Decision Support Systems 55 (2013) 12–30

Contents lists available at SciVerse ScienceDirect

Decision Support Systems

journal homepage: www.elsevier.com/locate/dss

Discovering role-based virtual knowledge flows for organizational knowledge support

Duen-Ren Liu *, Chih-Wei Lin, Hui-Fang Chen

Institute of Information Management, National Chiao Tung University, No. 1001 Ta Hsueh Rd., Hsinchu 300, Taiwan

A R T I C L E I N F O

Article history:
Received 20 May 2011
Received in revised form 24 October 2012
Accepted 4 November 2012
Available online 28 January 2013

Keywords:
Knowledge flow
Knowledge flow view
Knowledge support
Knowledge management
Role
Ontology

A B S T R A C T

In knowledge-intensive work environments, workers need task-relevant knowledge and documents to support the execution of tasks. A knowledge flow (KF) represents an individual's or group's knowledge-needs and referencing behavior of codified knowledge during the performance of organizational tasks. Through knowledge flows, organizations can provide workers with task-relevant knowledge to satisfy their knowledge-needs. In teamwork environments, knowledge workers with different roles and task functions usually have diverse knowledge-needs, but conventional KF models cannot satisfy such needs. In a previous work, we proposed a novel concept and theoretical model called Knowledge Flow View (KFV). Based on workers' diverse knowledge-needs, the KFV model abstracts knowledge nodes of partial KFs and generates virtual knowledge nodes through a knowledge concept generalization procedure. However, the KFV model did not consider the diverse knowledge-needs of workers who play different roles in a team. Therefore, in this work, we propose a role-based KFV model that discovers role-based virtual knowledge flows to satisfy the knowledge-needs of different roles. First, we analyze the level of knowledge required by workers to fulfill various roles. Then, we develop role-based knowledge flow abstraction methods that generate appropriate virtual knowledge nodes to provide sufficient knowledge for each role. The proposed role-based KFV model enhances the efficiency of KF usage, as well as the effectiveness of knowledge sharing and knowledge support in organizations.

1. Introduction

In knowledge-intensive work environments, workers need task-relevant knowledge and documents to support their execution of tasks. Thus, how to effectively fulfill workers' knowledge-needs by preserving, sharing and reusing task-relevant knowledge is an important issue in realizing knowledge management and promoting business intelligence. Organizations can provide task-relevant knowledge through knowledge flows (KF), which represent the flow of an individual's or group's knowledge-needs and referencing behavior of codified knowledge during the performance of organizational tasks [16].

In recent years, a considerable number of studies have focused on KF models and applications in business and scientific research contexts. One major area of research focuses on knowledge sharing among knowledge workers. For example, researchers cite prior research and propose new ideas through publishing papers, thereby creating KFs in the realm of science [46]; and in the business domain, KFs facilitate knowledge sharing during the execution of tasks [45]. KFs can be discovered by analyzing workers' knowledge-needs, after which appropriate codified knowledge can be recommended to workers based on the KFs [16,20]. When a task involves teamwork, knowledge workers have different roles and task functions, so they usually have diverse knowledge-needs. However, conventional KF models do not provide different perspectives of a KF to fulfill team members' diverse needs. Although several KF models have been proposed, they do not consider the concept of virtual knowledge flows, which provide abstracted knowledge. In a previous work [21], we proposed an approach called the knowledge flow view (KFV) model, which constructs virtual knowledge flows to serve workers' knowledge-needs. A virtual knowledge flow (VKF) is derived from a KF and provides abstracted knowledge. Given the knowledge-needs of different workers, the model abstracts some knowledge nodes of a KF to generate several virtual knowledge nodes by an order-preserving approach and a knowledge concept generalization procedure.

In the KFV model, we investigated VKFs and developed algorithms to generate such flows, but we did not consider workers' knowledge-needs in terms of the different roles they play in an organization. If a task involves teamwork, workers' knowledge-needs will vary, depending on the roles they play. To ensure effective cooperation, each worker not only needs a specific level of knowledge to perform his/her individual task, but will also need a general level of knowledge about the other workers' tasks. For example, in a computer manufacturing company, engineers are responsible for product development and marketing people design strategies to launch and promote new products. In this scenario, engineers need a specific level of technical knowledge, but marketing people need only a general level of such technical knowledge to assist

* Corresponding author. Tel.: +886 3 5131245; fax: +886 3 5723792.
E-mail addresses: dliu@mail.nctu.edu.tw (D.-R. Liu), jwlin.iim93g@nctu.edu.tw (C.-W. Lin), hfchen@iim.nctu.edu.tw (H.-F. Chen).

0167-9236/$ – see front matter
http://dx.doi.org/10.1016/j.dss.2012.11.018

them in communicating with the engineers. Thus, the KFV model is required to include the aspect of roles in order to apply knowledge management applications in teams [15]. Since workers' knowledge requirements may vary, it is essential that organizations provide role-based VKFs to ensure effective cooperation and knowledge sharing. The concept of role-based VKFs has not been addressed in previous studies on KFV models. Therefore, in an attempt to fill this research gap, we propose a role-based KFV model to fulfill workers' knowledge-needs. The model analyzes the levels of knowledge required by workers based on their roles, and develops role-based knowledge flow abstraction methods that generate virtual knowledge nodes to provide the appropriate level of knowledge for each role.

This work is targeted as a theoretical paper to establish the role-based KFV model and extend knowledge flow research to cooperative teams for organizational knowledge support. To the best of our knowledge, the proposed role-based KFV model is one of the pioneering theoretical studies to illustrate a comprehensive formal representation of VKFs from the perspectives of roles and operations. This work introduces three innovative concepts: (1) identifying VKFs in role and operation perspectives, (2) integrating an approach for evaluating role-knowledge node relevance and an order-preserving algorithm to derive VKFs, and (3) applying role-operation knowledge requirements to facilitate knowledge concept abstraction. From the perspective of cooperation and knowledge sharing, our research facilitates collaboration in an organization. The proposed role-based KFV model can enhance the theoretical scope of KF research. In addition, our work improves the efficiency of KFs, as well as the effectiveness of knowledge sharing and knowledge support in organizations.

The remainder of this paper is organized as follows. The next section contains a review of related work. Then, we illustrate the foundation of the KFV model in Section 3 and discuss the important concepts of the role-based KFV model in Section 4. Methods of generating role-based VKFs are introduced in Section 5. Next, we conduct a basic system design and present a case study in Sections 6 and 7, respectively. Finally, we summarize our conclusions and indicate possible directions for future research in Section 8.

2. Related work

In this section, we discuss the research background and related work on knowledge management, knowledge flow, ontology and process-view abstraction.

2.1. Knowledge management and knowledge support

Knowledge is one of the key assets that organizations use to maintain a competitive advantage [31]; and knowledge management provides the principles of creation, organization, transfer and application of knowledge within enterprises [40]. Generally, knowledge is classified as tacit or explicit knowledge, and information technology (IT) is regarded as a natural medium for managing knowledge [4]. Organizations usually use community-based electronic discussion platforms to transfer tacit knowledge from individuals to a knowledge repository [8]. In task-based business environments, knowledge management systems (KMS) facilitate the preservation, reuse and sharing of knowledge, and also support collaboration among workers. For example, based on the specifications and process-context of a task, the KnowMore system [1] provides context-aware knowledge retrieval and delivery to support workers' procedural activities. Holz et al. [13] proposed a similarity-based approach that organizes desktop documents and proactively delivers task-specific information to users. The task-based *K*-support system [24,25,39] provides knowledge to adaptively meet a worker's dynamic information needs by analyzing his/her access behavior or relevance feedback on documents. Furthermore, because of the nature of teamwork, a collaborative mechanism is essential for establishing KMS [2,43].

2.2. Knowledge flow

Knowledge flow (KF) research focuses on how KFs transmit, share and accumulate knowledge in a team. In a workflow situation, work knowledge may flow among workers, while process knowledge may flow among various tasks [45,47,48]. Thus, the KF reflects the level of knowledge cooperation between workers or processes, and influences the effectiveness of teamwork or workflow. The KF in a software development team can gather knowledge from one team member and carry it to another member, and thereby facilitate knowledge sharing [45]. To improve the efficiency of teamwork, Zhuge [47] proposed a pattern-based approach which combines codification and personalization strategies to design an effective knowledge flow network. To enable the automation of KFs, Sarnikar and Zhao [32,33] developed a framework of knowledge workflow to automate KFs by integrating workflow and knowledge discovery techniques. Rodriguez et al. [30] posited that KFs in communities of practice help members to share their knowledge and experience about a specific domain, in order to complete their tasks. KFs can also facilitate knowledge sharing and reuse in research environments. Specifically, a citation network can be seen as a KF that disseminates knowledge among researchers. Citing a scientific article implies the occurrence of a KF between the authors of the article and the person citing it.

In recent years, several KF models have been proposed. Luo et al. [26] designed a Textual Knowledge Flow (TKF) model for a semantic link network. TKF can recommend appropriate browsing paths to users after evaluating their interests and inputs. KFs can also represent the sequence of knowledge-needs and/or knowledge reference patterns when workers perform tasks. Lai and Liu [16] constructed a time-ordered KF model to determine the sequence of workers' knowledge referencing behavior. Workers can obtain knowledge to satisfy their needs through the KFs discovered from document access logs. Moreover, the referencing sequence in weblogs can be regarded as a KF, and described as a sender-message-receiver model, since a blogger's weblog post may include a hyperlink to a weblog post of another blogger [3]. Kim et al. [14] proposed a KF model that utilizes a process-oriented approach to capture, store and transfer knowledge. Zhang et al. [41] used Petri-net to model a KF. Specifically, a knowledge node can generate, learn, operate, understand, synthesize and deliver knowledge based on four types of flow relations: creation, merging, replication and broadcasting. Zhao et al. [42] introduced a method that integrates business processes and KFs by dividing KFs into sequence, distribution, combination and self-reflection, based on a Role-Activity-Diagram model.

2.3. Ontology

Ontology illustrates how people view the world. It represents the objects and their inter-relationships in a specific domain. More precisely, it is a conceptualization mechanism that defines the concepts of objects explicitly and constructs a hierarchical structure of the objects' inter-relationships [11,28]. The mechanism has been applied in many domains. For example, the common terminologies and knowledge concepts in an ontology can improve the problem-solving capability and efficiency of a supply chain [5]. Wikipedia articles and categories can also be used as an ontology to predict the concepts of documents [35]. In an organization, an ontology defines the knowledge concepts that are understood throughout the organization, and provides a hierarchical structure to demonstrate the relationships among the concepts [29]. Building ontology is an evolving process and involves many techniques and tools to facilitate the whole process. Obviously, the construction process would include an evaluation and feedback mechanism to gradually improve ontology quality and obtain common understanding in organizations [18,28,36]. For example, Uschold and King [37] proposed a skeletal methodology to build an enterprise ontology; it comprises four phases: scoping, building, evaluating and documenting. Du, Li and King [9] designed a six-phase process that includes the preparation, transformation, clustering, recognition, refinement and revision for extracting ontology from unstructured HTML pages. Therefore, involving users in the evaluation or refinement phase is essential for gradually adjusting the quality. Many ontology-building tools, such as Protégé, OntoEdit and SNet-Builder, can effectively support the ontology construction process to serve predefined purposes and meet users' requirements [6,27].

2.4. Process-view abstraction

Workflow Management Systems (WfMS) are effective process management tools that allow businesses to analyze, simulate, design, implement, control and monitor their overall business processes [10,17]. Processes describe the operation flows involved in performing business activities. In practice, participants involved in a workflow need a flexible workflow model that is capable of providing appropriate process information [22]. Because of the increasing complexity of business processes and the variety of participants, it is beneficial to define virtual processes from different perspectives. Liu and Shen [22] presented a novel concept of process abstraction called process-view. It is a virtual process derived from a base process to provide abstracted process information. The process-view is generated by an order-preserving approach, which ensures that the original order of the activities in a base process is preserved. Under the process-view concept, a WfMS can provide various views of a process for different participants within an organization or cross organizations [23]. Shen and Liu [34] proposed a role-based approach to discover role-relevant process views for different workflow participants. They developed algorithms to generate such views automatically, based on the relevance degrees between roles and tasks.

3. Foundation of knowledge flow view model

A virtual knowledge flow (VKF) is an abstract form of a base knowledge flow (BKF). It reveals abstract knowledge by providing customized views of a KF to team members. In our previous work [21], a novel knowledge flow view (KFV) model was proposed to construct VKFs to serve workers' knowledge-needs. The following summarizes the concepts and definitions of the KFV model. Please refer to [21] for detailed definitions, semantics and examples.

3.1. Virtual knowledge flow: an abstract form of a base knowledge flow

A KF that may have multiple VKFs is referred to herein as a base knowledge flow (BKF). A VKF is generated from a BKF and is considered to be a view of the BKF. We take an example from a mobile phone company to explain how to identify a BKF.

Fig. 1 shows a business process of mobile phone development. To derive BKFs of the business process, KF designers may either consult domain experts or investigate workers' document access logs to identify participants' knowledge-needs. A collection of the knowledge-needs and the order of referencing sequences would be used to construct BKFs, which represent participants' knowledge-needs by knowledge concepts. These knowledge concepts are stored in base knowledge nodes (BKNs), respectively. Fig. 2 illustrates an example of the BKFs to explain the concepts of base knowledge flows. In this example, KF designers derive the BKF's knowledge dependencies from the process level dependencies because it is a more intuitive and easier way for team members to understand. Nevertheless, KF designers can apply different ways to set the knowledge dependencies from other perspectives. Generally speaking, the knowledge dependencies in a BKF indicate the referencing sequence of knowledge (information) in task performance which may occur in a distributed software development team [44], an academic research project [16] or a web exploration [3]. So, knowledge dependencies do not always relate to process level dependencies. In practice, KF designers are responsible for setting knowledge dependencies based on the characteristics of applications.

In the KFV model, we adopt an order-preserving approach and a knowledge concept generalization mechanism to generate VKFs based on task functions and organizational security rules. Hence, team members have their own VKFs to fulfill their knowledge-needs and assist them to obey security policy.

Suppose that the BKF in Fig. 3 is the corresponding BKF of a manufacturing process. Product managers need not know all of the knowledge concepts in detail, but they must have general manufacturing knowledge to understand yield trends and increase communication effectiveness with factory members. So, KF designers may design an appropriate VKF for the product managers as follows: base knowledge nodes k_1 and k_2 are abstracted to a virtual knowledge node vk_1; k_3, k_4, k_5 and k_6 are abstracted to vk_2. In addition, manufacturers have their own VKF to serve for their knowledge-needs. Consequently, team members have respective VKFs to represent their knowledge-needs in collaborative environments.

