CHAPTER 3

BIRD'S EYE VIEW OF THE MODEL

3.1 INTRODUCTION

The context of distributed AI system widens the notion of collaborator in at least two general ways: first, a collaborator's "user" who imparts goals to it and delegates tasks, might be either a human or another collaborator. Second, a collaborator must have been designed with explicit mechanisms for communicating and interacting with other collaborators. Collaborators in a distributed AI environment have to solve problems involving large amount of inconsistent and incomplete data, resulting in excessive consultation, complicating integration and causing synchronization delays due to dependency issues (Durfee 1987b, Lesser 1991, Smith 1981). In addition problems are also highly dynamic in nature, that is, quite often information may change during its evolution. That is, in distributed AI systems, collaborators (or problem solvers) spend most of the time in exchanging information, making and revising decisions, planning their activities, group learning etc. In noncooperative domains, the coordination and cooperation issues are still more complicated, as part of the complexity lies in behavior, and is due to the conflicting nature of goals and plans. This leads to a significantly lowered performance and great difficulty in programming the individual collaborators. In this context it is important to device means of effective coordination in order to reach mutually beneficial agreements (based on the behavior analysis). Individual collaborators should have knowledge about their own data and about the behavior of other nodes and of the system
as a whole, which intern helps in effective coordination and reliable performance.

In this work, the above problems are tackled based on trust and behavioral aspects among the collaborators. A model called Behavior Model is designed to handle behavior aspects of other collaborators. In this model, each collaborator is associated with two trust factors. One of the trust factors indicates the collaborator's confidence (Anandakumar 1997a) on its own information, whereas the other trust factor reflects the behavior (Anandakumar 1998b), which is computed by each collaborator locally about the other collaborators. Then, depending on the environment, the appropriate trust factors are used. The model also supports partially cooperative environment, which is the combination of cooperative and noncooperative environments. The system incorporated with these trust factors is then used to resolve disparity and make decisions regarding exchange of information, partial results or allocation of sub problems in the distributed system.

3.2 COMMUNICATION MECHANISM BETWEEN COLLABORATORS

When there is a large problem to be solved, one of the collaborators will initiate the problem solving process. The process starts with the selection of group members. In a collaborative problem solving environment, choosing group members and appropriate task allocation to selected members is a crucial and important task. One way to find correct and promising collaborators, is to broadcast the task in hand and negotiate (Smith 1988) with all collaborators (or volunteers) who respond. However, achieving good coordination in this scenario, is a time consuming process. In this work, a protocol called Mind Protocol is designed, which avoids broadcasting by initially selecting appropriate collaborators. This
protocol helps in selecting friendly and helpful collaborators even in a noncooperative environment. Mind Protocol is based on intelligence collected locally by every collaborator about other collaborators they have worked for or interacted with over a period of time. This information is ordered in a relationship graph, which is constructed by time series prediction of behavior values, and is used by Mind Protocol for selecting appropriate set of collaborators. The announcer (collaborator with a problem) of the task communicates with only the most promising collaborators and selects a group of trustworthy collaborators even when working in a noncooperative environment.

The collaborator in a group sees only a small part of the problem environment, that is, it has only a partial and incomplete view of the problem and it also encounters possibly inexact knowledge domain (Hogg 1991). Problems requiring group collaboration have to deal with large amount of imperfect, garbled and dynamically changing data. Moreover the large number of widely distributed closely interconnected sub problems and partial results require effective and frequent interaction among the collaborators. Hence, to facilitate this information gathering or exchange, each collaborator uses a specially designed software agent called Catalyst Agent (Anandakumar 1998c). The Catalyst Agent physically moves to the destination site and does the job on behalf of the source collaborator. The negotiation (if necessary) between the collaborator is also carried out through the Catalyst Agent.

