Software Engineering
2017; 5(1): 8-25
http://www.sciencepublishinggroup.com/j/se
doi: 10.11648/j.se.20170501.12
ISSN: 2376-8029 (Print); ISSN: 2376-8037 (Online)

Review Article

A Concise Overview of Software Agent Research, Modeling, and Development

Salama A. Mostafa[1], Mohd Sharifuddin Ahmad[2], Aida Mustapha[3], Mazin Abed Mohammed[4]

[1]College of Graduate Studies, Universiti Tenaga Nasional, Selangor, Malaysia
[2]College of Information Technology, Universiti Tenaga Nasional, Selangor, Malaysia
[3]Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn, Johor, Malaysia
[4]Faculty of Communication and Information Engineering, University Technical Malaysia, Melaka, Malaysia

Email address:
semnah@yahoo.com (S. A. Mostafa), sharif@uniten.edu.my (M. S. Ahmad), aidam@uthm.edu.my (A. Mustapha),
mazin_top_86@yahoo.com (M. A. Mohammed)

To cite this article:
Salama A. Mostafa, Mohd Sharifuddin Ahmad, Aida Mustapha, Mazin Abed Mohammed. A Concise Overview of Software Agent Research, Modeling, and Development. *Software Engineering*. Vol. 5, No. 1, 2017, pp. 8-25. doi: 10.11648/j.se.20170501.12

Received: January 6, 2017; **Accepted**: January 24, 2017; **Published**: March 4, 2017

Abstract: Software agent technology has been intensively explored in the past three decades. It is explicitly or implicitly applied in many systems. Software agent research covers a wide range of area which makes it challenging for new researchers to comprehend the peculiarities and complexities of the technology. Consequently, this paper provides a concise overview of software agent research, modeling, and development. It summarizes and analyzes more than 100 sources of publication including research papers, articles, and books. The aim of the paper is to provide a quick start to new researchers in software agent and multi-agent systems. The paper offers the following contributions: (1) it determines the milestone achievements of software agent conceptualization, modeling and development platforms, (2) it defines the related terminologies of the field and reveals their redundancies, (3) it summarizes the multi-agent systems technology and finally, (4) it explores the current active research topics in software agent and multi-agent systems.

Keywords: Software Agent, Multi-agent System, Agent-Oriented Programming, Agent Models

1. Introduction

Conventional software systems are made to perform in normal circumstances, e.g., deal with valid data and static environments. The performance and the outcomes of these systems are straightforward. However, systems that are susceptible to uncertainties, work in dynamic environments, handle highly complex tasks and incomplete information demand advanced models and algorithms [1] [2] [3] [4].

Consequently, the cycle of technological development progresses at a much rapid pace. Research and development efforts introduce new solutions stemmed from the concepts and theories of Artificial Intelligence (AI). The AI discipline suggests several proposals and one of which is software agent technology [5] [6] [7] [8]. Software agent and multi-agent systems get their root from Distributed Artificial Intelligence (DAI) and distributed computing [7] [9]. Software agents and multi-agent systems significantly are facilitated solutions to many complex and distributed problems [4] [10]. Figure 1 shows the scope of software agents and multi-agent systems.

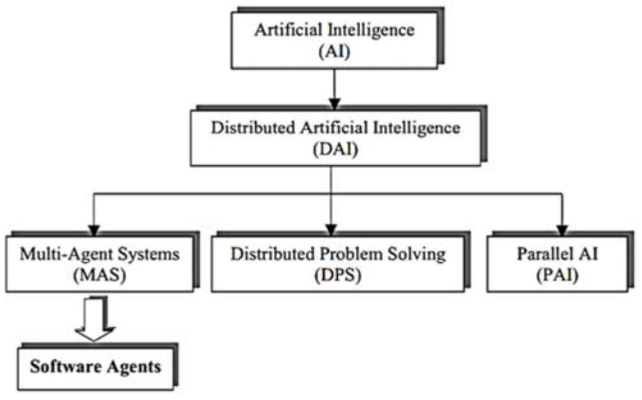

Artificial Intelligence (AI)
Distributed Artificial Intelligence (DAI)
Multi-Agent Systems (MAS)
Distributed Problem Solving (DPS)
Parallel AI (PAI)
Software Agents

Figure 1. *Artificial Intelligence and Software Agents [9].*

The motivation behind this contribution is the observed difficulties that new researchers of software agent field encounter [2] [10]. Software agent and multi-agent systems cover a wide scope of the research area. This issue presents a challenge of comprehending this technology by the new researchers of this field. The research involves many issues that are related to software agent and multi-agent systems theories, architectures and cognitions and their complex notations [11]. This paper is meant to facilitate the understanding of software agent and multi-agent system concepts and promote their technology.

This paper presents a concise overview of software agent and multi-agent systems research, modeling and development. It is divided into five main sections. This section introduces the motivation of the paper. Section II reviews the literature on software agents. It discusses different aspects of software agents including definitions, types, properties, architectures, and models. Section III reviews the literature on multi-agent systems including communication language and agent-oriented programming. Section IV presents agents' commitment, deliberation, situation awareness, adjustable autonomy, collective intelligence, norm, emotion, morality, and sincerity as active research topics. Finally, Section V concludes the paper by deliberating the notions of the research milestones.

2. Software Agent

Research in software agents has progressed over more than three decades due to the demands of dynamic and open environments and the complexity of tasks. Agents are capable of making autonomous decisions and performing goal-directed actions in many applications [1] [5] [11]. Software agents' applications range from personalized small systems, e.g., email filters to complex and critical systems, e.g., air traffic control [7]. This section reviews and discusses the issues and underlying concepts of software agents. It includes agents' definitions, types, properties, terminology, architectures, and models.

2.1. Software Agent Concept

The concept of agents is first introduced in mid-1950s when J. McCarthy and G. Selfridge proposed 'soft robot', a computer software that has a goal, carries out tasks and seek feedback from humans [12]. Hewitt [13] refined the idea and introduced the term 'actor'. An actor "is a computational agent which has a mail address and a behavior. Actors communicate by message-passing and carry out their actions concurrently" [13]. Hewitt proposed that a software agent actor is an executing object. It has the characteristics of self-contained, encapsulation, and interactivity. Subsequently, many views and perceptions of software agents are elaborated.

2.2. Software Agent Definitions

The essential definition of software agents comes from *agency*, which is the capability of autonomous and self-directed behavior [8] [14]. Software agents like many other concepts are defined from different views. The proposed definitions highlight some aspects of the agents and ignore others and as a result, there is no comprehensive definition of what a software agent is [7]. Some of the commonly accepted definitions of software agents in the literature are as follows:

- Wooldridge and Jennings [15] generally described an agent as a software or hardware computer system that is characterized by the properties of autonomy, social ability, reactivity and pro-activeness. They define agent as "*a computer system, situated in some environment that is capable of flexible autonomous action in order to meet its design objectives.*"
- Maes [6] defined autonomous agents 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.*"
- Nwana and Ndumu [16] defined an agent as a software and/or hardware components that are capable of acting in order.
- Franklin and Graesser [17] stated that agents have to have the ability to perform domain-oriented reasoning and autonomous execution. They define an autonomous agent as "*a system situated within and a part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to effect what it senses in the future.*"
- Shoham [18] "An agent is an entity whose state is viewed as consisting of mental components such as beliefs, capabilities, choices and commitment."
- Bradshaw [19] "*An agent is a program that is, to some degree, capable of initiating actions, forming its own goals, constructing plans of action, communicating with other agents, and responding appropriately to events – all without being directly controlled by a human.*"
- The International Business Machines (IBM) corporation definition of agents is *"Intelligent agents are software entities that carry out some set of operations on behalf of a user or another program with some degree of independence or autonomy, and in so doing, employ some knowledge or representation of the user's goals or desires."* [20]. IBM develops agent-based applications for different domains like customer help desk, web browser, and personal shopping assistant [21].

2.3. Software Agent Types

Software agent types are identified by their properties and each type might have different properties. The properties define the nature of an agent's behaviors. Properties selection and setting depend on the environment and the application domain specifications. Subsequently, there are some agents that have a combination of different types due to their behaviors' specifications. Some of the agent types that are proposed by Nwana [22] are as below:

- Reactive agents

- Collaborative agents
- Interface/Personal agents
- Information/Internet agents
- Proactive agents
- Hybrid agents
- Mobile agents

2.4. Software Agent Properties

An intelligent software agent is an artifact that has the properties or some of the properties (also named as attributes or characteristics) of an intelligent entity, i.e., autonomy, reactivity, goal-directedness, rationality, embodiments and sociality [7] [16] [21].

Agent properties acquire their importance based on the specifications of the research domain [11]. It is agreed among agent researchers that autonomy is a central property of software agents [11] [17] [23]. Some properties emerge from agents' design [14], e.g., situatedness, embodiment, and rationality, while others are explicitly formulated to a certain depth, e.g., goal-directedness requires formulating a plan. However, it is very uncommon to consider all the properties in an agent due to the complexity of such consideration. For example, many agent studies ignore formulating the learning (or adaptation) property of an agent because of its complexity, unless there is a need for such formulation [5]. Table 1 details the most agreed upon software agents' properties.

Table 1. *The main properties of software agents.*

Property	Synonym	Meaning
autonomy	-	An agent operates without the direct intervention of humans or others and has some kind of control over its actions and internal states [11] [23].
situatedness	-	An agent is being a part of its environment, observes its surrounding, perceive what observe and act based on its perception [7] [14].
sociality	communicative, interactive	An agent interacts with other agents (and possibly humans) via some kind of agent-communication language [7] [11] [23].
reactivity	responsive	An agent perceives some of its environment and responds in a timely fashion to changes that occur in the environment [23] [24].
goal-directivity	proactive, purposeful	An agent takes the initiative to attain a particular goal [21] [24].
adaptive	learning	An agent changes behavior based on its past experience [5] [6] [21] [24].
mobility	-	An agent transports itself from one machine to another and through different platforms [21] [22] [24].
rationality	-	An agent is capable of autonomous and goal-directed behaviors and intends to pursue successful performance that meets its interest [11] [21].
embodiment	-	An agent interacts through a physical body in an environment [14].
inferential	-	An agent decides based on its prior knowledge and might be beyond the given information [12] [15] [23].
self-organization	-	An agent configures its activities according to its environment demands in order to achieve its goal.
persistence	continuous	An agent has continuously running thread of processing [24].
flexibility	-	An agent filters its inputs and reason over its actions [24].
effective	-	An agent is successful at eventually achieving its goals [5].
efficient	-	An agent performs better than the conventional autonomic or predetermined reactive systems [5].
reproduction	cloning	An agent is capable of reproducing itself and adapting to changes [12].
personality	-	An agent has human-like nature such as an individual view of the world and emotion [12] [15] [23].

2.5. Software Agent Architectures and Models

Agent architectures and models vary across initiatives as a result of their continuous development by researchers and based on the essential needs of the technology. Agent development produces a number of models that investigate reactive and/or goal-directed behaviors [25]. Some well-known models are the reactive model, goal-directed model, BDI model and Soar model [7] [26]. This development helps the agent technology to embody sophisticated autonomous systems [4]. It provides mechanisms that develop dynamic autonomous systems for real-world environments [1]. Therefore, an agent is seen to be one of the core competences that contribute to software systems [2] [20]. Figure 2 presents agent models development across the initiatives.

