Chapter 2
Agent Based Systems

2.1 Introduction

The inspiration and motivation for this research work are covered in the previous chapter. This chapter talks about the technologies that will be used in handling the research problem. Basic concepts of agents and their various definitions along with characteristics and detailed classification of agents are given. An action an agent takes is influenced by the environment it is in. There are different kinds of environments and all of them are covered here. The attributes that constitute an intelligent agent are mentioned. Once agent concepts are established multi-agent systems are explained in detail. What a multi-agent system is and the different concepts in multi-agent systems are given. An in depth understanding of agents and multi-agent systems capabilities lays the foundation for this research work.

This chapter further talks about the framework that is used in this research and supporting technologies that are used to make the system more efficient. The concluding part of this chapter will be on literature survey. This section covers the history and evolution of multi-agent systems and their roots in various domains. Besides the conceptual information and history about the agents, a survey of present day available solutions to the research problem is given. Limitations of existing solutions lay the justification for this research work.

The chapter is organized as follows. Section 2.2 of this chapter covers Agent technology in detailed. Multi-agent systems and its concepts are presented in Section 2.3. In Section 2.4 technologies that make the framework for this research are enlisted. Section 2.5 talks about the literature survey. Section 2.6 concludes the chapter.
2.2 Agents

An agent constantly perceives the environment it is in through sensors and acts rationally using effectors on that environment towards achieving the goal as shown in Figure 2.1. Agents operate in an environment and it defines the properties of the world. Agents can be defined to be autonomous computational and goal oriented entities. They are capable of effective operation in dynamic and also open environments [11, 100, 26, 103 and 77].

![Agent Diagram]

*Figure 2.1: An Agent Interacting with an Environment*

Agent’s actions at any time are dependent on:
- Prior knowledge about the agent and the environment;
- Interaction with the environment through observations and past experiences;
- Goals that they must achieve; and
- Actions that an agent is capable of carrying out.

Two agents under the same circumstances of prior knowledge, history, goals and abilities should act in the same way. Changing one of these parameters can result in different actions. Each agent has some internal state where it can hold information about itself and
the environment. The information can be the goals it has to achieve, the actions it is going to take to achieve those goals, and its means to reason, perceive and learn.

Agents are very well suited to code and automate modular type programs. It offers a new platform to conceive, design and implement software systems. Agent technology offers a high level of abstraction and is suitable for software development where other methods have shortcomings [51]. Artificial Intelligence researchers consider an agent to be a computer system that is conceptualized or implemented using characteristics that are more usually applied to humans. In AI it is quite common to characterize an agent using notions, such as knowledge, belief, intention, and obligation [77].

Individuals with Object Oriented (OO) background usually confuse agents as an extension to OO concepts. Wooldridge and Jennings [81] explain the difference between agents and objects. Object oriented approach brought segmentation in code by introduction of methods. This allowed defining and controlling of local variables. An important feature of OO is encapsulation of state and this is achieved through methods. But still objects are considered passive because the methods can only be invoked through an external entity. If an object x invoked a method m on an object y then y has no control and authority on its actions. That is why objects are not autonomous. Agents have their own thread of control. Not just the code and state but also its initiation is localized. Each agent has its own rules and goals. They hold the decision power on how and when to act. Agent based way of thinking brings a different perspective to system development.

When the domains are of reasonable complexity, an agent will not have total control over its environment. Thus, even when the same action is performed by two different agents under identical situations might have completely different outcomes. Along with different outcomes failure is also a possibility. Sometimes the outcome of an agent may not produce the desired effect at all. They also learn and use knowledge to complete tasks. They can be very simple or complex.

2.2.1 Definitions of an Agent

There is no universal consensus definition of what an agent is. This is due to diversity in the research fields and the examples they encountered. Some of the popular definitions are given below:
SodaBot definition of an agent:

"Software agents are programs that engage in dialogs [and] negotiate and coordinate transfer of information." [67]

SodaBot is an environment for creating and using distributed software agents managed by Michael Coen at MIT. According to his definition agents communicate with one another when they are negotiating which involves sensing and acting on the environment.

Pattie Maes definition of an agent:

"Autonomous agents are computational systems that inhabit some complex dynamic environment, sense and act autonomously in this environment, and by doing so realize a set of goals or tasks for which they are designed." [89]

Pattie Maes, is one of the pioneers of agent research comes from MIT's Media Lab. According to her agents must act autonomously to meet their goals. Since the environment is explicitly defined as complex and dynamic agents operate in a constantly changing environment and need to adapt to changes.

Hayes-Roth definition of an agent:

“Intelligent agents continuously perform three functions: perception of dynamic conditions in the environment; action to affect conditions in the environment; and reasoning to interpret perceptions, solve problems, draw inferences, and determine actions." [25]

Barbara Hayes-Roth of Stanford’s Knowledge Systems Laboratory focuses on the reflexive nature of agents. If reasoning is interpreted then reflex actions as well as planned actions will be hindered.

Michael Wooldridge and Jennings definition of an agent:

“An agent is a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its delegated objectives”. [76]
According to Michael and Jennings agents act independently on behalf of users. Communication is a major requirement to solve a problem and meet the desired goals without human intervention.

