SETLabs Briefings
VOL 7 NO 5
2009

Managing Global Software Projects: The MAS Way

By Anjaneyulu Pasala PhD, Arun Sethuraman, Niranjani S and Ravi Prakash Gorthi PhD

Multi-agent systems are robust tools that address the challenges encountered while executing global software development projects

Software projects are increasingly getting executed in multiple geographic locations especially to take advantage of locally available specialized knowledge. This has led software project management tasks like work scheduling, inter-team communication, etc., to be more cumbersome. Therefore, project teams are looking for newer technologies and tools to assist them in various stages of software project management. We have designed a novel approach based on multi-agent technology that will assist the members of a project team in effectively tracing, monitoring and controlling project management processes alongside ensuring robust communication within teams. Based on the proposed approach we have developed an Agents Assisted Software Project Management tool. The results have indicated considerable productivity and quality gains over the traditional software project management tools.

In a global software development (GSD) setup, as projects are executed in direct coordination with clients, work is distributed among teams spread across multiple locations. In such scenarios, project management is a huge challenge given that multiple projects are managed concurrently.

Multiple factors affect project management in a GSD set up, some of them being different time zones, cross cultural issues, different languages, local legal issues (patents, etc.), different technical platforms used to develop software, etc. Therefore, these issues also need to be addressed additionally as a part of project management. These factors influence communication within the team. Therefore, the success of GSD depends on effective management of communication for information exchange, different and multiple skills, and work artifacts for effective sharing across teams beyond organizational and cultural barriers.

To overcome these challenges, project teams need new tools and technologies to assist them in various stages of software project management (SPM) viz., planning, development, testing, deployment, maintenance and re-engineering to help them cut costs while keeping an eye on the quality of service.

Our research on effective management of software projects using intelligent agents particularly multi-agent systems (MAS) has resulted in a prototype tool for quality review process [1, 2]. This tool effectively manages reviews across multiple project teams that are distributed globally.

Based on our research, it was found that the concepts and techniques of intelligent, cooperative, autonomous, independent decision making, adaptive and dialog based multi-agent technology can effectively address some of these challenges of project management in GSD. The MAS are known for the following advantages:

- Multi-agents effectively co-operate among themselves in providing solutions to the problems even in difficult environments viz., industrial furnace where they are required to read the temperature of furnace. They can also exhibit mobility within the system so that interoperability is achieved easily.

- Predictable and programmable behavior of agents result in credible performance of the MAS based systems.

- Multi-agents perform their assigned tasks consistently and efficiently 24X7 hours and thus reliability and repeatability becomes a primary characteristic of MAS behavior.

Also, owing to greater degrees of geographic distribution of software development teams, the use of wireless technologies as a means to enable ease of communication, anytime and anywhere has steadily increased [3]. It was found that the portability of agent technology onto light-weight wireless devices has been researched and is a challenging area, both in the field of artificial intelligence and in pervasive computing [4]. Hence, we propose an approach of deploying MAS onto personal digital assistants (PDAs) or other such light-weight devices that are capable of communicating with each other through wireless and wired communications. The aim is to considerably reduce the amount of manual effort that goes into a mundane task and to allow the agents to communicate effectively and efficiently. The proposed approach also has the built-in capability to handle scheduling when the users are disconnected from the network. The tool built on the basis of proposed approach exhibits a facet of the technology and has been successfully created to manage schedules among distributed projects teams.

RELATED RESEARCH WORK

The field of applications of MAS has been an active research area for over a decade and several researchers actively examined its utilities in the fields of business process management [5], transport facilities [6], simulation studies [7], software project management among others [1, 2, 8].

A generic framework of agent systems has been developed by Nienabar and Barnard to support the various aspects of SPM process [8]. Their framework proposes a set of specialized agents, namely, Personal Assistant Agent, Messaging Agent, Task Agent, etc., to assist

in implementing various activities of SPM in a globally distributed project environment. They present an abstract black-box design of some processes in SPM, where each black-box is composed of collaborative software agents that also interact between different black-boxes. This research shows promising results in enabling software developers to meet market expectations much faster and produce projects on time, within budget and to the users' satisfaction.

