MASK-SM: Multi-Agent System Based Knowledge Management System to Support Knowledge Sharing of Software Maintenance Knowledge Environment

Amir Mohamed Talib (Corresponding author)
Faculty of Computer Science & IT, University Putra Malaysia
43400 UPM, Serdang, Selangor, Malaysia
Tel: 60-1-7323-3051 E-mail: ganawa53@yahoo.com

Rusli Abdullah, Rodziah Atan & Masrah Azrifah Azmi Murad
Faculty of Computer Science & IT, Information System Department
University Putra Malaysia, 43400 UPM, Serdang, Selangor, Malaysia
E-mail: rusli@fsktm.upm.edu.my, rodziah@fsktm.upm.edu.my, masrah@fsktm.upm.edu.my

Abstract

Knowledge management (KM) has become an important topic as organizations wish to take advantage of the information that they produce and that can be brought to bear on present decisions. This paper described a system to manage the information and knowledge generated during the software maintenance process (SMP). Knowledge Management System (KMS) is utilizing to help employees build a shared vision, since the same codification is used and misunderstanding in staff communications may be avoided. The architecture of the system is formed from a set of agent communities each community of practice (CoP) is in charge of managing a specific type of knowledge. The agents can learn from previous experience and share their knowledge with other agents or communities in a group of multi-agent system (MAS). This paper also described on the theoretical concept and approach of multi-agent technology framework that could be implemented software maintenance process (SMP) in order to facilitate knowledge sharing among the maintainers of the learning organization. as well as to demonstrate it into the system wise, on how the multi-agent technology could be utilized in the software maintenance process (SMP) system model for serving the maintainer that is developed by using groupware such as Lotus Notes software. This architecture will be named as MASK-SM (MAS Architecture to Facilitate Knowledge Sharing of Software Maintenance). The author followed the Prometheus methodology to design the MAS architecture. This paper applied the definition of ISO 9241-11 (1998) that examines effectiveness, efficiency, and satisfaction. The emphasis will be given to the software maintenance process (SMP) activities that may concern with multi-agent technology to help the maintainers especially in learning organization to work collaboratively including critical success factor in order to ensure that software maintenance process (SMP) initiatives would be delivered competitive advantage for the community of practice (CoP) as well as users of the organization.

Keywords: Multi-agent system, Knowledge management, Software maintenance, Community of practice, Lotus notes, Knowledge sharing, Prometheus design tool, Prometheus methodology and usability

1. Introduction

1.1 Background

MASK-SM (MAS architecture to facilitate knowledge sharing in SM) is Architecture aims to provide facilitating knowledge sharing, supporting the system users to successful access to the system resources and enabling them to extract the knowledge from the SMP warehouse.

MASK-SM architecture has been built by using two layers: agent layer and knowledge layer. This architecture has five agents interface agent, send and receive mail agent, decryption and decryption file agent, file transferring schedule agent and personal agent.

This techniques was inspired by the "there is lack of Architecture of MAS-Based KMS in order to product the sharing of knowledge in SM (MASK-SM Architecture)" and also" there is inconsistency of MAS using in test of its functionality", the MASK-SM model has been developed to solve this problem.

Prometheus have been chosen because (a) it covers start-to-end development stages (b) simple method (c) easy to understand (d) mature or have been described in sufficient detail to be of real use (e) provide tool support and

(f) comprehensive documentation for reference. However, it is not advisable to follow the Prometheus methodology strictly to allow users to choose relevant steps and parts to help solving their problem (Padgham, L., And Winikoff., Winikoff, M., Padgham, 2001). Prometheus methodology primarily takes an implementation point of view and focus heavily on developing a system rapidly. This methodology also falls short in non-functional capability considerations of system development.

An experiment is setup setting based on the proposed usability testing model is discussed. The testing is specially designed to verify the significant of the send and receive mail agent, decryption and decryption file agent and file transferring schedule agent and the algorithms used, and to get user satisfaction on the overall system. The success of system is evaluated through user satisfaction survey which covers (i) File sent (ii) File encrypted/ decrypted (iii) File extracted from SMP data warehouse, and (iv) SMP data warehouse technique.

The main goal of this paper is to improve knowledge sharing of software maintenance by utilizing usability model with the multi-agent technology. More specifically, it proposes a system by designing MASK-SM model by using Prometheus Designing Tool (PDT) and applying the MASK-SM model to a collaborative environment of lotus notes to facilitate the knowledge sharing the software maintenance process (SMP) among the users of the community of practice. The three contributions mentioned in this paper are the identifying of components for the agent-technique-based knowledge sharing system of the SMP, the finding of approaches for MAS techniques which are suitable to be used in the knowledge sharing system of the SMP, and the providing of reasonable background for applying existing usability testing of MAS techniques which is significant in knowledge sharing of SMP through user satisfaction experiment.

1.2 Knowledge Management System

Knowledge comes not only from the expertise of the professionals involved in the process, but it is also intrinsic to the product being maintained, and to the reasons that motivate the maintenance (new requirements, user complains, etc.) and processes, methodologies and tool used in the organization. In addition, during software maintenance (SM) may occur substantial changes? One such example are changes in the maintenance staff (which could mean that the people's expertise changes as well), or the frequency with which each type of maintenance (corrective, perceptive, adaptive or preventive) is carried out. Using a KMS a new knowledge might be produced, thus obtaining the maximum performance from the current information. By reusing information and producing relevant knowledge the high costs of SM could also be decreased (De Looff L, 1990).

KM defines as a discipline that promotes an integrated approach to identifying managing and sharing of all of an enterprise’s information assets. These information assets may include database documents, policies procedures as well as previously unarticulated expertise and experience resident in individual workers. KM issues include developing, implementing and maintaining the appropriate technical and organizational infrastructure to enable knowledge sharing (GartnerGroup, 2005).

SMP-Based KMS communities contain the software engineers, software developers, workers’ knowledge or (system’s users) and maintenance engineers in order to facilitate knowledge sharing among CoP. SMP involve many activities in which different people intervene. Each person has partial information that is necessary to other members of the group. If the knowledge only exists in the software engineers and there is no system in charge of transferring the tacit knowledge (contained in the employees) to explicit knowledge (stored on paper, in files, etc) when an employee abandons the organization part of the intellectual capital goes with him/her. When this occurs in an organization involved in SM the end effect is a loss in intellectual capital and increased maintenance effort and costs. Unfortunately, this is often the case. Another well-known issue that complicates the maintenance process is the scarce documentation that exists related to a specific software system, or even if detailed documentation was produced when the original system was developed, it is seldom updated as the system evolves. For example, legacy software from other units often has not documentation which describes the features of the software (De Looff L, 1990).

For an organization that deals with SM, an interesting alternative is to have KMS which stores explicit knowledge and enables the organization to own its intellectual capital and share it with the sub-units. Otherwise, the software developers own this information and the company depends on them. Another advantage of using a KMS is that it reduces the time that a person needs to mature professionally because it favors the professional development of employees and in this way increases the intellectual capital of the organization. With the passing of time, workers acquire knowledge which they do not normally pass on to his/her peers in the same area (let alone to workers in different areas (De Looff L, 1990).

For this reason, different solutions are often used in order to solve the same problem. Using a KMS which acquires workers’ knowledge and transmits it, the above mentioned situation would decrease since all workers could benefit from other employees' experience and the organization would increase its expertise and coherence of information.

With a KMS the staff may also be informed about the location of information. It is critical for maintenance engineers to have access to the knowledge the organization has carried out a study which found that the number one barrier to knowledge sharing was "ignorance": the sub-units are ignorant of the knowledge that exists in the organizations, or the sub-units possessing the knowledge are ignorant of the fact that another sub-unit needs such knowledge. Sometimes the organization itself is not aware of the location of the pockets of knowledge or expertise (Orton, J.D., & Weick, K.E, 1990).This fact has been summarized by management practitioners as "the left hand not only does not know what the right hand is doing, but it may not even know there is a right hand" (Szulanski, G, 1994).

KMS also help employees build a shared vision, since the same codification is used and misunderstanding in staff communications may be avoided. Several studies have shown that a shared vision may hold together a loosely coupled system and promote the integration of an entire organization (International Standards Organization ISO 9241-11, 1998).

The above explained issues motivated us to design a KMS for capturing, managing, and disseminating knowledge in a SM organization, thus increasing the workers' expertise, the organization's knowledge and its competitiveness while decreasing the costs of the SMP. To have a shared vision of the maintenance process it is advisable to define a conceptualization of the domain. An explicit specification of such conceptualization is ontology (Hoffer , J. George, J. & Valchich, J, 2005).

Normally, programmers will start creating the knowledge that being proposed for a certain project in the organization. When he or she has finished depositing knowledge into the knowledge repositories, the system will trigger the event and pass it to any member specified through e-mail system. This notification will be done based on previous record in order to make sure alerts could be done to those who are interested in the particular knowledge in order to make a decision. Otherwise, this knowledge will be un-meaningful for the other member (Ginsawat, R., Abdullah, R. & Nor, M. Z, 2009). When the system analysts or supervisors also want to make a decision, they should open their mailboxes and look on the subject matter. If they are willing to know about the detail of the knowledge created, they are asked to enter the username and password for security purposes. At the same time another agent will work by updating the status of accessing document as users who are interested with the subject matter (Ginsawat, R., Abdullah, R. & Nor, M. Z, 2009)

CoP is groups of people who share a concern, set of problems, or a passion about a topic, and who deepen their knowledge and expertise in this area by interacting on an ongoing basis (Wenger, E, 2002).

1.3 Multi-Agent System to Facilitate Knowledge Sharing in Software Maintenance Environment

The changeable character of the SMP requires that the information generated be controlled, stored, and shared. We proposed in order to manage the knowledge generated during maintenance a MAS formed of five agents are under the client agents implementation. One agent, called the send and receive mail agent, is in charge of organizing the information sent and received from the group. The second agent is scheduler agent and the third agent for the security. The rest of the agents are also communicated, thus enabling them to interchange information. The main goal of this paper is to design and applying MAS techniques-based KMS in a collaborative environment of lotus notes to facilitate knowledge sharing of SMP among the users of the community of practice. This techniques was inspired by the "there is lack of model of MAS used in SM in order to product the sharing of knowledge in SMP" and also" there is inconsistency of MAS using in test of its functionality"

The rest of the agents are also communicated, thus enabling them to interchange information. The roles of these agents are summarized as follows:

- Comparing new information with that which has already been stored in order to detect inconsistencies between old and new information. If an inconsistency is detected the agent must inform the rest of the agents in order to discover why the inconsistency has occurred.
- Informing other agents about changes produced.
- Predicting new client's demands. Similar software projects often require similar demands. What a company has done before tends to predict what it can do in the future (Gupta & Govindarajan, 2000).

- Predicting possible mistakes by using historic knowledge. Since, as (Henninger & Schlabach, 2001) claim, KM avoids the repetition of common mistakes.
- Advising solutions to problems. Storing solutions that have worked correctly in previous situations helps to avoid the effect that (Zell, 2001) comments upon, indicating that due to the limited transfer of knowledge companies are forced to reinvent new practices, resulting in costly duplication of effort. The best practices often linger in companies for years unrecognized and unshared (Zell, 2001).
- Helping to make decisions. For instance to evaluate whether it is convenient to outsource certain activities. When knowledge is enhanced it is easier to improve problem identification, development of alternative solutions and the selection of the best solution (Gnyawali, Stwart & Grant, 1997).
- Advising certain employee to do a specific job. The system has information about each employee's skills, their performance metrics, and the projects they have worked on. Agents may process this information to suggest which person is most suitable to carry out a task.
- Estimating the cost of future interventions. Information available may be used to make statistical analyses that help predict effort and costs.

2. Literature Review

Knowledge about agent concept alone is not sufficient to build a good agent system. There are some fundamental issues needed to drive the design of an agent (Bigus, J. P., Bigus, J., 2001). The first is to view the agents as adding value to a single standalone application, or as a freestanding community of agents that interact with each other and other applications. The first type views the agent from the perspective of application-centric, where the agents are helpers to the application, while the second is more agent-centric, where the agents monitor and drive the application.

In recent years, Multi-Agent System (MAS) has been an active research topic. Due to the difficulties in solving process planning and production scheduling problems using traditional centralized problem solving methodology, MAS approach – a distributed problem-solving paradigm is used as another attempt to solve the planning and scheduling problems. As a distributed problem-solving paradigm, MAS breaks complex problems into small and manageable sub-problems to be solved by individual agents co-operatively (Vermeulen, S. Bohte, D. Somefun & Poutré J. L, 2006).

Agent paradigm lets users think in term of agents rather than objects / functions. The agent exhibits presents high dependencies compared with an object-oriented approach. Such a software application needs an appropriate software development method. An analysis and design methodology is intended to assist first in gaining understanding of a particular system, and secondly in designing it (Wooldridge, M, 2004). There are few choices of agent-oriented methodologies to help software engineers to specify, design and build agents to achieve the system's goals.

(Dignum, V., 2006) proposed Operation per Organizations (OperA), a model for agent's organization, society and interaction model. The Organizational Model implements the desired organizational structure of an agent society, the description of an agent population that will enact the roles described in the structure is detailed in the Social Model, and the specification of agent interactions to achieve the desired society global objectives is described in the Interaction Model. However, this model needs other agent oriented methodology to help designing the system.

(Park, S., Sugumaran, V., (2005) introduced a framework of multi-agent system (MAS) development that considers both functional (services to solve complex problems in distributed environments) and non-functional service (capability to reuse, easy to extend, adapt and process uncertain data) of the system. They also suggested that, in order to develop MAS in a systematic way, system should be analyzed in terms of its ultimate goals and the system should be designed both in the abstract as well as concrete by mapping the goals and the sub-goals to software agents.

(Elst,L. V., Dignum V., & Abecker A., 2004) asserted a three-dimension overview on agent-mediated knowledge management which includes (i) understanding the stage in a system's development process where agents are used (analysis, conceptual design, or implementation) (ii) analyzing the architecture / topology of the agent system, and (iii) identifying KM functionality / application focused on.

MAS developed for job shop scheduling problems in which standard operating procedures are combined with a look-ahead coordination mechanism that should prevent 'decision myopia' on part of the agents. Using their approach, system performance is said to improve in tightly-coupled, real-time job-shop scheduling environments. However, their coordination mechanism is not appropriate for competitive, self-interested agents, which makes it an undesirable choice for coordination in a de-icing setting (Liu & Sycara, K. P, 1996).

Analyzing the structural and representational aspects of large application software, explanation and/or predictions can be made about its maintainability. In that study, maintenance improved significantly when the language used supported (Rombach, H. D, 1987).