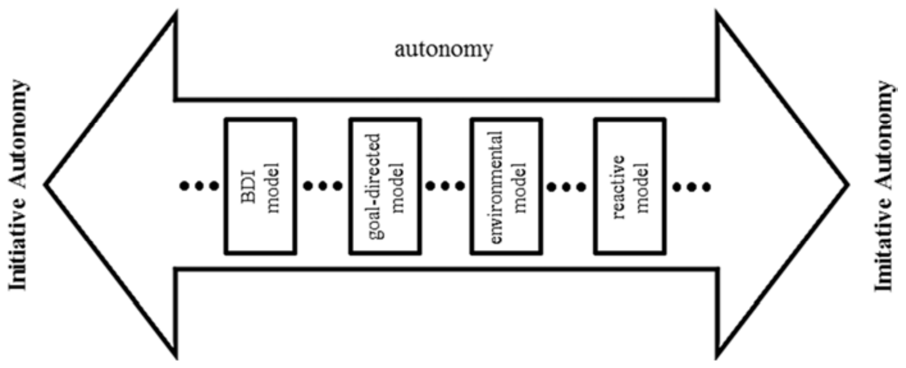


Figure 2. *The development of software agent models.*

2.5.1. Reactive Agent

At the right side of Figure 2 is the reactive or the executive agent model which is a type of agent that can only follow directly what it has been explicitly programmed to do. The agent is considered primitive and acts without much reasoning on the causes and effects of its actions [26]. Hence, it commits to a particular plan but the re-planning option does not exist. This means, if a system faces failure or has an opportunity, it will not proactively act unless it is programmed to do so [11] [27]. Practically, the reactive architecture shows great successes, especially in the industrial field due to its simplicity [7] [28]. An example is the famous subsumption architecture that is proposed by Brooks [29] [30]. Figure 3 shows a reactive agent architecture.

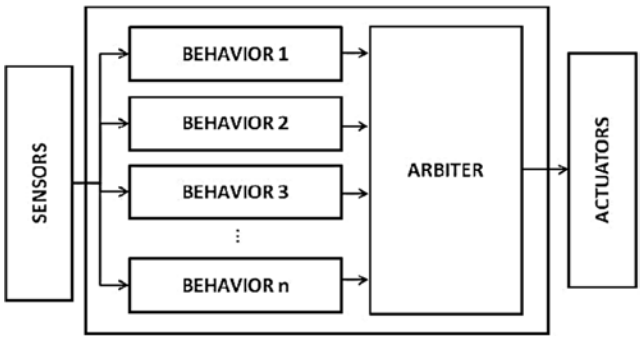


Figure 3. *An abstract reactive architecture [28].*

Nevertheless, a reactive agent's deficiencies are its dependency on the local information which affects its global view especially in the long run; its non-compliance with the adaptation and learning from experience; no clear principled methodology due to the emergent property of its autonomy and intelligence; and sensor data with no reasoning that might produce inappropriate actions [7] [26] [28]. For instance, a system's sensors indicate conflict information. This contradiction would more likely lead to a wrong conclusion as one of the sensors might have a faulty reading. In order to determine the faulty sensor, the agent needs to perceive additional states and reason on the situation, but the reactive model does not have such capability.

2.5.2. Environmental Agent

An environmental agent model is a logic-based model that adopts the knowledge base system architecture [11]. The agent has an inference engine as a reasoning mechanism and is situated in an environment that has symbolic representations. Its responses are configured based on sensing the changes that occur in the environment [31]. The advantages of this model are inherited from the knowledge base system adoption such as an explicit representation of the world, easy encoding and easy understanding [26]. However, in this model, the environment dependency stands against the agent's potentiality especially in uncertain and complex environments [1]. The symbolic representation of the environment constrains the autonomous behavior of the agent as the environmental inputs determine the number of possible actions.

2.5.3. Goal-Directed Agent

In Goal-Directed (GD) models, agents deliberate, plan, generate and implement their own goals based on the environment's current situation [28] [32] [33]. Examples of GD agents are found in [32], [33], [34] and [35]. The GD model transforms the agent from the task-oriented (reactive) model to the dynamic goal-oriented model [34]. It enables an agent to autonomously respond to unexpected situations using some goal reasoning strategies [33]. Hence, it is a key aspect of dynamic planning [35]. Figure 4 elucidates a dynamic planning GD agent architecture which is an extension of Nau [32] model.

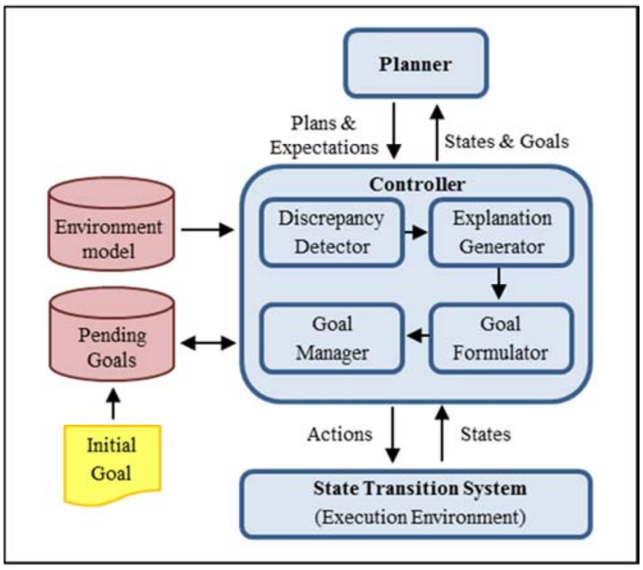


Figure 4. *A goal-directed agent architecture [35].*

To explain the GD methodology, consider a robot working on completing a task of moving a box from a location A to another location B. During the task performance, the robot faces a door. In order to be able to open the door, the robot needs to put the box down first. If the robot put the box down and opens the door it needs to remember to carry the box again before proceeding with its movement to the location B. This process is associated with a goal-oriented reasoning strategy of the GD agent.

GD agents have been applied in simulation domains as in [33], games as in [36] and Unmanned Systems (US) as in [1], [34] and [35]. In GD agents, however, goal revision mechanism is still an active research topic [36]. Foremost, there is an insufficiency in responding to unanticipated events and dealing with discrepancies of the environment as a result of the absence of retrospective process associated with the generated plans.

2.5.4. Belief-Desire-Intention Agent

The Belief-Desire-Intention (BDI) model is a goal-directed model but with specific architecture [10]. The BDI agent model is introduced in Georgeff and Lansky [37] and Rao and

Georgeff [38]. It espouses the philosophical human practical reasoning model that is proposed by Bratman [39]. The BDI is considered as the best-known architecture to model practical reasoning agent as it is synonymous to humans' practical reasoning [26] [27] [38] [40]. Agents' mental attitudes of the BDI represent respectively, the informational, the motivational, and the deliberation states of the agents [8] [38]. These mental attitudes determine the agents' behaviors and are critical for achieving adequate or optimal performance when deliberation is subject to resource bounds.

The BDI architecture supports agents' autonomous capabilities via enabling an agent to select a task based on its beliefs and decides what actions are needed to be performed to complete the tasks [41] [42]. The four important procedures in the BDI agent practical reasoning are summarized in the following and illustrated in Figure 5 [11] [41]:

1. observe the world and update beliefs;
2. deliberate desires to pursue:
   - determine the available options;
   - filters the options;
3. select intentions to satisfy the desires;
4. execute the selected intentions.

An agent's beliefs are its knowledge about the world, its internal state, and other agents shared states [42]. A belief defines a world state in some forms, e.g., a variable, data structure or logical expression [27]. When the agent observes the world, it perceives some of its states. The perceived states are represented in beliefs' forms [21]. The beliefs can be updated based on the continuous observation of the world. Beliefs and their updates are susceptible to constraints and two of which are world dynamism and uncertainty [1].

A desire is a representation of a goal. It defines an agent's state of the world in some forms, e.g. variables, data structure or logical expressions [38]. Going back to the robot example, the goal of moving the box from location A to location B represents the robot's desire. While dropping the box to open the door and carrying the box after opening the door represent other desires. However, the robot needs to have a repository of possible actions that can be performed in its environment to satisfy its desires, i.e., a plan.

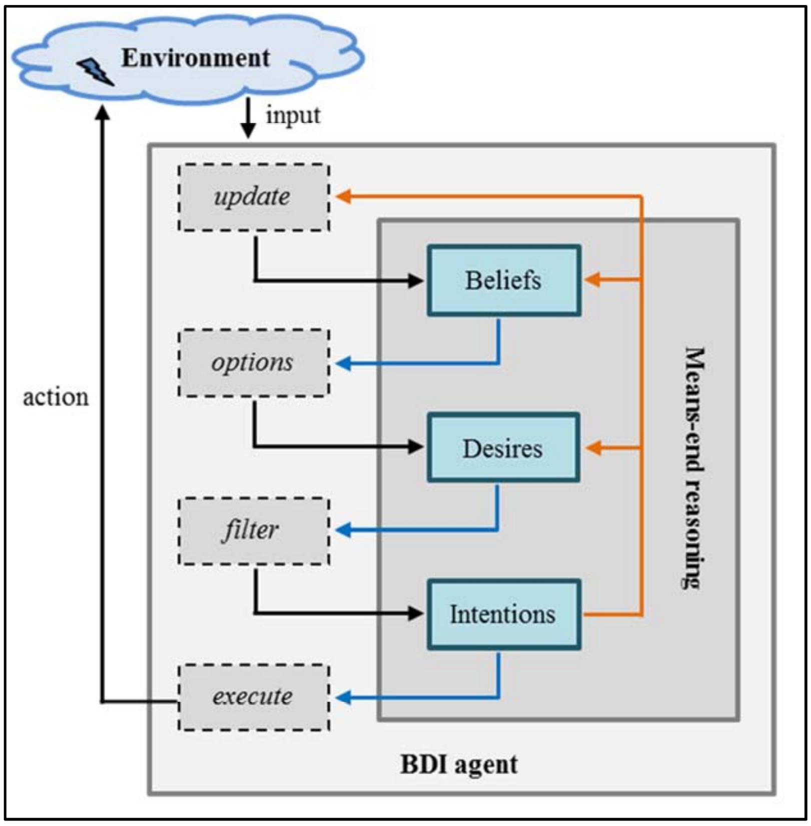


Figure 5. *An abstract BDI architecture.*

One formalization of plans is a task orientation in which each desire is allocated to a task [27]. A task accomplishment requires the agent to perform some actions. The actions define an agent's ability in an environment [43]. A rational BDI agent would select the best sequence of actions from the set of possible actions to perform based on its beliefs about the world [40] [44]. Since the agent is not being able to achieve all its desires, it must decide on some subset of its desires and commit resources to achieve them [8].