Sethuraman et al., have proposed a set of dedicated intelligent agents for assisting in the quality review process of software project management [1]. These agents are capable of performing the following tasks:

- Monitoring the work environment
- Periodically prompting actors in the project with review requests and consents to review
- Controlling the relay of review artifacts and their corresponding reviewer comments across the project teams.

The authors claim that 15% of time and effort was saved compared to the manual relay of information in a quality review process. Given the volume of projects executed by large software organizations, this is indeed a considerable saving on project execution costs.

Deployment of agent technology onto mobile devices has been experimented by several authors in tourism and mobile services applications [9, 10, 11]. Spanoudakis et al., in their work on the ASK-IT project, have developed an agent based approach for providing information services for tourism planning, travel, search for travel and accommodation services, among others [11]. They have proposed a generic agent architecture consisting of a set of mobile agents that support mobility impaired people who are on the move.

Lee et al., from British Telecom have implemented intelligent agents in their mPower project that facilitates faster communication among network and maintenance engineers during their work [10]. They have utilized this agent based approach to achieve workforce cooperation.

Our work focuses on the design and development of one such application of mobile agent technology – the meeting scheduling process, where we exhibit the advantages offered by this technology and present their application to a real life scenario.

MOBILE INTELLIGENT AGENTS BASED APPROACH TO PROJECT MANAGEMENT

We have used intelligent multi-agent technology on mobile devices to efficiently manage projects in the GSD context. We have proposed a set of dedicated personal assistant agents that assist team members, henceforth to be called as assistant agents. These dedicated assistant agents will be present at every hierarchical level of the project team, globally. Every assistant agent is equally capable of performing all the related activities of project management. Here we explain how MAS improves communication among team members. Each assistant agent is being fully capable of performing the following tasks:

- Establishing communication and exchanging information/data with other assistant agents in the system (communication being either information transfer, request for participation, accept/reject request, etc.).

- Accessing and using data available within the system as a whole i.e., the

assistant agent is interoperable across MAS knowledge base.

- Monitoring changes to environment variables in the system and reacting to those changes.

- Making pro-active decisions to avoid failure points by analyzing risks.

- Autonomous operation that makes a decision automatically and presents it to the owner without any manual intervention.

For instance, let us consider scheduling of a meeting in the context of global software development as one of the SPM activities to explain the usage of MAS in SPM. The following are the three scenarios that explain the disadvantages of present technology in detail:

1. First, when user A sends a meeting request to user B and B is not at her desk or on leave or offline, the meeting request is entertained only when B returns to work or checks her email manually. It might so happen that B is back after the scheduled meeting time and A does not get the response until B responds. This makes user A wait indefinitely without knowing the status of her request.

2. Second, whenever a user wants to set up a meeting, she would laboriously remember time zones manually for all the invites that are at different locations and carefully select the timings.

3. Third, whenever a high level meeting request is received by a user who has already set up a meeting during the same time with her peer, this meeting has to be postponed. Presently, the system does not allow automatic rescheduling of meetings. Therefore, members have to setup manually and every invitee has to respond again.

Agent technology addresses this problem in a manner where there will be an assistant agent deployed in each user's mobile device, laptop or desktop along with a server agent at calendar server [Fig. 1].

Scheduling is done by allowing these assistant agents to access the calendars or schedules of respective users. The meeting is setup automatically through collaboration among these assistant agents upon initiation by respective assistant agent of the initiating user. The invitees' agents, upon receiving a meeting request from initiation agent, proactively communicate with the other assistant agents and arrive at a decision or automatically accept or reject the request based on availability of the user without the knowledge/interaction from the user. However, in case the connectivity with a required assistant agent has been lost, the assistant agent queues up the meeting request (centrally with the server agent in this case) and automatically synchronizes with the required assistant agent as and when it enters the system again. These interactions are shown in detail in Figure 2.

The detailed explanation of the interactions among various assistant agents is given below.

The meeting is scheduled by means of calendar bookings that work on a proposal-negotiate and confirm protocol. Assume that the project manager (PM) wants to set up a

project reviewing meeting. There are two ways, in which the communication can take place between the PM i.e., the user and assistant agent as explained hereunder.

