Journal of Network and Innovative Computing
ISSN 2160-2174 Volume 2 (2014) pp. 071-080
© MIR Labs, www.mirlabs.net/jnic/index.html

# Knowledge Collaborator Agent in Expert Locator System: Multi-Agent Simulation in the Validation of GUSC Model

**Shahrinaz Ismail**[1] **and Mohd Sharifuddin Ahmad** [2]

[1] Malaysian Institute of IT, Universiti Kuala Lumpur,
50250 Kuala Lumpur, Malaysia
*shahrinaz@unikl.edu.my*

[2] College of Information Technology, Universiti Tenaga Nasional,
43000 Kajang, Malaysia
*sharif@uniten.edu.my*

***Abstract*****: This paper presents a multi-agent simulation that demonstrates the roles identified to assist human knowledge workers, based on the Get-Understand-Share-Connect (GUSC) Model. The system design is based on the content analysis from an interview survey conducted on selected organisations in Malaysia. A significant finding from the interview is the existence of the Knowledge Collaborator role, which the literature commonly refers to as the gatekeeper. According to the interview respondents, Knowledge Collaborator locates knowledge sources or experts upon request from the Knowledge Seeker within an organisation, which is based on the needs. A scenario of the mediation of Knowledge Seeker-Knowledge Collaborator tasks is simulated in this paper, animated in an agent-oriented development platform. This scenario is expanded to Knowledge Collaborator-Knowledge Expert mediation of tasks, to further prove the GUSC roles played by the agents.**

***Keywords***: software agent, multi-agent system, simulation, knowledge collaborator, GUSC.

## I. Introduction

The role of gatekeepers is almost unseen as significant to the growth of knowledge management implementation in organisations. While there are many citations by the literature in social sciences domain, this role is less explored in artificial intelligence domain despite the capability of the technology to better assist the human aspects of organisational knowledge management (OKM). This offers an opportunity for this study to proceed where the literature left off. In doing so, the bottom-up approach from personal knowledge management (PKM) to OKM is adopted, since the gatekeeper is of individualistic nature, or in other words personal.

Personal knowledge management (PKM), or knowledge management (KM) at individual knowledge workers' level, has been implemented and practised over the Web 2.0 tools and technologies in the last decade, but the intelligence underlying the PKM technology is somewhat ignored. In contributing to this deficiency, we propose an agent-mediated PKM system based on a model, which we called the Get-Understand-Share-Connect (GUSC) model [1, 2, 3]. In this model, software agent's intelligence is exploited to perform the tasks of managing personal knowledge on behalf of human knowledge workers. This paper presents a part of the work in progress of our research in agent-mediated Personal Knowledge Management.

Recent research on PKM suggests that circumstances dictate the need to find knowledge experts within and outside of the organisations [1]. In technology and knowledge-intensive organisations, the need to seek for new knowledge is compelling, which demands the role of a gatekeeper to identify the relevant and appropriate knowledge experts [2]. While the need persists, locating the right person who knows the right knowledge presents a challenge to the intelligent agent research community.

This paper aims to animate a knowledge collaborator as a mediator in knowledge expert locating process for a knowledge seeker. In order to achieve this aim, the objectives are: to understand the agent environment based on real human environment; to propose the system in visual diagrams; and to develop the multi-agent system that shows the simulation of knowledge collaborator in knowledge seeker-knowledge expert environment. The paper then presents the development of an agent-based simulation to demonstrate an agent-mediated PKM framework [3]. Intelligent software agents are deployed in the simulation to model the PKM processes, namely the get, understand, share and connect (GUSC) processes in the simulation scenarios.

The simulation is based on the scenarios of agent-mediation tasks performed by Knowledge Seeker, Knowledge Expert and Knowledge Collaborator.

## II. Related Works

### *A. The Concept of a Gatekeeper*

Often individuals know each other outside of formal arrangements with official alliances, and interact beyond official duties, leading to knowledge flows and learning [4]. This results in the unknown and unseen activities (of the individuals) to the executives and managers. Even though unseen, the network "is found to be a powerful, intangible infrastructure that crosses organisational boundaries and often into the World Wide/Semantic Web" [2]. These individuals, from within a firm, are influenced by ongoing relations with other persons within the firm, from other firms and from non-firm organisations [5]. The relations are beyond official organisational arrangements.

Sometimes, these individual knowledge workers take up the role of a gatekeeper – "the key person who is exposed to sources of knowledge outside the organisation and to whom others within the organisation frequently turn to for knowledge" [4]. In a recent study on agent-mediated PKM, the 'gatekeeper' role is referred to as knowledge source or internal knowledge expert [1]. Internal knowledge expert here basically means that the gatekeeper is considered as "the 'point of reference', and the recognition of expertise connected to this person depends on his/her recommendations" [2]. For the purpose of this paper, a more technical term is used to represent this role, which is 'knowledge collaborator'.

On a technical aspect, intelligent software agents can be mediated to find experts within social networks [2], where the agents are capable of detecting "which people possess the required expertise for solving a problem at hand" [6]. Software agents can also assess on how "some of the members are recognised as experts by their colleagues in the community" [6].

There are many aspects or criteria that could assist the gatekeeper or knowledge collaborator to identify knowledge experts, which is commonly depending on the situation, the need and the required knowledge. This paper focuses on a common scenario in organisations that implements knowledge management at organisational level, namely institutes of higher learning, oil and gas, telecommunication, to name a few. These scenarios are derived from the previous research, presented in 2013 [2].

### *B. The GUSC Model for Agent-mediation*

Current research in agent-mediated knowledge management (KM) shows promising results of intelligent agents performing tasks on behalf of their human counterparts. At this level of KM, the term used is personal knowledge management (PKM), in which knowledge is managed at individual human knowledge workers' levels. In a recent study on agent- mediated PKM processes [7, 8], the PKM processes are defined in a cycle of 'Get/retrieve knowledge', 'Understand/ analyse knowledge', 'Share knowledge' and 'Connect to knowledge source' (i.e. GUSC), which are translated into interactions between humans and agents. These interactions are proven to be possible between human-agent and agent-agent, while easing the human-human interactions [3].

