Constructing a Collaborative Multi-Agents System Tool for Real Time System Requirements

Asmaa Y. Hamo[1], Marwa A. Aljawaherry[2]

[1, 2] Software Engineering, Mosul University, Collage of Computer sc. And Mathematic ZIP/+964, Iraq

Abstract

Conventional software tools that support Requirements Engineering (RE) activities are typically used by stakeholders that work collaboratively in the same location or same office. At last years stakeholders are distributed around different locations and time-zones, and they still need to collaborate and communicate in an effective and efficient way to share, create and discuss knowledge. Nowadays. A challenge is to provide integrated collaborative tools that implement creatively techniques which allow distributed stakeholders to externalize, share and store their Knowledge in common repository. The vision of this proposed research is to design, implement and evaluate the Collaborative Requirements (Col_Req) tool that support collaborative acquisition, navigation and documentation activities, This tool is based on Multi-agent technologies that support collaborative environments. The tool is evaluated according to general criteria for collaborative tools and it satisfied 6 out of 7 criteria also the efficiency of mobile agent for transferring functional and non- functional requirements is measured when the net is free and busy and it show good result.

Keywords: *Collaborative Software Engineering, Multi Agent Systems, Requirements Engineering.*

1. Introduction

Nowadays, with rapid popularization of the WWW, Software collaborative systems can bring some dispersed teams together, automatically supporting communication and cooperation. Both teams and everyone are able to benefit from the collaborative process [19].Software engineering projects are inherently cooperative, requiring many software engineers to coordinate their efforts to produce a software system. [16]. Collaboration can be defined as an activity that involves two or more companies, departments or customers who combine their competencies and technologies to create new shared value while, at the same time, managing their respective costs and risks [7]. Software Engineering (SE) is undoubtedly a collaborative process. Developers within a SE project work together during all phases of the software development lifecycle [4]. Virtually all software development requires collaboration among developers within and outside their project teams, to achieve a common objective [10].Currently agents and Multi-agent technologies have been widely used in many different areas. In collaboration systems, effective collaborative working between the members is very important. Therefore, the systems use intelligent agents to meet its needs. [20] Because an agent can represent a real user, an agent-based approach is useful in a distributed software engineering environment in which users may not be able to meet face-to-face, or even be working at the same time [3]. A software agent "is an autonomous entity that performs one or several tasks in order to achieve some goals" [9], an agent can be either static or mobile [3].

2. Collaboration Modes:

There are essentially four modes of collaboration depending on Pattern of communication between the users in a project. The classification in the space–time coordinates shown in Figure (1), there are four quadrants in the system of coordinates, and every quadrant presents one collaboration mode. The types of Collaboration in the coordinates are briefly summarized below [19].

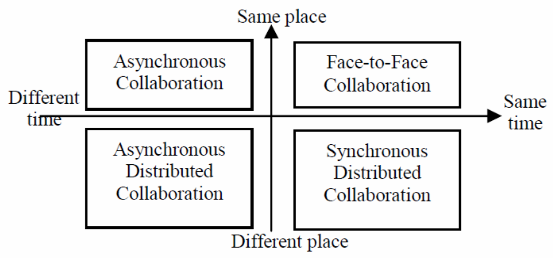

Same place
Asynchronous Collaboration
Face-to-Face Collaboration
Different time
Same time
Asynchronous Distributed Collaboration
Synchronous Distributed Collaboration
Different place

Figure (1) Collaboration modes [19] [20]

- Face-to-face Collaboration: this is normally collaboration mode. The members of the team engage in face-to face discussions in the same time and the same place
- Asynchronous Collaboration: this mode of collaboration always uses blackboards to inform some notices about software project in the development place. It always offers some information to the team.
- Synchronous Distributed Collaboration: this mode mainly involves real-time communication in different places. There are some existent tools such as video conferencing, and etc. to support real-time group discussion. This is one important collaboration mode in dispersed teams.
  We adopt this mode of collaboration for designing our tool.
- Asynchronous Distributed Collaboration: this mode of communication is widely used in the collaboration environment. It mainly involves communicate in different time and in different place. It always uses some tools (such as E-Mail) to share and exchange information between the members of the team [19] [20].