There are many more definitions of agents they can be found in the work done by Honavar [56]. From all the above definitions, it can be concluded that agents are autonomous and communicate with other agents or users in some environment. From these definitions important characteristics of an agent are presented in the next section.

2.2.2 Characteristics of Agents

As can be seen from the previous section there are various definitions of agents. From these definitions characteristics of agents can be inferred. These characteristics help to differentiate between an agent and a regular software program.

The main characteristics of agents are listed below in Table 2.1:

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autonomy</td>
<td>An agent acts on its own to control its actions and internal state without any direct human intervention.</td>
</tr>
<tr>
<td>Proactive</td>
<td>They are goal oriented and can accomplish their goals by taking initiative in order to satisfy their design objectives.</td>
</tr>
<tr>
<td>Social ability</td>
<td>By using agent communication languages agents are capable of communicating with other agents or humans.</td>
</tr>
<tr>
<td>Reactive</td>
<td>Agents are very good at adapting to the changes in the environment. They constantly perceive the environment and take actions in a timely fashion when needed.</td>
</tr>
</tbody>
</table>

Table 2.1: Main Characteristics of Agents
In addition to the main characteristics mentioned above an agent can also exhibit the following additional characteristics (Table 2.2):

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Persistence</td>
<td>Agents have their goals and decide how to accomplish them. They continuously keep running until their goals are met.</td>
</tr>
<tr>
<td>Reasoning</td>
<td>Since agents constantly perceive its environment they also have to reason before deciding on an outcome.</td>
</tr>
<tr>
<td>Productivity</td>
<td>Agents are goal oriented and capable of achieving the desired result.</td>
</tr>
<tr>
<td>Mobility</td>
<td>Agents have the capability to move from one platform to another. It can be another machine or across different architectures taking advantage of the Internet.</td>
</tr>
<tr>
<td>Personality</td>
<td>Qualities of human nature can be manifested in agents.</td>
</tr>
<tr>
<td>Rational</td>
<td>Based on the percepts it gets and with the background knowledge it has, an agent selects a particular action that takes it closer to its goal.</td>
</tr>
</tbody>
</table>

Table 2.2: Additional Characteristics of Agents

2.2.3 Classification of Agents

Field areas of research for agents are unlimited and practical applications in various domains are many fold. Each research or application needs are different and calls for different kinds of agents. There are various kinds of agents depending on the roles they play and their capabilities. Some of the popular agent types are covered here below.

2.2.3.1 Collaborative Agents

As the name implies these agents’ emphasize autonomy as well as communication and cooperation with other agents. They are responsible for negotiating with their peers to reach a mutually acceptable agreement during cooperative problem solving. They are more deliberative agents with very limited learning capabilities. They are used to solve problems
that are too large for a single agent to handle or to carry out functions that are beyond the capabilities of any of the agents.

2.2.3.2 Interface Agents

Interface agents emphasize autonomy and learning in order to perform tasks of their users. The idea behind interface agents is to make computer systems more like proactive agents. So these agents take the initiative rather than wait for the user to issue commands. Thus they are cooperating with the user to achieve the task rather than just act as servants. The way in which these agents help users can be to anticipate user needs, make suggestions or providing advice to them. These secretary agents have limited cooperation, reasoning or planning capabilities. Maes [90] defines interface agents as “Computer programs that employ AI techniques in order to provide assistance to a user dealing with a particular application.”

2.2.3.3 Mobile Agents

Mobile agent is a software program that transports data and state from one computer to another autonomously and recommencing execution at the remote site. Mobile agents have the features of autonomy, social ability, learning, and most importantly, mobility and they have the ability to decide when and where to move. The idea of mobile agents is to replace remote procedure calls as a way for process to communicate over network.

2.2.3.4 Internet / Information agents

Information agents have access to one or more information sources and are able to collate and manipulate information obtained from these sources. Their main tasks are proactive acquisition, mediation and maintenance of relevant information for users. The information sources can be of any type like tradition databases, internet, etc. With the explosive growth of data on internet it is impossible to manage information manually and these agents come in very handy. Users accessing data through these agents can save time and effort need to access and analyze data and in turn improves productivity.
2.2.3.5 Reactive agents

Reactive agents are simply reacting to the environment, without reasoning about it. They do not maintain any internal state. Reactive agents are able to reach their goal only by reacting reflexively on external stimuli. Even though reactive agents offer many advantages like simplicity, robustness against failure and elegance there are some drawbacks too. Since these agents make decisions based on agents current state they don’t take into account non-local information.

2.2.3.6 Hybrid agents

Reactive agents and proactive agents each have their own advantages and so they are the ideal choice in certain fields. When there is a requirement that an agent be able to handle both these kinds of behaviors an obvious choice is to create separate subsystems. The goal of these agents is to balance the qualities of reactivity and proactiveness. Hybrid systems integrate two or more different kinds of agents.

