CHAPTER 3

MULTI-AGENT SYSTEM

In the supply chain management system, there exists distinct, dynamic and distributed nature of data and applications for running the various activities. These activities require the solutions that are competent of responding to requirements for information and wisely look forward to become accustomed and dynamically support the users. The intelligent agents can sustain obviously distinct tasks or progression, interrelate with each other in a precise atmosphere such as inventory management, work together with other agents straightforwardly or through the messages, continuously connected real-time data like RFID, GPS, sensors, actuators and contribute to this data with other agents to offer real-time flexibility in demand driven supply chain system. This idea is the core of multi-agent system in implementing the supply chain management system. The intelligent agent provides real-time adaptability that affects huge collection of static or pre-set trade processes in the supply chain system.

3.1 WHY MULTI-AGENT SYSTEM?

It is expected that numerous processes may perhaps modify to develop into the paradigm move that is the base of the adaptable business network. In particular, real-time adaptability may transform supply chain management for encouragement of supply chain innovation through deployment of Multi-Agent Systems. In business supply chain software, the processes are described in the requisites of rates and flows of expenditure and manufacture. Their system variables like cost, rebates, transportation time & out-of-stock are being calculated to optimize for the most excellent outcome.

In the real world, the events are non-linear, actions are disconnected and information about data is scattered. The solutions shall come out agents-based software that works incessantly and autonomously in the surroundings and processes. The stability and self-sufficiency indicate that agents are capable to carry out process or actions in a flexible,
Intelligent manner that is mutually adaptable and receptive to changes in the situation exclusive of requiring steady human supervision. The agents are able to perform the functions constantly in a situation over an era of time with learning from experience. The intelligent agents can populate an environment with other intelligent agents having the capabilities of communication, cooperation and mobility from one location to another location. The portable, networked, self-directed, self-learning, adaptive intelligent agent may have fundamentally dissimilar ethics compared to those solutions that were developed for monolithic systems. Hence we have selected the groups of intelligent agents (Multi-agent system) that implement the supply chain management system with their capabilities.

3.2 INTELLIGENT AGENT

The intelligent agents are the fastest emergent area of information technology which is being utilized for developing the applications suited in various areas.

<table>
<thead>
<tr>
<th>S.N</th>
<th>Author</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Shoham et al. (1993)</td>
<td>An agent is a software entity that functions continuously and autonomously in a particular environment, often inhabited by other agents and processes</td>
</tr>
<tr>
<td>2</td>
<td>Smith et al. (1994)</td>
<td>a persistent software entity dedicated to a specific purpose</td>
</tr>
<tr>
<td>3</td>
<td>Selker et al. (1994)</td>
<td>computer programs that simulate a human relationship by doing something that another person could do for you</td>
</tr>
<tr>
<td>4</td>
<td>Janca et al. (1995)</td>
<td>a software entity to which tasks can be delegated</td>
</tr>
<tr>
<td>5</td>
<td>Maes et al. (1995)</td>
<td>Agents are computational systems that inhabit some complex, dynamic environment. They sense and act autonomously in this environment, and by doing so realize a set of goals or tasks for which they are designed</td>
</tr>
<tr>
<td>6</td>
<td>Wooldridge and Jennings et al. (1995):</td>
<td>An agent is a computer system situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objects.</td>
</tr>
</tbody>
</table>

Table 3.1 Intelligent agent’s definitions
There is no universally granted definition of an intelligent agent. Depending on these definitions, the intelligent agent may be defined as the autonomous program competent of controlling its own decision making and substitute based on its observations of its environment for the purpose of searching the predefined objectives.

In another sense, the intelligent agent may be classified as the computational entity that can identify its environment through sensors and perform the actions upon that environment through effectors. The structure of the intelligent agents resemble with the human beings. The human being uses their eyes, ears and supplementary organs for retrieving the information from the environment and hands, legs, mouth and other body parts for applying the action on the environment. The cameras and infrared range finders provide the inputs to the robotic agent as the sensors and various motors for the effectors. The intelligent agent has programmed bitwise strings as its precepts and actions. The generic intelligent agent is diagrammed in Figure 3.1.

