IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Journal Européen des Systèmes Automatisés
Vol. 55, No. 6, December, 2022, pp. 723-739
Journal homepage: http://iieta.org/journals/jesa

Past, Present and Future Trends in Multi-Agent System Technology

Sandali Thamalka Goonatilleke*, Budditha Hettige

Department of Computer Engineering, Faculty of Computing, General Sir John Kotelawala Defence University, Ratmalana 10390, Sri Lanka

Corresponding Author Email: goonatillekemast@kdu.ac.lk

https://doi.org/10.18280/jesa.550604

Received: 27 April 2022
Accepted: 6 June 2022

Keywords:
agents, agent classification, multi-agent systems, MAS applications, MAS trends

ABSTRACT

Multi-Agent System (MAS) technology is one of the cores and promising areas in the field of Artificial Intelligence (AI) as well as in the stream of Computer Science. The technology is comprised of multiple decision-making agents that exist in an environment to achieve common or conflicting goals. Multi-Agent System technology has a rapid growth and evolution due to its marvelous features such as flexibility and intelligence that are very useful when solving complex distributed problems. This paper focuses on the history and evolution of MAS technology, present applications, and future trends by addressing more detailed explanations about the foundations or key principles of Multi-Agent System technology such as agents, agent taxonomy, agent communication approaches, MAS development frameworks as well as the history of agent technology. The goal of this paper is to provide broad and comprehensive knowledge about Multi-Agents System technology.

1. INTRODUCTION

The world we are living in at present is similar to a Wonderland that was described by one of the British mathematicians named Charles Lutwidge Dodgson in his famous novels [1]. The reason for this amazing change is due to a various number of inventions and technologies such as self-driving cars, smart speakers, image recognition techniques, etc. Moreover, it could be shortly described as the rapid development and advances in the field of artificial intelligence (AI).

The roots of the origination of the field of Artificial Intelligence are stepped back to the 1940s, especially back to 1942. This era can be defined as the first season of AI namely AI spring or birth of AI. This novel concept came to the stage with a short story named "Runaround" which was written by an American science fiction writer named Isaac Asimov. In 1950, Alan Turing published an article named "Computing Machinery and Intelligence" that describes creating intelligent machines and testing their intelligence. The test was named as Turing test which is still considered the benchmark for artificial systems. In 1956, the word Artificial Intelligence was officially coined as a result of the Dartmouth Summer Research Project on Artificial Intelligence (DSRPAI) at Dartmouth College in New Hampshire. This project was hosted by Marvin Minsky and John McCarthy who were considered the founding fathers of AI [2]. Since then, the evolution of Artificial Intelligence has been progressed step by step and classified great concepts into different subfields.

In 1975, Distributed Artificial Intelligence (DAI) was emerged as one of the subfields of Artificial Intelligence. DAI is also known as Decentralized Artificial Intelligence which is mainly based on intelligent agents and their interaction. The arisen of this concept was mostly due to the insufficiency of the application of single agents as earlier Artificial Intelligence research has been mostly based on single-agent environments [3]. Accordingly, the agent evolves in a static environment and the concept becomes impractical for the real-world applications that contain multiple agents.

In recent years, Distributed Artificial Intelligence has become a tremendous research area as it can be used to address complex computing applications such as in electronic commerce where the DAI system learns financial trading rules from a large financial data sample, and in telecommunications where the DAI system used to control cooperative resources in WLAN networks, in routing where to model vehicle flow and also in electric power systems. The entire concept and theory of DAI can be defined by addressing three main characteristics namely, a method of task distribution among agents, a method of power distribution, and a method of communication among the agents [4]. Furthermore, there are some major advantages of using DAI such as it can be easily used for complex learning methods, reasoning, large-scale planning and decision making. Not only that but also, it can be used to easily process and analyze a large amount of data and to resolve problems very quickly. In addition, the algorithms in the DAI can be classified into three sub-fields regarding the methods which are used to solve different tasks. They are Multi-Agent Systems (MAS), Distributed Problem Solving (DPS), and Parallel AI [5]. DPS involves solving a complex problem with the use of multiple distributed agents who have inter-complementary knowledge. Parallel AI is involved with designing various types of languages, architectures, and algorithms to increase the efficiency of the classical and traditional AI algorithms.

Generally, these three fields are not mutually exclusive and build upon each other to some extent. When considering MAS, it has been acquired over many years and DPS can be introduced as a subset of MAS. When considering the differences between MAS, DPS and Parallel AI they are as

follows. DPS is mainly focusing on information management aspects of the systems with several branches working together towards a common goal while MAS mainly deals with behavior management that contains a collection of several independent entities. When it is so, parallel AI mainly aims to work in a parallel mode which is very much different from MAS and DPS.

In the 1980s, Multi-Agent Systems (MAS) concept was originated from the field of Distributed Artificial Intelligence (DAI) as a novel, more-developed, promising technology [6]. MAS technology shows rapid growth due to intelligence and flexibility when solving complex distributed problems. Mainly, the quality of the concept of MAS technology depends on intelligent and autonomous entities named agents. In contrast, MAS can be defined as a network that contains a collection of individual agents who intelligently communicate and share knowledge with others to accomplish a given problem [7]. The agents in multi-agent systems contain four properties such as social ability, reactivity, autonomy, and proactivity. Accordingly, these agents perceive their environment and react to the changes intelligently by generating different behaviors from their standard set of rules. In addition, these agents interact with their neighboring agents in the model and solve the given task. Due to these advantages, MAS technology is commonly used in the field of Computer Science, Engineering, Supply Chain Management, Healthcare, and Manufacturing. Figure 1 shows the timeline regarding the evolution of the Multi-Agent Systems concept.

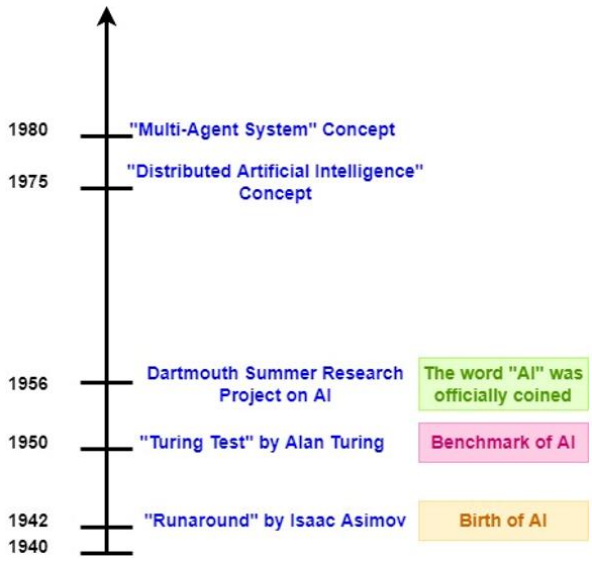

Figure 1. Timeline of the evolution of MAS concept

This paper presents a broad discussion about the multi-agent systems technology considering its concept, theory, frameworks, application domains, evaluation procedures, research challenges, past applications, current trends, and future enhancements and directions. Accordingly, we provide a detailed overview of MAS that will help researchers, and readers who are new to the area to clearly understand this broad technology. At first, we give an introduction to agents and gradually enter to describe the key concept and theory of the MAS technology while encountering related concepts such as expert systems. In the next step, we discuss different frameworks of MAS by giving a comparison of them considering types of characteristics. Then, we discuss past, present applications, and research challenges. In the next part, we discuss future trends and evaluation approaches to evaluate the performance of MAS.

The rest of the paper is organized as follows. Section II provides a detailed overview of agents and Section III describes the classification of agents. Then, Section IV gives a detailed explanation of Mult-Agent System technology while Section V describes some of the popular and latest agent development frameworks. Section VI has been focused on one of the important theories in MAS which is MAS communication approaches and Section VII gives a comparison between MAS with other related systems such as Object-Oriented Programming (OOP) and Expert Systems. As the next step, Section VIII describes MAS applications while addressing current trends and directions while Section IX gives predictions about the future directions of MAS technology. Finally, Section X concludes the paper with a discussion.

2. OVERVIEW OF AGENTS

There are different definitions for the word “Agent” as it is a universal word. Due to this reason, researchers in the field of AI have failed to introduce a better definition for the word “Agent” [7]. The agents can be existed in different types of physical forms and have vastly varied application domains. Therefore, agents have different names based on their application domain such as softbots for software agents, and taskbots for task-based agents. Russel and Norvig have defined an agent as anything that receives information from its outside environment through the sensors and perform actions through its effectors [8]. In addition, Ye et al. [9] have defined the agent as “an encapsulated computational system that is situated in some environment and that is capable of flexible, autonomous action in that environment in order to meet its design objective”. Burgin and Dodig-Crnkovic [10] have defined the agent as “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”. According to Russel’s and Norvig’s definition, the agent is mostly referred to as a physical agent while other authors' definitions are related to the virtual agent. Figure 2 shows the agent structure and behavior of the agent that has been explained by Russel and Norvig. Therefore, most of these definitions are specific and we have proposed a generic definition that can be broadly used for different fields and disciplines.

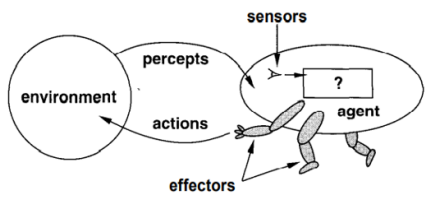

Figure 2. Russel’s and Norvig’s Agent structure and behavior

Agent: Any type of **entity** that is placed in a particular **environment** to sense different **parameters** and make a correct decision in order to perform necessary **actions** in that environment.

The definition we have stated above contains four keywords. Each keyword has been further described below.

2.1 Entity

This is the agent that can be mainly classified as a physical entity or a virtual entity. Physical entities are referred to a physical component in a physical system. A controller or protection relay which is directly used to control a power system is an example of a physical entity. A virtual entity can be introduced as an agent that operates in a virtual environment such as a computer system. Therefore, a virtual entity is a piece of computer software that receives inputs and sends outputs through a virtual environment.

2.2 Environment

This can be introduced as the place where the agent is located and the place in which the agent is sensing information to make proper decisions. The environments where the agents exist have different flavors and can be classified into different categories based on different properties.

Accessible: In an accessible environment the agent can detect the complete state of the environment using its sensors. This parameter also depicts the accuracy of the state detection. Accordingly, such kind of environment is very convenient as the agent doesn't need to maintain internal states to keep track of the world. In addition, in an inaccessible environment the agent sense noisy and incomplete data.