The order of the PKM processes between individuals (i.e. GUSC) is found to be different when the tasks are mediated using software agents. Instead of starting with G (i.e. get/retrieve knowledge), the human-agent interaction starts with C (i.e. connect), and this is followed by S, G and U (i.e. CSGU). The difference in this sequence is due to the "different environments in which knowledge is being translated between tacit and explicit forms" [1]. The first sequence (i.e. GUSC) represents the changes between tacit and explicit forms of knowledge during the interactions within a knowledge worker's mind in managing personal knowledge, whereas the second sequence (i.e. CSGU) represents the processes when software agents are used to mediate the task of finding knowledge experts on behalf of the human knowledge worker [1].

In deploying the GUSC model, software agents are found to have the capabilities that PKM processes need. Comparison analysis is made on the capabilities of software agents based on the definitions given by authors of the past two decades against the GUSC processes. Consequently, the challenge of making the agents to meet the expected capabilities is resolved by assigning the agents with the GUSC roles [8]. For example, software agents should be able to "engage in dialogues and negotiate and coordinate the transfer of information" [9] for which the processes of Get, Share and Connect are found to be the required PKM processes. On the other hand, the ability of software agents to "carry out some set of operations on behalf of a user or another agent or program with some degree of independence or autonomy, and in so doing, employ some knowledge or representation of the user's goals or desires" [10] requires the agents to have the role to Understand. The most cited definition of software agent, "an encapsulated computer system that is situated in some environment and that is capable of flexible action in that environment in order to meet its design objectives" [11], covers all the four processes of GUSC.

### *C. The BDI Agent Architecture and JACK*

The Belief-Desire-Intention (BDI) agent architecture is readily applicable to the task of modelling the reasoning components (i.e. the humans) in simulation, hence the decision in integrating BDI agents using the commercial JACK [12] platform in this work. JACK is a mature, cross-platform environment for building, running and integrating commercial-grade multi-agent systems, built upon a sound logical foundation: BDI (Belief-Desire-Intention).

"BDI agents are programmed using goals (or events) representing what the agent wants to achieve or respond to". Based on the goals, plans are triggered, describing different ways to achieve them. These plans are made up of sub-goals, which have associated actions. Agent beliefs are used in selecting which plan to instantiate in a particular situation

[13].

Based on these elements built in JACK (i.e. event, plan, action and belief), the simulations can be designed and developed in a proper manner for validation purposes. Agents in JACK are defined in terms of their beliefs (what they know and what they know how to do), their desires (what goals they would like to achieve), and their intentions (the goals they are currently committed to achieving). In general, each agent is defined in terms of its goals, knowledge and social capability, and is then left to perform its function autonomously within the environment it is embedded in [12].

## III. Methodology

An interview survey was conducted on eight respondents from various organisations that portray the need and use of gatekeepers, namely oil and gas, telecommunication, banking, business project investment, government agencies and universities [2]. An overall view of the interview results pertaining the scenario where gatekeepers are required, 'used' and being useful for locating knowledge experts is illustrated in the design phase.

For the system design phase, the Tropos software development methodology is used, since it is found to be suitable for agent-oriented modelling. The Tropos framework has also been applied for developing multi-agent systems, and it spans four phases of software development [14]:

- *early requirements analysis*: concerned with the understanding of a problem by studying an organisational setting – the output is an organisational model which includes relevant actors, their goals and inter-dependencies;
- *late requirements analysis*: where the system-to-be is described within its operational environment, along with relevant functions and qualities;
- *architectural design*: where the system's global architecture is defined in terms of subsystems, interconnected through data, control and other dependencies;
- *detailed design*: where behavior of each architectural component is defined in further detail.

The analysis and design phases of the Tropos methodology are useful to translate the interview results into a common scenario for all organisations. This is illustrated step by step in the next section, with highlights of the get, understand (or analyse), share and connect (i.e. the G-U-S-C). The Tropos modelling method complements our need to look at the overall system in a nodal view to visualise the multi-agent system that consists of agents as nodes, supplemented by the environment, goals, and tasks.

Since this paper contributes to the research on agent-mediated PKM, there is a need to validate the model. To achieve this, we developed a computer simulation of the GUSC model animated by software agents as a proof-of-concept. To produce this, two parts are developed: simulation on Knowledge Seeker-Knowledge Expert interaction; and simulation on Knowledge Seeker-Knowledge Collaborator interaction.

The simulation program is constructed with the scenario settings, in which the conditions and behaviours of the entities are analysed to produce a well-planned design of the simulation process. This is followed by the simulation settings, in which the agents and their environmental parameters are defined.

In this simulation, there are three main agents that work in conjunction with their human counterparts, namely the Knowledge Seeker agent (i.e. the one who seeks for knowledge sources), the Knowledge Expert agent (i.e. the one whose expertise is sought for) and the Knowledge Collaborator agent (i.e. the gatekeeper who accepts the task of seeking knowledge expert from the knowledge seeker).

## IV. From Data and Design

This section details the four phases of system design based on the Tropos methodology and reveals the interview results in sequence.

### A. Early Requirements Analysis

Early requirements analysis is the first step in revealing the organisational setting with which the scenario of knowledge expert locating is analysed based on the interview results. The output of this analysis is an organisational model that consists of relevant actors, their goals and inter-dependencies. Figure 1 illustrates the scenario of how the actors (i.e. Knowledge Seeker, Knowledge Collaborator and Knowledge Expert) depend on each other based on their individual goals within the defined scope of locating experts.

