

USING INTELLIGENT AGENTS FOR KNOWLEDGE MANAGEMENT IN GLOBALLY DISTRIBUTED SOFTWARE DEVELOPMENT

Pelluri Sudha, Osmania University,
Sudha392623@yahoo.com

ABSTRACT

A large part of all software development work being done today is globally distributed among geographically distributed teams. The present day project and product development demands availability of experts at specific locations as per the need of domain and technology involved. In such a multi-site development environment, the most important asset that can distinguish the successful projects from those that are not is – knowledge management. In view of the business competitiveness, naturally any organization trying to be a winner in the long run, such frameworks or technologies that foster their efficiency are very much required and are in demand. At every stage of software development it is the availability of this knowledge regarding processes, procedures, techniques, program components and technology and also the efficient utilization of the same in decision making, reusing and task parsing that holds key to success of individual projects and to the organization at large. The huge amount of knowledge that can accumulate within an organization over a period of time, through out the geographically distributed sites, across various domains, in various areas and different technologies can be huge. This necessitates suitable accumulation, mapping of that knowledge to suitably to the process work flow, storage, retrieval and more importantly, doing it such that the user of this distributed knowledge is able to leverage the advantage of this large amount of knowledge base available. There is a necessity to use an intelligent agent to be able to make abstractions and analogies between problems and past experiences, using cognitive knowledge available in the knowledge base and hence minimize human intervention and enhance the planning and management capabilities of the project managers.

Key words:

Knowledge management, Knowledge flow, Intelligent agents.

1. INTRODUCTION

Availability of qualified work force, cheaper resources, innovation and markets at distributed locations in the world has led to today's global software development environments. Many of the software organizations have gained the advantages of such an approach like the cycle-time acceleration, global resource pool, catering to local markets at cheaper rates.

Global software development also presents its own challenges and complexities in various areas like technical, organizational, and cultural issues and also those problems which basically depend on different time zones and languages. So, there is a need to focus on the engineering and management aspects of such distributed software development. There is now quest for techniques, technologies, frameworks and processes that can help organizations gain an advantage over their competitors, gain an edge over others because, Darwin's law of "Survival of the fittest" applies very much to all business houses.

As mentioned by Damian.D and Moitra.D in [2], with the intensification of globalization and the resulting growth in globally dispersed software development, the need for understanding the engineering and management approaches necessary for successful global software development has only become more pronounced. Also, over the years, the approaches and practices involved in organizing and managing global software development have undergone refinement, and new, effective practices have emerged.

As stated by Alho. K and Suleno.R in [1], Current distributed projects often incur substantial overhead in time and cost due to increased communications, management effort, misunderstandings, rework or lack of standards. Often this even results in lower quality in the results. In order to really save costs (e.g. through decreased travel or increased parallelism in development process), better methods and tools to manage distributed development projects are needed.

Gary and Judith Olson [8], while ascertaining that "distance matters," synthesize four concepts that will be crucial for

geographically distributed work: *collaboration readiness, common ground, coupling in work, and technology readiness.*

Collaboration readiness is the ability of an overall software development governance model to set business goals that easily translate into maintainable interdependent tasks across geographically dispersed teams. A firm's choice of a specific governance model at both the organizational and project levels plays an important role in influencing project performance.

Common ground reflects the shared knowledge of distributed-development participants who reside in distant locations. When personnel are separated by distance, establishing common ground is essential to avoid potential communication gaps that could deteriorate product development performance.

Coupling in work refers to the mechanisms for dividing labor in distributed product development. Managing complexity in work is a prerequisite for achieving efficient division of labor. The distributed product development process must result in guidelines for the software manager to dynamically control product modularity and the modularity's span of remote teams.

Technology readiness refers to the development infrastructure and personnel capability levels for using collaborative technologies such as groupware communication systems, videoconferencing, synchronous multi-authoring tools, and group debugging tools.

Distributed team software development is a kind of software development management paradigm that focuses on work co operation and resource sharing among the geographically distributed team members during the development process.