![Diagram](image)

**Figure 3.1 Intelligent agent**

The intelligent agents are designed to perform the functions incessantly and autonomously in a specified environment. Their continuity and autonomy features indicate that the intelligent agents are competent to perform the tasks in a flexible and intellectual approach having adjustment and awareness to modifications in situations [79]. The intelligent agents show the significant task in solving the real-time applications. The intelligent agents are those computational entities that take the input through sensors and works on the environment with the help of the effectors. The environment may be
classified as set of information concerning the real-time applications. There are following characteristics of the intelligent agents described as below:

- **Autonomy**  
The intelligent agents are capable to execute the preponderance of their problem solving tasks not including the straight interference of humans or other agents, and they have the full control over their own actions and interior circumstances.

- **Social ability**  
The intelligent agents are competent to work together with other intelligent agents and humans for resolving their own crisis and to assist others with their activities.

- **Responsiveness**  
The intelligent agents are capable of distinguishing their environment and react in a well-timed fashion to changes occurring in the environment. This factor indicates the intelligent agents’ efficiency regarding the current problem they are working.

- **Temporal stability**  
The intelligent agents are competent to judge the process exclusive of the end, incessantly running and developing its functions.

- **Adaptively**  
The intelligent agents support their adoption through the learning process. The intelligent agents are proficient to discover the changes in the environment & modify its activities based on that learning.

- **Mobility**  
The intelligent agents are capable of shifting from one circumstance to another or moving the agent’s code and starting the agent anew, or by serializing code and states, allowing the agent to persist completing in a new perspective, retaining its state to continue its work. This feature of the intelligent agent enables the designer to apply them in the distributed computing environment.

- **Persistence**  
The Intelligent agents support the quantity to which the infrastructure enables agents to preserve information and situation over extended periods of time in problem solving phenomena.
• **Pro-activeness**
  The intelligent agents act in reaction to their environment. They can demonstrate opportunistic, goal-directed behavior and taking the initiative regarding the situations [80].

These factors indicate the basic features of the intelligent agents that make them the best choice for resolving the real complex problems. The intelligent agents are more suitable in those situations having challenging domain environment. In most cases distinctive agent environments have following characteristics described below:

• **Dynamic**
  These environments are dynamic in regards the change in situations. The states of the environment change during period of performing the task. It will not stay motionless during achieving the goal.

• **Unpredictable**
  There is not full information available having all predefined events & stages to forecast the future states of the environment. They do not have the entire information about their surroundings.

• **Unreliable**
  The actions performed by intelligent agent may not succeed for reasons that are away from an agent’s control.

These factors show the characteristics of the environment involved in the agent applications. These facts show the complexity of intelligent agent based applications in which the environments have all above described features.

3.3 **STRUCTURE OF INTELLIGENT AGENT**

The term agency provides the theoretical and physical position for inhabiting and implementation. The main components of the agency are the agent platform which is responsible for providing local services for agents and includes proxies to access remote services through their component model infrastructure. The intelligent agent aims to provide the scrupulous facilities using service agents, such as Broker, Auctioneer and Community Maker agents.
The agent structure consists two basic components: Agent Program & Architecture. The agent program may be defined the function used for implementing the agent mapping from precepts to actions. The architecture generates the environment on which the program is being run. It comprises software for providing the padding between the computer system and the agent program. The architecture enables the agent program to make the precepts from the sensors accessible to the program and feeds the program’s action choices to the effectors. The relationship among agents, architectures, and programs can be summed up as follows:

\[ Agent = architecture + program \]

On the basis of the possible percepts and actions, the agent programs are being designed in the widest variety. The behavior of the agent is approved with the precept sequence generated by the environment, and the goals that the agent is invented to accomplish. Some real environments are actually moderately unsophisticated.

