Agent Architecture: An Overview

Kim On CHIN[1*], *Kim Soon GAN*[2], *Rayner ALFRED*[3], *Patricia ANTHONY*[4] *& Dickson LUKOSE*[5]

[1,2,3] Center of Excellence in Semantic Agent, Faculty of Computing and Informatics, Universiti Malaysia Sabah, Jalan UMS, 88400 Kota Kinabalu, Sabah, MALAYSIA.
[4]Department of Applied Computing, Faculty of Environment, Society and Design, Lincoln University, Christchurch, NEW ZEALAND.
[5]Artificial Intelligence Center, MIMOS Berhad, Technology Park Malaysia, Bukit Jalil, Kuala Lumpur, MALAYSIA.
*Corresponding author: kimonchin@ums.edu.my, Tel: +6088-320000, Ext: 3086

Received: *20 Aug 2014*
Revised: *30 Aug 2014*
Accepted: *2 Sept 2014*
Online: *30 Dec 2014*

Keywords:
Agent; Multi-Agent; Agent Architecture; Semantic Web Technology

A b s t r a c t

Agent architecture has been one of the core components in building an agent application. Agent architecture is considered as the functional brain of an agent in making decision and reasoning to solve problem and achieving goals. This paper reviews some of the existing agent architectures such as logic-based architecture, reactive architecture, BDI architecture, hybrid architecture, cognitive architecture, and semantic architecture. The purpose of this study is to identify distinctive features of the different types of agent architectures and how they are implemented to solve real world problems.

Introduction

The advancement of the Internet technology has increased the need for distributed, concurrent, heterogeneous and dynamic application systems. Agent technology is a new paradigm suitable for developing such systems that situates and operates in a dynamic and heterogeneous environment. What exactly is an agent? To date, there is no widely accepted definition of what an agent is. In this study, an agent is referred to as an autonomous software entity that is situated in some environment where it can monitor and response to changes proactively or reactively by itself or through communication with other agents to persistently achieve certain goal/task on behalf of user or other agents (Wooldridge, 2009). An agent possesses certain distinct characteristics such as (Wooldridge & Jennings, 1995).

- Autonomous: the ability to operate without the direct intervention of human and control over its internal state.
- Social: the ability to interact with human and other agents.
- Reactive: the ability to perceive changes in the environment and response to it in a timely fashion.
- Proactive: the ability to show goal directed behavior.

Other characteristics that an agent might have include mobility, benevolent, trustworthiness, rationality, and learning capability. Mobility is the ability to travel between different hosts in a computer network. Benevolent is the characteristic agent will always perform what it is asked to do.

Trustworthiness is the characteristic agent will not deliberately communicate false information. Rationality is the characteristic agent will always never to prevent its goals being achieved. Learning capability is the ability to adapt itself to fit its environment and to the desires of its users. In a complex system, an agent may not exist alone in an environment as they may be multiple agents that are situated in the same environment. Multi-agent system is the study of systems that are made up of multiple heterogeneous software entity (agents) that interact with each other (Wei B, 1999; Shoham & Leyton-Brown, 2008). In a multi-agent system environment, agents may have common or conflicting goal to be achieved (Yu *et al.*, 2010; Durfee & Rosenschein, 1994). The interaction between agents can happen directly or indirectly. Direct communication is achieved through channel such as message passing whilst indirect communication is achieved through affecting the environment and sense by other agents (Genesereth & Ketchpel, 1994; Maes, 1997). Normally, agents that have common goal in a multi-agent system will cooperate in order to achieve the goal (Doran *et al.*, 1997; Pozna *et al.*, 2011). In the case of agents with conflicting goals, the agents will compete against each other to obtain resources for personal goal attainment (Leyton-Brown , 2003). In order for agents to cooperate and coordinate in achieving their goals, agents need to reason about when and what to do under certain circumstances.

The foundation of the agent reasoning mechanism lies in the component called agent architecture. Agent architecture is the blueprint for building an agent just like a class in object-oriented programming. Wooldridge referred to agent architecture as software architecture that is intended to support decision making process (Wooldridge, 2001). Maes described agent architecture as architecture that encompasses techniques and algorithms to support decomposing set of components and how these components interact (Maes, 1991). Agent architecture is the building block for creating an agent much like creating an object in a class. The agent architecture is the *brain* of the agent as it determines how the knowledge/information is represented in the agent. It also determines the action the agent should take based on its underlying reasoning/interpretation mechanism. Thus, different architectures used different representation approaches for their reasoning mechanism to solve a variety of problems. These architectures can be broadly categorized into three groups, the classical architecture, the cognitive architecture and the semantic agent architecture. The classical architectures include logic-based architecture, reactive architecture, BDI architecture, and hybrid architecture. The logic-based architecture is an agent architecture that uses symbolic representation for reasoning. The reactive agent architecture is a direct stimulus-response agent architecture. On the other hand, the BDI architecture is a deliberative agent architecture based on mental states characteristic such as belief, desire, and intention. The layered architecture is the hybrid of reactive and deliberative agent architecture. The cognitive architecture is based on cognitive sciences and the semantic agent architecture utilizes semantic technology.

The remainder of this paper describes the various agent architectures that can be used to build agent and multi-agent system. Section 2 discusses the logic-based architecture. The reactive agent and BDI architecture are discussed in Section 3 and 4 respectively. Section 5 describes the layered architectures whilst Sections 6 and 7 present an overview of cognitive architecture and semantic agent architecture. Finally, we conclude our discussed on agent architectures in Section 8.

Logic-Based Architecture

Logic-based architecture also known as the symbolic-based or deliberative architecture is one the earliest agent architecture that rests on the physical-symbol systems hypothesis (Newell & Simon, 1976). This classical architecture is based on the traditional artificial symbolic approach by representing and modeling the environment and the agent behavior with symbolic representation. Thus, the agent behavior is based on the manipulation of the symbolic representation.

Agent's role in this classical architecture may also be considered as theorem provers (Shardlow, 1990). The syntactical manipulation of the symbolic representation is the process of logical deduction or theorem proving. As an instance of theorem proving, the agent specifications outlines how the agent behaves, how the goals are generated and what action the agent can take to satisfy these goals. An example of logic-based architecture formalism is as follows:

- Assume that the environment is described by sentences in L and the knowledge base that contains all the information regarding the environment $KB = P(L)$ where $P(L)$ is the set of possible environments.
- For each moment of the time t, an agent's internal state is represented by $KB = \{KB_1, KB_2, KB_3 ... KB_n\}$ where $KB_i \in KB$.
- The possible environment states are represented by $S = \{s_1, s_2, ...\}$.
- An agent's reasoning mechanism is modeled by a set of deduction rules, p which are the rules of inference.
- An agent perception functions as $see: S \rightarrow P$.
- The agent's internal state is updated by a perception function where $next: KB \times P \rightarrow KB$.
- Thus, agent can choose an action from a set $A = \{a_1, a_2, ...\}$, $action: KB \rightarrow A$ which is defined in terms of deduction rules. The outcome of an agent's actions is drawn via the function do where $do: A \times S \rightarrow S$.
- The decision making process is modeled through the rules of inference p, if a $do:A$ can be derived, the A is returned as an action to be best performed, else if $do:A$ cannot be derived, a special null action is returned.