Many work s has been on developing the multi-agent system. These works provide useful agent development tools or methodologies. But with the agent system becomes more open and complex how to support the system self-organization when design a multi-agent system is still lack efficient methodologies.

For instance, (AgentBuilder. http://www.agentbuilder.com, (2006)) provides graphical tools for supporting all phases of the agent construction process. Programming software agents is accomplished by specifying intuitive concepts such as the beliefs, commitments, behavioral rules and actions of the agent. AgentBuilder Pro makes it much easier to create, debug and test multi agent systems.

Prometheus have been chosen compared to other four methodologies (Tropos, MaSE, Use-Case BDI and Gaia) because (a) it covers start-to-end development stages (b) simple method (c) easy to understand (d) mature or have been described in sufficient detail to be of real use (e) provide tool support and (f) comprehensive documentation for reference.

However, it is not advisable to follow the Prometheus methodology strictly to allow users to choose relevant steps and parts to help solving their problem (Padgham, L., & Winikoff. M., (2002)); (Winikoff, M., Padgham, L., & Harland, J. (2001)). All the methodologies discussed above (Gaia, Tropos, MaSE, Prometheus, and Use-Case BDI) primarily take an implementation point of view and focus heavily on developing a system rapidly. These methodologies also fall short in non-functional capability considerations of system development.

Prometheus methodology has been chosen to help design the system. We had also adopted ideas proposed by (Park, S., Sugumaran, V., (2005)); (Elst,L. V., Dignum V., & Abecker A., (2004)). This means, our guidelines in designing our agent-based knowledge sharing system now consists of:

i. Covers functionality of the system. This macro-level design would involve analyze the system goals and objectives and identify knowledge management activities to be focused by each agent.

ii. Design each agent by following agent paradigm concept: assuring each agent to be autonomous and identify if the agent should be reactive, proactive and intelligent. This would need to study which artificial intelligent techniques would be suitable.

iii. Identify the agent-based environment requirements: percept, action, event, plan and communication protocol.

iv. Covers non-functionality of the agent-based system. This micro-level design would involve specification of the agent coordination and topology

In order to design functionality and goal description required in Prometheus system specification phase, an analysis of the agent-based knowledge sharing requirements have been carried out. The requirements were then aligned to technology and agent support. The following table aligns knowledge sharing process, requirements, user requirement, and technology and agent support.

Usability testing is a multidimensional construct and can be assessed using various criteria. This paper applies the definition of (Nielsen J, 1993) that examines effectiveness, efficiency, and satisfaction. Usability testing model is to explain how to identify the information which is necessary to take into account when specifying or evaluating usability testing of a visual display terminal in terms of measures of user performance and satisfaction. Guidance is given on how to describe the context of use of the product (hardware, software or service) and the relevant measures of usability testing in an explicit way. The guidance is given in the form of general principles and techniques, rather than in the form of requirements to use specific methods (Nielsen J, 1993).

The guidance in ISO 9241-11 (1998) can be used in procurement, design, development, evaluation, and communication of information about usability. ISO 9241-11 (1998) includes guidance on how the usability of a product can be specified and evaluated. It applies both to products intended for general application and products being acquired for or being developed within a specific organization (International Standards Organization ISO 9242-11, 1998). ISO 9241-11 (1998) also explains how measures of user performance and satisfaction can be used to measure how any component of a work system affects the whole work system in use. The guidance includes procedures for measuring usability but does not detail all the activities to be undertaken. Specification of detailed user-based methods of measurement is beyond the scope of ISO 9241-11 (1998) (International Standards Organization ISO 9242-11, 1998). According to the benefits and importance of ISO 9241-11 (1998), this paper proposed a testing model for assessing usability of SMP. As reflected in the definition, three central criteria for usability are the effectiveness, efficiency and satisfaction with which users can achieve specified goals (International Standards Organization ISO 9242-11, 1998) as shown in Figure 1 (Note 1).

3. Methodology

As shown in Figure 2 (Note 1) the methodology discusses about the usability testing model used as the methodology to describe that the MAS applied in collaborative environment will help the users according to their needs to support communities in the organization and performance aspects, as well as any other aspects that suggested by the respondents during the survey. Also in these following sections, each criteria of the usability testing model with domino designer, system configuration and others software used in development of the system will be further elaborated in order to achieve its objectives.

Our methodology composed of four main phases as followed:

3.1 Formulate Framework & Develop MASK-SM

3.1.1 MASK-SM designed by Prometheus Design Tool (PDT)

The Prometheus methodology consists of three phases (Padgham. L, & Winikoff. M, 2004):

• **System Specification:** where the system is specified using goals and scenarios; the system's interface to its environment is described in terms of actions, percepts and external data; and functionalities are defined.

• **Architectural design:** where agent types are identified; the system's overall structure is captured in a system overview diagram; and scenarios are developed into interaction protocols.

• Detailed design: where the details of each agent's internals are developed and defined in terms of capabilities, data, events and plans; process diagrams are used as a stepping stone between interaction protocols and plans.

Each of these phases includes models that focus on the dynamics of the system, (graphical) models that focus on the structure of the system or its components, and textual descriptor forms that provide the details for individual entities.

The main purpose of MASK-SM model is to apply intelligent agents in SM environment to support and facilitate knowledge sharing:

- Send and receive mail.
- Encryption and decryption file.
- File transferring schedule.

3.1.1.1 Systems Specifications

3.1.1.1.1 Goals

As shown in Figure 2 (Note 1) there are three main goals for the agents, and how they are achieved, are described as follows:

1) Send and Receive Mail

a) Send mail to the destination user.

b) Receives mail from the source user.

2) Encryption and decryption file

Once mail sent/received, the next goal is to define the file status, the sender must send the file as encrypted file in order to be protected; the receiver must encrypt the file and also the receiver followed the same thing.

3) File transferring schedule.

This in case of the schedule files, eg (meeting, announcement …)

3.1.1.1.2 Scenario

Three different scenarios are identified, as follows as shown in Figure 3, Figure 4 and Figure 5 (Note 1):

1) File sent/ received scenario

2) File encrypted/ decrypted scenario

3) File transferring scheduled scenario

3.1.1.1.3 System Roles

Based on the different functionality/scenarios, different roles may be extrapolated as above as shown in Figure 6 (Note 1).

3.1.1.2 Architectural Design

3.1.1.2.1 Agent Role Coupling Diagram

In term of collaboration and interaction between agents, the links between agents are as shown in the above Agent Acquintance Diagram. In this case, the most active agent is Interface Agent, which serves as intermediary between agents as shown in Figure 7 (Note 1).

3.1.1.2.2 System Overview Diagram

To explain in detail the functionality of each agent, the System Overview Diagram shall be used as shown in Figure 8 (Note 1). The above identifies the Scenarios, the Agents, the Data, the Actions and the messages that are used by all Agents.

3.1.1.3 Detailed Agent Design

3.1.1.3.1 Interface Agent: Interface Agent acts as an effective bridge between the user and the rest of the agents. Such agents actively assist a user in operating an interactive interface as shown in Figure 9 (Note 1).

3.1.1.3.2 Personal Agent: which obtains user profiles and information relevant to user' knowledge that helps to determine the knowledge that each person has or that a person may need as shown in Figure 10 (Note 1).

3.1.1.3.3 Send and Receive Mail Agent: is enables the users to share their knowledge among the groupware due to their emails. This process is provided by this agent. It's also learns about interactions of a user and E-mail application to perform the tasks on E-mail according to the user preferences as shown in Figure 11 (Note 1).

3.1.1.3.4 Encryption and Decryption File Agent: is use to protect the files that existed into knowledge repositories and data sources repositories. The process is happened when the user is willing to retrieve the file from the knowledge repositories or from data sources repositories and ensure that the file is submitted to the file order in a safety mode as shown in Figure 12 (Note 1).

3.1.1.3.5 File Transferring Schedule Agent: is use to extract the file from knowledge repositories or from data sources repositories and applying it to the user that requested to share it with other users. Storing knowledge helps to reduce dependency on key employees because at least some of their expert knowledge has been retained as shown in Figure 13 (Note 1).

3.1.2 MASK-SM Framework Developments

As shown in Figure 14 (Note 1) the process of sharing knowledge start when a user willing to share its knowledge (file) of SMP application. The user login to his/her email and if the file available with the user then the encryption and decryption agent will encrypt/decrypt the file, and send and receive email will activate directly after that the file will be shared (sent) to the requester user, or if the file exist into SMP warehouse then the file transferring agent will activate and the encryption and decryption agent will encrypt/decrypt the file and send and receive email will activate then the file will be shared (sent) to the requester user.

Figure 15 (Note 1), describes the communication between the agent and the whole system among the users mails and also demonstrate the agents into the system.

SMP requires that the information generated be controlled, stored, and shared. We propose in order to manage the knowledge generated during maintenance a MAS formed of three agents are under the client agents implementation. This part contains the description for the applied MAS and describes the name, job, and the language used. First agent, called the send and receive mail agent, is in charge of organizing the information sent and received from the group. The second agent is encryption and decryption agent and the third agent is file transferring scheduled agent. The send and receive agent communicates with the other two agents. When it receives information, it processes it in order to determinate to what agents should the information be sent to. Since specific information may influence the knowledge managed by different agents, the send and receive agent must know the relationship that exists between all the agents. They also mediate communication among people, and this is of prime importance. In this case, agents will act as a communicator for the user that is based on the direction given and produces the result when it is required to do so. Agents also could be categories in terms of its roles in knowledge searching, knowledge monitoring, and others.

3.2 MASK-SM Framework Evaluation

3.2.1 Participants

The respondents including System Analyst, System Developer, Software Engineer, and User and will be chosen to fill the questionnaire of this study. The respondents should be applying the system before solving the questionnaire to be situated.

3.2.2 Procedures

In the beginning, the respondents will receive a short, scripted verbal orientation explaining the purpose of the usability testing. Then they will be asked to complete a short background questionnaire to collect their demographic characteristics. The respondents will be asked to perform a set of information about how to share knowledge using the usability test as a kind of multi-agent technology for SMP. The tasks were written on a sheet of paper that included a space where respondents will be asked to indicate their answers. Once the tasks are completed, respondents will be asked to complete a short participant satisfaction questionnaire to collect and test their own perceptions towards SM.

3.2.3 Tasks

Respondents will complete three tasks:

1) They will complete a background/experience questionnaire that *including name, gender, age, education level, Major/Department, and years of experience.*

They will perform tasks using the questionnaire's sheet.

There is also a post-survey questionnaire that specifically examines MAS techniques. After completing a task, the respondents will ask to rank satisfaction and to write down comments.

3.2.4 Data collection

This evaluation model considers both quantifying elements of performance (experience and experiment) as well as subjective empirical. If the answer is wrong, or he/she not familiar with this question then skip to the second question until all the question will be solved. We will, however, record whether respondents are able to complete tasks successfully. The criteria for successful task completion are:

- ➢ Participant is able to give a correct answer based on his own information about the system. Any guessed or assumed answers, whether correct or not, are not record as successfully completed tasks.
- ➢ Participant is able to give a definite answer to the question. Where respondents indicated they are unsure about the answer or would seek clarification, the task will record as not successfully completed.

3.2.5 Questionnaires

The purpose of the questionnaire is to prove:

- ✓ Handle the interpretation of the term KM and the company's key objective in SM.
- ✓ Handle the aspects that come into play in KM, such as the existence of a strategy, the processes of quality control of data, the content that is being managed, and the functioning of communities of practice.
- ✓ Identify the Multi-Agent technique of willingness of cooperation for research work.
- ✓ Identify the Multi-Agent technique for helping the user according to his needs.

Basically, there are two types of questionnaire that we prepared as part of usability testing for the respondents for the level of the questions is shown in (Appendix B). (Note 4)

3.2.5.1 Pre-Survey questionnaire (background)

A series of questions designed to collect demographic information about the respondents to assess their level of his information about the system is shown in table 1 (Appendix A). (Note 4)

3.2.5.2 Post-Survey questionnaire

After the test subject completed each scenario, he/she should answer a specific questions related to the tasks. To indicate whether the tasks was clear and completed successfully.

After the test subjects complete all the scenarios, he/she will answer thirty six points, eighteen related to KMS and also eighteen related to MAS questionnaires to record user satisfactory.

4. The Integrated Development Environment (IDE) tools of Lotus Notes software

By using the groupware of Lotus Notes (Lotus company, 2007), the best agent technology capability that could be developed is used Java Script programming that comes along with this package. The examples of the IDE and scripting development of interface are shown in Figure 16(a), Figure 16(b), Figure 16(c) and Figure 16(d) as stated (Note 2).

5. Result and Discussions

The result was conducted according to the methodology described in the previous section. It is starts with an overview of data collected by analyzing trends. The Pre-Survey Questionnaire for the respondents shown in (Appendix A). The Post-Survey Questionnaire for both Quantitative and Qualitative MAS questions. Satisfaction is a multi-dimensional construct. This study applies MAS technology to support knowledge sharing of SMP. (Note 4)

5.1 Usability testing for the system

Several types of data were collected to assess user's performance and user's perceptions of negotiating the system. In addition, we examined selected features of the normal lotus notes system to determine their effectiveness. Tasks were deemed to be either completed or not completed.

5.1.1 Effectiveness

As we can notice all respondents were able to complete all the tasks. Effectiveness was measured by the number of tasks successfully completed. For other tasks respondents were able to complete in success percent ranging between (50% & 100%). The Successful task completion for the individual tasks is summarized in Figure 8 and Figure 9 below for the respondents. As we can see the average of successfully completion task are high, according to this results the successfully completion task that presented the effectiveness, achieved correctly. Moreover these results of the tasks successfully completed are high.

Figure 17 (Note 3) shows that Q2, Q6, Q8 and Q9 are completed answered successfully (100%) and the others questions got more than 70%.

Figure 18 (Note 3) shows that Q1, Q5 and Q9 are completed answered successfully (100%) and the others questions got more than 60%.

5.1.2 Efficiency

Completion time is the one factor used for measuring efficiency in this paper. Efficiency was measured by the amount of time taken to complete all tasks. An average of 44 minutes and 3.5 seconds per respondent was taken to complete the all tasks. However, there was much variation among the respondents, for example, the fastest respondent took only 18 minutes and the slowest took 37 minutes and 9 seconds which are about three times longer. Pearson's product-moment correlation analysis was conducted to see if the respondents' completion time is related. The results showed that total completion time is independent, (see Figure 19 and Figure 20) (Note 3).

As a result, efficiency was measured by evaluating completion time used in this survey by each respondent. Respondents who they were familiar with the systems in general tended to use less time to complete their tasks. When the respondent knows how to get the answer, it takes them fewer time while when they don't know how to use the system, they take more time.