3. Aims of Multi-Agent Based Collaborative Environment

Today, in collaborative environment agent technology has been widely used in the development of software project. The intelligent agents are able to tackle specialist problem and can interact with each other within a distributed environment. So the use of agent technologies in software collaborative environment is necessary and effective [19].
The general aims of the multi-agent based collaborative environment are: (1)To facilitate the teams and members to communicate with each other,(2)To support distributed collaborative design and development, (3) To assist in maintaining design and development constraints,(4) To record development process and histories, through the tracing of detailed Design processes and related information, (5) To support the reuse of development and design.
Agents operating in the collaborative environment fulfill differing roles, either as part of the problem solving process, or as part of the communications infrastructure [20].

4. JADE (Java Agent DEvelopment framework) platform

JADE simplifies the implementation of multi-agent systems. Multi-agent platforms designed by JADE can be distributed across machines, and using different operating systems [17]. JADE intends to provide software packages and tools to allow java developers to deploy FIPA (Foundation for Intelligent Physical Agents) agent systems.
A JADE platform is composed of agent containers that can be distributed over the network. Agents in JADE are implemented as java threads, JADE agents live in a container, which is the java process that launches them. At runtime, a JADE system will contain multiple containers [5], each container living in a separate Java Virtual Machine and delivering run-time environment support to some JADE agents. Java RMI (Remote Method Invocation) is used to communicate among the containers [2], a single special Main container must always be active in a platform and all other containers register with it as soon as they start [8].
Three tables are also maintained in the main container:

1- The container table (CT).
2- The global agent descriptor table (GADT).
3-The Message Transport Protocol (MTP) and
Internal Message Transport Protocol (IMPT)

Figure (2) shows the main architectural elements of a JADE platform. Any multi-agent system developed by JADE will have the following agents:

1- DF (Directory Facilitator) agent.
2- AMS (Agent Management System) gent.
3- RMA (Remote Monitoring Agent) agent [17].

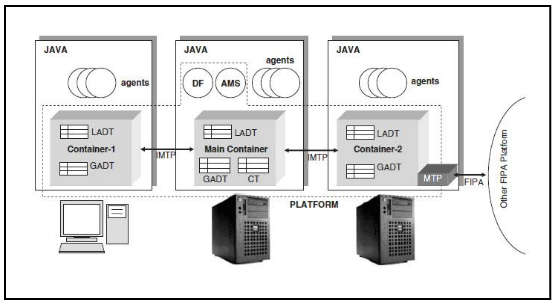


Figure (2) main architectural elements of JADE platform [5].

The most important feature provided by JADE is the communication ability between the Agents, which use the asynchronous communication transmission [18].

5. Requirements of Real Time Systems

Real time analysis and development is an intricate process. This is because these systems exhibit special behaviour. Real time timing, communication and reliability requirements are not easily explained and represented [12], While there are a number of useful taxonomies of requirements, the most established being

the simple functional versus non-functional classification; the standard scheme for real-time systems is defined by IEEE Standard [IEEE830]. Standard 830 [IEEE98] defines the following kinds of requirements [13].
Functional requirements: The functional requirements of real-time systems are concerned with the functions that a real-time computer system must perform [13].
Non-functional requirements (NFRs): is a broadly used term, however, there is no consensus about their nature, since various classifications exist. NFRs play a significant role during system design, since they depict the conditions under which specific system components should operate, leading to alternative design decisions [15].