Intentions are chosen desires through deliberation process to achieve goals, i.e., the committed plans [42]. Georgeff et al. [27] proposed that an intention is computationally defined as an executing thread in a process of performing actions. However, in the BDI agent as well as some other models, there is a challenge of modeling an efficient commitment strategy to control agent's commitment to its intention. Agent commitment is detailed in Section IV. A. The following algorithm is an abstract representation of the BDI agent:

1. sense the environment
2. update beliefs
3. select a plan
4. if there is no plan
   - choose a desire to pursue
   - find a plan to achieve the desire
5. decide on the actions of the plan
6. execute actions
7. validate the plan

2.5.5. Hybrid Agent

The hybrid model or integrated model provides an architecture that has a combination of a reactive and deliberative agent [28]. It attempts to exploit the advantages of the reactive and the deliberative models [26] [34]. The layered concept is applied in the agent's architecture to organize its behavior. A layer is a link that connects a sensor input with an action output as in the subsumption architecture of Brooks [29], i.e., the reactive part. The layers interact with each other to shape the agent's global behavior, i.e., the deliberative part [28] [31]. Ferguson [45] proposed a horizontal layered architecture and Müller et al. (1996) proposed a vertical layered architecture as shown in Figure 6. In the horizontal layered architecture, the layers are arranged horizontally and a layer represents an agent by itself while in the vertical layered architecture, the layers are arranged vertically and a layer forms a behavior of the agent [26] [31]. The vertical layering architecture enhances the layered agent via improving its deliberativeness and actions' consistency.

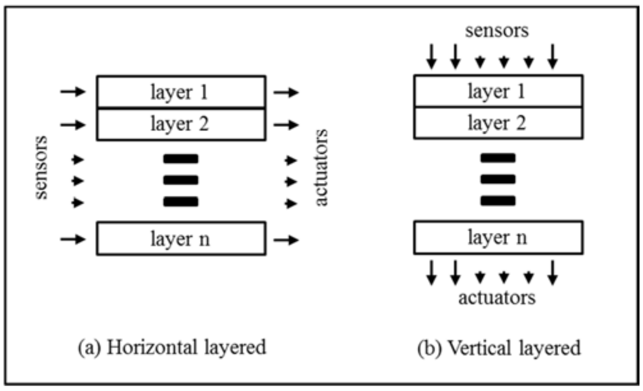


Figure 6. *The layered agent architectures.*

Three-layered architectures are used in many robotic systems. Ceballos et al. [34] proposed a three-layered architecture of agent that consists of a deliberative layer, a decision layer and a function layer to balance the deliberative and the reactive behavior of the agent and provide better performance. Kong and Xiao [47] proposed a typical three-layered architecture. The lowest is the execution or the reactive layer, which provides the inputs and processes the outputs. The middle layer is the detailed control layer that handles tasks' operations, such as finding the possible and the alternative procedures to carry out a task, tasks' timing and performance synchronization and adjustment. Finally, the top layer is the global control layer that is responsible for the planning aspect and deals with the agent's goals. Figure 7 shows the three-layered control architecture proposed by Kong and Xiao [47] including the environment.

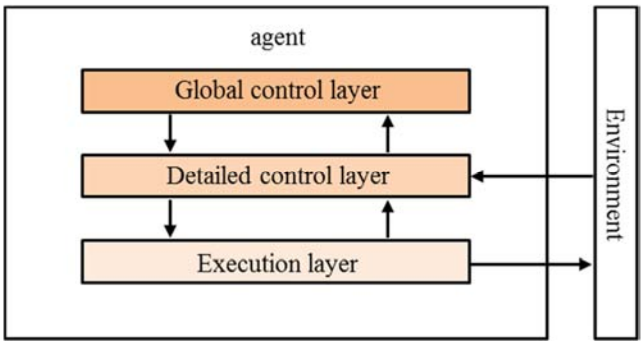


Figure 7. *The three-layered agent architecture [47].*

2.6. Other Models and Architectures

Some other popular agent models are the Soar model, PRS model, Sensible model and utility-based model. They are detailed in the following:

2.6.1. Soar Agent

The Soar agent model is a goal-directed rule base reasoning model. It adopts the reactive agent's use of operators and has a means-end reasoning strategy (chunking). A hierarchy of operators' execution leads to achieving a goal. It uses a forward chaining search engine for operators' selection in order to reach the goal state [48]. The satisfaction of the preconditions is determined by an operator that detects current states. The states represent the agent's goals and beliefs. The Soar and the BDI agent models are similar in the current states that correspond to the beliefs, the goals that correspond to the desires, the selected operators that correspond to the intentions and the existence of the commitment strategy in both models [27].

2.6.2. PRS Agent

The Procedural Reasoning System (PRS) is an agent-based application development platform [37]. The PRS agent adopts the BDI model and it is designed to work in a dynamic environment [27]. In the PRS agent, beliefs are the perception of the environment; desires are the tasks allocated to the agent; intentions are committed desires or tasks to be completed; plans are courses of actions that have a specific configuration and can complete the task. Tasks' accomplishment achieves the goal [26]. The beliefs, desires, intentions and plans are managed and revised by an interpreter. The interpreter processes the perceptions and updates the beliefs; generates new desires in the form of tasks based on the updated beliefs; selects a task as intentions and apply procedural knowledge to perform the actions that complete the task. The PRS is considered a successful model of the BDI agent. It is used in many applications, e.g. air-combat simulation and is able to perform in incomplete or incorrect conditions. Figure 8 shows the PRS agent architecture.

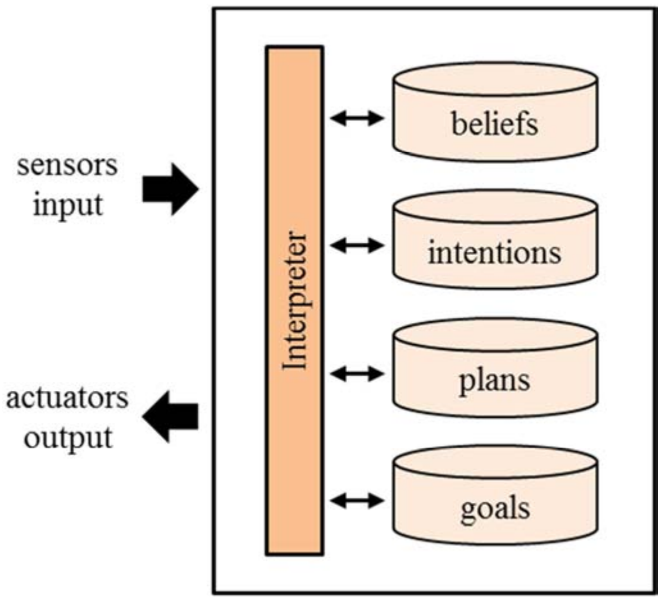


Figure 8. *The PRS agent.*

2.6.3. Sensible Agent

Barber [49] proposed a Sensible agent architecture to demonstrate flexible, receptive and adaptive automated systems. The Sensible model is designed to operate in dynamic environments. The model allows an agent to dynamically configure its autonomy level based on its understanding of the situations. The autonomy configuration for a particular situation is obtained through a predictive process of an autonomy reasoning module. In the Sensible model, the agent reasons about situation handling through two decision-making phases, which are tasks selection and tasks delegation. In the model, determining which agent can decide on the current goal is made via a voting scheme, whereby the agent needs to gain a certain number of votes in order to qualify for decision making. The agent behavior is, however, hard to predict as the autonomy configuration is based on the agent's internal states.

2.6.4. Utility-Based Agent

As mentioned earlier, a goal-directed agent has a set of goals to be achieved and its satisfaction is implicitly represented by the goals' achievement. The utility-based model directs an agent's behavior towards the specific level of satisfaction based on some explicit utility measures. Often, the utility measurement criterion is represented by agent's performance evaluation [31]. A utility function measures the agent's action choice towards maximizing its utility. The utility function is an efficient method to manipulate the agent's preferences and setup its commitment to delegated objectives [11]. A utility-based agent can be a convenient approach to a model rational agent. It is widely adopted in game theory to form worth-oriented decisions.

2.7. Software Agent and Game Theory

Game theory and software agent disciplines share many characteristics, hence, each of which adopts the other concepts. Game theory is another decision science discipline that studies strategic decision-making using mathematical and computational algorithms. It is used in computer science, logic, political science, economics, biology, and psychology. Game theory as software agents target complex, distributed and constrained problems that are attributed with cooperation, negotiation, optimization and conflict. However, game theory or decision theory is not good enough to some researchers and they argue that it cannot satisfy the introspection process and prefer human-like intuitive approaches, e.g., BDI. Moreover, in game theory some strategies are exhaustive and of high computational cost, e.g., dynamic planning ensues after performing every action.

3. Multi-Agent Systems

A Multi-agent system is defined as a loosely coupled network of agents that interact to achieve a common goal that is beyond an individual agent's achievement [31] [36] [50]. A multi-agent system provides a variety of agents' capabilities which facilitates flexibility to solving problems [21]. Individual agents' goals represent solving local problems and a multi-agent system's goal represents solving distributed problems via establishing agent groups [7]. Each agent in a group is equipped with communication, coordination, cooperation and/or negotiation capabilities [11] [51]. Communication is performed via Agent Communication Language (ACL) like KQML or FIPA ACL [12] [26]. Coordination is a process of aligning and synchronizing agents group activities in order to work together using coordination algorithms such as join intention or partial global planning. Cooperation is a group of agents working together to perform a task that solves a particular problem via sharing some information, i.e., cooperative distributed problem solving. Negotiation or bargaining is a process of reaching an agreement about a particular negotiation set according to some rules, strategies and protocols such as negotiating about tasks or resources [11]. According to Jennings et al. [7], multi-agent systems in general share the following foundations:

- The existence of complex distributed problems that cannot be solved by individual agents due to their lack of knowledge and/or capability about the problem;
- There is no global control agent of the system;
- The appropriateness of a decentralized data distribution;
- There exists coordination between the agents.

Subsequently, there is no standard multi-agent architecture and its formulation depends on the nature of the distributed problem that the system attempts to solve [31] [50]. Some examples of multi-agent architectures are centralized multi-agent architecture with facilitator or mediator agent and decentralized multi-agent architecture [51]. The multi-agent system is deployed to cater for complex distributed problems such as air-traffic control and design and manufacturing problems [7] [51].

3.1. Agent Communication Language

The Agent Communication Language (ACL) provides a means of sharing or exchanging information between the agents in a multi-agent system. An ACL provides a transmission approach (e.g., signaling, message passing or speech act) that consists of a set of agreed upon transmission rules or protocols. The transmitted context must be understandable by the agents in order to be able to perceive its meaning.

The Knowledge Query and Manipulation Language (KQML) is an ACL, developed as part of DARPA Knowledge Sharing Effort (KSE) [26]. KQML provides the facility for agents sharing knowledge during run-time. It uses performatives (i.e., descriptive utterances that are not true or false) of the speech acts theory as agent communication framework [12]. The performatives are classified into nine categories that define agents' speech acts sets [23]. The message structure of the KQML is layered into the content layer which includes the actual content of the message, the message layer which includes the performatives set and the communication layer which includes sender, receiver, message identities and message passing parameters. The KQML efforts are adopted by the Foundation of Intelligent Physical Agent (FIPA) which provides an enhanced and standardized ACL [26]. It added new parameters including user-defined message parameters.