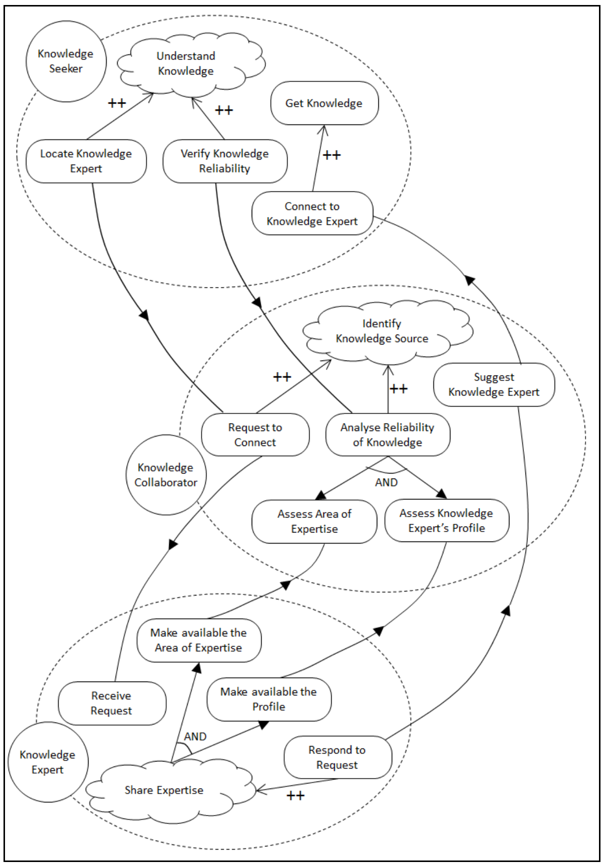


**Figure 1.** Early Requirements diagram

From Figure 1, the Knowledge Seeker is illustrated with two main goals: a soft goal of ‘understanding knowledge’; and a hard goal of ‘get knowledge’. In order to achieve the soft goal, the Knowledge Seeker needs to ‘locate a Knowledge Expert’ and ‘verify knowledge reliability’. There is a need to understand from the right person, especially when the person is unknown or unreachable by the Knowledge Seeker, thus the need of verifying the expertise of the Experts upon locating them. On the other hand, ‘connect to Knowledge Expert’ can only make the ‘get knowledge’ achievable, since without connecting to the source (i.e. Knowledge Expert), the Seeker could not get or receive new knowledge. This setting complements the feedback received from the interview respondents, which stated the need to locate knowledge expert as and when new knowledge is required to fulfil certain tasks new to a knowledge worker [2].

In the vast world of reachable information and knowledge, it would be difficult for a knowledge worker to know where to find the right source of knowledge and expertise, and it could waste the person’s time in hunting for something new and yet to be known. In huge organisations like oil and gas companies, there is a unit or a person who is dedicated to assist knowledge workers to locate experts and to ensure that the experts are reliable and able to share the right knowledge needed by them [2]. In this scenario (in Figure 1), the unit or person known as the gatekeeper is termed as the Knowledge Collaborator, who has two main goals: a soft goal of ‘identify knowledge source’; and a hard goal of ‘suggest Knowledge Expert’. Contributing to the soft goal, the Knowledge Collaborator needs to request the expert found for connection (i.e. ‘request to connect’), and ‘analyse reliability of knowledge’ by assessing the area of expertise and Knowledge Expert’s profile.

The concept of getting the right knowledge can only happen if the knowledge expert is willing to share, thus the need to request for consent to be connected before any sharing is possible. The setting of the Knowledge Expert in Figure 1 shows the two main goals: the soft goal of ‘share expertise’; and the hard goal of ‘receive request’. To an expert, sharing knowledge could increase the reputation and credibility in the area of expertise. In achieving the goal of sharing expertise, the Knowledge Expert needs to make available the area of expertise and profile (which is to be found or identified by the Knowledge Collaborator), and respond to request from the Knowledge Collaborator.

## B. *Late Requirements Analysis*

“Late requirements analysis describes the system-to-be as an actor within its operational environment, along with relevant functions and qualities” [14]. Figure 2 shows the result of this analysis based on the early requirements analysis and more details from the interview result.

As mentioned in the early requirements analysis, a Knowledge Expert has the intention to be known as a point of reference in the area of expertise. In the World Wide Web (WWW) and Web 2.0 technologies, an expert is commonly quoted and tagged by public and other experts in the field. At a certain point of advancement, a recommendation hit exists that identifies tags and quotes as recommendations by these people over the virtual network, termed as Recommender. This external actor, called Recommender is the link between the collaborator and the expert.

As shown in Figure 2, upon identifying the expert from recommendation hits, the Knowledge Collaborator could further query the knowledge source and analyse the expertise reliability from the knowledge source. This Knowledge Source is drawn as a node, in an actor symbol, which indicates any form of source, such as database and knowledge repository. The Knowledge Source is where the Knowledge Expert would share, upload and make available the profile and area of expertise, with an intention for others to be able to locate them. Once the reliability of the expertise is verified, the Knowledge Collaborator would ‘request to connect’ directly to the Knowledge Expert.

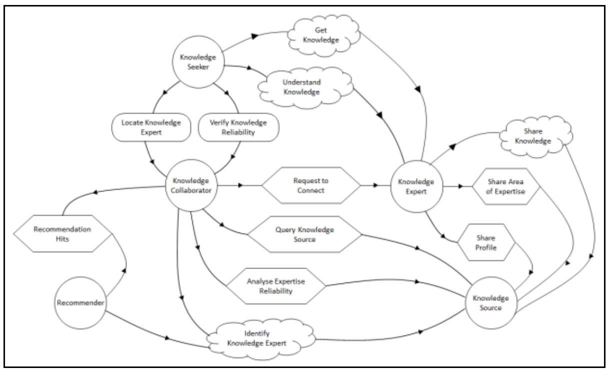


**Figure 2.** Late Requirements diagram

## C. *Architectural Design*

“Architectural design defines the system’s global architecture in terms of subsystems, interconnected through data, control and other dependencies” [14]. It is moving towards the design of the system, hence the unnecessary soft goals to be presently drawn.

Since the original requirements illustrate the three main actors or humans involved in the setting (i.e. Knowledge Seeker, Knowledge Collaborator and Knowledge Expert), the architectural design reveals these actors’ environment in agent-based subsystems. The actors mentioned are the agents that mediate the tasks on behalf of their human counterparts, as shown in Figure 3. This design is basically similar to the late requirements analysis, minus the soft goals that are non-computable in the multi-agent system.

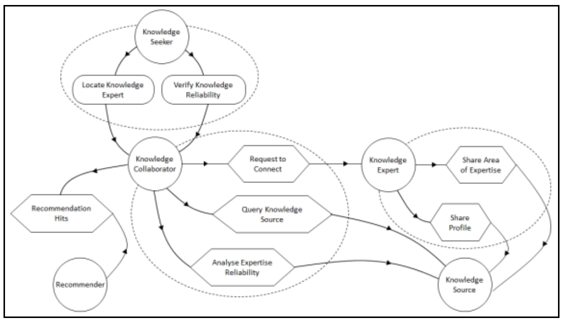


**Figure 3.** Architechtural Design diagram

The external actors shown in Figure 3 (i.e. the Recommender and the Knowledge Source) are not within the scope of the simulation presented in this paper. It will be further explored in future works, since it requires more in-depth analysis in the semantic world. For the purpose of this research, the activities performed by the said three actors are ample.