6. Distributed Requirements Engineering

Software Systems requirements engineering (RE) is the process of identifying the concerns of the stakeholders and documenting them in such a way that it is amiable to analysis, communication, and subsequent implementation. The main goal of requirement engineering is to meet the degree of end user's satisfaction in minimum cost and time [1].
In Global Software Development (GSD), distributed stakeholders (e.g. team members, customers, etc) have to collaborate and communicate in an efficient and effective way to share, create and discuss knowledge [11]. End user's requirements are gathered by requirement analysts who are geographically dispersed. Collected requirements are integrated later for a single software development [1].

7. The Goals of (Col-Req) Tool

The aim of the designed tool is to create the requirements document of real time systems that classified to functional and non–functional requirements. Gathering requirements phase is the most important phase in software engineering process, requirements analysis was done by navigation and communication and interaction between requirements engineers and customer to find a common format for requirements of the developed project where the work will be divided over the team members, the purpose of that is to do the work as fast as possible.
The collaborative environment for (Col_Req) tool achieves the efficiency in the work through sharing ideas and exchange the skill and knowledge. The tool has several faculties for software requirements engineering for real time systems these faculties include:

- Simplifying the communication process between the distributed requirements engineers and the customer.
- (Col_Req) tool offers the ability for distributed team members to be aware of the activates of each others.
- Deploying the updates of requirements in older to keep the consistency of the distributed requirements.
- Document the requirements and display them as tables.
- Distributed the tasks between the requirements engineers.
- Avoid the conflicts in writing the distributed requirements.
- The requirements engineers work in parallel manner to decrease the development time.
- (Col_Req) tool offers simple web interfaces for its users.
- (Col_Req) tool offers public conversation channel, this conversation displays all shared messages that exchange between the requirements engineers and the customer.
- (Col_Req) tool support exchange messages between the team members when they work in the same time.

While our tool doesn't have the following properties:

- (Col_Req) tool doesn't support generation of graphical documents but only text documents.
- (Col_Req) tool care with the requirements developments phase, but it doesn't cover all requirements management stage that include tracking requirements process and ensure from it is validity that is continue until the development process of project is finished.

8. The Architecture of Collaborative Requirements (Col_Req) Tool

(Col_Req) tool was designed to support software requirements engineers for real time systems during the requirements engineering (RE) phase, it helps them to develop and document requirements. This tool is used by small teams that consist of (3-7) software requirements engineers and a customer each of them has it is specific role; we can explain them as follows:
Customer: he represents the owner of the project that can be a person or a company. He works in one computer where all team members are always navigate and collaborate with the customer to make him know about newest version of requirements and to make sure from his needs and desires.
General requirements engineer: He interacts and collaborates with the other engineers and the customer to develop the requirements and exchange knowledge to find a common format for functional and non-functional requirements, there are five general requirements engineers use the tool.

Functional requirements engineer: he is responsible of updating the functional requirements and he only can modify them such as append, delete requirements.
Non-Functional requirements engineer: he is responsible of updating the non-functional requirements and he only can modify them such as append, delete, modify the requirements.
Each requirements engineer has its static personal agent that represent him see figure (3) , Also there are two other mobile agents that responsible of transferring the

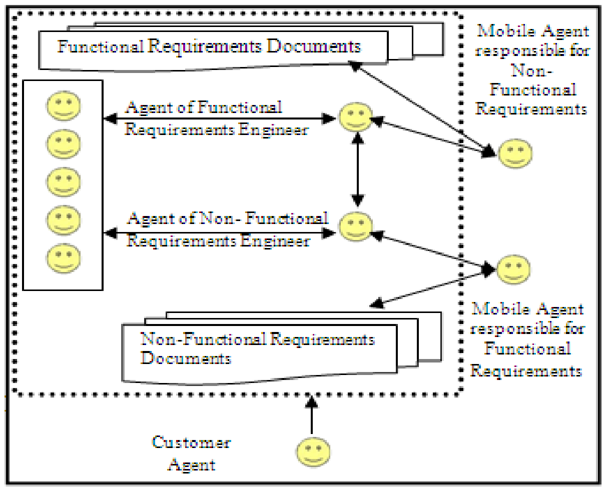

Functional Requirements Documents
Mobile Agent responsible for Non-Functional Requirements
Agent of Functional Requirements Engineer
Agent of Non- Functional Requirements Engineer
Mobile Agent responsible for Functional Requirements
Non-Functional Requirements Documents
Customer Agent

Figure (3) Collaborative Environment for (Col_Req Tool)

functional and Non- functional requirements to other engineers and customer in continuous manner to distribute the updates and keep the consistency of the functional and Non-functional requirements documents at all distributed sites.