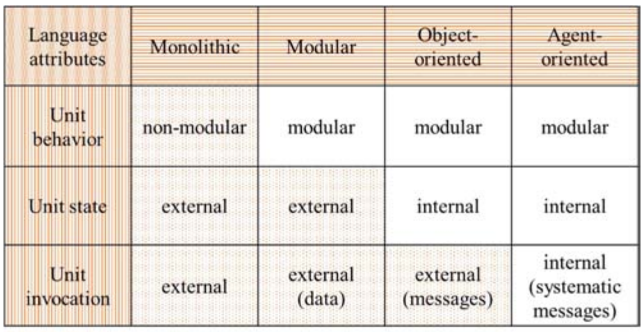
Finally, the Knowledge Interchange Format (KIF) is another agent communication formal language [12]. KIF is formalized based on the first order logic that describes the syntax and the semantics of the message. It can be used as a meta-language and enables the communications between agents with different languages.

3.2. Agent-Oriented Programming

The diversity of agents' and multi-agent systems' architectures mandates the production of a number of tools to cover the implementation of the architectures. Different programming languages are used as a development platform for the tools, e.g., Java and Prolog. The availability of the technologies facilitates the success of multi-agent systems development [7] [52]. However, selecting a suitable programming language and tool is constrained by the adopted agent and multi-agent system architectures and the application domain [18]. In this section, we study the well-known agent development programming languages and tools that implement successful agent-based applications. The study's main aim is to guide researchers to select suitable programming languages and tools in order to carry out their research.

The concepts of programming languages have been improved with time. Program architectures are enhanced from the monolithic non-modular programming languages to the object-oriented programming languages [53]. The modular programming concept offers reusable behaviors (e.g., loops and subroutines) and reduces the memory space. Object-oriented programming further introduces new properties including message passing, encapsulation, and inheritance. Object-oriented programming languages like C++, Java, and Smalltalk improves the modular concept by maintaining the subroutines, i.e., methods, and improving the structure of the method's local control over its variables, i.e., public, private and protected method options [12] [52].

The agent concept is close to the object concept as they share many properties [18] [53]. Therefore, object-oriented languages pave the way for agent-oriented programming [11] [19]. However, the objects produce passive behaviors and their methods are controlled based on some received messages [52]. But agents have the characteristics of autonomy, instructiveness, goal-directedness, reasoning and independent thread of control [18] [26]. Figure 9 shows the evaluation of programming languages from non-modular to the agent-oriented programming concepts.



Language attributes	Monolithic	Modular	Object-oriented	Agent-oriented
Unit behavior	non-modular	modular	modular	modular
Unit state	external	external	internal	internal
Unit invocation	external	external (data)	external (messages)	internal (systematic messages)

Figure 9. *Evolution of programming concepts [52].*

Some agent-oriented programming tools and platforms are summarized as follows:

3.2.1. AGENT10

It is the first agent-oriented programming language. The agent in AGENT10 consists of sets of capabilities, commitments, and beliefs. It uses a number of commitment rules, message conditions, and mental conditions to determine the committed action [11]. Agent execution cycle functions via reading messages, updating beliefs, setting commitments, selecting an action based on the capability conditions and executing the action.

3.2.2. Concurrent MetateM

It is a multi-agent system programming language that is developed by Fisher [54]. The Concurrent MetateM agent architecture has a computational engine which uses a number of temporal logic sets to specify agents' behaviors and interface for interaction.

3.2.3. KAoS

Knowledgeable Agent-oriented System (KAoS) is a fully object-oriented agent framework that works based on the distributed object technology and written in Java [55]. The agent architecture has knowledge, desires, intentions, and capabilities and provides a dynamic agent lifecycle that starts with agent birth and ends with agent death as shown in Figure 10. An agent has resistance capability via retaining and retrieving some aspects of its run cycle, i.e., cryogenic state. KAoS enhances the scalability, security of agent architecture, semantics, and extensibility of agent communication languages.

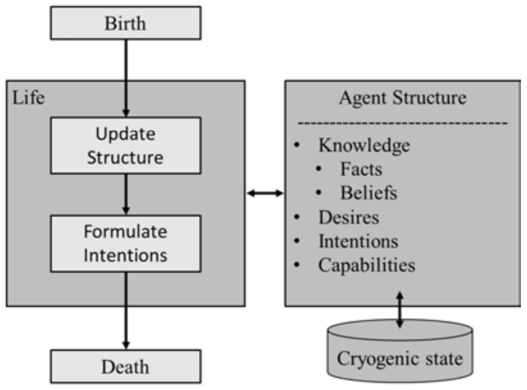


Figure 10. *The structure and dynamics of KAoS agents [55].*

3.2.4. JACK

JACK Intelligent Agents is a third-generation framework that is written in Java for a multi-agent system platform [56]. It is developed by Agent-Oriented Software (AOS) Pty. Ltd., a company in Melbourne, Australia, for research and industry applications.

JACK combines the Procedural Reasoning System (PRS) and Distributed Multi-Agent Reasoning System (dMARS) architectures. It adopts the BDI agent model for its agent architecture. It is not bound to any particular ACL and can implement KQML or FIPA.

JACK facilitates the planning process via providing JACK Plan Language (JPL) and graphical planning tools. However, it is not an open source software and licensed by the sponsored company.

3.2.5. RETSINA

It is developed via the Intelligent Software Agents Lab at Carnegie Mellon University [57]. RETSINA is a multi-agent system infrastructure that engineers heterogeneous autonomous agents. It is designed to cover a wide range of application domains. The types of agents in RETSINA are interface agent, task agent, information agent and middle agent. Figure 11 shows the architecture of RETSINA agent.

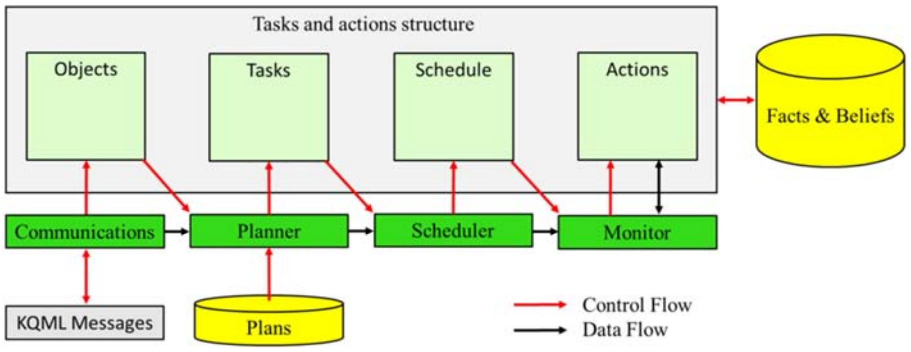


Figure 11. *RETSINA agent architecture [57].*

3.2.6. AnyLogic

AnyLogic is a multi-method tool that supports agent-based simulation modeling. It also supports the simulation of system dynamics and discrete event simulation methodologies. AnyLogic is developed by XJ Technologies and it is free for the use of educational purposes [58]. It is widely used as a research and development tool for different fields including optimization, strategic planning, logistics, forecasting and project management. Figure 12 shows AnyLogic modeling and simulation architecture.

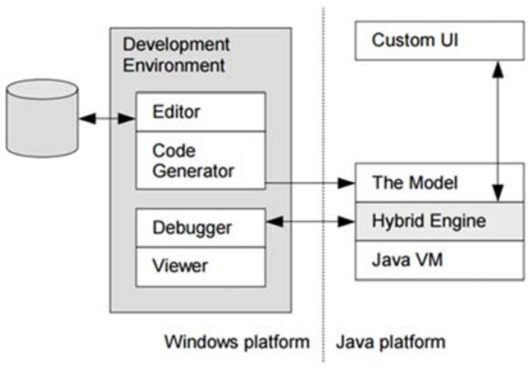


Figure 12. *AnyLogic architecture [58].*

AnyLogic is written in Java and 7.3.1 is its last version. It works based on a graphical modeling language and compiles the designed models into Java codes. It allows users to extend some of its simulation models by adding in Java codes. The extensions include modifying its agents and multi-agent system architectures and the simulation statistical outputs. The anyLogic architecture consists of Windows platform that contains the development environment and Java platform that handles the Windows platform.

The structure of AnyLogic projects consists of three core classes: The Main class, Agent classes, and Experiment classes. These classes hold the user contributions that include defining the environment, objects and assumptions of a project. Finally, AnyLogic is supported by many standard libraries that facilitate the simulated environment to users including Road Traffic, Pedestrian, Fluid and Rail environments.

3.2.7. JADE

Java Agent DEvelopment (JADE) framework is one of the widely-used platforms to develop agent-based applications. JADE is developed via the Research and Development department of Telecon Italia. It is purely written in Java and inherits features like the flexibility to work with other platforms via the aid of Java Virtual Machine [26]. JADE is in compliance with the FIPA specifications. It is an open source, well-documented and easy to use the tool.

JADE provides an abstract agent architecture that flexibly engineers different agent architectures on top of it. JADE architectural elements include the main container, which is the bootstrap point of JADE that registers other containers in a platform, distributable containers that join to the main container and host to execute the agents. Figure 13 schematizes JADE main architectural elements in a UML diagram.

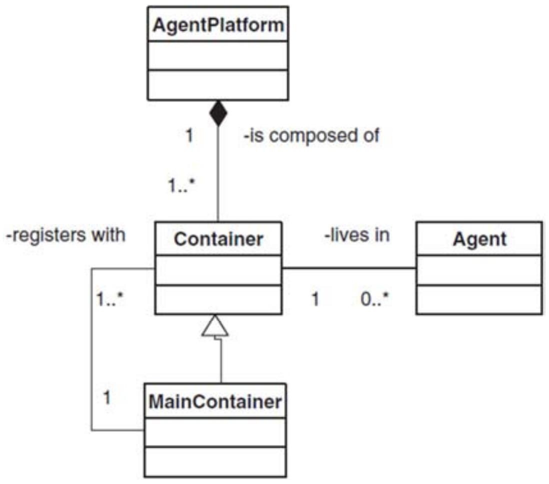


Figure 13. *JADE main architectural elements [26].*

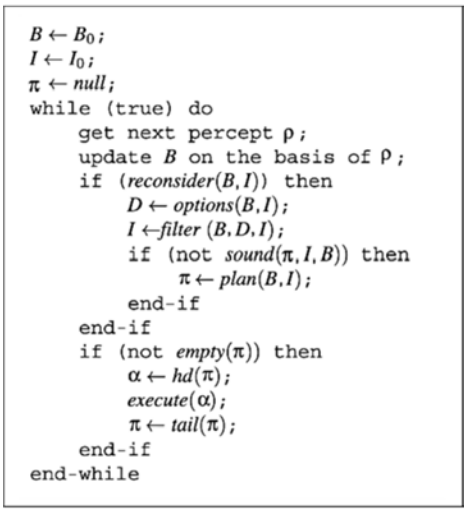
4. Software Agents Research

Agents research is meant to find solutions to many modern systems challenges. This results in many areas that are covered by software agent research and development. Hence, software agent challenges are replicating over time. This section presents a number of software agents' active research topics. They include commitment, deliberation, situation awareness, adjustable autonomy, collective intelligence, norm, emotion, morality, and sincerity. They are illustrated in the following subsections.