3.2. Basic definitions of the knowledge flow view model

BKFs and VKFs are formally defined in the KFV model. A BKF consists of base knowledge nodes and dependencies. A BKN contains knowledge concepts to represent knowledge-needs, and a dependency is the order between two BKNs. Likewise, a VKF, an abstract form of a BKF, consists of virtual knowledge nodes (VKNs) and virtual dependencies. A VKN is formed by abstracting a set of BKNs. Besides, a VKN contains knowledge concepts derived from its member knowledge nodes' knowledge concepts. Those knowledge concepts can be identified in organizational domain ontology. Fig. 4 shows the components of the KFV model, in which the basic definitions of BKF and VKF are presented accordingly.

3.2.1. Definition 1 (domain ontology, O)

Domain ontology is constructed to define the knowledge concepts and their hierarchical relationships in an organization. It is divided into different knowledge categories. We define the ontology as $O = \langle C, HR \rangle$, where C is a set of knowledge concepts derived from a domain; and HR is a set of hierarchical relations that define the parent–child relationships between the knowledge concepts in C. HR is formally expressed as $HR = \{hr \mid hr \in C \times C\}$. Given two knowledge concepts: x and y, if x has a downward link to y (or y has an upward link to x) in the concept hierarchy, then x is a parent concept of y and y is a child concept of x. Two semantic relations, *Generalization* and *Specialization*, are used to describe the parent–child relations. The

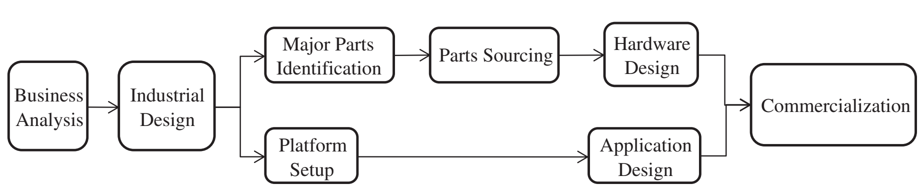

Fig. 1. Business process of mobile phone development.

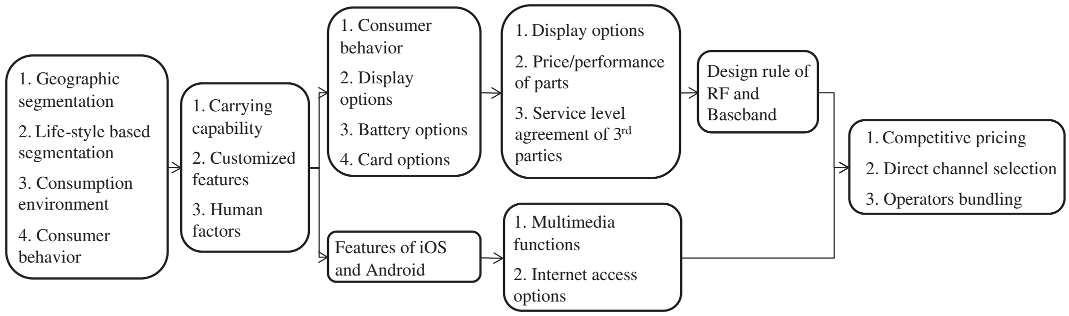

Fig. 2. A sample BKF of the mobile phone development process.

relation between a parent concept x and a child concept y is formally expressed as: $Specialization(x) = \{y \mid y \text{ is a child concept of } x\}$ and $Generalization(y) = \{x \mid x \text{ is a parent concept of } y\}$.

3.2.2. Definition 2 (concept level, CL)

A knowledge concept is mapped to the corresponding CL in the domain ontology. CL is defined by letting the root of the domain ontology be level one. If a knowledge concept is at level l, then its child concepts are at level $l+1$. The CLs indicate the levels of knowledge concepts and represent their granularity; that is, the knowledge concepts with larger CLs (i.e., in the lower levels of domain ontology) are more specific than those with smaller CLs (i.e., in the upper levels of domain ontology).

3.2.3. Definition 3 (knowledge concepts, KC)

Knowledge concepts represent the different types of knowledge in domain ontology. Some KCs are general and some are specific. A knowledge concept's generality (or specificity) is determined by its CL.

3.2.4. Definition 4 (base knowledge node, BKN)

A BKN *x* is a 2-tuple <*bid*, *BKC*>, where *bid* is the label of *x* and *BKC* is a set of knowledge concepts. *BKC* of *x* are denoted by $BKC_x = \{c_1, c_2, c_3, \ldots, c_m\}$, where the knowledge concept c_i can be identified in domain ontology and is associated with a corresponding CL.

3.2.5. Definition 5 (base knowledge flow, BKF)

A BKF is a 2-tuple <*BKNS*, *BD*>, where *BKNS* is a non-empty set of BKNs. *BD* is a non-empty set of dependencies. The preceding node and succeeding node of a dependency are in *BKNS*.

3.2.6. Definition 6 (virtual knowledge node, VKN)

A VKN *vx* is a 4-tuple <*vid*, *KNS*, *D*, *VKC*>, where *vid* is the label of *vx*, *KNS* is a non-empty set of BKNs or previously defined VKNs. *D* is a non-empty set of dependencies whose preceding and succeeding nodes are in *KNS*. The *VKC*, abstracted knowledge concept set, is a non-empty set of knowledge concepts defined in domain ontology. The knowledge concepts of *vx* are denoted as $VKC_{vx} = \{vc_1, vc_2, vc_3, \ldots, vc_q\}$, where vc_i is abstracted from some knowledge concepts of *vx*'s member knowledge nodes.

3.2.7. Definition 7 (virtual dependency, VD)

Given a based knowledge flow *BKF* = <*BKNS*, *BD*> and two virtual knowledge nodes *vx* and *vy*, a virtual dependency $vdep(vx, vy)$ from *vx* to *vy* exists if $dep\ (x, y)$ is in *BD*, where *x* is a member of *vx* and *y* is a member of *vy*. A virtual dependency is used to connect two virtual knowledge nodes *vx* and *vy*.

3.2.8. Definition 8 (virtual knowledge flow, VKF)

A virtual knowledge flow is a 2-tuple <*VKNS*, *VDS*>, where *VKNS* is a non-empty set of virtual knowledge nodes; and *VDS* is a non-empty set of virtual dependencies.

Based on the KFV model, KF designers can obtain VKFs from a given BKF by an order-preserving approach and a knowledge concept generalization method. The detailed algorithms and examples of the approach and the method are listed in our previous work [21]. Consequently, different VKFs can be derived to fulfill team members' knowledge-needs in teamwork environments.

4. Introducing roles into the knowledge flow view model

In this section, we extend the knowledge flow view (KFV) model to a role-based KFV model by adding the role aspect. The purpose of the role-based KFV model is to derive role-based virtual knowledge flows (VKFs) from a base knowledge flow (BKF). In the role-based KFV model, virtual knowledge nodes (VKNs) are generated from base knowledge nodes (BKNs) based on the relevance degrees between roles and BKNs. In addition, a concept abstraction method is developed to abstract knowledge concepts of BKNs for a VKN.

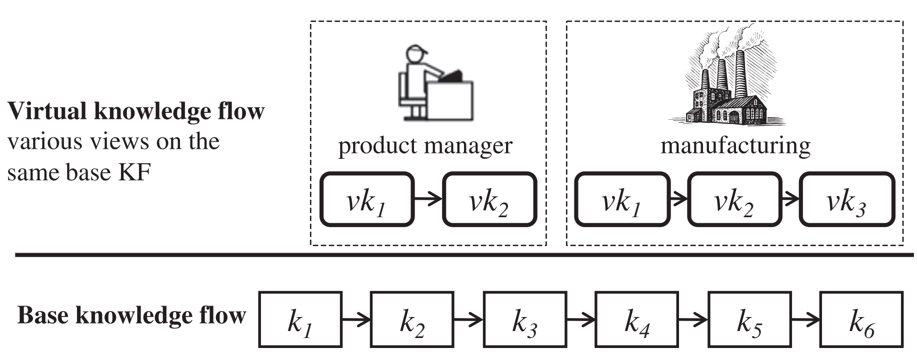

Fig. 3. A BKF derives multiple VKFs.

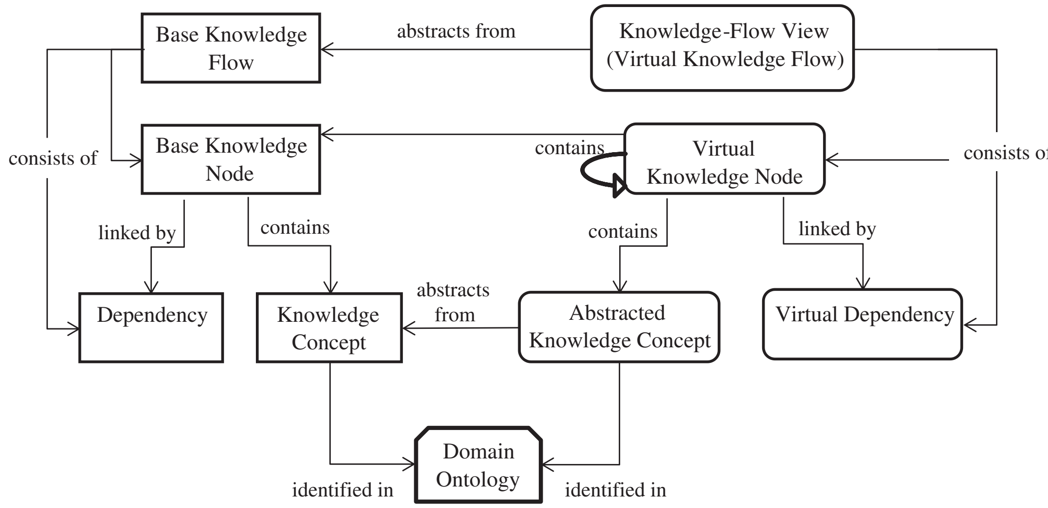

Fig. 4. KFV model.

4.1. Concepts of role-based virtual knowledge flows

In this sub-section, we present the concepts of role-based VKFs. A role-based VKF comprises a set of VKNs that are aggregated from BKNs according to their relevance to a role. Some BKNs may be more relevant to the role than others. A VKN in a role-based VKF denotes a meaningful (aware-needed) knowledge unit of interest to the role; thus, it should be relevant to the role. The process of identifying a role-relevant VKN involves aggregating BKNs based on their relevance to the role. Once the relevance of the aggregated BKNs to the role reaches a certain threshold, a VKN is identified for the role. The relevance of a BKN to a particular role can be specified by KF designers or derived from the relevance degrees between the role and the operations which are associated with the BKNs. The relevance degree of a BKN indicates how important it is to the role. Based on the relevance degrees of all BKNs, procedures can be clearly defined to generate appropriate role-based VKFs for different organizational roles.

After a VKN has been identified, KF designers can derive its knowledge concepts. The objective is to obtain abstractions of the knowledge concepts in the VKN's member knowledge nodes that do not conform to the knowledge required by the role. The knowledge requirements are represented by knowledge concepts which can be identified in domain ontology. Domain ontology is a hierarchical structure comprising knowledge concepts. The lower levels of the hierarchy contain specific knowledge, while the upper levels contain knowledge that is more general. The various roles in an organization have different knowledge requirements. Workers need specific knowledge about their own roles and tasks, but need general knowledge about other roles' tasks. For example, R&D workers design and develop products, so they must have specific technical skills and knowledge. The marketing and sales personnel, who launch and promote products, may not have specific technical knowledge about the products, but they must have general knowledge about the technical aspects.

4.2. Role-knowledge node relevance and knowledge requirement

The crucial part of deriving role-based VKFs is to find the relevance degree between each BKN and each role in a given BKF. Based on the derived relevance degrees, a proposed approach can then generate appropriate VKFs for different organizational roles. The process of generating a role-based VKF incorporates the concept of operations, which is essential in organizational environments. In this sub-section, *base knowledge node profile* and *role-operation relevance profile* are introduced for evaluating role-knowledge node relevance to generate VKNs. In addition, *operation required knowledge concept set* (*KCS*) *profile* and *role-operation knowledge requirement degree* (*krdeg*) *profile* are proposed for deriving roles' knowledge requirements to abstract knowledge concepts for VKNs.

4.2.1. Base knowledge node profile

A BKN is associated with different operations. The base knowledge node profile is a set of 2-tuple <base knowledge node *bkn*, operation *op*>, which expresses that operation *op* is associated with base knowledge node *bkn*.

4.2.2. Role-operation relevance profile

The profile is used to determine how important an operation is to a role, since some operations are more relevant to a role than others. The profile, which records the relevance degree between a role and an operation, is a set of 3-tuple <role *r*, operation *op*, operation relevance degree *ordeg*>. Each 3-tuple records that the operation relevance degree between role *r* and operation *op* is *ordeg*. Roles have different *ordeg* to their operations. The more the relevance between a role and an operation, the higher the *ordeg* will be. The *ordeg* is used as a quantified value to abstract BKNs into VKNs.

For example, suppose a software engineer *r* is assigned to perform an operation *op* called *developing a sales database system*. Because the engineer's job function is system development, the *op* is highly relevant to *r*. Therefore, the operation relevance degree of *op* to *r* is high. The *ordeg* is limited to the range [0, 1]. In this case, the value is between 0.7 and 1.0. The role-operation relevance profile is described as <*r*, *op*, 0.9>.

4.2.3. Operation required knowledge concept set (KCS) profile

The profile comprises a set of 3-tuple <operation *op*, knowledge category *ca*, knowledge concept set *kcs*>, which represents the set of required knowledge concepts *kcs* in knowledge category *ca* for performing operation *op*. The domain ontology is divided into knowledge categories, and an operation may be related to more than one knowledge category. A knowledge concept set (*kcs*) indicates which knowledge concepts in a given knowledge category are required to perform an operation.