Explicit Communication: In explicit communication the user explicitly informs her assistant agent about the timings of meeting and list of invitees required for the meeting. Agent sends the meeting request to required invitees and gets replies from these invitees' assistant agents and arrives at a decision through consensus and informs the user about the same.

Implicit Communication: In case of implicit communication, the assistant agent automatically picks up the meeting request from the user once the meeting is scheduled by the user using calendar. It proceeds further in calculating the timings of meeting based on their time zones and getting the other required details and negotiates with invitees' agents and finds a suitable time between them for the meeting to be scheduled.

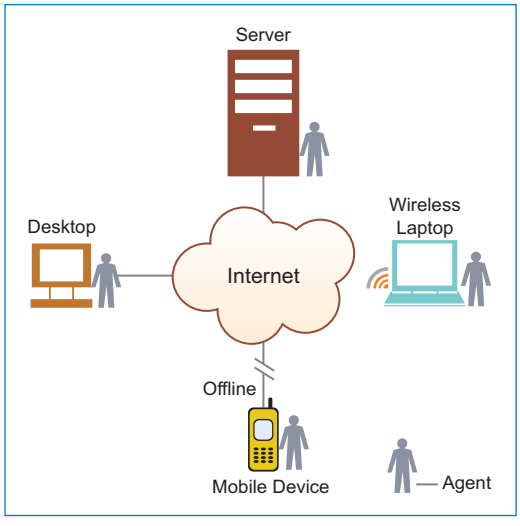

Figure 1: *Multi-agents based System of Collaboration*
Source: *Infosys Research*

Detailed below is the algorithm that depicts the negotiations between these assistant agents.

ALGORITHM

Setup Agent – Setting up a Meeting: Setup agent is the assistant agent of the user who sets up the meeting schedule and invitee agents are those assistant agents of the users to whom the meeting request has been sent.

1. User sets up a meeting using calendar by indicating the preferred timings of the meeting and invitees list.
   a. Assistant agent gets the meeting request implicitly.
   b. This assistant agent sends the meeting request to assistant agents of invitees by automatically capturing their locations and converting the timings in their local timings.
   c. Waits for the responses from other assistant agents.
   d. Receives the responses from other assistant agents.
      i. If all requests are accepted, the meeting is scheduled
      ii. Else if there is a reject for meeting request from a user of higher hierarchy, the meeting is cancelled and user informed who then sets up the meeting or acts accordingly
      iii. Else if there is a meeting request for rescheduling from one or more members, the calendar of the other users is checked and if they are free, the meeting request is re-sent to all the members following step (1.b) with new timings.
      iv. Else as there are rejects from few unimportant members go ahead