4.1. Commitment

A goal-directed agent commits to a plan in order to achieve goals. Apparently, there is no optimal strategy to design an agent's commitment and its reconsideration is constrained as it is influenced by different factors such as agents' mental ability, accessibility to environments, determinism of actions successfulness and environment dynamism [31] [41]. There are three main commitment strategies [27] [41] [59]:

- Bold commitment: agents never reconsider its commitment to the committed plan. The agent that used this strategy is called a *bold* agent. The *bold* agent has a reactive architecture. The *bold* agent is found to be more efficient in a less dynamic environment.
- Cautious commitment: agents reconsider their plan for every new option's occurrence. The agent that used this strategy is called a *cautious* agent. The *cautious* agent has a deliberative architecture. The *cautious* agent is more efficient in a highly dynamic environment.
- Balanced commitment: agents balance between the boldness and the cautiousness of the commitment. The agent that used this strategy is called a *balanced* agent. The *balanced* agent has a deliberative architecture with a reactive controlling mechanism. The *balanced* agent has the flexibility to efficiently perform in environments with different levels of dynamism.



```
$B \leftarrow B_0$;
$I \leftarrow I_0$;
$\pi \leftarrow null$;
while (true) do
    get next percept $\rho$;
    update $B$ on the basis of $\rho$;
    if ($reconsider(B, I)$) then
        $D \leftarrow options(B, I)$;
        $I \leftarrow filter(B, D, I)$;
        if (not $sound(\pi, I, B)$) then
            $\pi \leftarrow plan(B, I)$;
        end-if
    end-if
    if (not $empty(\pi)$) then
        $\alpha \leftarrow hd(\pi)$;
        $execute(\alpha)$;
        $\pi \leftarrow tail(\pi)$;
    end-if
end-while
```

Figure 14. *A BDI agent control loop [41].*

Different mechanisms are proposed to achieve a balanced commitment strategy. Kinny and George [59] used a plan utility function to measure the commitment degree in order to balance the commitment. Schut et al. [41] proposed a BDI agent that performs observation, deliberation, planning, and execution functions of a control loop [31]. The agent is supported by a commitment strategy that consists of reconsider function to decide on the deliberation state as shown in Figure 14. Ermon et al. [60] used the Markov decision processes and non-linear utility maximization function with the rewards strategy to determine planning preferences of agents. Other works that illustrate agents' commitment are McBurney & Luck [10], Ceballos et al. [34] and Pokahr et al. [61].

Schut et al. [41] experimentally proved that the environment's degree of dynamism is the major factor of intention reconsideration efficiency when neglecting the agent's ability to perceive and reason. Hence, it is inappropriate to estimate the efficiency of the agent's action without identifying the environment transition states specifications. An important assumption to be measured in commitment reconsideration is that the commitment reconsideration decision cost must be much lesser than the re-planning process [61].

4.2. Deliberation

Deliberation is an aspect of an autonomous agent or multi-agent systems since the agents are capable of making autonomous decisions [28] [61] [62]. When agents have delegated a task, their deliberation process is concerned with selecting a proper action or actions based on their knowledge about the world to complete the task [12] [62] [63]. An example of agents' deliberation process is shown in Figure 15.

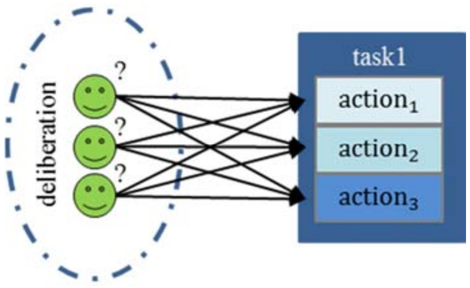


Figure 15. *Agents' deliberation [63].*

Agents' deliberation process might involve communication, filtering (e.g., probability) and selection (e.g., utility or cost) functions as proposed in decision theories [60] [64]. Moreover, goal deliberation needs to be associated with a commitment strategy [62] [63].

Deliberation research in the literature has a direct and indirect context [65]. The direct context studies individuals' behaviors when performing the deliberation process while the indirect (or inferred) context concerns with analyzing the deliberation contents [63]. The deliberation methods have different formulations and each of which achieves different objectives. Some of the deliberation measures' objectives assist:

- To prevent goals conflicts as in [61].
- To determine commitment strategies as in [62].
- To determine tasks' and actions' complexity granularity as in [63].
- To determine tasks' and actions' deliberating time/length as in [64].
- To determine agreement and disagreement levels of a decision-making group [65].
- To determine decisions' accuracy [66].

Pokahr et al. [61] proposed Easy Deliberation which is a goal deliberation strategy for a BDI agent. The strategy specifies the relationships between goals in a simple and in an intuitive manner the relationships between goals. It is formulated at the architectural level to prevent goals conflicts. The goal deliberation strategy is activated if there is a new option or if a considered option is no longer valid. Figure 16 is an agent algorithm that performs the Easy Deliberation strategy.

```
initialize-state();
repeat
options := option-generator (event-queue);
selected-options := deliberate (options);
update-intentions (selected-options);
execute ();
get-new-external-events ();
drop-successful-attitudes ();
drop-impossible-attitudes ();
end repeat
```

Figure 16. *An Agent with Easy Deliberation strategy [61].*

Dastani et al. [62] showed how a BDI agent is insufficient in performing deliberation. They proposed a model of deliberation that uses meta-language to implement the deliberation process in the BDI agent. They argue that updating the agent's mental attitude frequency affects its deliberation measures. They show that the agent's deliberation level characterizes its autonomy capabilities.

Mostafa et al. [63] proposed a mechanism to measure tasks' and actions' deliberation intensities for agents. Basically, the number of actions that agents need to do to complete a particular task determines the task's deliberation intensity. They assume that the actions of a system have three types: non-deliberative, pseudo-deliberative and deliberative actions. The deliberation intensity of a task or action determines its complexity granularity. They categorize the tasks and actions based on the tasks' and actions' deliberation intensities into high-complex, intermediate- complex and low- complex categories. Ultimately, the complexity granularity identifies different aspects of the agents and the actions including the deliberation length and autonomy configuration, distribution and adjustment parameters.

Larson and Sandholm [64] proposed a set of intuitive properties as a basis for explicit agent deliberation setting which are Preference formation-independent, Deliberation-proof, and Non-misleading. They use a preference mechanism to model agent deliberation process in which an agent's resources controls its preferences. The deliberation is measured via a cost function that limits agent's resources accessing options. The cost function operates according to the agent's performance profile. They conclude that it is difficult to produce a deliberation mechanism that satisfies the three proposed properties as they somehow clash.

Lizzeri and Yariv [66] proposed a model of deliberation measure that captures some key features of group members' deliberation process. The model uses key features to customize deliberation length, agreement, and disagreement processes and enhancing the decision accuracy within the deliberation and actions selection decision phases.

4.3. Situation Awareness

The principle of involving humans and agents to carry out some system's initiative manifests the notion of the intelligent interactive system [44] [67]. Improving agents' awareness of situations has aroused a lot of interests in agent research [68] [69]. In this section, we explore the proposed agent situation awareness mechanisms in the literature. Our aim is to exploit situation awareness mechanisms in agent models. The mechanisms improve agents' decision-making via decomposing events' contexts.

Situation awareness reflects the situatedness of agents in environments. An agent's decision in a particular event is formed based on its interpretation of the situation's context of the event [44] [70]. It implies that an agent acting on a situation in an environment by means of reasoning and acquired knowledge about the situation's parameters and performing situated actions [71]. Situated action means that the decision of an action selection regarding a situation is prone to the situation's constraints [68].

An agent's practical reasoning methodology encompasses implicit situation awareness capabilities to some limited depth.

Franklin and Graesser [17] stated that "an autonomous agent is a system situated within and a part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to effect what it senses in the future." Since one of the agent's core behaviors is observation, then, the perception of the surrounding is embodied in its design [44]. An agent's understanding of an event is built upon its knowledge of the event's situational elements and its interpretation (or beliefs) of the situation [70]. The agent's desires dictate its existing or newly generated intentions. The agent's intentions processing forms its projection of the situation's future scenarios (proactivity) [68].

However, building a mechanism that controls the behavior of agents toward an optimized behavior is the main challenge in modeling agents [17]. Agents need to be modeled to deal with the possibility of actions' failure when acting on an event, especially, in dynamic environments [28]. It implies that the agents need to understand the context of the perceived aggregated knowledge of events which is challenging [69]. Nevertheless, a situation awareness approach is found to be very useful in enhancing decisions [68].

Wardziński [71] emphasized the importance of situation awareness mechanism in improving an agent's knowledge and minimizing its action risk of failure, especially, in dynamic and uncertain environments [72]. McAree and Chen [73] claimed that the situation awareness capability enhancement of a system leads to the system's autonomy improvement. Hoogendoorn et al. [40] deployed a situation awareness mechanism on an agent's belief optimization in which the agent's degree of awareness on a situation is signified by an activation value of belief. Their aim is to generate complex beliefs from the observed beliefs that enable the agent to perform projection to future situations.

In summary, explicit situation awareness activities can be modeled and exploited for the agent to further improve its autonomy, especially when dealing with uncertainties in the event [68] [73]. This mechanism is needed to enable the agent to derive precise conclusions from an observed situation [40]. It assists the agent to reason over its decisions of actions and the actions outcomes [44]. Ultimately, equipping agents with situation awareness capabilities leads to produce enhanced interactive autonomous systems.

4.4. Adjustable Autonomy

The abstract definition of autonomy is the ability of a system to make unaided decisions [74]. However, a key aspect of the advanced autonomous system is its ability to concurrently communicate and cooperate with other systems in order to fulfill different situations' constraints [75]. The cooperation between the systems and their entities in an environment may operate at different levels of intelligence and with different degrees of autonomy [43]. Human and agent cooperation is a good example of entities in interaction and cooperation [44] [75].

Consequently, it is unreliable to completely make an agent handles its autonomous behavior only by its internal state [19] [76]. In some situations, agents are found incapable, uncertain, unpredictable and/or unauthorized and ultimately unreliable to decide on a situation and achieve tasks [77] [78]. Without adjustable autonomy, when an agent responds to a particular event it always pursues the corresponding tasks beyond oversight or intervention of control from others. Therefore, an agent's autonomy needs to be dynamically updated (internally or externally) to perform the interaction [67] [74] [76].

Managing agents' autonomy in dynamically interactive systems is a challenging task [1] [28] [79]. Giving an agent absolute control over its autonomy is a risky practice. The agent makes decisions based on its local state and theoretically the agent cannot always make optimal decisions unless it has global knowledge about its environment and this is impossible [12] [67]. For instance, we as humans need some help and support in doing our job. Subsequently, the fear that autonomous agents' behavior could wreak havoc and cause harm, fatalities, or catastrophes justifies the need for adjustable autonomy. Hence, many autonomy researchers adopt the adjustable autonomy approach, e.g., [1], [12], [67], [72], [76] and [79].

There are many opinions and diverse understanding of what adjustable autonomy is and how it can be efficiently formulated [72]. The adjustable autonomy or flexible autonomy is proposed to give agents a variable autonomy [76] [80]. It gives the option of agents working in different levels of autonomy and prone to human oversight or intervention to promote reliability [80] [81]. Moffitt et al. [82] define adjustable autonomy as "*a mechanism through which an operator delegates authority to the system that can be taken back or shared dynamically throughout mission execution*." Consequently, autonomy adjustment is a process of changing an agent's decision-making parameters, based on a situation of exigency, so as to influence the agent's decision to satisfy the situation needs [74].

