# ARTIFICIAL INTELLIGENCE: ANALYSIS OF VARIOUS AGENT PROGRAMMING LANGUAGES

**Poonam Priyanka Patra[1*], Dr. Prakash Kumar Pathak[2]**

*[1*]M.Tech (Data Science) Gandhi Engineering College (GEC, Bhubaneswar)*
*[2](Professor Dept. of CSE) Gandhi Engineering College (GEC, Bhubaneswar) Odisha (Department of Computer Science and Engineering)*

***Corresponding Author:***

## Abstract—

*Artificial Intelligence is one of the emerging areas of computer science which provides different methods or technologies that can be used to improve the efficiency of existing systems and make them more users friendly. Intelligent Agent is one of the AI technology that had tremendous potential. In this paper, we analyse the various programming languages emerged for the development of Intelligent Agent.*

**Keywords-***Artificial Intelligence, Intelligent Agent, Programming, Languages.*

**INTRODUCTION**

Artificial Intelligence is one of the newest fields of intellectual research; its foundation began thousands of years ago where human fantasy of having intelligent and thinking machines appears in myths or stories. [1] Human likeness believed to have intelligence tools was built in every major civilization such as cult images in Egypt and Greece, humanoid automations in Yanshi etc. [2].In beginning of 20 century, artificial beings or machines becomes a common character in novel or stories based on science fiction. The discipline of Artificial Intelligence was founded at a conference in Dartmouth College, Hanover, New Hampshire, Germany in 1956 [3].

Researchers working in the field of computer science describe – "Artificial Intelligence is the study of systems that act in a way that to any observer would appear to be intelligent". Artificial Intelligence provides methods and algorithms based on the intelligent behavior of humans and other animals to solve complex problems. To develop these methods, algorithms and tools, a continuous rigorous research is conducted which requires lot of resources. Sometimes this research had seen success in the form of expert systems and failure of LISP machines. Now days, AI becomes an essential part of technology industry and impact of AI can be felt around us as it is used in domains such as expert system, robotics, computer games, space research, automobile industry, defence etc. AI systematizes and automates intellectual tasks and is therefore potentially relevant to any sphere of human intellectual activity [4]. The main areas of AI research include reasoning, knowledge, planning, learning, communication, perception and the ability to move and manipulate objects [5]. One of the objectives of Artificial Intelligence researcher's to develop software's which are intelligent enough to know the preference, habits of the user and customize the software according to user preferences so obtained.

In this research paper, we first describe the Intelligent Agents and their importance in different environments. Next, we discuss the features of various programming approaches emerged and can be used to implement Intelligent Agents. Finally, we specify why JAVA is suitable for writing the code of Intelligent Agents.

## I. INTELLIGENT AGENT

An intelligent agent is a set of independent software tools or components linked with other applications and database running on one or several computer environments. The prime objective of an intelligent agent is to store the user preferences dynamically related to an application and implement the same when user accesses the same application. Hewitt in 1977, describes the agent for the first time in its Actor Model. According to Hewitt, agent is a self-contained, interactive and concurrently executing object having some internal state and communication capability.

Agents are linked with research areas like robotics, Artificial Intelligence, distributed systems and computer graphics. In simple words, an agent can be described as an entity that is able to carry out some task, usually to help a human user. [8] Agents embedded in the software are known as software agents which is a computer program designed to carry out some task on behalf of a user ex – Office Assistant in Microsoft Office. A software agent having intelligence as its property is known as Artificial Intelligence Agent. Intelligent agents are the foremost requirement of developing intelligent systems and software's.

Intelligent Agent (IA) is an autonomous entity which observes i.e. learn from its environment and use their knowledge to acts upon an environment and directs its activity towards achieving goals. [7] They may be very simple or very complex: a reflex machine such as a thermostat is an intelligent agent, as is a human being, as is a community of human beings working together towards a goal.