Vacuum cleaning example in (Russell & Norvig, 1995) illustrates the idea of logic-based architecture based on the specification above. The programmer has to encode the inference rules p in a way that enables the agent to decide what to do. Examples of this kind of classical agent architecture

approach include classical planning agent such as STRIPS (Fikes & Nilsson, 1971), IPEM (Ambros-Ingerson & Steel, 1988), Autodrive (Wood, 1993), Softbots (Etzioni *et al.*, 1994), Phoenix systems (Cohen *et al.*, 1989), IRMA (Bratman *et al.*, 1988), HOMER (Vere & Bickmore, 1990), and GRATE (Jennings, 1993). BDI architecture is also considered as the subset of logic-based architecture. However, due to its popularity and wide adoption of the architecture, the discussion on this particular architecture is detailed in Section 4.

Although, the simplicity and elegance of logical semantics of the logic based architecture is attractive, there are several problems associated with this approach. Firstly, the transduction problem implies the problem of translating modeling into symbolic representation. It is difficult to translate and model the environment's information into symbolic representation accurately for computation process especially complex environment. Secondly, it is also difficult to represent information in a symbolic form that is suitable for the agents to reason with and in a time constrained environment. Finally, the transformation of percepts input may not be accurate enough to describe the environment itself due to certain faults such as sensor error, reasoning error and etc. It is very difficult or sometimes impossible to put down all the rules for the situation that will be encountered by the agent in a complex environment since the deduction process is based on set of inference rules. The assumption in calculative rationality where the world does not change in a significant way while the agent is deliberating is not realistic. Assume that on time t_1, agent tries to reason an optimal action for that particular time. However, the reasoning result may only be available at time t_2 where the environment has already changed so much so that the optimal action for time t_1 may not be an optimal action for time t_2. Thus, due to the computational complexity of theorem proving over this approach, it is not appropriate for time constrained domain.

Building agent in logic-based approach is viewed as a deduction process. An agent is encoded as a logical theory by using specification and the process of selecting the action is through deduction process that reduces the problem to a solution such as in theorem proving. An improvement version logic based approach has been carried out in (Amir & Maynard-Reid, 2004; Amir & Maynard-Reid, 2000). In (Amir & Maynard-Reid, 2004) a logic-based AI architecture is implemented on Brooks' Subsumption architecture. In the implementation of this architecture, different layers of control is axiomatized in First-Order Logic (FOL), thus, independent theorem provers are used to derive each layer's output given its input. This architecture proved the versatility of the theorem provers which allow them to realize complex tasks, while keeping individual theories simple (Amir & Maynard-Reid, 2000).

Reactive Architecture

Reactive agent architecture is based on the direct mapping of situation to action. It is different from the logic-based architecture where no central symbolic world model and complex symbolic reasoning are used. Agent responses to changes in the environment in a stimulus-response based. The reactive

architecture is realized through a set of sensors and effectors, where perceptual input is mapped to the effectors to changes in the environment. Brook's subsumption architecture is known as the best pure reactive architecture (Brooks, 1986). This architecture was developed by Brook who has critiqued on many of the drawbacks in logic-based architecture. Figure 1 illustrates an example of reactive architecture. The figure shows that each of the percept situation is mapped into an action which specifically responses to the percept situation.

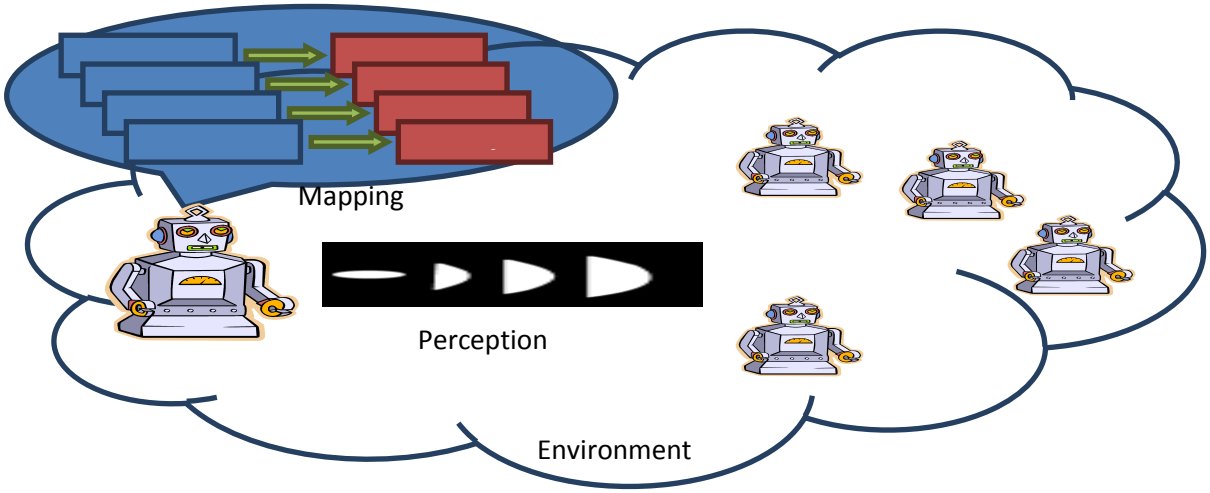

Figure 1: Reactive Architecture

The key idea of subsumption architecture is that intelligent behaviour can be generated without explicit representations and abstract reasoning with symbolic AI technique (Brooks, 1991a; Brooks, 1991b). Intelligence is an emergent property of certain complex systems. Subsumption architecture is implemented in finite state machines with different layers connected to sensors that perceive the environment changes and map the action to be performed (Brooks, 1986). A set of task-accomplishing behaviour are used in the decision making process. Each of the behaviour can be thought of as an individual function which maps changes in the environment with an action. Multiple behaviours that can be fired simultaneously is another characteristic of subsumption architecture. The subsumption architecture hierarchical structure represents different behaviours. The lowest layer in the hierarchy has the highest priority. Higher layer represent more abstract behaviour than the lower layer in the hierarchy. Complex behaviour is achieved through the combination of these behaviours. Figure 2 shows action selection in the layered architecture. In this layered architecture, the lower the layer the higher the priority. The lower layer will be the primitive behaviour and higher layer will represent a more abstract behaviour.