### 3.3.1 Agent programs

The intelligent agents are designed for accommodating precepts from a situation and executing the actions. The agent programs exploit various inner data structures updated with innovative precepts appear. These data structures are used in the decision-making process to cause an action preference through passing to the architecture to be executed. The example of the agent program is described as below:

```plaintext
function BUSINESS-AGENT( percept) returns action

static: demand, the customer agent’s requirement

    demand UPDATE-DEMAND(demand, percept)
    action CHOOSE-BEST-ACTION(demand)
    demand UPDATE-DEMAND(demand, percept)

return action
```

The agent program specifies the working logic of the intelligent agents during solving the complex real-time problems. The intelligent agent working transfers one state into another state depending on the attributes of the input process [81]. These agents are capable of taking the decisions depending on the current states’ attributes.
3.3.2 Agent architectures

The agent architectures correspond to the shift from hypothetical design to the software implementation. There may be following type of agent architectures as given below:

- **Deliberative architecture**
  
  In this architecture, the intelligent agents preserve the clear demonstration of their working environments in the form of some knowledge symbolic reasoning. The Belief-Desire-Intention (BDI) model is the most widespread model.

- **Reactive architecture**
  
  This type of the architecture is being applied to design the self-directed transportable robots having the capability of adapting the changes in their environment and shifting from one position to another. The agents are capable of making their decisions on exact opportune on the basis of imperfect quantity of information and simple situation-action rule.

- **Hybrid architecture**
  
  It may be defined as the grouping of the classical and alternative approaches to design the intelligent agents for having the advantages of previous approaches and avoid the disadvantages the existing approaches.

- **Layered architecture**
  
  It follows the approaches that different subsystems are prearranged into a hierarchy of layers with different decompositions for the agents to complete their tasks.

The agent architecture contains basic four components that provide the environment for the execution of the agent program as given below:

1. **Agent management system**

   This component provides the mechanism of controls formation, deletion, postponement, recommencement, verification, perseverance and immigration of intelligent agents.

2. **Agent communication channel**

   It passes the messages between the agents located locally & remotely. These messages are being designed in specific agent communication language. They are used to send the information to other agents in the form of the message
exchanges. The information sharing plays very important role in the process of solving problems.

3. **Directory facilitator**

   It provides “Yellow Pages” service for intelligent agents that roll agent capabilities for the suitable task-specific agent to grip the assignment found.

4. **Internal platform message transport**

   It provides the communication infrastructure for sending the message between the agents. It facilities the message passing between the agent platforms.

The intelligent agents implement their message structure with XML structured documents for messages and URL/URIs for naming and locating. The intelligent agents have a globally unique name (AID). For the efficient and reliable way for agents to switch over messages with other agents, the agent systems employ the extraordinary Agent Communication Language (ACL). The agent platform and additional service agents can observe and organize these message interactions to indemnify for confirmation of the desired protocol [82].

**3.4 TYPE OF INTELLIGENT AGENT**

The intelligent agents may be classified into various categories depending on their characteristics. Some agents are hybrids, which demonstrate properties of supplementary than one of the categories listed below. The intelligent agents have following type of categories as given below:

3.4.1 **Reflex agents**

   Reflex agent is a type of intelligent agent that takes the inputs from the environment and compare with rules to determine which actions to carry out. They store the predefined rule for performing the tasks. The example of this agent is the automatic mail filter. The main drawback of such agents is that’s they are no capable of responding the environment changes.

3.4.2 **Goal-based agents**

   Goal-based agents are additional multifaceted than reactive agents. Instead of applying the predetermined set of rules, the goal-based agent acts to attempt to accomplish the goal
with the help of search or planning. The agent will be designed for carrying out actions such as reaching the web pages on the particular topic.

### 3.4.3 Utility-based agents

The working of utility-based agent is related to a goal-based agent, but instead of attempting to accomplish a set of goals, the utility-based agent is also trying to exploit some usefulness assessment. The utility value may be defined as the contentment of the agent regarding the performance of intelligent agents. They use the heuristic-based search techniques to diminish the quantity of time it spends for achieving the targets.

### 3.4.4 Interface agents

These agents provide the interface between the user activities and the problem they tried to solve. These agents provide the assistance in solving the problems. The example is a tool used for assisting the user in operating newfangled software package. In given example, the interface agents get the required information about the software tools and provide the mechanism to operate that tool.

### 3.4.5 Mobile agents

Such type of the intelligent agents has a special feature of switching from one system to another system. Alike the other intelligent system, they are not fixed & capability of learning from other agents. These features make a vast improvement on the performance of the agents.