9. The Usage of (Col_Req) Tool

(Col_Req) tool used by small teams of requirements engineers that consist of (4-8) members, where each member works at personal computer and they are distributed over the world. All members communicate using this tool and they can share a public conversion through exchanging messages between them and display this conversion at shared white board. The functional and non-functional requirements are displayed on two tables. The first table display functional requirements and several paragraphs like write an introduction paragraph for the project, but the second table display non-functional requirements. These two tables are organized according to IEEE standard for real time systems. When the team members start their work, all exist members will exchange messages to navigate and give their opinion for each elements of requirements to find a common format for these requirements and then the functional and non-functional requirements engineers will document them at tables.

10. Several Programming Problems and Their Solutions

(Col_Req) tool is designed using JADE version (JADE 4.1) available yet is used. Several programming problems are faced they are:

- Sending messages :
  Every team member have it is own static agent where there are 8 static agents in the system every agent represent, It is member and he is responsible for communication between agents through sending and receiving ACL messages. All messages will be sent to all other agents and then they will be displayed on white shared board that exist on each site where each agent is resident at single computer.
- Conversions between agents :
  These kinds of conversion are called complex conversion because each agent will send and receive the messages from all other agent.
- Behaviours:
  We used one shot Behaviour type for sending the messages and cyclic Behaviour for receiving all other agents messages.
- Mobile agents:
  Two mobile agents in our tool are created. They are used to transfer the files of requirements for the functional and non–functional requirements engineers to other distributed sites and these mobile agents will deploy the updates in continuous manner to keep the consistency of the requirement at each site. The mobile agent will move using the doMove() method. It will move to all distributed sits and write the requirements file at each site and then it will be back to it is home location.
- The migration of mobile agent :
  The migration of mobile agent between remote platforms is implemented. Each static agent and mobile agent was created at different platform, therefore the Inter-Platform Mobility Services (IPMS) add-on is used to perform this kind of migration.
- The itineraries of mobile agent:
  Each mobile agent starts it's itineraries from its home location and it visits all other sites (computers) holding the requirements file and then it write this file at each visited site and finally it will return to its home location.
- Migration protocols:

The default migration protocol in JADE is Push Cache Transfer Protocol (PCTP) but this protocol does not support the migration of large size of mobile agents for that reason we used Fragment Transfer Protocol (FTP) that support transferring mobile agent that have big size .

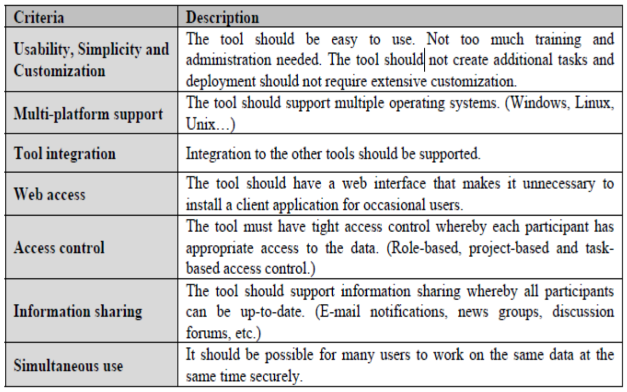
- Processing the increases size of mobile agent :
  there is always probability to increase the mobile agent size because the size of requirements file always increases therefore we face a problem with the heap space and we solve it by increasing that space by using the following commend line:
  -Xmaxs 512- Xminx 1024.
- Processing of mobile agent messages Queue:
  The probability of increasing the size of mobile agent cause an over loading in messages queue of mobile agent, therefore we reset the size of the message queue of the mobile agent and make it larger to be able to include the big size of mobile agent.
- Processing the migration time:
  Because the large size of the mobile agent, the default migration time in JADE was not enough to perform the migration process successfully for that reason we reset migration time in JADE and increase it.
- Read / Write The Requirements Files in a Fast Manner:
  We need to read/write requirements files in a fast manner because their big sizes, for that these files read/write as lines not bytes.
- Supporting the security of (Col_Req) Tool:
  In the login interface we use the name of the requirements engineer that he login to use the tool as the agent name that will be created to be the personal agent of the requirements engineer to ensure that there is no other user (engineer) having the same name because JADE prevents creating two agent having the same name at the same platform.