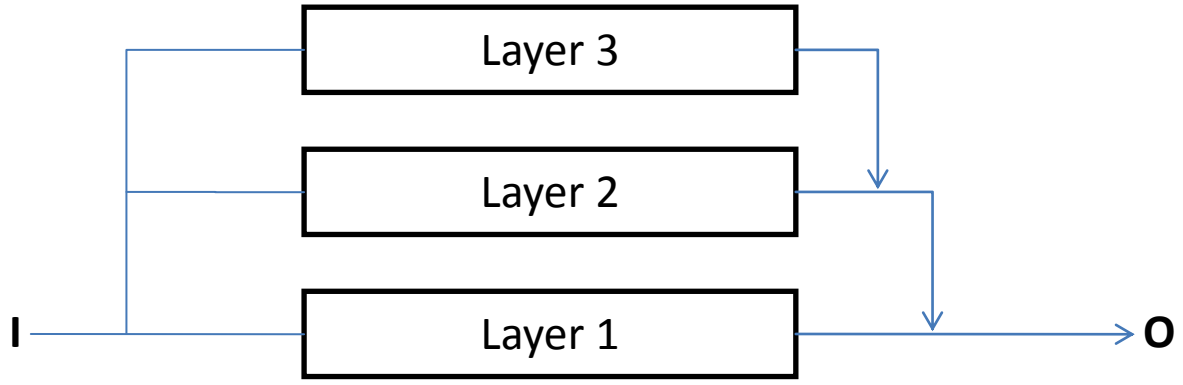

Figure 2: Action Selections in Layered Architecture

The subsumption architecture is implemented in Steel's study (Steels, 1990) for the mission to a distant planet to collect sample of rocks and minerals. A near-optimal performance can be obtained through simple adjustment and the solution is cheap in computing power and it is also robust. Chapman came up with similar approach to Brook's work in (Chapman & Agre, 1986) and is referred to as the new abstract reasoning. This approach is used in the celebrated PENGI system which simulated a computer game with central character control that can accomplish routine work with little variation (Agre & Chapman, 1987). Figure 3 shows a Pengo game in progress (Agre & Chapman, 1987).PENGI is a program written to play an arcade game called Pengo which is made up of a 2-Dmaze with unit-sized ice blocks. In this game, PENGI is programmed to move the Penguin in the game to avoid the bees attack and block slide to survive.

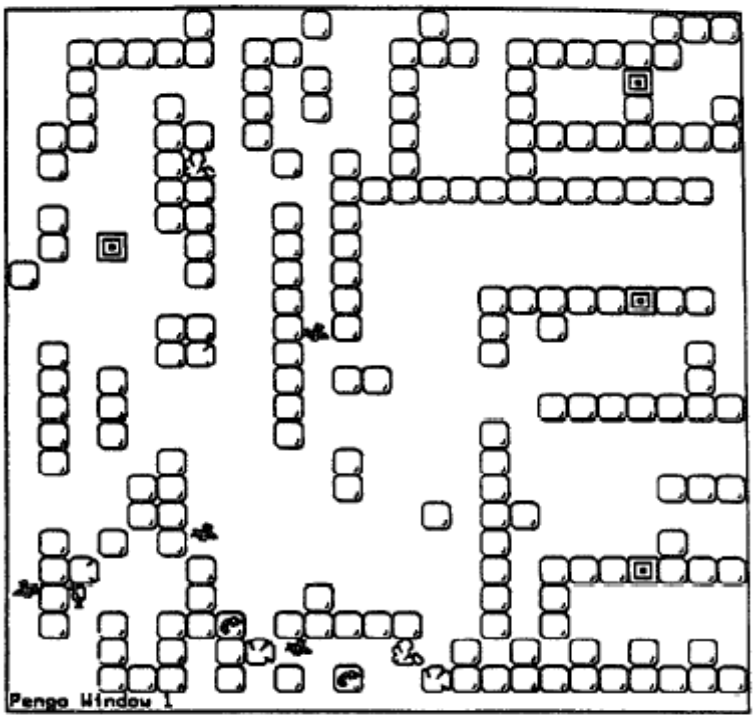

Figure 3: Pengo Game in Progress

Another similar approach is the situated automata paradigm by Rosenschein and Kaelbling (Kaelbling, 1991; Kaelbling & Rosenschein, 1990; Rosenschein, 1985; Rosenschein & Kaelbling, 1986). In this approach, the agent is specified in declarative terms such as beliefs and goals which are then compiled into a digital machine which satisfies this declarative specification. The digital machine can operate in a provably time-bounded fashion. Although the declarative terms are in beliefs and goals, the digital machine does not use any of the symbolic representation and hence, no symbolic reasoning actually occurred. The approach is also considered as reactive architecture as no symbolic reasoning actually occurred because the declarative terms has been compiled to a digital machine that reacts based on stimulus response.

Another reactive architecture is introduced by Maes called the Agent Network Architecture. In this architecture, the agent acts as a set of competence modules (Maes, 1991; Maes, 1989; Maes, 1990). These modules loosely resemble the behaviours of Brook's subsumption architecture. Pre and post-conditions determine the activation of each module based on an activation level. This activation level is stated in real value to indicate the relevancy of the module in a particular situation. Agent network architecture is somewhat similar to the neural network architecture with different meaning of the node context.

One of the advantages of reactive architecture is that it is less complicated to design and implement than logic-based architecture. An agent's behaviour is computationally tractable. The robustness of reactive architecture against failure is another advantage. Complex behaviours can be achieved from the interaction of simple ones. The disadvantages of reactive architecture include (1) insufficient information about agent's current state to determine an activation action due to modeling of environment available, (2) the processing of the local information limits the planning capabilities in long term or bigger picture and hence, learning is difficult to be achieved, (3) emergent behavior which is not yet fully understood making it even more intricate to engineer. Therefore, it is difficult to build task-specific agents and one of the solutions is to evolve the agents to perform certain tasks (Togelius, 2003). The work in this domain is referred to as artificial life.

Belief-Desire-Intention (BDI) Architecture

The BDI architecture is based on practical reasoning by Bratman's philosophical emphasis on intentional stance (Bratman, 1987). Practical reasoning is reasoning toward actions - the process of figuring out what to do. This is different from the theoretical reasoning process as it derives knowledge or reaches conclusions by using one's beliefs and knowledge. Human practical reasoning involves two activities namely deliberation and means-end reasoning. Deliberation decides what state of affairs needs to be achieved while means-end reasoning decides how to achieve these states of affairs. In BDI architecture, agent consists of three logic components referred as mental states/mental attitudes namely beliefs, desires and intentions. Beliefs are the set of information an agent has about the world. Desires are the agent's motivation or possible options to carry out the actions. Intentions

are the agent's commitments towards its desires and beliefs. Intentions are key component in practical reasoning. They describe states of affairs that the agent has committed to bringing about and as a result they are action-inducing. Forming the intentions is critical to an agent's success. BDI architecture probably is the most popular architecture (Rao & Georgeff, 1991) and Practical Reasoning System (PRS) is one of the well known BDI architectures (Georgeff & Lansky, 1986). PRS is a framework for building real-time reasoning systems that can perform complex tasks in dynamic environments. PRS used procedural knowledge representation in describing how to perform a set of actions in order to achieve goal. This architecture is based on four key data structures: beliefs, desires, intentions and plans, and an interpreter (see Figure 4). Figure 4 shows the four key data structures of BDI architecture.