For example, according to its operation characteristics, performing a *selling iPad* operation requires specific marketing-related knowledge concepts including *Apple*, *target user*, *benefit* and *discount* and general IT-related knowledge concepts including *requirement*, *design* and *programming*. These knowledge concepts can be mapped to *IT*

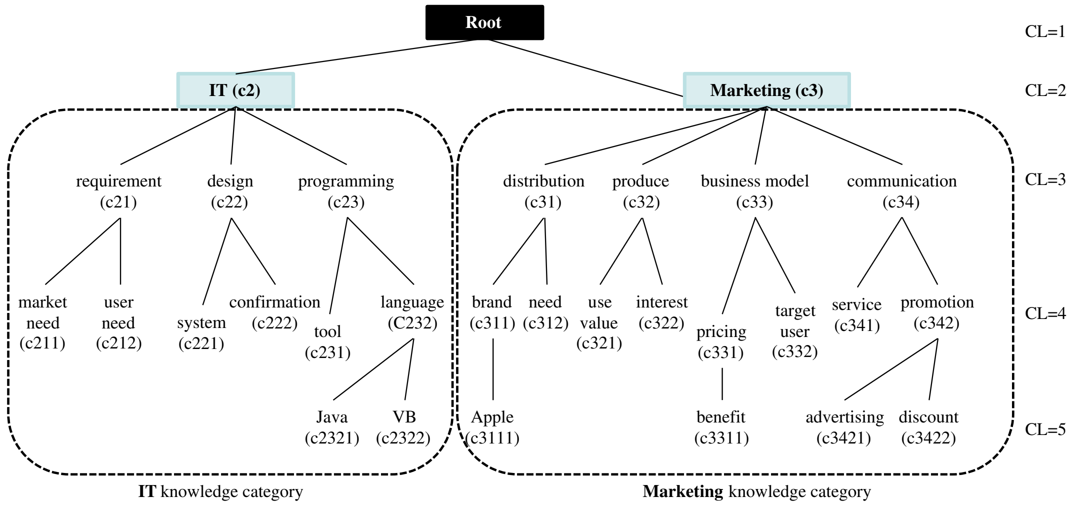

Fig. 5. Partial IT and Marketing knowledge categories in domain ontology.

or *Marketing* knowledge categories, as shown in Fig. 5 where *IT* occupies *c2* and *Marketing* occupies *c3*. Thus, the operation required KCS profiles for the operation *selling iPad* are <*selling iPad*, *IT*, {*c21*, *c22*, *c23*}> and <*selling iPad*, *Marketing*, {*c3111*, *c332*, *c3311*, *c3422*}>.

4.2.4. Role-operation knowledge requirement degree (krdeg) profile

For a given operation, different roles may require different degrees of knowledge in different knowledge categories. The profile is a set of 4-tuple <role *r*, operation *op*, knowledge category *ca*, knowledge requirement degree *krdeg*>. Each tuple indicates that the degree of knowledge required by role *r* with respect to operation *op* in knowledge category *ca* is *krdeg*. The value of *krdeg* is between 0 and 1 and it is used to define how specific or general the required knowledge in a certain knowledge category should be for role *r* while executing operation *op*.

Overall speaking, two phases are required to generate VKFs. Phase I generates VKNs and Phase II abstracts knowledge concepts for these VKNs. In Phase I, role-operation relevance profile, base knowledge node profile and a granular threshold *TH* (a parameter defined in Section 5.1.2) are used to generate VKNs. In Phase II, role-operation krdeg profile and operation required KCS profile are used to abstract knowledge concepts to required concept levels for VKNs. Table 1 shows the profiles and parameters exploited in the role-based KFV model. Items 1, 2 and 3 are related to Phase I and items 4 and 5 belong to Phase II. The profiles are derived by operation log analysis or opinions of domain experts in terms of the involved operations and participating roles in the BKF. Moreover, the parameters can be adjusted by roles and KF designers in terms of roles' requirements.

As mentioned earlier, the operation required KCS profiles record the required knowledge concepts for performing an operation and the role-operation *krdeg* profiles represent roles' knowledge requirement degrees. Thus, KF designers can use *krdeg*, which is estimated by role *r* and KF designers, as a basis to abstract the required knowledge concepts into appropriate concept levels. Role *r* should continue evaluating the fitness level of the abstracted knowledge concepts until they are satisfied with the derived VKFs.

Fig. 6 shows an example to express the two phases for generating role-based VKFs. In Phase I (generate VKN), the relevance degree *ordeg* of role *r* to operation op_1 and op_2 is 0.2 and 0.05 respectively (shown in role-operation relevance profile); op_1 and op_2 are associated with bkn_1 (shown in base knowledge node profile) and the threshold is set to 0.4. In Phase II (abstract knowledge concept), the knowledge concepts required to perform op_1 are {*c131*, *c132*} in *marketing* category and {*c621*} in *hardware* category (shown in operation required KCS profile); the knowledge requirement degree *krdeg* of role *r* to perform op_1 is 0.8 for *marketing* category and 1.0 for *hardware* category (shown in role-operation krdeg profile). The above information is used to generate role-relevant VKNs as well as derive the corresponding knowledge concepts for the VKNs.

Next, the following section discusses how to construct role-operation relevance profile and evaluate the relevance degrees between a given role and a BKN.

4.2.5. Construction of role-operation relevance profile

Initially, KF designers can obtain role-operation relevance profiles by consulting domain experts or by analyzing roles' operating logs through adopting process mining technique [38] or knowledge flow mining methodology [16].

Table 1
List of profiles and parameters.

Phase	Item	Name and definition	Meaning	Building methods
I. Generate VKN	1	Role-operation relevance profile <*r*, *op*, *ordeg*>	Relevance degree (*ordeg*) between *r* and *op*	Analyzing operation logs or KF designers consult domain experts
	2	Base knowledge node profile <*bkn*, *op*>	Associate *bkn* and *op*	KF designers consult domain experts
	3	Granular threshold *TH*	Parameter to determine the granularity of the generated VKNs	Decided by roles and KF designers
II. Abstract knowledge concepts	4	Role-operation krdeg profile <*r*, *op*, *ca*, *krdeg*> includes a parameter *krdeg*	Degree of knowledge required (*krdeg*) by role *r* with respect to operation *op* in category *ca*	Roles and KF designers input *krdeg* in the profile
	5	Operation required KCS profile <*op*, *ca*, *kcs*>	Required knowledge concept set (*kcs*) in category *ca* for performing *op*	KF designers consult domain experts

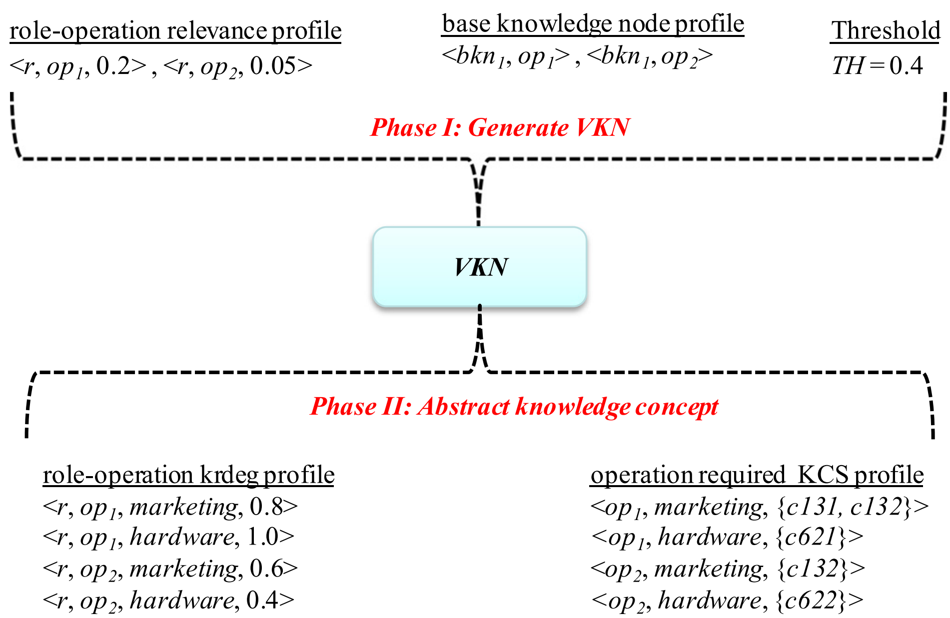

Fig. 6. Two-phase approach to generate role-relevant VKNs.

Let *T* denote the number of times that role *r* performs all of the assigned operations, and let *N* denote the number of times that role *r* performs an operation *op*. The default operation relevance degree of *r* with respect to *op* is *N*/*T*. For example, three operations, *selling iPAD* (op_1), *devising business model* (op_2), and *designing advertisement* (op_3), are assigned to a role *r*, *marketing manager*. The role *r* performs op_1 twice, op_2 seven times, and op_3 once; then *T* equals *10*. The operation relevance degree *ordeg* of *r* is 0.2 for op_1, 0.7 for op_2, and 0.1 for op_3. Intuitively, a higher *ordeg* indicates greater relevance between a role and an operation. Moreover, the higher the cost associated with an operation performed by a role, the higher the relevance degree of the operation to the role will be. Let *Q* denote the total cost of the operations assigned to role *r*, and let *C* denote the cost of a specific operation *op*. The *ordeg* of role *r* when performing operation *op* is *C*/*Q*. Based on activity-based costing (ABC) models, the cost can be measured in terms of the time and resources expended by roles when they perform assigned operations. In summary, different statistics can be extracted from the historical log data. Some decision-making methods can be employed to derive the *ordeg* by combining the statistics.

It is noteworthy, while implementing a role-based KFV system, that the proposed methods for constructing the role-operation relevance profiles need to be fine-tuned in terms of the culture of organizations, the accommodation of peripheral systems and the context of operations. Workers' future operation logs can also be used to adjust *ordeg* to satisfy their real time knowledge-needs. These adjustments are essential to obtain appropriate VKNs.

4.2.6. Evaluation of role-knowledge node relevance

The relevance degree between a given role and a BKN can be derived from *role-operation relevance profile* and *base knowledge node profile*. For example, given a role-operation relevance profile {<*r*, op_1, 0.2>, <*r*, op_2, 0.05>} and a knowledge node profile {<bkn_1, op_1>, < bkn_1, op_2>}, the relevance degree between *r* and bkn_1 is *max* (0.2, 0.05) = 0.2. That is, the relevance degree between *r* and bkn_1 is the largest *ordeg* among all operations related to bkn_1.

5. Discovering role-based virtual knowledge flows

In this section, we propose a role-based approach for discovering virtual knowledge flows (VKF) suitable for participating roles. Generating VKFs involves two phases: Phase I identifies role-relevant virtual knowledge nodes (VKNs); and Phase II derives the knowledge concepts of the identified VKNs.

Phase I is to aggregate the BKNs based on their relevance to a given role. The relevance degree between a BKN and a role, which is derived as discussed in Section 4.2, describes how important the BKN is to the role. Based on the relevance degrees, procedures can be defined to generate appropriate role-based VKFs for different roles. The knowledge concept abstraction process in Phase II obtains the knowledge concepts of VKNs based on *operation required KCS profile* and *role-operation krdeg profile*. An *operation required KCS profile* indicates the specific knowledge concepts required to perform an operation. For a given operation, different roles may require different degrees of knowledge. The *role-operation krdeg profile* indicates how specific the knowledge required for the role will be. The smaller the *krdeg*, the more general the required knowledge will be. The *krdeg* is used to adjust operation required knowledge concepts to a more general concept level in the domain ontology. The abstraction method analyzes the concept levels of required knowledge concepts and abstracts them to a suitable concept level.

In Section 5.1, we propose an approach for identifying role-relevant VKNs. In addition, the corresponding ideas and algorithms for generating role-based VKFs are introduced. We explain how to derive the knowledge concepts of VKNs in Section 5.2.

5.1. Phase I: generate virtual knowledge node

We propose algorithms that generate role-relevant VKNs automatically based on the role-knowledge node relevance degrees described in Section 4.2, which can be derived from *role-operation relevance profile* and *base knowledge node profile*. To obtain VKNs, KF designers take the highest order BKN *x* in the BKF as a seed node to identify a role-relevant VKN by combining adjacent BKNs based on their relevance to the given role. The detail steps are described in the following subsections.

5.1.1. Identifying role-relevant virtual knowledge nodes

A VKN in a role-based VKF denotes a meaningful knowledge unit of interest to the role; thus, it should be relevant to the role. Algorithm *VKNSGenerator* identifies VKNs based on a BKF <*BKNS*, *BD*>. The objective is to discover the VKNs whose relevance degrees approximate a granular threshold, *TH*, which is decided by roles and KF designers.

The algorithm *VKNSGenerator* in Fig. 7 selects the highest order BKN x in the BKF as the seed node to identify a role-relevant VKN. It uses the algorithm *GenerateRVKN* to aggregate the adjacent BKNs based on their relevance to the role. When the total relevance degree of the set of aggregated BKNs approximates *TH*, a VKN is identified for the role and deemed relevant to the role. The above steps are repeated on the remaining BKNs until all of the derived VKNs cover all BKNs of the BKF. The process yields a set of VKNs to build the target VKF. For any pair of VKNs, vx and vy, a virtual dependency $vdep\ (vx, vy)$ exists if $dep(x, y)$ exists in *BD*, where x is a member of vx and y is a member of vy.

Next we explain three important components of the algorithm *VKNSGenerator*: (1) determining the relevance between roles and BKNs, (2) generating a role-relevant VKN by algorithm *GenerateRVKN* and (3) verifying the order-preserving property.

5.1.2. Determining the relevance between roles and base knowledge nodes

From the *base knowledge node profile*, KF designers can determine whether a BKN is associated with certain operations. In addition, the *role-operation relevance profile* indicates the degree of relevance of an operation to a role. The relevance of a BKN to a role can be derived from the *base knowledge node profile* and the *role-operation relevance profile* by applying maximum function over the *ordeg* of the operations associated with the BKN, as described in Section 4.2.

Granular Threshold TH determines the granularity of a generated VKN. After calculating the degree of relevance of each BKN to the role, the algorithm *GenerateRVKN* combines adjacent BKNs to generate a role-relevant VKN. When the sum of the relevance degrees of the set of adjacent BKNs approximates *TH*, the set of adjacent BKNs can form a VKN, which is deemed relevant to the role. A larger *TH* corresponds to the generation of fewer VKNs, which means that more BKNs are included in one VKN.

Total relevance degree F_{TRD}: Let function f_{KRD} (role r, base knowledge node bkn) return the relevance degree of bkn to role r; and let function F_{TRD} (role r, base knowledge node set V) return the total degree of relevance of a virtual knowledge node vkn comprising the base knowledge node set V. $F_{TRD}\ (r, V) = \Sigma f_{KRD}\ (r, bkn_i)$ for all $bkn_i \in V$.