### 3.4.6 Information agents

This type of intelligent agents is being designed for the purpose of collecting the required information over the Internet. The user sets the type of the information he/she required. Then these agents search the web pages related topic & collect them for the propose of fulfilling the user requirement.

### 3.4.7 Multi-agent system

Sometime, there exist various problems in the real-time world which cannot be solved by a single agent. The solution of these problems is beyond of the capabilities of intelligent agents. Hence they requires group of the intelligent agents that work together to provide
the solution to these complex problems. Such group of intelligent agents is known as the multi-agent system. The multi-agent system has no central control and lacking of the local view of the system.

3.4.8 Collaborative agents

This category works together for solving the complex problems. These agents have collaborative nature with learning facility that’s extends the knowledge base over the time. These collaborative natures of intelligent agents provide more reliability and decrease the probability of failure of the system. They provide more availability in comparison of traditional intelligent system [83]. These different types of intelligent agents play an important role in developing the real-time applications. They are different from the ordinary software developed using particular programming languages.

3.5 COMMUNICATION BETWEEN AGENTS

Each activity performed by intelligent agents require passing the message for communicating with each other. The intelligent agents communicate with each other in view of information & knowledge sharing between them. The Agent Communication Languages enable the intelligent agents to pass the messages through the communication protocols such as TCP/IP, SMTP and HTTP. The major agent communication languages used in agent-oriented programming are described as given below:

- **Knowledge query manipulation language**
  
  KQML consists the message handling protocol to sustain the sharing the run-time knowledge among agents for the purpose of cooperative problem solving. It holds the layered architecture of communication providing functionality for purpose of occurrence the message transport specific by the application.

- **Knowledge interchange format**
  
  KIF maintains the aspect of syntax for message content that is designed first order predicate calculus with declarative semantics. It helps in understanding of the messages. Without understanding the message, the actual information cannot be retrieved by other agents. These agents share the common the knowledge interchange format (KIF) which is being specified by the FIPA (Foundation for intelligent physical agents).
• Foundation for intelligent physical agents ACL

This architecture supports general standards for agent interoperability. It includes the essential talkative events such as inform, request, propose, and accept, jointly with communication protocols.

In these architectures the agent are able to send the messages to other agents. The agents pass temporary & final result in the form of the messages.

3.6 BDI ARCHITECTURE OF INTELLIGENT AGENT

The Belief-Desire-Intention (BDI) model is the most widespread model. Its central concepts are:

• Beliefs

It may be defined as the information about situations in which agents be acquainted with their environment. They are initial states of the intelligent agent before they start their operations.

• Desires/Goals

They classify their motivational states that are required to accomplish. The operation of the intelligent agent completes when these states are being met. They are the final states for intelligent agents to be achieved.

• Intentions

They may be defined as the agents’ deliberative circumstances. It makes an impact on the agents’ working.

• Plans

It means the way of achieving certain future states. The performance of the intelligent agents directly depends on the plans made by it.

• Actions

It may be defined as the ways the agent can operate on the environment.

The psychological impression of goals is not adequate to articulate complex future-directed activities. The rational agent should assume only goals which it believes to be feasible. Before committing to the goal, the agent may have dissimilar desires, which may be in divergence like the human having conflicting desires at the same time. The coherent behavior stresses to choose just non-conflicting options.
In BDI architectures, desires are used as beginning stages of potential intentions. In beginning the agent collects pleasing options, and then it selects some of them as intentions for avoiding conflicts between intentions. Then it performs suitable events to accomplish the intentions. Sometimes the agent basically chooses a distinct desire and manipulates the fitting action sequence for fulfills this desire.

In case of the plan failure, the intelligent agent tries substitute plans that fit the same goal based on updated agent data awaiting the recent goal is effectively accomplished. In unavailability of the plans, the intelligent agent declares the goal abortive and forces failure of the plan that was called by the particular goal. By doing this, the agent automatically backs up to the goal that triggered the plan, trying other plans that fit the current situation. As a result, software agents build a successful completion path in real time, even when unplanned exceptions occur [84].