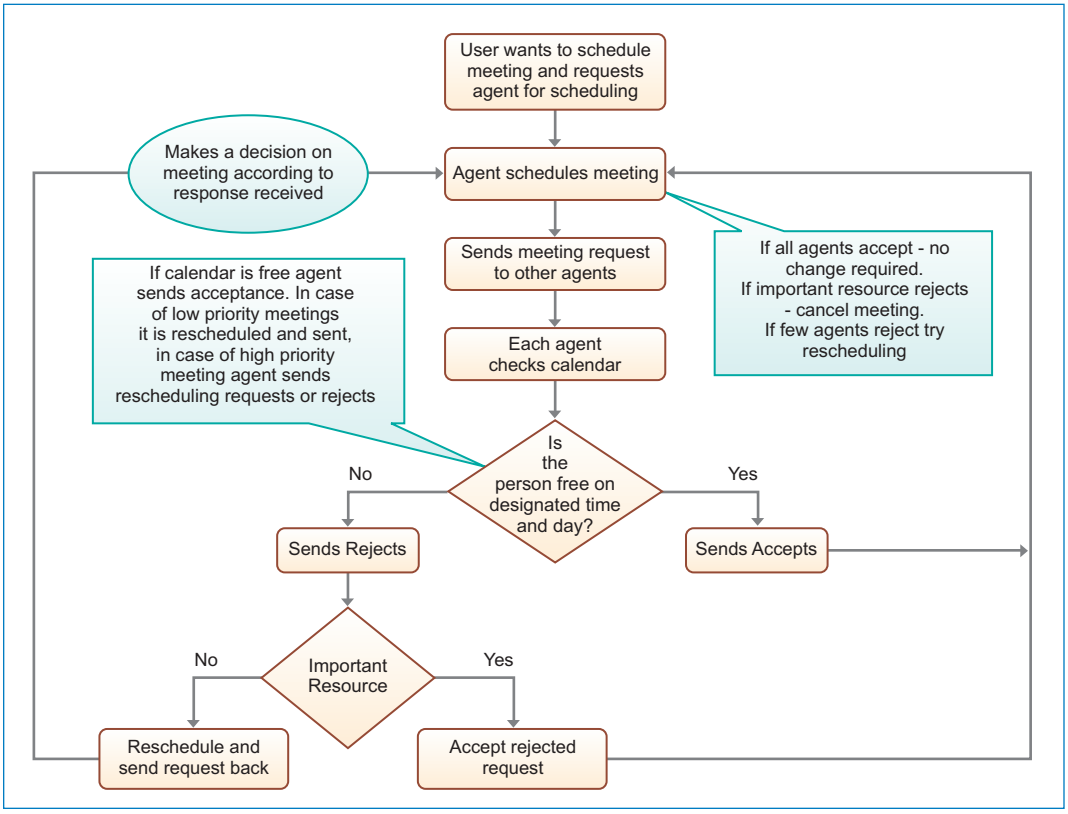

Figure 2: *Detailed Interactions among Agents* ***Source:*** *Infosys Research*

and schedule the meeting.

2. The user requests the assistant agent to set up the meeting with preferred data and timings, by giving the required number of hours for the meeting along with the invitees list.
   a. User requests to set up the meeting (for e.g., for one hour in the morning for a given day or in next two days)
   b. The setup agent would check busy/free time from her calendar.
   c. Negotiates with other invitee agents and fixes up the time.
      i. Setup agent finds busy/free hour of the user in the morning
      ii. Setup agent informs invitee agent of the busy/free time of the user
      iii. Invitee agents reply the setup agent of the free time of the invitee
      iv. The setup agents finds a common time between the free time of the user and invitee

If there is a suitable common timing found then setup agent sends the meeting request that would be accepted by both setup agent and the invitee agent.

If the suitable common time is not found the setup agent informs the user about the non availability of the common time and suggests common timings between them and go back to step iv.

Invitees Agent — Accepting/Rejecting/Re-scheduling Meeting Request

1. Invitee agent receives the meeting request
2. Checks the calendar of the user
   a. If the calendar is free, it sends out the acceptance message.
   b. Else if the calendar is busy, it checks the priority of the meeting that is already set up at that particular time.

If the already set up meeting is of low priority, initiates rescheduling of meeting by sending a fresh meeting request to the invitees in that meeting.

Else if the meeting is of high priority, then reject message is sent. If the reject message is from a user of higher hierarchy, then the meeting is cancelled and information is sent to every invitee.

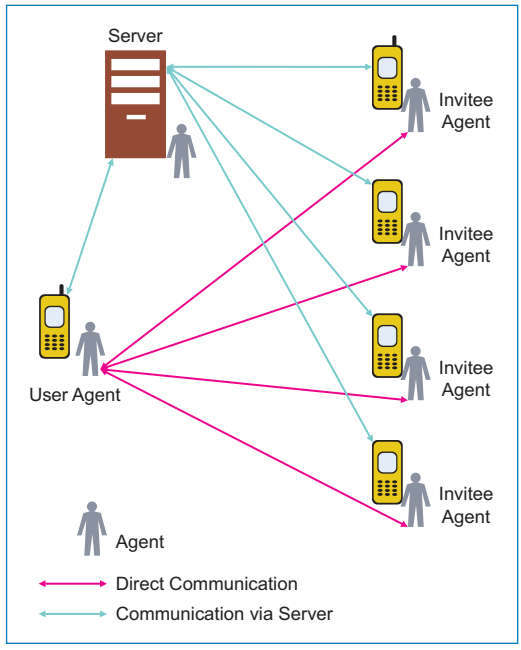

Figure 3: *Types of Communication between Agents through Server Agent*
Source: *Infosys Research*