The threshold *TH* in Phase I (Generate VKN) is determined by the roles and KF designers to obtain VKNs with their expected granularity. Abstracting BKNs based on the degrees of relevance of role-knowledge nodes could help to accumulate several less role-relevant BKNs until their total degree of relevance, F_{TRD}, is close to a threshold *TH*. In order to sustain productivity and conduct KM effectively, a role requires greater attention paid to the operations and associated knowledge concepts of BKNs with high degrees of relevance to the role. Roles can omit the specific (detailed) information of BKNs with low degrees of relevance. Accordingly, BKNs with low degrees of relevance to a role are aggregated until they form a role-relevant VKN that is sufficiently relevant to the role. Here, a granular threshold *TH* is set as the criterion to determine sufficient relevance. According to a role-relevant VKN, a role can have a general idea about the BKNs included in the VKN and omit the specific (detailed) information of those BKNs.

For example, given three BKNs: k_1, k_2 and k_3 with relevance degrees 0.1, 0.2 and 0.05 to role r, respectively, role r sets *TH* as 0.4. In other words, role r requests the detail information and knowledge concepts of these less role-relevant BKNs to be ignored until their total degree of relevance, F_{TRD}, is close to 0.4 after abstraction. Hence, KF designers aggregate k_1, k_2 and k_3 to a virtual knowledge node, vkn_1, following the proposed role-based knowledge flow abstraction methods. The degree of relevance between role r and vkn_1 would be 0.35, which is close to 0.4. After the abstraction, the virtual knowledge node, vkn_1, deserves role r's awareness.

5.1.3. Virtual knowledge node generation algorithm

The algorithm *GenerateRVKN* generates a VKN, which contains a given seed node, by repeatedly aggregating the adjacent BKNs according to the order of their relevance degrees sorted from high to low. Note that the aggregation of BKNs needs to satisfy the order-preserving property; this will be illustrated in Section 5.1.4. The process is repeated until the total relevance degree of the aggregated BKNs approximates *TH*. The steps of the algorithm are detailed in Fig. 8.

For a virtual knowledge node $vkn = <vid, V, D, VKC>$, the members of V must be identified first. Initially, V only contains the given seed knowledge node bkn. The algorithm then determines whether the BKNs which are adjacent to members of V can be added to maximize its total relevance degree $F_{TRD}\ (r, V)$ so that it approximates *TH*. V is updated during the *while* loop by adding the adjacent BKNs that satisfy three conditions: (1) V conforms to the order-preserving property; (2) the total relevance degree of V does not exceed the threshold ($F_{TRD}\ (r, KNS_{tmp}) \leq TH$); and (3) V does not overlap with previously derived VKNs ($KNS_{tmp}\ RKNS$). The *repeat-until* loop continues until no other adjacent BKNs can be added to V, i.e., no more adjacent BKNs can be added to V, which still satisfy the threshold limit and maintain order-preserving property.

5.1.4. Verification of the order-preserving property

Liu and Shen [22] developed an order-preserving approach that generates virtual processes from a base process in workflow environments.

Algorithm *VKNSGenerator*: generate a set of VKNs
  input: a base knowledge flow $BKF = <BKNS, BD>$
  output: the virtual knowledge node set (*VKNS*) of a virtual knowledge flow $VKF = <VKNS, VDS>$
begin
  i=1
  repeat
    Create a new virtual knowledge node $vkn_i = <vid_i, KNS_i, D_i, VKC_i> \leftarrow <i, \varnothing, \varnothing, \varnothing>$
    Residual Knowledge Node Set $RKNS \leftarrow BKNS - \{x \mid x$ belongs to one of KNS_i, for any $i\ \}$
    Select the highest order base knowledge node x from *RKNS*
    $vkn_i \leftarrow GenerateRVKN(i, x, RKNS, BKF)$
    $VKNS = VKNS \cup \{vkn_i\}$
    $i = i+1$
  until $\forall\ x \in BKNS$, x belongs to one of KNS_i, for any i
  return *VKNS*
end

Fig. 7. Algorithm for generating a set of VKNs.

```
Algorithm GenerateRVKN: generate a VKN whose total relevance degree approximates TH
  input: label i, seed knowledge node bkn, residual knowledge node set RKNS, BKF = <BKNS, BD>
  output: a virtual knowledge node vkn
begin
  vkn = ⟨vid, V, D, VKC⟩ ← ⟨i, {bkn}, ∅, ∅ ⟩
  repeat
    Temp Knowledge Node Set TKNS ← V
    Adjacent Knowledge Node Set AKNS ← { x | x, y ∈ RKNS, x ∉ V, y ∈ V, and dep(x, y) ∈ BD }
    while AKNS is not empty do
      select a BKN x from AKNS
      remove x from AKNS
      KNS_tmp ← OderPrsv(V ∪ {x}, BKF) /* check order-preserving property */
      if ( F_TRD (role r, KNS_tmp) ≤ TH ) and (KNS_tmp ⊆ RKNS ) then   /* check threshold */
        V← KNS_tmp
        AKNS ← AKNS − {y | y ∈ AKNS ∩ V}
      end if
    end while
  until V = TKNS
  A dependency set D ← { dep(x, y) | x, y ∈ V, and dep(x, y) ∈ BD }
  VKC = KCGEN-OKR (role r, virtual knowledge node vkn)
  return vkn = ⟨vid, V, D, VKC⟩
end
```

Fig. 8. Algorithm for generating a role-relevant VKN.

The approach ensures that the generated virtual process satisfies the following order-preserving property: *the implied ordering relations of activities in a virtual process must comply with the ordering relations of activities in the base process*. The generation algorithm *GenerateRVKN* adopts the order-preserving approach to ensure that a VKF maintains the knowledge referencing order in its corresponding BKF. A legal virtual knowledge node *vkn* = ⟨*vid, V, D, VKC*⟩ must satisfy the following order-preserving property: *the implied ordering relations of VKNs in a VKF must comply with the ordering relations of the BKNs in its corresponding BKF*. The order-preserving property is satisfied when the ordering relations between any BKN $x \notin V$ and all member knowledge nodes in *V* are identical as their ordering relations in the corresponding BKF. Regarding the function *OderPrsv* in algorithm *GenerateRVKN* for adopting the order-preserving approach, please refer to our previous work [19,21,22] for the detailed steps and examples.

5.1.5. Illustrating examples

Given a base knowledge node bkn_1 and its adjacent base knowledge nodes, bkn_2 and bkn_3, we explain how the algorithm generates a virtual knowledge node vkn_1.

Example 1. The role-knowledge node relevance degree is derived from the following profiles. (A) *Base knowledge node profile*: {<bkn_1, op_1>. <bkn_1, op_2>, <bkn_1, op_5>, <bkn_1, op_7>}, which indicates that operations op_1, op_2, op_5 and op_7 are associated with bkn_1. (B) *Role-operation relevance profile*: {<r_1, op_1, 0.6>, <r_1, op_2, 0.3>, <r_1, op_5, 0.4>, <r_1, op_7, 0.1>}, which indicates the relevance degrees between r_1 and the operations. Based on the above information, the maximum function is used to obtain the relevance degree between bkn_1 and r_1: *Max* (0.6, 0.3, 0.4, 0.1) = 0.6

Example 2. A VKN is generated from the following profiles. (A) *Base knowledge node profile*: {<bkn_2, op_3>, <bkn_2, op_5>, <bkn_3, op_4>, <bkn_3, op_6>}, which indicates bkn_2 is associated with op_3, op_5, and bkn_3 is associated with op_4, op_6. (B) *Role-operation relevance profile*: {<r_1, op_3, 0.2>, <r_1, op_5, 0.3>, <r_1, op_4, 0.1>, <r_1, op_6, 0.2>}. The profile indicates the relevance degrees between r_1 and the operations associated with bkn_2 and bkn_3. Based on the above information, the maximum function is used to obtain the relevance degrees of bkn_2 and bkn_3 for r_1. *Max* (0.2, 0.3) = 0.3 for bkn_2; *Max* (0.1, 0.2) = 0.2 for bkn_3. The relevance order of the adjacent knowledge nodes of bkn_1 is bkn_2 followed by bkn_3.

Taking the order structure into consideration, bkn_2 and bkn_3 are candidate nodes to be combined with bkn_1. Assuming that the threshold *TH* is 1, the relevance degrees for bkn_2 and bkn_3 are 0.3 and 0.2, respectively. According to the function F_{TRD}, the two nodes cannot be combined with bkn_1 together to form a VKN because $f_{KRD}(bkn_1)+f_{KRD}(bkn_2)+f_{KRD}(bkn_3)>1$. However, the total relevance degree $f_{KRD}(bkn_1)+f_{KRD}(bkn_2)\leq 1$, which is not greater than *TH*. Thus, bkn_1 is combined with bkn_2 to form a virtual knowledge node vkn_1.

5.2. Phase II: abstract knowledge concept

After a virtual knowledge node *vkn* has been generated, its knowledge concepts can be derived. Fig. 9 shows the algorithm *KCGEN-OKR* that derives the knowledge concepts of *vkn* based on the *operation required knowledge concept set* (*KCS*) *profiles* and the *role-operation knowledge requirement degree* (*krdeg*) *profiles*.

We explain how to derive the required KCS and corresponding *krdeg* for operations that the role performs on a VKN. The operation required knowledge concepts indicate the most specific level of knowledge in the domain ontology that are required to perform operations. Different roles may require different degrees of knowledge to perform operations. In this work, we use the knowledge requirement degree *krdeg* as the basis to abstract the operation required knowledge concepts into the appropriate concept levels CLs) in order to satisfy a role's operation knowledge requirement. The virtual knowledge node *vkn* may include more than one BKN, so the knowledge concepts of all BKNs should be abstracted to derive the knowledge concepts of *vkn*. The knowledge concept abstraction process involves two steps. The first step derives the required knowledge concept set (*KCS*) and corresponding knowledge requirement degrees (*krdeg*). The second step

Algorithm *KCGEN-OKR*: generalize knowledge concepts based on *operation required KCS profile* and *role-operation krdeg profile*

input: role *r*, a virtual knowledge node *vkn* = <*vid*, *KNS, D, VKC*>

output: *VKC* of *vkn*

begin

CRD_r^{vkn} = {} /* is a set of 3-tuple <concept *c*, category *ca*, knowledge requirement degree *krdeg*> */

OP_r^{vkn} is a set of operations associated with *vkn* and performed by role *r*

/* Step 1: derive required *KCS* and corresponding *krdeg* for each concept in *KCS**/

for each operation *op* in OP_r^{vkn} ***do***

KCS_{op}^{ca} is a set of knowledge concepts in category *ca* required to perform *op*

for each knowledge category *ca* related to operation *op* ***do***

for each concept *c* in KCS_{op}^{ca} and its associated role-operation knowledge req. degree $krdeg_{r,op}^{ca}$ ***do***

if concept *c* already exists in CRD_r^{vkn} ***then***

$krdeg_{r,c}^{ca}$ = the knowledge requirement degree of concept *c* in CRD_r^{vkn}

$CRD_r^{vkn} = max\ (krdeg_{r,c}^{ca}, krdeg_{r,op}^{ca})$

else

$krdeg_{r,c}^{ca} = krdeg_{r,op}^{ca}$

end if

$CRD_r^{vkn} = CRD_r^{vkn}$ ∪ {<concept *c*, category *ca*, $krdeg_{r,c}^{ca}$ >}

end for

end for

end for

/* Step 2: adjusts (generalizes) knowledge concepts to appropriate concept levels */

for each tuple <concept *c*, category *ca*, knowledge requirement degree $krdeg_{r,c}^{ca}$ > in CRD_r^{vkn} ***do***

gc = *GenConcept*(*c, ca,* $krdeg_{r,c}^{ca}$)

GKCS = *GKCS* ∪ {*gc*} /* *GKCS*: generalized knowledge concept set */

end for

for *each knowledge concept c_i in GKCS* ***do***

if *every path from c_i to leaf nodes in the domain ontology* ∃ c_j (c_i is not included) ∈ *GKCS* ***then***

remove c_i from *GKCS*

end if

end for

VKC of vkn = *GKCS*

end

Fig. 9. Algorithm for generalizing knowledge concepts of a VKN.

adjusts (generalizes) the knowledge concepts in *KCS* to the appropriate concept level according to the corresponding *krdeg* of the role.

An operation required KCS profile specifies the knowledge concepts required in different knowledge categories to perform an operation. The domain ontology is divided into different knowledge categories; an operation may be associated with more than one knowledge category. From the operation required KCS profile, the specific knowledge concepts required for each operation in each knowledge category are obtained. For a given operation, different roles may have different degrees of knowledge requirements in different knowledge categories. The role-operation *krdeg* profile is used to define how specific or general the required knowledge in a certain knowledge category should be for role *r* when it performs an operation. From the operation required KCS profile and role-operation *krdeg* profile, KF designers can generate the set of required knowledge concepts and their corresponding knowledge requirement degrees for the role to perform the operations associated with *vkn*. A role may have different knowledge requirement degrees to perform different operations that need the same knowledge concepts. Therefore, a knowledge concept required by a role may be associated with more than one *krdeg*. The maximum function is used to derive the final krdeg of a concept for the role.

The first step in algorithm *KCGEN-OKR* is to derive required *KCS* and corresponding *krdeg*, which uses several working sets and variables for computation, including: (1) OP_r^{vkn} is a working set of operations. The operations are associated with *vkn* and performed by *r*. Thereby; *vkn* comprises a set of merged BKNs. (2) KCS_{op}^{ca} is a working set of knowledge concepts. The knowledge concepts belong to knowledge category *ca* and they are required to perform operation *op*. (3) $krdeg_{r,op}^{ca}$ is a working variable of *r*'s knowledge requirement degree. The *krdeg* is related to knowledge category *ca* and operation *op*. (4) CRD_r^{vkn} is a working set associated with *r* and *vkn*. It is a 3-tuple <concept *c*, category *ca*, knowledge requirement degree *krdeg*>, which indicates the required *krdeg* of *c* in *ca*. CRD_r^{vkn} is derived from OP_r^{vkn}, KCS_{op}^{ca} and $krdeg_{r,op}^{ca}$ as detailed in algorithm *KCGEN-OKR*.