2.2.4 Agent environments

As mentioned above the main components of an agent are the percepts, actions and goals. All this is within an environment. The most important decision an agent has to make is which action to take to best satisfy its design objectives. The complexity of agent’s decision making process is directly dependent on the properties of agent’s environments. According to Russell and Norvig [103] agent environments can be classified as:

- **Accessible vs. inaccessible**
  When agent’s sensors give complete state of the environment at each point of time then that environment is fully observable or accessible. When the environment is fully accessible by agents then they need not maintain any internal state to keep track of the changes. Agents can then function optimally as it has clear data on all aspects that are relevant to the course of action. Some environments are only partially accessible either because of noise or inaccurate sensors or because parts of the state are missing from the sensor data. It is commonly observed that most moderately complex environments are inaccessible.
• **Deterministic vs. non-deterministic**
  If the conditions of the environment are determined by the current state and the actions an agent takes, then the environment is deterministic. An agent need not worry about uncertainty in a fully accessible, deterministic environment. But rarely it is the case and partial accessibility of state information to agents can lead to non-deterministic environments. This uncertainty presents a greater challenge to agent designer as they have to keep track of all the unobserved aspects.

• **Episodic vs. non-episodic**
  In an episodic environment, agent’s actions are divided into atomic episodes where in each episode consists of the agent perceiving and carrying out a single action. Agent’s actions in next episode are totally disconnected from the decision’s it made in the current episode. When short-term actions lead to long-term consequences then it leads to non-episodic environment. Playing chess and taxi driving are sequential environments where current decision could impact future decisions. Episodic environments are much easier because the agent need not have to think ahead.

• **Static vs. dynamic**
  Static environments don’t change with the passage of time. If the environment can change while the agent is still deliberating then it is dynamic. It is obvious that dynamic environments are much more complex to design.

• **Discrete vs. continuous**
  An environment can be said to be discrete if there are only a measurable number of actions and percepts. A chess game is a good example of a discrete environment. Driving a taxi is an example of a continuous environment as it is always in a continuous state with changing speed, location and other vehicles on the road.

2.2.5 **Attributes of Intelligent Agents**

Agents incorporate specific characteristics like behavioral and adaptation skills which when exploited in the right manner clearly show human like intelligence. A simple definition of an intelligent agent is a software program with the goal of assisting users [61, 51]. This is a very broad definition and not of practical use. Many researchers tried to define what an intelligent agent is, but there is no general consensus on how to define intelligent agents in
a more formal way. In the article [81] authors Jennings, Sycara and Wooldridge brought out the fact that the interest on intelligent agents did not emerge from a vacuum. There is no formal definition but it is an agreed upon fact that autonomy and the capability to act without humans or other systems intervention is a key feature of an intelligent agent. The kinds of capabilities that are found in an intelligent agent are reactivity, proactiveness and social ability. Importance of different attributes varies based on the domain of the agent. Intelligent agent’s theory is closely related to many different disciplines, researchers and developers from those fields have been studying about them from a long time. Main areas that lead to formalization of “intelligent agent’s theory” are:

- artificial intelligence [103];
- object-oriented programming [46] and concurrent object-based systems [44, 45];
- human-computer interface design [72].

In articles written by [84] and [72] authors have shown how Intelligent information agents can be used as “personal information agents”, “Tour guides”, “indexing agents”, “FAQ-finders” and “Expertise finders”. Interest in intelligent agents grew exponentially with the advancements in Internet. This is one area where most of the existing research is done with internet at focus. There are many other practical areas where intelligent agents have been used. Examples of these areas are telecommunications network management, air traffic control, business process re-engineering, data mining, information retrieval/management, electronic commerce [49, 50] and power management [41].

2.3 Multi-agent Systems

Multi-agent systems are designed and developed as several interacting agents. The primary goal of these systems is to come together to solve problems which cannot be solved by individual agents. Due to interactive and autonomous nature they are best suited where little or no interaction is required to maintain an application. They provide good solutions which have multiple problem solving methods, various perspectives and many problem solving entities. Multi-agent systems are typically distributed systems consisting of several distinct components. Each component is an independent problem-solving agent coming together to solve a problem [68]. A multi-agent system consists of a number of
agents which cooperate and communicate with each other in a distributed environment. This communication can be cooperative in that agents involved can contribute towards a common goal or selfish in which agents pursue their individual goals. Multi-agent systems are best suited to solve complex, real world problems. To build a robust and scalable software system requires autonomous agents that have the capability to achieve their objectives while situated in a dynamic and uncertain environment [96].

Important characteristics of multi-agent system are:

- Agents are autonomous;
- Each agent has a limited viewpoint so no single agent has a full view of the system;
- Data is decentralized and there is no global system control. It works as a monolithic system; and
- Computation is asynchronous.

Agents are simple and handle small pieces of the puzzle without much global view but multi-agent systems can materialize self-organization and steer the process in handling complex behaviors. Just like humans have some rules, conventions and common languages to communicate agents need an agreed language to share knowledge. A set of agents without rules that regulate their communication will be a chaotic group of entities just producing noise. ACL (Agent Communication Language) provides a standard format that implements the agent communication model. ACLs act as wrapper languages in that they communicate through a knowledge-level protocol that is unaware of content language and ontology used. The expressive power of ACL languages adds an additional advantage where sophisticated patterns of interaction are needed [81].