### *D. Detailed Design*

Encapsulating the three main actors as agents in a system, the Expert Locator System is defined and illustrated in the detailed design shown in Figure 4. Detailed design highlights the behavior of each component identified during the architectural design, and it reflects how the system is developed in the next phase.

In general, Figure 4 shows the boundary of the Expert Locator System developed and presented in this paper. The GUSC concept is clearly shown in the early requirements analysis, evolving into this stage of detailed design where the concept dilutes within the tasks assigned to the agents.

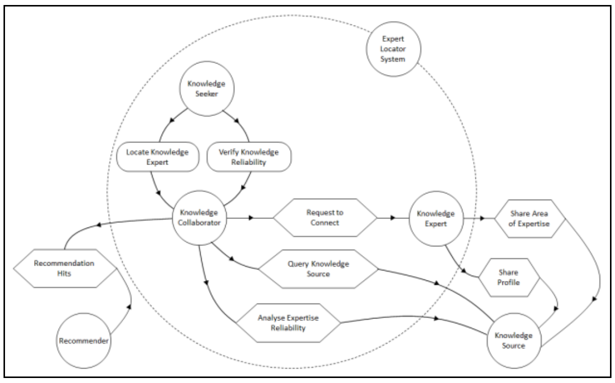


**Figure 4.** Detailed Design diagram

Figure 4 shows the scope of the Expert Locator System, with Knowledge Seeker, Knowledge Collaborator and Knowledge Expert act as agents within the system. Even though the human counterpart of Knowledge Expert could be from outside of the organisation, an agent is assigned to this human expert to assist the Knowledge Collaborator agent for further communication and getting the connection possible. The same goes to the Knowledge Source, an agent that sits on the source of any form (i.e. databases, knowledge repositories, etc.), even though the object/form is outside of the organisational boundaries.

## V. Scenario and Simulation Settings

There are two scenarios chosen to validate GUSC Model in this agent simulation environment: an agent-mediated search on a Knowledge Expert by the Knowledge Seeker; and an agent-mediated search on a Knowledge Expert by the Knowledge Seeker upon delegating the task to the Knowledge Collaborator. In both scenarios, it is assumed that the agents communicate with their human counterparts while progressing the workflow.

In the first scenario, two agents are animated: Knowledge Seeker and Knowledge Expert. In addition to this, a simplified profile of the Knowledge Expert is used as the search criterion. The scenario is about a Knowledge Seeker locating a Knowledge Expert and requesting to ‘connect’ to the expert. The Knowledge Seeker has the capability to ‘get’ knowledge from the belief set and ‘understand’ which profile has the closest match with the knowledge and expertise it is looking for. The belief set is a form of knowledge ‘share’ for both the Knowledge Seeker and the Knowledge Expert. The choice to connect is given to the Knowledge Seeker, and the Knowledge Expert is also given the choice to connect once the Knowledge Seeker agrees to send the request to connect.

In the second scenario, two agents are animated: Knowledge Seeker and Knowledge Collaborator. The simplified profile of the Knowledge Expert is used for the Knowledge Collaborator to understand and suggest further actions. The scenario is about a Knowledge Seeker locating a Knowledge Expert and request to ‘connect’ with the Knowledge Expert. In this case, the request to locate the Knowledge Expert is sent to the Knowledge Collaborator. The Knowledge Collaborator ‘understands’ the request of the Knowledge Seeker, ‘gets’ the knowledge from the belief set, ‘understand’ the match of the knowledge needed from the belief set, and ‘connect’ to Knowledge Seeker with a suggestion on the chosen Knowledge Expert. The ‘sharing’ is done by the Knowledge Expert in the belief set. In other words, the Knowledge Collaborator is the middle person, or the gatekeeper to the knowledge repository.

The introduction of the Knowledge Collaborator as the gatekeeper is based on real case scenarios in organisations that implement knowledge management system either manually or automatically using the IT infrastructure. According to the interview respondents during the survey [2], gatekeepers facilitate personal knowledge management within organisations as they become the point-of-reference in connecting knowledge workers and knowledge experts from within and without the organisations.

### *A. Simulation Settings for Knowledge Seeker Agent*

The primary actor in this simulation is the Knowledge Seeker agent (i.e. ks_agent), that initiates the whole simulation process or reacts to the need of locating Knowledge Expert from the human Knowledge Seeker. In other words, the ks_agent mediates the task on behalf of its human counterpart, the human Knowledge Seeker. The main tasks performed by the ks_agent are helping the Knowledge Seeker (KS) locate Knowledge Expert (KE) and sending requests for help on behalf of KS. These tasks define the capability of the ks_agent. The profile of the KS is stored in belief set (i.e. ksBS) and it is ready to be shared when a request for help is sent to the KE.

Figure 5 shows the JACK agent action diagram that includes the elements in the KS agent’s (i.e. ks_agent) environment. This is the basic form of agent environment for ks_agent that mediates the tasks for the human Knowledge Seeker.

The ks_agent commences the task with its ability to use the plan, PlnFindKE, to find KE by searching the KE’s profile (kePRF) located in keData (assumed to be located randomly in the WWW). The search for KE is based on the KS input criteria, such as research topic, as this simulation is based on the scenario of locating research knowledge expert. The plan, PlnFindKE, looks up in the belief set kePRF for a subject whose profile matches the search criteria. The moment a match is found, ks_agent notifies its human counterpart, reporting on the expert profile, such as name and date of birth. For the purpose of a simple validation, we include the name and date of birth as the profile fields to assist the Knowledge Seeker in deciding based on the name (that could be a well-known and recommended name/expert in the research topic) and date of birth (to verify the expertise in terms of experience via the age of the Knowledge Expert).