While adjustable autonomy is considered as a successful approach, it shows some deficiencies that are crucial, especially, in systems where many players are involved in its control [67] [72]. The dependency that it provides has a positive impact by increasing system initiative level (i.e., human contributions) and negative impact in dealing with dynamic [80]. Apparently, the ensuing continuous interrupts make such systems dependent and slow, especially, in systems where a dialogue is utilized in the sequential decision of problem-solving, e.g., communication delays [75] [83].

In this technology, there are still many aspects that need to be further studied and improved. Therefore, Schurr et al. [83], among others, stated that "adjustable autonomy in teams is an inherently distributed and complex problem that cannot be solved optimally and completely online." The main challenges in adjustable autonomy formulation are addressed in the literature and some of which are:

- Determining dynamic autonomy distribution and adjustment mechanisms for agents that improve their performance when encountering some environment constraints.
- Determining variable autonomy levels of operations for agents that satisfy dynamic and complex environments.
- Determining autonomy degrees of an agent that enables

it to interact and successful perform.

- Determining when and how a human should intervene.
- Directing autonomy distribution and adjustment of a system's operations to satisfy its users' preference.
- Reducing the disturbance of a system during autonomy distribution and adjustment to avoid the destabilization of the system.
- Testing and validating the viability of autonomous systems when operating in dynamic environments.

4.5. Collective Intelligence

Intelligence is the ability of an entity to strategically deploy its knowledge to solve problems. The logical ability is influenced by the knowledge depth that the entity had comprehended throughout experiences [84]. Collective intelligence incorporates entities' individual diversity of knowledge and experiences in the pursuit of a common goal [85]. It is the transaction of all the collective knowledge intersections to form a meaningful solution. The process of knowledge interactions and intersection represents the emergent process of collective intelligence in reaching common goals [86].

Knowledge is transferred from one entity to another through communication skills. Consequently, an inherent process guides the transformed knowledge into decision-making options [87]. Figure 17 characterizes the process of the intersection of knowledge in which PI_1 and PI_2 denote personal intelligence of two individuals.

Bio-inspired collective intelligence algorithms such as bee colony and ant colony algorithms are introduced to solve challenging optimization problems. Humans also are biological agents that communicate formally and informally to execute the task and solve problems [88]. Seemingly, this collective composition of cognition and behavior formulate efficient solution resulting in a dynamic process of intelligent group discussion, reasoning and decision-making, thus optimizing problem-solving method [86].

Collective intelligence in human entities is a valid and profound idea if the emergent intelligence resulting from discussions leads to successful outcomes [84]. A successful outcome is demonstrated by the achievement of a common goal. Embarking upon the theory of collective intelligence that emerges from the intellectual discussions amongst human entities reveals a higher rate of success in goals attainment [88]. As an example, Gunasekaran et al. [89] discussed the topology of a collective intelligent mechanism and its influence in the internet technology. The topology provides a case structure of how collective intelligence is used as an online mechanism in improving issues related to climate change. As such, they look at the notion of collective intelligence from the perspectives of social sciences.

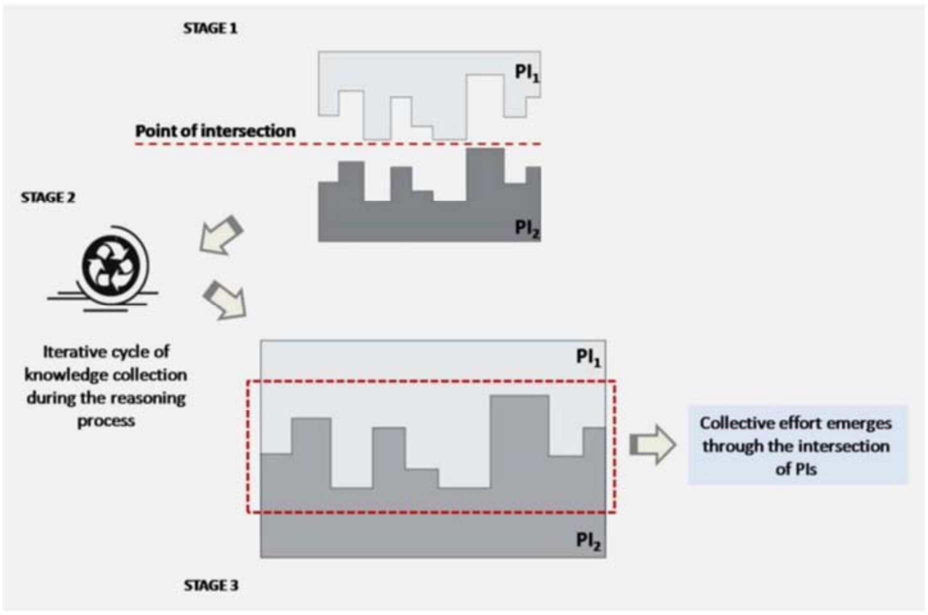

STAGE 1
PI1
Point of intersection
PI2
STAGE 2
Iterative cycle of knowledge collection during the reasoning process
PI1
Collective effort emerges through the intersection of PIs
PI2
STAGE 3

Figure 17. *The intersection of knowledge [84].*

In a multi-agent environment, a series of interactions through communications emerges to determine the flow of actions that each agent should execute in order to accomplish its individual goal [88]. Ultimately, each goal aligns to manifest a common goal. In a collective multi-agent environment, these agents retain only one common goal from the start, which is achieved through a series of processes that involve discussions, group reasoning, decision-making, and actions. Both the reasoning and decision-making phases diffuse knowledge in the form of proven beliefs between these agents [84].

Swarm intelligence algorithms of multi-agent systems are formed based on specialized abilities and agreed upon outcomes. The algorithms manifest collective behaviors in observation, navigation and collective decision-making [87]. These collective behaviors impose the privileges of communication, argumentation and group goal attainment [86]. The collective behavior process is fundamental so as to coherently fulfill their group optimal goal. Sufficiency in terms of the collective behavior process is reflected by the efficiency of the goal being achieved.

Gunasekaran et al. [84] proposed a collective intelligence model of a multi-agent system that mimics the actual collective behaviors process occurs between two or more human entities. They analyze a group of humans meeting activities and extract number of collaborative intelligence attributes, parameters and goal attainment strategies. The results show that humans exhibit higher level intelligence while augmenting their intelligence may steadily improve the operations on decision making. Subsequently, the model is applied in a multi-agent system optimization problem. The model contributes significantly in optimizing solutions.

4.6. Norm

Norms are informal rules that represent behaviors for a natural or artificial community population towards specific situations. The rules usually indicate actions that are performed based on observation of facts [90]. For instance, the norm of the attendees of a formal meeting is they behave in a polite manner. Research in norms has progressed over the past decades across many fields including philosophy, sociology and artificial intelligence [91].

In multi-agent societies, the concept of norms determines the behaviors of agents. It is mainly adopted in decentralized multi-agent systems [92]. The concept is used as a means to normalize or constrain the behaviors of agents within their communities [93]. These constraints define obligatory, prohibitive or permissive behaviors to create solutions to particular problems of the multi-agent societies.

Agents are designed to achieve certain goals. The norms influence them to behave according to their societies' expectations when achieving their goals [94]. The word 'influence' indicates that the agents have complete control over their behaviors as the agents autonomously perform in their communities. Elsewise, the agents are behaving without their will and the norms intervene in their goals' achievement. Instead, the norms should only affect the means of achieving the goals. Hence, norms' trust is introduced to influences agents' compliance with the norms and to enforce the stability of the norms in normative multi-agent systems [95].

4.7. Emotion

Emotion research covers the disciplines of artificial intelligence, cognitive science, behavioral science, philosophy, phenomenology, and physiology. In artificial intelligence, it plays an essential role in supporting cognitive activities including decision-making, rational thinking, and learning [96]. It should receive better attention due to its potentials to manifest creative solutions and credible characters [97]. However, the outcomes of emotion research in the artificial intelligence discipline have not shown noticeable progress.

Recently, emotion is introduced in software agent research and applications. The integration of emotion in software agents can enhance the agents' autonomy, adoption and social interactions [98] [99]. As an example, when a human assistant agent knows that the human is upset, it avoids disturbing him/her. Tremendous efforts have been devoted to developing emotional software agents for robotics applications [97]. Other attempts include e-commerce, medical care, toys, games, and domestic applications. The aim is to create a system that supports and facilitates humans' daily routines [99].

The main issue is that emotional behavior is missing in most well-known agent models. Modeling emotions in an agent require identifying its emotional factors including causes, effects, and drivers. It also requires equipping the agent with mechanisms to expresses its emotions and captures others emotions [99]. The emotion model enables the agent to manage its emotions internally by changing its mental attitudes or externally by translating them into actions. Certainly, when an agent is equipped with emotions, its emotional states influence its decisions to a certain instinct. For instance, when a civilian agent is encountering threats, its emotional states of risk and fear affects its commitment to pursuing its goal more than a soldier agent. Hence their responses to the threat differ according to their emotional states' stress.

In multi-agent systems, it is found that emotion in agents can be directed to manage an individual and team goals [98]. A good example is the impact of emotions on a football team. The emotional states of the players are changed according to the changes of the game scenarios. The individual player is emotionally guided by its personal success and the overall team success. A failure of a player might lead to the failure of the team. Hence, organizing the emotions of a team play a key role in the team's success.

4.8. Morality and Sincerity

Sincerity characteristic of agents is adapted from humans' behavioral characteristic as many of other software agent characteristics. It presents humans sincere attitude in a multi-agent environment. Sincerity implies the attitude of a proactive volunteer to help others without expecting a tangible gain. Sincere behavior supports cooperation, facilitate negotiation and enforces corrective and preventive actions in multi-agent systems. It facilitates achieving tasks that involve many parties of agents [101]. It has an important effect when agents' local goals are envisioned for a global goal. The agents are willing to do the extra effort and help each other in order to ensure that the global goal can be achieved.

Depending on functions like utility or cost in a collaborating environment might create selfish and greedy agents. The agents' concern is confined to maximizing their utilities, profits or minimizing costs [102]. Previous research elaborates that applying morality to agents improves their collaboration e.g., [103], [104] and [105].

Nominating morality and sacrificing culture among different parties maintain cohesive productive framework. Ahmad et al. [106] proposed a framework of a multi-agent system with sacrifice behavior. The framework enforces a sacrifice culture of sharing resources among agents in order to complete delegated tasks. The agents comply with the sacrifice via a penalty mechanism.

Sacrifice can be requested by a third party or sincerely occurs in an agent. Sincerity represents a strong moral behavior to be imposed on agents. Jaafar et al. [107] proposed a sincerity framework that instills the sacrifice behavior in agents. They propose three types of agents: leader agent, problem agent and helping agent. The framework enables the agents to work as a team and helps each other while carrying out their own tasks. Agents teaming conditions are extracted according to a delegated task's level of importance and urgency. The framework is able to formulate a sincere behavior in the agent-based systems.

5. Conclusion

This paper presents a concise literature review on software agents and multi-agent systems. It results from reviewing more than 300 references of the field. It summarizes and analyzes more than 100 sources including papers, articles, and books. The aim of this paper is to provide a quick start to new researchers of the field.

The paper covers the issues of software agents that include agents' definitions, properties, types, terminology, architectures, and models. Subsequently, the paper discusses the issues of multi-agent systems that include conceptualization, architectures, properties and development platforms. Finally, it explores the currently active research topics of the field including agents' commitment, deliberation, situation awareness, adjustable autonomy, collective intelligence, norm, emotion, morality, and sincerity.