5.1.3 Satisfaction

Respondents Satisfaction measured by using the two scales (YES= respondent agreed, NO= respondent not agreed). Satisfaction was measured by a rating scale for several satisfaction elements.

According to the result below, the satisfaction for the respondents were in moderate level.

Figure 21 (Note 3) shows the respondent satisfaction for Quantitative analysis for MAS. The Q2 and Q8 give high satisfaction (Mean=5.10) out of (Maximum =6) and Q6 give high satisfaction (Mean=4.70), and Q7 also give high satisfaction (Mean=4.40) out of (Maximum=5). All respondents feel satisfied with the system when they fail to perform the task correctly.

Figure 22 (Note 3) shows the respondent satisfaction for Qualitative analysis for MAS. Q8 give high satisfaction (Mean=5.40) and Q9 give high satisfaction (Mean=5.50) out of (Maximum =6) and Q7 give high satisfaction

(Mean=4.20) out of (Maximum=5). 30% of respondents feel less satisfied with the system when they fail to perform the task correctly.

Groupware software has been used which is Lotus Notes (Lotus company, 2007). This software provides various types of services for a CoP. The most important service offers by this software or product is particularly in term of SMP warehouse. This service will serve the CoP to share their knowledge and any things that are stored in SMP warehouse. Another common service is agent technology, which it allows people to share the knowledge or notification regarding the latest knowledge of SM in the SMP warehouse at any time and any place. The set of the agents are communicated to show how the system running as shown in section 4 (system development). Usability testing model cafeterias (effectiveness, efficiency & satisfaction) are supporting this idea based to the (Appendix B). (Note 4)

6. Conclusion

SM is one of the most important stages of the software life cycle. This process takes a lot of time and effort. Besides, it generates a huge amount of different kinds of knowledge that must be suitably managed. MAS in charge of managing this knowledge might improve the maintenance process since agents would help developers find information and solutions to problems and to make decisions, thus increasing organization's competitiveness. KMS is a good place where people could share their knowledge between the CoP. In this case, agent's technology is a tool that could be used in order to act on behalf of CoP of SM to do something repetitively and time based system especially in maintaining various types of maintenance such as Adaptive, Perfective, Corrective and Preventive. The agent techniques describes send and receive agent use to enable the user to share their knowledge among their emails and file transferring scheduled agent use to extract the file from the SMP warehouse and encryption/decryption agent use as the security agent to protect the file. Usability testing model use the three main criteria effectiveness was measured by the number of tasks successfully completed, efficiency was measured by amount of time taken to complete the tasks, and satisfaction was measured by a rating scale for several satisfaction elements. We have briefly presented the Prometheus methodology for designing our MAS. The methodology provides detailed guidance in terms of processes as well as notations. It is not intended to be prescriptive, but is rather an approach which has evolved out of experience, and which the authors expect to be further adapted, refined and developed to suit the needs of agent software developers.

References

AgentBuilder. Available at: http://www.agentbuilder.com/(OL) , 2006. (Accessed on April 2009)

Bigus, J. P., Bigus, J. (2001). "Constructing Intelligent Agents Using Java". Second Edition 2001. Wiley Computer Publishing, John Wiley & Sons, Inc.

De Looff, L. (1990). "Information Systems Outsourcing Decision Making": *a Managerial Approach. Hershey, PA: Idea Group Publishing.*

Dignum, V. (2006). "A Model for Organizational Interaction: Based on Agents, Founded in Logic". PHD Thesis, Utrecht University

Elst,L. V., Dignum V., & Abecker A., (2004). "Towards Agent-Mediated Knowledge Management". Achmea & University Utrecht

GartnerGroup. (2005). "Available at: http://www.gartner.com". Accessed on 23/10/08.

Ginsawat, R., Abdullah, R. & Nor, M. Z. (2009). "Applying Knowledge Management System Architecture in Software Maintenance Environment" *Journal of Computer and Information Science.* Vol. 2, No. 4

Gnyawali, D.R., Stewart, A.C., and Grant J.H. (1997). "Creating and Utilization of Organizational Knowledge": An Empirical Study of the Roles of Organizational Learning on Strategic Decision Making", Academy of Management Best Paper Proceedings, pp. 16-20.

Gruber, T. (1995). "Twards Principle for the design of ontologies used for Knowledge Sharing". *International Journal of Human-Computer Studies*, 43(5/6). pp 907-928.

Gupta, A., and Govindarajan, V. (2000). "Knowledge Flows within Multinational Corporations". *Strategic Management Journal*, 21(4), pp. 473-496.

Henninger, S., and Schlabach, J. (2001). "A Tool for Managing Software Development Knowledge", 3ª International.

Hoffer , J., George, J., & Valchich, J. (2005)."*Modern Systems Analysis and Design*". Fourth edition. Upper Saddle River, NJ: Prentice Hall. Design, pp 327- 436.

International Standards Organization (1998). ISO document 9241- Ergonomic requirements for office work with visual display terminals.

Liu & Sycara. K. P. (1996). “Multiagent coordination in tightly coupled task scheduling”. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pages 164-171.

Lotus Company. (Access on August 2007). available at:www.lotus.com

Nielsen J. (1993). “Usability Engineering”, Academic Press, London.

Orton, J.D., & Weick, K.E. (1990). “Loosely coupled systems”: *A reconceputalization. Academy of Management Review, 15-2*

Padgham, L., & Winikoff. M. (2002). “Prometheus: A Pragmatic Methodology For Engineering Intelligent Agents”. RMIT University.

Park, S., & Sugumaran, V. (2005). “Designing Multi-agent Systems: A Framework and Application”. Expert System with Applications 28 259-271. Elsevier Ltd.

Rombach, H. D. (1987). “A Controlled Experiment on the Impact of Software Structure on Maintainability”, IEEE Trans. Software Eng. 13, 344-355.

Szulanski, G. (1994). “Intra-Firm Transfer of Best Practices Project” American Productivity and Quality Centre, Houston, Texas

Vermeulen, S. Bohte, D., Somefun, & Poutré J. L. (2006). “Improving patient activity schedules by multi-agent Pareto appointment exchanging”.

Wenger, E. (2002). ”Communities of Practice: learning, meaning and identity” New York: Cambridge University Press.

Winikoff, M., Padgham, L., & Harland, J. (2001). “Simplifying the Development of Intelligent Agents”. Melbourne, Australia.

Wooldridge, M. (2004). “An Introduction to Multi-Agent Systems” John Wiley & Sons Ltd. (Chapter 2)

Zell, D. (2001). ”Overcoming Barriers to Work Innovations: Lessons Learned at Hewlet-Packard. Organizational Dynamics”, pp 77-86.

Notes

Note 1

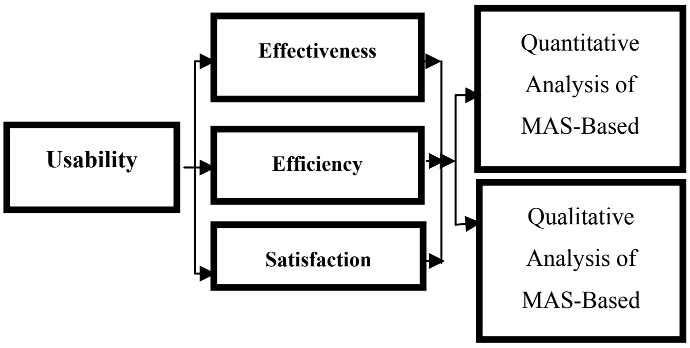
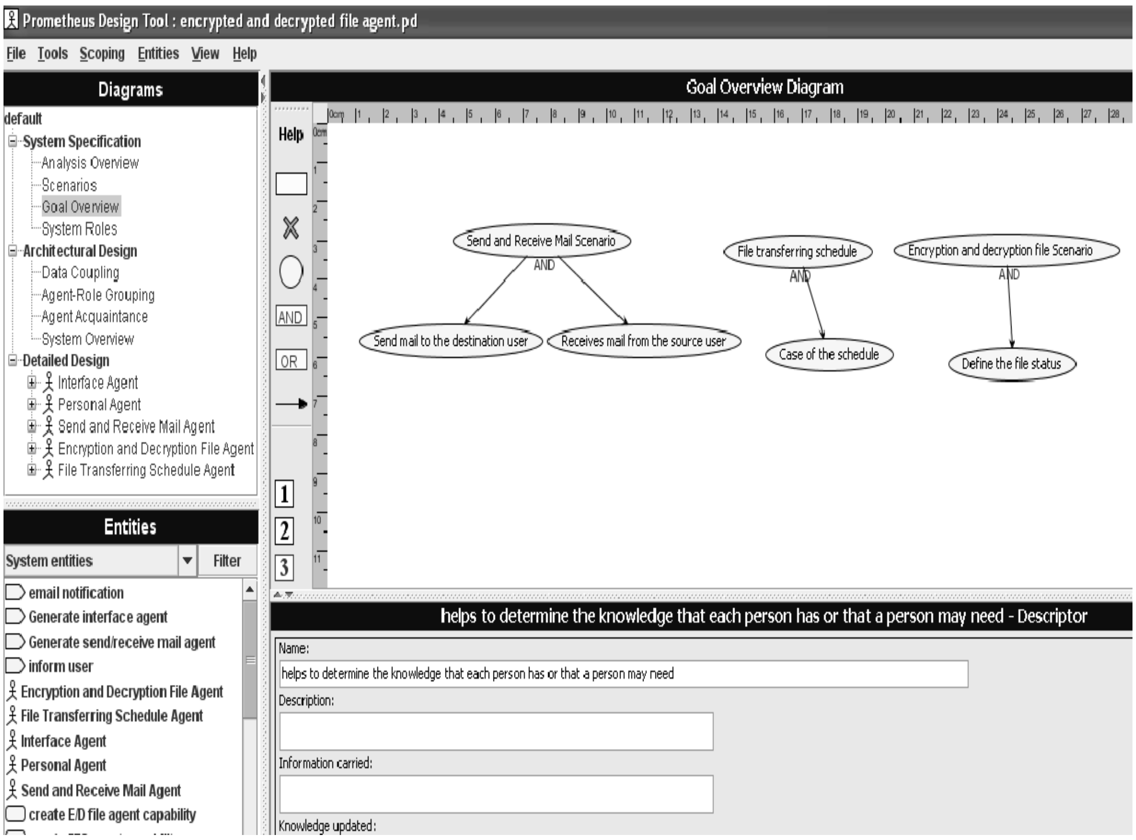

Usability
Effectiveness
Efficiency
Satisfaction
Quantitative Analysis of MAS-Based
Qualitative Analysis of MAS-Based

Figure 1. Utilized Usability Model


Prometheus Design Tool : encrypted and decrypted file agent.pd
File Tools Scoping Entities View Help
Diagrams
default
System Specification
Analysis Overview
Scenarios
Goal Overview
System Roles
Architectural Design
Data Coupling
Agent-Role Grouping
Agent Acquaintance
System Overview
Detailed Design
Interface Agent
Personal Agent
Send and Receive Mail Agent
Encryption and Decryption File Agent
File Transferring Schedule Agent
Entities
System entities
Filter
email notification
Generate interface agent
Generate send/receive mail agent
inform user
Encryption and Decryption File Agent
File Transferring Schedule Agent
Interface Agent
Personal Agent
Send and Receive Mail Agent
create E/D file agent capability
Goal Overview Diagram
Help
AND
OR
1
2
3
Send and Receive Mail Scenario
AND
Send mail to the destination user
Receives mail from the source user
File transferring schedule
AND
Case of the schedule
Encryption and decryption file Scenario
AND
Define the file status

helps to determine the knowledge that each person has or that a person may need - Descriptor

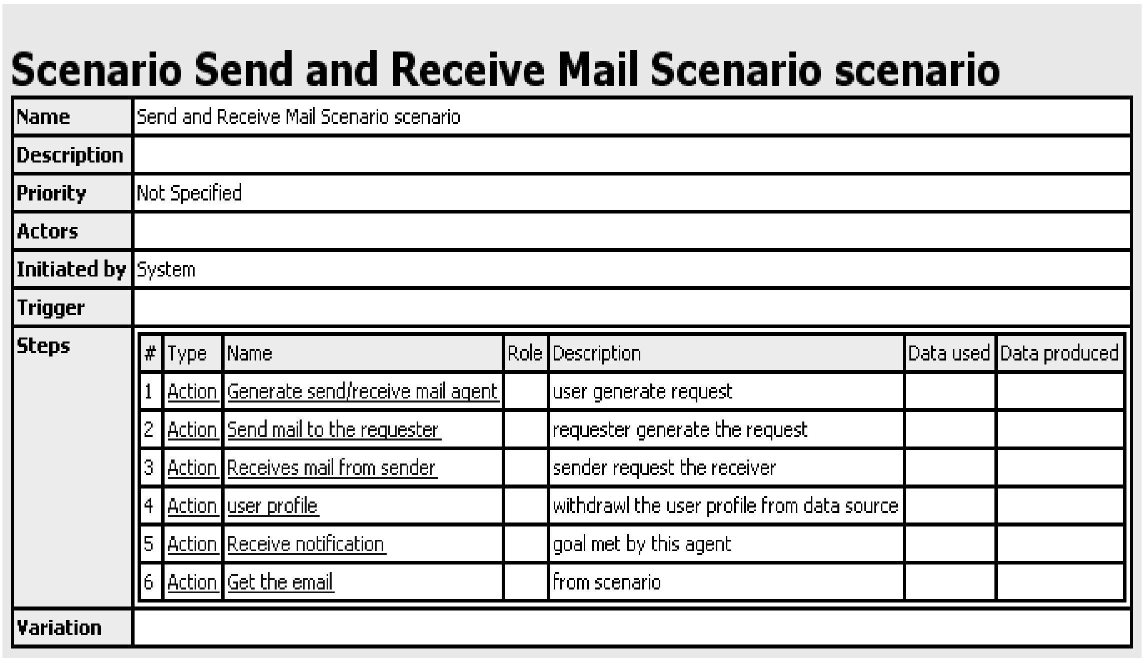
Name:
helps to determine the knowledge that each person has or that a person may need

Description:

Information carried:

Knowledge updated:

Figure 2. Goal Overview Diagram

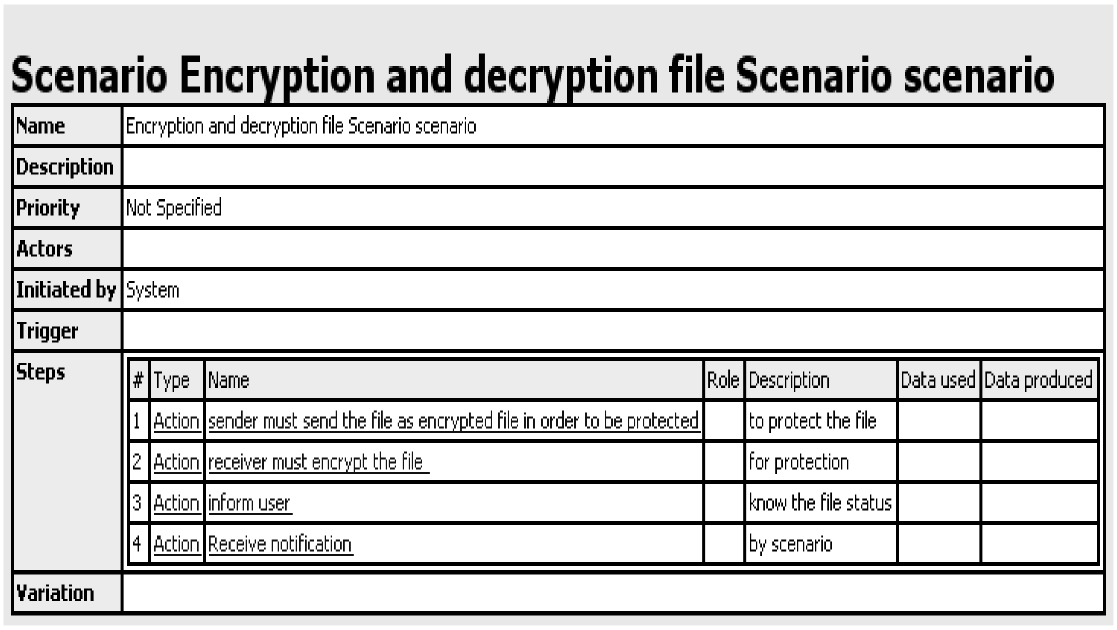


Scenario Send and Receive Mail Scenario scenario

Name	Send and Receive Mail Scenario scenario
Description	
Priority	Not Specified
Actors	
Initiated by	System
Trigger	
Steps	see below
Variation	

#	Type	Name	Role	Description	Data used	Data produced
1	Action	Generate send/receive mail agent		user generate request		
2	Action	Send mail to the requester		requester generate the request		
3	Action	Receives mail from sender		sender request the receiver		
4	Action	user profile		withdrawl the user profile from data source		
5	Action	Receive notification		goal met by this agent		
6	Action	Get the email		from scenario		

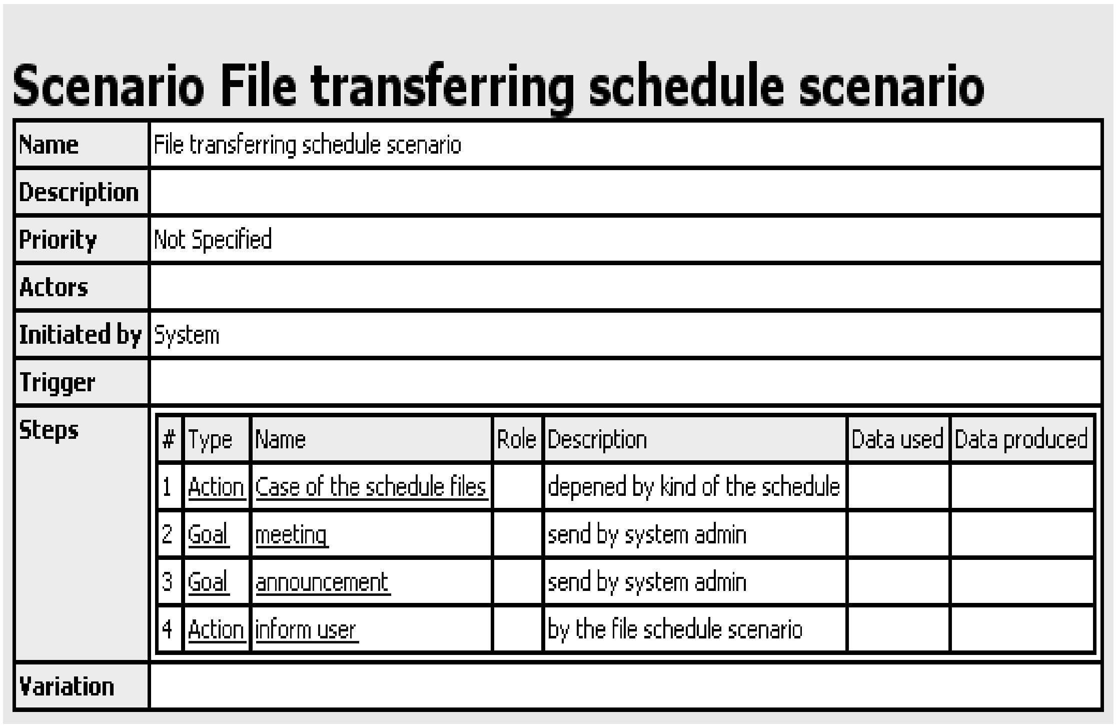
Figure 3. File sent/ received scenario


Scenario Encryption and decryption file Scenario scenario

Name	Encryption and decryption file Scenario scenario
Description	
Priority	Not Specified
Actors	
Initiated by	System
Trigger	
Steps	see below
Variation	

#	Type	Name	Role	Description	Data used	Data produced
1	Action	sender must send the file as encrypted file in order to be protected		to protect the file		
2	Action	receiver must encrypt the file		for protection		
3	Action	inform user		know the file status		
4	Action	Receive notification		by scenario		

Figure 4. File encrypted/ decrypted scenario


Scenario File transferring schedule scenario

Name	File transferring schedule scenario
Description	
Priority	Not Specified
Actors	
Initiated by	System
Trigger	
Steps	see below
Variation	

#	Type	Name	Role	Description	Data used	Data produced
1	Action	Case of the schedule files		depened by kind of the schedule		
2	Goal	meeting		send by system admin		
3	Goal	announcement		send by system admin		
4	Action	inform user		by the file schedule scenario		

Figure 5. File transferring scheduled scenario

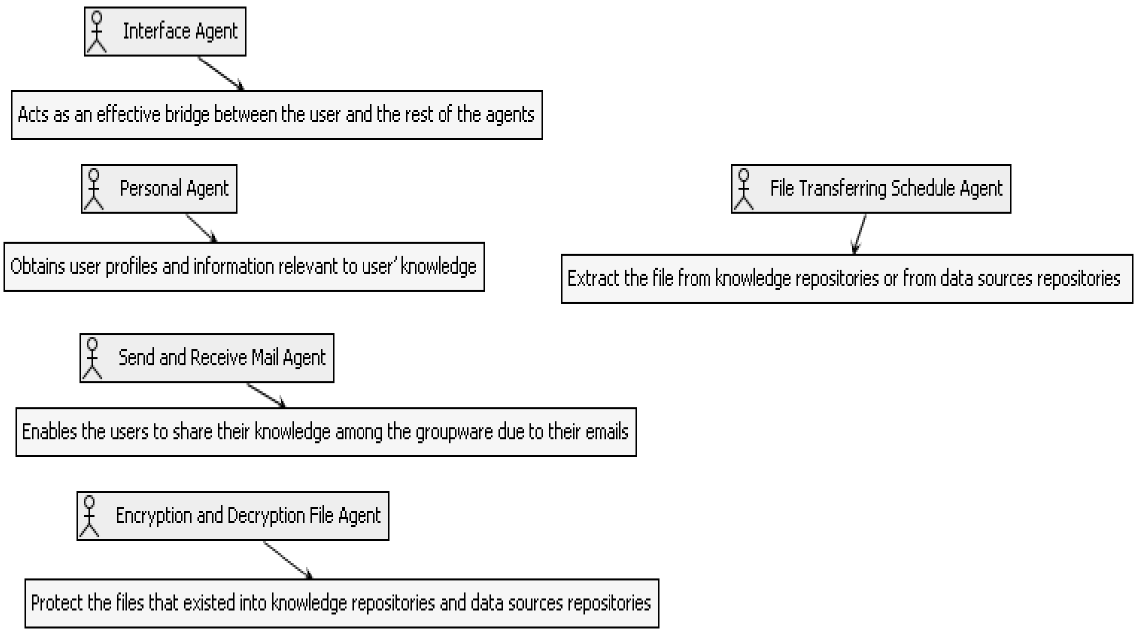
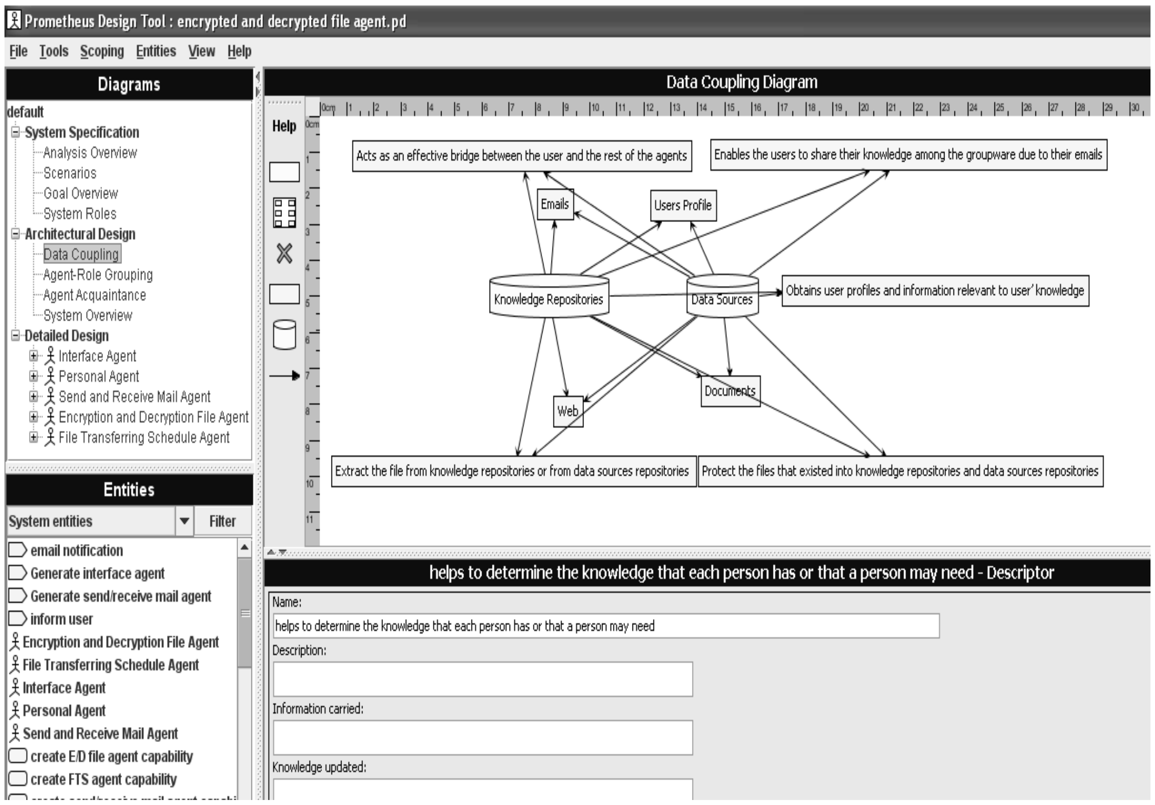

Interface Agent
Acts as an effective bridge between the user and the rest of the agents
Personal Agent
Obtains user profiles and information relevant to user' knowledge
File Transferring Schedule Agent
Extract the file from knowledge repositories or from data sources repositories
Send and Receive Mail Agent
Enables the users to share their knowledge among the groupware due to their emails
Encryption and Decryption File Agent
Protect the files that existed into knowledge repositories and data sources repositories

Figure 6. System Role Diagram


Prometheus Design Tool : encrypted and decrypted file agent.pd
File Tools Scoping Entities View Help
Diagrams
default
System Specification
Analysis Overview
Scenarios
Goal Overview
System Roles
Architectural Design
Data Coupling
Agent-Role Grouping
Agent Acquaintance
System Overview
Detailed Design
Interface Agent
Personal Agent
Send and Receive Mail Agent
Encryption and Decryption File Agent
File Transferring Schedule Agent
Entities
System entities
Filter
email notification
Generate interface agent
Generate send/receive mail agent
inform user
Encryption and Decryption File Agent
File Transferring Schedule Agent
Interface Agent
Personal Agent
Send and Receive Mail Agent
create E/D file agent capability
create FTS agent capability
Data Coupling Diagram
Help
Acts as an effective bridge between the user and the rest of the agents
Enables the users to share their knowledge among the groupware due to their emails
Emails
Users Profile
Knowledge Repositories
Data Sources
Obtains user profiles and information relevant to user' knowledge
Documents
Web
Extract the file from knowledge repositories or from data sources repositories
Protect the files that existed into knowledge repositories and data sources repositories

helps to determine the knowledge that each person has or that a person may need - Descriptor

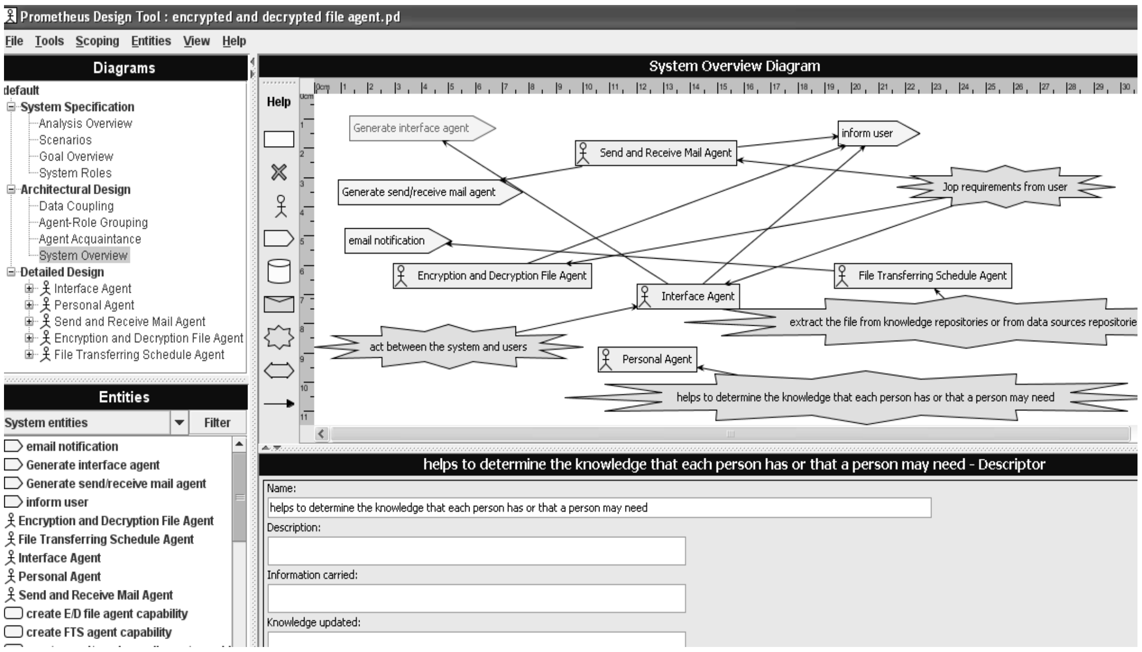
Name: helps to determine the knowledge that each person has or that a person may need