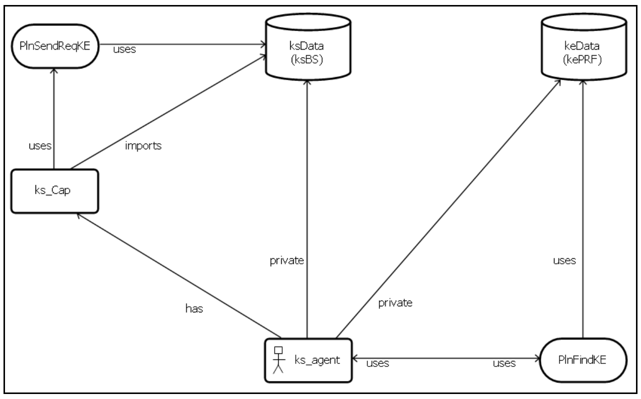


**Figure 5.** Agent Action Diagram for ks_agent

When the KS is informed of the search result, it is a choice to be made by the human KS to command the ks_agent for the next task. As KS commands the ks_agent to engage KE, ks_agent executes the PlnSendReqKE plan to send help request to KE, along with the Knowledge Seeker's bio data. Figure 1 graphically explains that the ks_agent has the capability (ks_Cap) of using the send request to PlnSendReqKE by using the KS's belief set (ksBS). In order to have this capability, the ks_Cap imports the belief set from the ksBS stored in ksData.

In sending the request to KE, the ks_agent posts the event EvtSendReqKE, which handles the plan PlnSendReqKE. This post triggers the message to be sent to the KE agent (i.e. MsgEvtRecReqKE). Figure 6 shows this flow of agent action.

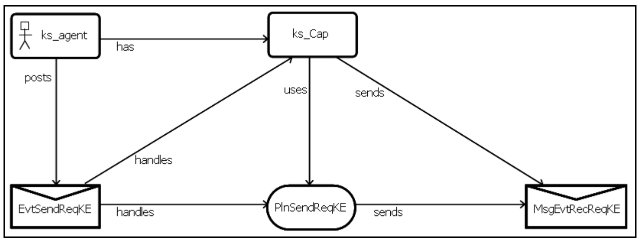


**Figure 6.** Agent Action Diagram on ks_agent posting Event to Send Request

Table 1 summarises the elements in the ks_agent environment, based on the agent action diagram of Figures 5 and 6. The elements include the agent, belief set, profile, capability and events.

| Name / Role | Code Name | Description |
|---|---|---|
| Knowledge Seeker (KS) | ks_agent | Knowledge Seeker's agent that mediates the human Knowledge Seeker |
| KS's belief set | ksBS | The location where the Knowledge Seeker's profile/bio data is stored |
| KS's Capability | ks_Cap | Knowledge Seeker's capability |
| KE's profile | kePRF | Knowledge Expert's profile |
| KS's Plan | PlnFindKE | Knowledge Seeker's plan to find/locate Knowledge Expert |
| | PlnSendReqKE | Knowledge Seeker's plan to send request to Knowledge Expert |
| KS's Event | EvtSendReqKE | An event of Sending Request to Knowledge Expert |
| | MsgEvtRecReqKE | An event of Receiving Request Message that is sent to Knowledge Expert |

*Table 1.* Simulation settings of elements in ks_agent environment.

## B. *Simulation Settings for Knowledge Expert Agent*

An agent that mediates the human Knowledge Expert is called the ke_agent. In general, ks_agent and ke_agent act as intermediaries for the human KS and KE. As ke_agent receives request from ks_agent, it triggers the PlnNotifyKE plan to notify the human KE on the request along with the information on KS, such as bio data. This is shown in the agent action diagram in Figure 7. As shown in the figure, the main tasks performed by the ke_agent are to notify KE on the request from KS (PlnNotifyKE) and send a response to the request on behalf of KE (PlnSendRespKE). These tasks define the capability of the ke_agent (ke_Cap).

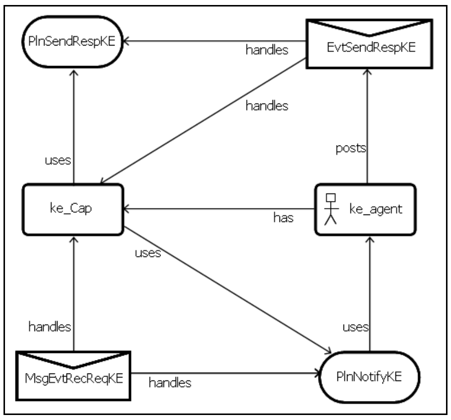


**Figure 7.** Agent Action Diagram for ke_agent

In the event KE accepts the request from KS, ke_agent executes the plan, PlnSendRespKE to inform ks_agent with KE's email address (or other contact details shared by KE) attached in the message MsgEvtKSRecRespKE event. For this purpose, the ke_agent posts the EvtSendRespKE event, which handles the PlnSendRespKE plan. Upon receiving the acceptance, ks_agent executes the plan called KENotifyKS, displaying KE's contact details. Figure 8 shows this process as an extension to the previous Figure 7 (on ke_agent action) and Figure 6 (on ks_agent action).

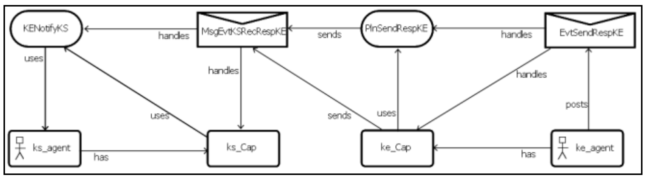


**Figure 8.** Agent Action Diagram on ke_agent posting Event to Send Response to KS's Request

Table 2 summarises the elements in the ke_agent environment, based on the agent action diagram of Figures 7 and 8.

| Name / Role | Code Name | Description |
|---|---|---|
| Knowledge Expert (KE) | ke_agent | Knowledge Expert's agent that mediates the human Knowledge Expert |
| KE's Capability | ke_Cap | Knowledge Expert's capability |
| KE's Plan | PlnNotifyKE | Knowledge Expert's plan to notify the Knowledge Expert on the request along with the information about Knowledge Seeker's bio data |
| | PlnSendRespKE | Knowledge Expert's plan to respond to the Knowledge Seeker's request along with the information on how to contact the Knowledge Expert (e.g. email address). |
| KE's Event | EvtSendRespKE | An event of Sending Response to Knowledge Seeker |
| | MsgEvtKSRecRespKE | An event of Receiving Response Message that is sent to Knowledge Seeker |

*Table 2.* Simulation settings of elements in ke_agent environment.