Acknowledgment

This research project is sponsored by the Malaysian Ministry of Higher Education (MoHE) under the Exploratory Research Grant Scheme (ERGS) No. ERGS/1/2012/STG07/ UNITEN /02/5.

References

[1] Mostafa, S. A., Ahmad, M. S., Annamalai, M., Ahmad, A., & Gunasekaran, S. S. (2013). A dynamically adjustable autonomic agent framework. *Advances in Intelligent Systems and Computing*, Springer Verlag, 206, 631-642.

[2] Mohammed, K. A., Mostafa, S. A., Ahmad, M. S., & Mahmoud, M. A. (2014, November). A qualitative analysis of human-agent functions for collaborative multi-agent system. In *Information Technology and Multimedia (ICIMU), 2014 International Conference on* (pp. 244-249). IEEE.

[3] Pătraşcu, M., & Drăgoicea, M. (2014). Integrating agents and services for control and monitoring: managing emergencies in smart buildings. In *Service Orientation in Holonic and Multi-Agent Manufacturing and Robotics* (pp. 209-224). Springer International Publishing.

[4] Byrski, A., Dreżewski, R., Siwik, L., & Kisiel-Dorohinicki, M. (2015). Evolutionary multi-agent systems. *The Knowledge Engineering Review*, *30* (02), 171-186.

[5] Maes, P. (1993). Modeling adaptive autonomous agents. *Artificial life*, *1* (1-2), 135-162.

[6] Maes, P. (1995). Artificial life meets entertainment: Lifelike autonomous agents. *Communications of the ACM*, *38* (11), 108-114.

[7] Jennings, N. R., Sycara, K., & Wooldridge, M. (1998). A roadmap of agent research and development. *Autonomous agents and multi-agent systems*, *1* (1), 7-38.

[8] Magill, K., & Erden, Y. J. (2012). Autonomy and desire in machines and cognitive agent systems. *Cognitive Computation*, *4* (3), 354-364.

[9] Chira, C. (2003). Software agents. *IDIMS Report*, *21*.

[10] McBurney, P., & Luck, M. (2007). The agents are all busy doing stuff!. *IEEE Intelligent Systems*, (4), 6-7.

[11] Wooldridge, M. (2009). *An introduction to multi-agent systems*. John Wiley & Sons.

[12] Ehlert, P. (2001). Intelligent driving agents: The agent approach to tactical driving in autonomous vehicles and traffic simulation.

[13] Hewitt, C. (1977). Viewing control structures as patterns of passing messages. *Artificial intelligence*, *8* (3), 323-364.

[14] Florian, R. V. (2003). Autonomous artificial intelligent agents. *Center for Cognitive and Neural Studies (Coneural), Str. Saturn*, *24*, 3400.

[15] Wooldridge, M., & Jennings, N. R. (1995). Intelligent agents: Theory and practice. *The knowledge engineering review*, *10* (02), 115-152.

[16] Nwana, H. S., & Ndumu, D. T. (1997). An introduction to agent technology. In *Software Agents and Soft Computing Towards Enhancing Machine Intelligence* (pp. 1-26). Springer Berlin Heidelberg.

[17] Franklin, S., & Graesser, A. (1997). Is it an agent, or just a program?: A Taxonomy for Autonomous Agents. In *Intelligent agents III agent theories, architectures, and languages* (pp. 21-35). Springer Berlin Heidelberg.

[18] Shoham, Y. (1997). An overview of agent-oriented programming. *In Software agents. Bradshaw, J. M editor, AAAI Press / The MIT Press*, Cambridge, Massachusetts.

[19] Bradshaw, J. M. (1997). *Software agents*. MIT press.

[20] Hexmoor, H., Castelfranchi, C., & Falcone, R. (2003). A prospectus on agent autonomy. In *Agent Autonomy* (pp. 1-10). Springer US.

[21] Bhatia, R. (2014). Intelligent agents: A Deep insight. *IJCAIT*, 4 (2), 11-13.

[22] Nwana, H. S. (1996). Software agents: An overview. *The knowledge engineering review*, *11* (03), 205-244.

[23] Nwana, H. S., & Wooldridge, M. (1996). Software agent technologies. *BT Technology Journal*, *14* (4).

[24] Franklin, S., & Graesser, A. (1997). Is it an agent, or just a program?: A Taxonomy for Autonomous Agents. In *Intelligent agents III agent theories, architectures, and languages* (pp. 21-35). Springer Berlin Heidelberg.

[25] Mohammed, K. A., Ahmad, M. S., Mostafa, S. A., & Firdaus, M. A. (2012). A Nodal Approach to Modeling Human-Agents Collaboration. *International Journal of Computer Applications*, *43* (12), 33-40.

[26] Bellifemine, F. L., Caire, G., & Greenwood, D. (2007). *Developing multi-agent systems with JADE* (Vol. 7). John Wiley & Sons.

[27] Georgeff, M., Pell, B., Pollack, M., Tambe, M., & Wooldridge, M. (1999). The belief-desire-intention model of agency. In *Intelligent Agents V: Agents Theories, Architectures, and Languages* (pp. 1-10). Springer Berlin Heidelberg.

[28] Durand, B., Godary-Dejean, K., Lapierre, L., & Crestani, D. (2009). Inconsistencies evaluation mechanisms for a hybrid control architecture with adaptive autonomy. In *CAR'09: 4th National Conference on Control Architectures of Robots*.

[29] Brooks, R. (1986). A robust layered control system for a mobile robot. *Robotics and Automation, IEEE Journal of*, *2* (1), 14-23.

[30] Brooks, R. A. (1991). Intelligence without representation. *Artificial intelligence*, *47* (1), 139-159.

[31] Schumann, R. (2011). Engineering coordination: a methodology for the coordination of planning systems.

[32] Nau, D. S. (2007). Current trends in automated planning. *AI magazine*, *28* (4), 43.

[33] Molineaux, M., Klenk, M., & Aha, D. W. (2010). *Goal-driven autonomy in a Navy strategy simulation*. KNEXUS Research Corp Springfield VA.

[34] Ceballos, A., Bensalem, S., Cesta, A., De Silva, L., Fratini, S., Ingrand, F., Ocon, J., Orlandini, A., Py, F., Rajan, K., &Rasconi, R. (2011). A goal-oriented autonomous controller for space exploration. *ASTRA*, *11*.

[35] Wilson, M. A., McMahon, J., & Aha, D. W. (2014). Bounded expectations for discrepancy detection in goal-driven autonomy. In *Workshops at the Twenty-Eighth AAAI Conference on Artificial Intelligence*.

[36] Muñoz-Avila, H., Aha, D. W., Jaidee, U., Klenk, M., & Molineaux, M. (2010). Applying goal driven autonomy to a team shooter game. In *FLAIRS Conference*.

[37] Georgeff, M. P., & Lansky, A. L. (1987). Reactive reasoning and planning. In *AAAI* (Vol. 87, pp. 677-682).

[38] Rao, A. S., & Georgeff, M. P. (1995). BDI agents: From theory to practice. In *ICMAS* (Vol. 95, pp. 312-319).

[39] Bratman, M. (1987). Intention, plans, and practical reason.

[40] Hoogendoorn, M., Van Lambalgen, R. M., & Treur, J. (2011). Modeling situation awareness in human-like agents using mental models. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence* (Vol. 22, No. 1, p. 1697).

[41] Schut, M., Wooldridge, M., & Parsons, S. (2004). The theory and practice of intention reconsideration. *Journal of Experimental & Theoretical Artificial Intelligence*, *16* (4), 261-293.

[42] Pantelis, P. C., Baker, C. L., Cholewiak, S. A., Sanik, K., Weinstein, A., Wu, C. C., Tenenbaum, J. B., & Feldman, J. (2014). Inferring the intentional states of autonomous virtual agents. *Cognition*, *130* (3), 360-379.

[43] Mostafa, S. A., Ahmad, M. S., Ahmad, A., Annamalai, M., & Mustapha, A. (2014). A dynamic measurement of agent autonomy in the layered adjustable autonomy model. *Studies in Computational Intelligence*, Springer-Verlag, 513, 513, 25-35.

[44] Mostafa, S. A., Ahmad, M. S., Tang, A. Y., Ahmad, A., Annamalai, M., & Mustapha, A. (2014). Agent's autonomy adjustment via situation awareness. *Lecture Notes in Computer Science*, Springer-Verlag, 8397, 443-453.

[45] Ferguson, I. A. (1991). Toward an architecture for adaptive, rational, mobile agents. *ACM SIGOIS Bulletin*, *13* (3), 15.

[46] Müller, J. P. (1996). *The design of intelligent agents: a layered approach* (Vol. 1177). Springer Science & Business Media.

[47] Kong, L., & Xiao, L. (2007). A multi-layered control architecture of intelligent agent. In *Control and Automation, 2007. ICCA 2007. IEEE International Conference on* (pp. 1454-1458). IEEE.

[48] Wallace, S. A., & Henry, M. (2008). Towards a generic infrastructure to adjust the autonomy of Soar agents. In *FLAIRS Conference* (pp. 119-120).

[49] Barber, K. S. (1996). The architecture for sensible agents. In Proceedings of the International Multidisciplinary Conference, Intelligent Systems: A Semiotic Perspective (pp. 49-54).

[50] Torreño, A., Onaindia, E., & Sapena, Ó. (2015). An approach to multi-agent planning with incomplete information. *arXiv preprint arXiv:1501.07256*.

[51] Andreadis, G., Bouzakis, K. D., Klazoglou, P., & Niwtaki, K. (2014). Review of Agent-Based Systems in the Manufacturing Section. *Universal Journal of Mechanical Engineering*, *2* (2), 55-59.

[52] Parunak, H. V. D. (1997). " Go to the ant": Engineering principles from natural multi-agent systems. *Annals of Operations Research*, *75*, 69-101.

[53] Odell, J. (2002). Objects and agents compared. *Journal of object technology*, *1* (1), 41-53.

[54] Fisher, M. (1994). A survey of Concurrent METATEM—the language and its applications. In *Temporal Logic* (pp. 480-505). Springer Berlin Heidelberg.

[55] Bradshaw, J. M., Dutfield, S., Benoit, P., & Woolley, J. D. (1997). KAoS: Toward an industrial-strength open agent architecture. *Software agents*, 375-418.

[56] Howden, N., Rönnquist, R., Hodgson, A., & Lucas, A. (2001). JACK intelligent agents-summary of an agent infrastructure. In *5th International conference on autonomous agents*.

[57] Sycara, K., Paolucci, M., Van Velsen, M., & Giampapa, J. (2003). The retsina MAS infrastructure. *Autonomous agents and multi-agent systems*, *7* (1-2), 29-48.

[58] Grigoryev I. " AnyLogic 7 in Three Days: A Quick Course in Simulation Modeling ". [Hampton, NJ]: Kindle Edition, 2014. http://www.anylogic.com/books

[59] Kinny, D., & George, M. (1991). Commitment and effectiveness of situated agents. In *IJCAI-91* (pp. 82-88).