Description:

Information carried:

Knowledge updated:

Figure 7. Agent Role Coupling Diagram


Prometheus Design Tool : encrypted and decrypted file agent.pd
File Tools Scoping Entities View Help
Diagrams
default
System Specification
Analysis Overview
Scenarios
Goal Overview
System Roles
Architectural Design
Data Coupling
Agent-Role Grouping
Agent Acquaintance
System Overview
Detailed Design
Interface Agent
Personal Agent
Send and Receive Mail Agent
Encryption and Decryption File Agent
File Transferring Schedule Agent
Entities
System entities
Filter
email notification
Generate interface agent
Generate send/receive mail agent
inform user
Encryption and Decryption File Agent
File Transferring Schedule Agent
Interface Agent
Personal Agent
Send and Receive Mail Agent
create E/D file agent capability
create FTS agent capability
System Overview Diagram
Help
Generate interface agent
Send and Receive Mail Agent
inform user
Jop requirements from user
Generate send/receive mail agent
email notification
Encryption and Decryption File Agent
File Transferring Schedule Agent
Interface Agent
extract the file from knowledge repositories or from data sources repositorie
act between the system and users
Personal Agent
helps to determine the knowledge that each person has or that a person may need

helps to determine the knowledge that each person has or that a person may need - Descriptor

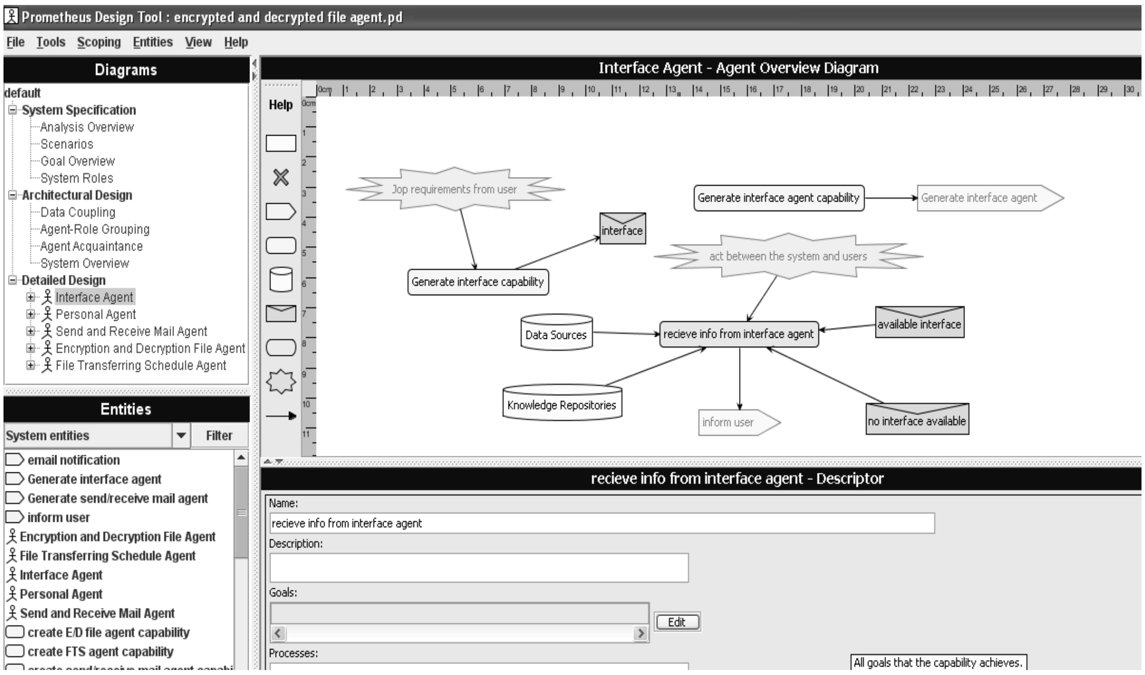
Name: helps to determine the knowledge that each person has or that a person may need

Description:

Information carried:

Knowledge updated:

Figure 8. System Overview Diagram


Prometheus Design Tool : encrypted and decrypted file agent.pd
File Tools Scoping Entities View Help
Diagrams
default
System Specification
Analysis Overview
Scenarios
Goal Overview
System Roles
Architectural Design
Data Coupling
Agent-Role Grouping
Agent Acquaintance
System Overview
Detailed Design
Interface Agent
Personal Agent
Send and Receive Mail Agent
Encryption and Decryption File Agent
File Transferring Schedule Agent
Entities
System entities
Filter
email notification
Generate interface agent
Generate send/receive mail agent
inform user
Encryption and Decryption File Agent
File Transferring Schedule Agent
Interface Agent
Personal Agent
Send and Receive Mail Agent
create E/D file agent capability
create FTS agent capability
Interface Agent - Agent Overview Diagram
Help
Jop requirements from user
Generate interface agent capability
Generate interface agent
interface
act between the system and users
Generate interface capability
Data Sources
recieve info from interface agent
available interface
Knowledge Repositories
inform user
no interface available
Edit
All goals that the capability achieves.

recieve info from interface agent - Descriptor

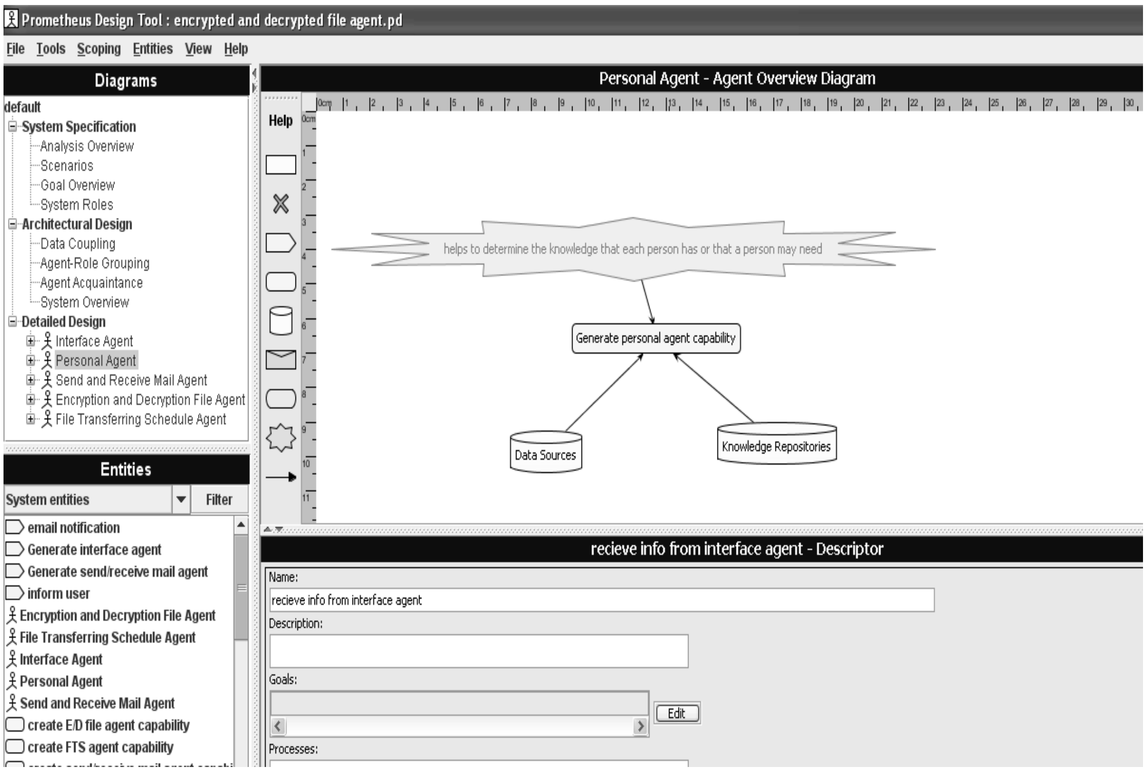
Name: recieve info from interface agent

Description:

Goals:

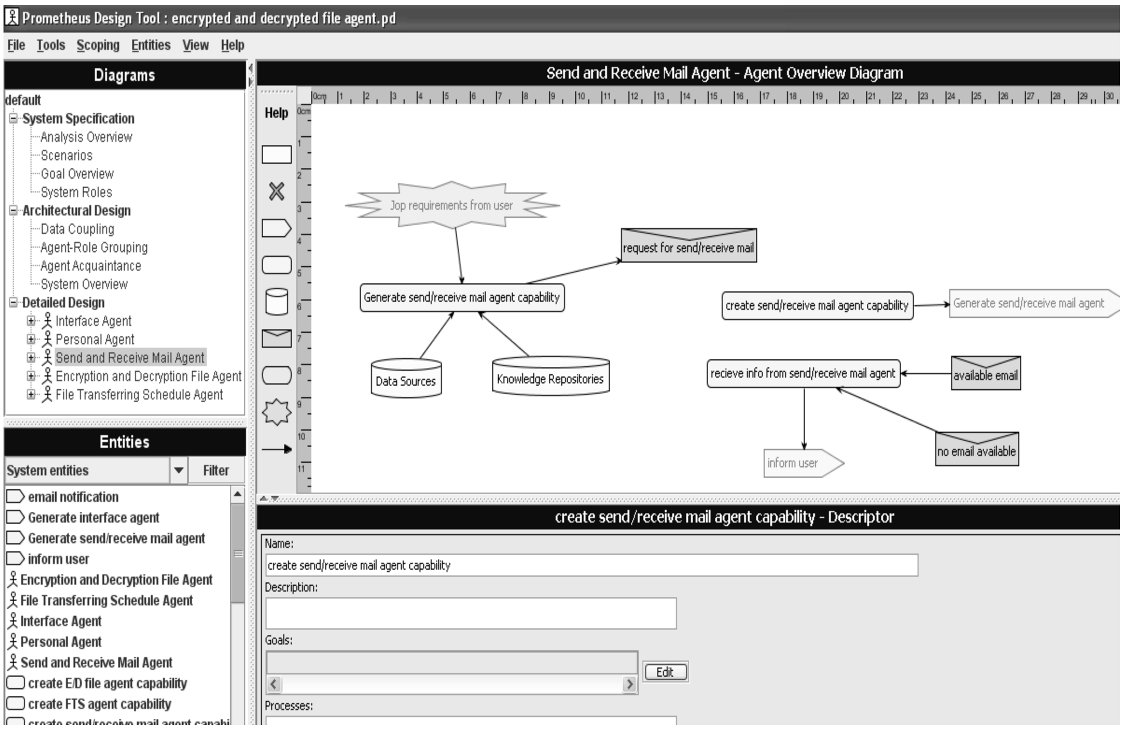
Processes:

Figure 9. Interface Agent


Prometheus Design Tool : encrypted and decrypted file agent.pd
File Tools Scoping Entities View Help
Diagrams
default
System Specification
Analysis Overview
Scenarios
Goal Overview
System Roles
Architectural Design
Data Coupling
Agent-Role Grouping
Agent Acquaintance
System Overview
Detailed Design
Interface Agent
Personal Agent
Send and Receive Mail Agent
Encryption and Decryption File Agent
File Transferring Schedule Agent
Entities
System entities
Filter
email notification
Generate interface agent
Generate send/receive mail agent
inform user
Encryption and Decryption File Agent
File Transferring Schedule Agent
Interface Agent
Personal Agent
Send and Receive Mail Agent
create E/D file agent capability
create FTS agent capability
Personal Agent - Agent Overview Diagram
Help
helps to determine the knowledge that each person has or that a person may need
Generate personal agent capability
Data Sources
Knowledge Repositories
recieve info from interface agent - Descriptor
Edit

Name: recieve info from interface agent
Description:
Goals:
Processes:

Figure 10. Personal Agent


Prometheus Design Tool : encrypted and decrypted file agent.pd
File Tools Scoping Entities View Help
Diagrams
default
System Specification
Analysis Overview
Scenarios
Goal Overview
System Roles
Architectural Design
Data Coupling
Agent-Role Grouping
Agent Acquaintance
System Overview
Detailed Design
Interface Agent
Personal Agent
Send and Receive Mail Agent
Encryption and Decryption File Agent
File Transferring Schedule Agent
Entities
System entities
Filter
email notification
Generate interface agent
Generate send/receive mail agent
inform user
Encryption and Decryption File Agent
File Transferring Schedule Agent
Interface Agent
Personal Agent
Send and Receive Mail Agent
create E/D file agent capability
create FTS agent capability
Send and Receive Mail Agent - Agent Overview Diagram
Help
Jop requirements from user
request for send/receive mail
Generate send/receive mail agent capability
create send/receive mail agent capability
Generate send/receive mail agent
Data Sources
Knowledge Repositories
recieve info from send/receive mail agent
available email
no email available
inform user
create send/receive mail agent capability - Descriptor
Edit

Name: create send/receive mail agent capability
Description:
Goals:
Processes:

Figure 11. Send and Receive Mail Agent

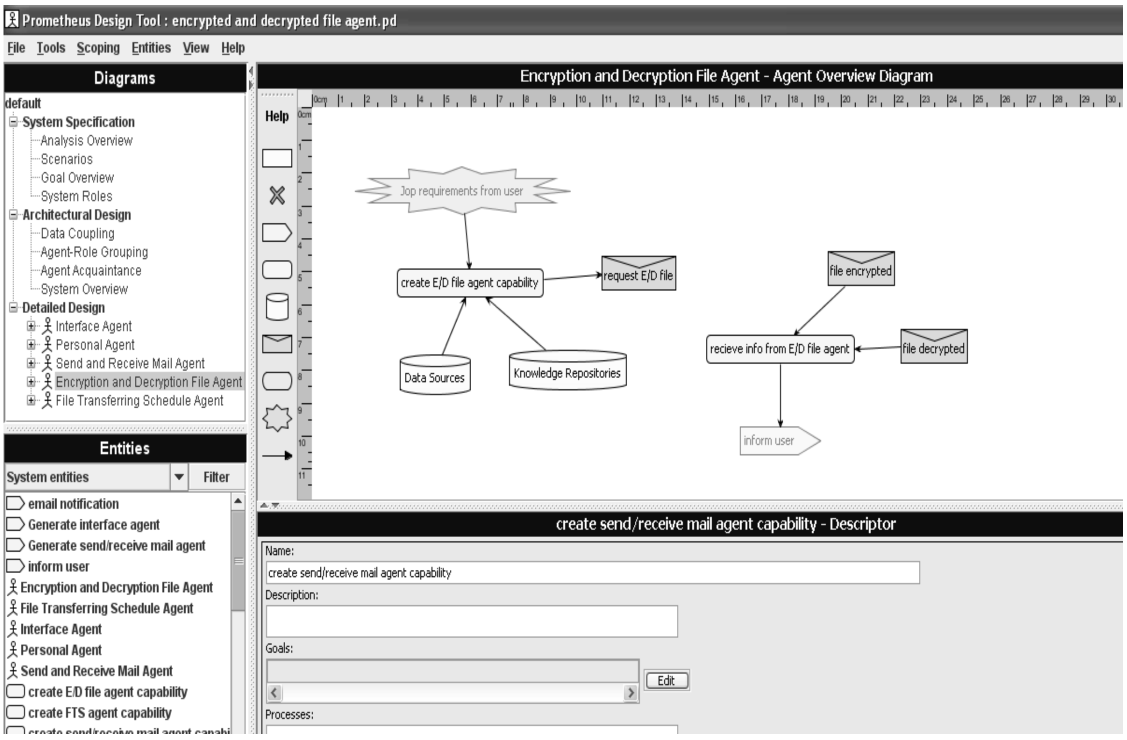

Prometheus Design Tool : encrypted and decrypted file agent.pd
File Tools Scoping Entities View Help
Diagrams
default
System Specification
Analysis Overview
Scenarios
Goal Overview
System Roles
Architectural Design
Data Coupling
Agent-Role Grouping
Agent Acquaintance
System Overview
Detailed Design
Interface Agent
Personal Agent
Send and Receive Mail Agent
Encryption and Decryption File Agent
File Transferring Schedule Agent
Entities
System entities
Filter
email notification
Generate interface agent
Generate send/receive mail agent
inform user
Encryption and Decryption File Agent
File Transferring Schedule Agent
Interface Agent
Personal Agent
Send and Receive Mail Agent
create E/D file agent capability
create FTS agent capability
Encryption and Decryption File Agent - Agent Overview Diagram
Help
Jop requirements from user
create E/D file agent capability
request E/D file
Data Sources
Knowledge Repositories
file encrypted
recieve info from E/D file agent
file decrypted
inform user
create send/receive mail agent capability - Descriptor
Name:
create send/receive mail agent capability
Description:
Goals:
Edit
Processes:

Figure 12. Encryption and Decryption File Agent

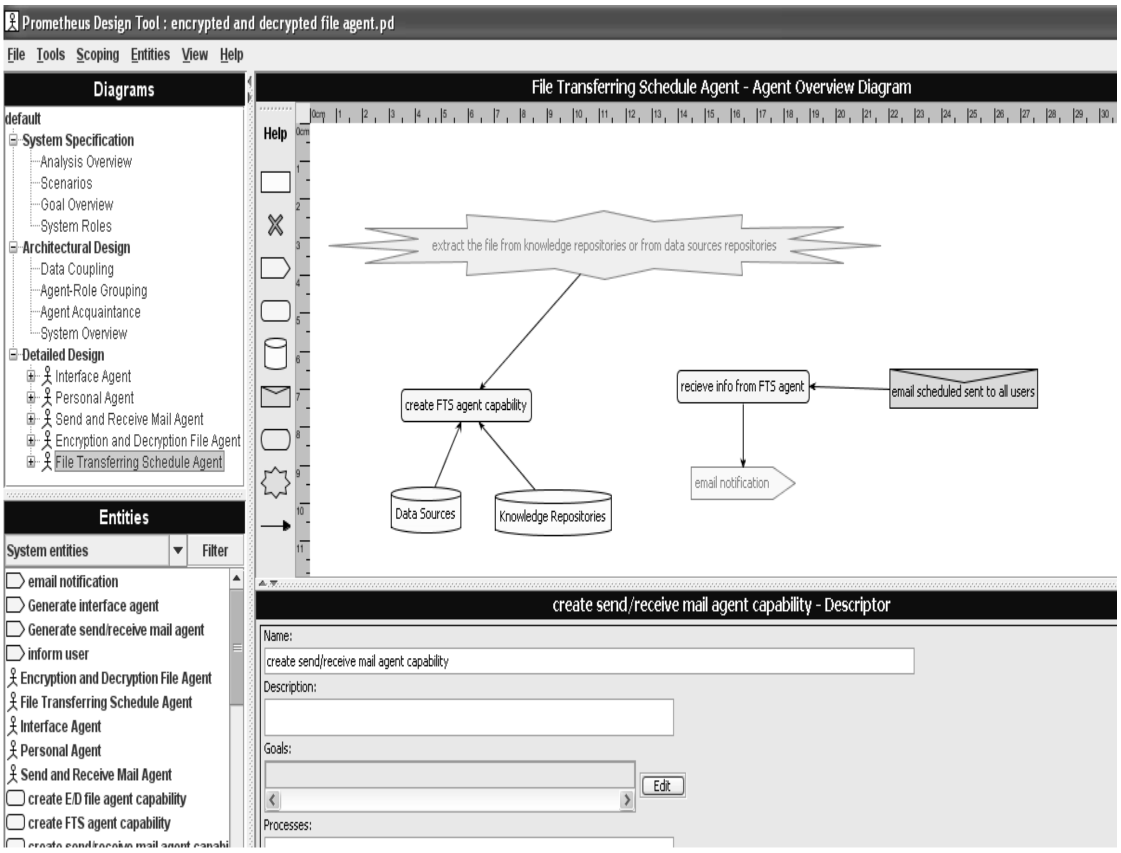

Prometheus Design Tool : encrypted and decrypted file agent.pd
File Tools Scoping Entities View Help
Diagrams
default
System Specification
Analysis Overview
Scenarios
Goal Overview
System Roles
Architectural Design
Data Coupling
Agent-Role Grouping
Agent Acquaintance
System Overview
Detailed Design
Interface Agent
Personal Agent
Send and Receive Mail Agent
Encryption and Decryption File Agent
File Transferring Schedule Agent
Entities
System entities
Filter
email notification
Generate interface agent
Generate send/receive mail agent
inform user
Encryption and Decryption File Agent
File Transferring Schedule Agent
Interface Agent
Personal Agent
Send and Receive Mail Agent
create E/D file agent capability
create FTS agent capability
File Transferring Schedule Agent - Agent Overview Diagram
Help
extract the file from knowledge repositories or from data sources repositories
create FTS agent capability
Data Sources
Knowledge Repositories
recieve info from FTS agent
email scheduled sent to all users
email notification
create send/receive mail agent capability - Descriptor
Name:
create send/receive mail agent capability
Description:
Goals:
Edit
Processes:

Figure 13. File Transferring Schedule Agent

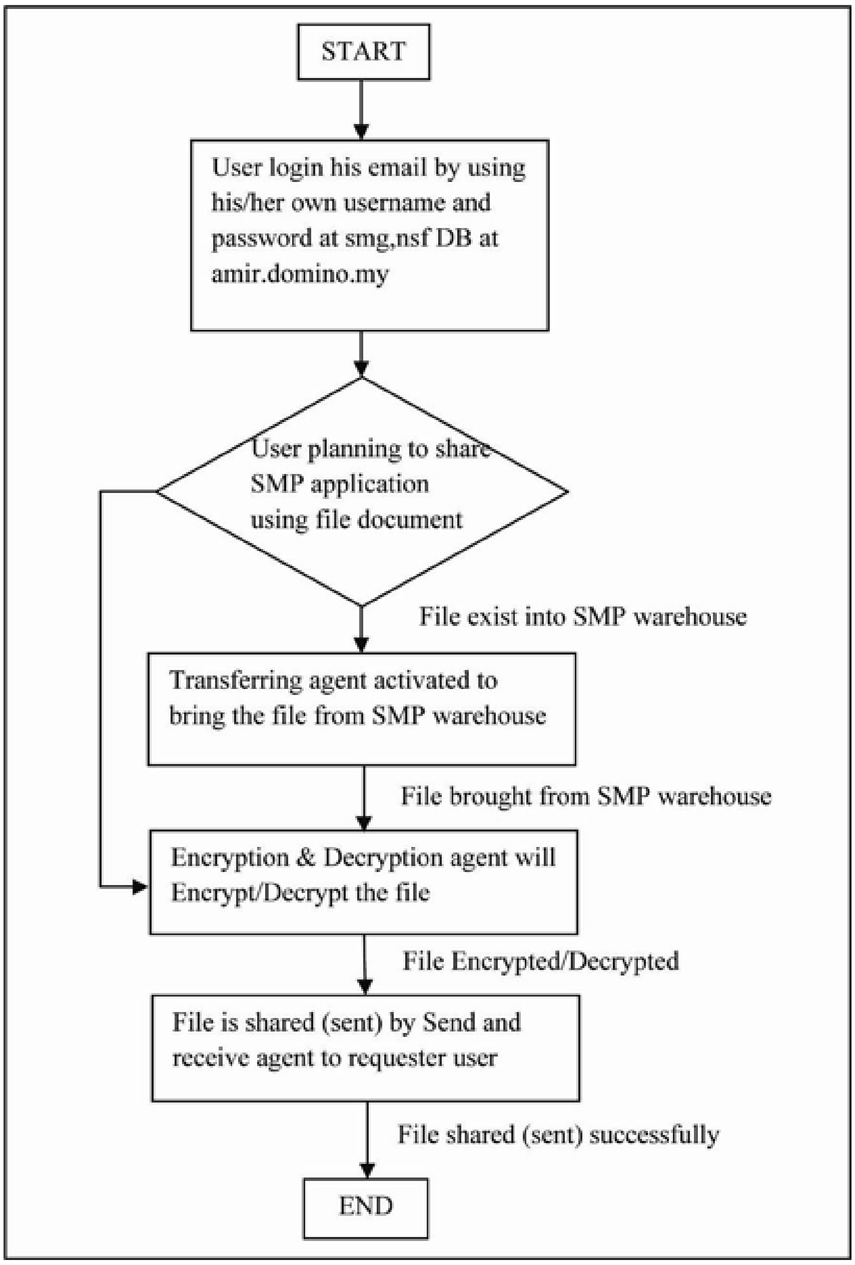

START
User login his email by using his/her own username and password at smg,nsf DB at amir.domino.my
User planning to share SMP application using file document
File exist into SMP warehouse
Transferring agent activated to bring the file from SMP warehouse
File brought from SMP warehouse
Encryption & Decryption agent will Encrypt/Decrypt the file
File Encrypted/Decrypted
File is shared (sent) by Send and receive agent to requester user
File shared (sent) successfully
END

Figure 14. System Flowchart

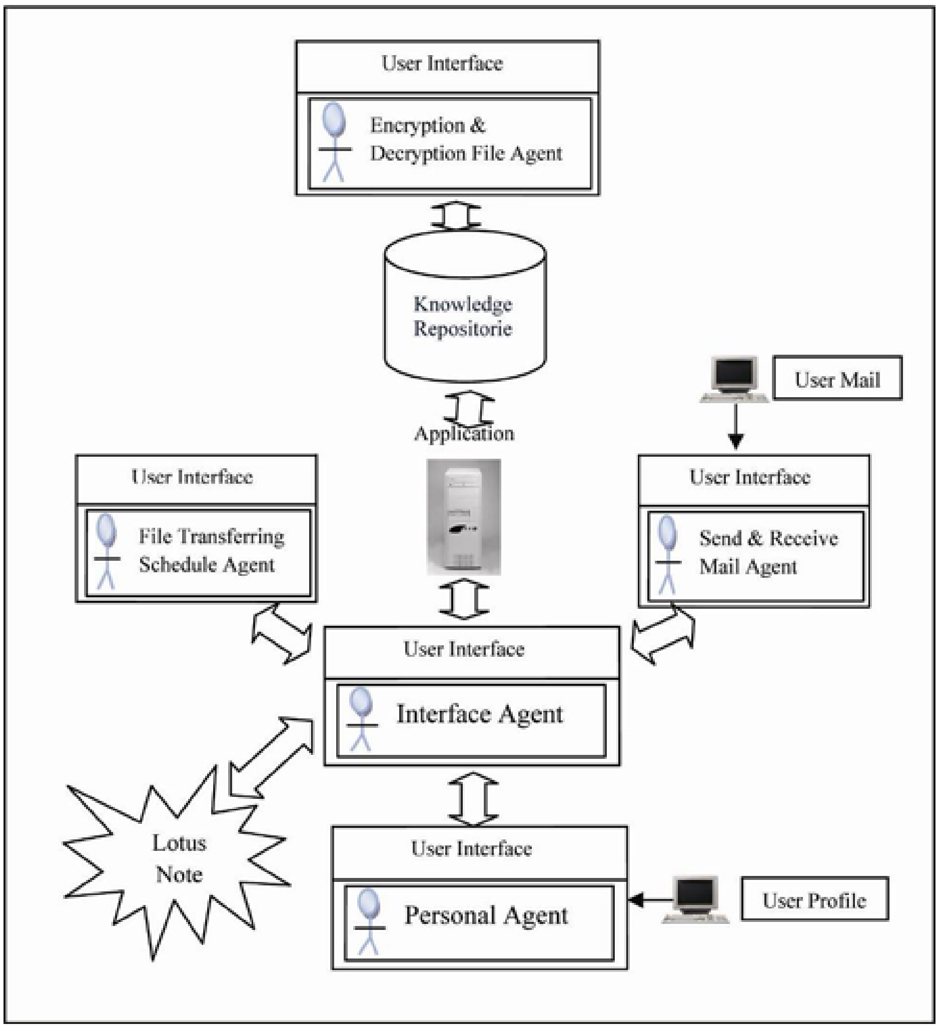
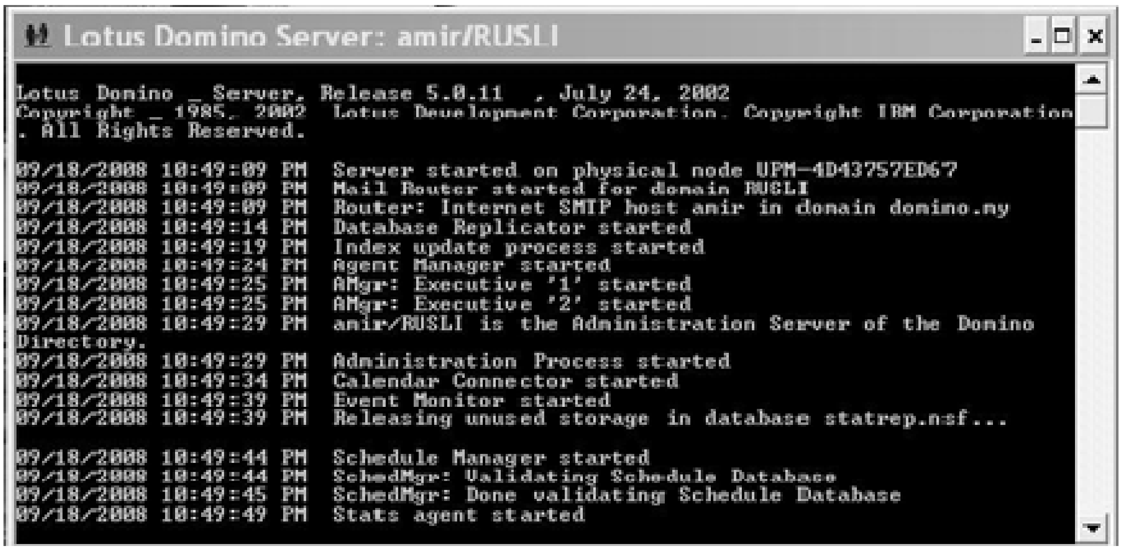