In algorithm *KCGEN-OKR*, the second step is to adjust (generalize) the knowledge concepts to the appropriate concept level according to *krdeg*. For each knowledge concept in CRD_r^{vkn} <concept *c*, category *ca*, knowledge requirement degree *krdeg*>, *krdeg* determines how many levels of the original concept *c* should be abstracted. The value of *krdeg* is between 0 and 1; the larger the value, the more specific the knowledge concepts that are required; conversely, the smaller the value, the more general the knowledge concepts that are required. If *krdeg* is 1.0, the most specific level of knowledge concept is required, so there is no need to perform abstraction. A function $GenConcept(c, ca, krdeg_{r,c}^{ca})$ is used to adjust the concept *c* in category *ca* to an appropriate concept level according to $krdeg_{r,c}^{ca}$. For example, if $krdeg_{r,Apple}^{Marketing} = 0.8$, *GenConcept* (*Apple*, *marketing*, 0.8) returns the parent knowledge concept *brand* of knowledge concept *Apple* in category *marketing*. According to Fig. 5, the concept level (CL) of *brand* is 4 and the CL of *Apple* is 5. That is, *GenConcept* (*Apple*, *marketing*, 0.8) executes one-level up abstraction of *Apple* to get generated concept (gc) through the domain ontology *as shown in* algorithm *KCGEN-OKR*. If $krdeg_{r,c}^{ca} = 0.6$, *GenConcept*(*c*, *ca*, 0.6) returns the grandparent knowledge concept of concept *c* in category *ca* by executing two-level up abstraction through the domain ontology. If there are insufficient levels in the domain ontology can be obtained when executing the function *GenConcept*, the concept *c* is abstracted to the most general concept level. If $krdeg_{r,c}^{ca} = 1.0$, no abstraction is needed; *GenConcept*(*c*, *ca*, 1.0) returns the concept *c*. It is notable that the mapping between the value of *krdeg* (0.8, 0.6, …) and the levels of abstraction (one-level up abstraction, two-level up abstraction,…) can be adjusted by KF designers depending on applications.

The function *GenConcept*(*c*, *cl*, *acl*) is used to adjust concept *c* to an appropriate concept *gc*. Let *GKCS* (generalized knowledge concept set) denote the set of knowledge concepts for *vkn*. For each knowledge concept *c* of the member knowledge nods in *vkn*, *GKCS* is added with the generalized knowledge concept *gc*, which is derived from concept *c*, to meet *r*'s knowledge requirement. The final step of the procedure removes the implied (redundant) concepts from *GKCS* and yields the knowledge concepts *VKC* of *vkn*.

Example 3. The knowledge concepts of *vkn*, which combines bkn_1 and bkn_2, are derived as follows. (A) *Base knowledge node profile*: {<bkn_1, op_1>, <bkn_1, op_2>, <bkn_2, op_3>}. (B) *Operation required KCS profiles*: <op_1, *Marketing*, {*Apple* (*c3111*), *benefit* (*c3311*), *discount* (*c3422*)}>, <op_1, *IT*, {*Java* (*c2321*), *user need* (*c212*)}>, <op_2, *Marketing*, {*need* (*c312*), *Apple* (*c3111*), *discount*(*c3422*)}>, <op_3, *IT*, {*Java* (*c2321*), *system* (*c221*)}>. These knowledge concepts are in different knowledge categories as shown in Fig. 5. (C) *Role-operation krdeg profiles*: <*r*, op_1, *Marketing*, 0.8>, <*r*, op_1, *IT*, 0.6>, <*r*, op_2, *Marketing*, 0.6>, <r_1, op_3, *IT*, 0.4>.

Investigating the *operation required KCS profiles* in Example 3, op_1 and op_2 require the same knowledge concepts *Apple* and *discount* in *Marketing* knowledge category. Moreover, op_1 and op_3 require the same knowledge concept *Java* in *IT* knowledge category. Thus, the *krdeg* of knowledge concept *Apple*, *discount* and *Java* should be preprocessed by a maximum function before abstraction. In *Marketing* knowledge category, the *krdeg* of *Apple* and *discount* for op_1 is 0.8, for op_2 is 0.6. After applying the maximum function *Max* (0.8, 0.6), the final *krdeg* of *Apple* and *discount* is 0.8, which means that the two knowledge concepts should be abstracted by one-level up to their parent knowledge concepts *brand* and *promotion* respectively, according to the domain ontology in Fig. 5. In *IT* knowledge category, the *krdeg* of *Java* for op_1 is 0.6, for op_3 is 0.4. Thus, the final *krdeg* of *Java* is *Max* (0.6, 0.4) = 0.6. Thus, the knowledge concept *Java* should be abstracted by two-level up to knowledge concept *programming* as well.

6. Designing a role-based KFV system

We analyze the proposed model and concepts to conduct the basic system design of a role-based KFV system. The basic system design constructs an overall architecture by identifying important functional modules and decomposing them into layers. An activity diagram models the procedural flow of actions from the perspectives of KF designers, roles and experts. Moreover, Protégé 4.0 software is used to build an ontology prototype to represent knowledge concepts and their hierarchical relationships in the mobile phone development domain. The system architecture, activity diagram and ontology prototype are the fundamental elements for putting the theoretical model into practice.

Fig. 10 depicts the system architecture to implement the role-based KFV system, which comprises four layers: data link layer, configuration layer, modeling layer and application layer.

6.1. Data link layer

This layer enables data links to other legacy systems to collect information from external data sources during the design time and run time. The operation logs preserve the history of roles' operations in workflow management systems. The knowledge databases store codified knowledge which is labeled by the knowledge concepts of domain ontology. Enterprise knowledge-based management systems can be utilized to manage these knowledge databases and provide interfaces for the role-based KFV system to access required codified knowledge. Domain ontology stores the pre-defined knowledge concepts and their hierarchical relationships for the purpose of representing knowledge-needs and facilitating knowledge concept abstraction.

6.2. Configuration layer

This layer comprises three parts: profile management module, BKF and role-based VKF repository as well as relevance degree calculating engine. KF designers and experts utilize the profile management module to collect essential information, such as role-operation relevance profiles, operation required KCS profiles and base knowledge node profiles. The BKF and role-based VKF repository preserves model definitions and enactment instances. The role-knowledge node relevance degree calculating engine is responsible for obtaining the relevance degrees between roles and BKNs.

6.3. Modeling layer

This layer includes three definition tools to define BKFs, role-based VKFs and roles' knowledge requirement degrees. Roles use the knowledge degree definition tool to set role-operation krdeg profiles to reflect roles' *krdeg* based on their knowledge-needs. Meanwhile, KF designers work with experts to specify BKFs and corresponding role-based VKFs by BKF and role-based VKF definition tools, respectively. Moreover, the algorithms shown in Figs. 7, 8 and 9 for generating role-based VKFs will be realized in the definition tools.

6.4. Application layer

An integrated platform is built in this layer for the operations of KF designers, experts and roles. This layer mainly provides an interface for them to get visualization support and maintain profiles.

We produced the activity diagram of the role-based KFV system, as shown in Fig. 11. The actors in the activity diagram are KF designers, roles and experts. They perform these procedural activities and produce relevant material for generating role-based VKFs. The activity diagram is organized into three partitions to indicate the major responsible person of the activities of each partition. The rounded rectangles represent the activities which are performed by actors manually or executed by the tools of the role-based KFV system. And the rectangles show the material such as profiles, parameters or intermediate output that passed between activities. We omitted the detailed explanation of the activity diagram here because

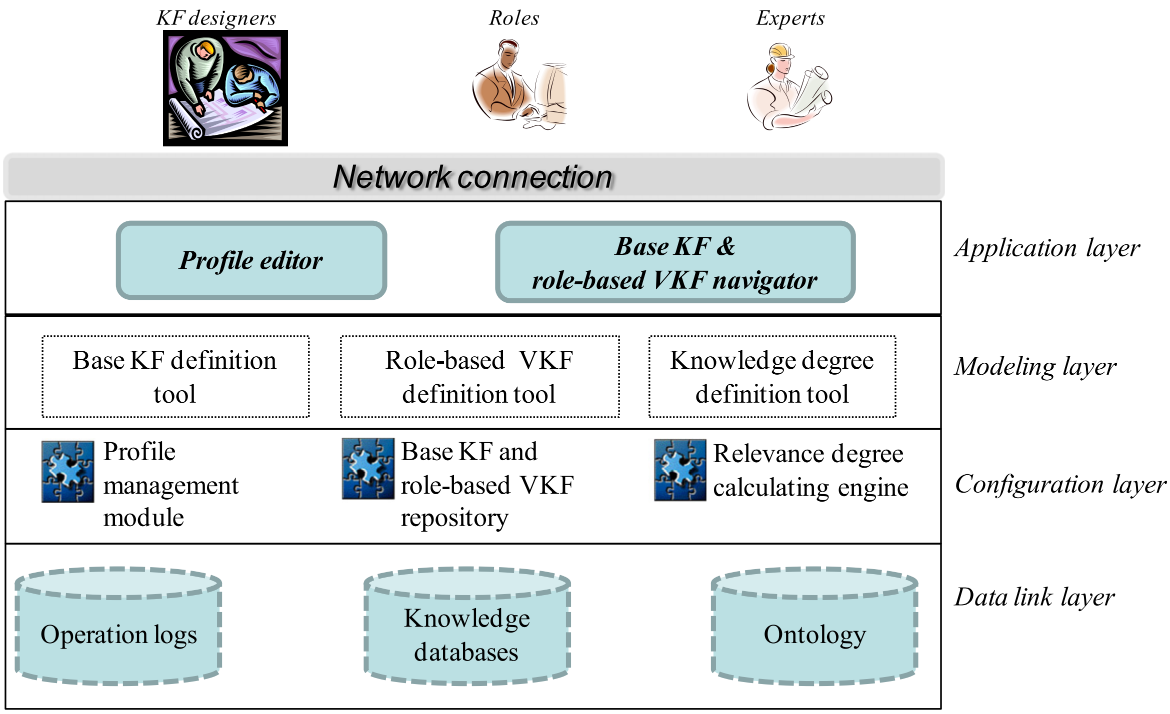

Fig. 10. System architecture of the role-based KFV system.

it is somewhat self-explanatory. The activity diagram is useful for system modeling to describe the control flow of the role-based KFV system, such as exploring the knowledge concept abstraction and knowledge node generation approaches, as well as parameter evaluation and adjustment methods.

An approach of iterative evaluation and adjustment is adopted in the design to fine-tune profiles and parameters to improve the fitness level of VKFs. At the bottom of the left partition in the activity diagram, role *r* should evaluate the fitness level of the generated VKF. If role *r* is satisfied with the VKF, the procedure stops. Otherwise, role

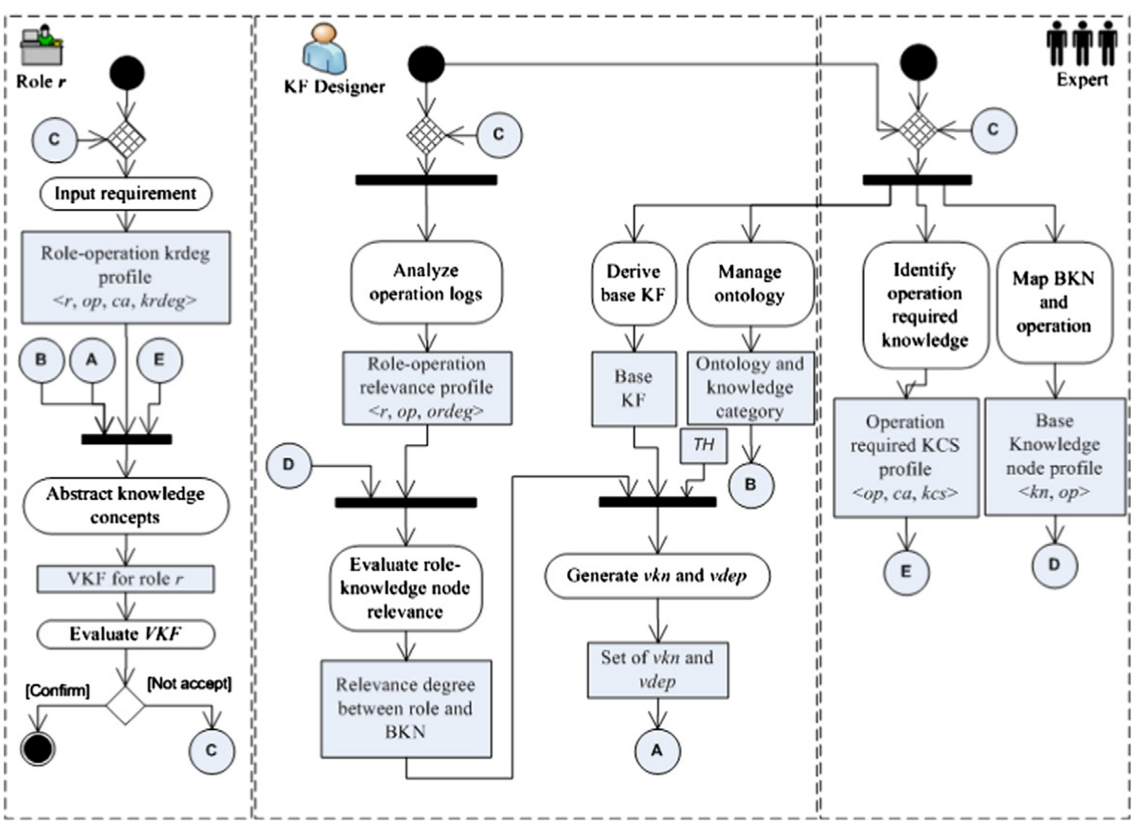

Fig. 11. Activity diagram.

r would reflect the discrepancies of current VKF to KF designers and/or experts. Then, they may adjust BKF, ontology, *krdeg*, or *TH* and estimation methodologies, and regenerate VKFs again until role r satisfies the result. For example, the iteration of *krdeg*'s adjustment is shown by the merging of a start point (●) and the label © in the top of the left partition in Fig. 11.

The role-based concept has been adopted in many workflow related studies to reduce the impact of people turnover. Hence, the role-based VKFs are generated from a role perspective instead of an individual user perspective. The knowledge concepts in a VKN are the required knowledge concepts for roles to perform corresponding operations and communicate with other teammates. In case, some certain users playing the roles may already have the knowledge contained in the VKN or they cannot be satisfied with the knowledge concepts of the VKN. The certain users can fine-tune *krdeg* to adjust the knowledge concepts in the VKN to meet their requirements afterwards.