Deterministic: The environment is deterministic whenever the next state of the environment can be completely determined using the current state and the actions performed by the agents. Therefore, in this environment, the results will be predictable as the agent precisely knows the next action to be performed in the next state. Whenever the environment is inaccessible, then the environment may be deterministic.

Episodic: The environment is episodic when the agent's experience is divided into multiple episodes. An episode is comprised of agents perceiving and acting. These episodes are independent of other episodes. Accordingly, subsequent actions do not depend on the actions that were occurred in previous episodes. Due to this reason, episodic environments are much simpler.

Static: If the environment is changed while the agent is perceiving inputs and performing actions, then the environment is dynamic for the agent, otherwise it is static. When the environment is dynamic, the decision-making process becomes much more difficult as previously acquired data may no longer be accurate. Therefore, the agents in the dynamic environment should always sense the state of the environment and need to update sensed information. Therefore, static environments are easier to deal with as the acquired data can be used to make decisions during the lifetime of the agent.

Discrete: The environment is known as discrete when there are clearly defined distinct number of percepts and actions. Otherwise, the environment is continuous.

These five types of environments share similarities and differences. The major similarity between accessible environment, deterministic environment, and episodic environment is that the agent in those environments can observe or sometimes can't observe its environment. In addition, the difference between an accessible environment and a deterministic environment is, that when the agent exists in the accessible environment, the agent gives its full access to the complete state of the environment while when the agent exists in the deterministic environment, it doesn't have full access to the complete state and the next state is only determined by the current state. In brief, it can be described as whenever the environment is inaccessible, then the environment may be deterministic. Moreover, another similarity between all of these environments is that they could be existed as static or dynamic at different times while highlighting their unique property.

2.3 Parameters

Parameters are the different types of information that the agent sense from the environment. For instance, the parameters of a robot that plays chess are the positions of the opponents.

2.4 Actions

They are the reactions/outputs that the agent gives for the sensed information. Accordingly, an agent can perform a set of discrete or continuous actions. If the actions are continuous, the agent performs unlimited actions while the actions are discrete, it performs a discrete number of actions.

The aim of any agent is to solve a dedicated task(s) with some other constraints such as a deadline. To fulfill this aim, at first agent senses different parameters from its environment and get a piece of knowledge about the environment. When there are many agents, the agent should use the knowledge of its neighbors. Finally, the agent performs the correct and appropriate actions considering the knowledge and previous actions. The agents share four main characteristics to solve a variety number of complex tasks with a broad application capability.

Autonomy: The agents can be operated without any direct intervention of human beings. They have particular control over their internal states and actions.

Sociability: The agents have the ability to communicate with other agents like humans using an agent-communication language (ACL).

Reactivity: The agents sense the changes in the environment and can react in a timely fashion manner.

Pro-activeness: The agents depict a goal-directed behavior. Accordingly, the agent changes its behavior dynamically to achieve its desired goals successfully. In this process, the agent uses sensed parameters, data from other agents, and past actions to meet the goal effectively.

The agent senses its environment to acquire different parameters and perform necessary actions to fulfill a particular goal by sharing common characteristics. While they share similar properties, they can be broadly classified into different types.

3. CLASSIFICATION OF AGENTS

The agent topology is a wide classification with many categories. Primarily, the agents that exist in the world can be divided into three main categories as physical agents, mental agents, and structural or informational agents. Human beings, animals, and robots are the physical agents who have a physical structure and exist in the real world. Software agents which are related to computing systems and exist in a virtual environment are examples of mental agents. The head of the Turing machine is an example of a structural agent. Furthermore, the physical agents can be divided into three categories such as biological agents, artificial agents, and

hybrid agents. Living beings such as humans, animals, and microorganisms are examples of biological agents while robots are examples of artificial agents. Hybrid agents are agents who have both biological and artificial parts [10]. Figure 3 shows the general classification of agents.

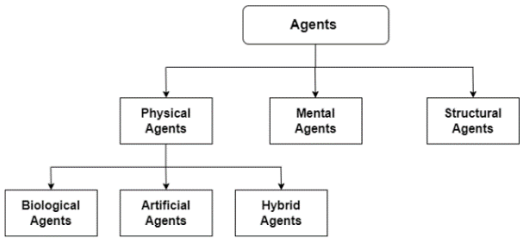

Figure 3. General classification of agents

Mizzaro [11] has classified the agents considering three parameters such as perception, reasoning, and memory. At first, he has classified the agents considering their levels of perception. Complete perceiving agents are the highest perceiving agents who have a complete perception of their environment while opposite to perceiving agents are the agents who are completely isolated from the environment. Secondly, agents have been classified considering their reasoning capability. The highest-level reasoning agents are known as omniscient agents who have high logic reasoning capabilities. These agents can derive new knowledge components using their existing knowledge while nonreasoning agents are unable to do the reasoning activities. Thirdly, there are agents with permanent memories who will never lose any small portion of their knowledge while no-memory agents are unable to maintain the knowledge state. Figure 4 shows the summary of agent classification by Stefano Mizzaro.

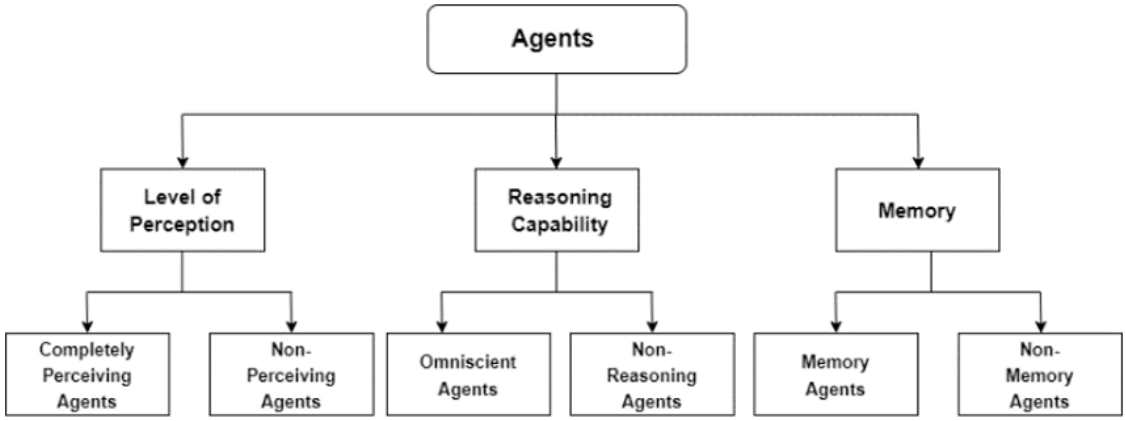

Figure 4. Summary of agent classification by Mizzaro

Agents can be categorized into six main categories based on attributive dimensions such as intelligent/cognitive agents, dynamic criterion, interaction criterion, autonomy criterion, learning criterion, and cooperation criterion. Agents can be divided into five subcategories based on intelligence or cognitive criteria. Intelligent or cognitive agents use their skills and resources to achieve their desired goal. The significance of these agents is they have extended intellectual capabilities and they are designed according to the BDI (Beliefs, Desires, Intentions) architecture [12]. The five subcategories of intelligent/cognitive agents have been briefly described below.

3.1 Simple reflex agents

They are also known as tropistic or behavioristic agents. Simple reflex agents have a fast response time and therefore, they may facilitate emergent behaviors. They sense the environment and make decisions with the use of a condition-action rules database and perform behaviors with the help of the action selection module. These agents depict very simple behaviors and the behaviors are only dependent on current percepts rather than the percept history. Figure 5 shows the structure of the simple reflex agent.

3.2 Model-based agents

Simple reflex agents are not suitable for partially observable environments and model-based agents have been designed to overcome that limitation. Model-based agents perform behaviors based not only on current percepts and also on previous percepts. These agents maintain an internal state that contains percept history and unobserved aspects of the present percepts. Figure 6 shows the structure of the model-based agent.

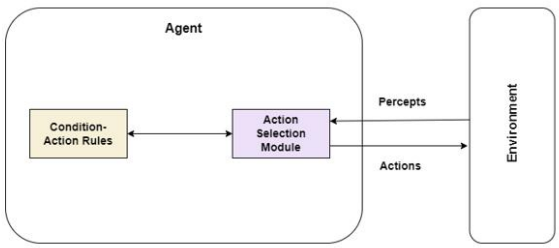

Figure 5. The structure of the simple reflex agent

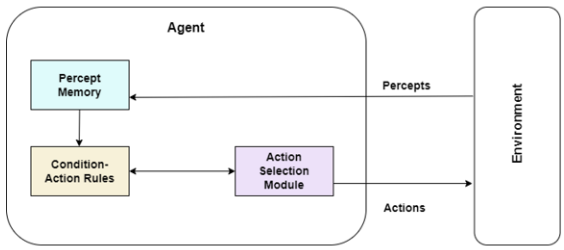

Figure 6. The structure of the model-based agent

3.3 Goal-based agents

These agents are directed by goals that are initially defined by the user. Goal-based agent receives percepts from the environment and its memory manager stores these percepts in the percept memory. Then, these percepts will be used by the action selection module to perform appropriate actions to reach the desired goal. Goal-based agents can take more elaborate decisions as they have a collection of percepts. Goal-Based agents can be mainly classified into two types based on their goal model as task-oriented model and state-oriented model. In the task-oriented model, the agents live in a task-oriented domain and the agent's goal is performing a finite set of tasks. In the state-oriented model, the agent lives in a state-oriented domain and its environment is comprised of sequential finite sent of states. The goal of the agent is a final state that it tries to reach by passing states sequentially. Figure 7 shows the structure of the goal-based agent.

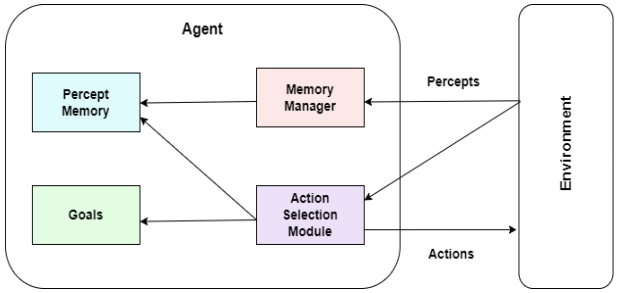

Figure 7. The structure of the goal-based agent

3.4 Utility-based agents

These agents are a little bit different from other agents as they are using a performance measurement index which is named as utility function to evaluate their behavior. Utility-based agents use a specific path that optimizes a given utility function to reach their desired goals. These agents recognize the specific goal, identify different paths to achieve the goal, and among them choose the best path to reach the goal. Figure 8 shows the structure of the utility-based agent.