Figure 15. the rest of the agents in MASK-SM Framework

Note 2

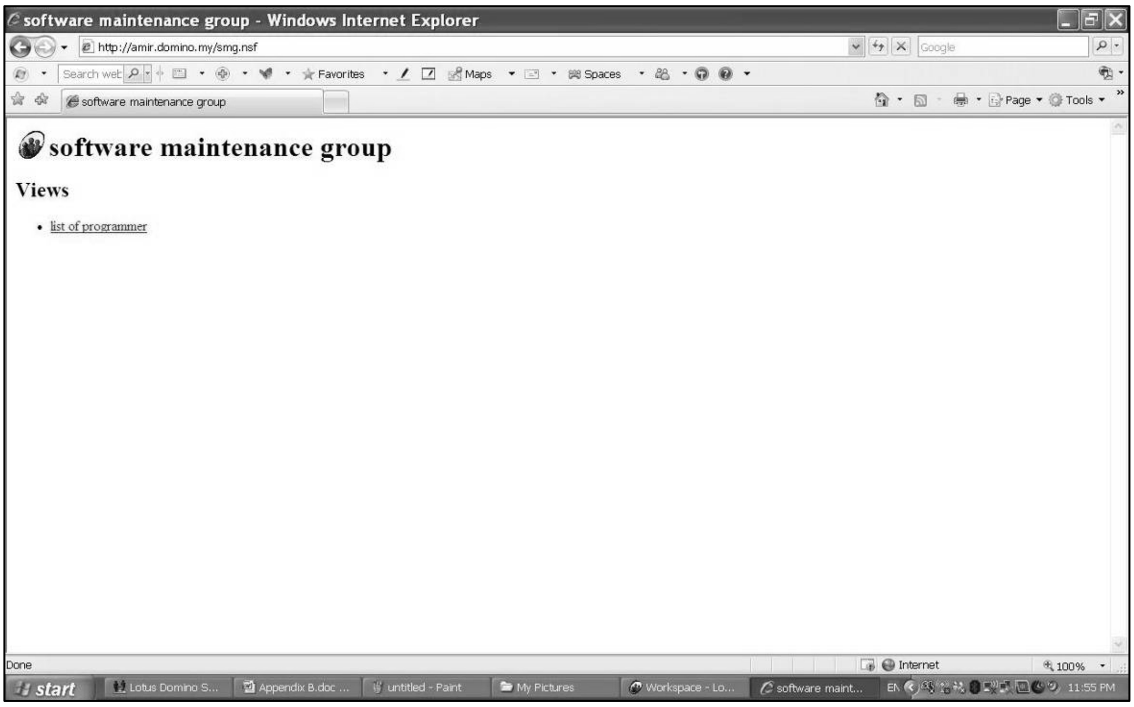


```
Lotus Domino _ Server, Release 5.0.11  , July 24, 2002
Copyright _ 1985, 2002  Lotus Development Corporation. Copyright IBM Corporation
. All Rights Reserved.

09/18/2008 10:49:09 PM  Server started on physical node UPM-4D43757ED67
09/18/2008 10:49:09 PM  Mail Router started for domain RUSLI
09/18/2008 10:49:09 PM  Router: Internet SMTP host amir in domain domino.my
09/18/2008 10:49:14 PM  Database Replicator started
09/18/2008 10:49:19 PM  Index update process started
09/18/2008 10:49:24 PM  Agent Manager started
09/18/2008 10:49:25 PM  AMgr: Executive '1' started
09/18/2008 10:49:25 PM  AMgr: Executive '2' started
09/18/2008 10:49:29 PM  amir/RUSLI is the Administration Server of the Domino
Directory.
09/18/2008 10:49:29 PM  Administration Process started
09/18/2008 10:49:34 PM  Calendar Connector started
09/18/2008 10:49:39 PM  Event Monitor started
09/18/2008 10:49:39 PM  Releasing unused storage in database statrep.nsf...

09/18/2008 10:49:44 PM  Schedule Manager started
09/18/2008 10:49:44 PM  SchedMgr: Validating Schedule Database
09/18/2008 10:49:45 PM  SchedMgr: Done validating Schedule Database
09/18/2008 10:49:49 PM  Stats agent started
```

Figure 16 (a). Lotus Domino Server amir/RUSLI

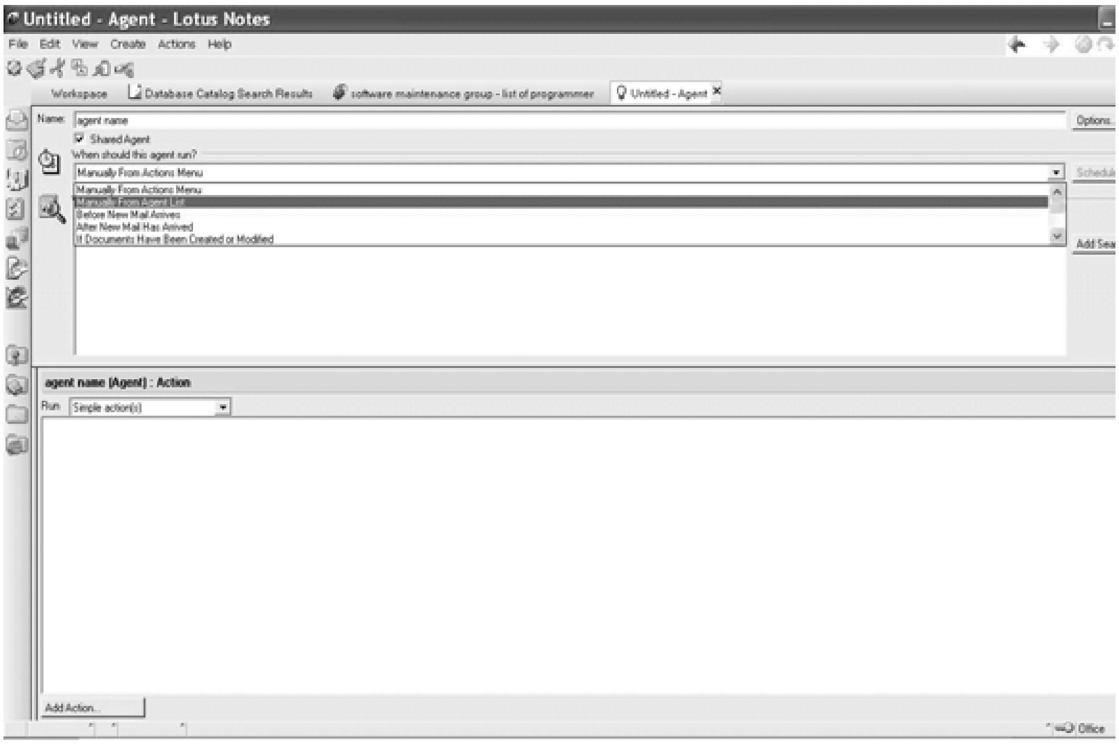

software maintenance group - Windows Internet Explorer
http://amir.domino.my/smg.nsf
software maintenance group
Done
Internet
100%
start
11:55 PM

software maintenance group

Views

- list of programmer

Figure 16 (b). Software Maintenance Group


Untitled - Agent - Lotus Notes
File Edit View Create Actions Help
Workspace
Database Catalog Search Results
software maintenance group - list of programmer
Untitled - Agent
Options...
Schedule...
Add Search...
Add Action...
Office

Name: agent name

☑ Shared Agent

When should this agent run?

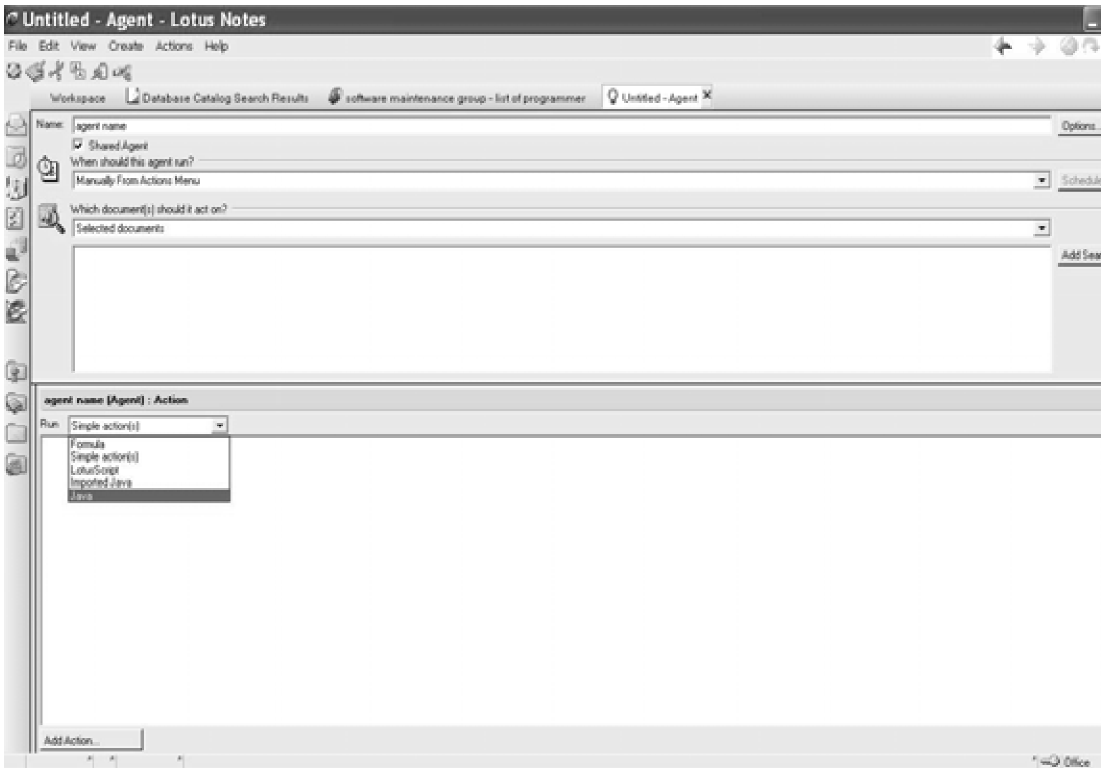
Manually From Actions Menu

- Manually From Actions Menu
- Manually From Agent List
- Before New Mail Arrives
- After New Mail Has Arrived
- If Documents Have Been Created or Modified

agent name (Agent) : Action

Run: Simple action(s)

Figure 16 (c). Selecting trigger to run the agent


Untitled - Agent - Lotus Notes
File Edit View Create Actions Help
Workspace
Database Catalog Search Results
software maintenance group - list of programmer
Untitled - Agent
Options...
Schedule
Add Search
Add Action...
Office

Name: agent name

☑ Shared Agent

When should this agent run?

Manually From Actions Menu

Which document(s) should it act on?

Selected documents

agent name (Agent) : Action

Run: Simple action(s)