3.7 AGENT-ORIENTED PROGRAMMING

The Agent-oriented programming languages are an innovative division of programming languages that is oriented on the core uniqueness of multi-agent systems. It implements all features of the intelligent agents efficiently. It is capable of developing the intelligent regarding all agent architectures.
This programming approach is totally different from the object-oriented programming approach. The working of the OOP is totally oriented on the object concepts. But in case of AOP, it empathizes on designing of the intelligent agents. The basic difference between the OOP and AOP are being given as below:

<table>
<thead>
<tr>
<th>Building Block</th>
<th>OOP</th>
<th>AOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object having some properties &amp; states</td>
<td>Agent having specified features</td>
<td></td>
</tr>
<tr>
<td>Parameters involved with building blocks</td>
<td>Unconstrained</td>
<td>Goals, Beliefs, desired commitments, capabilities, choices</td>
</tr>
<tr>
<td>Communication</td>
<td>message passing and response methods</td>
<td>Agent Communication Language</td>
</tr>
<tr>
<td>Types of message</td>
<td>Unconstrained</td>
<td>notify, appeal, proffer, assure, reject</td>
</tr>
<tr>
<td>Constraints on methods</td>
<td>None</td>
<td>sincerity, steadiness</td>
</tr>
</tbody>
</table>

Table 3.2 OOP Vs AOP

There exists number of programming languages for automating the agent-based system. The selection of programming language is very critical issue in developing the real-time applications [85].

3.7.1 Why java?

Java is one of the programming language used for developing the real-time applications. It supports a numerous of features that make Java as first choice of the developer. The Java language has following features as given below:

- **Compiled and interpreter**
  In the execution of the Java code, the code has been both compiled and interpreted. It generates the bytes codes instead of than generating the code in the machine language.

- **Platform independent**
  This language is platform independent. It supports the capability of executing the program byte code on different platform with same outputs. There is no need of making the modifications during other platforms.
• **Object-oriented**
  It is an OOP Language with having full support of all OOP features. It contains bundles of classes and packages. It has great facility of the code reusability and maintainability.

• **Robust**
  The Java program code is capable of checking the reliability of the code before its execution. It checks the data type conversions and running the garbage collection mechanisms.

• **Secure**
  The Java code does not allow executing unsecured applets for maintaining the security of designing the system. It provides the secure environment. It implements several security mechanisms to protect you against malicious code.

• **Distributed**
  Java contains the remote method invocation (RMI) that enables the user to execute the Java program on the network. The Java programs can be run on the server-client configurations using TCP/IP protocols such as HTTP and FTP.

• **Simple**
  It is the simple to learn & design the program. It doesn’t employ the pointers for representing the storage locations & doesn’t support Multiple Inheritance.

• **Multithreaded**
  It supports the multi-threading features and synchronization of multiple processes running at the same time. The thread is a light weight process used to complete various functionalities.

• **Dynamic**
  By nature Java has dynamic in the natures. A Java program is capable of reflecting the changes during their execution.

• **Secure**
  Java was designed with security in mind. As Java is intended to be used in networked/distributor environments so it implements several security mechanisms to protect you against malicious code that might try to invade your file system. These mechanisms enable the Java program more secure.
- **Architectural neutral**

  It is architecture neutral. It follows 'Write-once-run-anywhere' approach through the byte-code format which is independent of machine architecture but can be easily translated into a specific machine by a Java Virtual Machine for that machine.

  These factors show the reasons of using Java as the most suitable language for designing the intelligent agents.

### 3.7.2 Creating intelligent agent in java

Agent-oriented programming allows the Java programmer to create the intelligent agent easily. It requires two easy steps in creating the instance of the intelligent agents. First the programmer extends the Agent class in own class and then write down the working logic in setup() method as described in the given code below:

```java
import jade.core.Agent;

public class SCMAgent extends Agent {
    protected void setup() {
        // Working logic of the agent
    }
}
```

Given an example explains the way of creating the SCMAgent as the instance of Agent class. The working logic is handled in the setup() method including all intelligent agents’ methods.

The setup() method is coded to provide the agent initializations. The way of performing a task is being described by the agent behaviors. There exists a number of the agent behaviors which help the programmer to define the mechanism of solving the problem in action() method. After completion of the task, the agent instance will be destroyed in the doDelete() method.