CONTEXT AWARE MOBILE AGENTS

When the user is working on a wireless enabled device, for instance, a laptop or a mobile phone while on the road, she may experience disconnections in the communication and thus cannot communicate with other invitees in the list. When connectivity is lost, the assistant agent will queue up all such requests and correspondingly synchronize with the server agent i.e., the agent residing on the server, as and when the communication is re-established. The server agent in turn will be able to forward the requests to other assistant agents on re-establishing connection. In this manner, assistant agents need not spend time indefinitely on synchronizing their requests as they both simultaneously need to establish connection at the same time. Synchronization can be speeded up by using the server agent. This scenario is depicted in Figure 3.

An instance of communication between agents during disconnection is explained here:

- A server agent S is connected with two assistant agents A and B
- Upon registration with the server agent S, agent A wants to schedule a meeting with agent B
- Meanwhile agent B goes offline and thus, the scheduling cannot be achieved right away
- Hence, the server agent S queues up the meeting request until agent B re-registers with it and automatically meeting request is forwarded
- Agent B takes a decision on the request If agent B is in a different time zone or has overshot the period within which agent A wanted to meet, then agent A

accordingly calculates new timings for the meeting and repeats the scheduling process with the agent B

- Agent B communicates the decision on the meeting request to the server agents whenever it connects to it
- Server agent S forwards the same to the assistant agent A on getting connected to it
- Assistant agent A takes a decision based on the response from assistant agent B.

The architecture of the agent based system has been chosen by analyzing the above requirements of the system and is discussed in the following sub-section.

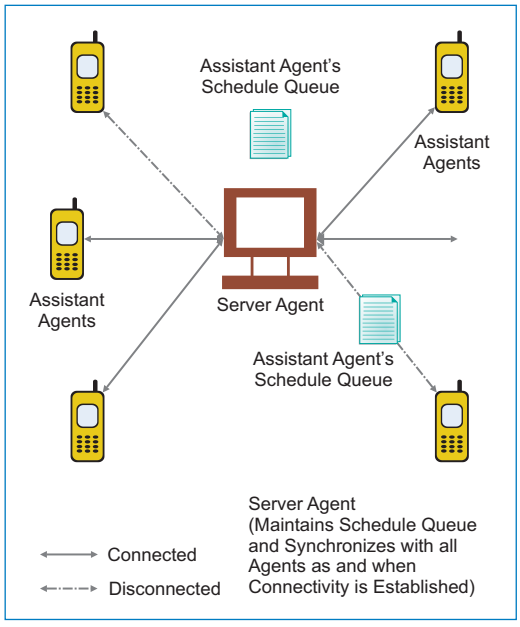

Figure 4: *General Architecture of Communication Relay*
Source: *Infosys Research*

Architecture of Approach

Multi-agent architectures are realized in four different types, based on the system's requirements. They are:

Market Mechanism: Where tasks are matched to agents by a generalized agreement or mutual selection

Contract Network: Where agents are either managerial or contractors, with the manager agents negotiating activities between the contractor agents

Multi-agent Planning Systems: Where all agents have planning capabilities and their behavior is mapped by this centralized or decentralized plan

Organizational Structure: Where each actor in an organization has one or more assistant agents that have different capabilities based on the actor's requirements.

On analyzing the pros and cons of these four types, the organizational structure based approach has been chosen to manage communication in project management. Also, we have proposed a server agent that resides at the central server to manage the agents in the organizational structure. Hence, every agent in the system registers with the server agent when it enters the system. The introduction of the server agent is for centralized accountability. The advantages of this architecture are:

- The registration and searching process for assistant agents across the network happens only once
- Once the search and registration process is complete, the other assistant agents have access to the new assistant agent's credentials and hence communicate with them directly

- All assistant agents are similar and hence can be instantiated easily across the network

- In a disconnected scenario, when the assistant agents schedule meetings, these are queued up centrally with the server through server agent and hence do not have to be scheduled in individual queues.