2. BACKGROUND

2.1. Knowledge and its management

Knowledge is the key determinant of future projects' success, as it aids organizational learning. Till recently, one of the most important resources for any organization was the human capital. This was because of the knowledge they brought and accumulated over a period of time. The present scenario of globally distributed work makes it essential to be able to store the knowledge accumulated for two basic reasons – to be able to reuse the knowledge gained and also to ensure that knowledge of the organization only accumulates and is not lost when the critical team member leaves the organization. Unless great emphasis is given to this, business competitiveness would make any organization obsolete, inefficient and probably threaten the very existence of an otherwise competitive organization.

As mentioned by Desouza.K.C. and Evaristo.J.R. in [4], in the realm of project management, much of the effort in incorporating technology has involved fostering ubiquitous communication among team members while facilitating knowledge exchange. Knowledge has been categorized as – knowledge in projects, knowledge about projects and knowledge from projects. So, it includes schedules, milestones, meeting minutes and training manuals, cost benefit analysis, deadlines, customer commitments and expectations. It also includes post hoc analysis and audit of key insights generated from carrying out projects. A hybrid approach to knowledge management which helps maximize the benefits of the centralized and peer –to – peer approaches.

As stated by Zhuge.H [11], Knowledge management specifies rules and provides functions for information gathering, access, update, retrieval, organization and archival .It also provides functions for information integration from different sources. The actual data and information are stored in a document repository to manage multiple projects over time. .By collecting data and information from multiple projects KM allows project managers to compare or aggregate information across projects to derive patterns and thus create knowledge.

As mentioned by Desouza.K.C. *et.al* [3] Common knowledge management problems in global software efforts include the inability to seek out relevant knowledge, poor aggregation and integration procedures for knowledge synthesis, and delays or blockages of knowledge transfer. Unless organizations manage knowledge and leverage it appropriately in global software development efforts, the effectiveness is lost and would become a burden rather than a competitive advantage. Different models have been identified as client-server model, peer- to –peer model and hybrid model in for setting up a knowledge management system(KM s) .For deploying the KMs three primary models have been identified as centralized model, shared – services model, and watcher model.

2.2. Software agents

A software agent is a piece of software that acts for a user or other program in a relationship of agency. Such “action on behalf of” implies the authority to decide when (and if) action is appropriate. The idea is that agents are not strictly invoked for the task, but activate themselves. Various authors have proposed different definitions of agents. These commonly include the concepts such as:

- i) *Persistence* – Code is not executed on demand but runs continuously and decides for itself when it should perform some activity.
- ii) *Autonomy* – Agents have the capabilities of task selection, prioritization, goal-directed behavior, decision making without human intervention.
- iii) *Social ability*- Agents are able to engage other components through some sort of communication and coordination; they may collaborate on a task.
- iv) *Reactivity* – Agents perceive the context in which they operate and react to it appropriately.

Related and derived concepts include *intelligent* agents (in particular exhibiting some aspect of Artificial Intelligence, such as learning and reasoning), *autonomous* agents (capable of modifying the way in which they achieve their objectives), *distributed* agents (being executed on physically distinct machines), *multi-agent systems* (distributed agents that do not have the capabilities to achieve an objective alone and thus must communicate), and *mobile* agents (agents that can relocate their execution onto different processors)

- i) *Intelligent agents* - The design of intelligent agents is a branch of Artificial Intelligence research. Capabilities of intelligent agents include :
 - a) *Ability to adapt* - Adaptation implies sensing the environment and reconfiguring in response. This can be achieved through the choice of alternative problem-solving -rules or algorithms, or through discovery of problem solving strategies. Adaptation may also include other aspects of an agent’s internal construction, such as recruiting processor or storage resources.
 - b) *Ability to learn* – Learning implies a capability of introspection and analysis of behavior and success. Alternatively, learning may proceed by example and generalization, hence a capacity to abstract and generalize.
- ii) *Autonomous agents* – Autonomous agents are software agents that claim to be autonomous, being self-contained and capable of making independent decisions, and taking actions to satisfy internal goals based upon their perceived environment. All software agents in important applications are closely supervised by people who start them up, monitor and continually modify their behavior, and shut them when necessary.
- iii) *Distributed agents*- Since agents are well suited to include their required resources in their description, they can be designed to be very loosely coupled and it becomes easy to have them executed as independent threads and on distributed processors. They become distributed agents and the considerations of distributed computing apply.
- iv) *Multi-agent Systems* – When several agents interact, they may form a multi-agent system or, multiple agent system. Characteristically such agents will not have all data or all methods available to achieve an objective and thus will have to collaborate with other agents.