Formula
Simple action(s)
LotusScript
Imported Java
Java

Figure 16 (d). Selecting java to write the agent code

Note 3

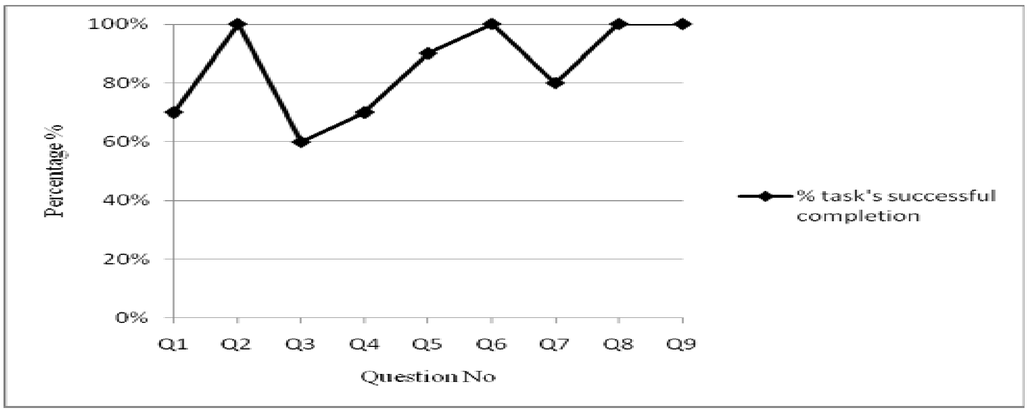


Figure 17. Completed Tasks Successfully for Quantitative analysis for MAS

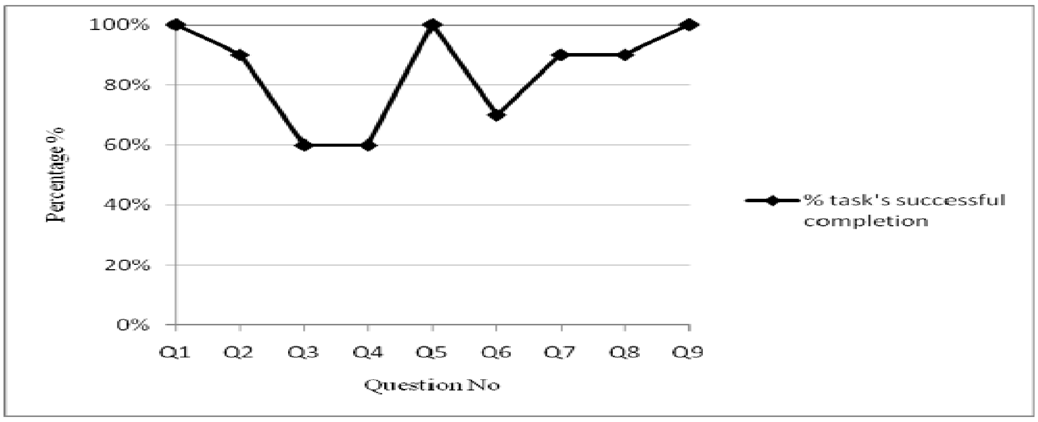


Figure 18. Completed Tasks Successfully for Qualitative analysis for MAS

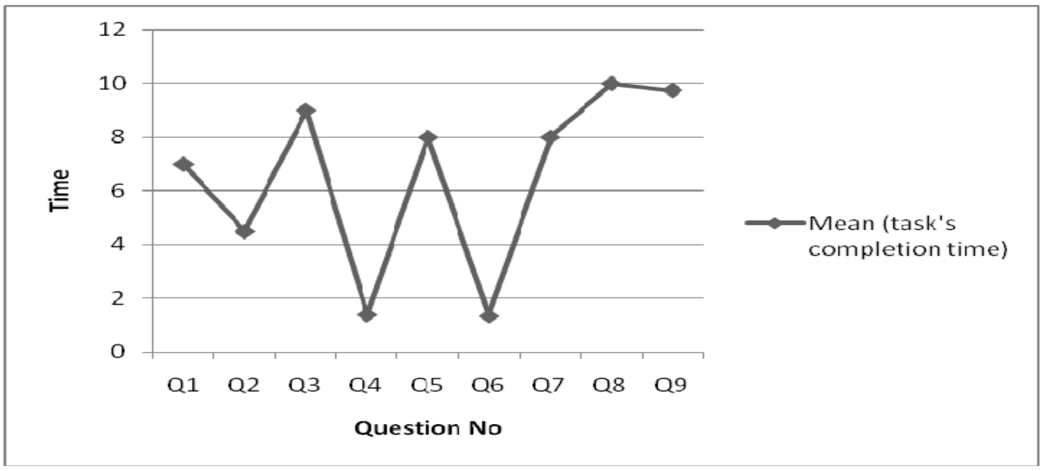


Figure 19. Time used to completed for Quantitative analysis for MAS

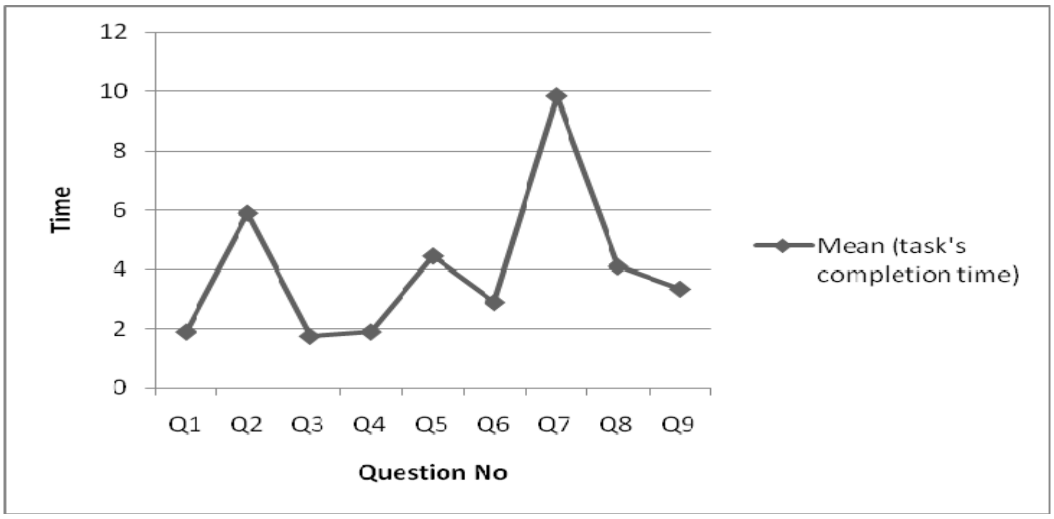


Figure 20. Time used to completed for Qualitative analysis for MAS

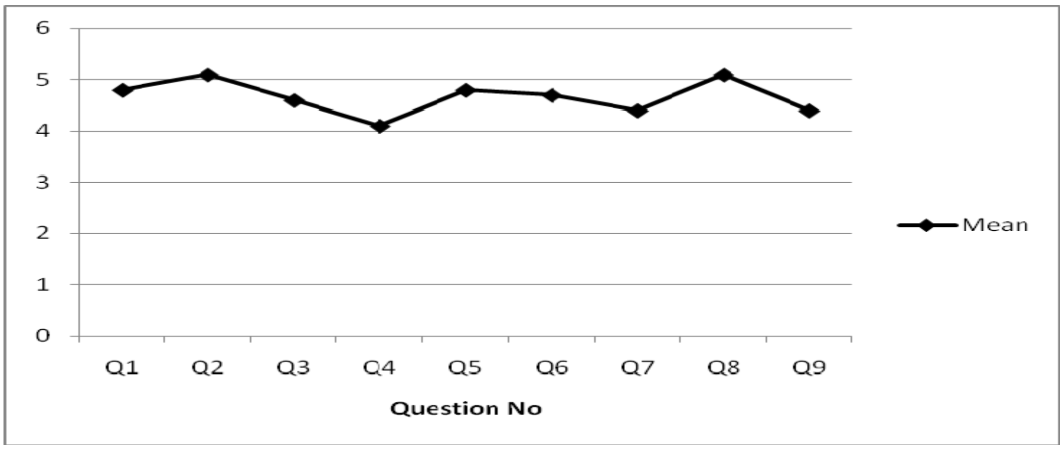


Figure 21. Respondent satisfaction for Quantitative analysis for MAS

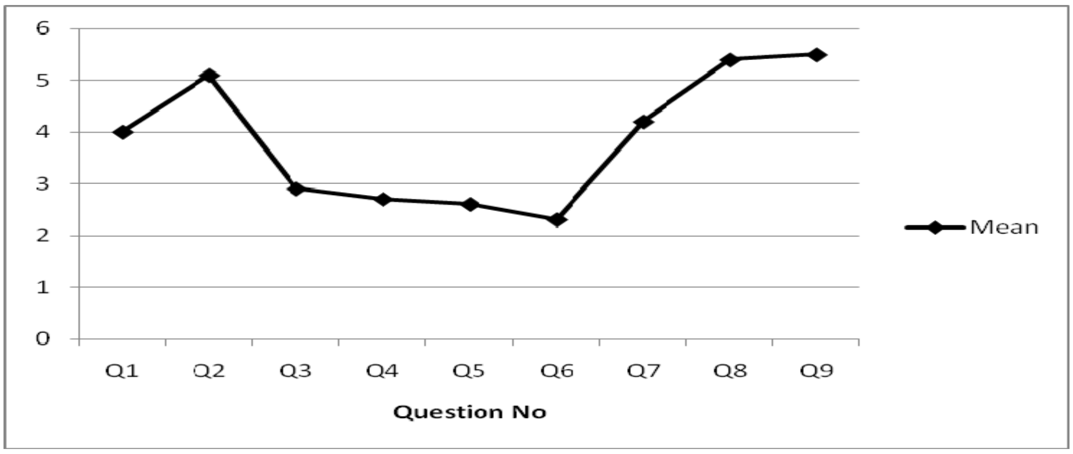


Figure 22. Respondent satisfaction for Qualitative analysis for MAS

Note 4

APPENDIX A

Respondent demographics

Ten individuals participated in this study. After collecting the background questionnaire we calculate the demographic characteristic of the respondents.

Table A.1: Sample representative of the respondents

		Sample N=10	**Percentage %**
Gender	Male	8	80%
	Female	2	20%
Education	Degree (Bachelor, Diploma)	5	50%
	Postgraduate (Master, PhD)	5	50%
Age group	20 to 29	2	20%
	30 to 39	5	50%
	40 to 49	3	30%
	50 to 59	1	10%
System Usage	Daily	4	40%
	From time to time	6	60%
System Shown	First time	6	60%
	Familiar	4	40%

Table A.1 summarized the basic characteristics of the initial sample as well as those of the respondent. As can be noted from the table above, the sample is rather skewed towards males in the age group between 23 and 40 years old and of higher education. Most participants were experienced and familiar with lotus notes software: forty percent (40%) used the system on a daily basis and were familiar with the lotus notes system and its applications; moreover sixty percent (60%) have shown and described the system for them.

APPENDEX B

QUESTIONNAIRE SHEET

Appendix B.1 Pre-Survey Questionnaire

Thank you very much for agreeing to participate in this experiment. All of your personal data that we collect will be entirely confidential. I would like to gather a bit of background information about u.

Participant Name____

Gender: ______Male ______Female

Date_______________________

How old are you? 20-29 30-39 40-49 50-59 60 or above

Level of education:

_____Certification Bachelor ________ Certification Diploma

__________Degree Postgraduate

Race: __________Malaysian (Local) __________International

Years of Experience _______________________

Appendix B.2 Usability Testing Questions

The goal of this Survey to evaluate the KMS by using Usability testing questions and prove the KMS is a useful support system.

I will ask you a series of questions and would like you to think out loud while you look for the answer. Please remember that we are testing the effectiveness of the KM and this is not a test of you. The whole test should take less than one hour. Thank you

Description for How to Answer the Question:

Evaluation of the matrix: Assign yourself the following points for each

NA = 0, where 0 is doing nothing at all = NONE and

1 = Don't Know, Not Sure or Can't Say = NO

2 = Not Important or as Not been Addressed = MINIMALLY

3 = Partially Beneficial or somewhat Effective or Less Scope for Overall Improvement =

PARTIALLY

4 = Important or May not be effective but other associated necessary actions being taken =SUBSTANTIALLY

5 = Critical or already in place and effective = FULLY

Also, the scale can generally be summarized as follows for majority situations

'NA 1 2 3 4 5' is calibrated as in

'5 (Always) 4 (Often) 3 (Sometimes) 2 (Occasionally) 1 (Never)'

NA (Not Applicable), (Note: "NA" and "1" scale values are equivalent.)

QUESTIONNAIRE - Part One (Quantitative Analysis)

1. Is recording and sharing knowledge a routine and like any other daily habits for the employees?

NA 1 2 3 4 5

2. Are the employees co-operative and helpful when asked for some information or advice?

NA 1 2 3 4 5

3. Is Knowledge sharing seen as strength and knowledge hoarding as a weakness?

NA 1 2 3 4 5

4. Is good knowledge management behavior like sharing, reusing knowledge actively promoted on a day-to-day basis?

NA 1 2 3 4 5

5. Are people in the organization aware of the need to proactively manage knowledge assets?

NA 1 2 3 4 5

6. Do people at all levels in the organization participate in some kind of a community or communities of practice?

NA 1 2 3 4 5

7. Is there top management representation for KM?

NA 1 2 3 4 5

8. Is knowledge management a formal function area in the organization?

NA 1 2 3 4 5

9. Are the teams in the organization effective? Are self managed teams composed of individuals capable of learning from each other?

NA 1 2 3 4 5

QUESTIONNAIRE- Part Two (Qualitative Analysis)

1. Do the employees share their knowledge?

Yes No

2. Is the intranet used to share knowledge in an informal manner (non-routine, personal and unstructured way)?

Yes No

3. Do workplace settings and format of meetings encourage informal knowledge exchange?

Yes No

4. Are there incentives given for knowledge contribution, exchange or on knowledge sharing in your firm?

Yes No

5. Is the support from executive management to KM (Knowledge Management)\ knowledge sharing VISIBLE?

Yes No

6. Are there specific knowledge roles identified and assigned?

Yes No

7. Are all senior managers and professionals trained in knowledge management techniques?

Yes No

8. Is knowledge validated through peer or superior review or, is there some kinds of librarians or information management staff that coordinate knowledge repositories.

Yes No

9. Is knowledge sharing across departmental boundaries actively encouraged? (Not similar to ‘’incentives’’)

Yes No

Appendix B.3 Post-Survey Questionnaire

Thanks again for participating in this experiment. This questionnaire gives you an opportunity to tell us your reactions to the system you used. Please circle a number on the scale to indicate your reactions. Thank you

The goal of this part to evaluate the MAS that applying into the Lotus Notes Domino and to prove the MAS will help the users according to their needs.

QUESTIONNAIRE - Part One (Quantitative Analysis)

1. Is it possible to change the send and receive agent schedule.

NA 1 2 3 4 5

2. We can run the send and receive agent "After new mail arrives" and "Before new mail arrives".

NA 1 2 3 4 5

3. Send and receive agent option will appear in the current mail file.

NA 1 2 3 4 5

4. One of our users left the office without enabling the send and receive agent. We can enable it for him or her.

NA 1 2 3 4 5

5. I sent to someone multiple e-mails while that person is out of the office. So I will receive only one e-mail notification.

NA 1 2 3 4 5

6. To customize the "Welcome Back" message, the "Disable Reminder" message, or the default wording of the e-mail notifications sent to all senders of e-mail.

NA 1 2 3 4 5

7. In order to notice the Domino Designer 5 client has new agent properties, such as "Allow user activation" and "Run on behalf of." The both of these we need to set in the mail template (on the server) or in the individuals' mail files for the send and receive agent to work properly.

NA 1 2 3 4 5

8. The send and receive agent work in a clustered environment.

NA 1 2 3 4 5

9. We can enable the scheduler agent for leaving "Today" instead of the recommended "Tomorrow" or another date in the future.

NA 1 2 3 4 5

QUESTIONNAIRE- Part Two (Qualitative Analysis)

1. We can set the scheduler agent for an absence period of a half day or a few hours.

Yes No

2. Whenever we receive a warning in Designer while attempting to save an agent
"You do not have execution access privileges for this agent on server ". This indicates one of two things: either the agent signer does not have the rights on the scheduled server, or that server is not reachable to check the signer rights. Running agent "test" in the Designer will give you a better indication.

Yes No

3. "Do you know why I get 'Object variable not set'?" This is a result of a logic error in the code. The problem should become clear if you single step through the code in debugger (File - Tools –Lotus Script debugging). Server might be configured to delay execution of your agents.

Yes No

4. If these tips don't help you figure it out on your own, when you post in the forum please include in your post screen shot of server log output with agent manager debug flags set to '*' (best) and/or diagnostic output of "agent test" (a good second choice when you don't have access to the server log).

Yes No

5. It is possible to pass parameters between agents.

Yes No

6. It is easy to sign an agent with a server.id For Lotus Notes 5.

Yes No

7. It is easy to console commands from send and receive and scheduler agent.

Yes No

8. Agents runs but mail is not being sent. If our agent runs to completion (i.e. no run time errors that stop the agent before it gets to the send logic) this symptom usually means that it is configuration issue, not an agent problem.

Yes No

9.Does the agents that applied will help the users of the system?

Yes No

Comment about the system:____.