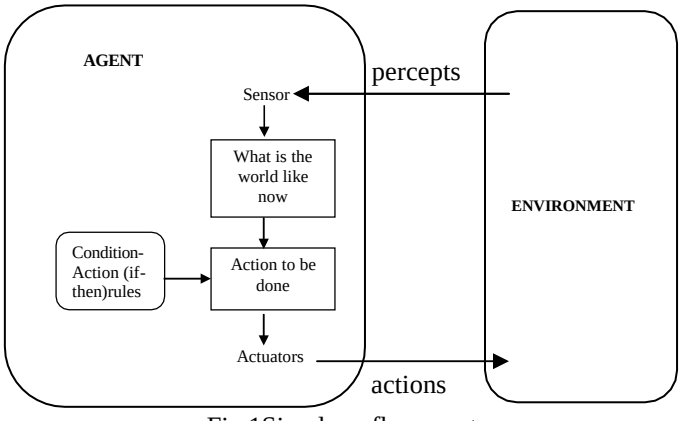


Fig.1Simplexreflex agent

Intelligent agents are often described schematically as an abstract functional system similar to a computer program. For this reason, intelligent agents are sometimes called abstract intelligent agents (AIA) to distinguish them from their real world implementations as computer systems, biological systems, or organizations. In computer science, the term intelligent agent may be used to refer to a software agent that has some intelligence, regardless if it is not a rational agent by Russell and Norvig's definition.[6]For example, autonomous programs used for operator assistance or data mining (sometimes referred to as bots) are also called "intelligent agents". An Artificial Intelligent agent should have the properties like: Intelligence, Autonomy, Mobility, Reliability, Ability to learn, Cooperation, Versatility etc. Implementation of agents can be done using different programming approaches or languages as the agent possess

different properties and used in different environments. In next section, we describe the programming approaches that can be used for developing the various agents.

**PROGRAMMINGLANGUAGES**

To develop any piece of software components, specialized languages are required commonly known as programming languages. In history of computer science different programming paradigms are emerged to fulfill the need of software industry. From the birth of Intelligent Agents, till date various programming languages have been developed to develop different types of agents. These languages are described below-

***A. Agent Programming Languages***

In mid 1980's, the need for standardization of software tools is identified for the development of reliable agents based software as a number of prototypical agent languages is emerging. [Gasseret al., 1987]. By an agent language, we means a system that allows one to program hardware or software computer systems in terms of some of the concepts developed by agent theorists. Agent language at least must include some structure corresponding to an agent and attributes such as beliefs, goals or other mentalistic notions used to program agents.

Concurrent Object Languages

Concurrent object languages are in many respects the ancestors of agent languages. The notion of a self contained concurrently executing object, with some internal state that is not directly accessible to the outside world, responding to messages from other such objects is very close to the concept of an agent as we have defined it. The earliest concurrent object frame work was Hewitt's Actor Mode [*Hewitt,1977;Agha,1986*]; another well known example is the ABCL system [*Yonezawa, 1990*].

***B. Agent Oriented Programming***

The key idea that informs this agent oriented programming paradigm is that of directly programming agents in terms of the socialist, mentalist, intentional notions that agent theorists have developed to represent the properties of agents. The motivation behind such a proposal is that humans use intentional stance as an abstraction mechanism for representing the properties of complex systems. In the same way, language use the intentional stance to describe humans, it might be useful to use the intentional stance to program machines. According to Shoham, agent programming language must have: A logical system for defining the mental state of agents; Interpreted programming language for programming agents and Gentrification process, for compiling agent programs into low level executable systems. AGENT0 was developed by Shoham which includes the components specified above.

In this language, an agent is specified in terms of a set of capabilities, a set of initial beliefs and commitments, and a set of commitment rules. The key component, which determines how the agent acts, is the commitment rule set. Each commitment rule contains a message condition, a mental condition, and an action. In order to determine whether such a rule fires, the message condition is matched against the beliefs of the agent. Actions may be private, corresponding to an internally executed subroutine, or communicative messages. Messages are constrained to be one of three types: 'requests' or' unrequests' to performs or refrains from actions, and informs messages, which pass on information. Request and unrequest messages typically result in the agent's commitments being modified; inform messages typically result in the agent's commitments being modified; inform messages result in a change to the agent's beliefs. In agent oriented programming, agents are not able to plan and communicate requests for action via high level goals.