The most popular ACLs are:

- FIPA-ACL (by the Foundation for Intelligent Physical Agents) [43]
- KQML (Knowledge Query and Manipulation Language) [120]

The number of multi-agent based applications being developed and deployed in real world settings is rapidly increasing. They have been successfully implemented in air traffic control, manufacturing, telecommunication systems, process control, traffic and transportation management, information filtering and gathering, electronic commerce, business process management, entertainment and medical care. No matter which field area they are
implemented in they take up different roles as personal agents, information filtering agents, e-mail agents, scheduling agents and much more. Practical applications of multi-agent systems have been documented in a number of papers [70, 62 and 33] and [71, 101 and 52]. Some of the fundamental concepts of multi-agent systems are given below.

2.3.1 Cooperation in Multi-agent Systems

Multi-agents have many agents working together to solve complex problems. Cooperation is very crucial while they are working towards achieving their goals. Cooperation can be either communicative or non communicative. In communicative type of cooperation agents actually send and receive signals or messages. In non communicative type they observe the activity of other agents and react accordingly. The type of cooperative methods used in multi-agent systems also depends on the type of agents involved. Reflex or reactive agents cooperate without much reflection on the possible actions. This kind of cooperation can be called “emergent” as there is no prediction or predictive planning. Deliberative agents look at all possible combinations of actions before they perform. These agents use some means to choose which action to take.

Some of the conditions that must be met to achieve cooperation between agents are given by P. M. Jones & Jacobs [88]. The conditions are listed below:

- The existence of two or more reasoning agents;
- Actions can be mutually perceived only when they exist in the same environment;
- Agents must be working together and contributing by some productive elements; and
- Availability of local goals.

2.3.2 Competitive Multi-agent Systems

Depending on the type of application and agents function competitive behavior is also equally important along with cooperation. In this each agents goal is to maximize their own actions while trying to negotiate with other agents that have conflicting interests. Auction agents exhibit such competitive behavior. Buying and selling agents involve in the process of negotiating in order to resolve differences. Each agent acts on behalf of users and may not act for the benefit of the entire system. By implementing rules and procedures, agents
with conflicting goals can cooperate in a competitive environment and be at least social for the benefit of the whole system.

2.3.3 Negotiation in Multi-agent Systems

Negotiation in multi-agent systems is needed when two or more agents are involved in decision making process with the goal of reaching a consensus. The involved agents have their personal and possibly conflicting goals. However, these agents may participate in negotiation process as they share a common goal. Negotiate is crucial to maintain inter-agent dependencies. All involved agents submit a proposal at each round of the negotiation process. Negotiation process involves the following steps:

- Interaction is governed by a set of rules called negotiation protocols;
- The ranges of issues over which agreement must be reached are covered under negotiation objects. Agents might be involved in negotiation process to solve a single issue like price or multiple issues like price, quality and time, etc; and
- Decision making process is aided by agents reasoning models. Negotiation protocol and objects involved determine sophistication of the models.

2.3.4 Homogeneous Multi-agent Systems

When individual agents come together with the same goal and apply the same strategies it is called a homogeneous multi-agent system. Besides having identical components, rules of behavior utilized by such systems are often simple. Communication between agents in the homogeneous system can be direct and can be achieved without the use of a communication protocol. The disadvantage is that it cannot be employed to solve problems that involve agents with different goals.

2.3.5 Heterogeneous Multi-agent Systems

When individual agents with different goals and different domain knowledge are involved in a multi-agent system, the system is considered as a heterogeneous multi-agent system. Components of the system are non-identical. They are simple and follow simple rules of behavior. The scope of the problems they tackle is determined by the different types of agents involved. Since different kinds of agents are involved, heterogeneous multi-agent
systems are more suited to solving complex problems. Further such systems are more suitable to multi-disciplinary domains.

2.4 Frameworks/Technologies for Multi-agent System Implementation

In the previous two sections concepts of agents and how multiple agents can work together in a multi-agent system to solve a complex problem are described in detail. There are lots of components involved in building a working multi-agent system. This section introduces different frameworks and describes them in brief by giving their characteristics and applications.

Usage of readily available established framework makes the development process easy. JADE is the framework that is chosen to develop the multi-agent system as part of this research work. Along with third party tools/technologies used in agents to aid information gathering and dissemination two other promising technologies Ontologies and fuzzy logic are also used.