We used Protégé 4.0 software to build an ontology prototype, as shown in Fig. 12, to represent a part of knowledge concepts and their hierarchical relationships in mobile phone development domain. The process of ontology construction would include an evaluation and feedback mechanism to gradually improve ontology quality and result in a common understanding in organizations. The ontology prototype is appropriate for system designers to understand the concepts of ontologies in system design phase. Since new knowledge is continually generated with changes in technology and business environments, KF designers should periodically maintain domain ontologies and BKFs to check if any knowledge concepts should be added to or removed from BKNs. After the adjustment of BKFs, KF designers can regenerate new VKFs per roles' demands.

7. Case illustration and discussion

This section uses a base knowledge flow (BKF) of a mobile phone company, named Smart-Tech Company, to illustrate the application of the role-based KFV model. The BKF represents the knowledge which a project team requires while conducting a mobile phone development process. KF designers build the BKF based on the mobile phone development process, as shown in Fig. 1. They consult domain experts to identify important knowledge-needs and seek proper knowledge concepts in domain ontology for the purpose of building the BKF to represent these knowledge-needs. The participants of the mobile phone development team work for different departments and play different roles in the team. For example, a sourcing planner role performs the logistics of parts outsourcing and a sourcing department manager role evaluates project performance and the sourcing planner role's productivity. The sourcing department manager role is responsible for communicating with the project manager about the project status and outsourcing strategy, as well as appraising the performance of the sourcing planner role. Therefore, KF designers can generate a role-based VKF from the BKF to represent the sourcing department manager role's knowledge-needs to support task execution. The following discussion illustrates the process used to generate the role-based VKF. First, the BKF and the relevance degrees between the sourcing department manager role and the BKNs are obtained by the approach described in Section 4.2. The BKF in Fig. 13 includes eight BKNs, k_0 to k_7. Each BKN contains multiple knowledge concepts and has distinct operation relevance degrees to r, herein; r stands for the sourcing department manager role.

Next, KF designers generate virtual knowledge nodes (VKN) based on a threshold (*TH*) 0.4. They select a BKN with the highest order, k_0, as a seed node to generate the first virtual knowledge node vkn_1. The adjacent base knowledge node set *AKNS* of k_0 is $\{k_1\}$. So, vkn_1 is considered a legal VKN having two member knowledge nodes k_0, k_1. After checking F_{TRD} $(r, \{k_0, k_1\})$, KF designers can find $F_{TRD}(r, \{k_0, k_1\}) = f_{KRD}(r, k_0) + f_{KRD}(r, k_1) = 0.1 \leq TH$, which meet the threshold requirement. And vkn_1 also complies with the order-preserving property. So, vkn_1 is a legal VKN when it has two member KNs k_0, k_1. Because F_{TRD} $(r, \{k_0, k_1\})$ is not approximately close to *TH*, KF designers continue to evaluate *AKNS* of $\{k_0, k_1\}$, which is $\{k_2, k_5\}$. According to the order-preserving property, we should add BKNs k_2, k_3, k_4, k_5 and k_6

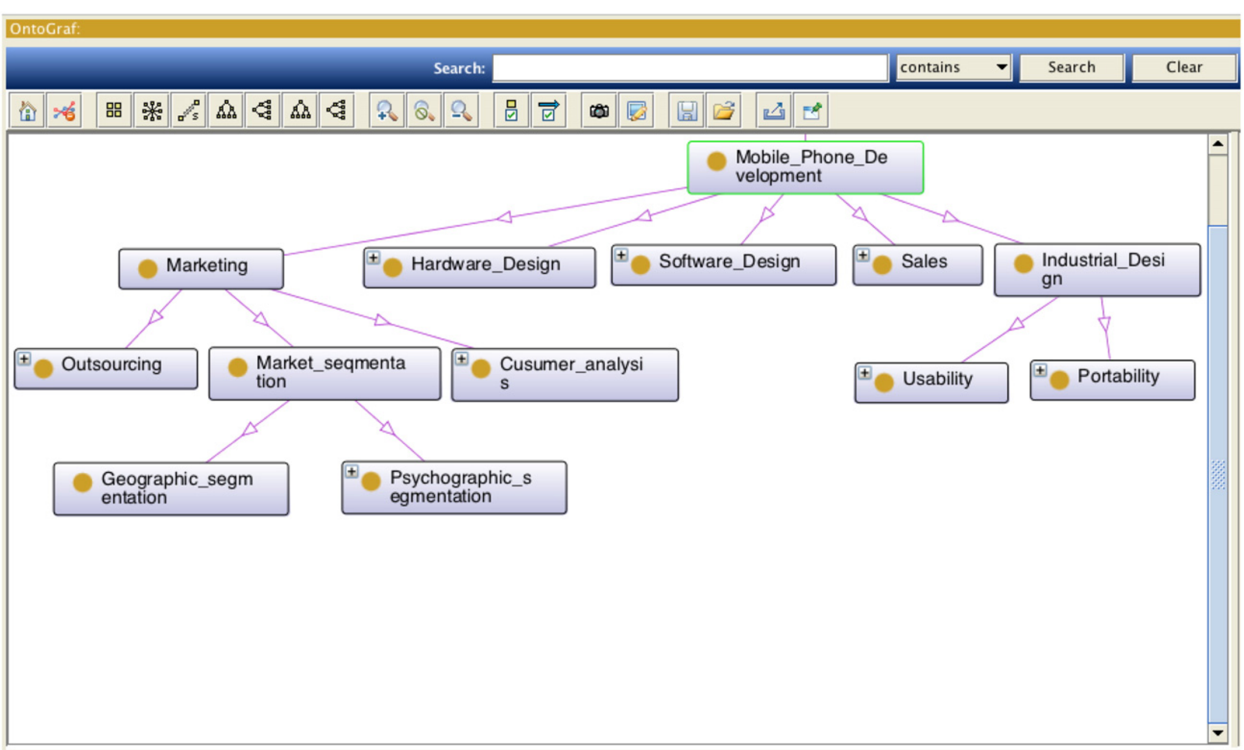

Fig. 12. An ontology prototype.

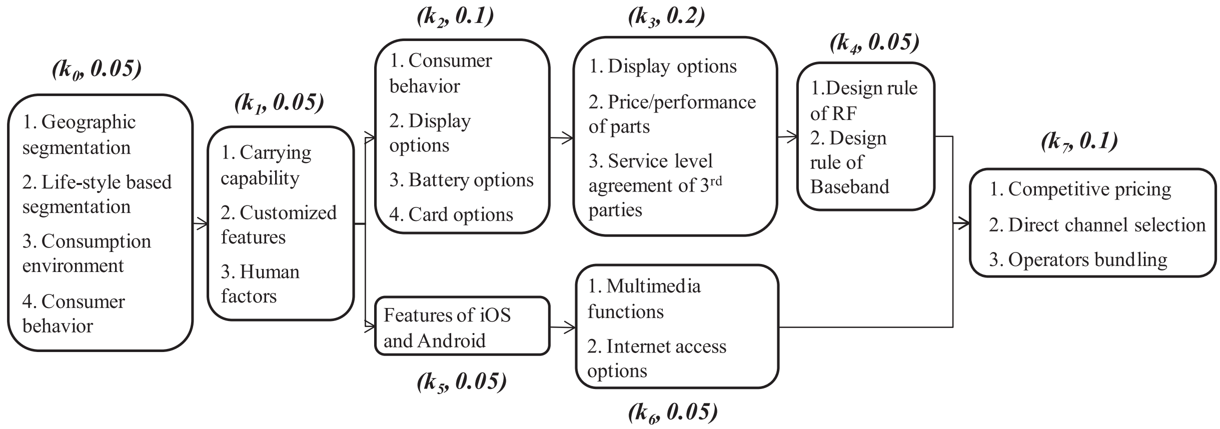

Fig. 13. Operation relevance degrees between role *r* and BKNs.

into vkn_1. However, F_{TRD} $(r, \{k_0 \ldots k_6\}) = \sum f_{KRD}$ (r, k_j) $\{j = 0 \ldots 6\} =$ 0.55 exceeds *TH*. Therefore, the first virtual knowledge node vkn_1 and its member knowledge nodes, k_0 and k_1, are determined. Repeating the iteration, the BKNs are merged into four virtual knowledge nodes vkn_1, vkn_2, vkn_3 and vkn_4 to form a role-based VKF, as shown in Fig. 14.

The following discussion takes virtual knowledge node vkn_2 as an example to illustrate the concept abstraction method. First, KF designers and role *r* set *r*'s knowledge-needs in terms of *krdeg* by different knowledge categories. A partial domain ontology which includes five knowledge categories, such as: *Marketing*, *Industrial design*, *Hardware design*, *Software design* and *Sales*, is illustrated in Fig. 15. It is to be noted that the partial domain ontology is only used for concept explanation and case illustration here, instead of implementing a role-based KFV system in an organization. Ontology construction for organization use is a complex task and needs to further consider users' requirements, IT environments and the context of applications. This type of ontology is much more complete and complex than the partial domain ontology in Fig. 15.

Based on the two-phase approach for generating role-based VKFs shown in Fig. 6, KF designers, experts and role *r* determine the relevant profiles and parameters as shown in Fig. 16. According to the algorithm in Fig. 9, the knowledge requirement degree $krdeg_{r,c}^{ca}$, which represents the knowledge requirement degree of role *r* in regard to the knowledge concept *c* in the knowledge category *ca*, can be obtained as below through computing the operation required KCS profiles and the role-operation krdeg profiles.

$krdeg_{r,customerbehavior}^{marketing} = 0.6$ $\quad krdeg_{r,card\ options}^{Hardware\ Design} = 0.6$
$krdeg_{r,battery\ options}^{Hardware\ Design} = 0.6$ $\quad krdeg_{r,Baseband\ design\ rule}^{Hardware\ Design} = 0.6$
$krdeg_{r,RF\ design\ rule}^{Hardware\ Design} = 0.6$ $\quad krdeg_{r,service\ level\ agreement\ of\ thirdparties}^{marketing} = 0.8$
$krdeg_{r,price/performance\ of\ parts}^{marketing} = 0.8$ $\quad krdeg_{r,display\ options}^{Hardware\ Design} = 1.0.$

Finally, the knowledge concepts are generalized to required concept levels. Consequently, the knowledge concepts of virtual knowledge node vkn_2 are *marketing*, *outsourcing*, *hardware design* and *display options*. The same method can apply to other VKNs. Fig. 17 shows the final result of the role-based VKF for sourcing department manager role.

7.1. Simulation by examples

As shown in Table 1, two phases are required to generate VKFs. Phase I generates VKNs and Phase II derives knowledge concepts for these VKNs. The profiles, including role-operation relevance profile and base knowledge node profile in Phase I, role-operation krdeg profile and operation required KCS profile in Phase II, are predetermined by KF designers according to the involved operations and the participating roles in the BKF. The parameters, including threshold *TH* in

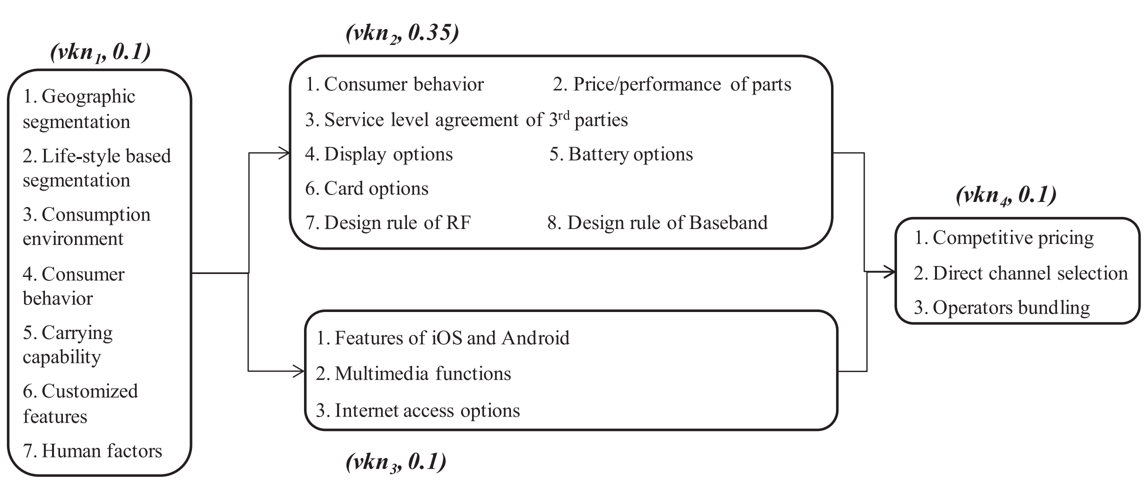

Fig. 14. VKNs and their relevance degrees when $TH = 0.4$.

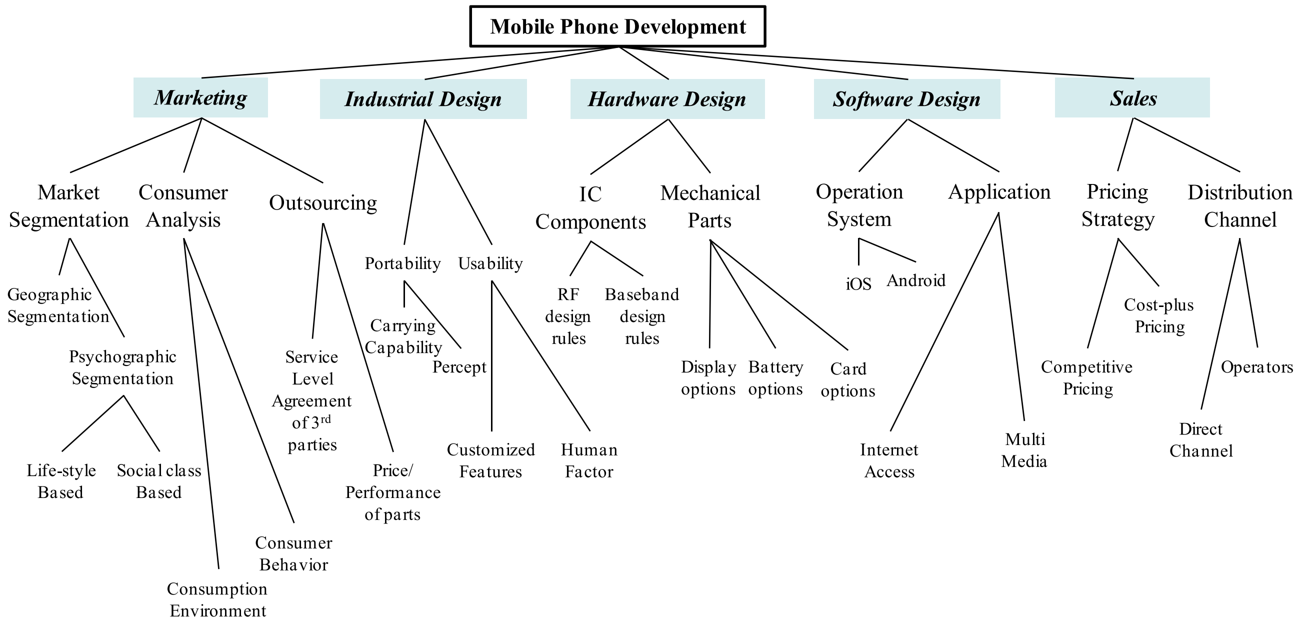

Fig. 15. Partial knowledge categories of Marketing and Hardware Design.