3. DESIGN OF THE NEW MODEL

3.1 The Need

Though independently there has been research on different aspects like what is knowledge, what models can be followed for accumulation, underlining the need for knowledge management, knowledge sharing, few have proposed a knowledge flow. But, actually for global business competitive edge, focus is actually needed on: Accumulation of vast amount of knowledge in the complex knowledge grid, continuously making current knowledge available for future use, reuse and sharing of accumulated knowledge hence, good abstraction mechanism is required when

cognitive knowledge is accumulated in the grid. For the abstraction of required knowledge, use of intelligent agents would be most suitable technology. In this paper the efforts are in that direction.

Here the approach stated for knowledge management utilizes two basic concepts- ***Knowledge flow and intelligent agents***.

3.2. Knowledge flow

The most important concept that is to be addressed in the design of a knowledge flow management is the fact that it is dynamically changing and also, it is flowing, not available at only at a place where it is generated or the source. Any one of these not being present the current situation would not be adequately represented. Apart from maintaining knowledge in projects, about project and from projects, the important addition to the existing scheme of knowledge flow maintenance, it is essential to keep specialized knowledge repositories. These specialized bodies can be called “specialized centers”. These are useful to transform cross- product experience into coherent and repeatable body of knowledge. This availability will lead to better and easy use of the collection of skilled resources in a specific area.

Regarding knowledge being treated as a continuous moving entity, Zhuge.H [11] suggests that, a knowledge flow is a carrier of human knowledge, which passes a team member’s knowledge to the succeeding team member according to definite process logic, shares its content with the receiver and accumulates knowledge of the team members. A complete knowledge flow passing through all team members during a team development process constitutes a knowledge flow network .The output of a knowledge node is a knowledge flow that depends on corresponding team member’s cognitive ability and the input knowledge flow. The propagation of the active nodes on the KNF usually keeps pace with the execution of the corresponding workflow process.

This approach is different from the traditional concept of knowledge existing as database in distributed form. Here it is to be noted that this concept is unlike the other simple ways of sharing information like the E-mail and Blackboard approaches.

Advantage over E- mail based and Blackboard based approaches –

Using the knowledge flow approach , team members of a development team can make use of the tools and also knowledge i.e., experience, decision, method, skill of the predecessors and the experience of the team accumulated during development of previous projects, so redundant work is avoided, this information accumulation characteristic of the knowledge flow enables a team to keep its cognitive ability in case of changing team members and other benefits of globally distributed work are realized to the fullest.

3.3. Intelligent agents

The knowledge flow scheme can be implemented by using the intelligent agents. Mobile agents have been under research, solutions for fundamental problems have been proposed, some have been implemented in the recent past. Agent based solutions are exactly applicable to the current dynamic and complex environment since they are appropriate in highly dynamic, complex, centralized, work as well in distributed environments. In addition to the advantages of distributed and concurrent problem-solving, agent technology has the advantage of sophisticated pattern of interaction, namely cooperation, coordination and negotiation.

Some suggestions where agent based solutions are suggested are available in:

Hartmut V and Buchmann .A [5], have discussed X-TRA, an extensible, trusted agent system and how it can be scaled for large distributed transaction based system.

Berger .M *et al* [6] , have shown the main issues in downsizing the Lightweight Extensible Agent Platform (LEAP) to PDAs, mobile phones and applying the agent technology in the area of virtual team support.

Connor .R and Jenkins .J [9] have proposed an architecture and framework for use as the basis of an intelligent agent assistance system for use by software project managers and a series of user trials of this prototype were also conducted.

Berger .M *et al* [7], have described specific results of the project LEAP. They are- a generic service composition framework, integration of the framework into the multi-agent system, two applications in the area of mobile workforces and results of the corresponding field trials.

3.4. Design issues

To start implementing the knowledge flow system using agents, an architecture level specification is required. The architecture is given in Fig.1. The actual code level specification can be worked out to suit this framework. The specification here is only at the framework level. This design is a high level and preliminary design. Various specific implementation dependent issues will be an improvement the need to be proposed as an extension of current work.