10. The Results

We have two kinds of result in our research.

- Measuring the collaborative of (col-Req) tool according to general criteria
  There are several general criteria used to evaluate the performance of collaborative tools. The common criteria for all these tool classes are presented in Table (1).These criteria are important for tools that are meant to be used in collaborative environment [6]. We evaluate (col-Req) tool according these criteria see table (2), so its satisfy 6 out of 7 criteria and that is a good result.

Table (1) General criteria for collaborative tools [6].



Criteria	Description
Usability, Simplicity and Customization	The tool should be easy to use. Not too much training and administration needed. The tool should not create additional tasks and deployment should not require extensive customization.
Multi-platform support	The tool should support multiple operating systems. (Windows, Linux, Unix…)
Tool integration	Integration to the other tools should be supported.
Web access	The tool should have a web interface that makes it unnecessary to install a client application for occasional users.
Access control	The tool must have tight access control whereby each participant has appropriate access to the data. (Role-based, project-based and task-based access control.)
Information sharing	The tool should support information sharing whereby all participants can be up-to-date. (E-mail notifications, news groups, discussion forums, etc.)
Simultaneous use	It should be possible for many users to work on the same data at the same time securely.

Table (2) evaluate (Col_Req tool) with General criteria for collaborative tools

Criteria No.	Criteria	Result
1	Usability, Simplicity and Customization	✓
2	Multi-platform support	✓
3	Tool integration	✗
4	Web access	✓
5	Access control	✓
6	Information sharing	✓
7	Simultaneous use	✓

- Measuring the efficiency of mobile agent in transferring requirements files:
  In our tool mobile agent have most important role because it transfers the requirements files, these files can have different sizes, we perform in our work four experiment to measure the migration time of mobile agent between remote platforms where the mobile agent transfers the following sizes of txt files.
- 0 kb.
- 1 kb.
- 10 kb.
- 100 kb.
- 500 kb.
- 1 Mb.
- 4 Mb.
- 8 Mb.
- 16 Mb.

The experiments performed in the following situations:

- Mobile agent migrates from home location to destination location when the LAN was empty.
- Mobile agent migrates from home location to destination location when the LAN was busy.

- Mobile agent migrates from home location and visits seven destination locations, and then it will return back to its home location when the LAN was empty.
- Mobile agent migrates from home location and visits seven destination locations, and then it will return back to its home location when the LAN was busy.

12. The Experiments

The approach that we adopt to perform the first and second experiments was connecting two computers, the first computer was Dell, speed (1.18GHz), memory (3.46 GHz) with address 192.168.0.1 , and the second computer was ASRock, speed (3.06GHz), memory (504Mb, with address 192.168.0.2, The speed of the LAN was (10-100 Mbps), The first experiment mobile agent was resident at the first computer and then migrate to second computer when the LAN was empty and we implements all the situations that we mentioned before. We use in our work the Wireshark program. This program is used for monitoring and recording the inputs and outputs packages in/to the computer. We also calculate the difference in migration time of mobile agent between the first and second experiment, where the second experiment was implemented in the same manner of the first experiment except that the LAN was busy, see figure (4).

