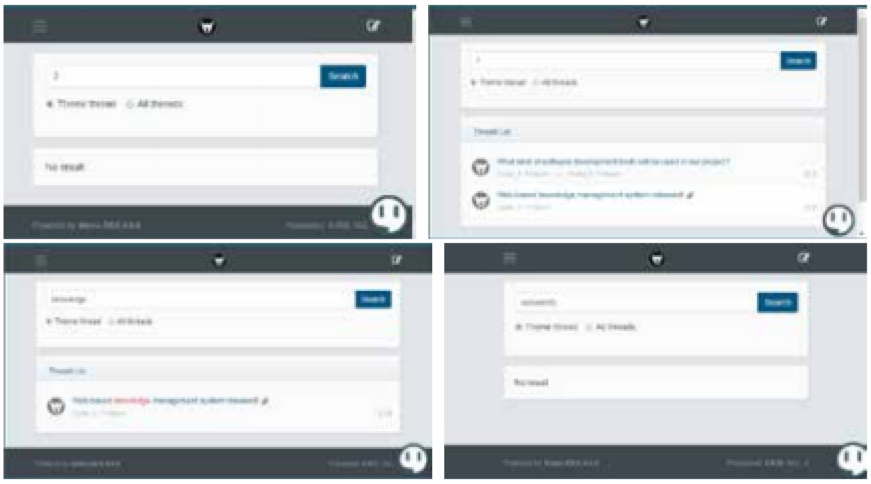


Figure 21. – User/knowledge search function testing

Table 7. – Permission setting function testing

Input and step	1 Set the permission that only team members can read and write the knowledge in the system.
Output	1 Only team member can read and write the knowledge and the visitors is forbidden.
Result	Pass

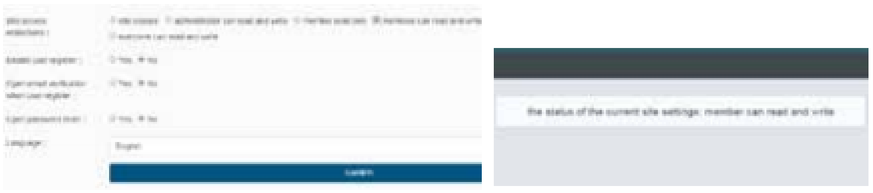

the status of the current site settings: member can read and write

Figure 22. – Permission setting function testing

Conclusions

According to the characteristics and features of distributed software development team, this paper proposed a web-based knowledge management system. Especially, it has been verified that there are a great deal of merits for knowledge transformation from tacit knowledge to explicit knowledge and knowledge sharing. Meanwhile, the knowledge transformation often occurs in the procedure of communication among team members who have similar interests. Thus, this work emphasis the social interaction and devise a similarity calculation model to evaluate the similarity among team members.

In this work several concepts about knowledge management are proposed: knowledge type (tacit knowledge and explicit knowledge, formal knowledge and informal knowledge); knowledge management in software development methods; knowledge management process (knowledge creation, knowledge storage and retrieve), knowledge transferring and knowledge sharing. Finally, the work described the framework and implementation of knowledge management system. The framework of knowledge management system is illustrated by four elements: roles, product development, structure and forum.

References:

1. Zhen L., Jiang Z., Song H. Distributed recommender for peer-to-peer knowledge sharing. Information Science, 180, 2010. – P. 3546–3561.
2. Zhuge H. A knowledge flow model for peer-to-peer team knowledge sharing and management. Expert System with Application, 2002. – P. 23–30.
3. Walsh J.P., Henderson C.M., Deighton V. Negotiated belief structures decision performance: an empirical investigation. Organizational Behavior Human Decision Processes, 42. – P. 194–216.