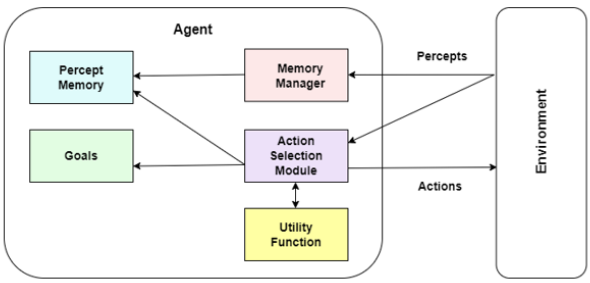

Figure 8. The structure of the utility-based agent

3.5 Learning agents

A learning agent can be a simple reflex agent, model-based agent, goal-based agent, or utility-based agent who frequently learns to reach the goal effectively. They can modify their behavior while communicating with the environment and as a result, they can perform the given task very confidently. The main advantage of a learning agent is, it can also be operated in an unknown environment and become more competent as it always updates the knowledge. Figure 9 shows the structure of the learning agent.

Agents can be categorized into two main categories based on their dynamicity. Static agents are entities that cannot move from one place to another place. Agents in a computer system are one of the best examples of static agents. Secondly, mobile agents are capable of moving to some extent of freedom and mobility can be realized on different levels. Mobile robots are ideal mobile agents. Further, these mobile agents can be classified into two groups based on the method of moving. Effector mobile agents are the agents that use effectors for their movement. The second category is receptor mobile agents who are also known as sensor mobile agents. These agents mainly use receptors for their movement [13].

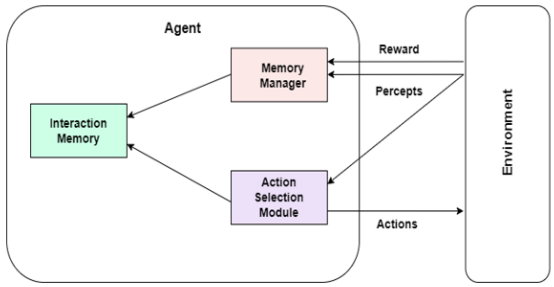

Figure 9. The structure of the learning agent

The third classification is based on the agent's interaction. Accordingly, the agents can be mainly classified into two groups such as deliberative agents and reactive agents. Deliberative agents are also known as proactive agents which have the ability to think by searching behaviors, maintaining an internal state, and predicting their actions. The significance of such an agent is that it has an explicit view of its environment and decisions are taken by logical symbolic reasoning. Therefore, these deliberative agents try to anticipate the change which is going to happen in the environment and organize their activities considering the predictions accordingly. Reactive agents are also known as sensing/acting agents who do not maintain internal states and have pre-set behaviors. Using these behaviors, they respond to the changes happening in their environment in a timely fashion manner.

Agents can be categorized into two main groups based on the autonomous criterion. They are autonomous agents and dependent agents. Autonomous agents are widely used in virtual environments such as software and also, and they have been embodied in physical environments such as an autonomous robots. According to the explicit meaning of the word "autonomous", such an agent is independent of its user or owner and capable of performing a particular task without any external help by controlling its actions and internal states. Dependent agents' characteristic prevails on the opposite side of the autonomous agents as they are dependent on their user or owner and acquire external help to perform a particular action [14].

The agents can be classified into three main groups considering the learning criterion and they are learning agents, remembering agents, and conservative agents. Learning agents are the most complex agents in the agent family and they can perform functions very well in an unknown environment. In addition, they require a very long time to improve their behavior to an intelligent level. They gain much knowledge about the environment and take complex decisions. The

second category of remembering agents is also known as memorizing agents which are the lowest level of learning. Thirdly, conservative agents do not gain knowledge about the environment and therefore, they do not learn [15].

The final classification of agents is based on the cooperation criterion. The agents in this criterion can be classified into three groups such as competitive agents, individualistic agents, and collaborative agents. The competitive agents are individual agents who have non-aligned goals and their main aim is maximizing their gains. The significance of such agents is that they do not collaborate with other agents and only compete with others to reach their desires. As the name implies, individualistic agents also like to be alone and do not interact with other agents. But they do not compete with other agents like competitive agents. Collaborative agents do not compete with other agents and strongly collaborate to reach their goals [16]. Figure 10 shows the summary of agents according to the attributive dimension criteria.

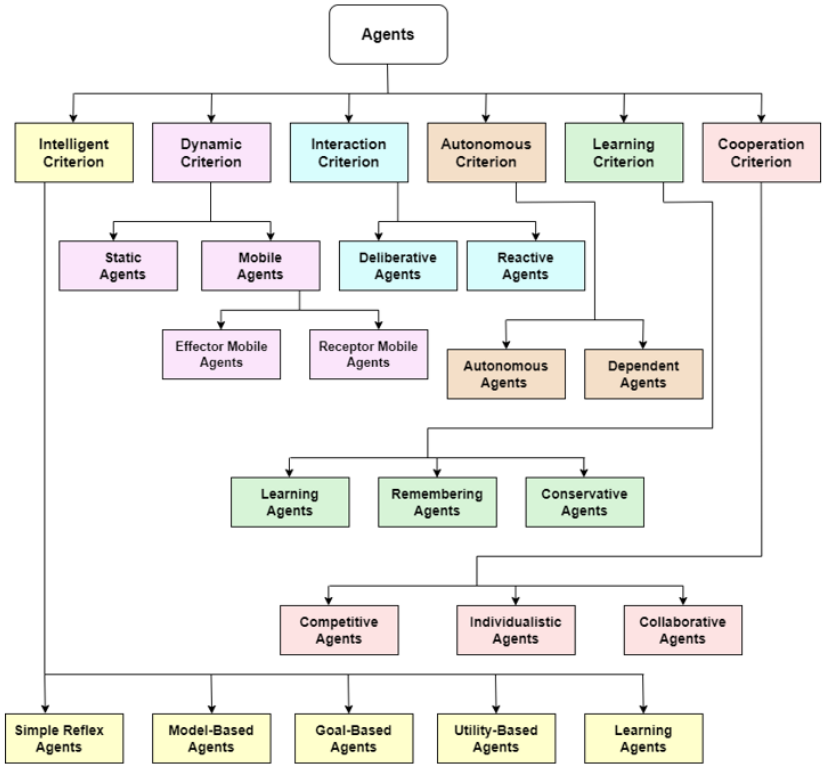

Figure 10. Summary of agents according to the attributive dimension criteria

4. MULTI-AGENT SYSTEMS (MAS)

Multi-agent systems technology is a novel and promising area in computer science that was mainly derived from the field of distributed artificial intelligence (DAI) and the extended version of agent technology. The whole theory and principle of MAS are mainly depending on and dealing with a compound of relatively intelligent and loosely connected autonomous entities named agents. These agents exist and act in a certain environment which could be dynamic, unpredictable, and open to achieving a common goal by competing or cooperating, sharing or not sharing knowledge with other agents. The core functionality of a multi-agent system is the interaction between agents. These interactions can be described as distributing organized structure that operates within agent communities and as modes of cooperation which is a combination of collaboration, coordination, negotiation, and conflict resolution between individual agents to accomplish a given goal. This interaction phenomenon in-order to solve a problem can be identified as a result of collective behavior by mainly modeling the problem as a structured set of entities that are able to act in a particular environment. Accordingly, these agents perceive their environment locally and adapt their behaviors according to the received perceptions and also possess their resources, skills, tendencies, and objectives which could be explicit or implicit, and finally communicate and cooperate with neighboring agents by formulating, and sending individual messages to fulfill the given common task [17].

The multi-agent system technology is comprised of several different characteristics which make the technology more promising and popular. Openness can be defined as one of the challenging characteristics in MAS which refers to the ability to create new agents, leaving and joining agents into the environment. However, this characteristic makes it more difficult and complex to understand and analyze the behavior of the multi-agent system. Other main characteristics of a MAS are autonomy (the capacity of control that an agent has over a result of a certain decision-making process), complexity (related to learning, reasoning, and decision-making), adaptability which describes the adjustment of agents' activities according to the changes in the environment, communication methodology among agents (inter-agent, intra-agent), mobility which describes the movements in between different platforms and environments, distribution (agents are frequently operating on different hosts which distributed over a network), security and privacy.

The multi-agent systems are rapidly becoming very popular due to four salient and outstanding features such as efficiency, flexibility, reliability, and low cost. The efficiency of a multi-

agent system is very high as the overall task or the operation is divided into smaller parts and assigned each task to a particular agent. As a result of this feature, the energy required for the processing is existed at a very low level which often results to have a low-cost solution. Flexibility can be defined as the most important promise of multi-agent technology. This feature for the multiagent system is mainly as each agent solves its dedicated task with a pre-defined knowledge. In addition, the distributed nature of problem-solving makes the multiagent system more reliable [5].

At present, Multi-Agent System technology has generated many excitements due to seven promising and important features such as leadership, mobility, topology, heterogeneity, agreement parameters, time delays, and frequency in data transmission [5]. A brief introduction of each feature is mentioned below.

4.1 Leadership

Multi-Agent System can be introduced as an agent team that contains several agents who are working together to achieve a common goal. The agent team may or may not contain a leader that defines tasks or goals to other agents in order to achieve the common global goal. The leader can be a predefined agent or an agent who has been collaboratively chosen by other agents in the system. Accordingly, the overall system can be classified into two categories considering the existence of a leader as leader-follow and leaderless. In a leader-follow system, the leader agent defines actions for the followers (i.e., other agents). Secondly, in a leaderless system, agents autonomously decide their actions based on their own goals.

4.2 Mobility

The agents in a Multi-Agent System can be classified into two categories as static agents and mobile agents considering their dynamicity. A static agent is an agent who is always located in the same position while the mobile agent is the agent who can freely move around in its environment. The significance of such a mobile agent is that it shares and uses its neighbor agent's resources and monitors them.

4.3 Topology

The word "Topology" with the means of agents, defines the structure of the network of agents. Furthermore, it describes the way the agents are interconnected with their neighbor agents and their locations. Therefore, the Multi-Agent Systems can be divided into two sections based on the topology such as static and dynamic. If the topology is static, the position and the connection of the agents remain unchanged over the lifetime of agents. The dynamic topology is also known as switching topology and in this topology, the position and connection of the agents' changes. Accordingly, in the dynamic topology agents establish new connections or communications, move, join, and leave the Multi-Agent System.