The basic components required are:

Knowledge Grid: Basic constituent of the knowledge base should be the knowledge grid. The actual implementation of the knowledge grid would take into account issues pertaining to distributed database security, replication, fault tolerance, A knowledge grid that includes a knowledge space for storing knowledge and a knowledge operation interface for sharing and managing the knowledge in space is required. A local knowledge grid serves for one or a group of people. As stated by Venugopal .S *et al.* [10] , architecture of data grids and the fundamental requirements of data transport mechanisms, data replication systems, and resource allocation and job scheduling are discussed which enable sharing data and collaboratively managing and executing large-scale scientific applications that process large data sets distributed around the world.

Specialized centers: These are at the nodes of the grid. The coordinates of such nodes will be available to the agent controller. User requirements can be satisfied through the agent, which is made available by the agent controller after having accessed the knowledge grid suitable to the users' need. Abstraction of knowledge from these points is made easy and useful to the most frequently required centers by implementing the framework and grid layout using intelligence embedded into the agents. Implementation of the same with suitable language, thin client and suitable back end is required.

Agent controller: responsible for controlling entire set of agents, scheduling and execution, work in co-ordination with the agent library and centers of expertise to present solutions to the user.

Agent library: the purpose of the agent library is to manage the agents and to service the request for agent interactions from agent controller. A map of agents from the knowledge base consisting of unique or reusable knowledge components is through these agents stored in the library.

User interface: responsible for managing front end element, validation of data entered by users, passing on clear messages to rest of system and displaying as per specified and standard formats.

A CORBA bus can be used for transparent access to distributed access to heterogeneous network of systems and platforms. The GUI can be ported to a client machine and the agent library and knowledge base can typically reside on the server which maintains replicas of knowledge base of more frequently accessed centers of expertise. Sequentially the workflow is mapped stepwise by the agent controller from the knowledge domain.

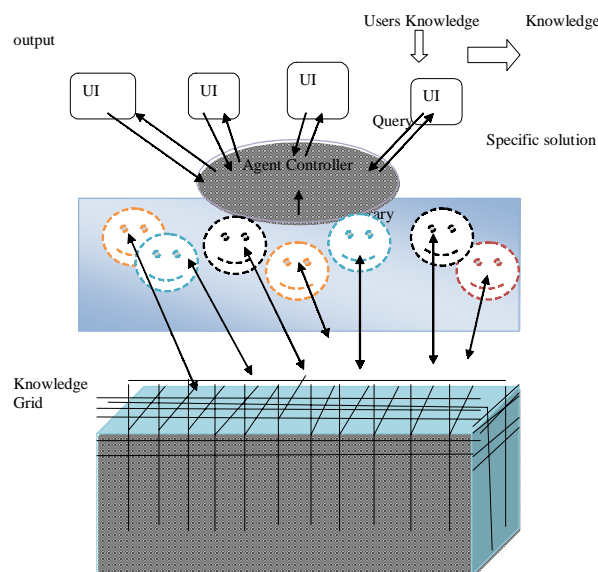


Fig.1: Architectural specification of agent based knowledge management model

4. GAPS THAT AGENT BASED KNOWLEDGE FLOW APPROACH WILL ADDRESS

Basic idea is to have an agent based framework which will encapsulate expert knowledge and make it available to all users.

1. Some of the potential benefits of this approach as applied to the decision-making process in the domain of software project management are:
 - a) Suggestions are made which help the user to balance cost, quality and time in making decisions about the use of project resources - naturally that is the most important aspect of Project Management
 - b) Benefits of sharing knowledge -about different lifecycle models and why one or another may be most suitable for the users projects.
 - c) Success of a project needs to be measured, measurements are suggested which will enable the user to see how well the project is reaching greater organizational goals and re-plan the ways to reach these goals, if necessary.
 - d) This approach is of great benefit when training new personnel, when on- the job training is impossible, when a bad decision can be disastrous.
2. Best Utilization of previously accumulated knowledge, knowledge of relevant area specific to certain users is dynamic yet always updated. Hence always and at any point in time current knowledge would be available.
3. Specifically, the knowledge flow scheme that is being proposed would take current state of knowledge available in the grid, present inputs, specific to current project and output at the end of the knowledge node is sent to a team knowledge repository. Team members can query the required knowledge base according to-project version, revision version and date. Hence, each team member can use the experience of their predecessors and experience of team accumulated during the development of previous projects.
4. Using the intelligence that can be embedded in these agents, which can be made to specialize in a specific problem area, best fit solution to our requirements can be obtained. This abstraction can be based upon the comparisons of a set of attributes, available on the knowledge grid at specific “specialized centers”.
5. It has been found that in the present environments of distributed resources, distributed analysis, design and implementation, and geographically separated work force, even the most experienced project manager may have difficulty knowing the best planning options, even if the critical input parameters of resources, constraints and requirements are known. Such new support systems will provide for the distributed cross-platform nature of modern client-server development.