Phase I and knowledge requirement degree *krdeg* in Phase II, can be adjusted by roles and KF designers to reflect roles' requirements. The simulation by examples is performed to evaluate these two parameters, which is based on the BKF shown in Fig. 13.

In Phase I, the granular threshold *TH* is a parameter to determine the granularity of the generated VKNs. Fig. 18 shows the simulation result when the threshold *TH* increases from 0.1 to 0.7, which illustrate different level of aggregate abstraction of the BKF. The different level of aggregate abstraction can provide appropriate VKFs for concealing the detail structure of the corresponding BKFs to reduce the complexity and improve comprehension for roles. As shown in Fig. 18, larger *TH* deriving fewer VKNs can conceal the more detailed structure of BKFs. On the other hand, the smaller *TH* deriving more VKNs can illustrate the details of BKFs.

In Phase II, the *krdeg* is a parameter for roles to set the expected concept levels (CLs) of knowledge concepts in VKNs. Fig. 19(a) takes the vkn_2 in Fig. 18(c) as an example. The *TH* of the corresponding VKF is 0.4 and the knowledge concepts in the VKF can be found in

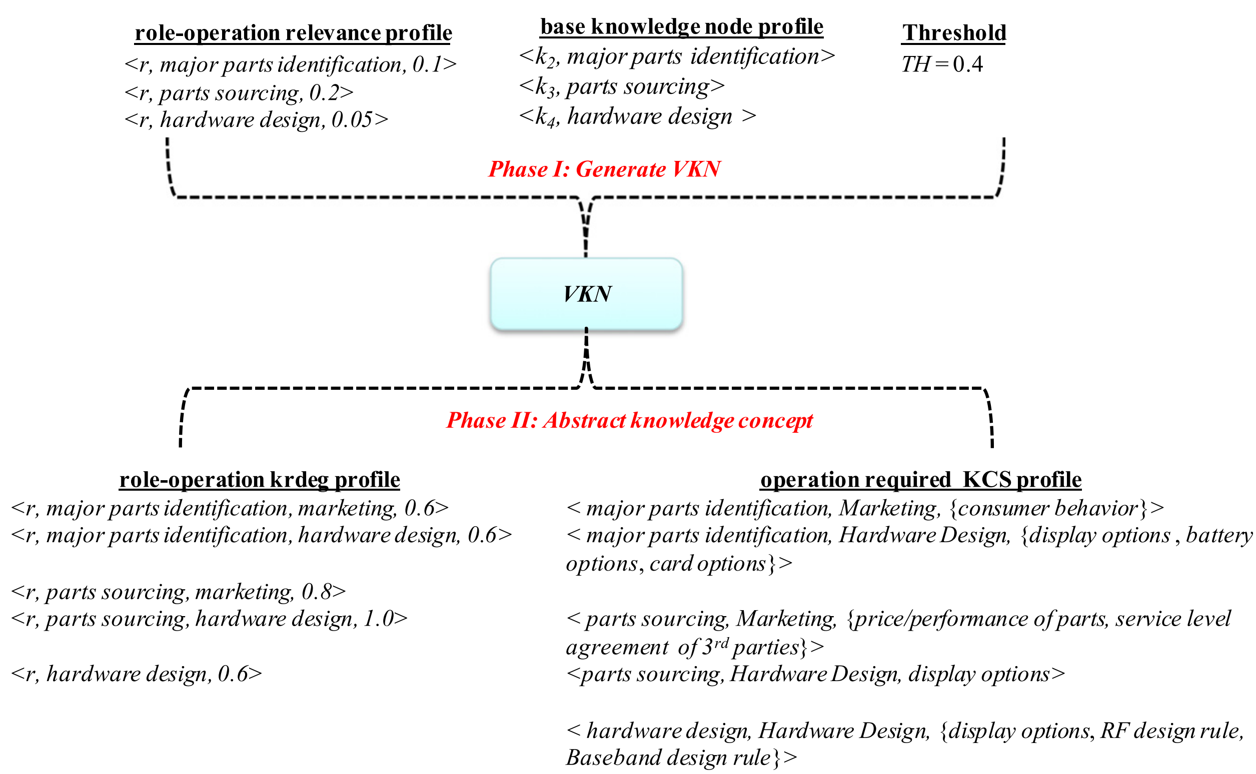

Fig. 16. Information of sourcing department manager role.

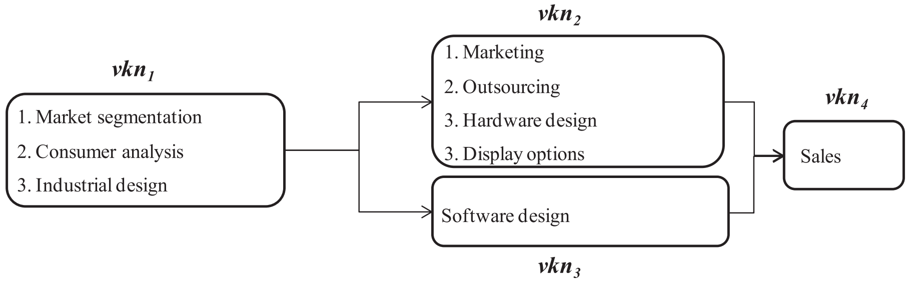

Fig. 17. Role-based VKF for sourcing department manager role.

Fig. 14. Following the algorithm in Fig. 9 to perform knowledge concept abstraction, the Fig. 19(b), (c) and (d) demonstrate the result of knowledge concepts abstraction for vkn_2 when *krdeg* is set to 1.0, 0.8 and 0.6 for all knowledge categories respectively, which means no abstraction, one-level up abstraction and two-level up abstraction. Because the concept level reflects the granularity of knowledge concepts, the concept levels of knowledge concepts increases while *krdeg* increases, which means that the knowledge concepts are more specific. Thus, roles can select the smaller *krdeg* to obtain more general knowledge concepts to communicate with other teammates or they can select the larger *krdeg* to get more specific knowledge concepts to perform their operations. Based on the investigation of the simulations, the parameters *TH* in Phase I and *krdeg* in Phase II serve different purposes and they are set by KF designers and roles according to their requirements.

7.2. Feasibility evaluation and implications

To investigate the feasibility of the role-based KFV model, a preliminary analysis was conducted. We illustrated the case of mobile phone development and provided system design-related documents to several professionals to ask for their opinions about the feasibility of the proposed model from a practical perspective. Overall, there was general agreement that it is justified and feasible to realize the role-based KFV model in organizations. According to their opinions on system implementation, the provided system architecture and activity diagram can be the base to conduct detailed system design for building functional specifications as well as for implementation.

Moreover, these interviewees highlighted that a role-based KFV system tends to evolve in a longitudinal process. The proposed approaches heavily rely on the knowledge of domain experts, KF designers and roles while building KFs and virtual KFs. People put their knowledge in profiles and estimate related parameters while realizing the model, but their experience and knowledge is implicit and hard to obtain systematically. Hence, the back-and-forth between system evaluation and parameter adjustment is necessary for obtaining a well-run system; it will not be trivial work and needs lots of time and effort. From the industrial professionals' perspective, profile setting and parameter estimation according to people' opinions is a common practice for implementing IT systems, especially in design and pilot run phases.

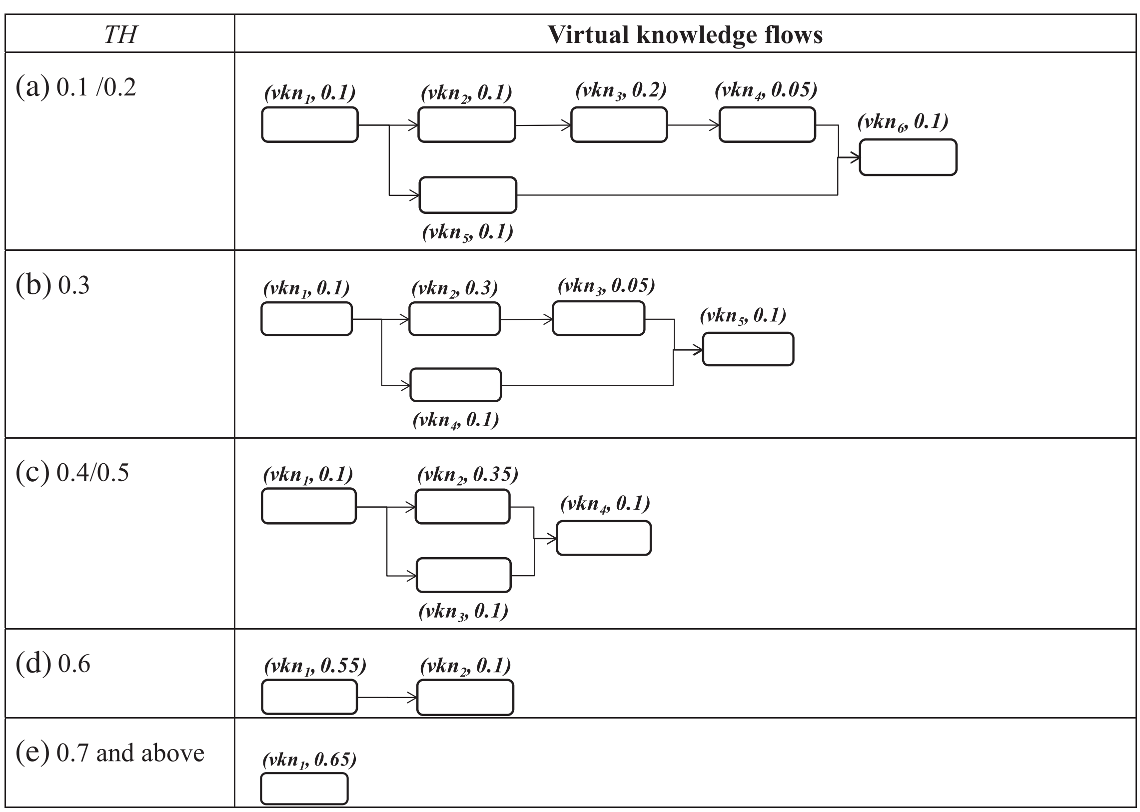

Fig. 18. Simulation result of different *TH* (the larger *TH* obtains fewer VKNs).

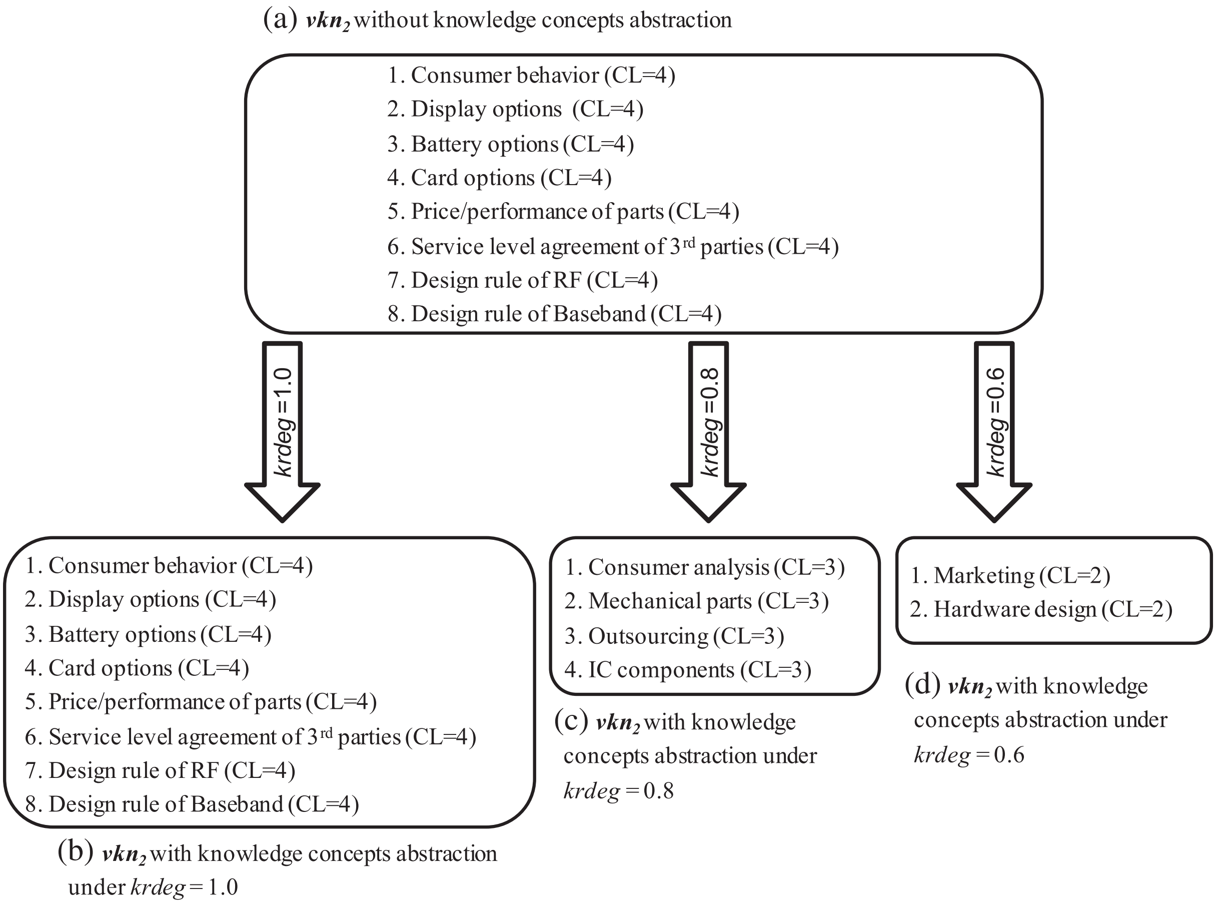

Fig. 19. Abstract knowledge concepts under different *krdeg*.