## C. Overview of ks_agent and ke_agent Interaction

Figure 9 shows the agent environment in which the ks_agent and ke_agent interacts with their capabilities in triggering events to send messages across the environment for the Connect process. The agent simulation setting is further developed in the physical model, with simple interface to present a better simulation flow on how the agents work.

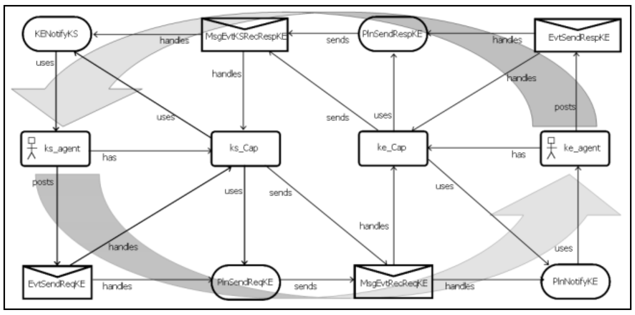


**Figure 9.** Overview of Agent Action Diagram on ks_agent and ke_agent Interaction

## D. Simulation Settings for Knowledge Seeker and Knowledge Collaborator Agents

Similar to the previous section, the primary actor in this simulation is the Knowledge Seeker agent (i.e. ks_agent), that initiates the whole simulation process or reacts to the need of locating Knowledge Expert from the human Knowledge Seeker.

The Knowledge Seeker (KS) can send direct request to Knowledge Collaborator (KC) via the interaction. The KS's agent (ks_agent) carries out the task by executing PlnSendReqKC plan, with its capability ks_Cap. In order to perform the task of sending request to KC, ks_agent posts the EvtSendReqKC event that handles the plan (i.e. PlnSendReqKC), which sends the message MsgEvtRecReqKC event for the KC agent (kc_agent) to pick up.

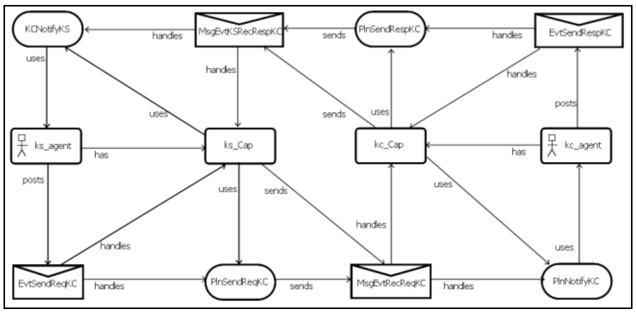


**Figure 10.** Overview of Agent Action Diagram on ks_agent and kc_agent Interaction

The kc_agent has the capability (kc_Cap) of notifying the human Knowledge Collaborator (using the PlnNotifyKC plan) and sending response on behalf of the human Knowledge Collaborator (using the PlnSendRespKC plan). Upon receiving the request from ks_agent, kc_agent executes the plan PlnNotifyKC to inform its human counterpart. As the KC accepts or acknowledges this request, kc_agent executes the PlnSendRespKC to inform ks_agent. The kc_agent posts the event EvtSendRespKC that executes the PlnSendRespKC to send the message MsgEvtKSRecRespKC to the ks_agent. The ks_agent informs the Knowledge Seeker on the response received from Knowledge Collaborator by executing the plan called KCNotifyKS.

The whole process of interaction between ks_agent and kc_agent is illustrated in Figure 10. Table 3 summarises the elements in the kc_agent environment, based on the agent action diagram in Figure 10.

| Name / Role | Code Name | Description |
|---|---|---|
| Knowledge Collaborator (KC) | kc_agent | Knowledge Collaborator's agent that mediates the human Knowledge Collaborator |
| KC's Capability | kc_Cap | Knowledge Collaborator's capability |
| KC's Plan | PlnNotifyKC | Knowledge Collaborator's plan to notify the Knowledge Collaborator on the request along with the information about Knowledge Seeker's bio data |
| | PlnSendRespKC | Knowledge Collaborator's plan to respond to the Knowledge Seeker's request. |
| KC's Event | EvtSendRespKC | An event of Sending Response to Knowledge Seeker |
| | MsgEvtKSRecRespKC | An event of Receiving Response Message that is sent to Knowledge Seeker |

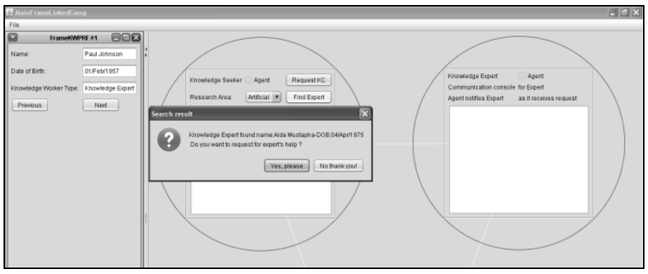
*Table 3.* Simulation settings of elements in kc_agent environment.

## VI. The Physical Model and Interface Design

The physical model of ks_agent and ke_agent is coded in Java as a method of the Knowledge Seeker and Knowledge Expert agent interaction. The ability to incorporate standard Java code to provide this functionality is a positive aspect of the JACK environment. In demonstrating the interaction simulation in a graphically viewable format, the standard GUI (Graphical User Interface) components in Java are used, whose graphics context provides the capabilities of drawing on the screen, as well as interacting with the underlying operating system to perform the drawing [15]. A simple graphical display is coded in Java, since the JACK Intelligent Agents framework is also based on Java.

For the first case scenario, Figure 11 shows the graphical display or interface design of the simulation, consisting two nodes in circular shapes: Knowledge Seeker and Knowledge Expert. The overall interface includes two frames: the right frame displays the simulation in nodal form; and the left frame displays the records of each profile stored in the database. The profile data includes the research area which is the Knowledge Seeker's search criterion.