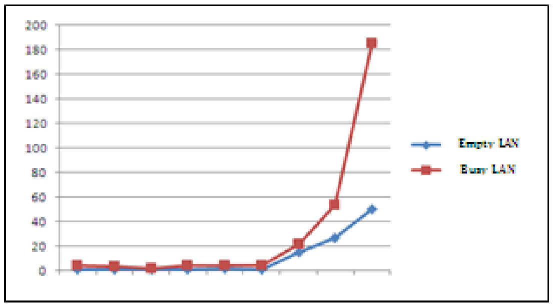


Figure (4) the average of migration time of mobile agent in the first and second Experiments

The approach that we adopt to perform the third and the fourth experiments was connecting eight computers with Network LAN Wired with star topology by using Network Interface Card (NTC) one card for each computer with speed (10-100 Mbps) and all computers connected to two switches each one have (8) ports of type (RJ-45). The used Transmission Media was UTP wire with speed (100 Mbps) where each computer is connecting to switch by wire. There are two connectors for each computers of (RJ-45) type one of the connected computers was Dell speed (1.18GHz), Memory (3.46 GHz) and the other computers were LG, speed (3.06GHz), Memory (3.46GB). The computers have this address: 192.168.0.1, 192.168.0.2, 192.168.0.3, 192.168.0.4, 192.168.0.5, 192.168.0.6, 192.168.0.7, 192.168.0.8.

We calculate the difference in migration time of mobile agent between the third and fourth experiment, where the fourth experiment was implemented in the same manner of the third experiment except that the LAN was busy see figure (5) .

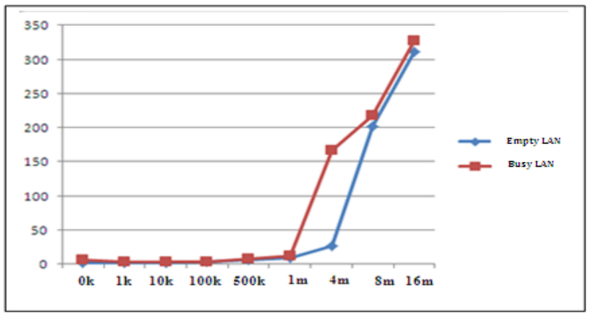


Figure (5) the average of migration time of mobile agent in the third and fourth Experiments

13. Conclusions

Through designing Collaborative Requirements (Col_Req) tool in this research, we extract several important issues:

- The Use of The Multi Agent Systems (MAS) to design Collaborative software engineering tools is so important and benefit because the Collaborative work style that found in multi agent systems was convenient with the Collaborative work that found in each phase of software engineering especially requirements engineering phase.
- The Use of the Multi Agent Systems (MAS) in designing (Col_Req) tool reduces the communication problems.
- Through the results that we achieve in our experiments to measure the migration time of mobile agent while it transfers a file in different sizes, we can say that the mobile agent can transfer different sizes of data and if that sizes of data are closed from each other (0k-1M) byte, then the migration time of mobile agent to transfer data was also closed and there was no big difference between them, But we can see the difference in migration time of mobile agent when it transfers a big files with big difference size between them (4M-16M) byte.

We see several requirements documents for many universal systems and we found that its size was less than (1M) byte. Therefore (Col_Req) tool is good in achieving Collaborative between requirements engineers for real time systems.

14. Future Works and Recommendations

- The most important further Recommendations are:
- Developing of (Col_Req) tool to contain requirements management phase to include tracking requirements process, this process is performed during all software development lifecycle.
- Developing of (Col_Req) tool to include other software engineering phases like design.
- Developing of (Col_Req) tool to be able to generate graphical documents.
- Developing of (Col_Req) tool to contain a private conversation channel to exchange private messages between its users.
- Developing of (Col_Req) tool to work on mobile devices where JADE offers this ability by using (Leap add-on).
- Developing of (Col_Req) tool to be work in offline state and save messages.