MODELING MAS USING GAIA

We choose to model this agent-based systems using GAIA methodology to enable us to analyze and implement the requirements [12, 13]. The GAIA methodology deals with the modeling of both internal and external operational aspects of the system components and the system as a whole. It defines the structure of an MAS based system in terms of a *Roles Model*, wherein it identifies key role players within the MAS and the interaction protocols between them. Each *Role* is composed of the following four attributes:

Responsibilities — expressed in terms of:

- *Liveness Properties* - the tasks that the agent role must fulfill given certain environmental conditions.
- *Safety Properties* – those which ensure an acceptable state of affairs maintained during the execution cycle.

Permissions — screening information resources accessible to agents

Activities — tasks that the agent performs without interacting with other agents

Protocols — patterns of interaction between different agents in MAS. Each protocol is composed of - purpose, initiator, responder, inputs, outputs and processing.

The following roles have been identified for the meeting scheduling process within the generalized project scheduling scenario:

Server Agent: The server agent monitors the environment and sends out registration requests as and when other agents are detected within the range of the server. Post registration, this agent coordinates the meeting request queues that have been pending from the different agents in the system. It resolves conflicts and calculates a mutually convenient time for the meeting and books the meeting.

Assistant Agent(s): These are the personal assistant agents that are deployed onto the devices of individuals in the project. These agents are capable of registering with the server agent, calculating a convenient meeting time that is based on the user's availability, accounting for difference in time zones and offline time and beaming out the same back to the server agent.

DESIGN AND DEVELOPMENT USING JADE

We have used the Java agent development environment (JADE) platform to design, code and test the implementation of the multi-agent system prototype for calendar and meeting management [12]. JADE is a JAVA based software development framework to develop applications based on the multi-agent paradigm.

We represent each simple or complex behavior as a role. A 'ω' in a liveness formula in the GAIA role model will thus mean that the behavior has to remain in the scheduler until the role that initiated this behavior removes

it. Similarly, a '||' would mean that the two behaviors have to be executed concurrently. Based on these principles, we have designed various agent communication language (ACL) messages such as MeetingRequest, RegistrationRequest, MeetingConfirmation, etc., the data structures and the behaviors of the modeled roles.

IMPLEMENTATION

A JADE agent is implemented as a single thread and agent tasks are modeled as subclasses of the 'Behaviour' object. Then, using the 'addBehaviour(Behaviour)' method, we add to the task-queue of a specific agent at any time during execution. An internal scheduler of agents has been created, which carries out a least recently used (LRU) based preemptive scheduling policy. The structure of each agent is in the form of a closed finite state machine. Each step for each of the agents is defined and the agents react sequentially to follow the quality review process model. JADE agents can hence be deployed and launched on the same host and on a remote host that connects itself to the main container of the agent platform. The system hence emulates a distributed system that appears like a single agent platform from the outside.

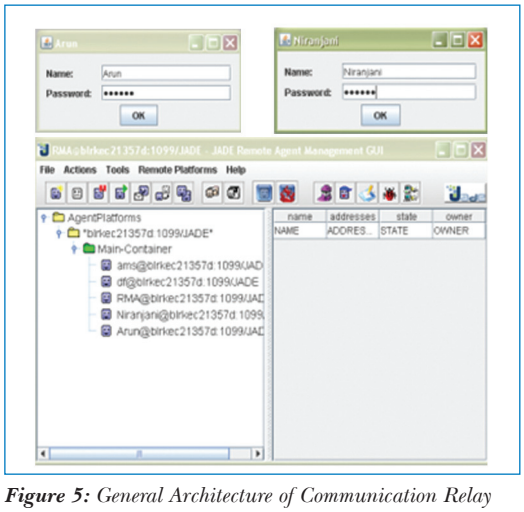

Figure 5: *General Architecture of Communication Relay*
Source: *Infosys Research*

The screenshot shown in Figure 5 exhibits the agents that have been deployed onto a single machine and the negotiation process between them towards scheduling a meeting. Figure 5 shows a typical scheduling screen, where the agents display a scheduler window that would aid the team members to specify their preferences for timings. The agents then keep track of the user's preferences and correspondingly schedule the meetings.