2.4.1 JADE

JADE (Java Agent Development Framework) [62] is a software framework that is compliant with FIPA specification and is fully implemented in Java language. JADE is a middleware that aims at supporting the development of multi-agent systems and facilitates communication between them. It has a set of graphical tools that make debugging and deployments lot easier. The nuances of starting agents and their life cycle activities are handled by JADE so developers can concentrate on solving complex problems. JADE is easily integrated into eclipse which is a platform for developing Web applications. All agents in JADE are implemented as Java threads which live inside a container which is the Java process that launches them.
As shown in Figure 2.2 during runtime JADE will have multiple containers and one of them will be the main container and all other agents must register with this main container. If there is more than one main container then each one will work as its own platform. Main container is responsible for maintaining the list of all the containers, a table to hold all the registered agents and managing life cycle of agents which includes creating as well as destroying them. Sending and receiving messages in JADE is done through the standard FIPA ACL specification. Communication is transparent to users so programmers need not worry about the intricate details of messaging. JADE run-time environment handles the communication and also it provides a number of Graphical User Interface (GUI) based tools to enable a user to monitor and control run time system.
2.4.2 Ontology

Diversity and size of information on Web keep growing. Gathering relevant information on the Web has become a very complex and daunting task. The growing availability of information demands intelligent ways to ease information access. The main problem with the existing information retrieval approaches is that they neglect the context of the pages. Most retrieval systems are keyword based. One way to handle this issue is to focus not just on the lexical information features but also to consider its semantics, that is the meaning attached to it. Taking context into account leads to more relevant information gathering.

Information on Web is free flowing with a very high degree of irregularities. Complete understanding of the content on the web is infeasible for the foreseeable future. The sheer amount of information on Web aggravates the problem. This has lead to shallow text processing approaches which means generic language regularities which are known to cause complexity problems are ignored or not handled properly. The use of Ontologies to organize the knowledge and to express semantic meaning has been gaining attention.

The term ‘ontology’ is derived from the Greek words “onto” and "logia". "onto" means being, and “logia” means written or spoken discourse. It has also reference in philosophy which means the study of the nature of being. There is a lot of debate on the exact definition of ontology but the most common definitions state that ontology is a specification of a conceptualization [108] or that ontology is the shared understanding of some domain of interest [75]. Ontology consists of statements that define concepts, relationships, and constraints. It is similar to a database schema or an object-oriented class diagram. Ontology forms an information domain model. Prior to initiation of any software development process a data model for database is specified in the same way a data model for a multi-agent system must also be specified. Agents communicate with each other by passing messages which are nothing but a bunch of parameters. If the parameters are not well defined in the domain model, then the communication will not happen properly and information will be lost or will not be delivered. Ontologies help to describe the concepts and relationships used to interact in the domain [31].

When Tim Berners-Lee invented the Web, its aim was to facilitate communication. The initial goal was to get the information bridge between humans but now there is also a need
to allow the participation of machines. The term ‘Semantic Web’ was first fabricated in paper written in 2001. He stated in the paper that information on the Internet is designed and presented more for human consumption. With well-designed and carefully structured data computers can better understand them. Without the context information even when the information is retrieved from database with well-defined meanings the implications of the data are not easily evident [106]. The Semantic Web relies heavily on formal structure of data and Ontologies play a key role. Ontology consists of statements that define concepts, relationships, and constraints. It corresponds to a database schema or an object-oriented class diagram.

There are special requirements for interoperability in knowledge based systems. Statements in a formal knowledge representation help in operation and communication of such systems. These kinds of sharing activities demand that human and/or machine agents agree on common and explicit Ontologies. Negotiation and exchange of knowledge becomes easier with the use of standardized Ontologies. When developing Ontologies to share knowledge across different communities or domains they should have the characteristics of extensibility, visibility and be inferable.

The use of Ontologies brings many advantages [109]. Communication between agents becomes easier in a domain when concepts are defined in Ontologies. There is an increased flexibility with the use of Ontologies in the development of information gathering systems on the Web. Domain knowledge can be defined with suitable granularity representing the subtle differences in hierarchy between the entities. Ontologies are also often used to build information models. These Ontologies capture the terms of the items, their meaning and properties, association between the items and their definitions. Reusability is another added advantage of using Ontologies. Before creating a new ontology users must first check to see if any existing Ontologies match their domain specification needs. The benefit of using existing Ontologies is that systems become more interoperable and sharing data between systems become lot easier.

2.4.3 Fuzzy Logic

Many domains of practical interest have subjective and inherently vague knowledge which cannot be expressed using the crisp logical structures. This puts a hurdle in exploitation of
knowledge from such domains. Data retrieved from Web is customized by application of personalization and filtering methods based on information collected from user’s profile. Information collected from user’s profile pages is collected in language that is understood by humans and there is a certain degree of uncertainty and vagueness. Agents cannot guarantee good results if they have to converse, reason and make rational decisions in an environment of imprecision.

The ability to incorporate computation with words provides solution for uncertainty and vagueness problems. Fuzzy logic can be used to reasoning that is appropriate rather than exact. This technology promises capacity of implementing machine intelligence. This bridges the gap between human understandable soft logic and machine understandable hard logic. Fuzzy logic helps in quantifying vague or uncertain terms and is very helpful where human like reasoning is required. Fuzzy logic makes human intelligence to be easily coupled with machine processing capabilities.