3.3 DECOMPOSITION AND LEARNING

Once the Mind Protocol selects the promising members of the group (Anandakumar 1999c), the collaborator who initiates the problem solving process starts the decomposition and task allocation process (via negotiation).
Using the relationship graph, the appropriate members who can respond to the request for cooperation have been already selected by the Mind Protocol. Now the task in hand has to be decomposed and allocated. In order to decompose the task, background knowledge can be used because, the incorporation of background knowledge can improve efficiency by narrowing the focus of decomposition or allocation (Yagnanarayana 1998, Frawley 1991). The theta(decomposition) (Anandakumar 1998d) method, which uses background knowledge for task decomposition is used in this work. The background knowledge about other collaborator's past behavior, experience, skills etc., is accumulated by each collaborator over a period of time. This background knowledge has to be updated periodically to cope up with the dynamically changing environment. Instead of just updating the knowledge the model proposed here uses learning techniques (Anandakumar 1997b, 1998a) both for self-evaluation and incremental development so as to reflect the generalization of the background knowledge.

3.4 INTERNAL ARCHITECTURE OF EACH COLLABORATOR

The term agent is becoming popular both in distributed AI system and Intranet & Internet Web (Falchuk 1998, Manindar 1997, Hendler 1996, Riecken 1994). Agents are computer programs that simulate a human relationship by doing something that another person could otherwise do for you. The traditional definition of agents is that, they are autonomous software entities that can navigate heterogeneous computing environment and can, either alone or working with other agents, achieve some goal. Thus, they require on-board intelligent tools to achieve their task (such as planning, reasoning, and learning) (O’Leary 1997). The software agents are becoming an essential part of complex software applications because they migrate complexity. The agent-based approach to information management is both scaleable and cost effective. In order to reduce internal workload of the
collaborator (intelligent node), and to improve the quality of the development process local agents (software agent) are used. Agent-based technology (Maes 1997) also provides a high degree of decentralization of capabilities, which is the key to system capability and extensibility. In this work, each collaborator has nine functional local agents. Here, the term agent stands for local agents, and the term collaborator stand for intelligent node. Special-purpose local agents such as:

1. Agent = Control and Analysis Agent (CAA),
2. Agent = Data Analysis Agent (DAA),
3. Agent = Trust Computing Agent (TCA),
4. Agent = Similarity Analysis Agent (SAA),
5. Agent = Behavior Analysis Agent (BAA),
6. Agent = Incremental Learning Agent (ILA),
7. Agent = Skill Managing Agent (SMA),
8. Agent = Mind Protocol Agent (MPA), and
9. Agent = Decomposing Agent (DA)

helps each collaborator, and allow for easy and dynamic reconfiguration of system capabilities. These agents are organized hierarchically (3 layers) as in Figure 3.1. The functionality of each of the local agents of a collaborator is described.

**Control and Analysis Agent (CAA):** CAA initializes the working environment of its own collaborator, defines the role of each local agent, issues appropriate instructions when needed, and monitors the activities. It also keeps track of local agent's interactions and development progress.
Figure 3.1 Organization of local functional agents within a Collaborator
Data Analysis Agent (DAA): It analyses the local database(s) attached to the collaborator and produces the summary reports of the data. It also produces Auto-Biographical profile about itself, and Biographical profiles about the other collaborators with the available information in local database(s). Depending on the situation, it decides whether to analyze the entire database or not. That is, the sample size would be entire database or partial database. The sample size for other collaborator (when it is requested) is set depending on the relationship with that collaborator.

Trust Computing Agent (TCA): The summary reports of DAA are received by TCA, which computes the Local Trust Factor (LTF). It reflects the self-confidence of that Collaborator on it's own data. This local trust factor is periodically announced or sent on request. The TCA also stores the LTFs received from other collaborators for future usage. Using the statistical inference, it studies the LTF of each collaborator, whether it is growing, constant or decreasing.

Similarity Analysis Agent (SAA): It performs similarity analysis of locally available information with all or some profiles or partial results or sub goals received from the co-collaborators during the problem solving process. It classifies the co-collaborators into “Similar” or “Partially Similar” or “Dissimilar” category. Using the statistical inference, it sets the percentage of minimum match needed from the particular collaborator.

Behavior Analysis Agent (BAA): The SAA outputs and SMA outputs are used by the BAA to compute the Behavior Trust Factor (BTF) of each co-collaborator. After the problem is solved, the behavior of each collaborator is computed. Using prediction, it predicts the future behavior of the collaborators.
Skill Managing Agent (SMA): It creates catalyst agent(s) as per the instruction from CAA. Each collaborator is attached with skill-base, in which the collaborators store their own skills (methods or language). The skills are selected depending on their needs. Skill-base is monitored by this agent and the agent also replicates the catalyst agent as instructed by the CAA.