This method frees the memory allocated to that agent instance. The Agent class does not consist any type of constructor of the initialization of the agents attributes. These attributes are initialized in the setup() method [86]. The working of intelligent agent’s instance is shown with given flowchart as below:
The figure 3.3 indicates the execution of the agent life cycle which starts with the setup() method and terminates with done() method.

### 3.8 MULTI-AGENT SYSTEM

The multi-agent system is the main category of intelligent agent. This type of intelligent agents is intended to design for providing the solution of those complex problems which are not beyond the scope of single intelligent agent.
As shown above in figures 3.4, there exist multiple intelligent agents interacting with the particular environment and communicate with each other.

### 3.8.1 Features of multi-agent system

There are following factors showing the ability of the multi-agent system to solve the complex problem as given below:

- The multi-agent systems are proficient to solve problems that are excessively outsized for the centralized agent to solve due to the resource margins.
- These systems are capable of interconnecting and interoperation of multiple existing legacy systems to meet with changing business needs.
- They are capable to provide solutions to problems that can logically be regarded as a culture of self-sufficient interacting components-agents. For example the meeting scheduling system for managing the user meeting in the autonomous and interacting mode.
- The multi-agent systems are expert in providing the solutions for the situations using spatially distributed information sources like the sensor networks, seismic monitoring, and information gathered from the internet.
• The multi-agent systems are being designed for providing solutions in situations where expertise is spread like concurrent engineering, health care, and manufacturing.

• The multi-agent systems support the faculties to improve performance along the dimensions of computational factors such as competence, consistency, extensibility, robustness, maintainability, sensitivity, suppleness and reuse.

These facts show the basic features supported by the multi-agent system for solving the complex problem under unpredictable & inaccessible environment.

3.8.2 Comparisons with conventional agents

This approach is different from the conventional intelligent agent system. The multi-agent system differs from the conventional intelligent agent as described below:

• In case of the single intelligent system, the agent contains the full information about the working environment. But in case of the multi-agent system, every agent consists the partial information or capabilities for solving the problem. They contains partial viewpoint.

• Conventional intelligent agent has the control over the entire system. But multi-agent system consist no organization comprehensive control for solving the complex problem.

• The data is stored centralized in conventional intelligent agent system bur data are decentralized in working of the multi-agent system.

• Instead of conventional intelligent agent system, the agents perform the computation asynchronously in the multi-agent system. The results are being shared between the intelligent agents.

Intelligent agents are used to interact in a flexible and dynamic way to solve problems more efficiently. The working of the multi-agent system is based on the cooperative distributed problem solving approach [87].

3.8.3 Methodologies for MAS development

The methodology may be defined as guideline for solving the problem with precise mechanism such as phases, tasks, methods, techniques and tools. It provides the analysis of the philosophy of applied methods and rules and carries the methodical
revision of methods that have been applied through description of methods. The assortment of an accurate methodology is critical for any software project. As the external viewpoint, the intelligent agents are computational models that support the features of autonomy, decision-taking, mobility and dynamic for solving the problems in both centralized and distributed systems easily. The basic purpose of agent organization is to produce the module usability that minimizes the design issues difficulties. It is a combination of various methods and actions providing the services and utilities for equally the horizontal and vertical collapse of multifaceted organization expansion.

The organization of the intelligent agents is designed for the following steps as given below:

- It identifies the roles in the application domain for fulfilling the system requirement.
- After knowing the each role, then they identify its connected responsibilities, and the services provided by the intelligent systems.
- There exist multiple interactions for fulfilling the services. Next the interactions are being identified that is associated with the service for managing the control relationships between agents.
- In the last phase, the agent hierarchy is being redefined. It makes assigning of services to agents, and services relationships and interactions between them.

Let’s discuss some methodologies used in the multi-agent system. The BDI methodology provides the analysis of the mechanism for achieving the goals divided subgoals and actions. It builds the attitude of the system for analyzing the circumstance and situation making the control over implementation of activities. Vowels engineering had introduced the methodology that emphases mainly on the agent, its environment, their interactions, organization implemented, and user response.