In the PRS system, beliefs represent the information an agent has about its environment. Desires represent the tasks allocated to the agent corresponding to the goals that should be accomplished by agent. Intentions represent the agent's commitment towards the goals. Finally, plans specify some courses of action for the agent in order to achieve its intentions. The plans in the plan library are pre-compiled plan rather than instantaneously generated. The agent interpreter is responsible for updating beliefs from observations made from the environment, generating new desires (tasks) on the basis of new beliefs, and selecting from the subset of currently active desires to act as intentions. Lastly, the interpreter must select an action to perform the agent's current intentions and procedural knowledge.

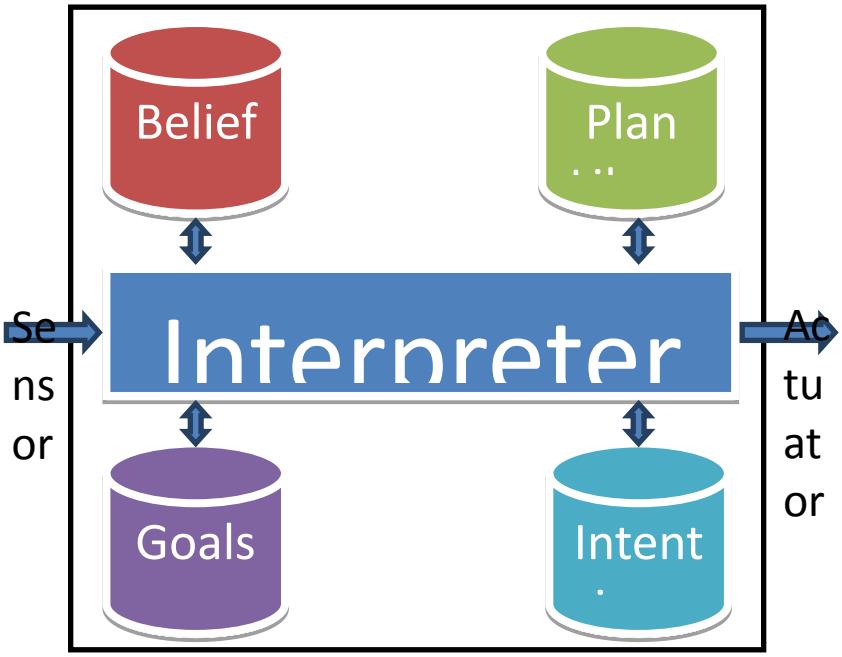

Figure 4: Practical Reasoning Systems

Many agent frameworks have been implemented in BDI architecture including:

- JAM, a hybrid intelligent agent architecture that draws upon the theories and ideas of the PRS, Structured Circuit Semantics (SCS), and Act Plan Interlingua (Huber, 1999).
- JACK, a commercial platform for developing industrial and research purpose multi-agent application in Java (Howden *et al.,* 2001).
- dMARS is a platform for intelligent software agents that makes use of the belief–desire–intention software model (BDI) for building complex, distributed, time-critical systems in C++ (d'Inverno *et al.*, 1998).

The advantages of BDI architecture are that the design of the architecture is clear and intuitive. The functional decomposition of the agent subsystem is clear and the BDI logic has formal logic properties that can be studied. However, with BDI architecture the question of how to efficiently implement the functionality in subsystem is not clear and so the agents need to achieve a balance between commitment (Rao & Georgeff, 1991) and reconsideration (Wooldridge & Parsons, 1998). If an agent did not stop to reconsider, it might be trying to achieve intention which is not achievable or no longer valid. If an agent reconsiders often, it might face the risk of not achieving them due to insufficient time working on the task.

Layered (Hybrid) Architecture

Layered (hybrid) architecture is an agent architecture which allows both reactive and deliberate agent behavior. Layered architecture combines both the advantages of reactive and logic-based architecture and at the same time alleviates the problems in both architectures. Subsystems are decomposed into a layer of hierarchical structure to deal with different behaviours. There are two types of interaction that flow between the layer namely horizontal and vertical. In the horizontal layer architecture, each layer is directly connected to the sensory input and action output (see Figure 5). Each layer is like an agent mapping the input to the action to be performed.

TouringMachine is an example of horizontally layered agent architecture. The TouringMachine agent architecture consists of three activity-producing layers: a reactive layer R, a planning layer P, and a modeling layer M (Ferguson, 1992). These three layers operate concurrently and independently in mapping the perception into action (see Figure 6). Each of the layers has its own internal computation processing mechanism.

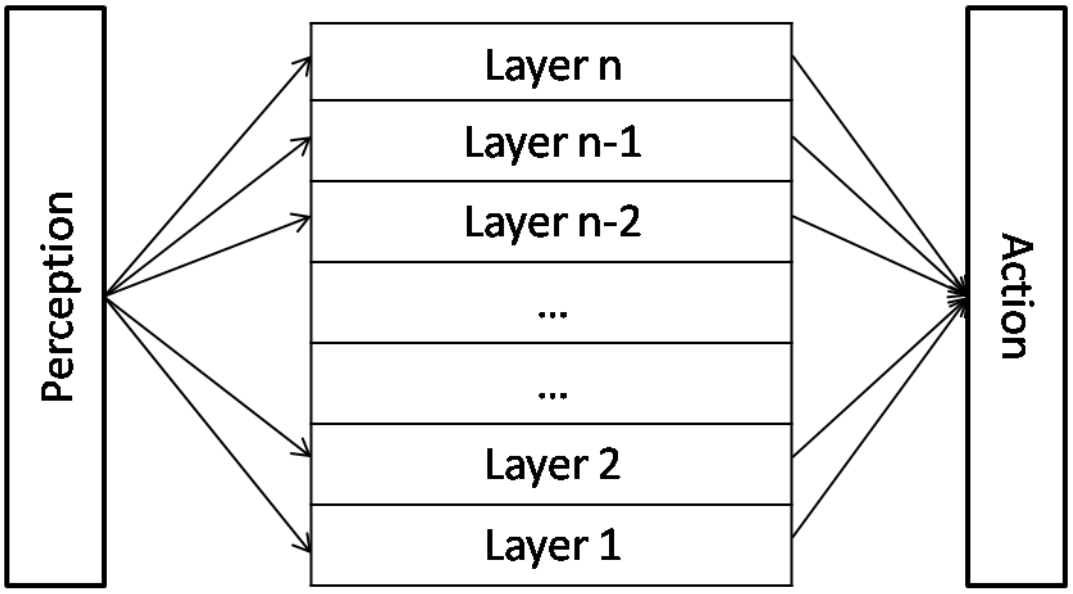

Figure 5: Horizontal Layer Architecture

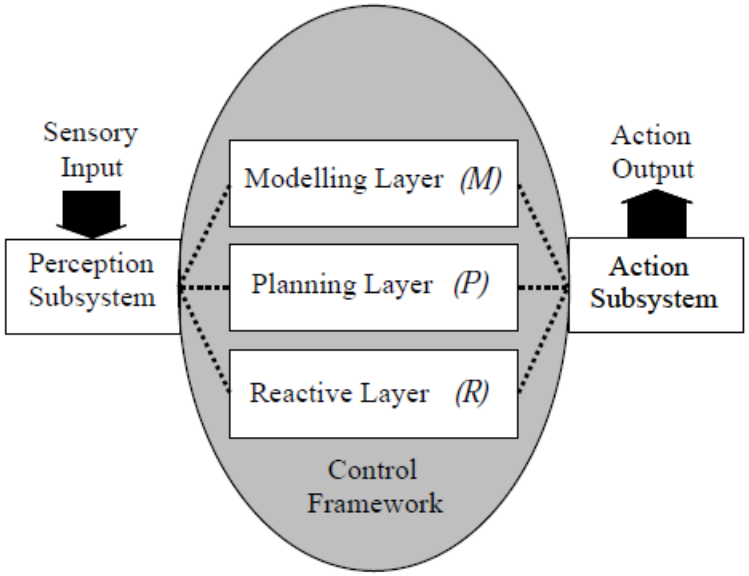

Figure 6: The TouringMachine Agent Control Architecture

The advantage of horizontal layer architecture is that only *n* layers are required for mapping to *n* different types of behaviours. However, a mediator function is used to control the inconsistent actions between layer interactions. Another complexity is the large number of possible interactions between horizontal layers–m^n (where m is the number of actions per layer).