Mind Protocol Agent (MPA): It works on intelligence collected locally by every collaborator about other collaborators they have worked or interacted over a period of time. This information is ordered in a relationship graph. This graph is constructed by time series prediction of behavior values (BTFs), and is used by Mind Protocol for selecting appropriate set of collaborators for the problem at hand. It communicates with only the most promising collaborators and selects a group of trustworthy collaborators even while working in a noncooperative environment.

Decomposing Agent (DA): Once the group is formed, the DA divides the problem based on the background knowledge or domain knowledge. To decompose the problem, the agent uses the other collaborator's biographical information (prepared by DAA). If it is not available, the collaborator uses supplied information of that collaborator. Then it passes all the sub modules to MPA for distribution and allocation.

Incremental Learning Agent (ILA): When the problem is solved, it receives statistical information from all the local agents and produces impressions (summaries) of the collaborators who participated in solving the single global problem. These impressions acts as a background knowledge for the next problem.

The scenario of interaction between the different functional agents are shown in the diagram (see Figure 3.2).
3.5 ASSUMPTIONS ABOUT THE ENVIRONMENT

The assumptions made in this work are:

i) **Collaborators are heterogeneous in nature**: The work group may be homogeneous work group or heterogeneous work group. In homogeneous work group, every one in the group fulfills the same role, that is, all members are assumed to have the same level of training and skills. Heterogeneous work group, involve many different roles and job descriptions, each member has its own knowledge, reasoning, plans, results etc.
ii) **The domain is cooperative or noncooperative or partially cooperative:** in cooperative environment, the collaborators trust each other, and information obtained (supplied) from (to) them are considered as it is. But, in noncooperative environment, one cannot trust the information received from other co-collaborators. In such an environment, the collaborator uses past experience with that co-collaborator, to judge the co-collaborator's decision or information. The combination of the above two environments is called as partially cooperative in which equal priority is given to the past experience and the information given by the co-collaborator.

iii) **Each collaborator is willing to negotiate about disparities and revise any knowledge:** since the group is a heterogeneous group, disparities may arise while solving the problem. In such a scenario, through negotiation (if necessary) collaborators are ready to resolve their disparities, and update or revise their knowledge.

iv) **Each collaborator is attached with local databases:** for example, the application considered in this thesis, each collaborator is attached with project database, skill base and knowledge base. The project database contains all the details of previously solved projects, skills used, with whom it is solved, etc., skill base, contains all the skill (technical or managerial) known to the collaborator etc. In order to give equal probability to the information stored in the local databases, it is assumed that, the data stored in the databases is uniformly distributed. The gained knowledge is stored in the knowledge base.
v) **Two-way binding:** They have a belief, based on the principle "believe me to believe you", that is, a kind of two-way binding exists between the collaborators.

### 3.6 APPLICATIONS CONSIDERED IN THIS WORK

Today's Internet technology provides a powerful and cost-effective means of overcoming the obstacles like closed architecture, platform dependent Client/Server technology, fixed database drivers etc. It allows distributed network, global access, platform independence, information sharing and inter-mobilization. The Web is becoming a new media for software development and distribution (Oreizy 1997). Fielding and Kaiser (1997), discuss Apache HTTP Server. Here, a group of Webmasters knows as volunteers gathered together via the Internet. Fujistu has taken the first steps in constructing an enterprise-wide, Internet-based infrastructure that allows teams around the world to collaborate on every phase of the life cycle of a global software product (Jerry 1999).

Hence, in this work, to bring out the salient features of the proposed model, Global Software Development is considered. The problem is, a software is to be developed for a company (Book Shop), which is establishing a new catalog (Book catalogue) sales division to sell its product (Book), and outdoor merchandise. The catalogue will be published on the Web, and orders can be placed by e-mail, via the Web site, or via telephone, or by fax.