The term “fuzzy logic” emerged in 1965 with the theory of fuzzy sets by Lotfi Zadeh [65]. Zadeh modified conventional set theory where an individual could have a degree of membership with values ranging over a continuum of values, rather than being either 0 or 1. Zadeh says that fuzzy theory is a process of fuzzification from crisp (discrete) to continuous (fuzzy) form. These terms are known as linguistic variables [65], or fuzzy variables. Along with fuzzy linguistic variables, IF-THEN rules are also defined in the form of fuzzy rules. Vague inputs are converted into crisp outputs using these rules. Linguistic classifications work very well especially for user interface applications. Fuzzy logic has been used in a number of different applications. In a variety of domains where human like reasoning and behavior are required this technique has been used and it returned impressive results [12]. Some examples of the use of fuzzy linguistics in multi-agent systems can be found in [69 and 32].

Fuzzy logic systems are simple and flexible; a fuzzy system can be created to include any set of input-output data. If there are any changes in input or output data or inference rules they can easily added to the system. Direct use of fuzzy logic is very limited. Applications use it as the underlying logic system for fuzzy expert system. A fuzzy expert system is a collection of fuzzy membership functions and rules as illustrated in Figure 2.3.
Below is an example of a simple rule in fuzzy expert system.

\[
\text{if } a \text{ is } f_1 \text{ and } b \text{ is } f_2 \text{ then } c = f_3
\]

Where \( a \) and \( b \) are input variables and \( c \) is the output variable. \( f_1 \) is a membership function defined on \( a \), \( f_2 \) is a membership function on \( b \) and \( f_3 \) is a membership function on \( c \).

A fuzzy system has the following structure:

i. Fuzzification: In fuzzification process actual values of input variables are taken and membership functions are applied to them. This will determine the degree of truth for each rule premise.

ii. Inference: In this step the truth value for each rule is computed. These are then applied to the conclusion part of each rule. The resulting fuzzy subset will be assigned to each output variable for each rule. For inference rules usually only MIN or PRODUCT are used.

iii. Composition: From the previous step a fuzzy subset is generated when each rule is applied on individual output variables. In composition step a single fuzzy subset is formed by combining together all fuzzy subsets from previous step. For this usually MAX or SUM are used.

iv. Defuzzification: This step is optional depending on the applications use. It converts the fuzzy output as crisp number again. Out of a number of defuzzification methods the two that are mostly used are CENTROID and MAXIMUM methods.
2.5 Literature Survey

This section covers the history of multi-agent systems. The links between multi-agent systems and Artificial Intelligence are illustrated and how multi-agent systems differ from object oriented concepts is also talked about. Use of multi-agent systems in real world applications is increasing every day. Examples of real world applications that are implemented successfully in various domains are elucidated. This section also looks into the existing solutions that are available for the research problem taken. The limitations of these systems and a justification for the prototype are presented at the end of this chapter.

2.5.1 A Brief Retrospective

The roots of multi-agent systems are spread into different academic disciplines. Out of the many roots the most obvious one for multi-agent systems lies in the field of Artificial Intelligence (AI). Agents appeared in the earliest AI literature. The idea of an agent originated in mid 1950’s with John McCarthy [10]. The term was coined jointly by John McCarthy and Oliver G. Selfridge [10] a few years later, when they were both at the Massachusetts Institute of Technology. Both of them viewed agents as a "soft robots" carrying out business in computer world. The system they visioned would carry out tasks to achieve a goal on behalf of humans and reach out to them when stuck [10]. According to Nwana [51] agent research can be divided into two main areas: the first beginning about 1977, and the second around 1990. Earlier research was primarily in Distributed Artificial Intelligence (DAI). Those agents were deliberate and concentrated on communication between agents. They also handled the logistics of division of tasks, coordination and cooperation. The second main wave came in 1990’s when more diverse applications are being developed using agent technology. By this time the focus has shifted from deliberation to doing and from reasoning to remote action. As different types of agents have proliferated software agent technology started becoming mainstream [61].

The focus in AI was mainly in representing the knowledge and reasoning of one intelligent agent. In contrast DAI the aim is to reproduce the knowledge and reasoning of several heterogeneous agents that are needed to coordinate to jointly solve various problems. The focus shifted from agents and its autonomy to interactions of multiple agents in DAI. More and more AI scientists started taking inspiration from biology emphasizing on reactive
behavior and worked in area that is called Artificial Life today [76]. This research which is influenced by social and life sciences is based more on physics and the complexities and tries to examine scientific questions while focusing on the interactions between elementary entities and their mode of organization [1].

2.5.2 Evolution of Multi-agent Systems

Between the years 1960 and 1990 AI witnessed a lot of improvements in areas of knowledge representation, inference techniques and machine learning. With the advancements in computer science fields like real-time systems and communication it was possible to design, implement and build agent based systems. The advances in AI and computer science have made it possible for applications in distributed databases, information gathering, and collaborative computing to be developed in multi-agent systems. An agent is loosely defined as a self-contained, simultaneously executing software process that encapsulates some state and is able to interact with other agents. Artificial Intelligence researchers regard an agent to be a computer system that is realized using notions and behaviors that are more usually applied to humans. For example, it is quite common in AI to characterize an agent using notions, such as Knowledge, belief, intention, and obligation [121].