The search for Knowledge Expert is based on Research Area. For the purpose of simulating the agents interaction (i.e. limited functions are provided for the user or Knowledge Seeker to key in), we list the Research Area in a combo box for a quick selection on the Research Area options. Once the research area is selected, clicking the "Find Expert" button runs the search on the keyword.

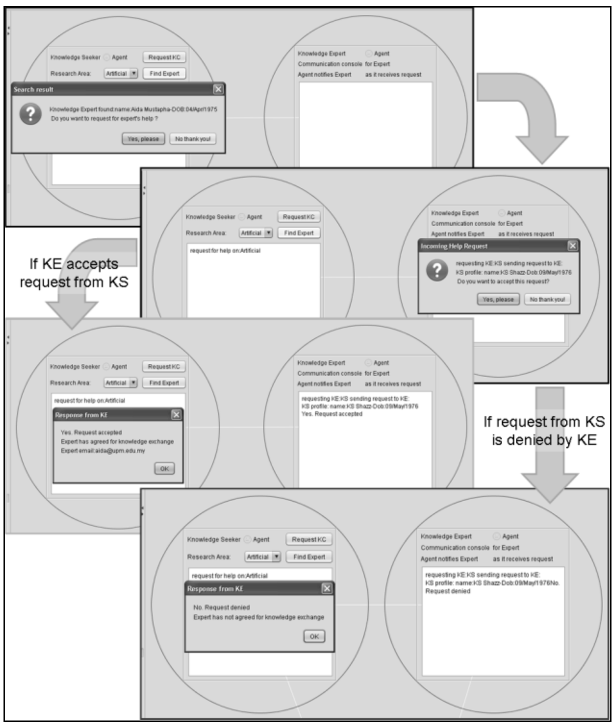


Name: Paul Johnson
Date of Birth: 01/Feb/1957
Knowledge Worker Type: Knowledge Expert
Research Area: Artificial

Knowledge Expert found:name:Aida Mustapha-DOB:04/Apr/1975
Do you want to request for expert's help ?

**Figure 11.** Interface Design for Knowledge Seeker – Knowledge Expert Interaction Prototype

Figure 12 shows the simulation flow for the Knowledge Seeker-Knowledge Expert interaction. In the case of a Knowledge Seeker looking for experts, if an expert is found and it is reported back to the seeker, the ks_agent would send a response message, "Knowledge Expert found: name: [expert's name] - DOB: [expert's date of birth]. Do you want to request for expert's help?" The human Knowledge Seeker needs to respond with a Yes/No option that activates the connection and a question message attached to the connection action. The Knowledge Expert receives this request from the seeker with a message that starts with, "Request for help on: [search keyword]", followed by the content of the message. Upon receiving request from the Knowledge Seeker, the Knowledge Expert has the option to accept or deny the request. The connection interaction between the Knowledge Seeker and the Knowledge Expert is recorded in log file.



Research Area: Artificial

Knowledge Expert found:name:Aida Mustapha-DOB:04/Apr/1975
Do you want to request for expert's help ?

request for help on:Artificial

requesting KE:KS sending request to KE:
KS profile: name:KS Shazz-Dob:09/May/1976
Do you want to accept this request?

Yes. Request accepted
Expert has agreed for knowledge exchange
Expert email:aida@upm.edu.my

requesting KE:KS sending request to KE:
KS profile: name:KS Shazz-Dob:09/May/1976
Yes. Request accepted

No. Request denied
Expert has not agreed for knowledge exchange

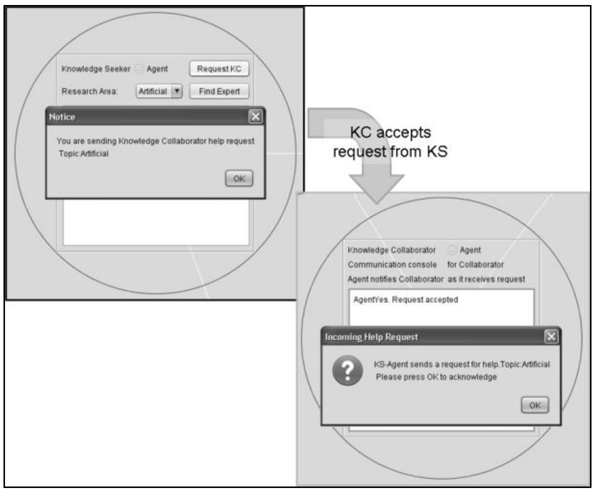
requesting KE:KS sending request to KE:
KS profile: name:KS Shazz-Dob:09/May/1976No.
Request denied

**Figure 12.** Simulation Flow for Knowledge Seeker-Knowledge Expert Interaction

For the second scenario, the physical model of ks_agent and kc_agent is also coded in Java as a method of the Knowledge Seeker and Knowledge Collaborator agent interaction. The interface design is developed in the same environment as the simulation for ks_agent and ke_agent (as discussed earlier). In other words, the interaction simulation of the ks_agent and kc_agent is also coded in Java with the standard Java GUI components.

The search is also based on Research Area, via a selection from the Research Area combo box. Once the research area is selected, the "Request KC" button is used to run the search on the keyword, which activates the agent to request for experts from the Knowledge Collaborator. As shown in Figure 13, the Knowledge Seeker is notified with a message, "You are sending Knowledge Collaborator help request Topic [search keyword]".

The simulation is a time-step simulation with the time rate of 600 milliseconds set for an iterative process that keeps on synchronising the agents' information and checking for new incoming requests for knowledge expertise search in the environment. If there is no request, the log file records, "Check for incoming request. No incoming request." Upon receiving an incoming request, the Knowledge Collaborator can view the message, "KS-Agent sends a request for help. Topic: [message keyed in by Knowledge Seeker]. Please press OK to acknowledge."



You are sending Knowledge Collaborator help request
Topic:Artificial

Communication console for Collaborator
Agent notifies Collaborator as it receives request

AgentYes. Request accepted

KS-Agent sends a request for help.Topic:Artificial
Please press OK to acknowledge

**Figure 13.** Simulation Flow for Knowledge Seeker-Knowledge Collaborator Interaction

## VII. Conclusion and Further Work

This paper presents the proof-of-concept of a PKM (GUSC) model by simulating a personal knowledge management process of locating experts. It is an extension to our previous work in [16]. The proof entails the deployment of a multi-agent system that animates the tasks of a knowledge seeker in connecting (C) to a knowledge expert (and via a knowledge collaborator) and the ensuing get (G), share (S) and understand (U) actions following the connection. The result of the simulations demonstrates that the simulation and animation of the tasks validates the GUSC model.