4.4 Heterogeneity

Homogeneity can be identified as one of the important concepts that relate to the uniformity of a substance. Considering the heterogeneity of the agents in an agent system, the multi-agent systems can be mainly divided into two categories, homogeneous multi-agent systems and heterogeneous multi-agent systems. The homogeneous MAS contains agents who have the same characteristics and functionalities while heterogeneous MAS consists of agents with diverse features as they may comprise many subagents which are not uniform throughout.

4.5 Agreement parameters

Agreement parameters are matrices that should be agreed upon by the agents in a multi-agent system for certain types of applications. According to the number of matrices, the MAS can be categorized as first-order MAS, second-order MAS, and higher-order MAS. If the multi-agent system is first order, the agents of that system agree on one parameter or metric. As the name implies, the agents in the second-order MAS agree on two parameters. Finally, the high-order MAS agrees on more than two metrics.

4.6 Time delays

Time delays are a common factor in any system. Similarly, in a multi-agent system, agents will have delays due to different sources while they perform tasks. Generally, these delays can be occurred in a multi-agent system due to the problems in communication media, performance drawbacks in the agent communication approach, and also when allocating resources for agents. Moreover, considering those delay types, the multi-agent system can be simply classified as delay MAS and non-delay MAS.

4.7 Frequency in data transmission

The agents in a multi-agent system always perceive their environment, collect sensed data and share these data with other agents. Accordingly, in real-time multi-agent systems, the frequency of data transmission between the agents in the network is inevitably constrained. There are different methods to reduce the communication costs which take place in communication among agents. Therefore, the multi-agent systems can be divided into two categories such as event-triggered MAS and time-triggered MAS. When the MAS is event-triggered, the agent only perceives its environment when a certain event occurs while in time-triggered MAS, the agents perceive the environment in fixed periods. The summary of MAS feature categorization is shown in the Table 1.

Table 1. Summary of MAS feature categorization

Feature	Categories
Leadership	Leader-follow Leaderless
Mobility	Static Mobile
Topology	Static Dynamic
Heterogeneity	Heterogeneous Homogeneous
Agreement Parameters	First-order Second-order High-order
Time Delay	Delay MAS Non-delay MAS
Data Transmission Frequency	Time-triggered Event-triggered

5. MULTI-AGENT SYSTEMS DEVELOPMENT FRAMEWORKS

Multi-agent system can be generally identified as an agent-oriented system that can be treated as a first-class abstraction. Moreover, it can be identified with a programming perspective such as Agent-Oriented Programming (AOP). In an agent-based programming language, agents can be defined as the building blocks and programs that can be derived from agents' behaviors, their goals, and their interoperations.

The concept of Agent-Oriented Programming first emerged in the year of 1993 as a specialization of Object-Oriented Programming (OOP). Gradually, different types of reasoning and cognitive models have been designed for agent-oriented programming. Mainly, there are three models which were used in the past as well as at present that laid the foundation to design different agent programming languages. They are the Procedural Reasoning System (PRS), Belief-Desire-Intention (BDI) model, and Situation Calculus. Among these three models, the BDI model is the most common and widely used [18].

There are many types of multi-agent system development frameworks that have been designed considering and following the above-mentioned three main models and other models. In addition, most of these frameworks have been implemented using Java by gaining the advantage of its Java Virtual Machine. This section is going to discuss some of the latest and most popular frameworks which have been used over the past five years (2015 – 2020). They are ASTRA, Chromar, GOAL, Jason, Gwendolen, JADE, JADEL, Jadex, LightJason, JaCaMo, MaSMT, PLACE. A brief description of these frameworks has been given below.

5.1 ASTRA

ASTRA can be introduced as a new version of AgentSpeak language and contains a number of extensions to the traditional AgentSpeak language. AgentSpeak(L) or more popularly AgentSpeak is a type of agent-oriented programming language that is based on logic programming and follows the BDI architecture for autonomous agents. Similarly, ASTRA also shares the same features and is based on JAVA [19].

5.2 Chromar

Chromar is a rule-based notation that consists of stochastic semantics that are based on a continuous-time Markov chain (CTMC). Chromar contains objects named as agents and they have different attributes which are defined at the type level. Chromar is embedded in Haskel which can better provide an increased expressive power [20].

5.3 GOAL

GOAL is one of the agent programming languages that follow an agent-oriented programming paradigm. Moreover, it can be defined as a logic-based or rule-based agent programming language which is designed to handle cognitive agents. Accordingly, the philosophy of GOAL is that create programs by writing rules that can be chosen for each situation. In addition, these rules are set which facilitates acquiring a priority on what needs to be done first by an agent. Using this language can design agents that can maintain a mental state which comprised of goals and beliefs and also these agents are capable of performing actions using their beliefs and goals. GOAL agents have a knowledge base to represent conceptual and domain knowledge and use Prolog to represent the knowledge, beliefs, and goals of an agent. Unlike other agent programming frameworks, GOAL provides facilities to develop high-level strategies for agents and this feature can be identified as one of the strengths of GOAL.

The design of the GOAL is comprised of two parts as basic reasoning cycle and the modular programming section. The basic reasoning cycle has two phases and the first phase is used to process all the events including messages and precepts by updating the agent's mental state. Accordingly, in the first phase, the agent acquires and processes all information sensed from its outside environment. The second phase has been allocated for decision-making activities and the agent will decide the action which is needed to do next. Finally, after the completion of the second phase, the cycle will be repeated. The modular programming section is considered as the main programming construct in GOAL that is constructed to structure and write very large agent programs. Not only that but also, GOAL supports communication between agents and these communications can be either agent-to-agent communication or broadcasting communication [21].

5.4 Jason

Jason can be introduced as an open-source interpreter for an extended version of AgentSpeak which was developed by Jomi F. Hubner and Rafael H. Bordini. Moreover, it is a logic-based agent-oriented programming language written in Java that facilitates the developers to design and build complex multi-agent systems that are capable of working in such environments previously considered too unpredictable for computers to handle. One of the advantages of this framework is that it can be easily customized and is very suitable to implement reactive planning systems through BDI architecture. Furthermore, Jason is available Open Source and is distributed under GNU LGPL [22].

5.5 Gwendolen

Gwendolen was initially started as a small subset of Jason in-order to develop verifiable agent programs. However, at present, it has been grown up with its syntax and semantics. Moreover, it is designed to exhibit many typical features of BDI languages. The programs in Gwendolen are presented as a library of plans and those plans will be enabled when such an agent has beliefs and goals to attain a goal. Accordingly, agents in Gwendolen distinguish two types of goals such as achievement goals and performance goals. Achievement goals make statements about beliefs the agent wishes to hold and remain those goals until the agent acquires a proper belief. Secondly, perform goals simply introduce a sequence of deeds to be performed and cease to be a goal quickly when that sequence is completed [23].

5.6 JADE

JADE or Java Agent Development Framework is an open-source software framework that was fully implemented using the JAVA language. The JADE framework was distributed by Telecom Italia which is a copyright holder, under the terms and conditions of the LGPL (Lesser General Public License) version 2 license. This framework is used to simplify the implementation of multi-agent systems with the help of a middle-ware that complies with FIPA specifications and also

through a set of graphical tools that help at the debugging and deployment phases. In addition, JADE-based systems can be distributed across machines that do not have the same operating system and the configuration can be controlled via a remote graphical user interface. The JADE software model can be divided into two key categories such as agents who are also named as peers and secondly, as services that can be introduced as non-autonomous components, which can run on a single node or cooperatively on multiple nodes. In addition, these services can be triggered by the peers in the system. The peers in the JADE environment can be appeared and disappeared in the system considering the needs of the application and requirements. In addition, these agents are fully distributed across both wireless and wired networks. The communication strategy between these agents is completely symmetric and every single peer can play both initiator and responder roles.

JADE contains libraries which are the Java classes that are needed to develop agent applications and the run-time environment in-order to offer basic services. Besides, each instance of the JADE run-time is known as a container as it comprises agent(s) and a set of containers is named as a platform which provides a homogenous layer to hide the diversity and the complexity of tiers such as operating system, hardware, network, and the JVM.

JADE is comprised of very important driving principles which are mentioned below.

- Interoperability – Due to FIPA specifications, the peers (agents) in the JADE can interoperate with other peers sharing the same standard.
- Uniformity and portability – JADE has a homogenous set of APIs which are independent of its network and provides the same APIs for the JEE, JSE, and JME environments.
- Easy to use – The complexity of the middleware is hidden behind a simple and intuitive set of APIs.
- Pay-as-you-go philosophy – The programmers do not need to use every feature provided by the middleware and also if the programmer didn't use any feature, they do not require to know anything about them.

Compared with other MAS development frameworks JADE has become the most comprehensive and most popular FIPA-complaint agent platform due to various reasons. The main reason is documentation of JADE APIs and a wide variety of extensions are clear and exhaustive. This reason will mainly lead to developing various agent applications such as smart emergency applications and localization [24].

5.7 JADEL

Although JADE is considered the most popular and comprehensive MAS development framework, there are some major drawbacks and issues with this framework. At first, one of the main problems is the complexity of the JADE framework as it highly requires not only expert knowledge in the area of MAS and also deep knowledge to understand the JADE mechanism. Secondly, the lack of a fixed agent model leads to cause unclear and incorrect utilization of different agent technologies. At last, sometimes the procedures and patterns of JADE are repetitive, hence not clear considering the concepts of AOP due to the gap between the AOP paradigm and OOP paradigm. As a solution for these mentioned problems, JADEL came to the stage and it can be introduced as a novel programming language that supports development agents and Multi-Agent Systems using JADE and without directly using Java. Moreover, it can be further identified as an extension of JADE.

JADEL is an agent-oriented DSL (Domain Specific Language) that mainly aims to reduce the complexity of JADE agents and multi-agent systems to make the model more clear and visible by avoiding technical and implementation details. Accordingly, JADE provides a new syntax which is completely relying on agent abstractions and this syntax will lead to creating expressions and constructs to write very simple and clear agent programs. This framework is mainly developed with the Xtext framework. The main reason to endorse this Xtext framework is that it provides strong integration with the JVM and this will cause to have a more specific language named Xtend, a dialect of Java that can be used as a host language of the DSL. Accordingly, the advantage of using this specific language is that the JADEL programs generate an easily readable Java source code that runs fast. Due to these reasons, JADEL is much more straightforward for Java and JADE users [25].

5.8 Jadex

Jadex is a BDI reasoning engine that allows to create intelligent agents and multi-agent systems using Java and XML. This framework is very flexible compared with others and can be used on top of different middleware infrastructures. Jadex framework provides some promising features such as multiple interaction styles, a runtime infrastructure for the agents, an extensive runtime tool suite, simulation facilities, and automatic overlay network formation [26].