Vertical layer architecture eliminates some of these issues as the sensory input and action output are each dealt with by at most one layer each (creating no inconsistent action suggestions). There are two types of vertical layered architectures namely one-pass and two-pass control architectures. In one-pass architecture, control flows from the initial layer that gets data from sensors to the final layer that generates action output (see Figure 7). In two-pass architecture, data flows up the sequence of layers and control then flows back down (see Figure 8).

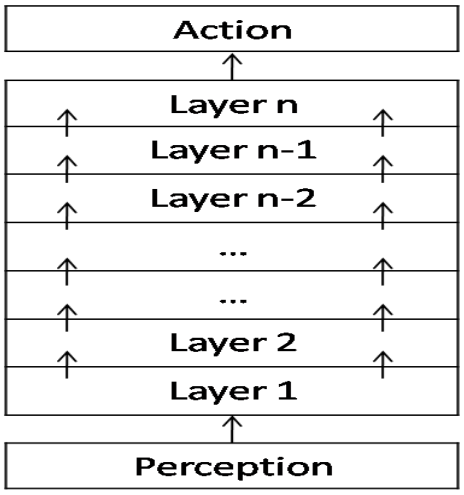

Figure 7: Vertical layer architecture: one-pass

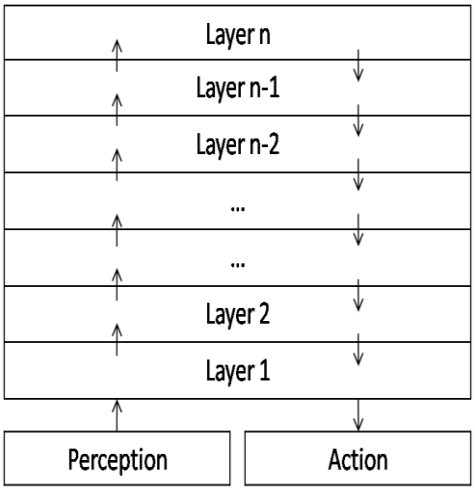

Figure 8: Vertical layer architecture: two-pass

InteRRaP is an example of vertically two-pass agent architecture (Muller & Pischel, 1993). InterRRaP comprises of three control layers namely behavior layer, local planning layer and cooperative planning layer (see Figure 9). The behaviour layer is the lowest layer in InteRRap which responses reactively towards the world model. The local planning is the middle layer that uses planning knowledge for the routine planning to achieve the agent's goal. Finally, the cooperative planning is the highest layer in InteRRaP that deals with the social interaction. The main difference between InteRRaP and TouringMachine is the interaction between layers. There are two types of flow control in InteRRaP which are the bottom up and top down. The bottom up activation deals with the lower layer which passes control to the higher layer when it cannot process the current situation. Top down execution deals with the higher layer which uses the action execution of the lower layer to achieve goals and tasks. There are two general functions that are implemented in each layer known as the situation recognition and goal activation function and planning and scheduling function. The situation recognition and goal activation function is responsible to map the knowledge base and current goals to a new set of goals. The planning and scheduling function is responsible for selecting which plans to execute, based on the current plans, goals and knowledge base of that layer.

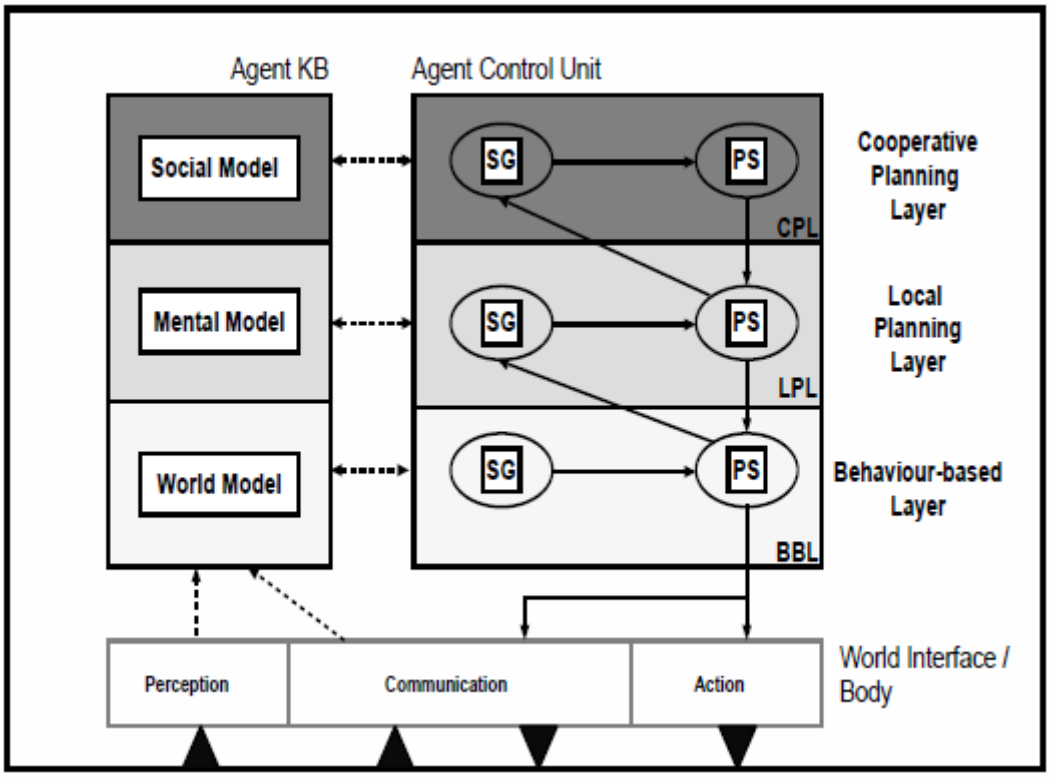

Figure 9: InteRRap Architecture

The main advantage of vertical layered architecture is the interaction between layers is reduced significantly to $m^2(n-1)$. The main disadvantage is that the architecture depends on its robustness, so if one layer fails, the entire system fails.