Even though the G, U, S and C are shown clearly only in the early requirements analysis as soft and hard goals, it is part of the whole process of expert locating mission that the system is built for. The GUSC Model is still applied in other forms despite goals, such as agent roles, where the agents' tasks could be further defined according to the roles or objectives like "get expert's profile", "share expert's profile with seeker", and "understand recommendation points". Further development of the Expert Locator System may look into these aspects in greater details.

In a nutshell, this is the first step in proving that agents can be animated to perform real humans' tasks, especially in complementing OKM. It is based on the understanding of knowledge workers' way of managing their personal knowledge by reaching out to knowledge experts. In making the agents fully perform and are trusted to perform the tasks will require a long-term exploration in a real organisation environment.

## References

[1] S. Ismail, and M.S. Ahmad. "Emergence of Social Intelligence in Social Network: A Quantitative Analysis for Agent-mediated PKM Processes". *In Proceedings of the IEEE International Conference on Information Technology and Multimedia (ICIMµ 2011)*, Nov. 2011.

[2] S. Ismail, and M.S. Ahmad. "Emergence of Personal Knowledge Networks in Agent-mediated PKM Processes: A Qualitative Analysis in Malaysian Context". *In Proceedings of the International Conference on Computer and Information Science (ICCIS2012)*, Jun. 2012.

[3] S. Ismail, and M.S. Ahmad. "Effective Personal Knowledge Management: A Proposed Online Framework", *World Academy of Science, Engineering and Technology (WASET)*, 72, pp. 542-550, 2012.

[4] F. Huber. "Contextualising the Role of Extra-Firm Personal Networks as a Source of Work-Related Knowledge". *In Proceedings of the Organisational Learning, Knowledge and Capabilities (OLKC) Conference*, 2011.

[5] F. Huber. "On the Socio-spatial Dynamics of Personal Knowledge Networks: Formation, Maintenance and Knowledge Interactions", *Environment and Planning A*.

[6] J. M. Pujol, R. Sanguesa, and J. Delgado. "Extracting Reputation in Multi Agent Systems by Means of Social Network Topology". *In Proceedings of the AAMAS'02*, 2002.

[7] J. Grundspenkis. "Agent based approach for organization and personal knowledge modelling: Knowledge management perspective", *Journal of Intelligent Manufacturing*, 18, pp. 451-457, 2007.

[8] S. Ismail, and M.S. Ahmad. "Emergence of Personal Knowledge Management Processes within Multi-agent Roles", in *PKAW 2012, LNAI 7457*, D. Richards and B.H. Kang (eds.), Springer, Berlin, pp. 221–228, 2012.

[9] M.H. Coen. *SodaBot: A Software Agent Construction System*, MIIT AI Laboratory, Cambridge, 1991.

[10] D. Gilbert, M. Aparicio, B. Atkinson, S. Brady, et al. *IBM Intelligent Agent Strategy*, 1995.

[11] N.R. Jennings, P. Faratin, A.R. Lomuscio, S. Parsons, C. Sierra, M. Wooldridge. "Automated Negotiation: Prospects, Methods and Challenges", *International Journal of Group Decision and Negotiation*, pp. 1-30, 2000.

[12] AOS Group. “JACK”, *AOS: Autonomous Decision-Making Software*, 2011.

[13] L. Padgham, D. Scerri, G. Jayatilleke, S. Hickmott. “Integrating BDI Reasoning into Agent Based Modeling and Simulation”. *In Proceedings of the 2011 Winter Simulation Conference*, S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, and M. Fu (eds.), pp. 345-356, 2011.

[14] M. Kolp, P. Giorgini, J. Mylopoulos. “Information Systems Development through Social Structures”. *In Proceedings of the ACM 14th International Conference on Software Engineering and Knowledge Engineering (SEKE ‘02)*, Jul. 2001, pp. 183-190, 2001.

[15] H.C. Chua. “Customer Graphics Programming – Java Programming Tutorial”, *Yet Another Insignificant... Programming Notes, 2013*, http://www.ntu.edu.sg/home/ehchua/programming/java/J4b_CustomGraphics.html.

[16] S. Ismail, T.D. Nguyen, and M.S. Ahmad. “A Multi-agent Knowledge Expert Locating System: A Software Agent Simulation on Personal Knowledge Management (PKM) Model”. *In Proceedings of the International conference on Intelligent Systems Design and Applications (ISDA)*, Dec. 2013.

## Author Biographies

**Shahrinaz Ismail** Born in Kedah, Malaysia, on May 9th, 1976, she earned a Bachelor of Applied Science with Honours in mathematical modelling and computer modelling, from University of Science, Malaysia (USM), Malaysia in 1999. She continued her study in Master of Science in Information Technology in Business (IT in Business) from University of Lincoln, United Kingdom in 2004, as part-time during her career in a Malaysia based quantity surveying firm. She completed her Doctor of Philosophy in Information and Communication Technology (ICT) from Universiti Tenaga Nasional (UNITEN), Malaysia, in July 2014, which was also a part-time basis along with her lecturing career. Her major field of study is agent-mediated knowledge management and technology in education. She is currently into a new interest of artificial intelligence in quality management system.

**Mohd Sharifuddin Ahmad** Born in Johor, Malaysia, on April 30th, 1957, he received his Bachelor of Science in Electrical and Electronic Engineering from Brighton Polytechnic, UK in 1980. He started his career as a power plant engineer specialising in Instrumentation and Process Control in 1980. After completing his Master of Science in Artificial Intelligence from Cranfield University, United Kingdom in 1995, he joined UNITEN as a Principal Lecturer and Head of Department of Computer Science and Information Technology. He obtained his PhD from Imperial College, London, United Kingdom in 2005. He has been an Associate Professor at UNITEN since 2006. His research interests includes applying constraints to develop collaborative frameworks in multi-agent systems, collaborative interactions in multi-agent systems and tacit knowledge management using artificial intelligence (AI) techniques.