CommonKADS methodology projected six models to recognize the society in which the knowledge base system works with intelligent agents design. In case of the MAS-CommonKADS, it supports the facilities of the OMT object-oriented information to composition systems for capturing necessities and message sequence charts to explain agent connections [88]. All these methodologies enable the developer to create the intelligent agents.
3.8.4 Agent platforms for MAS development

The agent platform may be defined as technological architecture which generates the mechanism to run the agents for achieving the goals. The basic aim of using the agent platform is to develop the intelligent agents for designing the real-time applications. It supplies the agents with various functionalities such as agent intercommunication, agent autonomy, yellow pages, mobility etc. The agent platforms contain solitary core container which is the instance of the virtual machine.

The agent platform provides the mechanism of communication and exchanging the service. It should follow required specification suggested by the Foundation of Intelligent Physical Agents (FIPA). The selection of the agent platform is dependent on the factor indicating how much the platform following the specifications. The FIPA has listed following platforms which are more acquiescent with FIPA specifications:

- Agent Development Kit
- April Agent Platform
- Comtec Agent Platform, FIPA-OS
- Grasshopper
- JACK Intelligent Agents
- JADE
- Java Agent Services API
- LEAP (now subcomponent of JADE)
- ZEUS

The agent platforms are used to develop the multi-agent system. These platforms are divided into three categories as follows:

1. Those agent platforms which are oriented on interior agent reasoning mechanism and hold strategy, goals. Examples of this type of agent platforms are PRS, UMPRS, JAM, JACK, DECAF, Zeus, AgentBuilder, JADEX.

2. Those agent platforms that spotlight on inter-agent communications. These agents usually provide infrastructure for inter-agent communication. The JADE, OAA and Zeus come under this category.
3. Those agent platforms that focus on mobile agents. Examples: Grasshopper and Aglets.

Each type of agent platform category provides some specific benefits over others. The first class is most useful in providing sustained for implementing designs developed through Prometheus methodology. The inter-agent communication is an area of expertise of the second class [89].

3.8.5 Agent behaviors

The agent behaviors may be defined the specified pattern of performing the pre-described tasks as the goals. The agent behavior is an important issues in executing the task at required constraints. For using the different type of availed behavior the programmer requires to extend the jade.core.behaviours.Behaviour. There is the real way to extend the working in different agent behavior. It requires to call the addBehaviour() method of the Agent class. Behaviors can be coded in the setup() method or from within other behaviors. There exist following type of Agent Behavior as given below:

- **Simple behavior**
  
  It is an atomic behavior of the intelligent agents. This abstract class models behaviors that are made by a distinct, monolithic task which cannot be interrupted. This behavior is being further extended to create the different type of the agent behaviors.

- **Cyclic behavior**
  
  It is atomic agent behavior that allows the agents to perform the specified task forever. This category provides the specified way to execute the agent task fully or some portion of the task. It allows executing some user defined portion forever then it is known as Ticker Behavior.

- **OneShot behavior**
  
  It is the intelligent behavior that executes the task only one time. It allows the execution of a task on the basis of various factors such as time, message received. The task is executed on specified time then it is known as Waker Behavior. When the task is executed on the message received then it is called as Receiver Behavior. This behavior plays very important role in the agent working on the most complex applications.
- **Parallel behavior**
  
  It provides the mechanism to execute more than one behavior at the same time in parallel. The main aspect of this behavior is the termination condition that specifies when these all behavior terminates.

- **Sequential behavior**
  
  It allows the execution of the behaviors one after the other and terminates when the last behavior has terminated.

These behaviors help the programmer to execute the tasks in the sequential mode or parallel. It enables the developer to execute the task on the specified time or basis of message arrival. They provide the facility of executing the task repetitively [90]. These features enable the intelligent agent to perform the complex task easily.

### 3.8.6 Teamwork in multi-agent environments

Multi-agent systems are loosely coupled autonomous agents working together for solving the particular problem. They solve the complex problems through cooperation, coordination and negotiation with one another. The main concern of this system is the working of the intelligent agents in the groups with enhanced interactions. The concept of teamwork is being applied to achieve the system goals through achieving the individual goals by every intelligent agent.