CONCLUSION

This discussion focuses on a solution framework based on MAS on wireless devices to address the new challenges such as efficient and effective communication faced by the PMs during execution of globally distributed software projects. Based on the proposed solution, a calendar scheduling has been designed and implemented using GAIA methodology and JADE platforms.

We have chosen the scheduling problem to merely exhibit the possible applications of this technology to the field of SPM. Further research involves in deployment of the work in live projects and analyzing how much cost savings can be achieved using MAS in SPM.

REFERENCES

1. Arun Sethuraman, Krishna Kumar Yalla, Ankur Sarin and Ravi Prakash Gorthi, Agents Assisted Software Project Management, in Proceedings of the 1st Bangalore Annual Compute Conference, 2008
2. Anjaneyulu Pasala, Arun Sethuraman,

Niranjani and Ravi Prakash Gorthi, Context-aware Mobile Agents in Software Project Management, IEEE TENCON 2008

3. Bob Emmerson and David Greetham, Computer Telephony and Wireless Technologies: Future Directions in Communications, Computer Technology Research Corporation, First edition, 1997
4. Habin Lee, Patrik Mihailescu and John Shepherdson, Realizing Teamwork in the Field: An Agent-Based Approach, IEEE Pervasive Computing, Vol 6, No 2, 2007
5. Nicholas R Jennings, P Faratin, T J Norman, P O'Brien and B Odgers, Agent Based Business Process Management, International Journal of Applied Artificial Intelligence, 2000, Vol 14, No 2
6. B M Balachandran and M Enkhsaikhan, Development of a Multi-agent System for Travel Industry Support, International Conference on Computational Intelligence for Modelling Control and Automation, and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, 2006
7. N Cannata, F Corradini, E Merelli, A Omicini and A Ricci, An Agent-oriented Conceptual Framework for Biological Systems Simulation, NETTAB Workshop on Models and Metaphors from Biology to Bioinformatics Tools, September 2004
8. R Nienaber and A Barnard, A Generic Agent Framework to Support the Various Software Project Management Processes, Interdisciplinary Journal of Information, Knowledge and Management Vol 2, 2007
9. Mathieson, L Padgham and B Q Vo, Agent Based Travel and Tourism Planning, AAMAS'05, Utrecht, Netherlands, ACM, 2005
10. Habin Lee, Patrik Mihailescu and John Shepherdson, Realizing Teamwork in the Field: An Agent-Based Approach, IEEE Pervasive Computing, Vol 6, No 2, 2007
11. Pavlos Moraitis and Nikolaos Spanoudakis, Argumentation-Based Agent Interaction in an Ambient-Intelligence Context, IEEE Intelligent Systems, Vol 22, No 6, 2007
12. Pavlos Moraitis and Nikolaos Spanoudakis, The GAIA2JADE Process for Multi-Agent Systems Development, Applied Artificial Intelligence, Taylor & Francis, 2006
13. Franco Zambonelli, Nicholas R Jennings and Michael Wooldridge, Developing Multiagent Systems: The GAIA Methodology, ACM Transactions on Software Engineering Methodology, Vol 12 No 3, July 2003.

Authors' Profile

ANJANEYULU PASALA

Anjaneyulu Pasala PhD is a Senior Research Associate at SETLabs, Infosys. His research interests include Software Engineering and Software Verification and Validation. He can be reached at Anjaneyulu_Pasala@infosys.com.

ARUN SETHURAMAN

Arun Sethuraman was a Junior Research Associate at SETLabs, Infosys. His research interests include Intelligent Multi-Agent Systems and Phylogenetics.

NIRANJANI S

Niranjani S is Software Engineer in Test Automation Lab at SETLabs, Infosys. She can be reached at Niranjani_S@infosys.com

RAVI PRAKASH GORTHI

Ravi Prakash Gorthi PhD is a Principal Researcher with SETLabs, Infosys. His research interests include Knowledge Engineering and Model Driven Software Engineering. He can be reached at Ravi_Gorthi@infosys.com.

For information on obtaining additional copies, reprinting or translating articles, and all other correspondence, please contact:

Telephone: 91-80-41173871

Email: SetlabsBriefings@infosys.com