Cognitive Architecture

As stated in (Langley, 2004), there are several ways of constructing intelligent agents which are the software engineering approach, the multi-agent approach and the cognitive architecture approach. The cognitive architecture is based on the stream of cognitive science. The study of cognitive science focuses on the human cognition and psychology. The origin of cognitive architecture started with a specific class of architecture known as production systems (Neches *et al.*, 1987; Newell, 1973) and evolves over time. ACT is one of the earliest cognitive architecture build for modeling human behavior (Anderson, 1976). The construction of intelligent agent using cognitive architecture focuses on the cognition part, the simulation and modeling of human behaviour. The cognitive architecture approach is said to be different from multi-agent approach in the following manners (Langley *et al.*, 2009):

- cognitive architecture comes with a programming formalism to encode knowledge and associates it with its interpreter,
- it has strong assumptions on the representation of knowledge and the processes that operate on them,

- it assumes a modular representation of knowledge,
- it offers intelligent behavior at the systems level, rather than at the level of component methods designed for specialized tasks, and
- it provides a unified approach in which a common set of representations and mechanisms reduces the need for such careful crafting.

Cognitive architectures are used to construct intelligent system/agent that models human performance (Newell, 1990; Meyer & Kieras, 1997). Thisunderlying cognitive architecture should typically possess the characteristics of (Langley et al., 2009):

- short and long-term memories for storing the agent's beliefs, goals, and knowledge,
- representation of memories and their organization, and
- functional processes that operate on these structures.

Memory and learning are the two important key components in developing cognitive architecture. The combination of these two components form a taxonomy of three main categories of cognitive agent architecture namely symbolic, emergent, and hybrid models (Duch *et al.*, 2008) (see Figure 10). Symbolic architecture is a classical AI top-down analytical approach that focuses on information processing by using high-level symbols or declarative knowledge. Examples of cognitive symbolic architecture are SOAR (Laird *et al.*, 1987; Newell, 1990), EPIC (Meyer & Kieras, 1997), ICARUS (Langley, 2005). SOAR is a platform created to demonstrate general intelligence behaviour. The platform is based on symbolic representation which utilizes operator associated to problem space in production rules. This approach is referred as procedural long term knowledge. The platform has been used to develop many of the applications including expert system, intelligent control, and human behaviour interaction simulation. SOAR platform can also be viewed as a theory of general intelligence or as theory of human cognition. The architecture is based on human problem solving and unified theory of cognition. The theory of the architecture is that the unification of different theories or overlap theories without conflict can produce general intelligence behaviour with appropriate learning mechanism.

A bottom-up approaches which differentiates it from the symbolic architecture is adopted in emergent architecture whereby low-level activation signals flowing through a network consisting of numerous processing units is used. IBCA (O'Reilly *et al.*, 1999), Cortonis (Hecht-Nielsen, 2007), NuPIC (Hawkins & Blakeslee, 2004) are a few of the systems under the category of emergent architecture. IBCA is based on the working memory as controlled processing on how the components in brain dynamically interact with each other to bring about the cognition function. The emergent property of the dynamic interactions of the components contributes the working model of IBCA. There are several important components in IBCA and each has own specific function namely the posterior perceptual and motor cortex (PMC) which are responsible for the sensor and motor processing based on inference and generalization. The prefrontal cortex (PFC) is responsible for

dynamic and active memory and hippocampus (HCMP) is in charge of the rapid learning of arbitrary information.

Hybrid architecture is the combination of the symbolic and emergent paradigms. Examples in this category of hybrid architectures are ACT-R (Anderson & Lebiere, 2003), CLARION (Sun & Alexandre, 1997; Sun *et al.*, 2001) and LIOA (Franklin, 2006). ACT-R is one of the earliest developed cognitive architecture which is based on the modeling of human behaviour. ACT-R is organized into of a set of modules namely, sensory module, motor module, intentional module and declarative module. Each module has a temporary storage which holds the short term memory. A long-term memory is achieved through the combination of these modules. The action selection is based on the utility calculation, in which the highest utility actions are selected.Figure 10 shows the taxonomy of cognitive architecture based on the two important design properties which are memory and learning.

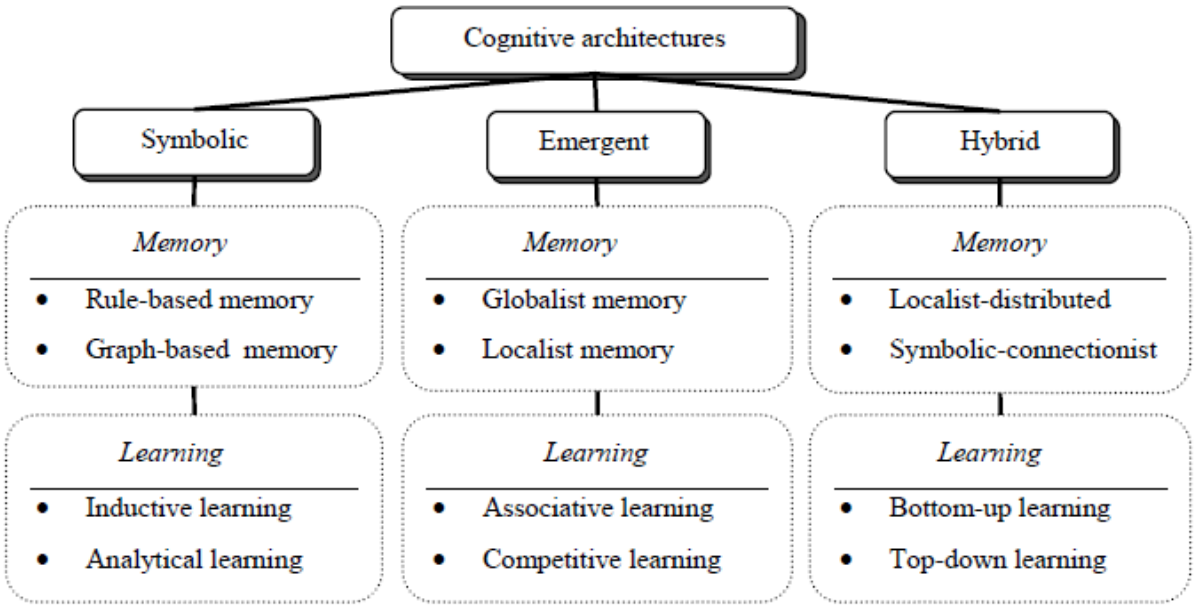

Figure 10: Taxonomy of Cognitive Architectures (Duch *et al.*, 2008)

Hybrid architecture is the combination of the symbolic and emergent paradigms. Examples in this category of hybrid architectures are ACT-R (Anderson & Lebiere, 2003), CLARION (Sun & Alexandre, 1997; Sun *et al.*, 2001) and LIOA (Franklin, 2006). ACT-R is one of the earliest developed cognitive architecture which is based on the modeling of human behaviour. ACT-R is organized into of a set of modules namely, sensory module, motor module, intentional module and declarative module. Each module has a temporary storage which holds the short term memory. A long-term memory is achieved through the combination of these modules. The action selection is based on the utility calculation, in which the highest utility actions are selected. Figure 10 shows the taxonomy of cognitive architecture based on the two important design properties which are memory and learning.