***C. Planning CommunicatingAgents(PLACA)***

To overcome limitations of AGENT0, Thomas in 1993 develop Planning Communicating Agents (PLACA). Agents in PLACA are programmed in much the same way as in AGENT0's, in terms of mental change rules. [9] The logical component of PLACA is similar to AGENT0's, but includes operators for planning to do actions and achieve goals. The semantics of the logic and its properties are examined in detail.

***D. Concurrent METATEM***

Fisher in 1994 designs a new language which overcomes the drawback of both AGENT0 and PLACA, that is, the relationship between the logic and interpreted programming language is only loosely defined: in neither case the programming language can truly execute the associated language. A Concurrent METATEM system contains a number of concurrently executing agents, each of which is able to communicate with its peers via asynchronous broadcast message passing. Each agent is programmed by giving it a temporal logic specification of the behaviour which is intended to be displayed by the agent. An agent's specification is executed directly to generate its behaviour. Execution of the agent program corresponds to iteratively building a logical model for the temporal agent's specification. It is possible to prove that the procedure used to an execute agent specification is correct, in that if it is possible to satisfy the specification, then the agent will do so [Barringer et al., 1989]. The logical semantics of Concurrent METATEM are closely related to the semantics of temporal logic itself. This means that, amongst other things, the specification and verification of Concurrent METATEM systems is a realistic proposition.

***E. TELESCRIPT***

TELESCRIPT is first commercial agent language developed by General Magic Inc. It is a language based environment for constructing agent societies. TELESCRIPT technology is based on places and agents which are key concepts of it. Places are virtual locations that are occupied by agents. Agents are the providers and consumers of goods in

the electronic marketplace applications that TELESCRIPT was developed to support. Agents are software processes that can move from one place to another, for that their program and state are encoded and transmitted across a network to another place, where execution recommences. Agents are able to communicate with one-another in standard way whether they are present on same or different location.
TELESCRIPT technology consists of four components: (1) TELESCRIPT language which is designed for carrying out complex communication tasks: such as navigation,(2)Authentication control,
(3) TELESCRIPT protocol set - These protocols deal primarily with the encoding and decoding of agents, to support transportation between places.(4) Set of software tools to support the TELESCRIPT applications.

### *F. Agent Behaviour Language(ABLE)*

ABLE was developed by a group at Philips research labs in the UK to develop agents in terms of simple, rule like licences. Licences may include some representation of time: they loosely resemble behaviors in the subsumption architectures. ABLE can be compiled down to a simple digital machine realized in the 'C' programming language. The idea is similar to situated automata, though there appears to be no equivalent theoretical foundation. The result of compilation process is a very fast implementation, which has been used to control a Compact Disk-Interactive (CD-I) application. ABLE has recently been extended to a version called Real-Time ABLE (RTA).

### *G. APRIL and MAIL*

APRIL [McCabe and Clark,1995] and MAIL [Haugeneder et al., 1994] are two languages for developing multi-agent applications that were developed as part of the ESPIRIT project IMAGINE. The two languages are intended to fulfil quite different roles. APRIL was designed to provide core features which are required to realize most agent architectures and systems. It provides facilities for multitasking, communication, pattern matching and symbolic processing capabilities. The generality of APRIL comes at the expense of powerful abstractions, including plans and multi- agent plans. The language was originally envisaged as the implementation language for MAIL. The MAIL system has been used to implement several prototype multi-agent systems, including an urban traffic management scenario. [Haugeneder and Steiner, 1994]