Practical implications are obtained from the preliminary analysis. These experts recognized the business value of the role-based KFV model because it enhances the efficiency of knowledge flow usage and the effectiveness of knowledge sharing and support in organizations. We summarize practical benefits as follows: (1) the role-based VKFs show roles with a full picture of knowledge-needs by presenting corresponding knowledge concepts with proper concept levels; (2) workers can describe their knowledge-needs precisely by *krdeg* and *TH* and gain a consensus quickly in teams by common domain ontology; (3) KF designers can avoid complex and time-consuming tasks of estimating parameters for each different role; (4) the iterative evaluation and adjustment approach can fine-tune the fitness level of VKFs to increase roles' satisfaction; and (5) the proposed model facilitates organizational knowledge support platforms to help teams improve their communication quality and increase members' productivity.

For a new discipline or a new research topic, theoretical papers are required to explore the basic theory by illustrating term definitions and establishing relationships between concepts [7,12]. In order to explore the new topic, knowledge flow view, this work is targeted as a theoretical paper to establish the role-based KFV model and extend knowledge flow research to cooperative teams for organizational knowledge support. It is an originative study addressing an important extension of knowledge flow research, which considers a phenomenon that workers in teams usually have different knowledge-needs to support task execution according to their individual roles and task functions. We summarize the theoretical contributions in the knowledge flow research field as follows: (1) define a formal role-based KFV model to illustrate virtual knowledge flows; (2) design a kernel approach to derive role-based VKFs from a BKF in role and operation perspectives to extend the application of knowledge flow to teamwork environments; (3) adopt the role-operation krdeg profile to accurately illustrate workers' knowledge-needs and effectively facilitate knowledge concept abstraction; (4) employ role-operation relevance to systematically generate VKNs and (5) establish a role-based KFV system architecture as the base of system implementation.

8. Conclusions and future work

Through knowledge flows (KF), organizations can provide workers with task-relevant knowledge to meet their knowledge-needs. In teamwork environments, knowledge workers with different roles and task functions usually have different knowledge-needs; however conventional KF models do not adapt KFs to satisfy the different knowledge-needs of individuals or team workers. In our previous work, we proposed a novel concept and theoretical model called the knowledge flow view (KFV) model. Based on a worker's specific knowledge-needs, the KFV model abstracts the knowledge nodes of a partial knowledge flow to generate virtual knowledge nodes through knowledge concept generalization.

In this work, we extended the KFV model to discover role-based virtual knowledge flows (VKF). The knowledge-needs of workers in a team may vary because they have different roles when they execute tasks. The role-based KFV model examines the worker's knowledge-needs in terms of his/her role. Discovering role-based VKFs involves two phases: identifying role-relevant virtual knowledge nodes (VKNs) is Phase I and deriving the knowledge concepts of VKNs is Phase II. A role-based VKF comprises a set of VKNs, which are generated from the base knowledge nodes (BKNs) in Phase I according to the relevance degrees between these BKNs and the role. The relevance degrees can be obtained through base knowledge node profiles and role-operation relevance profiles. Based on the relevance degrees, we proposed procedures for generating appropriate role-based VKFs for different organizational roles. Once the accumulated relevance degree of the aggregated BKNs reaches a certain threshold *TH*, a VKN is identified for the role. Then, KF designers can derive the knowledge concepts of the VKN in Phase II based on the role's knowledge requirement level. The approach adjusts (generalizes) the knowledge concepts of VKNs to the appropriate concept levels based on the operation required KCS profiles and the role-operation krdeg profiles.

The role-based VKFs are essential for teamwork because they provide effective knowledge support for team members who are engaged in knowledge-intensive tasks within organizations. This paper is valuable in regard to enhancing knowledge flow's efficiency and increasing the effectiveness of knowledge support through: (1) building the role-based KFV model and illustrating its innovative concepts, (2) undertaking a case analysis and conducting simulation by examples to validate the model's feasibility in a preliminary manner, and (3) conducting elementary system design to further support system implementation.

One limitation of this work is the lack of a rigorous validation for the proposed model by system implementation. Developing a role-based KFV system is an interesting and valuable research direction. However, constructing a system is very complex and difficult to control due to an organization's culture, peripheral system accommodation and operational context [7]. A well-running role-based KFV system also tends to take place in longitudinal processes and needs to adjust parameters and fine-tune methodologies. Consequently, we completed the basic system design in this work and we are going to conduct a detailed system design and implementation, which can rigorously validate the role-based KFV model and the proposed approaches. Furthermore, we have not addressed the issue of generating VKFs in the context of workflow or process environments. A BKF may be modeled in accordance with a process to provide needed knowledge to perform a task. It would be interesting to investigate the interactions between a BKF and a process. Our future work will consider the discovery of role-based VKFs in a process context and explore the synergy between role-based VKFs and process views.

Acknowledgements

This research was supported by the National Science Council of Taiwan under grant no. NSC 99-2410-H-009-034-MY3.

References

[1] A. Abecker, A. Bernardi, K. Hinkelmann, O. Kuhn, M. Sintek, Context-aware, proactive delivery of task-specific information: the knowmore project, Information Systems Frontiers 2 (3) (2000) 253–276.
[2] H.J. Ahn, H.J. Lee, K. Cho, S.J. Park, Utilizing knowledge context in virtual collaborative work, Decision Support Systems 39 (4) (2005) 563–582.
[3] A. Anjewierden, R. de Hoog, R. Brussee, L. Efimova, Detecting knowledge flows in weblogs, Proceedings of the Thirteenth International Conference on Conceptual Structures, Atlanta, 2005, pp. 1–12.
[4] U.M. Borghoff, R. Pareschi, Information technology for knowledge management, Journal of Universal Computer Science 3 (8) (1997) 835–842.
[5] C. Chandra, A. Tumanyan, Organization and problem ontology for supply chain information support system, Data & Knowledge Engineering 61 (2) (2007) 263–280.
[6] Y.-J. Chen, Development of a method for ontology-based empirical knowledge representation and reasoning, Decision Support Systems 50 (1) (2010) 1–20.
[7] D.T. Croasdell, M. Jennex, Z. Yu, T. Christianson, M. Chakradeo, W. Makdum, A meta-analysis of methodologies for research in knowledge management, organizational learning and organizational memory: five years at HICSS, Proceedings of the 36th Annual Hawaii International Conference on System Sciences, 2003.
[8] T.H. Davenport, D.W.D. Long, M.C. Beers, Successful knowledge management projects, Sloan Management Review 39 (2) (1998) 43–57.
[9] T.C. Du, F. Li, I. King, Managing knowledge on the Web — extracting ontology from HTML Web, Decision Support Systems 47 (4) (2009) 319–331.
[10] D. Georgakopoulos, M. Hornick, A. Sheth, An overview of workflow management: from process modeling to workflow automation infrastructure, Distributed and Parallel Databases 3 (2) (1995) 119–153.
[11] T.R. Gruber, A translation approach to portable ontology specifications, Knowledge Acquisition 5 (2) (1993) 199–220.
[12] Z. Guo, J. Sheffield, A paradigmatic and methodological examination of knowledge management research: 2000 to 2004, Decision Support Systems 44 (3) (2008) 673–688.
[13] H. Holz, H. Maus, A. Bernardi, O. Rostanin, A lightweight approach for proactive, task-specific information delivery, Proceedings of the 5th International Conference on Knowledge Management (I-Know), Graz, Austria, 2005, pp. 413–420.
[14] S. Kim, H. Hwang, E. Suh, A process-based approach to knowledge-flow analysis: a case study of a manufacturing firm, Knowledge and Process Management 10 (4) (2003) 260–276.
[15] M.M. Kwan, P. Balasubramanian, KnowledgeScope: managing knowledge in context, Decision Support Systems 35 (4) (2003) 467–486.
[16] C.-H. Lai, D.-R. Liu, Integrating knowledge flow mining and collaborative filtering to support document recommendation, Journal of Systems and Software 82 (12) (2009) 2023–2037.
[17] F. Leymann, W. Altenhuber, Managing business processes as an information resource, IBM Systems Journal 33 (2) (1994) 326–348.
[18] S.-T. Li, H.-C. Hsieh, Managing operation knowledge for the metal industry, Journal of Universal Computer Science 9 (6) (2003) 472–480.
[19] C.-W. Lin, D.-R. Liu, H.-F. Chen, Modeling knowledge-flow view for knowledge support in teamwork, Proceedings of the 21st IASTED International Conference on Modelling and Simulation, Banff, Alberta, Canada, 2010.
[20] D.-R. Liu, C.-H. Lai, Mining group-based knowledge flows for sharing task knowledge, Decision Support Systems 50 (2) (2011) 370–386.
[21] D.-R. Liu, C.-W. Lin, Modeling the knowledge-flow view for collaborative knowledge support, Knowledge-Based Systems 31 (2012) 41–54.
[22] D.-R. Liu, M. Shen, Workflow modeling for virtual processes: an order-preserving process-view approach, Information Systems 28 (6) (2003) 505–532.
[23] D.-R. Liu, M. Shen, Business-to-business workflow interoperation based on process-views, Decision Support Systems 38 (3) (2004) 399–419.
[24] D.-R. Liu, I.-C. Wu, Collaborative relevance assessment for task-based knowledge support, Decision Support Systems 44 (2) (2008) 524–543.
[25] D.-R. Liu, I.-C. Wu, K.-S. Yang, Task-based k-support system: disseminating and sharing task-relevant knowledge, Expert Systems with Applications 29 (2) (2005) 408–423.
[26] X. Luo, Q. Hu, W. Xu, Z. Yu, Discovery of textual knowledge flow based on the management of knowledge maps, Concurrency and Computation: Practice and Experience 20 (15) (2008) 1791–1806.
[27] A. Öhgren, K. Sandkuhl, Towards a methodology for ontology development in small and medium-sized enterprises, IADIS Conference on Applied Computing, Algarve, Portugal, 2005.
[28] Á.E. Prieto, A. Lozano-Tello, Use of ontologies as representation support of workflows oriented to administrative management, Journal of Network and Systems Management 17 (3) (2009) 309–325.
[29] T.S. Raghu, A. Vinze, A business process context for knowledge management, Decision Support Systems 43 (3) (2007) 1062–1079.
[30] O.M. Rodríguez, A.I. Martínez, J. Favela, A. Vizcaíno, M. Piattini, Understanding and supporting knowledge flows in a community of software developers, 10th Collaboration Researchers International Working Group (CRIWG) Conference, San Carlos, Costa Rica, 2004, pp. 52–66.
[31] V. Sambamurthy, M. Subramani, Special issue on information technologies and knowledge management, MIS Quarterly 29 (1) (2005) 1–7.
[32] S. Sarnikar, J.L. Zhao, Automating knowledge flows by integrating workflow and knowledge discovery techniques, Proceedings of the 2007 Winter Conference on Business Intelligence, Salt Lake City, UT, 2007.
[33] S. Sarnikar, J. Zhao, Pattern-based knowledge workflow automation: concepts and issues, Information Systems and E-Business Management 6 (4) (2008) 385–402.
[34] M. Shen, D.-R. Liu, Discovering role-relevant process-views for disseminating process knowledge, Expert Systems with Applications 26 (3) (2004) 301–310.
[35] Z.S. Syed, T. Finin, A. Joshi, Wikipedia as an ontology for describing documents, Proceedings of the Second International Conference on Weblogs and Social Media, Seattle, Washington, 2008.
[36] M. Uschold, M. Gruninger, Ontologies: principles, methods and applications, Knowledge Engineering Review 11 (2) (1996) 93–136.
[37] M. Uschold, M. King, Towards a methodology for building ontologies, Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 1995.
[38] A.J.M.M. Weijters, W.M.P. van der Aalst, Process mining: discovering workflow models from event-based data, Proceedings of the 13th Belgium–Netherlands Conference on Artificial Intelligence (BNAIC 2001), 2001.
[39] I.-C. Wu, D.-R. Liu, W.-H. Chen, Task-stage knowledge support: coupling user information needs with stage identification, IEEE International Conference on Information Reuse and Integration (IRI), 2005, pp. 19–24.
[40] M.H. Zack, Managing codified knowledge, Sloan Management Review 40 (4) (1999) 45–58.
[41] Z. Zhang, Z. Yang, Q. Liu, Modeling knowledge flow using Petri Net, IEEE International Symposium on Knowledge Acquisition and Modeling Workshop, China, 2008, pp. 142–146.
[42] W. Zhao, W. Dai, Integrated modeling of business processes and knowledge flow based on RAD, IEEE International Symposium on Knowledge Acquisition and Modeling Workshop, China, 2008, pp. 49–53.
[43] J.L. Zhao, H.H. Bi, H. Chen, D.D. Zeng, C. Lin, M. Chau, Process-driven collaboration support for intra-agency crime analysis, Decision Support Systems 41 (3) (2006) 616–633.
[44] H. Zhuge, Knowledge flow management for distributed team software development, Knowledge-Based Systems 15 (8) (2002) 465–471.
[45] H. Zhuge, A knowledge flow model for peer-to-peer team knowledge sharing and management, Expert Systems with Applications 23 (1) (2002) 23–30.

[46] H. Zhuge, Discovery of knowledge flow in science, Communications of the ACM 49 (5) (2006) 101–107.
[47] H. Zhuge, Knowledge flow network planning and simulation, Decision Support Systems 42 (2) (2006) 571–592.
[48] H. Zhuge, W. Guo, Virtual knowledge service market — for effective knowledge flow within knowledge grid, Journal of Systems and Software 80 (11) (2007) 1833–1842.

Dr. Duen-Ren Liu is a professor of the Institute of Information Management at the National Chiao Tung University of Taiwan. He received the BS and MS degrees in Computer Science from the National Taiwan University and his PhD in Computer Science from the University of Minnesota. His research interests include information systems, knowledge engineering and management, workflow systems, electronic commerce and recommender systems.

Dr. Chih-Wei Lin received his PhD degree in Information Management from National Chiao Tung University in 2012. His research interests include knowledge management, workflow management systems, and electronic commerce.

Hui-Fang Chen is a PhD student of the Institute of Information Management at the National Chiao Tung University of Taiwan. She received the MS degree in Information Management from the National Chiao Tung University. Her research interests include information systems, knowledge management, and recommender systems.