Conclusion

Agent architecture is the key component to constructing agent. It acts as the brain and heart of the agent to reason and perform action based on its knowledge base. This paper provides the state of the art of agent architecture. It outlined four main agent architectures: logic, reactive, BDI and layered architecture. Apart from these four, the cognitive architecture which is based on the human behaviour is another architecture that is widely adopted in building multi-agent system. Many of the foundations of cognitive architecture rest on the theories in cognitive sciences domain. The adoption semantic web technology to enhance the cognitive architecture in analyzing human behaviour will be a very interesting area to explore. Finally, the semantic technology adoption into agent architecture is also another interesting topic that should be studied as well. For future work, we would like to draw on these preliminary reviews to study the potential of a new agent architecture that uses semantic web technology to enable agent to process the information in a more meaningful way that will in turn improve its decision making which closely mimics the human reasoning process.

Acknowledgements

This project is funded by the Ministry of Higher Education, Malaysia under FRG0303-TK-1-2012.

References

Agre, P. & Chapman, D. 1987. PENGI: An implementation of a theory of activity. In Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87), 268-272, Seattle, WA.

Ambros-Ingerson, J. & Steel, S. 1988. Integrating planning, execution and monitoring. In: Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88), 83-88, St. Paul, MN.

Amir, E. & Maynard-Reid, P. 2000. Logic-based subsumption architecture: Empirical evaluation, in Proceedings of the AAAI Fall Symposium on Parallel Architectures for Cognition.

Amir, E. & Maynard-Reid, P. 2004. Logic-based subsumption architecture. *Artificial Intelligence*, 153:167-237.

Anderson, J. 1976. Language, Memory and Thought. Hillsdale, NJ: Erlbaum Associates.

Anderson, J. R. & Lebiere, C. 2003. The Newell test for a theory of cognition. *Behavioral and Brain Science* 26: 587-637.

Berges, I., Bermúdez, J., Goñi, A. & lllarramendi, A. 2008. Semantic Web Technology for Agent Communication Protocols, The Semantic Web: Research And Applications Lecture Notes in Computer Science, Volume 5021/2008:5-18.

Berners-Lee, T. 1999. Weaving the Web. Orion Business, London.

Berners-Lee, T., Hendler, J. & Lassila, O. 2001. The Semantic Web. Scientific American, May 2001, 28-37.

Bratman, M. E. 1987. Intentions, Plans, and Practical Reason. Harvard University Press: Cambridge, MA.

Bratman, M. E., Israel, D. J. & Pollack, M. E. 1988. Plans and resource-bounded practical reasoning. *Computational Intelligence* 4:349-355.

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

Brooks, R. A. 1991a. Intelligence without reason. In Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91), 569-595, Sydney, Australia.

Brooks, R. A. 1991b. Intelligence without representation. *Artificial Intelligence*, 47:139-159.

Chapman, D. & Agre, P. 1986. Abstract reasoning as emergent from concrete activity. In Georgeff, M. P. and Lansky, A. L., editors, Proceedings of the Reasoning About Actions &Plans, 411-424. Morgan Kaufmann Publishers: San Mateo, CA.

Cohen, P. R., Greenberg, M. L., Hart, D. M. & Howe, A. E. 1989. Trial by fire: Understanding the design requirements for agents in complex environments. *AI Magazine* 10(3): 32-48.

Comuzzi, M., Kritikos, K., & Plebani, P. 2009. A Semantic Based Framework for Supporting Negotiation in Service Oriented Architectures. IEEE Conference on Commerce and Enterprise Computing, 2009. CEC '09, 137-145, 20-23 July.

d'Inverno, M., Kinny, D., Luck, M. & Wooldridge, M. 1998. A Formal Specification of dMARS. In Singh, M.P., Rao, A.S. and Wooldridge, M. (eds), Intelligent Agents IV: Proceedings of the Fourth International Workshop on Agent Theories, Architectures, and Languages, 155–176, Springer.

Doran, J. E., Franklin, S., Jennings, N. R. & Norman, T. J. 1997. On Cooperation in Multi-Agent Systems. *The Knowledge Engineering Review*, 12(3): 309-314.

Duch, W., Oentaryo, R. J. & Pasquier, M. 2008. Cognitive Architectures: Where do we go from here? In Proceedings of the 2008 conference on Artificial General Intelligence, Pei Wang, Ben Goertzel, and Stan Franklin (Eds.). 122-136. IOS Press, Amsterdam, Netherland.

Durfee, E. H. & Rosenschein, J. S. 1994. Distributed problem solving and multiagent systems: Comparisons and examples. In Klein, M. editor, Proceedings of the 13th International Workshop on DAI, 94-104, LakeQuinalt, WA.

Erdur, R. C. & Seylan, I. 2008. The design of a semantic web compatible content language for agent communication. *Expert Systems* 25(3): 268-294.

Etzioni, 0. Lesh, N. & Segal, R. 1994. Building softbots for UNIX. In: Etzioni, 0. (ed.) Software Agents-Papers from the 1994 Spring Symposium (Technical Report SS-94-03), 9-16, AAAI Press.

Ferguson, I. A. 1992. TouringMachines: An Architecture for Dynamic, Rational, Mobile Agents. PhD thesis, Clare Hall, University of Cambridge, UK.

Fikes, R. E. & Nilsson, N. 1971. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 5(2): 189-208.

Franklin, S. 2006. The LIDA architecture: Adding new modes of learning to an intelligent, autonomous, software agent. In Proceeding of the International Conference on Integrated Design and Process Technology. San Diego, CA.

Genesereth, M. R. & Ketchpel, S. P. 1994. Software Agents. Communications of the ACM, Vol. 37, No. 7: 48-53.

Georgeff, M. P. & Lansky, A. L. 1986. Reasoning About Actions & Plans. Proceedings of the 1986 Workshop. Morgan Kaufmann Publishers: San Mateo, CA.

Hawkins, J. & Blakeslee, S. 2004. On intelligence: How a New Understanding of the Brain will Lead to the Creation of Truly Intelligent Machines. Times Books.

Hecht-Nielsen, R. 2007. Confabulation Theory: The Mechanism of Thought. Springer.

Howden, N., Ronnquist, R., Hodgson, A. & Lucas, A. 2001. JACK Intelligent Agents – Summary of an Agent Infrastructure. In Proceedings of the 5th International Conference on Autonomous Agents.

Huber, M. 1999. JAM: A BDI-Theoretic Mobile Agent Architecture. In Proceedings of the 3rd International Conference on Autonomous Agents, 236–243, New York, NY.

Jennings, N. R. 1993. Specification and implementation of a belief desire joint-intention architecture for collaborative problem solving. *Journal of Intelligent and Cooperative Information Systems*, 2(3):289-318.

Kaelbling, L. P. 1991. A situated automata approach to the design of embedded agents. *SIGART Bulletin*, 2(4):85-88.