5.9 LightJason

LightJason is a concurrent BDI model multi-agent system framework that supports to create of multi-agent systems with Java. This framework has been inspired by AgentSpeak and Jason frameworks. LightJason which is a newly designed software paradigm provides novel concurrent semantics for the agent development as well as a very efficient implementation of agent perception compared with Jason. Moreover, this framework caters considerable extensions to the AgentSpeak which is a logic programming language such as parallel executions, thread-safe variables, muli plans, multi-variable assignments, lambda expressions, and explicit repair actions. Not only that but also, LightJason offers a modular runtime system [27].

5.10 JaCaMo

JaCaMo framework has been derived from a specific programming model called JaCa. JaCaMo is a multi-agent system development framework that programs agent organizations in Moise and programs autonomous agents in Jason. Furthermore, this framework is working in shared distributed artifact-based environments which are programmed in CArtAgo. Accordingly, JaCaMo deals with these three platforms and as a result, this framework has its own set of programming abstractions as well as its reference programming model and the meta-model. Due to this significant feature, JaCaMo is considered as a keystone to define the global programming meta-model [28].

5.11 MaSMT

MaSMT is a free and lightweight multi-agent system development framework that is fully implemented through the Java environment [29]. This framework mainly contains three types of agents named ordinary agent, manager agent, and root agent. The root agent is capable of handling a group of manager agents and the manager agent is capable of handling a set of ordinary agents. This framework also supports different types of agent communication through its message header. MaSMT was originally designed for the purpose of machine translation through communication among different types of language agents [30].

5.12 PLACE

PLACE or Planning-based Language for Agents and Computational Environments is an agent-oriented programming language that contains a syntactic structure that is very much closer to the BDI model. Some of the frameworks which are using the BDI model made different assumptions and sometimes unrealistic to develop real-world applications. As a result, the PLACE framework attempts to fill the above-mentioned limitation by following a look-ahead planning-based approach. In that approach, the actions of the agents are durative, priorities have been assigned to goals and plans have been repaired. In addition, Hierarchical Task Network (HTN) planning techniques have been used to do the plan synthesis of the PLACE framework. This framework uses a proactive-reactive plan merging algorithm in-order to streamline the tasks properly. The important point is that the actions in the existing plan can be executed in a parallel manner and without postponing higher priority tasks. Furthermore, the agents in this framework can constantly monitor the execution plans without causing any failure to the original plan when sudden unexpected changes happen in the environment. Not only that but also, this framework is easier for the users to monitor the agent's executions as their environments have been modeled visually. Finally, this framework offers a notable difference as it presents a new language with its syntax and semantics though it provides Java APIs for the programming environment [31].

Table 2 shows the summary of the Agent Programming Languages (APL) or frameworks respective to their type of the model and implementation language.

Table 2. Summary of agent programming languages and frameworks

APL/Framework	Model	Implementation Language
ASTRA	BDI	Java
Chromar	Rule-based	Haskell
GOAL	Rule-based	Java
Jason	BDI	Java
Gwendolen	BDI	Java
JADE	FIPA	Java
JADEL	DSL	Java/JADE
Jadex	BDI and OOP	Java
LightJason	BDI	Java
JaCaMo	BDI	Java
MaSMT	FIPA	Java
PLACE	BDI, HTN	

There are many different APLs and their differences mainly exist with respect to the model or architecture they are using such as BDI, DSL, FIPA, rule-based and also respect to the implementation language. Not only that but also, most of these languages are extended versions of the pre-existing languages. For example, ASTRA is the new version of AgentSpeak language that shares many functions similar to the AgentSpeak and the difference between them is ASTRA contains a many number of extensions and additional features to the traditional AgentSpeak language. In addition, both of them are following the same BDI architecture. However, comparing ASTRA/AgentSpeak language with Chromar, the main difference between them is the model or the architecture as ASTRA/AgentSpeak follows the BDI architecture while Chromar follows a rule-based architecture. Whilst, these agent programming languages will be different from each other due to their programming language. For example, compared with other APLs GOAL has a particular knowledge base to represent conceptual and domain knowledge and uses Prolog as the programming language. More importantly, unlike other agent programming languages/frameworks, GOAL provides facilities to develop high-level strategies for agents and this feature can be identified as one of the strengths of GOAL. Moreover, Jason can be identified as an extended version of AgentSpeak as same as ASTRA but the major difference among them is Jason is an open-source platform. In addition, it can be identified as a more enhanced APL and facilitates developers to design and build complex multi-agent systems that are capable of working in such environments previously considered too unpredictable for computers to handle and this can be noted as one of the strengths of Jason.

6. AGENT COMMUNICATION APPROACHES

Multi-Agent Systems perform distributive problem solving with the involvement of agents by coordinating their actions. As a result, agent communication takes place with individual agents to interact using cooperation and by information sharing. Moreover, the communication can be classified by considering the mechanist manner such as via the sendee-addressee link and considering the nature of the medium. Therefore, the communication can be point-to-point, broadcast or mediated. If the communication is point-to-point, an agent can directly communicate with another agent while if the communication is broadcast, an agent sends some information to a group of agents. Thirdly, when the communication is mediated, the communication between two agents is mediated by a third party such as facilitators.

Apart from the discussed basic communication methods, there are three common and widely used communication approaches that have been studied for over 50 years such as speech act, blackboard method and message passing method. A brief introduction of each approach is given below.

6.1 Speech act approach

Speech act communication approach was first introduced by a British philosopher named J.L Austin in 1975 in his well-known book of "How to do things with words". Generally, the speech act communication consider the language as a sort of action rather than a medium that convey and express. Furthermore, he named several utterance verbs or sentences as speech acts that change the physical environment. Accordingly, using this approach, an agent can act as a speaker who produces utterances to change the beliefs of the hearer. In the year of 1990, SAT was used as one of the design tools in

AI to establish and maintain inter-agent communication. However, in most research work, SAT has been combined with agent communication and used to design and develop agent communication languages. Accordingly, there are two agent communication languages that are based SAT. Among them, one language was proposed by the Foundation for Intelligent Physical Agents (FIPA-ACL) while the other language was named Knowledge Query Manipulation Language (KQML) which is a standard agent communication language. In 2008, a general agent automated negotiation protocol was presented that was based on SAT. More recently, in the year of 2017, an approach named ACMICS was discovered and presented to simulate the communication between agents in a crown simulation system. Accordingly, this new approach uses a message structure that was based on SAT [32].

6.2 Message passing approach

Message passing method facilitates agents to directly communicate with other agents using broadcast and point-to-point methods. For an example, if there are two agents namely agent A and agent B, agent A can directly talk to agent B if he knows the address of agent B. When it comes to the broadcast method, agent A sends the message to all its neighbors. In addition, to maintain the message interpretability, the agents must use an agreed structure when maintaining the communication. In one of the recent works, researchers named Ermon, Gomes and Selman have modified an existing message-passing approach to multi-agent gaussian inference for dynamic processes [33].

6.3 Blackboard approach

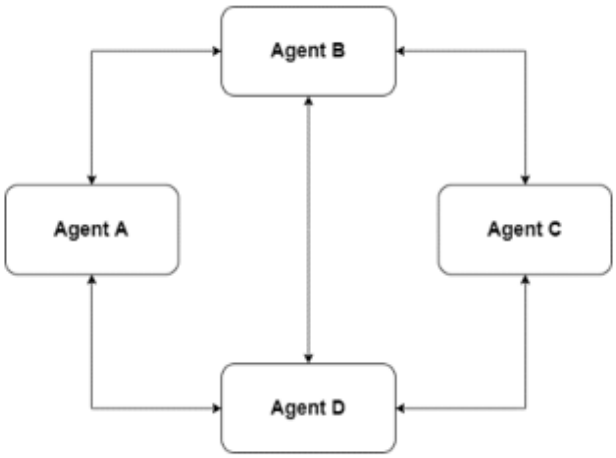

Figure 11. Message passing approach

This approach is much different and special from other approaches as all the agents collaboratively share data with other agents using a central repository which is named Blackboard. Accordingly, each agent can save its data on the blackboard and other agents can read the data. Moreover, in-order to control the access of the agents, the blackboard approach uses a control knowledge and agents can read multiple data which are defined in the control knowledge. Recently, the blackboard approach has been modernized with the involvement of Bayesian machine learning settings, using agents to add and remove Bayesian network nodes. Secondly, the blackboard approach has been used to construct large-scale intelligent systems [34]. Figures 11 and 12 shows the message-passing approach and blackboard approach respectively.

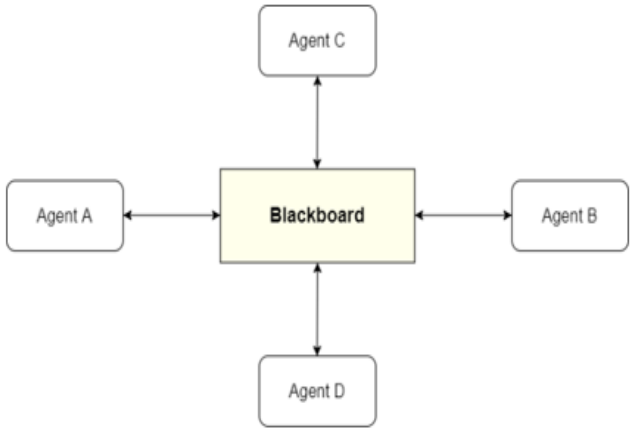

Figure 12. Blackboard approach

7. MULTI-AGENT SYSTEMS VS. OTHER RELATED SYSTEMS

Multi-Agent Systems (MAS), Expert Systems, and Object-Oriented Programming (OOP) almost seem to be similar concepts as they are all involved with decision making and knowledge sharing. When considering an expert system, it senses the environment, learns the knowledge, and finally makes a decision to solve a particular task. Unlike MAS has the capability of agent-agent communication, the expert system can only communicate and exchange data with pre-defined entities. In addition, there is a significant difference between the decision-making process between MAS and expert systems. The final decision of an expert system as well as in MAS is based on the perceived data that are sensed from its environment. However, according to the decision, the expert system advises the controller to perform relevant action, and sometimes the controller can reject the given decision as it is a separate system while according to the final decision, the agents in the MAS directly act on the environment [35].

Secondly, In OOP, an object (object z) can share its knowledge and resources with pre-defined objects by creating a public function. When the other objects need to communicate with that object z, they should invoke the created public function. Accordingly, when the object z permits other objects to use its functions, the object z will be unable to control the frequency of its function. Whilst in MAS, an agent can communicate with any node in the network as well as can control the frequency although other agents request their resources. In OOP, the objects have a limited number of pre-defined inputs while in MAS agents use multiple inputs. Table 3 shows the comparison between MAS, OOP, and Expert Systems.