Research on multi-agent systems grew independently and simultaneously until about 1990’s. Agents then concentrated more on production systems that consist of a set of rules, working memory to store facts and a pattern matching logic to produce the right actions. The main hindrance of this approach is the unstructured knowledge of the system. The first solution to this problem has been provided by blackboard systems [112] which can also be called as the first real multi-agent systems. The blackboard system is a collection of knowledge sources from individual entities. When a knowledge source finds a solution or a partial solution or even solution to a partial problem it is written on the blackboard. This process goes on until the problem is solved. So this system and other multi-agent systems developed in mid 1980’s the main focus was in problem solving or distributed problem solving [76].

Individuals with object oriented (OO) background usually confuse agents as an extension to OO concepts. Wooldridge and Jennings [79] explain the difference between agents and objects. Object oriented approach brought segmentation in code by introduction of
methods. This allowed defining and controlling of local variables. An important feature of OO encapsulation of state is achieved through methods. But still objects are considered passive because the methods can only be invoked through an external entity. Agents have their own thread of control. Not just the code and state but also its initiation is localized. Each agent has its own rules and goals. They hold the decision power on how and when to act. Agent based way of thinking brings a different perspective to system development.

Interest in multi-agent modeling took a big leap in 1990s. Along with steady interest in agents and more applicability in industry internet revolution also boosted its usability. The growing popularity of internet made distributed and networked systems possible and agents started gaining importance in electronic commerce to automate tasks. During the same time the idea of mobile computing was developed. From mid-1990s interest in standardization also grew. The remarkable number of conferences also indicates the significance of multi-agent systems in various fields. MABS (workshop on Multi-agent systems and Agent-Based Simulation) [14], AAMAS (Conference on Autonomous Agents and Multi-Agent Systems) [13] and Agent-Based Models conferences [27] are some of the events that are dedicated to multi-agent systems.

As stated earlier multi-agent systems are designed and developed as several interacting agents. Due to interactive and autonomous nature they are best suited where little or no interaction is required to maintain an application. They provide good solutions which have multiple problem solving methods, various perspectives and many problem solving entities. The expressive power of ACL languages adds an additional advantage where sophisticated patterns of interaction are needed [81].

2.5.3 Applications of Multi-agent Systems

The number of agent based applications being developed and deployed in real world settings is rapidly increasing. They have been successfully implemented in manufacturing, process control, telecommunication systems, air traffic control, traffic and transportation management, information filtering and gathering, electronic commerce, business process management, entertainment and medical care. No matter which field area they are implemented in they take up different roles as personal agents, information filtering agents, e-mail agents, scheduling agents and much more. Ferber [59] gives the five categories
where multi-agent systems can be applied: problem solving in the broadest sense, multi-agent simulation, building artificial worlds, collective robotics and program design.

Parunak [52] describes about a multi-agent system in manufacturing which efficiently manages production process at different plants. Multi-agent approach handles enormous complex tasks and constantly changing parameters like products to be manufactured, available resources, time constraints, and so on. Distribution of tasks to different agents helps achieve goals in this application.

ARCHON ™ (Architecture for Cooperative Heterogeneous ON-line systems) was devised as a general-purpose architecture, software framework, and methodology which have been used to support the development of DAI systems in a number of real world industrial domains. Electricity transportation management and particle accelerator control are two of these applications, which use ARCHON and have been run successfully [80]. This is also one of the world’s earliest field-tested multi-agent systems. Multi-agent system has gained significant attention in solving control problems because of the advantages it offers in solving complex problems with uncertainties. Other areas where agent-based process control systems have been used are for monitoring and diagnosing faults in nuclear power plants, spacecraft control climate control and steel coil processing control.

Ljunberg and Lucas [71] describe a sophisticated agent-realized air traffic control system known as oasis. This system has been implemented in Sydney airport in Australia, where agents are used to represent aircraft as well as various air-traffic control systems in operation. The agent approach to traffic control process is predicated on adaptive agents taking independent actions that provide good system-level behavior. The domains of traffic and transportation management are well suited for an agent-based approach because of its geographically distributed nature. For example, [23] describe a multi-agent system for implementing a future car pooling application.
Multi-agents have also found to be equally useful in entertainment applications. Grand and Cliff [101] built the highly successful Creature’s game using agent techniques where users can interact in real-time. Creature’s game uses a number of synthetic agents and provides a rich, simulated environment. Wavish et al., [92] also describe several applications of agent technology to computer games. These applications demonstrate how multi-agent applications can be given a user-friendly Web interface which still remains a loosely coupled component of the system by using the gateway agents.

Just like in entertainment applications multi-agents have also been very successful in medical and pharmaceutical applications. The guardian system [24] is developed to help manage patient care in the Surgical Intensive Care Unit. Siebers et al. [91] used multi-agents to understand the dependency between human resource management practices and how it affects retail productivity. These systems use multiple agents which are loosely coupled to perform perception, action, and cognition functions.