References

[1] Asghar .S, and Umar M., "*Requirement* Engineering Challenges in Development of Software Applications and Selection of Customer-off-the-Shelf (COTS) Components", International Journal of Software Engineering (IJSE), vol 1, Issue 2, 2011.
[2] Bellifemine1 F., Poggi A., and et al, ,"Developing multi-agent systems with a FIPA-compliant agent framework", John Wiley & Sons, Ltd, Italy, 31:103–128, 2001 .
[3] Cai L., Carl K., and et al, "Supporting Agent-based Distributed Software Development through Modeling and Simulation", The Ninth IEEE Workshop on Future Trends of ISBN, pp1, 2003.
[4] Cook C., "Towards Computer-Supported Collaborative Software Engineering", Ph.D Thesis, Submitted to the University of Canterbury Christchurch, New Zealand, 2007.
[5] Fabio B., Caire G., and Greenwood D., "Developing Multi-Agent Systems with JADE", wiley, Telecom Italia, Italy, 2007.
[6] Heinonen S., "Requirements managements Tool Support for Software Engineering in Collaboration", M.Sc Thesis Submitted to Department of Electrical and Information Engineering, University of Oulu, 2006.
[7] Liukkunen K., Lindberg K., and et al, "Supporting collaboration in the geographically distributed work with communication tools in the remote district SME´s", IEEE, 978-0-7695-4122, 2010.
[8] Maghrabi F., Faheem H. M., and et al, "Developing a Multiagent System for Integrating Biological Data Using JADE", International Multi Conference of Engineers and Computer Scientists Vol.(I), IMECS, Hong Kong, 2008.
[9] Outtagarts A.," Mobile Agent-based Applications: a Survey", IJCSNS International Journal of Computer Science and Network Security, VOL.9 No.11, 2009.
[10] Sarma, A., "A survey of collaborative tools in software development", (ISKTech. Rep. UCI-ISR-05-3). Irvine, CA: University of California, Irvine, 2005.
[11] Sol´ýs C. and Ali N., "Distributed Requirements Elicitation Using A Spatial HypertextWiki", IEEE, 978-0-7695-4122-8/10, 2010.
[12] Staines A.S., "A Comparison of Software Analysis and Design Methods for Real Time Systems", University of Malta, Msida, MSD, 2007.
[13] Stankovic J. A., "Real-Time Systems, Design Principles for Distributed Embedded Applications ", Second Edition, Springer, 2011.
[14] Su C., and Yi Lin T., "Toward Effective Global Sourcing A Mobile Multi- Agent Information Infrastructure for the Textile and Apparel Industry", Department of Industrial Engineering and Management, Yuan Ze University, China, , Kitakyushu, 2009 .
[15] Tsadimas A., Nikolaidou.M. And et al, "Handling non-functional requirements in Information System Architecture Design", Fourth International Conference on Software Engineering Advances, IEEE, 978-0-7695, 2009.
[16] Whitehead J.," Collaboration in Software Engineering: A Roadmap", IEEE, 0-7695-2829, 2007.
[17] Yang Q.," A Multi-agent Prototype System for Helping Medical Diagnosis", M.Sc Thesis, Submitted to the Memorial University of Newfoundland, 2008.
[18] Yi W., Ping J., and et al, "Study on Intelligent Decision-Making Platform in the Agricultural Production", Fourth International Conference on Intelligent Computation Technology and Automation, IEEE, 978-0-7695-4353, 2011.
[19] Zhang C., "A Software Collaborative Developing Environment based on Intelligent Agents", IEEE 978-1-4244-9857, 2011.
[20] Zhang C., Xi1 J., and Yang X., "An Architecture for Intelligent Collaborative Systems based on Multi-Agent", IEEE, 978 -1-4244-1651, 2008.