Table 3. Summary of Key Differences between MAS, OOP and Expert Systems

Parameter	MAS	OOP	Expert System
Decision Making	Inputs, Knowledge, Goal	Inputs	Inputs, Knowledge
Autonomy	Sense, Make decisions, Act on the environment autonomously	Objects with pre-defined actions and inputs	Sense and make decisions
Communication Strategy	With agents	With Objects	With entities
Action Performance in the Environment	Agent behave according to the decision	Perform actions according to the pre-defined actions	The controller performs actions according to the given decision

8. MULTI-AGENT SYSTEMS WITH PRESENT DIRECTIONS

Artificial intelligence and its branches such as Multi-Agent System technology, Natural Language Processing, Machine Learning, and Robotics are progressing rapidly into diverse areas in modern society. Accordingly, they have become inseparable areas with people in the modern world. Among those novel technological areas, the Multi-Agent System technology provides a powerful platform for modeling and solving real-world problems very efficiently and effectively. This section provides MAS applications based on broad disciplines which are currently encountered in the modern world.

8.1 Agents for power engineering applications

Power Engineering or power system engineering is one of the sub-fields which has been developed within Electrical Engineering that mainly deals with the generation, transmission, distribution, and utilization of electric power. MAS technology is being used as a novel technology to address challenges in power engineering by addressing a large range of applications such as condition monitoring, diagnostics, power system restoration, market simulation, network control, and automation. Moreover, these applications can be further classified into four categories considering overall applications and aspects.

8.1.1 Monitoring and diagnostics

Multi-Agent System technology is widely used for the management and interpretation of data of power engineering and diagnostic functions. This is mainly because MAS technology is one of the best tools to collect and manipulate distributed information and knowledge. As an example, can consider transformers as power plant items and can use different types of sensors to monitor those transformers such as UHF monitoring and acoustic monitoring of partial discharge and also online dissolved gas in oil measurement. In such situations, the MAS technology can be applied and the agents of the system can monitor the output of the UHF sensors and accordingly can inform the engineer about the detected significant partial discharge activity. Moreover, can consider some promising properties cater by MAS technology such as flexibility of the system and extensible MAS architecture that allows integration of available diagnostic data, information, knowledge, and the introduction of new sensors and interpretation algorithms seamlessly to the system respectively [36].

Using these facts, some developers have designed and developed condition monitoring systems for transformers using MAS technology. Accordingly, McArthur, Strachan, and Jahn have developed a multi-agent transformer condition monitoring system to employ data generated by the UHF monitoring of partial discharge activity. Furthermore, it describes the rationale behind the MAS techniques and the problems overcome through the technology. Not only that but also they give a detailed explanation about the design and performance of the intelligent interpretation techniques [37].

8.1.2 Distributed control

The operation of modern power systems has been become very complex due to the introduction of distributed power generation, market operations, the complexity of distributed networks, and their interconnections. However, at present, MAS technology is being used to control these modern power systems very flexibly by ignoring the past solution/approach which was the central SCADA systems and smaller distributed SCADA systems. As a result, intelligent agents have been allocated to distribute the management and control functions.

8.1.3 Modelling and simulation

Modeling and simulation using traditional methods have been become challenging tasks due to the complex operations of modern power systems.

Marketplace simulation is one of the applications that uses intelligent agents to represent autonomous actions. Therefore, this method is used beneficially for the energy markets, energy networks, complex power systems, and energy utilization. Marketplace simulation is mainly used to support decision-makers. It goes beyond traditional concepts based on analytical approaches, such as the Bass model by modeling market behavior and innovation diffusion from a micro-level perspective thereby taking into consideration heterogeneous market players such as producers, intermediaries, or consumers and their (inter)actions such as word-of-mouth communication. Accordingly, these stakeholders are represented as agents with individual preferences, knowledge (beliefs), and behaviors. Therefore, innovation diffusion emerges in the course of the simulated period during which agents act and react.

8.1.4 Protection

The protection aspect of the field of power system engineering is the connection between agents and protective devices. Accordingly, most of the systems have used protection relays and other associated equipment as the agents and all the relevant functionalities have been augmented. In addition, at present many novel protection schemes for fault-tolerant and self-coordinating are being developed by the researchers.

8.2 Agents for smart grids

A smart grid concept is a novel form of the power grid that mainly aims to upgrade the existing power system

infrastructure into an efficient, intelligent, and robust electricity grid. Furthermore, it is a combination of information technology, communication technology, and power system engineering. As an intelligent electrical network, a smart grid is deployed to enhance the reliability, security, efficiency, flexibility, and sustainability of the network by making it controllable, automated, integrated, and observable. In addition, the grid has a digital structure that is comprised of different types of sensors and two-way communication [38].

The smart grid is a combination of software and hardware protocols in-order to exchange status and control signals. Recently, this task was achieved by using SCADA (Supervisory Control and Data Acquisition) system and at present, it has been moved to mixed with MAS technology. Mainly, MAS technology is used in the area of smart grids as a development tool that helps designers to create sophisticated supervisory and control applications. Therefore, intelligent agents in a smart grid can be classified into two groups as one group embodies the energy management, market the energy, price and schedule the energy while the other group is responsible for efficiency, security, reliability, and fault-handling. Besides, these agents have been designated according to the function they are performing at the smart grid such as consumer agents, device agents, distributed resource agents, intelligent response control agents, intelligent prevention control agents, and graphical user interface (GUI) agents. Moreover, unified-energy agents can be named as the base agents in the near future of advanced smart grids.

Multi-Agent technology is used to control the smart grid and also to meet the technological requirements. In addition, it also helps for information processing, analyzing electricity consumer behavior, market integration, and agent-oriented decision support. Considering those advantages offered by the agent technology, many researchers have applied the technology for the smart grids and several applications have been discussed below. Rahman and others have used agent technology to improve the transient stability of smart grids in-order to avoid the loss of synchronous operation in a power system [39]. Jin et al. [40] have done an analysis using cloud computing to develop smart grids. Accordingly, they have used the power of agent technology to control the nodes of the network with the use of the cloud architecture of the smart grid.

Moreover, MAS technology can be applied to improve the performance of the smart grid significantly. This has been further discussed in detail and reported in ref. [41]. Table 4 represents a comparison between grip operation via a conventional mode and using MA technology. According to the comparison, it can be observed that the variability in power consumption and generation in the grid that uses MAS technology is very much low.

Table 4. Grid operation comparison between different modes

Smart Grid Operation	Peak Power Consumption (W)	Peak Power Generation (W)
Conventional	5500	7000
MAS	850	800

The novel and the enhanced smart grid concept is known as the “soft grid” that uses software to coordinate the control of the grid. Dillon and others have mixed the power of software technology with web technology to design the architecture of the grid. Moreover, this approach has been helped to overcome the limitations of the existing web-architecture-based grids. Not only that but also Kamdar and others have introduced a LABVIEW software-based MA approach to control and restore the grid during faulty and outage events [42].

8.3 Agents for healthcare

The Healthcare industry is one of the most rapidly growing sectors in any country in the world. At present, many countries even the developed nations are also facing different types of challenges due to the rapid growth of the population. Accordingly, the demand for health services is also increasing which results in a huge challenge. However, Multi-Agent System technology provides a more convenient powerful platform to model and solve existing real-world healthcare problems. Generally, two major specific technologies can be used to model healthcare problems with the involvement of agents such as Wireless Sensor Networks (WSN) and Body Area Sensor Networks (BASN) [43].

A wireless sensor network can be identified as a collection of nodes that are organized into a cooperate network where each node is capable of sensing, processing, and communicating. Wireless sensor networks share two distinct properties such as agents being homogenous and numerous. Therefore, the agents or nodes in the network are the same or share the same hardware, software, and ability as well as they make more sense with real-world applications respectively. Secondly, the body area sensor network (BASN) is a special type of wireless sensor network that gathers information about the human body using sensors as its main goal. Accordingly, BASN also shares the same properties as wireless sensor networks and also some distinguished properties such as heterogeneity, a smaller number of agents, and the quality of the wireless signal.

Healthcare applications can be classified into three groups considering two major specific technologies such as telemedicine, daily living, and monitoring, detection, and assistance.

8.3.1 Telemedicine

Telemedicine mainly facilitates monitoring the patient remotely and also establishing and maintaining real-time communication with the physician. Accordingly, sensor nodes have been attached to the body of the patient and provide a facility to monitor different biological properties. In addition, these properties are available in real-time and this results from the physician communicating directly with the patient. This communication can be established and continued over various mediums such as phone calls, emails, and video conferencing technology. Many applications have been designed and developed considering this novel area.

8.3.2 Daily living and monitoring

Applications in this area mainly use wearable and implantable sensors to monitor the vital signs of the patients unobtrusively. The significance of such systems is that they have been designed to store very important historical records of the patients, provide insight, and report abnormalities. Most of the medical cases such as brain tumors, cancer, diabetes, and different physical disabilities have been benefited from these applications.

8.3.3 Detection and assistance

This category is similar to the daily living and monitoring applications with additional functionality such as providing assistance for the patients when deemed necessary.

At present, there are many existing healthcare applications that were mixed with MAS technology. Therefore, many of these applications focus on heart rate monitoring using ECG, pulse oximetry, fall detections, gait analysis, and Parkinson's episode detection. Camarinha-Matos et al. [44] have designed an application named TeleCARE which is a configurable framework focused on virtual communities for elderly support. Therefore, virtual communities help elder people to stay at home and to maintain their typical lifestyles. Moreover, this project has been mixed with MAS technology as the base infrastructure instead of using TCP/IP over the Internet due to two major reasons. The first reason is to provide real-time responses by continuously providing services by reducing dependencies and delays of the network. The second reason is to build mobile agents and to send them for remote executions in-order to achieve high flexibility and scalability.

A unified body sensor network is a combination of two technologies such as Wireless Sensor Networks (WSN) and Body Area Sensor Networks (BASN). One of the applications of the Unified body sensor network is MADIP which stands for Multi-Agent Distributed Information Platform. Furthermore, it is a mobile agent-based distributed information platform for pervasive health care monitoring that was designed using JADE. MADIP is capable of automatically notifying the responsible care provider, offering distance medical advice, and performing continuous health monitoring. Not only that but also, this application is well suited for daily living and monitoring areas as it provides a well detection and assistance for elder people. This system contains six types of agents such as user agent, resource agent, physician agent, diagnostic agent, knowledge-based data server agent and agents for external services. User agents act as intelligent gateway interfaces for patients and physicians and execute according to the user requests. Secondly, the resource agent mediates agents' access to resources in the system and operates at a higher level of trust while the physician agent is used by medical staff to perform tasks as a mobile agent. Then, the diagnostic agent can be considered as a data-analysis engine that is a static agent and it mainly analyzes the data of the patients and indicates/predict the sudden change of the patients' status while the Knowledge-based data server agent contains two information repositories such as user status and profiles and used to store physiological information collected by the physician agent.