2.5.4 Multi-agent System in Agricultural Domain

There are very good working websites related to farming which are started and maintained by either government of India or independent organizations like Farmer’s portal [42], Agricultural Meteorology Division [19], Department of Agriculture and Cooperation [30], Indian Farmers [22], Indian Society of Agribusiness Professionals [55] etc. All these websites provide good content in various divisions like seeds, pesticides, market prices, government policies etc. Most of them don’t have any personalization features which mean they are
providing the same content to all the users. Often the page loading times are slow and the layout of the pages is not very easy to navigate for less experienced individuals. The huge amounts of content they provide is great but it also means dedicated individuals who constantly have to update the information. There is a lot of maintenance involved.

There are farming portals like Way 2 farm [118], Krishi World [55] etc. They have good intentions to get farming community online and introduce them to global market. They include content that caters to farmers, seed distributor, fertilizer manufactures, advisors, etc. It is not enough if portals just provide content and add some forums where discussion with other farmers is possible. If the content is not organized well enough and there is no easy way to search through available information then that information cannot reach target audience. When audience’s education levels are low and resources are limited even more care should be taken on how to present content.

There are also a number of websites like Navadanya [87], Villgrow [115], Indian Council of Agricultural Research [55], etc. These sites get innovative and new trending ideas to users. Farming is a major source of income in all parts of India. Each state has their own farming site focusing on locally grown crops. Similarly all agriculture institutions have their own websites which focus on the research going on in their institutions. So the information is everywhere and it is scattered it takes a lot of effort and research time to look for the right information.

Sxagro [16] is a decision support systems that is catered to farming industry and also the advantages of these kinds of systems are mentioned in papers like [95] and [74]. They provide necessary information to farmers for making appropriate decisions at the right time. The usefulness of decision support systems is as good as the input data. With the dynamic nature of data it becomes hard to maintain good data.

The various solutions mentioned earlier provide the content but it is not brought together to be more accessible. Resource limitations and education background are not taken into account to make the applications more users friendly. Sites which provide useful information just tabularize data instead of grouping them to make it easily accessible. User profile information is not collected at all or it is not given much importance providing the
same content to all. The goal of this research is to provide more than just decision support system. The portal proposed in this research wants to integrate all the information relevant to users needs and also provide good advisory system. Combining the advantages of local knowledge bases and tapping into the vast resources available on internet provides the best solutions for information overload problems. Making use of personalization and feedback techniques through profile information content can be customized to their needs. This speeds up page loading times and also filters unwanted data.

Along with looking at the existing online resources available various farmers were interviewed. Talking to them helped in grasping present day scenarios and the issues they are facing at the moment. Making a list of all the problems they have, and looking at their needs the objectives for this research are arrived at. As part of research I have visited a number of libraries in India and in US. A hard look at the resource availability is also made before deciding on the technologies to be used. Since multi-agent systems are very good at solving complex real world scenarios it is chosen as the technology that will be used to implement the prototype presented in this research. Multi-agent systems are very robust and at the same time flexible enough so other technologies can be easily integrated into them. They have already been proven to be successful in a number of practical applications making it an ideal choice for this research work.

2.6 Conclusion

This chapter gives an overview of what an agent is and its capabilities. There are variety of agents and each has its own characteristics all of these are covered. One of the factors that determine outcome from agents is its environment. Different kinds of agent environments are also looked in detailed in this section. Finally attributes that constitute intelligent agents are described towards the end of the section.

This chapter also introduces the concepts that constitute a multi-agent system. Multi-agent systems have been used in a lot of real world applications. A mention of domains where they are used and a reference to papers that talk about them are given. There are important characteristics that are mark of these systems and studying them gives a better understanding of the workings of multi-agent systems. Different standard languages to communicate between multiple agents that are available are discussed. Important
fundamental concepts of multi-agent systems are cooperation, competition, negotiation, homogeneous systems and heterogeneous systems. A detailed discussion of the process involved and the issues in each one of them is covered. Understanding the effects of each of the concepts helps in designing a better system which is covered in the next chapter.

Beside the basics of agents and multi-agent systems, this chapter covered the components that are used in this research in conjunction with multi-agent systems. JADE is a free open source framework that aids in the development of multi-agent system proposed in this research work. JADE is complaint with FIPA specification and is established software that is distributed by Telecom Italia. The move towards semantic web makes the use of Ontologies a must. A part of the application is developed using Ontologies and the ways to run the queries and retrieve information from them is given in next two chapters. The uncertainty involved in user inputs is tackled with the use of fuzzy logic. This section covers the basics of fuzzy logic and the steps involved in the fuzzy system.

Final section of this chapter looks at the literature survey and how it is carried out. The concept of multi-agent systems started evolving a long time ago so a brief history of its roots and how they have matured is presented. The similarities with Artificial Intelligence and how they differ from object oriented concepts are also covered. Multi-agent systems have been used to handle a number of real world applications. Some of the applications that were implemented successfully are given. There is a wide range of domains where they are used. In the final section present day scenarios and existing solutions available are researched. Their limitations and proposed prototype and how they fit into multi-agent systems is looked at. Based on the literature survey and study of the fundamental technologies such as Ontologies and fuzzy logic, a framework based on multi-agent system for agricultural activities is designed. Chapter 3 illustrates the general framework.