[60] Ermon, S., Gomes, C., Selman, B., & Vladimirsky, A. (2012). Probabilistic planning with non-linear utility functions and worst-case guarantees. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, Vol 2 (pp. 965-972). International Foundation for Autonomous Agents and Multi-agent Systems.

[61] Pokahr, A., Braubach, L., & Lamersdorf, W. (2005). A goal deliberation strategy for BDI agent systems. In *Multiagent System Technologies* (pp. 82-93). Springer Berlin Heidelberg.

[62] Dastani, M., Dignum, F., & Meyer, J. J. (2004). Autonomy and agent deliberation. In *Agents and Computational Autonomy* (pp. 114-127). Springer Berlin Heidelberg.

[63] Mostafa, S., Gunasekaran, S. S., Ahmad, M. S., Ahmad, A., Annamalai, M., & Mustapha, A. (2014). Defining tasks and actions complexity-levels via their deliberation intensity measures in the layered adjustable autonomy model. In *Intelligent Environments (IE), 2014 International Conference on* (pp. 52-55). IEEE.

[64] Larson, K., & Sandholm, T. (2005). Mechanism design and deliberative agents. In Proceedings of the fourth international joint conference on Autonomous agents and multi-agent systems (pp. 650-656). ACM.

[65] Black, L. W., Burkhalter, S., Gastil, J., & Stromer-Galley, J. (2011). Methods for analyzing and measuring group deliberation. *Sourcebook of political communication research: Methods, measures, and analytical techniques*, 323-345.

[66] Lizzeri, A., & Yariv, L. (2010). Sequential deliberation. *Available at SSRN 1702940*.

[67] Fleming, M., & Cohen, R. (2004). A decision procedure for autonomous agents to reason about interaction with humans. In *Proceedings of the AAAI 2004 Spring Symposium on Interaction between Humans and Autonomous Systems over Extended Operation* (pp. 81-86).

[68] Mostafa, S., Ahmad, M. S., Ahmad, A., & Annamalai, M. (2013). Formulating situation awareness for multi-agent systems. In *Advanced Computer Science Applications and Technologies (ACSAT),* (pp. 48-53). IEEE.

[69] Mostafa, S. A., Ahmad, M. S., Annamalai, M., Ahmad, A., & Gunasekaran, S. S. (2015). Formulating dynamic agents' operational state via situation awareness assessment. *Advances in Intelligent Systems and Computing*, Springer Verlag, 320, 545-556.

[70] Ferrando, S. P., & Onaindia, E. (2013). Context-aware multi-agent planning in intelligent environments. *Information Sciences*, *227*, 22-42.

[71] Wardziński, A. (2006). The role of situation awareness in assuring safety of autonomous vehicles. In *Computer Safety, Reliability, and Security* (pp. 205-218). Springer Berlin Heidelberg.

[72] Mostafa, S. A., Ahmad, M. S., Ahmad, A., Annamalai, M., & Gunasekaran, S. S. (2015, August). An autonomy viability assessment matrix for agent-based autonomous systems. In *Agents, Multi-Agent Systems and Robotics (ISAMSR), 2015 International Symposium on* (pp. 53-58). IEEE.

[73] McAree, O., & Chen, W. H. (2013). Artificial situation awareness for increased autonomy of unmanned aerial systems in the terminal area. *Journal of Intelligent & Robotic Systems*, *70* (1-4), 545-555.

[74] Mostafa, S. A., Ahmad, M. S., Annamalai, M., Ahmad, A., & Gunasekaran, S. S. (2013). A conceptual model of layered adjustable autonomy. *Advances in Intelligent Systems and Computing*, Springer Verlag, 206, 619-630.

[75] Jennings, N. R., Moreau, L., Nicholson, D., Ramchurn, S., Roberts, S., Rodden, T., & Rogers, A. (2014). Human-agent collectives. *Communications of the ACM*, *57* (12), 80-88.

[76] Bradshaw, J. M., Feltovich, P. J., Jung, H., Kulkarni, S., Taysom, W., & Uszok, A. (2004). Dimensions of adjustable autonomy and mixed-initiative interaction. In *Agents and Computational Autonomy* (pp. 17-39). Springer Berlin Heidelberg.

[77] Mostafa, S. A., Ahmad, M. S., Annamalai, M., Ahmad, A., & Basheer, G. S. (2013). A layered adjustable autonomy approach for dynamic autonomy distribution. *Frontiers in Artificial Intelligence and Applications*. IOS Publisher. 252, 335-345.

[78] Alzahrani, A., Callaghan, V., & Gardner, M. (2013). Towards Adjustable Autonomy in Adaptive Course Sequencing. In *Intelligent Environments (Workshops)* (pp. 466-477).

[79] Alan, A., Costanza, E., Fischer, J., Ramchurn, S. D., Rodden, T., & Jennings, N. R. (2014). A field study of human-agent interaction for electricity tariff switching. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems* (pp. 965-972). International Foundation for Autonomous Agents and Multi-agent Systems.

[80] Johnson, M., Bradshaw, J. M., Feltovich, P. J., Jonker, C. M., Van Riemsdijk, M. B., & Sierhuis, M. (2014). Coactive design: Designing support for interdependence in joint activity. *Journal of Human-Robot Interaction, 3 (1), 2014*.

[81] Johnson, M., Bradshaw, J. M., Feltovich, P. J., Jonker, C., Van Riemsdijk, B., & Sierhuis, M. (2012). Autonomy and interdependence in human-agent-robot teams. *Intelligent Systems, IEEE*, *27* (2), 43-51.

[82] Moffitt, V. Z., Franke, J. L., & Lomas, M. (2006). Mixed-initiative adjustable autonomy in multi-vehicle operations. *Proceedings of AUVSI, Orlando, Florida*.

[83] Schurr, N., Marecki, J., & Tambe, M. (2008). RIAACT: A robust approach to adjustable autonomy for human-multi-agent teams. In *Proceedings of the 7th international joint conference on Autonomous agents and multi-agent systems, Volume 3* (pp. 1429-1432). International Foundation for Autonomous Agents and Multi-agent Systems.

[84] Gunasekaran, S. S., Mostafa, S. A., & Ahmad, M. S. (2015). Knowledge Transfer Model in Collective Intelligence Theory. In *Advances in Intelligent Informatics* (pp. 481-491). Springer International Publishing.

[85] Salminen, J. (2012). Collective intelligence in humans: A literature review.*arXiv preprint arXiv:1204.3401*.

[86] Gunasekaran, S. S., Mostafa, S. A., Ahmad, M. S., & Tang, A. Y. (2015, August). Identifying variables dependency that influences a high level deliberation process in a CI-based Multi-agent System. In *Agents, Multi-Agent Systems and Robotics (ISAMSR), 2015 International Symposium on* (pp. 24-29). IEEE.

[87] Brambilla, M., Ferrante, E., Birattari, M., & Dorigo, M. (2013). Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, *7* (1), 1-41.

[88] Gunasekaran, S. S., Mostafa, S. A., & Ahmad, M. S. (2013, December). Personal and extended intelligence in collective emergence. In *Intelligent Systems Design and Applications (ISDA), 2013 13th International Conference on* (pp. 199-204). IEEE.

[89] Gunasekaran, S. S., Mostafa, S. A., & Ahmad, M. S. (2014, November). Using the Internet as a Collective Intelligence platform in harnessing issues on Climate Change. In *Information Technology and Multimedia (ICIMU), 2014 International Conference on* (pp. 130-135). IEEE.

[90] Ahmad, A., Ahmed, M., M. Yusoff, M. Z, Ahmad, M. S, & Mustapha, A. (2011). Resolving Conflicts between Personal and Normative Goals in Normative Agent Systems, The Seventh International Conference on IT in Asia 2011 (CITA 2011), pp. 153 – 158, Kuching, Sarawak, 12 – 14 July 2011.

[91] Mahmoud, M. A., Ahmad, M. S., Mohd Yusoff, M. Z., & Mustapha, A. (2014). A review of norms and normative multiagent systems. The Scientific World Journal, 2014.

[92] Savarimuthu B. T. R., S. Cranefield, M. Purvis, M. Purvis, (2010). Obligation Norm Identification in Agent Societies. Journal of Artificial Societies and Social Simulation, 13 (4).

[93] Alberti, M., Gomes, A. S., Goncalves, R., Leite, J., & Slota, M., (2011). Normative Systems Represented as Hybrid Knowledge Bases, Proceedings of the 12th International Conference on Computational Logic in Multi-agent Systems, CLIMA'11, Lecture Notes in Computer Science, pp 330-346.

[94] C. D. Hollander and A. S. Wu, "The Current State of Normative AgentBased Systems," Journal of Artificial Societies and Social Simulation, 14 (2), pp. 6, 2011.

[95] Hamid, A., Hamimah, N., Ahmad, M. S., Ahmad, A., Mahmoud, M. A., Mohd Yusoff, M. Z., & Mustapha, A. (2014, November). Trusting norms in normative multi-agent systems. In *Information Technology and Multimedia (ICIMU), 2014 International Conference on* (pp. 217-222). IEEE.

[96] Hsu, C. M., Chen, T. T., & Heh, J. S. (2014, July). Emotional and Conditional Model for Pet Robot based on Neural Network. In *Ubi-Media Computing and Workshops (UMEDIA), 2014 7th International Conference on* (pp. 305-308). IEEE.

[97] Subramainan, L., Yusoff, M. Z. M., & Mahmoud, M. A. (2015, August). A classification of emotions study in software agent and robotics applications research. In Agents, Multi-Agent Systems and Robotics (ISAMSR), 2015 International Symposium on (pp. 41-46). IEEE.

[98] Harley, J. M., Bouchet, F., Hussain, M. S., Azevedo, R., & Calvo, R. (2015). A multi-componential analysis of emotions during complex learning with an intelligent multi-agent system. *Computers in Human Behavior*, *48*, 615-625.

[99] Nair, R., Tambe, M., & Marsella, S. (2005). The role of emotions in multiagent teamwork. Who Needs Emotions, 311-329.

[100] Ngo, T. D., & Bui, T. D. (2015, January). A Vietnamese 3D taking face for embodied conversational agents. In Computing & Communication Technologies-Research, Innovation, and Vision for the Future (RIVF), 2015 IEEE RIVF International Conference on (pp. 94-99). IEEE.

[101] Velez, R. A. (2015). Sincere and sophisticated players in an equal-income market. *Journal of Economic Theory*, *157*, 1114-1129.

[102] Velez, R. A. (2013, January). Sincere and sophisticated players in the envy-free allocation problem. In *EC* (pp. 853-854).

[103] Sullins, J. P. (2006). When is a robot a moral agent?

[104] Floridi, L., & Sanders, J. W. (2011). On the morality of artificial agents. *Machine ethics*, 151-160.

[105] Floridi, L. (2013). Distributed morality in an information society. *Science and engineering ethics*, *19* (3), 727-743.

[106] Ahmad, A., Ahmed, M., Yusof, M. Z. M., Ahmad, M. S., & Mustapha, A. (2016). Resolving Conflicts between Personal and Normative Goals in Normative Agent Systems. *Journal of IT in Asia*, *4* (1), 1-12.

[107] Jaafar, N. H., Ahmad, M. S., & Ahmad, A. (2015). Operational rules for implementing sincere software agents in corrective and preventive actions environment. In *Computational Intelligence in Information Systems* (pp. 307-314). Springer International Publishing.