8.4 Agents in robotics

Research regarding multi-agent-based robot systems has been started conducted in the 80s to provide more robust and efficient robotic systems. Moreover, it has been started to study formally with the aid of the first article which was published in 1996, specially outlined the advantages and disadvantages of agents in robotics. As a result of such kind of research, two heterogeneous multi-agent robots named ALLIANCE and ACTRESS have been developed. In addition, different cooperative multi-agent robot applications such as soccer robots, unmanned guided vehicles (UGVs), and unmanned aerial vehicles (UAVs) were developed gradually [45].

According to Ismail et al. [46], cooperative multi-agent robotic systems have focused on three main elements as a type of agent such as homogenous or heterogenous, the control architecture such as reactive, deliberative, and hybrid, and finally the type of communication such as implicit and explicit. Accordingly, to facilitate efficient coordination among those agent-based robots, the control architecture and the communication type must process a coherent behavior with the agents.

Robots and robotic systems are widely used in the field of agriculture for four decades. Arguenon et al. [47] have developed multi-agent-based agriculture robots which are employed in the harvesting of a vineyard. Accordingly, this system contains three types of robots such as harvesting robots, small transport robots, and large transport robots that have specific tasks to accomplish. The harvesting robot is responsible for harvesting and waiting for a transportation robot while the large transportation robots are responsible for three roles such as moving to one of the small transport robots, moving to the processing center, and waiting for the relevant instructions. Then the small transportation robot also has three roles such as move to one of the large transport robots, move to one of the harvesting robots, and also wait for instructions. Therefore, all of these communications are done with the aid of multi-agent system technology that follows ORIS platform.

Duan et al. [48] have proposed an agent-based soccer robot to study the complexity of their decision-making capability. Accordingly, in this system, the robots are grouped into multiple teams. Accordingly, the agents learn knowledge about the opponent team by perceiving their environments as well as sharing those learned knowledge with other agents in the team. In addition, reinforcement learning has been used with probabilistic neural networks to increase the accuracy of the decision-making capability of the agents.

In addition to those applications, MAS technology has been commonly applied for agricultural tasks and applications these days mainly to model and automate agricultural tasks. Accordingly, the technology is used to develop simulation models. Goonatilleke et al. [49] have developed a system to establish and maintain the communication platform between the key persons in the rice production industry using the power of MAS technology. Accordingly, these key persons such as farmers, buyers and transporters act as agents and communicate effectively mainly to reduce transport costs.

9. FUTURE DIRECTIONS OF MULTI-AGENT SYSTEM TECHNOLOGY

Future trends and directions of Multi-Agent System technology can be guessed to be mixed with some of the disciplines such as Machine Learning (ML), Complex Adaptive Systems, Robotics and Parallel Computing. When considering the current world and the future world that we are living in is surrounded by data and everything around us is connected to a data source. Therefore, data can be introduced as the new DNA of the 21st century and upcoming future. As a result of this, information extraction using data is used to create various types of applications in different fields, and in order to achieve this, it is very much needed some different types of data management tools can extract data very quickly and intelligently. However, at present, Machine Learning technology is being used as a promising technology that can analyze data and develop real-world applications intelligently using a diverse array of different ML algorithms. Moreover, Machine Learning is considered as one of the most popular and rapidly using technologies in the near future. Whilst there are some drawbacks of ML due to several limitations such as data acquisition problems as ML needs a massive data set with

inclusive, unbiased, and good quality. Secondly, it requires more time to let the algorithms learn with an amount of accuracy and relevancy and more resources which means an additional requirement of computer power. The third challenge is the ability to interpret results generated by the algorithms accurately. In addition, another disadvantage is while ML is autonomous it is highly susceptible to errors. However, most of the ML researchers face problems currently with their data set. This is because the current level of ML is capable of dealing with a static and given data set that mainly works with supervised learning algorithms. However, this problem can be solved with MAS technology. Therefore, one of the future trends of the technology is going to mix Machine Learning and Multi-Agent System Technology. Agent-Based frameworks can be used to uplift the intelligent decision-making capability in Machine Learning in a dynamic and complex environment.

In Machine Learning or Deep Learning, the learning algorithm and the capacity of the data set are the key facts to produce the best effective solution. At this level, ML has the capability to provide the solution using the existing data set. Unfortunately, at present, it can't deal with dynamic and complex environments, especially with dynamic data sets. Entering new data set means the environment of ML becomes dynamic and it leads to have a complex scenario. Accordingly, in this situation, it is very difficult to follow the traditional Machine Learning mechanism because the scenario is always changing with time. As a result of this, it is needed to propose a new mechanism to produce the best and most effective solution as it works in an adaptive environment. In such cases, Multi-Agent System technology is very useful. This is mainly due to the "learning" feature of both Machine Learning and Multi-Agent Systems technology. The word "learning" is common for both technologies. As a result of this, a learning agent can be placed to learn in a timely changing environment in-order to make the intelligent, effective best solution.

Moreover, Expert Systems can also be combined with Machine Learning to give efficient live solutions. Accordingly, it is very much clear that the future of the Multi-Agent System technology will be mixed with Machine Learning and come with Hybrid systems that can handle the complexity of the existing systems.

The second future trend of MAS technology can be thought will join with Complex Adaptive System technology. At present, Complex Adaptive System technology is capable of making centralized decisions with the help of simple agents that are working together to achieve a common goal. However, in the future, Complex Adaptive System will be mixed with MAS and will be capable to take decentralized decisions that will be lead to overcoming drawbacks of exiting technology and facing better dynamic environmental conditions. Not only that but also, many technologies today have a static ontology. However, due to the advancements and evolutions of technology in the future, there will be ontology-changing incidents. To overcome this issue, MAS technology can be used as it helps to increase the adaptability of systems.

The future of the MAS technology with its agents can be thought to develop with different features. This is because, at present agents are considered as simple or small entities that follow rules to achieve a common task. Accordingly, these agents generally have simple tasks to be performed to achieve the common goal. However, in the future, it can be guessed that the agents in the system will be capable of performing a single complex task such as Machine Learning applications and will be led to perform an extremely complex task. Moreover, they will perform very complex tasks and will communicate with others to give very intelligent efficient solutions.

As the next future trends of MAS can be thought to mix with the area of parallel computing where agents are collaboratively engaging to streamline different types of actions simultaneously. Weerasinghe and others have developed a system named ITray which is a multi-agent solution for LAN-based file sharing. In this research, MAS technology has been applied to handle the complexity of the computer network, gain better performance, and reduce resource wastage. This system presents a distributed multi-agent system that can be used to reduce resource wastage in the Local Area Network. This system is comprised of a managing agent and four ordinary agents such as a file send agent, file receive agent, download agent and load balancing and dynamic scheduling agent. Accordingly, when the user feeds an URL to the system for download, the system communicates with other manager agents in the Local Area Network and checks whether the file is available or not. If the file is available, get that file from that client. Otherwise, the system will download the file through the file download agent. When the user feeds several URLs, the system follows the previous procedure and allocates downloading tasks to other clients through the load balancing and dynamic scheduling agent when they are free. Task allocation, file sending and reserving has been done, through agent-agent communication. Accordingly, this system depicts the behavior of parallel computing with the involvement of MAS technology [50, 51]. Secondly, when considering a high-performance cluster computer, it has individual multiple nodes that run parallelly. Accordingly, each node can be considered as an agent as these nodes are working individually while communicating with other nodes to give a high performance. In such a case parallel computing/ distributed computing is mixed with MAS technology. Furthermore, the main application areas are endorsed with high-performance cluster computers and computer networks. In addition, robotic technology also will be mixed with MAS technology. The automation and the coordination of the entire robotic system can be guided by agents.

10. CONCLUSION

This paper presented a broad explanation of Multi-Agent System technology by addressing many MAS-related topics. Chapter 1 of the paper has given a very clear detailed overall explanation of Multi-Agent System technology by deeply presenting the birth point and evolution of the technology. Moreover, it provides a clear development history of MAS technology. In the next step, the paper gave a detailed explanation of agents which are the key entity of the technology by stating different definitions. Moreover, the paper has given a very detailed classification of agents. As the next step, the paper has been focused on the MAS technology by addressing its features, communication approaches and comparisons between other related systems. Then, it has given existing MAS applications that are mostly and widely available in different disciplines by explaining the current directions of the technology. Finally, several future trends of the MAS technology have been noted. We expect this article will be insightful and comprehensive material on MAS for researchers in the area.

REFERENCES

[1] Haenlein, M., Kaplan, A. (2019). Guest editorial to the special issue, A brief history of AI: On the past, present, and future of artificial intelligence. California Management Review, 61(4): 5-14. https://doi.org/10.1177/0008125619864925

[2] Turing, A.M. (1950). I.-Computing machinery and intelligence. Mind, 59(236): 433-460. https://doi.org/10.1093/mind/LIX.236.433

[3] Chaib-Draa, B., Moulin, B., Mandiau, R., Millot, P. (1992). Trends in distributed artificial intelligence. Artificial Intelligence Review, 6(1): 35-66. https://doi.org/10.1007/BF00155579

[4] Vlassis, N. (2007). A concise introduction to multiagent systems and distributed artificial intelligence. Synthesis Lectures on Artificial Intelligence and Machine Learning. https://doi.org/10.2200/S00091ED1V01Y200705AIM002

[5] Dorri, A., Kanhere, S.S., Jurdak, R. (2018). Multi-agent systems: A survey. IEEE Access, 6: 28573-28593. https://doi.org/10.1109/ACCESS.2018.2831228

[6] Bouquet, F., Chipeaux, S., Lang, C., Marilleau, N., Nicod, J.M., Taillandier, P. (2015). Introduction to the agent approach. In Agent-based Spatial Simulation with NetLogo, pp. 1-28. https://doi.org/10.1016/B978-1-78548-055-3.50001-0

[7] Balaji, P.G., Srinivasan, D. (2010). An introduction to multi-agent systems. Innovations in Multi-Agent Systems and Applications-1. https://doi.org/10.1007/978-3-642-14435-6_1

[8] Drozdov, V.N., Kim, V.A., Lazebnik, L.B. (2011). Modern approach to the prevention and treatment of NSAID-gastropathy. Experimental & Clinical Gastroenterology, 2011(2): 106-110.