The teamwork concept plays very important role in the working of multi-agent systems. These systems perform combined actions with criteria of shared awareness, obligation to cooperative actions, pledge to reciprocate hold and development of sub-plans. There are following fundamental aspects of the teamwork is given below:

- working mutually to accomplish a familiar objective;
- persistently monitoring the evolution of the team endeavor as a whole;
- serving one another as needed;
- coordinating individual events without interfering with others;
- communicating (fractional) successes and failures required for the team to accomplish something;
- No antagonism amongst the team members for achieving the common goal.
With respect of all these points, to maintain teamwork in the multi-agent system is an extremely multifaceted issue. There are following reason behind the facts as given below:

1. There is no global system view. Every intelligent agent contains only a limited view of the entire system
2. The computations performed by individual intelligent agents are unsynchronized with respect of other intelligent agents [91].

The figure 3.5 shows the result of unguided teamwork in the organization. Hence for proper utilization of the teamwork in the multi-agent system the coordination is required. With the help of the coordination between various intelligent agents, the better & more efficient teamwork can be achieved.

![Figure 3.5 Result of mismanaged Teamwork](image)

3.9 COORDINATION IN MULTI-AGENT SYSTEM

The coordination in multi-agent systems may be defined as the process in which group of intelligent agents’ works together melodiously. There exist two types of coordination—resource coordination and goal interaction coordination. The resource coordination deals with the intelligent resource allocation between the different agents. The goal interaction coordination deals with planning for achieving one of its goals.

The basic aims of maintaining the coordination between agents to develop the multi-agent system more stable, fair, efficient and robust. The stability of the multi-agent system refers to the quality of being unwavering to be long-term equilibrium. The stability is a consequence of the means of the agents interacting with each other. The fairness feature of the multi-agent system provides the environment the agents in which they are neither discriminated by other agents, nor by the environment. This feature consists the reasonable process prioritizing strategies regarding their submission to
common rules, and the ability to share out the rewards and punishments to the agents. The efficiency of the system defines the superiority of execution and performing high-quality with the slightest attempt. The robustness of multi-agent system is concerned with chance of being failures of the system during the operations performed by agents. The coordination of the multi-agent system defines the mechanism of allocating the resources to the intelligent agents. This process involves the request the resource from another agent or reassigns a resource from one of its other goals.

![Diagram](image)

**Figure 3.6 Without coordination**

Without proper resource coordination, the utilization of the resources is needed for developing the real-time applications. In this case, the resource is reassigned from one of its other goals to the goal that needs a resource. The environment is dynamic if during the operation the real-time state transforms. It may alter the sequence of the actions of the other agents in the environment. The goal interaction coordination is required to handle state change made by one agent in the environment interferes with the planning of the other members. The state change of an intelligent agent may affect the planning of other agents in achieving the system goal [92].
Both resource & goal coordination makes the multi-agent system more efficient in resolving the real time problems by providing the optimized solution through the teamwork of various intelligent agents.

3.10 MULTI-AGENT BASED SIMULATION

Multi-agent based simulation is used in numerous areas these days increasingly replacing the variety of the micro-simulation, the object-oriented or individual-based simulation techniques previously used. The simulation technology helps in the process of designing of multifaceted, energetic and stochastic systems in nature. It simulates the strategic and tactical decision making process.

The simulation model consists of the collection of rules that describe working of the system with respect to the changes over time. These types of models are not solving the problem in run time. It demonstrates the customs of solving the problems. The simulation does not make the decision making process but it supports the decision allowing better informed decisions to be made.

There exist various multi-agent based simulators for analyzing processes of the proposed system as given below:

- MASON (Multi-Agent Simulator Of Neighborhoods... or Networks)
- SeSAm (Shell for Simulated Agent Systems)
- MATSim (Multi-agent-Based Transport Simulations)
- MASS (Multi-Agent Simulation Suite)
- PlaSMA (Platform for Simulations with Multiple Agents)
- RegMAS (Regional Multi Agent Simulator)
- GALATEA (Glider with Autonomous, Logic-based Agents, TEmporal reasoning and Abduction)

This type of software allows appropriate modeling capabilities in a number of important fields such social science, Biology. It performs modeling socio-technical systems and test environments for agent-based software. They allow simulating the systems that are particularly difficult to treat with traditional approaches [93]. The PlaSMA software simulates the supply chain management system in which the distribution center are located at different locations.