In a Web oriented collaborative problem solving environment, the problems faced by the collaborative group are, (i) Choosing correct group members, (ii) Problem decomposition, (iii) Appropriate task allocation to selected members, (iv) need of good interactive communication mechanism, (v) effective leaning method(s), etc. The Web security (Sen 1997, Atkinson 1997, Biber 1997) has been assumed, that is, Confidentiality and Integrity is
maintained; confidentiality refers to the protection of information while it en route to its destination, and integrity refers to the exact preservation of the transmitted data so that it reaches its destination without any alteration, accidental or deliberate. Hence, the major steps considered in the above application are,

Step1: Effective group formation-in order to form a most promising group, a high-level intelligent protocol called Mind Protocol is designed in the work. This protocol is based on the Behavior Model, which imitate the collaborator's behavior.

Step2: Decomposition, a new decomposition method called theta-decomposition is used to divide the task according to their capabilities (personal parameters). It utilizes the background knowledge available locally in the local database, while decomposing the task. This helps in improving group performance.

Step3: Negotiation approach for task allocation- in order to allocate sub tasks to the member's taste and interest, the Mind Protocol performs, negotiation with that collaborator, before assigning the sub tasks. This is to ensure, the assigned task is its interest, which intern improves the overall performance of the group. The Mind Protocol, uses special kind agents, the catalyst agents for negotiation. That is, here the catalyst agents acts as a negotiating agent.

Step 4: Effective communication mechanism- here, the work introduces a special kind of software agents called catalyst agents. In distributed AI system, the collaborators quite often exchange their partial results (or sub goal) with its co-collaborators in a Functionally Accurate/Cooperative distributed system (FA/C) manner. Here, the catalyst agent acts as a information gathering/exchanging agent.
To gain more knowledge, and to have more understanding capability, learning is very important. Hence in this work, each collaborator uses three different learning methods. The relationship graph constructed by the Mind Protocol is treated as a probabilistic network, and the collaborator uses probabilistic reasoning to find relationship between unknown collaborators, Action-Research (learning from experience) learning technique to enhance knowledge about its co-collaborators, and finally, the collaborator may use rule-induction method to derive new relationship, behavior etc.

Further, to highlight the features of Behavior Model, Web based publishing application is used (Anandakumar 1999b). When Web publishing (Pefaffebber 1997, Title 1996, Larry 1995) is used for product promotion, initially vendors prepare an advertisement of the company product and make it available on the Web. Customers have now access to a vast amount of product related advertisements (or ads) (David 1996, Nelson 1994), which may however sometimes contain misleading information. The customer thus needs a yardstick to measure the product ads. In such situation, the vendor's behavior plays an important role, and the consumer will be divulged by a number of ads on the Web. This ambiguity is removed by using Behavior Model in selection criteria.

Behavior Model can be used to resolve disparities among the group members, especially under noncooperative domains. It is demonstrated in (Anandakumar 1999a). In group activities, that too under noncooperative domains, the collaborator's (member) behavior plays an important role. The Behavior Model helps in capturing intelligence (behavior knowledge) in the form of trust factors. The model is tested by replacing the truth value (fourth parameter) of the epistemic system.
Characteristics of the Catalyst agents are brought out using two more applications. In both the application, the catalyst agent acts as a communication agent. (i) Catalyst agents in reliability checking (Ehrtich 1990, Anandakumar 1999c) - since in global software development, the modules are developed by different teams at different places, the reliability checking process become much more complicated. In order to maintain uniformity and to control reliability errors in early stages of the development cycle, the process is automated through the catalyst agents. The team leader, with the help of the catalyst agent, computes the group reliability, and internal consistency, which helps in monitoring overall reliability of the software product. (ii) Catalyst agents in software review process - here, the review process (O'Hare 1992) is automated, in which, the catalyst agents are used by the review team members. The catalyst agent exchange review forms, to construct single integrated reports.

3.7 CONCLUSION

This chapter highlights the importance and necessity of some of the basic issues, which have been tackled in the work described here. The chapter gives a complete overview of the work described in this thesis. It also mentions the basic assumptions which form the basis of this work and also describes the applications which have been considered to highlight the important features of the work. The Behavior Model has been discussed in detail in subsequent chapters.