[9] Ye, D., Zhang, M., Vasilakos, A.V. (2016). A survey of self-organization mechanisms in multiagent systems. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 47(3): 441-461. https://doi.org/10.1109/TSMC.2015.2504350

[10] Burgin, M., Dodig-Crnkovic, G. (2009). A systematic approach to artificial agents. arXiv preprint arXiv:0902.3513.

[11] Mizzaro, S. (2021). Towards a theory of epistemic information. http://www.dimi.uniud.it/˜mizzaro.

[12] Roche, R., Lauri, F., Blunier, B., Miraoui, A., Koukam, A. (2013). Multi-agent technology for power system control. In Power Electronics for Renewable and Distributed Energy Systems, pp. 567-609. https://doi.org/10.1007/978-1-4471-5104-3_15

[13] Kallem, S.R. (2020). artificial intelligence in the movement of mobile agent (robotic). www.iaeme.com/ijcet.asp.

[14] Bösser, T. (2001). Autonomous agents. International Encyclopedia of the Social & Behavioral Sciences, 1002-1006. https://doi.org/10.1016/B0-08-043076-7/00534-9

[15] Vitek, M., Peer, P. (2020). Intelligent agents in games: Review with an open-source tool. Advances in Computers, 116(1): 251-303. https://doi.org/10.1016/BS.ADCOM.2019.07.005

[16] https://www.researchgate.net/publication/221622801_An_Overview_of_Cooperative_and_Competitive_Multiagent_Learning.

[17] Berna-Koes, M., Nourbakhsh, I., Sycara, K. (2004). Communication efficiency in multi-agent systems. In IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004, New Orleans, LA, USA, pp. 2129-2134. https://doi.org/10.1109/robot.2004.1307377

[18] Cardoso, R.C., Ferrando, A. (2021). A review of agent-based programming for multi-agent systems. Computers, 10(2): 16. https://doi.org/10.3390/COMPUTERS10020016

[19] Collier, R.W. (2015). PRIMA 2015: Principles and practice of multi-agent systems. 18th International Conference, Bertinoro, Italy, October 26-30, 2015, Proceedings. https://doi.org/10.1007/978-3-319-25524-8

[20] Honorato-Zimmer, R., Millar, A.J., Plotkin, G.D., Zardilis, A. (2019). Chromar, a language of parameterised agents. Theoretical Computer Science, 765: 97-119. https://doi.org/10.1016/j.tcs.2017.07.034

[21] Hindriks, K.V., Dix, J. (2014). GOAL: a multi-agent programming language applied to an exploration game. In Agent-Oriented Software Engineering, pp. 235-258. https://doi.org/10.1007/978-3-642-54432-3_12

[22] Bordini, R.H., Hübner, J.F., Wooldridge, M. (2007). Programming Multi-Agent Systems in Agentspeak Using Jason. John Wiley & Sons.

[23] Dennis, L.A., Farwer, B. (2008). Gwendolen: A BDI language for verifiable agents. In Proceedings of the AISB 2008 Symposium on Logic and the Simulation of Interaction and Reasoning, Society for the Study of Artificial Intelligence and Simulation of Behaviour, pp. 16-23.

[24] Bellifemine, F., Caire, G., Poggi, A., Rimassa, G. (2008). JADE: A software framework for developing multi-agent applications. Lessons learned. Information and Software technology, 50(1-2): 10-21. https://doi.org/10.1016/j.infsof.2007.10.008

[25] Bergenti, F., Iotti, E., Monica, S., Poggi, A. (2017). Agent-oriented model-driven development for JADE with the JADEL programming language. Computer Languages, Systems & Structures, 50: 142-158. https://doi.org/10.1016/J.CL.2017.06.001

[26] Bordini, R.H., Dastani, M., Dix, J., Seghrouchni, A.F. (2005). Multi-agent programming: languages, platforms and applications. New York Springer, Multiagent Syst. Artif. Soc. simulated Organ. https://doi.org/10.1007/b137449

[27] Aschermann, M., Dennisen, S., Kraus, P., Müller, J.P. (2018). LightJason, a highly scalable and concurrent agent framework: Overview and application. In AAMAS, Stockholm, Sweden, pp. 1794-1796.

[28] Boissier, O., Bordini, R.H., Hübner, J.F., Ricci, A., Santi, A. (2013). Multi-agent oriented programming with JaCaMo. Science of Computer Programming, 78(6):747-761. https://doi.org/10.1016/j.scico.2011.10.004

[29] Hettige, B., Karunananda, A., Rzevski, G. (2016). A multi-agent solution for managing complexity in English to Sinhala machine translation. International Journal of Design & Nature and Ecodynamics, 11(2): 8-96. https://doi.org/10.2495/DNE-V11-N2-88-96

[30] Hettige, B., Karunananda, A.S., Rzevski, G. (2021). MaSMT4: The AGR organizational model-based multi-agent system development framework for machine translation. In Inventive Computation and Information Technologies, 691-702. https://doi.org/10.1007/978-981-33-4305-4_50

[31] Hashmi, M.A., Akram, M.U., Fallah-Seghrouchni, A.E. (2017). Place: Planning based language for agents and computational environments. In International Workshop on Engineering Multi-Agent Systems, Sao Paulo, Brazil, pp. 142-158. https://doi.org/10.1007/978-3-319-91899-0_9

[32] Austin, B.J., Heine, V., Sham, L.J. (1962). General theory of pseudopotentials. Physical Review, 127(1): 276. https://doi.org/10.1103/PhysRev.127.276

[33] Ermon, S., Gomes, C., Selman, B. (2011). A message passing approach to multiagent gaussian inference for dynamic processes. In The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 3, Taipei, Taiwan, pp. 1277-1278.

[34] Jagannathan, V. (1989). Blackboard architectures and applications. Elsevier.

[35] Al-Azawi, R.K., Ayesh, A. (2013). Comparing agent-oriented programming versus object-oriented programming. In Proceedings of the ICIT 2013 The 6th International Conference on Information Technology, Amman, Jordan, pp. 8-10.

[36] McArthur, S.D., Davidson, E.M., Catterson, V.M., Dimeas, A.L., Hatziargyriou, N.D., Ponci, F., Funabashi, T. (2007). Multi-agent systems for power engineering applications-Part I: Concepts, approaches, and technical challenges. IEEE Transactions on Power Systems, 22(4): 1743-1752. https://doi.org/10.1109/TPWRS.2007.908471

[37] McArthur, S.D., Strachan, S.M., Jahn, G. (2004). The design of a multi-agent transformer condition monitoring system. IEEE Transactions on Power Systems, 19(4): 1845-1852. https://doi.org/10.1109/TPWRS.2004.835667

[38] Colak, I., Sagiroglu, S., Fulli, G., Yesilbudak, M., Covrig, C.F. (2016). A survey on the critical issues in smart grid technologies. Renewable and Sustainable Energy Reviews, 54: 396-405. https://doi.org/10.1016/j.rser.2015.10.036

[39] Rahman, M.S., Mahmud, M.A., Pota, H.R., Hossain, M.J. (2015). A multi-agent approach for enhancing transient stability of smart grids. International Journal of Electrical Power & Energy Systems, 67: 488-500. https://doi.org/10.1016/j.ijepes.2014.12.038

[40] Jin, X., He, Z., Liu, Z. (2011). Multi-agent-based cloud architecture of smart grid. Energy Procedia, 12: 60-66. https://doi.org/10.1016/j.egypro.2011.10.010

[41] Palicot, J., Moy, C., Résimont, B., Bonnefoi, R. (2016). Application of hierarchical and distributed cognitive architecture management for the smart grid. Ad Hoc Networks, 41: 86-98. https://doi.org/10.1016/j.adhoc.2015.12.002

[42] Kamdar, R., Paliwal, P., Kumar, Y. (2018). Labview based multi-agent approach towards restoration in smart grid. Materials Today: Proceedings, 5(2): 4684-4691. https://doi.org/10.1016/j.matpr.2017.12.040

[43] Shakshuki, E., Reid, M. (2015). Multi-agent system applications in healthcare: current technology and future roadmap. Procedia Computer Science, 52: 252-261. https://doi.org/10.1016/j.procs.2015.05.071

[44] Camarinha-Matos, L.M., Afsarmanesh, H. (2004). TeleCARE: Collaborative virtual elderly support communities. Proceedings of TELECARE 2004 - Int. Workshop on Tele-Care and Collaborative Virtual Communities in Elderley Care. https://doi.org/10.5220/0002677300010012

[45] Dudek, G., Jenkin, M.R., Milios, E., Wilkes, D. (1996). A taxonomy for multi-agent robotics. Autonomous Robots, 3(4): 375-397. https://doi.org/10.1007/BF00240651

[46] Ismail, Z.H., Sariff, N., Hurtado, E.G. (2018). A survey and analysis of cooperative multi-agent robot systems: challenges and directions. In Applications of Mobile Robots, pp. 8-14. https://doi.org/10.5772/INTECHOPEN.79337

[47] Arguenon, V., Bergues-Lagarde, A., Rosenberger, C., Bro, P., Smari, W. (2006). Multi-agent based prototyping of agriculture robots. In International Symposium on Collaborative Technologies and Systems (CTS'06), Vegas, NV, USA, pp. 282-288. https://doi.org/10.1109/CTS.2006.57

[48] Duan, Y., Cui, B.X., Xu, X.H. (2012). A multi-agent reinforcement learning approach to robot soccer. Artificial Intelligence Review, 38(3): 193-211. https://doi.org/10.1007/S10462-011-9244-8

[49] Goonatilleke, M.A.S.T., Jayampath, M.W.G., Hettige, B. (2018). Rice express: A communication platform for rice production industry. In International Conference of the Sri Lanka Association for Artificial Intelligence, Moratuwa, Sri Lanka, pp. 269-277. https://doi.org/10.1007/978-981-13-9129-3_19

[50] Weerasinghe, L.D.S.B., Hettige, B., Kathriarachchi, R.P.S., Karunananda, A.S. (2016). ITray: Multi-agent solution for LAN based file sharing. Proceedings in Computing, 9th International Research Conference-KDU, Sri Lanka, pp. 81-86.

[51] Thepperumal, S.K., Margabandu, V., Radhakrishnan, R., Amaladas, J.R., Ananthakrishnan, S.V. (2021). Machinability investigations on aerospace custom 450 alloy using TiAlN/TiCN, TiCN/TiAlN coated and uncoated carbide tools. Journal Européen des Systèmes Automatisés, 54(2): 325-334. https://doi.org/10.18280/jesa.540215