Kaelbling, L. P. & Rosenschein, S. J. 1990. Action and planning in embedded agents. In Maes, P., editor, Designing Autonomous Agents, 35-48. The MIT Press: Cambridge, MA.

Laird, J. E., Rosenbloom, P. S., & Newell, A. 1987. Soar: An architecture for general intelligence. *Artificial Intelligence* 33: 1-64.

Langley, P. 2004. An cognitive architectures and the construction of intelligent agents. In Proc. Workshop on Intelligent Agent Architectures, 82. Stanford, CA.

Langley, P. 2005. An adaptive architecture for physical agents. In Proc. of the 2005 IEEE/WIC/ACM Int. Conf. on Intelligent Agent Technology. Compiegne, France: IEEE Computer Society Press, 18-25.

Langley, P., Laird, J. E. & Rogers, S. 2009. Cognitive architectures: Research issues and challenges. *Cognitive Systems Research* 10(2): 141-160.

Leyton-Brown, K. E. 2003. Resource Allocation in Competitive Multiagent Systems. Ph.D. Dissertation. Stanford University, Stanford, CA, USA.

Neches, R., Langley, P., & Klahr, D. 1987. Learning, development, and production systems. In D. Klahr, P. Langley, & R. Neches (Eds.), Production system models of learning and development. Cambridge, MA: MIT Press.

Newell, A. 1973. Production systems: Models of control structures. In W. G. Chase (Ed.), Visual information processing. New York: Academic Press.

Newell, A. 1990. Unified Theories of Cognition: Harvard University Press.

Newell, A. & Simon, H. A. 1976. Computer science as empirical enquiry. *Communications of the ACM* 19: 113-126.

Maes, P. 1989. The dynamics of action selection. In Proceedings of the Eleventh International Joint Conference on Artificial Intelligence (IJCAI-89), 991-997, Detroit, MI.

Maes, P. 1990. Situated agents can have goals. In Maes, P., editor, Designing Autonomous Agents, pages 49-70. The MIT Press: Cambridge, MA.

Maes, P. 1991. The agent network architecture (ANA). *SIGART Bulletin*, 2(4):115-120.

Maes, P. 1997. On Sofware Agents: Humanizing The Global Computer," Internet Computing, IEEE, vol.1, no.4: 10-19.

Meyer, D. E. & Kieras, D. E. 1997. A computational theory of executive cognitive processes and multiple-task performance: Part 1. Basic mechanisms. *Psychological Review*, 104(1):3-65.

Muller, J. P. & Pischel, M. 1993. The Agent Architecture InteRRaP: Concept and Application, DFKI Saarbrucken.

Munir, M., Mathieu, V., Thomas, M. & Stefan, B. 2011. A Layered Manufacturing System Architecture Supported with Semantic Agent Capabilities. In Atilla, E., Mamadou, K. & Mehmet, O. Semantic Agent Systems: Studies in Computational Intelligence., 215-242, Springer Berlin pp

O'Reilly, R. C., Braver, T. S. & Cohen J. D. 1999. A biologically-based computational model of working memory. In A. Miyake & P. Shah (Eds.), Models of Working Memory. 375-411, Cambridge University Press

Pozna, C., Precup, R. E., Kovacs, J. & Foldesi, P. 2011. Cooperation in multiagent systems. 9th International Symposium on Intelligent Systems and Informatics (SISY). IEEE. 195-200. Sept.

Rao, A. S. & Georgeff, M. P. 1991. Modeling rational agents within a BDI-architecture. In Fikes, R. and Sandewall, E., editors, Proceedings of Knowledge Representation and Reasoning (KR&R-91), 473-484. Morgan Kaufmann Publishers: San Mateo, CA.

Rosenschein, S. 1985. Formal theories of knowledge in AI and robotics. *New Generation Computing*, 345-357.

Rosenschein, S. & Kaelbling, L. P. 1986. The synthesis of digital machines with provable epistemic properties. In Halpern, J. Y., editor, Proceedings of the 1986 Conference on Theoretical Aspects of Reasoning About Knowledge, 83-98. Morgan Kaufmann Publishers: San Mateo, CA.

Russell, S. & Norvig, P. 1995. Artificial Intelligence: a Modem Approach. Prentice-Hall, Englewood Cliffs, NJ.

Sadaf, A., Amal, T., Amna, B. & Sergio, C. 2009. Semantic Agent Oriented Architecture for Researcher Profiling and Association (SemoRA). IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technologies. vol.3, 559-562, 15-18 Sept.

Shardlow, N, 1990. Action and agency in cognitive science. Master's thesis. Department of Psychology, University of Manchester, Oxford Road, Manchester MI3 9PL, UK.

Shoham, Y. & Leyton-Brown, K. 2008. Multiagent Systems - Algorithmic, Game-Theoretic, and Logical Foundations. Cambridge University Press.

Steels, L. 1990. Cooperation between distributed agents through self organization. In Demazeau, Y. and Müller, J.-P., editors, Decentralized AI - Proceedings of the First European Workshop on Modelling Autonomous Agents in Multi-Agent Worlds (MAAMAW-89), pages 175-196. Elsevier Science Publishers B.V.: Amsterdam, Netherlands.

Sun, R. & Alexandre, F. 1997. Connectionist symbolic integration. Hillsdale, NJ: Erlbaum.

Sun, R., Merrill, E. & Peterson, T. 2001. From implicit skills to explicit knowledge: A bottom-up model of skill learning. *Cognitive Science*, 25(2): 203-244.

Togelius, J. 2003. Evolution of the layers in a subsumption architecture robot controller. Dissertation for the Master of Science. University of Sussex

Vere, S. & Bickmore, T. 1990. A basic agent. *Computational Intelligence* 6:41-60.

WeiB, G. (ed.). 1999. Multi-Agent Systems. MIT Press, Cambridge, MA.

Wooldridge, M. 2001. Introduction to Multiagent Systems. 1st Edition. John Wiley & Sons, Inc., New York, NY, USA.

Wooldridge, M. 2009. An Introduction to Multiagent Systems. 2nd Edition. John Wiley & Sons, Inc., New York, NY, USA.

Wooldridge, M. & Jennings, N. R. 1995. Intelligent agents: theory and practice. *The Knowledge Engineering Review*, 10(2): 115-152

Wooldridge, M & Parsons, S. 1998. Intention Reconsideration Reconsidered. In Proceedings of the 5th International Workshop on Intelligent Agents V, Agent Theories, Architectures, and Languages (ATAL '98), Jörg P. Müller, Munindar P. Singh, and Anand S. Rao (Eds.). Springer-Verlag, London, UK, UK, 63-79.

Wood, S. 1993. Planning and Decision Making in Dynamic Domains. Ellis Horwood.

Yu, C.H., Werfel, J. & Nagpal, R. 2010. Collective decision-making in multi-agent systems by implicit leadership. In Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 3 - Volume 3 (AAMAS '10), Vol. 3. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 1189-1196.

Zhai, S. P. 2011. A semantic negotiation model for service property value between service agents. 3rd International Conference on Advanced Computer Control (ICACC), 552-556, 18-20 Jan.