## IV. JAVAFORINTELLIGENT AGENT

Java is an object oriented programming language which was originally designed for developing embeddedsoftware for consumer electronics initially named "OAK". Java is developed in 1995 and is a remake of Oak developed in 1991. Java is considered as one of the most suitable programming language to develop Intelligent Agents for different applications working in different domains. Characteristics of Java programming language which makes it suitable for development of intelligent agent or applications based on them – Java is "architecture neutral" which makes it ideal for the development of intelligent agents as they can be transported across a network and executed on any target computer system. [10]
Java Virtual Machine has inbuilt security mechanisms to protect against malicious or errant code.
Autonomy –In Agent application, an agent is autonomous component that can have a separate thread of control. Java supports threaded applications and provides support for autonomy.
Communication -The agent is informed about any change in the environment by sending it an event. An event is nothing more than a method or message call, in which information is passed along with the method call that defines what happened or what action agent has to perform, as well as data required to process the event. [11]
Intelligence–This property of agent can be implemented using range of hard coded procedures or object-oriented logic to sophisticated reasoning and learning capabilities. Java provides the entire basic functions needed to support these behaviors. Object Oriented – Intelligent agent basedapplications have two major aspects which are- knowledge representation and algorithms that manipulate those representations. All knowledge representations are based on the use of slots or attributes that hold information regarding some entity, and links or references to other entities. Java objects can be used to encode the data and behaviour as well as the relationships between entities. Java can easily be used to implement Artificial Intelligence knowledge representation methods such as frames, semantic nets, and if-then rules etc.
Mobility – There are several different aspects to mobility in the context of Intelligent Agents and their applications. Java's portable byte codes and Java archive (JAR) files allow groups of compiled Java classes to be sent over a network and then executed on the target machine. Java applets provide a mechanism for running Java code remotely via a Web browser.
Mobile programs must have the ability to save the state of the running process, ship it off, and then resume where the process left off, only now it is running on a different system.[12] Java inbuilt support for networking, remote invocation and serializable allows agents to freely move over the network without bothering about underlying platforms and tools uses for communication, managing its status and invoking remote agents. JavaBean delegation event model allows dynamic registration of Event Sources and Event Listeners which allows a mobile Java agent to plug into an already running server environment when it arrives, and then "unplug" itself, when it is time to move on.

## CONCLUSION

In this paper, we first discuss the Artificial Intelligence, its emergence and various applications of AI in modern world. Intelligent Agents are discussed as foremost component required for developing intelligent systems. In the section related to Intelligent Agent, we list various traits of the agent and after it different programming approaches or languages emerged for development of the agent. We also identify the limitation of each approach and finally figure out

how Java is most suitable language for the development of Intelligent Agents. We conclude this paper with remark that there is need for programming language for agent development which requires least coding and provide inbuilt support for methods of knowledge representation so that developer can focus more on functional implementation of Intelligent Agents.

**REFERENCES**

[1] McCorduck, *Artificial Intelligence in myth*, 2004,pp.4-5.
[2] Needham, *Humanoid Automata*,1986,pp.53.
[3] Russel and Norvig, Dartmouth Conference,2003, pp.17.
[4] JosephP. Bigus&Jennifer Bigus, *Constructing Intelligent Agents using java*, 2nd edition, wiley.
*[5]* Luger&Stubblefield2004,Nilsson1998,*IntelligentTraits.*
[6] Russel and Norvig,DartmouthConference,2003,pp.17.
[7] Nick M. M., *Building and Running Long-Lived Experience-Based Systems*, PhD thesis, Dept. of Computer Science, University of Kaiserslautern, Kaiserslautern, 2004.
[8] *Thomas, S. R. (1993).* PLACA, *an Agent Oriented Programming Language.* PhD thesis, Computer Science Department, Stanford University, Stanford, CA 94305. (Available as technical report STAN-CS-93-1487).
[9] Schank R.C., *Dynamic memory:a theory of reminding and learning in computers and people*. Cambridge Cambridgeshire: New YorkCambridge University Press, ISBN: 0521248582, 1982.
[10] Lind J., *Iterative software engineering for multiagent systems: theMASSIVEmethod*, vol.1994. Berlin: Springer,3540421661,2001.