5. CONCLUSIONS AND IMPLICATIONS

In today’s business scenario it is important to focus on continuous improvement in the management of software development projects. Such a focus will enable firms to adopt newer business models, such as distributed development, as market conditions demand. As globally dispersed teams become pervasive, top management needs frameworks, techniques apart from technologies to enhance its performance and to initiate activities for continuous improvement in management of such teams.

Whether in a single organization or across several working in partnership, project and process management issues associated with defining integration and synchronization points are still an issue for many distributed teams. Incompatible data formats and exchanges remain a challenge although, some tools and resource sharing systems are made available in recent years.

This knowledge framework with knowledge repository available on the grids, abstraction using agents is an attempt to satisfy this need and develop a rigorous distributed product development model. Though there is a great demand for such advanced schemes that can bring about great changes in businesses, it is difficult to incorporate this approach across various categories of business houses mainly because of certain amount of uncertainty that comes with anything that is new and obviously that which could have great impact. Organizations must expand the knowledge search and utilization behaviors from local to global levels. They must also motivate and train individuals

to use the Knowledge Management Systems. There is a need to actively attempt to take the lead and then reap the advantages. There is still need for us to bring this into practice and stand ahead in global competitiveness. This model of Knowledge management using agents and incorporating knowledge flow for continuously keeping the knowledge updated and current, will ensure smooth knowledge capture, assimilation and reuse. Hence, organizations can save enormous costs, time consuming repetition of mistakes, make informed decisions, and hence gain competitive business advantage and lead. Actual implementation with test beds and results and observations thereafter is an extension of this proposed work

REFERENCES

- Alho.K and Sulonen. R, " Supporting virtual software Projects on the web", *Enabling Technologies: Infrastructure for Collaborative Enterprises, 1998. (WET ICE apos;98) Proceedings.*, Seventh IEEE International Workshops on Volume, Issue, 17-19 Jun 1998 Page(s):10 – 14
- Damian.D and Moitra.D, "Global Software Development :How far Have We come? ". *IEEE Software*, Sept/Oct 2006. pp17-19.
- Desouza.K.C., Awazu.Y, Baloh.P "Managing knowledge in Global Software Development Efforts : Issues and Practices", *IEEE Software* Sept/Oct 2006. pp30-37.
- Desouza.K.C. and Evaristo.J.R. , " Managing knowledge in Distributed Projects", *Communications of the ACM* April 2004/ Vol.47,No.4. pp87-9
- Hartmut Vogler and Alejandro Buchmann "Using multiple mobile agents for distributed transactions". *Third International Conference of Cooperative Information Systems*, 1998. p. 114
- Michael Berger, Bernhard Bauer and Michael watzke . "Towards an agent based infrastructure for distributed virtual organizations". *WETICE ,Proceedings of the 10th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises* Pages: 354 - 355
- Michael Berger , Makram Bouzid, Mark Buckland, Habin Lee, Nicholas Lhuillier, Dieter Olpp, Jerome Picault and John Shepherdson. " An approach to agent based service composition" *IEEE Transactions on Mobile Computing* Volume 2 , Issue 3 (March 2003) Pages: 197 - 206
- Oslon G.M., Oslon J.S. " Distance Matters" , *Human Computer Interaction*, Vol 15, 2000, pp139- 179.
- Rory O' Connor and John Jenkins . "Using agents for distributed software project management". *Proceedings of 8th International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises*, IEEE Computer Society Press, ISBN 0-7695-0365-9, 1999. pp. 54 - 60
- Venugopal.S, Buyya.R, Rammohanrao. K, " A taxonomy of data grids for distributed data sharing, management, and processing". *ACM Computing Surveys*, Vol.38, March 2006, Article3.
- Zhuge H. "Knowledge flow management for distributed team software development". *Knowledge-Based Systems*, Volume 15, Issue 8, 1 November 2002, Pages 465-471

